An Invitation to 3-D Vision

From Images to Models

Yi Ma
Jana Košecká
Stefano Soatto
Shankar Sastry

November 19, 2001

This book is dedicated to people who love this book.

Preface

This book is intended to give students at the advanced undergraduate or introductory graduate level and researchers in computer vision, robotics, and computer graphics a self-contained introduction to the geometry of 3-D vision: that is, the reconstruction of 3-D models of objects from a collection of 2-D images. The only prerequisite for this book is a course in linear algebra at the undergraduate level.

As timely research summary, two bursts of manuscripts were published in the past on a geometric approach to computer vision: the ones that were published in mid 1990's on the geometry of two views (e.g., Faugeras 1993, Weng, Ahuja and Huang 1993, Maybank 1993), and the ones that were recently published on the geometry of multiple views (e.g., Hartley and Zisserman 2000, Faugeras, Luong and Papadopoulo 2001).[1] While a majority of those manuscripts were to summarize up to date research results by the time they were published, we sense that now the time is ripe for putting a coherent part of that material in a unified and yet simplified framework which can be used for pedegogical purposes. Although the approach we are to take here deviates from those old ones and the techniques we use are mainly linear algebra, this book nonetheless gives a comprehensive coverage of what is known todate on the geometry of 3-D vision. It

[1] To our knowledge, there are also two other books on computer vision currently in preparation: Ponce and Forsyth (expected in 2002), Pollefeys and van Gool (expected in 2002).

also builds on a homogeneous terminology a solid analytical foundation for future research in this young field.

This book is organized as follows. Following a brief introduction, Part I provides background materials for the rest of the book. Two fundamental transformations in multiple view geometry, namely the rigid-body motion and perspective projection, are introduced in Chapters 2 and 3 respectively. Image formation and feature extraction are discussed in Chapter 4. The two chapters in Part II cover the classic theory of two view geometry based on the so-called *epipolar constraint*. Theory and algorithms are developed for both discrete and continuous motions, and for both calibrated and uncalibrated camera models. Although the epipolar constraint has been very successful in the two view case, Part III shows that a more proper tool for studying the geometry of multiple views is the so-called *rank condition on the multiple view matrix*, which trivially implies all the constraints among multiple images that are known todate, in particular the epipolar constraint. The theory culminates in Chapter 10 with a unified theorem on a rank condition for arbitrarily mixed point, line and plane features. It captures *all* possible constraints among multiple images of these geometric primitives, and serves as a key to both geometric analysis and algorithmic development. Based on the theory and conceptual algorithms developed in early part of the book, Part IV develops practical reconstruction algorithms step by step, as well as discusses possible extensions of the theory covered in this book.

Different parts and chapters of this book have been taught as a one-semester course at the University of California at Berkeley, the University of Illinois at Urbana-Champaign, and the George Mason University, and as a two-quater course at the University of California at Los Angles. There is apparantly adequate material for lectures of one and a half semester or two quaters. Advanced topics suggested in Part IV or chosen by the instructor can be added to the second half of the second semester if a two-semester course is offered. Given below are some suggestions for course development based on this book:

1. *A one-semester course:* Appendix A and Chapters 1 - 7 and part of Chapters 8 and 13.
2. *A two-quater course:* Chapters 1 - 6 for the first quater and Chapters 7 - 10 and 13 for the second quater.
3. *A two-semester course:* Appendix A and Chapters 1 - 7 for the first semester; Chapters 8 - 10 and the instructor's choice of some advanced topics from chapters in Part IV for the second semester.

Exercises are provided at the end of each chapter. They consist of three types: 1. drill exercises that help student understand the theory covered in each chapter; 2. programming exercises that help student grasp the algorithms developed in each chapter; 3. exercises that guide student to

creatively develop a solution to a specialized case that is related to but not necessarily covered by the general theorems in the book. Solutions to most of the exercises will be made available upon the publication of this book. Software for examples and algorithms in this book will also be made available at a designated website.

Yi Ma
Stefano Soatto
Jana Košecká
Shankar Sastry

Café Nefeli, Berkeley
July 1, 2001

Contents

Chapter 1
Introduction

1.1 Visual perception: from 2-D images to 3-D models

The sense of vision plays an important role in the life of primates: it allows them to infer spatial properties of the environment that are necessary to perform crucial tasks towards survival. Even relatively simple organisms are capable of remote, non-contact sensing of the space surrounding them. Sensing information is then processed in the brain to infer a representation of the world; this can be used to their advantage in order to move within the environment, detect, recognize and fetch a prey etc.

In broad terms, a visual system is a collection of devices that transform measurements of light intensity into estimates of geometric properties of the environment. For instance, the human visual system measures the intensity of light incident the retinas and processes the measurements in the brain to produce an estimate of the three-dimensional layout of the scene that is sufficient, for instance, to reach for an object in the scene and grasp it. Only in recent years have artificial vision systems been developed that process light intensity measured by a camera and produce estimates of the three-dimensional layout of the scene being viewed. Artificial vision systems offer the potential of relieving humans from performing spatial control tasks that are dangerous, monotonous or boring such as driving a vehicle, surveying an underwater platform or detecting intruders in a building. In addition, they can be used to augment human capabilities or replace lost skills.

A single image of a scene is already a rich source of information on the three-dimensional structure, combining different cues such as texture, shading, T-junctions, cast shadows etc. (see figure 1.1). An image, however, is

Figure 1.1. *Some "pictorial" cues for three-dimensional structure: texture (left), shading and T-junctions (right).*

intrinsically ambiguous since one dimension is lost in the projection from the three-dimensional world onto the two-dimensional imaging surface. Inferring 3-D structure from 2-D images depends upon underlying assumptions about the scene. For instance, in exploiting shading cues one assumes that the visible surfaces have (locally) homogeneous reflective properties, so that the different appearance in different locations on the image is a function of the geometric properties at the corresponding location in space. This is one of the dominant cues in the perception of the shading of a marble statue, or of a hill covered with snow (see Figure **??**). Similarly, in the use of cast-shadows one assumes that objects have homogeneous radiance, so that the shadow can be related to a three-dimensional spatial configuration (Figure **??**). In the use of texture, one assumes that the scene presents some statistically homogeneous patterns, such as in a tiled floor or a flower bed (Figure **??**). The problem of dividing the image into regions where some of the properties mentioned above are homogeneous is called "segmentation".

Since all these cues are present in one single picture, they are usually called "pictorial". For each cue it is possible to construct examples where the underlying assumption is violated, so that we have different three-dimensional scenes that produce the same image, thereby giving rise to an optical "illusion". In fact, pictures are the most common instance of an illusion: they are objects with a particular three-dimensional shape (usually a plane) whose image on our retina is *almost* equal to the image of the three-dimensional scene they portrait. We say "almost" because the human visual system exploits other cues, that usually complement pictorial cues and disambiguate between different spatial interpretations of the same image. This has been known to Leonardo da Vinci, who noticed that the image of a scene on the left eye is different from the image on the right eye. Their "difference", known as Da Vinci stereopsis, can be used to infer the three-dimensional structure of the scene.

Artificial images that contain only the stereo cue can be constructed using so-called "random-dot stereograms", invented by Bela Julesz in 1960 (see Figure **??**). However, even stereo relies on an assumption: that points put into *correspondence* between the left and the right eye represent the same spatial location. When this assumption is violated we have, again, illusions. One example is given by those patterned pictures called "autostereograms" where, by relaxing the eyes as to look at infinity, one sees a three-dimensional object emerging out of the flat picture. Stereo is ex-

Figure 1.2. *Stereo as a cue in "random dot stereogram". When the image is fused binocularly, it reveals a "depth structure". Note that stereo is the only cue, as all pictorial aspects are absent in the random dot stereograms.*

ploited by the human visual system to infer the "depth" structure of the scene in the close-range. More generally, if we consider a stream of images taken from a moving viewpoint, the two-dimensional image-motion[1] can be exploited to infer information about the three-dimensional structure of the scene as well as its motion relative to the viewer.

That image-motion is a strong cue is easily seen by eliminating all pictorial cues until the scene reduces to a cloud of points. A still image looks like a random collection of dots but, as soon as it starts moving, we are able to perceive the three-dimensional shape and motion of the scene (see figure 1.3). The use of motion as a cue relies upon the assumption that we are

Figure 1.3. *2-D image-motion is a cue to 3-D scene structure: a number of dots are painted on the surface of a transparent cylinder. An image of the cylinder, which is generated by projecting it onto a wall, looks like a random collection of points. If we start rotating the cylinder, just by looking at its projection we can clearly perceive the existence of a three-dimensional structure underlying the two-dimensional motion of the projection.*

able to assess which point corresponds to which across time (the correspondence problem, again). This implies that, while the scene and/or the sensor moves, certain properties of the scene remain constant. For instance, we may assume that neither the reflective properties nor the distance between any two points on the scene change.

In this book we will concentrate on the motion/stereo problem, that is the reconstruction of three-dimensional properties of a scene from collections of two-dimensional images taken from different vantage points.

The reason for this choice does not imply that pictorial cues are not important: the fact that we use and like photographs so much suggests the contrary. However, the problem of stereo/motion reconstruction has now reached the point where it can be formulated in a precise mathematical sense, and effective software packages are available for solving it.

1.2 A historical perspective

`give a succinct history of vision`

[1] We use the improper diction "image-motion" or "moving image" to describe the time-change of the image due to a relative motion between the scene and the viewer.

1.3 A mathematical approach

As mentioned above, computer vision is about the study of recovering three-dimensional information (e.g., shape or layout) of a scene from its two-dimensional image(s). It is this nature of computer vision that claims itself as the inverse discipline of *computer graphics*, where the typical problem is to generate realistic two-dimensional images for a three-dimensional scene. Due to its inverse problem nature, many problems that arise in computer vision are prone to be ill-conditioned, or sometimes impossible to solve.

The motion/stereo problem, however, is one of the problems in computer vision that can be well posed. With reasonable assumptions on lighting condition, surface reflective property, and geometry (or shape) of objects, it is possible to fully capture information of a three-dimensional scene by a collection of two-dimensional images (typically taken from different vantage points). These assumptions usually include:

1. *Lighting condition* remains constant while all the images are taken for a scene;
2. *Surface reflectance* of objects in the scene is *Lambertian*, i.e. color and brightness intensity of a point on a surface does not change with the vantage point;
3. *Objects* in the scene consist of only geometric primitives: vertex, edge, plane, and some simple curves or surfaces.

It also makes sense to require that there is always sufficient variation of color and reflectance among these primitives such that it is possible to distinct them from one or another - a scene of purely white objects against a white background does not reveal itself well from any view. Under these assumptions, at least in principle, we may expect that a full reconstruction of a scene can be obtained from its multiple images. The rest of this book is going to show how.

While assumptions listed above indeed make the motion/stereo problem tractable to some extent, they do not imply that a constructive solution to a full reconstruction would then be easy. In order to find such a solution, it is imperative to understand first the geometric relationships between a three-dimensional scene and its multiple two-dimensional images. Mathematical problems raised in this context can be roughly thought as result of the interplay of two fundamental and important transformations: 1. the Euclidean motion which models the motion of camera; 2. the perspective projection which models the formation of image. Sometimes, for technical or practical reasons, the Euclidean motion can also be replaced by other types of motions, such as affine or projective transformations.[2] Euclidean

[2]For example, if the camera is not calibrated, it is usually easier to recover the camera motion up to a projective transformation.

motion, also known as rigid body motion, has been a classic subject in geometry, physics (especially mechanical physics), and robotics. Perspective projection, with its root traced back to Renaissance art, has been a widely studied subject in projective geometry and computer graphics. Important in their own right, it is however the study of computer vision and computer graphics that has brought together these two separate subjects, and therefore generated an intriguing, challenging, and yet beautiful new subject: *multiple view geometry*.

1.4 Organization of the book

```
let us leave this last.
```

Part I

Introductory material

Chapter 2

Representation of a three dimensional moving scene

The study of the geometric relationship between a three-dimensional scene and its images taken from a moving camera boils down to the interplay between two fundamental transformations: the rigid body motion that models how the camera moves, and the perspective projection which describes the image formation process. Long before these two transformations were brought together, the theories for each had been developed independently.

The study of the principles of motion of a material body has a long history belonging to the foundations of physics. For our purpose, the more recent noteworthy insights to the understanding of the motion of rigid bodies came from Chasles and Poinsot in the early 1800s. Their findings led to current treatment of the subject which has since been widely adopted.

We start this chapter with an introduction to the three dimensional Euclidean space as well as to rigid body transformations. The next chapter will then focus on the perspective projection model of the camera.

2.1 Three-dimensional Euclidean space

We will use $\mathbb{E}^3$ to denote the familiar three-dimensional Euclidean space. In general, a Euclidean space is a set whose elements satisfy the five axioms of Euclid []. More practically, the three-dimensional Euclidean space can be represented by a (global) Cartesian coordinate frame: every point $p \in \mathbb{E}^3$ can be identified with a point in $\mathbb{R}^3$ by three coordinates: $\mathbf{X} \doteq [X_1, X_2, X_3]^T$. Sometime we will use $[X, Y, Z]^T$ to indicate individ-

ual coordinates instead of $[X_1, X_2, X_3]^T$. Through such an assignment of Cartesian frame, one establishes a one-to-one correspondence between $\mathbb{E}^3$ and $\mathbb{R}^3$, which allows us to safely talk about points and their coordinates as if they were the same thing.

Cartesian coordinates are the first step towards being able to measure distances and angles. In order to do so, $\mathbb{E}^3$ must be endowed with a *metric*. A precise definition of metric relies on the notion of *vector*. In the Euclidean space, a *vector* is identified by a pair of points $p, q \in \mathbb{E}^3$; that is, a vector v is defined as a directed arrow connecting p to q. The point p is usually called the base point of v. In coordinates, the vector v is represented by the triplet $[v_1, v_2, v_3]^T \in \mathbb{R}^3$, where each coordinate is the difference between the corresponding coordinates of the two points: if p has coordinates $\mathbf{X}$ and q has coordinates $\mathbf{Y}$, then v has coordinates[1]

$$v = \mathbf{Y} - \mathbf{X} \in \mathbb{R}^3.$$

One can also introduce the concept of *free vector*, a vector whose definition does not depend on its base point. If we have two pairs of points (p, q) and (p', q') with coordinates satisfying $\mathbf{Y} - \mathbf{X} = \mathbf{Y}' - \mathbf{X}'$, we say that they define the same free vector. Intuitively, this allows a vector v to be transported in parallel anywhere in $\mathbb{E}^3$. In particular, without loss of generality, one can assume that the base point is the origin of the Cartesian frame, so that $\mathbf{X} = 0$ and $v = \mathbf{Y}$. Note, however, that this notation is confusing: $\mathbf{Y}$ here denotes the coordinates of a vector, that happen to be the same as the coordinates of the point q just because we have chosen the point p to be the origin. Nevertheless, the reader should keep in mind that points and vectors are different geometric objects; for instance, as we will see shortly, a rigid body motion acts differently on points and vectors. So, keep the difference in mind!

The set of all (free) vectors form a *linear (vector) space*[2], where a linear combination of two vectors $v, u \in \mathbb{R}^3$ is given by:

$$\alpha v + \beta u = (\alpha v_1 + \beta u_1, \alpha v_2 + \beta u_2, \alpha v_3 + \beta u_3)^T \in \mathbb{R}^3, \quad \forall \alpha, \beta \in \mathbb{R}.$$

The *Euclidean metric* for $\mathbb{E}^3$ is then defined simply by an inner product on its vector space:

Definition 2.1 (Euclidean metric and inner product). *A bilinear function $\langle \cdot, \cdot \rangle : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ is an* inner product *if it is linear, symmetric and positive definite. That is, $\forall u, v, w \in \mathbb{R}^3$*

1. $\langle u, \alpha v + \beta w \rangle = \alpha \langle u, v \rangle + \beta \langle u, w \rangle, \quad \forall\ \alpha, \beta \in \mathbb{R},$
2. $\langle u, v \rangle = \langle v, u \rangle.$
3. $\langle v, v \rangle \geq 0$ *and* $\langle v, v \rangle = 0 \Leftrightarrow v = 0,$

[1] Note that we use the same symbol v for a vector and its coordinates.

[2] Note that points do not.

The quantity $\|v\| = \sqrt{\langle v, v\rangle}$ is called the *Euclidean norm* (or 2-norm) of the vector v. It can be shown that, by a proper choice of the Cartesian frame, any inner product in $\mathbb{E}^3$ can be converted to the following familiar form:

$$\langle u, v\rangle \;=\; u^T v = u_1 v_1 + u_2 v_2 + u_3 v_3. \tag{2.1}$$

In most of this book (but not everywhere!) we will use the canonical inner product $\langle u, v\rangle \;=\; u^T v$ and, consequently, $\|v\| = \sqrt{v_1^2 + v_2^2 + v_3^2}$. When the inner product between two vectors is zero, $\langle u, v\rangle$, they are said to be *orthogonal*.

Finally, a Euclidean space $\mathbb{E}^3$ can then be formally described as a space which, with respect to a Cartesian frame, can be identified with $\mathbb{R}^3$ and has a metric (on its vector space) given by the above inner product. With such a metric, one can measure not only distances between points or angles between vectors, but also calculate the length of a curve, or the volume of a region[3]

While the inner product of two vectors returns a scalar, the so-called *cross product* returns a vector instead.

Definition 2.2 (Cross product). *Given two vectors $u, v \in \mathbb{R}^3$, their* cross product *is a third vector with coordinates given by:*

$$u \times v = \begin{bmatrix} u_2 v_3 - u_3 v_2 \\ u_3 v_1 - u_1 v_3 \\ u_1 v_2 - u_2 v_1 \end{bmatrix} \in \mathbb{R}^3.$$

It is immediate from this definition that the cross product of two vectors is linear: $u \times (\alpha v + \beta w) = \alpha u \times v + \beta u \times w \;\forall\; \alpha, \beta \in \mathbb{R}$. Furthermore, it is immediate to verify that

$$\langle u \times v, u\rangle \;=\; \langle u \times v, v\rangle \;= 0, \quad u \times v = -v \times u.$$

Therefore, the cross product of two vector is orthogonal to each of its factors, and the order of the factors defines an *orientation*.

If we fix u, the cross product can be interpreted as a map $v \mapsto u \times v$ between $\mathbb{R}^3$ and $\mathbb{R}^3$. Due to the linearity property, this map is in fact linear and, therefore, like all linear maps between vector spaces, it can be represented by a matrix. We call such a matrix $\widehat{u} \in \mathbb{R}^{3\times 3}$. It is immediate

[3]For example, if the trajectory of a moving particle p in $\mathbb{E}^3$ is described by a curve $\gamma(\cdot) : t \mapsto \mathbf{X}(t) \in \mathbb{R}^3, t \in [0, 1]$, then the total length of the curve is given by:

$$l(\gamma(\cdot)) = \int_0^1 \|\dot{\mathbf{X}}(t)\|\, dt.$$

where $\dot{\mathbf{X}}(t) = \frac{d}{dt}(\mathbf{X}(t)) \in \mathbb{R}^3$ is the so-called tangent vector to the curve.

to verify by substitution that this matrix is given by

$$\widehat{u} = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}. \tag{2.2}$$

Hence, we can write $u \times v = \widehat{u}v$. Note that[4] $\widehat{u}$ is a 3×3 skew-symmetric matrix, i.e. $\widehat{u}^T = -\widehat{u}$. It is immediate to verify that for $e_1 \doteq [1, 0, 0]^T, e_2 \doteq [0, 1, 0]^T \in \mathbb{R}^3$, we have $e_1 \times e_2 = [0, 0, 1]^T \doteq e_3$. That is for, a standard Cartesian frame, the cross product of the principal axes X and Y gives the principal axis Z. The cross product therefore conforms with the *right-hand rule*.

The cross product allows us to define a map between a vector, u, and a skew-symmetric, 3×3 matrix $\widehat{u}$. Is the converse true? Can every 3×3 skew-symmetric matrix be associated with a three-dimensional vector u? The answer is yes, as it is easy to verify. Let $M \in \mathbb{R}^{3\times 3}$ be skew-symmetric, that is $M = -M^T$. By writing this equation in terms of the elements of the matrix, we conclude that $m_{11} = m_{22} = m_{23} = 0$ and $m_{ij} = -m_{ji},\ i, j = 1, 2, 3$. This shows that a skew-symmetric matrix has only three degrees of freedom, for instance m_{21}, m_{13}, m_{32}. If we call $u_1 = m_{32};\ u_2 = m_{13};\ u_3 = m_{21}$, then $\widehat{u} = M$. Indeed, the vector space $\mathbb{R}^3$ and the space of skew-symmetric 3×3 matrices $so(3)$ can be considered as the same thing, with the cross product that maps one onto the other, $\times : \mathbb{R} \to so(3);\ u \mapsto \widehat{u}$, and the inverse map, called "vee", that extracts the components of the vector u from the skew-symmetric matrix $\widehat{u}$: $\vee : so(3) \to \mathbb{R};\ M = -M^T \mapsto M^\vee = [m_{32}, m_{13}, m_{21}]^T$.

2.2 Rigid body motion

Consider an object moving in front of a camera. In order to describe its motion one should, in principle, specify the trajectory of every single point on the object, for instance by giving its coordinates as a function of time $\mathbf{X}(t)$. Fortunately, for rigid objects we do not need to specify the motion of every particle. As we will see shortly, it is sufficient to specify the motion of a point, and the motion of three coordinate axes attached to that point.

The condition that defines a rigid object is that the distance between any two points on it does not change over time as the object moves. So if $\mathbf{X}(t)$ and $\mathbf{Y}(t)$ are the coordinates of any two points p and q respectively, their distance between must satisfy:

$$\|\mathbf{X}(t) - \mathbf{Y}(t)\| = \text{constant}, \quad \forall t \in \mathbb{R}. \tag{2.3}$$

[4] In some computer vision literature, the matrix $\widehat{u}$ is also denoted as $u_\times$.

In other words, if v is the vector defined by the two points p and q, then the norm (or length) of v remains the same as the object moves: $\|v(t)\| =$ constant. A *rigid body motion* is then a family of transformations that describe how the coordinates of every point on the object change as a function of time. We denote it by g:

$$\begin{aligned} g(t): \mathbb{R}^3 &\rightarrow \mathbb{R}^3 \\ \mathbf{X} &\mapsto g(t)(\mathbf{X}) \end{aligned}$$

If, instead of looking at the entire continuous moving path of the object, we concentrate on the transformation between its initial and final configuration, this transformation is usually called a *rigid body displacement* and is denoted by a single mapping:

$$\begin{aligned} g: \mathbb{R}^3 &\rightarrow \mathbb{R}^3 \\ \mathbf{X} &\mapsto g(\mathbf{X}) \end{aligned}$$

Besides transforming the coordinates of points, g also induces a transformation on vectors. Suppose v is a vector defined by two points p and q: $v = \mathbf{Y} - \mathbf{X}$; then, after the transformation g, we obtain a new vector:

$$g_*(v) \doteq g(\mathbf{Y}) - g(\mathbf{X}).$$

That g preserves the distance between any two points can be expressed in terms of vectors as $\|g_*(v)\| = \|v\|$ for $\forall v \in \mathbb{R}^3$.

However, preserving distances between points is not the only requirement that a rigid object moving in space satisfies. In fact, there are transformations that preserve distances, and yet they are not physically realizable. For instance, the mapping

$$f : [X_1, X_2, X_3]^T \mapsto [X_1, X_2, -X_3]^T$$

preserves distance but not orientation. It corresponds to a reflection of points about the XY plane as a double-sided mirror. To rule out this type of mapping, we require that any rigid body motion, besides preserving distance, preserves orientation as well. That is, in addition to preserving the norm of vectors, it must preserve their cross product. The coordinate transformation induced by a rigid body motion is called a *special Euclidean transformation*. The word "special" indicates the fact that it is orientation-preserving.

Definition 2.3 (Rigid body motion or special Euclidean transformation). *A mapping* $g: \mathbb{R}^3 \rightarrow \mathbb{R}^3$ *is a* rigid body motion *or a* special Euclidean transformation *if it preserves the norm and the cross product of any two vectors:*

1. *Norm:* $\|g_*(v)\| = \|v\|$, $\forall v \in \mathbb{R}^3$.
2. *Cross product:* $g_*(u) \times g_*(v) = g_*(u \times v)$, $\forall u, v \in \mathbb{R}^3$.

In the above definition of rigid body motion, it is explicitly required that the distance between points be preserved. Then how about angles between vectors? Although it is not explicitly stated in the definition, angles are indeed preserved by any rigid body motion since the inner product $\langle \cdot, \cdot \rangle$ can be expressed in terms of the norm $\| \cdot \|$ by the *polarization identity*:

$$u^T v = \frac{1}{4}(\|u+v\|^2 - \|u-v\|^2). \tag{2.4}$$

Hence, for any rigid body motion g, one can show that:

$$u^T v = g_*(u)^T g_*(v), \quad \forall u, v \in \mathbb{R}^3. \tag{2.5}$$

In other words, a rigid body motion can also be defined as one that preserves both inner product and cross product.

How do these properties help us describe rigid motion concisely? The fact that distances and orientations are preserved in a rigid motion means that individual points cannot translate relative to each other. However, they can rotate relative to each other, but they have to collectively, in order to not alter any mutual distance between points. Therefore, a rigid body motion can be described by the motion of any one point on the body, and the rotation of a coordinate frame attached to that point. In order to do this, we represent the *configuration* of a rigid body by attaching a Cartesian coordinate frame to some point on the rigid body and keeping track of the motion of this coordinate frame relative to a fixed frame.

To see this, consider a "fixed" (world) coordinate frame, given by three *orthonormal* vectors $e_1, e_2, e_3 \in \mathbb{R}^3$; that is, they satisfy

$$e_i^T e_j = \delta_{ij} \begin{cases} \delta_{ij} = 1 & \text{for} \quad i = j \\ \delta_{ij} = 0 & \text{for} \quad i \neq j \end{cases}. \tag{2.6}$$

Typically, the vectors are ordered so as to form a right-handed frame: $e_1 \times e_2 = e_3$. Then, after a rigid body motion g, we have:

$$g_*(e_i)^T g_*(e_j) = \delta_{ij}, \quad g_*(e_1) \times g_*(e_2) = g_*(e_3). \tag{2.7}$$

That is, the resulting three vectors still form a right-handed orthonormal frame. Therefore, a rigid object can always be associated with a right-handed, orthonormal frame, and its rigid body motion can be entirely specified by the motion of such a frame, which we call the object frame. In in Figure 2.1 we show an object (in this case a camera) moving relative to a fixed "world" coordinate frame W. In order to specify the configuration of the camera relative to the world frame W, one may pick a fixed point o on the camera and attach to it an orthonormal frame, the camera coordinate frame C. When the camera moves, the camera frame also moves as if it were fixed to the camera. The configuration of the camera is then determined by (1) the vector between the origin of the world frame o and the camera frame, $g(o)$, called the "translational" part and denoted as T, and (2) the relative orientation of the camera frame C, with coordinate axes

$g_*(e_1), g_*(e_2), g_*(e_3)$, relative to the fixed world frame W with coordinate axes e_1, e_2, e_3,called the "rotational" part and denoted by R.

In the case of vision, there is no obvious choice of the origin o and the reference frame e_1, e_2, e_3. Therefore, we could choose the world frame to be attached to the camera and specify the translation and rotation of the scene (assuming it is rigid!), or we could attach the world frame to the scene and specify the motion of the camera. All that matters is the *relative* motion between the scene and the camera, and what choice of reference frame to make is, from the point of view of geometry[5], arbitrary.

Remark 2.1. *The set of rigid body motions, or special Euclidean transformations, is a (Lie) group, the so-called* special Euclidean group*, typically denoted as $SE(3)$. Algebraically, a* group *is a set G, with an operation of (binary) multiplication $\circ$ on elements of G which is:*

- closed*: If $g_1, g_2 \in G$ then also $g_1 \circ g_2 \in G$;*
- associative*: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$, for all $g_1, g_2, g_3 \in G$;*
- unit element *e: $e \circ g = g \circ e = g$, for all $g \in G$;*
- invertible: *For every element $g \in G$, there exists an element $g^{-1} \in G$ such that $g \circ g^{-1} = g^{-1} \circ g = e$.*

In the next few sections, we will focus on studying in detail how to represent *the special Euclidean group $SE(3)$. More specifically, we will introduce a way to realize elements in the special Euclidean group $SE(3)$ as elements in a group of $n \times n$ non-singular (real) matrices whose multiplication is simply the matrix multiplication. Such a group of matrices is usually called a general linear group and denoted as $GL(n)$ and such a representation is called a* matrix representation*. A representation is a map*

$$\begin{aligned} \mathcal{R}: SE(3) &\rightarrow GL(n) \\ g &\mapsto \mathcal{R}(g) \end{aligned}$$

which preserves the group structure of $SE(3)$.[6] *That is, the inverse of a rigid body motion and the composition of two rigid body motions are preserved by the map in the following way:*

$$\mathcal{R}(g^{-1}) = \mathcal{R}(g)^{-1}, \quad \mathcal{R}(g \circ h) = \mathcal{R}(g)\mathcal{R}(h), \quad \forall g, h \in SE(3). \tag{2.8}$$

We start with the rotational component of motion.

[5]The neuroscience literature debates on whether the primate brain maintains a view-centered or an object-centered representation of the world. From the point of view of geometry, the two are equivalent, for they only differ by an arbitrary change of coordinates.

[6]Such a map is called *group homeomorphism* in algebra.

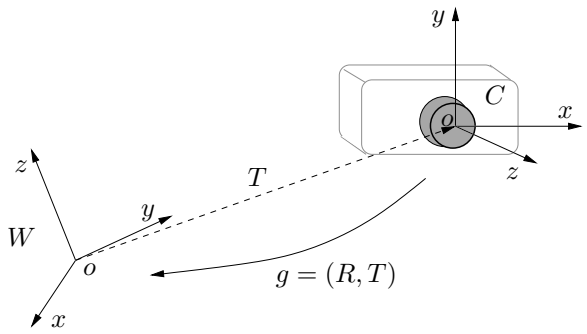

Figure 2.1. A rigid body motion which, in this instance, is between a camera and a world coordinate frame.

2.3 Rotational motion and its representations

Suppose we have a rigid object rotating about a fixed point $o \in \mathbb{E}^3$. How do we describe its orientation relative a chosen coordinate frame, say W? Without loss of generality, we may always assume that the origin of the world frame is the center of rotation o. If this is not the case, simply translate the origin to the point o. We now attach another coordinate frame, say C to the rotating object with origin also at o. The relation between these two coordinate frames is illustrated in Figure 2.2. Obviously, the configuration of

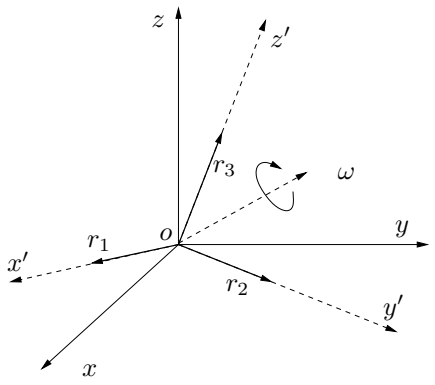

Figure 2.2. Rotation of a rigid body about a fixed point o. The solid coordinate frame W is fixed and the dashed coordinate frame C is attached to the rotating rigid body.

the object is determined by the orientation of the frame C. The orientation of the frame C relative to the frame W is determined by the coordinates of the three orthonormal vectors $r_1 = g_*(e_1), r_2 = g_*(e_2), r_3 = g_*(e_3) \in \mathbb{R}^3$ relative to the world frame W, as shown in Figure 2.2. The three vectors r_1, r_2, r_3 are simply the unit vectors along the three principal axes X', Y', Z'

of the frame C respectively. The configuration of the rotating object is then completely determined by the following 3×3 matrix:

$$R_{wc} = [r_1, r_2, r_3] \in \mathbb{R}^{3\times 3}$$

with r_1, r_2, r_3 stacked in order as its three columns. Since r_1, r_2, r_3 form an orthonormal frame, it follows that:

$$r_i^T r_j = \delta_{ij} \begin{cases} \delta_{ij} = 1 & \text{for} \quad i = j \\ \delta_{ij} = 0 & \text{for} \quad i \neq j \end{cases} \qquad \forall i, j \in \{1, 2, 3\}.$$

This can be written in matrix form as:

$$R_{wc}^T R_{wc} = R_{wc} R_{wc}^T = I.$$

Any matrix which satisfies the above identity is called an *orthogonal matrix*. It follows from the definition that the inverse of an ortohgonal matrix is simply its transpose: $R_{wc}^{-1} = R_{wc}^T$. Since r_1, r_2, r_3 form a right-handed frame, we further have that the determinant of R_{wc} must be positive 1. This can be easily seen when looking at the determinant of the rotation matrix:

$$detR = r_1^T (r_2 \times r_3)$$

which is equal to 1 for right-handed coordinate systems. Hence R_{wc} is a *special orthogonal matrix* where, as before, the word "special" indicates orientation preserving. The space of all such special orthogonal matrices in $\mathbb{R}^{3\times 3}$ is usually denoted by:

$$\boxed{SO(3) = \{R \in \mathbb{R}^{3\times 3} \mid R^T R = I, \det(R) = +1\}}.$$

Traditionally, 3×3 special orthogonal matrices are called *rotation matrices* for obvious reasons. It is straightforward to verify that $SO(3)$ has a group structure. That is, it satisfies all four axioms of a group mentioned in the previous section. We leave the proof to the reader as an exercise. Therefore, the space $SO(3)$ is also referred to as the *special orthogonal group* of $\mathbb{R}^3$, or simply the *rotation group*. Directly from the definition of the rotation matrix, we can show that rotation indeed preserves both the inner product and the cross product of vectors. We also leave this as an exercise to the reader.

Going back to Figure 2.2, every rotation matrix $R_{wc} \in SO(3)$ represents a possible configuration of the object rotated about the point o. Besides this, R_{wc} takes another role as the matrix that represents the actual coordinates transformation from the frame C to the frame W. To see this, suppose that, for a given a point $p \in \mathbb{E}^3$, its coordinates with respect to the frame W are $\mathbf{X}_w = [X_{1w}, X_{2w}, X_{3w}]^T \in \mathbb{R}^3$. Since r_1, r_2, r_3 form a basis for $\mathbb{R}^3$, $\mathbf{X}_w$ can also be expressed as a linear combination of these three vectors, say $\mathbf{X}_w = X_{1c} r_1 + X_{2c} r_2 + X_{3c} r_3$ with $[X_{1c}, X_{2c}, X_{3c}]^T \in \mathbb{R}^3$. Obviously, $\mathbf{X}_c = [X_{1c}, X_{2c}, X_{3c}]^T$ are the coordinates of the same point p with respect

to the frame C. Therefore, we have:

$$\mathbf{X}_w = X_{1c}r_1 + X_{2c}r_2 + X_{3c}r_3 = R_{wc}\mathbf{X}_c.$$

In this equation, the matrix R_{wc} transforms the coordinates $\mathbf{X}_c$ of a point p relative to the frame C to those $\mathbf{X}_w$ relative to the frame W. Since R_{wc} is a rotation matrix, its inverse is simply its transpose:

$$\mathbf{X}_c = R_{wc}^{-1}\mathbf{X}_w = R_{wc}^T\mathbf{X}_w.$$

That is, the inverse transformation is also a rotation; we call it R_{cw}, following an established convention, so that

$$R_{cw} = R_{wc}^{-1} = R_{wc}^T.$$

The configuration of the continuously rotating object can be then described as a trajectory $R(t) : t \mapsto SO(3)$ in the space $SO(3)$. For different times, the composition law of the rotation group then implies:

$$R(t_2, t_0) = R(t_2, t_1)R(t_1, t_0), \quad \forall t_0 < t_1 < t_2 \in \mathbb{R}.$$

Then, for a rotating camera, the world coordinates $\mathbf{X}_w$ of a fixed 3-D point p are transformed to its coordinates relative to the camera frame C by:

$$\mathbf{X}_c(t) = R_{cw}(t)\mathbf{X}_w.$$

Alternatively, if a point p fixed with respect to the camera frame with coordinates $\mathbf{X}_c$, its world coordinates $\mathbf{X}_w(t)$ as function of t are then given by:

$$\mathbf{X}_w(t) = R_{wc}(t)\mathbf{X}_c.$$

2.3.1 Canonical exponential coordinates

So far, we have shown that a rotational rigid body motion in $\mathbb{E}^3$ can be represented by a 3×3 rotation matrix $R \in SO(3)$. In the matrix representation that we have so far, each rotation matrix R is described and determined by its $3 \times 3 = 9$ entries. However, these 9 entries are not free parameters - they must satisfy the constraint $R^TR = I$. This actually imposes 6 independent constraints on the 9 entries. Hence the dimension of the rotation matrix space $SO(3)$ is only 3, and 6 parameters out of the 9 are in fact redundant. In this and the next section, we will introduce a few more representations (or parameterizations) for rotation matrix.

Given a curve $R(t) : \mathbb{R} \to SO(3)$ which describes a continuous rotational motion, the rotation must satisfy the following constraint:

$$R(t)R^T(t) = I.$$

Computing the derivative of the above equation with respect to time t, noticing that the right hand side is a constant matrix, we obtain:

$$\dot{R}(t)R^T(t) + R(t)\dot{R}^T(t) = 0 \quad \Rightarrow \quad \dot{R}(t)R^T(t) = -(\dot{R}(t)R^T(t))^T.$$

The resulting constraint which we obtain reflects the fact that the matrix $\dot{R}(t)R^T(t) \in \mathbb{R}^{3\times 3}$ is a skew symmetric matrix (see Appendix A). Then, as we have seen, there must exist a vector, say $\omega(t) \in \mathbb{R}^3$ such that:

$$\widehat{\omega}(t) = \dot{R}(t)R^T(t).$$

Multiplying both sides by $R(t)$ to the right yields:

$$\dot{R}(t) = \widehat{\omega}(t)R(t). \tag{2.9}$$

Notice that, from the above equation, if $R(t_0) = I$ for $t = t_0$, we have $\dot{R}(t_0) = \widehat{\omega}(t_0)$. Hence, around the identity matrix I, a skew symmetric matrix gives a first-order approximation of rotation matrix:

$$R(t_0 + dt) \approx I + \widehat{\omega}(t_0)\ dt.$$

The space of all skew symmetric matrices is denoted as:

$$\boxed{so(3) = \{\widehat{\omega} \in \mathbb{R}^{3\times 3} \mid \omega \in \mathbb{R}^3\}}$$

and thanks to the above observation it is also called the *tangent space* at the identity of the matrix group $SO(3)$[7]. If $R(t)$ is not at the identity, the tangent space at $R(t)$ is simply $so(3)$ transported to $R(t)$ by a multiplication of $R(t)$ to the right: $\dot{R}(t) = \widehat{\omega}(t)R(t)$. This also shows that elements of $SO(3)$ only depend upon three parameters.

Having understood its local approximation, we will now use this knowledge to obtain a representation for rotation matrices. Let us start by assuming that the matrix $\widehat{\omega}$ in (2.9) is constant:

$$\dot{R}(t) = \widehat{\omega}R(t). \tag{2.10}$$

In the above equation, $\widehat{\omega}$ can be interpreted as the *state transition matrix* for the following linear ordinary differential equation (ODE):

$$\dot{x}(t) = \widehat{\omega}x(t), \quad x(t) \in \mathbb{R}^3.$$

It is then immediate to verify that the solution to the above ODE is given by:

$$x(t) = e^{\widehat{\omega}t}x(0) \tag{2.11}$$

where $e^{\widehat{\omega}t}$ is the matrix exponential:

$$e^{\widehat{\omega}t} = I + \widehat{\omega}t + \frac{(\widehat{\omega}t)^2}{2!} + \cdots + \frac{(\widehat{\omega}t)^n}{n!} + \cdots. \tag{2.12}$$

$e^{\widehat{\omega}t}$ is also denoted as $\exp(\widehat{\omega}t)$. Due to the uniqueness of the solution for the ODE (2.11), and assuming $R(0) = I$ as initial condition, we must have:

$$\boxed{R(t) = e^{\widehat{\omega}t}} \tag{2.13}$$

[7]Since $SO(3)$ is a Lie group, $so(3)$ is also called its Lie algebra.

To confirm that the matrix $e^{\widehat{\omega}t}$ is indeed a rotation matrix, one can directly show from the definition of matrix exponential:

$$(e^{\widehat{\omega}t})^{-1} = e^{-\widehat{\omega}t} = e^{\widehat{\omega}^T t} = (e^{\widehat{\omega}t})^T.$$

Hence $(e^{\widehat{\omega}t})^T e^{\widehat{\omega}t} = I$. It remains to show that $\det(e^{\widehat{\omega}t}) = +1$ and we leave this fact to the reader as an exercise. A physical interpretation of the equation (2.13) is: if $\|\omega\| = 1$, then $R(t) = e^{\widehat{\omega}t}$ is simply a rotation around the axis $\omega \in \mathbb{R}^3$ by t radians. Therefore, the matrix exponential (2.12) indeed defines a map from the space $so(3)$ to $SO(3)$, the so-called *exponential map*:

$$\begin{aligned} \exp: so(3) &\rightarrow SO(3) \\ \widehat{\omega} \in so(3) &\mapsto e^{\widehat{\omega}} \in SO(3). \end{aligned}$$

Note that we obtained the expression (2.13) by assuming that the $\omega(t)$ in (2.9) is constant. This is however not always the case. So a question naturally arises here: Can every rotation matrix $R \in SO(3)$ be expressed in an exponential form as in (2.13)? The answer is yes and the fact is stated as the following theorem:

Theorem 2.1 (Surjectivity of the exponential map onto $SO(3)$). *For any $R \in SO(3)$, there exists a (not necessarily unique) $\omega \in \mathbb{R}^3$, $\|\omega\| = 1$ and $t \in \mathbb{R}$ such that $R = e^{\widehat{\omega}t}$.*

Proof. The proof of this theorem is by construction: if the rotation matrix R is given as:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

the corresponding t and ω are given by:

$$\boxed{t = \cos^{-1}\left(\frac{\text{trace}(R) - 1}{2}\right), \quad \omega = \frac{1}{2\sin(t)} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}}.$$

□

The significance of this theorem is that any rotation matrix can be realized by rotating around some fixed axis by a certain angle. However, the theorem only guarantees the surjectivity of the exponential map from $so(3)$ to $SO(3)$. Unfortunately, this map is not injective hence not one-to-one. This will become clear after we have introduced the so-called *Rodrigues' formula* for computing $R = e^{\widehat{\omega}t}$.

From the constructive proof for Theorem 2.1, we now know how to compute the exponential coordinates (ω, t) for a given rotation matrix $R \in SO(3)$. On the other hand, given (ω, t), how do we effectively compute the corresponding rotation matrix $R = e^{\widehat{\omega}t}$? One can certainly use the

series (2.12) from the definition. The following theorem however simplifies the computation dramatically:

Theorem 2.2 (Rodrigues' formula for rotation matrix). *Given $\omega \in \mathbb{R}^3$ with $\|\omega\| = 1$ and $t \in \mathbb{R}$, the matrix exponential $R = e^{\widehat{\omega}t}$ is given by the following formula:*

$$\boxed{e^{\widehat{\omega}t} = I + \widehat{\omega}\sin(t) + \widehat{\omega}^2(1 - \cos(t))} \tag{2.14}$$

Proof. It is direct to verify that powers of $\widehat{\omega}$ can be reduced by the following two formulae:

$$\begin{aligned} \widehat{\omega}^2 &= \omega\omega^T - I, \\ \widehat{\omega}^3 &= -\widehat{\omega}. \end{aligned}$$

Hence the exponential series (2.12) can be simplified as:

$$e^{\widehat{\omega}t} = I + \left(t - \frac{t^3}{3!} + \frac{t^5}{5!} - \cdots\right)\widehat{\omega} + \left(\frac{t^2}{2!} - \frac{t^4}{4!} + \frac{t^6}{6!} - \cdots\right)\widehat{\omega}^2.$$

What in the brackets are exactly the series for $\sin(t)$ and $(1-\cos(t))$. Hence we have $e^{\widehat{\omega}t} = I + \widehat{\omega}\sin(t) + \widehat{\omega}^2(1 - \cos(t))$. □

Using the Rodrigues' formula, it is immediate to see that if $t = 2k\pi, k \in \mathbb{Z}$, we have

$$e^{\widehat{\omega}2k\pi} = I$$

for all k. Hence, for a given rotation matrix $R \in SO(3)$ there are typically infinitely many exponential coordinates (ω, t) such that $e^{\widehat{\omega}t} = R$. The exponential map $\exp : so(3) \to SO(3)$ is therefore *not* one-to-one. It is also useful to know that the exponential map is not *commutative* either, i.e. for two $\widehat{\omega}_1, \widehat{\omega}_2 \in so(3)$, usually

$$e^{\widehat{\omega}_1}e^{\widehat{\omega}_2} \neq e^{\widehat{\omega}_2}e^{\widehat{\omega}_1} \neq e^{\widehat{\omega}_1 + \widehat{\omega}_2}$$

unless $\widehat{\omega}_1\widehat{\omega}_2 = \widehat{\omega}_2\widehat{\omega}_1$.

Remark 2.2. *In general, the difference between $\widehat{\omega}_1\widehat{\omega}_2$ and $\widehat{\omega}_2\widehat{\omega}_1$ is called the* Lie bracket *on $so(3)$, denoted as:*

$$[\widehat{\omega}_1, \widehat{\omega}_2] = \widehat{\omega}_1\widehat{\omega}_2 - \widehat{\omega}_2\widehat{\omega}_1, \quad \forall \widehat{\omega}_1, \widehat{\omega}_2 \in so(3).$$

Obviously, $[\widehat{\omega}_1, \widehat{\omega}_2]$ is also a skew symmetric matrix in $so(3)$. The linear structure of $so(3)$ together with the Lie bracket form the Lie algebra *of the (Lie) group $SO(3)$. For more details on the Lie group structure of $SO(3)$, the reader may refer to [MLS94]. The set of all rotation matrices $e^{\widehat{\omega}t}, t \in \mathbb{R}$ is then called a* one parameter *subgroup of $SO(3)$ and the multiplication in such a subgroup is commutative, i.e. for the same $\omega \in \mathbb{R}^3$, we have:*

$$e^{\widehat{\omega}t_1}e^{\widehat{\omega}t_2} = e^{\widehat{\omega}t_2}e^{\widehat{\omega}t_1} = e^{\widehat{\omega}(t_1+t_2)}, \quad \forall t_1, t_2 \in \mathbb{R}.$$

2.3.2 Quaternions and Lie-Cartan coordinates

Quaternions

We know that complex numbers $\mathbb{C}$ can be simply defined as $\mathbb{C} = \mathbb{R} + \mathbb{R}i$ with $i^2 = -1$. Quaternions are to generalize complex numbers in a similar fashion. The set of quaternions, denoted by $\mathbb{H}$, is defined as

$$\mathbb{H} = \mathbb{C} + \mathbb{C}j, \quad \text{with } j^2 = -1 \text{ and } i \cdot j = -j \cdot i. \tag{2.15}$$

So an element of $\mathbb{H}$ is of the form

$$q = q_0 + q_1 i + (q_2 + iq_3)j = q_0 + q_1 i + q_2 j + q_3 ij, \quad q_0, q_1, q_2, q_3 \in \mathbb{R}. \tag{2.16}$$

For simplicity of notation, in the literature ij is sometimes denoted as k. In general, the *multiplication* of any two quaternions is similar to the multiplication of two complex numbers, except that the multiplication of i and j is *anti-commutative*: $ij = -ji$. We can also similarly define the concept of *conjugation* for a quaternion

$$q = q_0 + q_1 i + q_2 j + q_3 ij \quad \Rightarrow \quad \bar{q} = q_0 - q_1 i - q_2 j - q_3 ij. \tag{2.17}$$

It is direct to check that

$$q\bar{q} = q_0^2 + q_1^2 + q_2^2 + q_3^2. \tag{2.18}$$

So $q\bar{q}$ is simply the square of the norm $\|q\|$ of q as a four dimensional vector in $\mathbb{R}^4$. For a non-zero $q \in \mathbb{H}$, i.e. $\|q\| \neq 0$, we can further define its *inverse* to be

$$q^{-1} = \bar{q}/\|q\|^2. \tag{2.19}$$

The multiplication and inverse rules defined above in fact endow the space $\mathbb{R}^4$ an algebraic structure of a *skew field*. $\mathbb{H}$ is in fact called a *Hamiltonian field*, besides another common name *quaternion field*.

One important usage of quaternion field $\mathbb{H}$ is that we can in fact embed the rotation group $SO(3)$ into it. To see this, let us focus on a special subgroup of $\mathbb{H}$, the so-called *unit quaternions*

$$\mathbb{S}^3 = \{q \in \mathbb{H} \mid \|q\|^2 = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1\}. \tag{2.20}$$

It is obvious that the set of all unit quaternions is simply the unit sphere in $\mathbb{R}^4$. To show that $\mathbb{S}^3$ is indeed a group, we simply need to prove that it is closed under the multiplication and inverse of quaternions, i.e. the multiplication of two unit quaternions is still a unit quaternion and so is the inverse of a unit quaternion. We leave this simple fact as an exercise to the reader.

Given a rotation matrix $R = e^{\widehat{\omega}t}$ with $\omega = [\omega_1, \omega_2, \omega_3]^T \in \mathbb{R}^3$ and $t \in \mathbb{R}$, we can associate to it a unit quaternion as following

$$q(R) = \cos(t/2) + \sin(t/2)(\omega_1 i + \omega_2 j + \omega_3 ij) \in \mathbb{S}^3. \tag{2.21}$$

One may verify that this association preserves the group structure between $SO(3)$ and $\mathbb{S}^3$:

$$q(R^{-1}) = q^{-1}(R), \quad q(R_1R_2) = q(R_1)q(R_2), \quad \forall R, R_1, R_2 \in SO(3). \tag{2.22}$$

Further study can show that this association is also *genuine*, i.e. for different rotation matrices, the associated unit quaternions are also different. In the opposite direction, given a unit quaternion $q = q_0 + q_1 i + q_2 j + q_3 ij \in \mathbb{S}^3$, we can use the following formulae find the corresponding rotation matrix $R(q) = e^{\widehat{\omega} t}$

$$t = 2\arccos(q_0), \quad \omega_m = \begin{cases} q_m/\sin(t/2), & t \neq 0 \\ 0, & t = 0 \end{cases}, \quad m = 1, 2, 3. \tag{2.23}$$

However, one must notice that, according to the above formula, there are two unit quaternions correspond to the same rotation matrix: $R(q) = R(-q)$, as shown in Figure 2.3. Therefore, topologically, $\mathbb{S}^3$ is

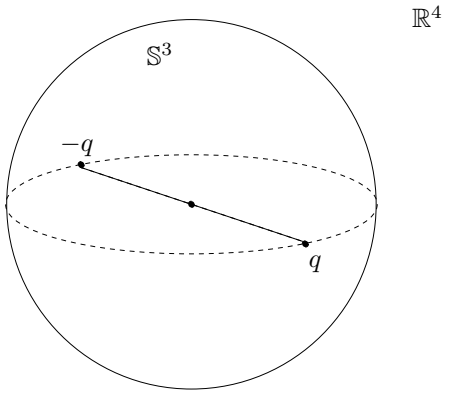

Figure 2.3. Antipodal unit quaternions q and $-q$ on the unit sphere $\mathbb{S}^3 \subset \mathbb{R}^4$ correspond to the same rotation matrix.

a double-covering of $SO(3)$. So $SO(3)$ is topologically the same as a three-dimensional projective plane $\mathbb{RP}^3$.

Compared to the exponential coordinates for rotation matrix that we studied in the previous section, using unit quaternions $\mathbb{S}^3$ to represent rotation matrices $SO(3)$, we have much less redundancy: there are only two unit quaternions correspond to the same rotation matrix while there are infinitely many for exponential coordinates. Furthermore, such a representation for rotation matrix is smooth and there is no singularity, as opposed to the *Lie-Cartan coordinates* representation which we will now introduce.

Lie-Cartan coordinates

Exponential coordinates and unit quaternions can both be viewed as ways to *globally* parameterize rotation matrices – the parameterization works for

every rotation matrix practically the same way. On the other hand, the Lie-Cartan coordinates to be introduced below falls into the category of *local* parameterizations. That is, this kind of parameterizations are only good for a portion of $SO(3)$ but not for the entire space. The advantage of such local parameterizations is we usually need only three parameters to describe a rotation matrix, instead of four for both exponential coordinates: $(\omega, t) \in \mathbb{R}^4$ and unit quaternions: $q \in \mathbb{S}^3 \subset \mathbb{R}^4$.

In the space of skew symmetric matrices $so(3)$, pick a *basis* $(\widehat{\omega}_1, \widehat{\omega}_2, \widehat{\omega}_3)$, i.e. the three vectors $\omega_1, \omega_2, \omega_3$ are linearly independent. Define a mapping (a parameterization) from $\mathbb{R}^3$ to $SO(3)$ as

$$\alpha : \ (\alpha_1, \alpha_2, \alpha_3) \quad \mapsto \quad \exp(\alpha\widehat{\omega}_1 + \alpha_2\widehat{\omega}_2 + \alpha_3\widehat{\omega}_3).$$

The coordinates $(\alpha_1, \alpha_2, \alpha_3)$ are called the *Lie-Cartan coordinates of the first kind* relative to the basis $(\widehat{\omega}_1, \widehat{\omega}_2, \widehat{\omega}_3)$. Another way to parameterize the group $SO(3)$ using the same basis is to define another mapping from $\mathbb{R}^3$ to $SO(3)$ by

$$\beta : \ (\beta_1, \beta_2, \beta_3) \quad \mapsto \quad \exp(\beta_1\widehat{\omega}_1)\exp(\beta_2\widehat{\omega}_2)\exp(\beta_3\widehat{\omega}_3).$$

The coordinates $(\beta_1, \beta_2, \beta_3)$ are then called the *Lie-Cartan coordinates of the second kind*.

In the special case when we choose $\omega_1, \omega_2, \omega_3$ to be the principal axes Z, Y, X, respectively, i.e.

$$\omega_1 = [0, 0, 1]^T \doteq \mathbf{z}, \quad \omega_2 = [0, 1, 0]^T \doteq \mathbf{y}, \quad \omega_3 = [1, 0, 0]^T \doteq \mathbf{x},$$

the Lie-Cartan coordinates of the second kind then coincide with the well-known ZYX *Euler angles* parametrization and $(\beta, \beta_2, \beta_3)$ are the corresponding Euler angles. The rotation matrix is then expressed by:

$$R(\beta_1, \beta_2, \beta_3) = \exp(\beta_1\widehat{\mathbf{z}})\exp(\beta_2\widehat{\mathbf{y}})\exp(\beta_3\widehat{\mathbf{x}}). \tag{2.24}$$

Similarly we can define YZX Euler angles and ZYZ Euler angles. There are instances when this representation becomes singular and for certain rotation matrices, their corresponding Euler angles cannot be uniquelly determines. For example, the ZYX Euler angles become singular when $\beta_2 = -\pi/2$. The presence of such singularities is quite expected because of the topology of the space $SO(3)$. Globally $SO(3)$ is very much like a sphere in $\mathbb{R}^4$ as we have shown in the previous section, and it is well known that any attempt to find a global (three-dimensional) coordinate chart for it is doomed to fail.

2.4 Rigid body motion and its representations

In Section 2.3, we have studied extensively pure rotational rigid body motion and different representations for rotation matrix. In this section, we

will study how to represent a rigid body motion in general - a motion with both rotation and translation.

Figure 2.4 illustrates a moving rigid object with a coordinate frame C attached to it. To describe the coordinates of a point p on the object with

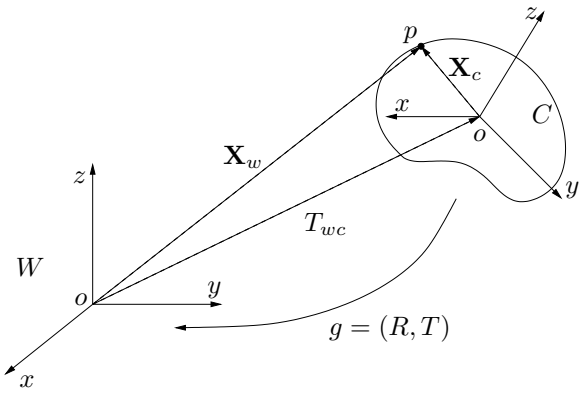

Figure 2.4. A rigid body motion between a moving frame C and a world frame W.

respect to the world frame W, it is clear from the figure that the vector $\mathbf{X}_w$ is simply the sum of the translation $T_{wc} \in \mathbb{R}^3$ of the center of frame C relative to that of frame W and the vector $\mathbf{X}_c$ but expressed relative to frame W. Since $\mathbf{X}_c$ are the coordinates of the point p relative to the frame C, with respect to the world frame W, it becomes $R_{wc}\mathbf{X}_c$ where $R_{wc} \in SO(3)$ is the relative rotation between the two frames. Hence the coordinates $\mathbf{X}_w$ are given by:

$$\mathbf{X}_w = R_{wc}\mathbf{X}_c + T_{wc}. \tag{2.25}$$

Usually, we denote the full rigid motion as $g_{wc} = (R_{wc}, T_{wc})$ or simply $g = (R, T)$ if the frames involved are clear from the context. Then g represents not only a description of the configuration of the rigid body object but a transformation of coordinates between the frames. In a compact form we may simply write:

$$\mathbf{X}_w = g_{wc}(\mathbf{X}_c).$$

The set of all possible configurations of rigid body can then be described as:

$$\boxed{SE(3) = \{g = (R, T) \mid R \in SO(3), T \in \mathbb{R}^3\} = SO(3) \times \mathbb{R}^3}$$

so called special Euclidean group $SE(3)$. Note that $g = (R, T)$ is not yet a matrix representation for the group $SE(3)$ as we defined in Section 2.2. To obtain such a representation, we must introduce the so-called homogeneous coordinates.

2.4.1 Homogeneous representation

One may have already noticed from equation (2.25) that, unlike the pure rotation case, the coordinate transformation for a full rigid body motion is not linear but *affine* instead.[8] Nonetheless, we may convert such an affine transformation to a linear one by using the so-called *homogeneous coordinates*: Appending 1 to the coordinates $\mathbf{X} = [X_1, X_2, X_3]^T \in \mathbb{R}^3$ of a point $p \in \mathbb{E}^3$ yields a vector in $\mathbb{R}^4$ denoted by $\bar{\mathbf{X}}$:

$$\bar{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \in \mathbb{R}^4.$$

Such an extension of coordinates, in effect, has embedded the Euclidean space $\mathbb{E}^3$ into a hyper-plane in $\mathbb{R}^4$ instead of $\mathbb{R}^3$. Homogeneous coordinates of a vector $v = \mathbf{X}(q) - \mathbf{X}(p)$ are defined as the difference between homogeneous coordinates of the two points hence of the form:

$$\bar{v} = \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{X}(q) \\ 1 \end{bmatrix} - \begin{bmatrix} \mathbf{X}(p) \\ 1 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix} \in \mathbb{R}^4.$$

Notice that, in $\mathbb{R}^4$, vectors of the above form give rise to a subspace hence all linear structures of the original vectors $v \in \mathbb{R}^3$ are perfectly preserved by the new representation. Using the new notation, the transformation (2.25) can be re-written as:

$$\bar{\mathbf{X}}_w = \begin{bmatrix} \mathbf{X}_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & T_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X}_c \\ 1 \end{bmatrix} =: \bar{g}_{wc} \bar{\mathbf{X}}_c$$

where the 4×4 matrix $\bar{g}_{wc} \in \mathbb{R}^{4\times 4}$ is called the *homogeneous representation* of the rigid motion $g_{wc} = (R_{wc}, T_{wc}) \in SE(3)$. In general, if $g = (R, T)$, then its homogeneous representation is:

$$\bar{g} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}. \tag{2.26}$$

Notice that, by introducing a little redundancy into the notation, we represent a rigid body transformation of coordinates by a linear matrix multiplication. The homogeneous representation of g in (2.26) gives rise to a natural matrix representation of the special Euclidean group $SE(3)$:

$$\boxed{SE(3) = \left\{ \bar{g} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \,\middle|\, R \in SO(3), T \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}}$$

[8] We say two vectors u, v are related by a *linear* transformation if $u = Av$ for some matrix A, and by an *affine* transformation if $u = Av + b$ for some matrix A and vector b.

It is then straightforward to verify that so-defined $SE(3)$ indeed satisfies all the requirements of a group. In particular, $\forall g_1, g_2$ and $g \in SE(3)$, we have:

$$\bar{g}_1\bar{g}_2 = \begin{bmatrix} R_1 & T_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2 & T_2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1R_2 & R_1T_2 + T_1 \\ 0 & 1 \end{bmatrix} \in SE(3)$$

and

$$\bar{g}^{-1} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R^T & -R^TT \\ 0 & 1 \end{bmatrix} \in SE(3).$$

In homogeneous representation, the action of a rigid body transformation $g \in SE(3)$ on a vector $v = \mathbf{X}(q) - \mathbf{X}(p) \in \mathbb{R}^3$ becomes:

$$\bar{g}_*(\bar{v}) = \bar{g}\bar{\mathbf{X}}(q) - \bar{g}\bar{\mathbf{X}}(p) = \bar{g}\bar{v}.$$

That is, the action is simply represented by a matrix multiplication. The reader can verify that such an action preserves both inner product and cross product hence $\bar{g}$ indeed represents a rigid body transformation according to the definition we gave in Section 2.2.

2.4.2 Canonical exponential coordinates

In Section 2.3.1, we have studied exponential coordinates for rotation matrix $R \in SO(3)$. Similar coordination also exists for the homogeneous representation of a full rigid body motion $g \in SE(3)$. For the rest of this section, we demonstrate how to extend the results we have developed for rotational motion in Section 2.3.1 to a full rigid body motion. The results developed here will be extensively used throughout the entire book.

Consider that the motion of a continuously moving rigid body object is described by a curve from $\mathbb{R}$ to $SE(3)$: $g(t) = (R(t), T(t))$, or in homogeneous representation:

$$g(t) = \begin{bmatrix} R(t) & T(t) \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}.$$

Here, for simplicity of notation, we will remove the "bar" off from the symbol $\bar{g}$ for homogeneous representation and simply use g for the same matrix. We will use the same convention for point: $\mathbf{X}$ for $\bar{\mathbf{X}}$ and for vector: v for $\bar{v}$ whenever their correct dimension is clear from the context. Similar as in the pure rotation case, lets first look at the structure of the matrix $\dot{g}(t)g^{-1}(t)$:

$$\dot{g}(t)g^{-1}(t) = \begin{bmatrix} \dot{R}(t)R^T(t) & \dot{T}(t) - \dot{R}(t)R^T(t)T(t) \\ 0 & 0 \end{bmatrix}. \tag{2.27}$$

From our study of rotation matrix, we know $\dot{R}(t)R^T(t)$ is a skew symmetric matrix, i.e. there exists $\widehat{\omega}(t) \in so(3)$ such that $\widehat{\omega}(t) = \dot{R}(t)R^T(t)$. Define a

vector $v(t) \in \mathbb{R}^3$ such that $v(t) = \dot{T}(t) - \widehat{\omega}(t)T(t)$. Then the above equation becomes:

$$\dot{g}(t)g^{-1}(t) = \begin{bmatrix} \widehat{\omega}(t) & v(t) \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4}.$$

If we further define a matrix $\widehat{\xi} \in \mathbb{R}^{4\times 4}$ to be:

$$\widehat{\xi}(t) = \begin{bmatrix} \widehat{\omega}(t) & v(t) \\ 0 & 0 \end{bmatrix},$$

then we have:

$$\dot{g}(t) = (\dot{g}(t)g^{-1}(t))g(t) = \widehat{\xi}(t)g(t). \tag{2.28}$$

$\widehat{\xi}$ can be viewed as the "tangent vector" along the curve of $g(t)$ and used for approximate $g(t)$ locally:

$$g(t+dt) \approx g(t) + \widehat{\xi}(t)g(t)dt = \left(I + \widehat{\xi}(t)dt\right) g(t).$$

In robotics literature a 4×4 matrix of the form as $\widehat{\xi}$ is called a *twist*. The set of all twist is then denoted as:

$$\boxed{se(3) = \left\{ \widehat{\xi} = \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix} \;\middle|\; \widehat{\omega} \in so(3), v \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}}$$

$se(3)$ is called the tangent space (or Lie algebra) of the matrix group $SE(3)$. We also define two operators $\vee$ and $\wedge$ to convert between a twist $\widehat{\xi} \in se(3)$ and its *twist coordinates* $\xi \in \mathbb{R}^6$ as follows:

$$\begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix}^{\vee} \doteq \begin{bmatrix} v \\ \omega \end{bmatrix} \in \mathbb{R}^6, \qquad \begin{bmatrix} v \\ \omega \end{bmatrix}^{\wedge} \doteq \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4}.$$

In the twist coordinates ξ, we will refer to v as the *linear velocity* and ω as the *angular velocity*, which indicates that they are related to either translational or rotational part of the full motion. Let us now consider a special case of the equation (2.28) when the twist $\widehat{\xi}$ is a constant matrix:

$$\dot{g}(t) = \widehat{\xi}g(t).$$

Hence we have again a time-invariant linear ordinary differential equation, which can be intergrated to give:

$$g(t) = e^{\widehat{\xi}t}g(0).$$

Assuming that the initial condition $g(0) = I$ we may conclude that:

$$\boxed{g(t) = e^{\widehat{\xi}t}}$$

where the twist exponential is:

$$e^{\widehat{\xi}t} = I + \widehat{\xi}t + \frac{(\widehat{\xi}t)^2}{2!} + \cdots + \frac{(\widehat{\xi}t)^n}{n!} + \cdots. \tag{2.29}$$

Using Rodrigues' formula introduced in the previous section, it is straightforward to obtain that:

$$\boxed{e^{\widehat{\xi}t} = \begin{bmatrix} e^{\widehat{\omega}t} & (I - e^{\widehat{\omega}t})\widehat{\omega}v + \omega\omega^T vt \\ 0 & 1 \end{bmatrix}} \tag{2.30}$$

It it clear from this expression that the exponential of $\widehat{\xi}t$ is indeed a rigid body transformation matrix in $SE(3)$. Therefore the exponential map defines a mapping from the space $se(3)$ to $SE(3)$:

$$\begin{aligned} \exp: \ se(3) &\rightarrow SE(3) \\ \widehat{\xi} \in se(3) &\mapsto e^{\widehat{\xi}} \in SE(3) \end{aligned}$$

and the twist $\widehat{\xi} \in se(3)$ is also called the *exponential coordinates* for $SE(3)$, as $\widehat{\omega} \in so(3)$ for $SO(3)$.

One question remains to answer: Can every rigid body motion $g \in SE(3)$ always be represented in such an exponential form? The answer is yes and is formulated in the following theorem:

Theorem 2.3 (Surjectivity of the exponential map onto $SE(3)$). *For any $g \in SE(3)$, there exist (not necessarily unique) twist coordinates $\xi = (v, \omega)$ and $t \in \mathbb{R}$ such that $g = e^{\widehat{\xi}t}$.*

Proof. The proof is constructive. Suppose $g = (R, T)$. For the rotation matrix $R \in SO(3)$ we can always find (ω, t) with $\|\omega\| = 1$ such that $e^{\widehat{\omega}t} = R$. If $t \neq 0$, from equation (2.30), we can solve for $v \in \mathbb{R}^3$ from the linear equation

$$(I - e^{\widehat{\omega}t})\widehat{\omega}v + \omega\omega^T vt = T \quad \Rightarrow \quad v = [(I - e^{\widehat{\omega}t})\widehat{\omega} + \omega\omega^T t]^{-1}T.$$

If $t = 0$, then $R = I$. We may simply choose $\omega = 0, v = T/\|T\|$ and $t = \|T\|$. □

Similar to the exponential coordinates for rotation matrix, the exponential map from $se(3)$ to $SE(3)$ is not injective hence not one-to-one. There are usually infinitely many exponential coordinates (or twists) that correspond to every $g \in SE(3)$. Similarly as in the pure rotation case, the linear structure of $se(3)$, together with the closure under the Lie bracket operation:

$$[\widehat{\xi}_1, \widehat{\xi}_2] = \widehat{\xi}_1\widehat{\xi}_2 - \widehat{\xi}_2\widehat{\xi}_1 = \begin{bmatrix} \widehat{\omega_1 \times \omega_2} & \omega_1 \times v_2 - \omega_2 \times v_1 \\ 0 & 0 \end{bmatrix} \in se(3).$$

makes $se(3)$ the Lie algebra for $SE(3)$. The two rigid body motions $g_1 = e^{\widehat{\xi}_1}$ and $g_2 = e^{\widehat{\xi}_2}$ commute with each other : $g_1 g_2 = g_2 g_1$, only if $[\widehat{\xi}_1, \widehat{\xi}_2] = 0$.

2.5 Coordinates and velocity transformation

In the above presentation of rigid body motion we described how 3-D points move relative to the camera frame. In computer vision we usually need to know how the coordinates of the points and their velocities change with respect to camera frames at different locations. This is mainly because that it is usually much more convenient and natural to choose the camera frame as the reference frame and to describe both camera motion and 3-D points relative to it. Since the camera may be moving, we need to know how to transform quantities such as coordinates and velocities from one camera frame to another. In particular, how to correctly express location and velocity of a (feature) point in terms of that of a moving camera. Here we introduce a few conventions that we will use for the rest of this book. The time $t \in \mathbb{R}$ will be always used as an index to register camera motion. Even in the discrete case when a few snapshots are given, we will order them by some time indexes as if th! ey had been taken in such order. We found that time is a good uniform index for both discrete case and continuous case, which will be treated in a unified way in this book. Therefore, we will use $g(t) = (R(t), T(t)) \in SE(3)$ or:

$$g(t) = \begin{bmatrix} R(t) & T(t) \\ 0 & 1 \end{bmatrix} \in SE(3)$$

to denote the relative displacement between some fixed world frame W and the camera frame C at time $t \in \mathbb{R}$. Here we will ignore the subscript cw from supposedly $g_{cw}(t)$ as long as the relativity is clear from the context. By default, we assume $g(0) = I$, i.e. at time $t = 0$ the camera frame coincides with the world frame. So if the coordinates of a point $p \in \mathbb{E}^3$ relative to the world frame are $\mathbf{X}_0 = \mathbf{X}(0)$, its coordinates relative to the camera at time t are then:

$$\boxed{\mathbf{X}(t) = R(t)\mathbf{X}_0 + T(t)} \tag{2.31}$$

or in homogeneous representation:

$$\boxed{\mathbf{X}(t) = g(t)\mathbf{X}_0}. \tag{2.32}$$

If the camera is at locations $g(t_1), \ldots, g(t_m)$ at time $t_1, \ldots, t_m$ respectively, then the coordinates of the same point p are given as $\mathbf{X}(t_i) = g(t_i)\mathbf{X}_0, i = 1, \ldots, m$ correspondingly. If it is only the position, not the time, that matters, we will very often use g_i as a shorthand for $g(t_i)$ and similarly $\mathbf{X}_i$ for $\mathbf{X}(t_i)$.

When the starting time is not $t = 0$, the relative motion between the camera at time t_2 relative to the camera at time t_1 will be denoted as $g(t_2, t_1) \in SE(3)$. Then we have the following relationship between coordinates of the same point p:

$$\mathbf{X}(t_2) = g(t_2, t_1)\mathbf{X}(t_1), \quad \forall t_2, t_1 \in \mathbb{R}.$$

Now consider a third position of the camera at $t = t_3 \in \mathbb{R}^3$, as shown in Figure 2.5. The relative motion between the camera at t_3 and t_2 is $g(t_3, t_2)$ and between t_3 and t_1 is $g(t_3, t_1)$. We then have the following relationship

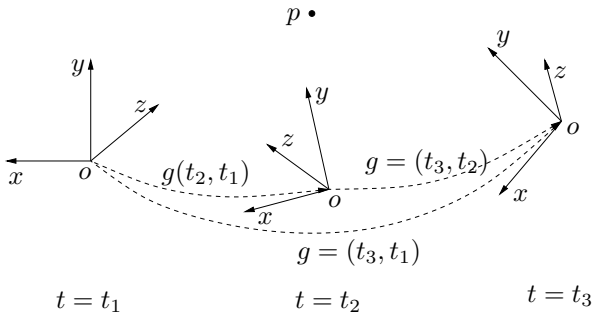

Figure 2.5. Composition of rigid body motions.

among coordinates

$$\mathbf{X}(t_3) = g(t_3, t_2)\mathbf{X}(t_2) = g(t_3, t_2)g(t_2, t_1)\mathbf{X}(t_1).$$

Comparing with the direct relationship between coordinates at t_3 and t_1:

$$\mathbf{X}(t_3) = g(t_3, t_1)\mathbf{X}(t_1),$$

the following *composition rule* for consecutive motions must hold:

$$g(t_3, t_1) = g(t_3, t_2)g(t_2, t_1).$$

The composition rule describes the coordinates $\mathbf{X}$ of the point p relative to any camera position, if they are known with respect to a particular one. The same composition rule implies the *rule of inverse*

$$g^{-1}(t_2, t_1) = g(t_1, t_2)$$

since $g(t_2, t_1)g(t_1, t_2) = g(t_2, t_2) = I$. In case time is of no physical meaning to a particular problem, we might use g_{ij} as a shorthand for $g(t_i, t_j)$ with $t_i, t_j \in \mathbb{R}$.

Having understood transformation of coordinates, we now study what happens to velocity. We know that the coordinates $\mathbf{X}(t)$ of a point $p \in \mathbb{E}^3$ relative to a moving camera, are a function of time t:

$$\mathbf{X}(t) = g_{cw}(t)\mathbf{X}_0.$$

Then the velocity of the point p relative to the (instantaneous) camera frame is:

$$\dot{\mathbf{X}}(t) = \dot{g}_{cw}(t)\mathbf{X}_0.$$

In order express $\dot{\mathbf{X}}(t)$ in terms of quantities expressed in the moving frame we substitute for $\mathbf{X}_0 = g_{cw}^{-1}(t)\mathbf{X}_0$ and using the notion of twist define:

$$\widehat{V}_{cw}^c(t) = \dot{g}_{cw}(t)g_{cw}^{-1}(t) \in se(3) \tag{2.33}$$

where an expression for $\dot{g}_{cw}(t)g_{cw}^{-1}(t)$ can be found in (2.27). The above equation can be rewritten as:

$$\boxed{\dot{\mathbf{X}}(t) = \widehat{V}_{cw}^{c}(t)\mathbf{X}(t)} \tag{2.34}$$

Since $\widehat{V}_{cw}^{c}(t)$ is of the form:

$$\widehat{V}_{cw}^{c}(t) = \begin{bmatrix} \widehat{\omega}(t) & v(t) \\ 0 & 0 \end{bmatrix},$$

we can also write the velocity of the point in 3-D coordinates (instead of in the homogeneous coordinates) as:

$$\boxed{\dot{\mathbf{X}}(t) = \widehat{\omega}(t)\mathbf{X}(t) + v(t)}. \tag{2.35}$$

The physical interpretation of the symbol $\widehat{V}_{cw}^{c}$ is the velocity of the world frame moving relative to the camera frame, as viewed in the camera frame – the subscript and superscript of $\widehat{V}_{cw}^{c}$ indicate that. Usually, to clearly specify the physical meaning of a velocity, we need to specify: It is the velocity of which frame moving relative to which frame, and in which frame it is viewed. If we change where we view the velocity, the expression will change accordingly. For example suppose that a viewer is in another coordinate frame displaced relative to the camera frame by a rigid body transformation $g \in SE(3)$. Then the coordinates of the same point p relative to this frame are $\mathbf{Y}(t) = g\mathbf{X}(t)$. Compute the velocity in the new frame we have:

$$\dot{\mathbf{Y}}(t) = g\dot{g}_{cw}(t)g_{cw}^{-1}(t)g^{-1}\mathbf{Y}(t) = g\widehat{V}_{cw}^{c}g^{-1}\mathbf{Y}(t).$$

So the new velocity (or twist) is:

$$\widehat{V} = g\widehat{V}_{cw}^{c}g^{-1}.$$

This is the simply the same physical quantity but viewed from a different vantage point. We see that the two velocities are related through a mapping defined by the relative motion g, in particular:

$$\begin{aligned} ad_g : \ se(3) &\rightarrow se(3) \\ \widehat{\xi} &\mapsto g\widehat{\xi}g^{-1}. \end{aligned}$$

This is the so-called *adjoint map* on the space $se(3)$. Using this notation in the previous example we have $\widehat{V} = ad_g(\widehat{V}_{cw}^{c})$. Clearly, the adjoint map transforms velocity from one frame to another. Using the fact that $g_{cw}(t)g_{wc}(t) = I$, it is straightforward to verify that:

$$\widehat{V}_{cw}^{c} = \dot{g}_{cw}g_{cw}^{-1} = -g_{wc}^{-1}\dot{g}_{wc} = -g_{cw}(\dot{g}_{wc}g_{wc}^{-1})g_{cw}^{-1} = ad_{g_{cw}}(-\widehat{V}_{wc}^{w}).$$

Hence $\widehat{V}_{cw}^{c}$ can also be interpreted as the *negated* velocity of the camera moving relative to the world frame, viewed in the (instantaneous) camera frame.

2.6 Summary

The rigid body motion introduced in this chapter is an element $g \in SE(3)$. The two most commonly used representation of elements of $g \in SE(3)$ are:

- Homogeneous representation:

$$\bar{g} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \text{ with } R \in SO(3) \text{ and } T \in \mathbb{R}^3.$$

- Twist representation:

$$g(t) = e^{\widehat{\xi} t} \text{ with the twist coordinates } \xi = (v, \omega) \in \mathbb{R}^6 \text{ and } t \in \mathbb{R}.$$

In the instanteneous case the velocity of a point with respect to the (instanteneous) camera frame is:

$$\dot{\mathbf{X}}(t) = \widehat{V}^c_{cw}(t)\mathbf{X}(t) \text{ where } \widehat{V}^c_{cw} = \dot{g}_{cw} g^{-1}_{cw}$$

and $g_{cw}(t)$ is the configuration of the camera with respect to the world frame. Using the actual 3D coordinates, the velocity of a 3D point yields the familiar relationship:

$$\dot{\mathbf{X}}(t) = \widehat{\omega}(t)\mathbf{X}(t) + v(t).$$

2.7 References

The presentation of the material in this chapter follows the development in [**?**]. More details on the abstract treatment of the material as well as further references can be also found there.

2.8 Exercises

1. **Linear vs. nonlinear maps**
   Suppose $A, B, C, X \in \mathbb{R}^{n\times n}$. Consider the following maps from $\mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ and determine if they are linear or not. Give a brief proof if true and a counterexample if false:

   $$\begin{aligned} &\text{(a)} \quad X \mapsto AX + XB \\ &\text{(b)} \quad X \mapsto AX + BXC \\ &\text{(c)} \quad X \mapsto AXA - B \\ &\text{(d)} \quad X \mapsto AX + XBX \end{aligned}$$

   Note: A map $f : \mathbb{R}^n \to \mathbb{R}^m : x \mapsto f(x)$ is called linear if $f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$ for all $\alpha, \beta \in \mathbb{R}$ and $x, y \in \mathbb{R}^n$.

2. **Group structure of $SO(3)$**
   Prove that the space $SO(3)$ is a group, i.e. it satisfies all four axioms in the definition of group.

3. **Skew symmetric matrices**
   Given any vector $\omega = [\omega_1, \omega_2, \omega_3]^T \in \mathbb{R}^3$, define a 3×3 matrix associated to ω:
   $$\widehat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{2.36}$$
   According to the definition, $\widehat{\omega}$ is *skew symmetric*, i.e. $\widehat{\omega}^T = -\widehat{\omega}$. Now for any matrix $A \in \mathbb{R}^{3\times 3}$ with determinant $\det A = 1$, show that the following equation holds:
   $$A^T \widehat{\omega} A = \widehat{A^{-1}\omega}. \tag{2.37}$$
   Then, in particular, if A is a rotation matrix, the above equation holds.
   Hint: Both $A^T\widehat{(\cdot)}A$ and $\widehat{A^{-1}(\cdot)}$ are linear maps with ω as the variable. What do you need to prove that two linear maps are the same?

4. **Rotation as rigid body motion**
   Given a rotation matrix $R \in SO(3)$, its action on a vector v is defined as Rv. Prove that any rotation matrix must preserve both the inner product and cross product of vectors. Hence, a rotation is indeed a rigid body motion.

5. **Range and null space**
   Recall that given a matrix $A \in \mathbb{R}^{m\times n}$, its *null space* is defined as a subspace of $\mathbb{R}^n$ consisting of all vectors $x \in \mathbb{R}^n$ such that $Ax = 0$. It is usually denoted as $Nu(A)$. The *range* of the matrix A is defined as a subspace of $\mathbb{R}^m$ consisting of all vectors $y \in \mathbb{R}^m$ such that there exists some $x \in \mathbb{R}^n$ such that $y = Ax$. It is denoted as $Ra(A)$. In mathematical terms,
   $$\begin{aligned} Nu(A) &= \{x \in \mathbb{R}^n \mid Ax = 0\}, \\ Ra(A) &= \{y \in \mathbb{R}^m \mid \exists x \in \mathbb{R}^n, y = Ax\} \end{aligned}$$
   (a) Recall that a set of vectors V is a subspace if for all vectors $x, y \in V$ and scalars $\alpha, \beta \in \mathbb{R}$, $\alpha x + \beta y$ is also a vector in V. Show that both $Nu(A)$ and $Ra(A)$ are indeed subspaces.
   (b) What are $Nu(\widehat{\omega})$ and $Ra(\widehat{\omega})$ for a non-zero vector $\omega \in \mathbb{R}^3$? Can you describe intuitively the geometric relationship between these two subspaces in $\mathbb{R}^3$? (A picture might help.)

6. **Properties of rotation matrices**
   Let $R \in SO(3)$ be a rotation matrix generated by rotating about a unit vector $\omega \in \mathbb{R}^3$ by θ radians. That is $R = e^{\widehat{\omega}\theta}$.

(a) What are the eigenvalues and eigenvectors of $\widehat{\omega}$? You may use Matlab and try some examples first if you have little clue. If you happen to find a brutal-forth way to do it, can you instead use results in Exercise 3 to simplify the problem first?

(b) Show that the eigenvalues of R are $1, e^{i\theta}, e^{-i\theta}$ where $i = \sqrt{-1}$ the imaginary unit. What is the eigenvector which corresponds to the eigenvalue 1? This actually gives another proof for $\det(e^{\widehat{\omega}\theta}) = 1 \cdot e^{i\theta} \cdot e^{-i\theta} = +1$ but not -1.

7. **Adjoint transformation on twist**
Given a rigid body motion g and a twist $\widehat{\xi}$

$$g = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \in SE(3), \quad \widehat{\xi} = \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix} \in se(3),$$

show that $g\widehat{\xi}g^{-1}$ is still a twist. Notify what the corresponding ω and v terms have become in the new twist. The adjoint map is kind of a generalization of $R\widehat{\omega}R^T = \widehat{R\omega}$.

Chapter 3
Image formation

This chapter introduces simple mathematical models of the image formation process. In a broad figurative sense, vision is the inverse problem of image formation: the latter studies how objects give raise to images, while the former attempts to use images to recover a description of objects in space. Therefore, designing vision algorithms requires first developing a suitable model of image formation. Suitable in this context does not necessarily mean physically accurate: the level of abstraction and complexity in modeling image formation must trade off physical constraints and mathematical simplicity in order to result in a manageable model (i.e. one that can be easily inverted). Physical models of image formation easily exceed the level of complexity necessary and appropriate to this book, and determining the right model for the problem at hand is a form of engineering art.

It comes at no surprise, then, that the study of image formation has for centuries been in the domain of artistic reproduction and composition, more so than in mathematics and engineering. Rudimentary understanding of the geometry of image formation, which includes various models for projecting the three-dimensional world onto a plane (e.g., a canvas), is implicit in various forms of visual arts from all ancient civilization. However, the roots to formalizing the geometry of image formation into a mathematical model can be traced back to the work of Euclid in the 6th century B.C.. Examples of correct perspective projection are visible in the stunning frescoes and mosaics of Pompeii (Figure 3.1) from the first century B.C. Unfortunately, these skills seem to have been lost with the fall of the Roman empire, and it took over a thousand years for correct perspective projection

to dominate paintings again, in the late 14th century. It was the early renaissance painters who developed systematic methods for determining the perspective projection of three-dimensional landscapes. The first treatise on perspective was published by Leon Battista Alberti [**?**], who emphasizes the "eye's view" of the world and capturing correctly the geometry of the projection process. The renaissance coincided with the first attempts to formalize the notion of perspective and place it on a solid analytical footing. It is no coincidence that the early attempts to formalize the rules of perspective came from artists proficient in architecture and engineering, such as Alberti and Brunelleschi. Geometry, however, is only part of the

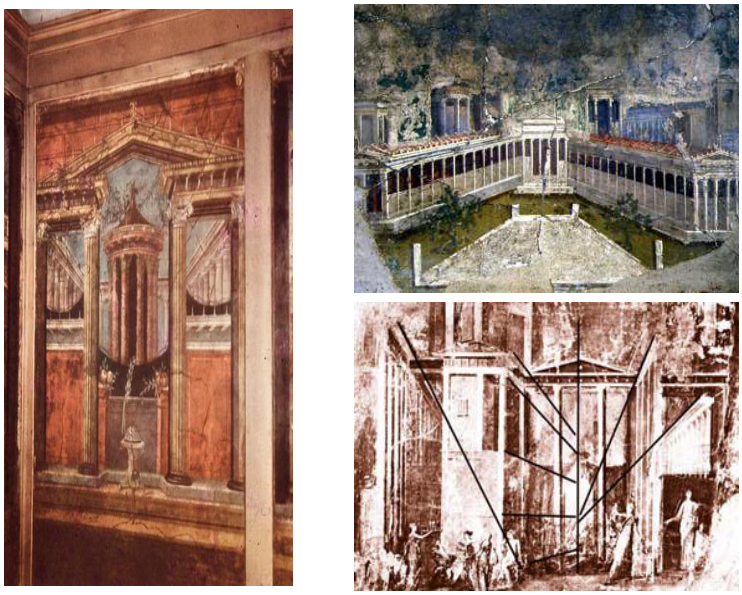

Figure 3.1. Frescoes from the first century B.C. in Pompeii. More (left) or less (right) correct perspective projection is visible in the paintings. The skill was lost during the middle ages, and it did not reappear in paintings until fifteen centuries later, in the early renaissance.

image formation process: in order to generate an image, we need to decide not only where to draw what, but what "color", or grey-scale value, to assign to any particular location on an image. The interaction of light with matter is at the core of the studies of Leonardo Da Vinci in the 1500s, and his insights on perspective, shading, color and even stereopsis are vibrantly expressed in his notebooks. Renaissance painters, such as Caravaggio or Raphael, exhibited rather sophisticated skills in rendering light and color that remain remarkably compelling to this day.

In this chapter, we will follow the historical development of models for image formation by giving a precise description of a simple model of the *geometry* of image formation, and then describing a rather crude model of its *photometry*, that is of how light interacts with geometry to give rise to images. As we will see, we will use assumptions strong enough to reduce the image formation process to an entirely geometric model, which we can then study analytically.

3.1 Representation of images

An *image*, as far as this book is concerned, is a two-dimensional brightness array. In other words, it is a map I, defined on a compact region Ω of a two-dimensional surface, taking values in the positive reals. For instance, in the case of a camera, Ω is a planar, rectangular region occupied by the photographic medium (or by the CCD sensor), so that we have:

$$I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}_+; \quad (x, y) \mapsto I(x, y). \tag{3.1}$$

Such an image can be represented, for instance, as the graph of I, as in Figure 3.2. In the case of a digital image, both the domain Ω and the range $\mathbb{R}_+$ are discretized. For instance, $\Omega = [1, 640] \times [1, 480] \subset \mathbb{Z}^2$ and $\mathbb{R}_+$ is approximated by the interval $[0, 255] \subset \mathbb{Z}_+$. Such an image can be represented as an array of numbers as in Table 3.1.

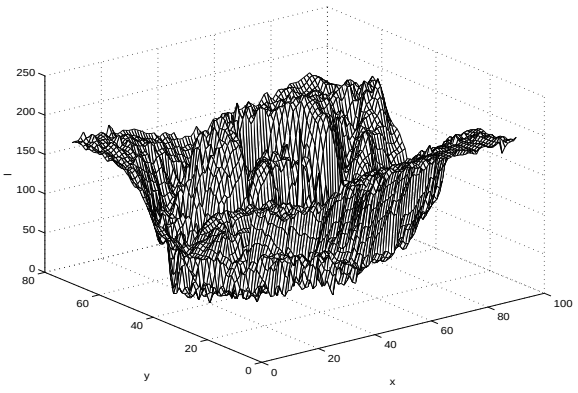

Figure 3.2. An image I represented as a two-dimensional surface.

The values taken by the map I depend upon physical properties of the scene being viewed, such as its shape, its material reflectance properties and the distribution of the light sources. Despite the fact that Figure 3.2

188	186	188	187	168	130	101	99	110	113	112	107	117	140	153	153	156	158	156	153
189	189	188	181	163	135	109	104	113	113	110	109	117	134	147	152	156	163	160	156
190	190	188	176	159	139	115	106	114	123	114	111	119	130	141	154	165	160	156	151
190	188	188	175	158	139	114	103	113	126	112	113	127	133	137	151	165	156	152	145
191	185	189	177	158	138	110	99	112	119	107	115	137	140	135	144	157	163	158	150
193	183	178	164	148	134	118	112	119	117	118	106	122	139	140	152	154	160	155	147
185	181	178	165	149	135	121	116	124	120	122	109	123	139	141	154	156	159	154	147
175	176	176	163	145	131	120	118	125	123	125	112	124	139	142	155	158	158	155	148
170	170	172	159	137	123	116	114	119	122	126	113	123	137	141	156	158	159	157	150
171	171	173	157	131	119	116	113	114	118	125	113	122	135	140	155	156	160	160	152
174	175	176	156	128	120	121	118	113	112	123	114	122	135	141	155	155	158	159	152
176	174	174	151	123	119	126	121	112	108	122	115	123	137	143	156	155	152	155	150
175	169	168	144	117	117	127	122	109	106	122	116	125	139	145	158	156	147	152	148
179	179	180	155	127	121	118	109	107	113	125	133	130	129	139	153	161	148	155	157
176	183	181	153	122	115	113	106	105	109	123	132	131	131	140	151	157	149	156	159
180	181	177	147	115	110	111	107	107	105	120	132	133	133	141	150	154	148	155	157
181	174	170	141	113	111	115	112	113	105	119	130	132	134	144	153	156	148	152	151
180	172	168	140	114	114	118	113	112	107	119	128	130	134	146	157	162	153	153	148
186	176	171	142	114	114	116	110	108	104	116	125	128	134	148	161	165	159	157	149
185	178	171	138	109	110	114	110	109	97	110	121	127	136	150	160	163	158	156	150

Table 3.1. The image I represented as a two-dimensional table of integers.

and Table 3.1 do not seem very indicative of the properties of the scene they portray, this is how they are represented in a computer. A different representation of the same image that is better suited for interpretation by the human visual system is obtained by generating a *picture*. A picture is a scene - different from the true one - that produces on the imaging sensor (the eye in this case) the same image as the original scene. In this sense *pictures* are "controlled illusions": they are scenes different from the true ones (they are *flat*), that produce in the eye the same image as the original scenes. A picture of the same image I described in Figure 3.2 and Table 3.1 is shown in Figure 3.3. Although the latter seems more *informative* on the content of the scene, it is merely a different representation and contains exactly the same information.

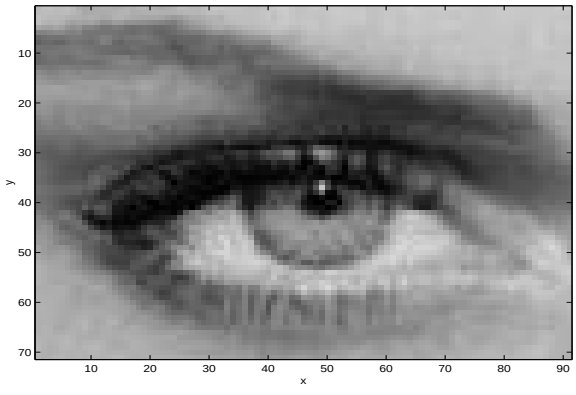

Figure 3.3. A "picture" of the image I (compare with image 3.1).

3.2 Lenses, surfaces and light

In order to describe the image formation process, we must specify the value of the map I at each point (x, y) in Ω. Such a value $I(x, y)$ is typically called *image brightness*, or more formally *irradiance*. It has the units of power per unit area ($Watts/m^2$) and describes the energy falling onto a small patch of the imaging sensor. The irradiance at a point of coordinates (x, y) is typically obtained by integrating energy both in time (for instance the shutter interval in a camera, or the integration time in a CCD) and in space. The region of space that contributes to the irradiance at (x, y) depends upon the geometry and optics of the imaging device, and is by no means trivial to determine. In Section 3.2.1 we will adopt common simplifying assumptions to approximate it. Once the region of space is determined, the energy it contributes depends upon the geometry and material of the scene as well as on the distribution of light sources.

Before going into the details of how to construct a model of image formation, we pause to consider a particular class of objects that we can use to manipulate how light propagates in space. This is crucial in building devices to capture images, and has to be taken into account to a certain extent in a mathematical model of image formation.

3.2.1 Imaging through lenses

An optical system is a set of lenses used to "direct" light. By directing light we mean a controlled change in the direction of propagation, which can be performed by means of diffraction, refraction and reflection. For the sake of simplicity, we neglect the effects of diffraction and reflection in a lens system, and we only consider refraction. Even so, a complete description of the functioning of a (purely refractive) lens is well beyond the scope of this book. Therefore, we will only consider the simplest possible model, that of a *thin lens* in a black box. For a more germane model of light propagation, the interested reader is referred to the classical textbook of Born and Wolf [**?**].

A thin lens is a mathematical model defined by an axis, called the *optical axis*, and a plane perpendicular to the axis, called the *focal plane*, with a circular aperture centered at the *optical center*, i.e. the intersection of the focal plane with the optical axis. The thin lens is characterized by one parameter, usually indicated by f, called the *focal length*, and by two functional properties. The first is that all rays entering the aperture parallel to the optical axis intersect on the optical axis at a distance f from the optical center. The point of intersection is called the *focus* of the lens (see Figure 3.4). The second property is that all rays through the optical center are undeflected. Now, consider a point $p \in \mathbb{E}^3$ not too far from the optical axis at a distance Z along the optical axis from the optical center. Now draw, from the point p, two rays: one parallel to the optical axis, and one

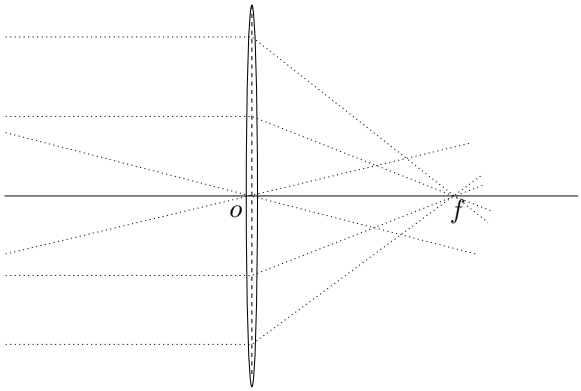

Figure 3.4. The rays parallel to the optical axis intersect at the focus.

through the optical center (Figure 3.5). The first one intersects the optical axis at the focus, the second remains undeflected (by the defining properties of the thin lens). Call $\mathbf{x}$ the point where the two rays intersect, and let z be its distance from the optical center. By decomposing any other ray from p into a component parallel to the optical axis and one through the optical center, we can argue that all rays from p intersect at $\mathbf{x}$ on the opposite side of the lens. In particular, a ray from $\mathbf{x}$ parallel to the optical axis, must go through p. Using similar triangles, from Figure 3.5 we obtain the following *fundamental equation of the thin lens*:

$$\boxed{\frac{1}{Z} + \frac{1}{z} = \frac{1}{f}}$$

The point $\mathbf{x}$ is called the *image* of the point p. Therefore, under the

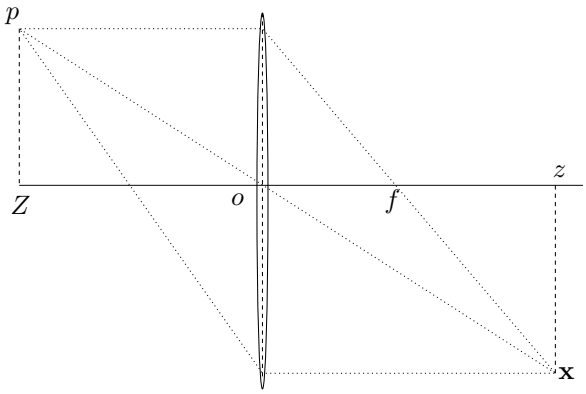

Figure 3.5. Image of the point p is the point $\mathbf{x}$ of the intersection of rays going parallel to the optical axis and the ray through the optical center.

assumption of a thin lens, the irradiance at the point $\mathbf{x}$ of coordinates (x, y) on the image plane is obtained by integrating all the energy emitted from the region of space contained in the cone determined by the geometry of the lens. If such a region does not contain light sources, but only opaque surfaces, it is necessary to assess how such surfaces radiate energy towards the sensor, which we do in Section 3.3. Before doing so, we introduce the notion of *"field of view"*, which is the angle subtended by the aperture of the lens seen from the focus. If D is the diameter of the lens, then the field of view is $\arctan(D/2f)$.

3.2.2 Imaging through pin-hole

If we let the aperture of a thin lens decrease to zero, all rays are forced to go through the optical center, and therefore they remain undeflected. Consequently, the aperture of the cone decreases to zero, and the only points that contribute to the irradiance at the points $\mathbf{x}$ on a line through p. If we let p have coordinates $\mathbf{X} = [X, Y, Z]^T$ relative to a reference frame centered at the optical center, with the optical axis being the Z-axis, then it is immediate to see from Figure 3.6 that the coordinates of $\mathbf{x}$ and p are related by an ideal *perspective projection*:

$$\boxed{x = -f\frac{X}{Z}, \quad y = -f\frac{Y}{Z}} \tag{3.2}$$

Note that any other point on the line through p projects onto the same coordinates $[x,\ y]^T$. This imaging model is the so-called *ideal pin-hole*. It

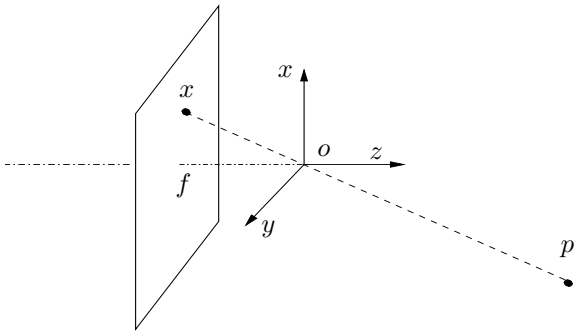

Figure 3.6. Image of the point p is the point $\mathbf{x}$ of the intersection of the ray going through the optical center o and an image plane at a distance f from the optical center.

is doubtlessly ideal since, when the aperture decreases, diffraction effects become dominant and therefore the (purely refractive) thin lens model does not hold. Furthermore, as the aperture decreases to zero, the energy going through the lens also becomes zero. The pin-hole is a purely geometric

model that approximates well-focused imaging systems. In this book we will use the pin-hole model as much as possible and concentrate on the geometric aspect of image-formation.

Notice that there is a negative sign in each of the formulae (3.2). This makes the image of an object appear to be upside down on the image plane (or the retina for the human eye). To eliminate this effect, we can simply flip the image: $(x, y) \mapsto (-x, -y)$. This corresponds to placing the image plane $\{Z = -f\}$ in front of the optical center instead: $\{Z = +f\}$.

3.3 A first model of image formation

In this section we go through the process of deriving a geometric model of image formation. We start by showing under what conditions one may study photometry by means of simple geometry. Section 3.3.1 may be skipped at a first reading. We then proceed to develop the purely geometric model that we will use for the remainder of the book in Section 3.3.2.

3.3.1 Basic photometry

Let S be a smooth surface patch in space; we indicate the tangent plane to the surface at a point p by τ and the inward unit normal vector by ν. At each $p \in S$ we can construct a local coordinate frame with origin at p, e_3-axis parallel to the normal vector ν and (e_1, e_2)-plane parallel to τ (see Figure 3.7). The change of coordinates between the reference point at p and an inertial reference frame (the "world" frame) is indicated by g_p; g_p maps points in the local coordinate frame at p into points in the inertial frame. For instance, g_p maps the origin o to the point p (with respect to the world frame): $g_p(o) = p$; and the vector e_3 into the surface normal ν (again with respect to the world frame): $g_{p*}(e_3) = \nu$.[1] In what follows we will not make a distinction between a change of coordinates g and its representation, and we will also consider interchangeably points $p \in \mathbb{E}^3$ and their representation $\mathbf{X} \in \mathbb{R}^3$.

Consider then a distribution of energy dE over a compact region of a surface in space L (the light source). For instance, L can be the hemisphere and dE be constant in the case of diffuse light on a cloudy day, or L could be a distant point and dE a delta-measure in the case of sunlight on a clear day (see Figure 3.7). The effects of the light source on the point $p \in S$ can be described using the infinitesimal energy $dE(\lambda_p)$ radiated from L to p

[1] We recall from the previous chapter that, if we represent the change of coordinates g with a rotation matrix $R \in SO(3)$ and a translation vector T, then the action of g on a point p of coordinates $\mathbf{X} \in \mathbb{R}^3$ is given by $g(p) \doteq R\mathbf{X} + T$, while the action of g on a *vector* of coordinates u is given by $g_*(u) \doteq Ru$.

along a vector λ_p. The total energy reaching p, assuming additivity of the energy transport, is $E(p) = \int_L dE(\lambda_p)$ which, of course, depends upon the point p in question. Note that there could be several light sources, including indirect ones (i.e. other objects reflecting energy onto S).

The portion of energy coming from a direction λ_p that is reflected onto a direction $\mathbf{x}_p$ is described by $\beta(\mathbf{x}_p, \lambda_p)$, the *bidirectional reflectance distribution function* (BRDF). The energy that p reflects onto $\mathbf{x}_p$ is therefore obtained by integrating the BRDF against the energy distribution

$$\mathcal{E}(\mathbf{x}_p, p) \doteq \int_L \beta(\mathbf{x}_p, \lambda_p) dE(\lambda_p) \tag{3.3}$$

which depends upon the direction $\mathbf{x}_p$ and the point $p \in S$, as well as on the energy distribution E of the light source L.

The geometry of the sensor is described by a central projection π (see Figure 3.7). For a point p with coordinates $\mathbf{X} \in \mathbb{R}^3$ expressed in the camera coordinate frame, which has its e_3-axis parallel to the optical axis and the (e_1, e_2)-plane parallel to the lens, the projection can be modeled as[2]

$$\pi : \mathbb{R}^3 \to \Omega; \quad \mathbf{X} \mapsto \mathbf{x} = \pi(\mathbf{X}) \tag{3.4}$$

where $\Omega \subset \mathbb{R}^2$ with $\pi(\mathbf{X}) \doteq \mathbf{X}/Z$ in the case of planar projection (e.g., onto the CCD), or $\Omega \subset \mathbf{S}^2$ with $\pi(\mathbf{X}) \doteq \mathbf{X}/\|\mathbf{X}\|$ in the case of a spherical projection (e.g., onto the retina). We will not make a distinction between the two models, and indicate the projection simply by π.

In order to express the direction $\mathbf{x}_p$ in the camera frame, we consider the change of coordinates from the local coordinate frame at the point p to the camera frame. For simplicity, we let the inertial frame coincide with the camera frame, so that $\mathbf{X} \doteq g_p(o) = p$ and $\mathbf{x} \sim g_{p_*}(\mathbf{x}_p)$ where[3] we note that g_{p_*} is a rotation, so that $\mathbf{x}$ depends on $\mathbf{x}_p$, while $\mathbf{X}$ depends on p. The reader should be aware that the transformation g_p itself depends on local shape of the surface at p, in particular its tangent plane τ and its normal ν at the point p. Once we substitute $\mathbf{x}$ for $\mathbf{x}_p$ into $\mathcal{E}$ in (3.3) we obtain the *radiance*

$$R_1(p) \doteq \mathcal{E}(g_{p_*}^{-1}(\mathbf{x}), p) \quad \text{where} \quad \mathbf{x} = \pi(p). \tag{3.5}$$

[2]We will systematically study the model of projection in next few sections, but what is given below suffices our discussions here.

[3]The symbol $\sim$ indicates projective equivalence, that is equality up to a scalar. Strictly speaking, $\mathbf{x}$ and $\mathbf{x}_p$ do not represent the same vector, but only the same direction (they have opposite sign and different lengths). However, they do represent the same point in the projective plane, and therefore we will regard them as one and the same. In order to obtain the same embedded representation (i.e. a vector in $\mathbb{R}^3$ with the same coordinates), we would have to write $\mathbf{x} = \pi(-g_{p_*}(\mathbf{x}_p))$. The same holds if we model the projective plane using the sphere with antipodal points identified.

Our (ideal) sensor can measure the amount of energy received along the direction $\mathbf{x}$, assuming a pin-hole model:

$$I_1(\mathbf{x}) = R_1(p) \quad \text{where} \quad \mathbf{x} = \pi(p). \tag{3.6}$$

If the optical system is not well modeled by a pin-hole, one would have to explicitly model the thin lens, and therefore integrate not just along the direction $\mathbf{x}_p$, but along all directions in the cone determined by the current point and the geometry of the lens. For simplicity, we restrict our attention to the pin-hole model.

Notice that R_1 in (3.5) depends upon the shape of the surface S, represented by its location p and surface normal ν, but it also depends upon the light source L, its energy distribution E and the reflectance properties of the surface S, represented by the BRDF β. Making this dependency explicit we write

$$I_1(\mathbf{x}) = \int_L \beta(g_{p_*}^{-1}(\mathbf{x}), \lambda_p) dE(\lambda_p) \quad \text{where} \quad \mathbf{x} = \pi(p) \tag{3.7}$$

which we indicate in short-hand notation as $I_1(\mathbf{x}) = R_1(p; \nu, \beta, L, E)$ where we emphasize the dependence on ν, β, L, E in addition to p. I_1 is called the (image) *irradiance*, while R_1 is called (scene) *radiance*.

When images are taken from different viewpoints, one has to consider the change of coordinates g relative to the inertial reference frame. Assuming that the inertial frame coincides with the image I_1, we can obtain a new image I_2 by moving with g (see Figure 3.7). The coordinates of the point p in the first and second camera frames are related by $p_2 = g(p)$, and $\mathbf{x} = \pi(p)$, $\mathbf{x}_2 = \pi(p_2) = \pi(g(p))$. Using the fact that $\mathbf{x}_2 = \pi(g(p)) = \pi(g(g_p(o)))$, we have that the second image takes the light (reflected from p) in the direction $\mathbf{x}_{p2} \sim (gg_{p_*})^{-1}(\mathbf{x}_2)$ (with respect to the local coordinate frame at p). Therefore, the (scene) radiance in the direction of the new viewpoint is given by

$$R_2(p, g; \nu, \beta, L, E) \doteq \mathcal{E}((gg_{p_*})^{-1}(\mathbf{x}_2), p)$$

and the (image) *irradiance* is $I_2(\mathbf{x}_2) = R_2(p, g; \nu, \beta, L, E)$ where $\mathbf{x}_2 \doteq \pi(g(p))$. So far, we have used the first image as a reference, so that $\mathbf{x}_1 = \mathbf{x}$, g_1 is the identity transformation and $p_1 = p$. This needs not be the case. If we choose an arbitrary inertial reference and indicate with g_k the change of coordinates to the reference of camera k, p_k the coordinate of the point p in this frame and $\mathbf{x}_k$ the corresponding direction, then for each of $k = 1 \dots M$ images we can write $I_k(\mathbf{x}_k) = R_k(p, g_k; \nu, \beta, L, E)$.

The irradiance I_k of image k can be measured only up to noise. Since we assume that energy transport phenomena are additive, such a noise will be additive, but will have to satisfy the constraint that both radiance and

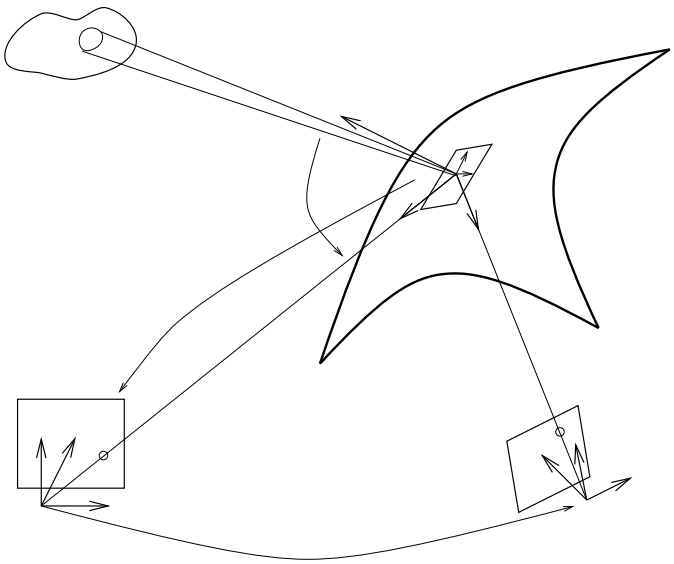

Figure 3.7. **Generative model**

irradiance must be *positive*

$$\begin{cases} I_k(\mathbf{x}_k) = R_k(p, g_k; \nu, \beta, L, E) + n_k(\mathbf{x}_k) \\ \text{subject to} \quad \mathbf{x}_k = \pi(g_k(p)) \ \text{ and } \ R_k \geq 0 \end{cases} \tag{3.8}$$

for $k = 1 \dots M$.

This model can be considerably simplified if we restrict our attention to a class of materials, called *Lambertian*, that do not change appearance depending on the viewpoint. Marble and other matte surfaces are to a large extent well approximated by the Lambertian model. Metal, mirrors and other shiny surfaces are not.

According to Lambert's model, the BRDF only depends on how the surface faces the light source, but not on how it is viewed. Therefore, $\beta(\mathbf{x}_p, \lambda_p)$ is actually independent of $\mathbf{x}_p$, and we can think of the radiance function as being "glued", or "painted" on the surface S, so that at each point p we have that the radiance R only depends on the geometry of the surface, and not explicitly on the light source. In particular, we have $\beta(\mathbf{x}_p, \lambda_p) = \langle \lambda_p, \nu_p \rangle$, independent of $\mathbf{x}_p$, and $R(p, \nu_p) \doteq \int_L \langle \lambda_p, \nu_p \rangle dE(\lambda_p)$. Since ν_p is the normal vector, which is determined by the geometry of the surface at p, knowing the position of the generic point $p \in S$ one can differentiate it to compute the tangent plane. Therefore, effectively, the radiance R only depends on the surface S, described by its generic point p:

$$I(\mathbf{x}) = R(p) \tag{3.9}$$

where $\mathbf{x} = \pi(g(p))$ relative to the camera reference frame. In all subsequent sections we will adopt this model. In the next section we will show how to relate this model, expressed in a very particular camera reference frame (centered in the optical center, with Z-axis along the optical axis etc.) to a general world reference frame.

3.3.2 The basic model of imaging geometry

Under the Lambertian assumption, as we have seen in the previous section, we can reduce the process of image formation to tracing rays from objects to pixels. The essence of this process is captured by the geometry of perspective projection, which can be reduced to simple changes of coordinates and a canonical projection. In order to establish a relationship between the position of points in the world, expressed for instance with respect to an inertial reference frame, and the points measured in an image, it is necessary to describe the relation between the camera reference frame and the world frame. In this section we describe such a relation in detail for this simplified image formation model as a series of transformations of coordinates. Inverting such a chain of transformations is the task of *"camera calibration"*, which is the subject of Chapter 6.

Consider an orthogonal inertial reference frame $\{o, X, Y, Z\}$, called the *world* frame. We write the coordinates of a point p in the world frame as

$$\mathbf{X}_w = [X_w, Y_w, Z_w]^T \in \mathbb{R}^3.$$

In order to write the coordinates of the same point with respect to another reference frame, for instance the camera frame c, $\mathbf{X}_c$, it is necessary to describe the transformation between the world frame and the camera frame. We will indicate such a transformation by g_{wc}. With this notation, the coordinates in the world frame and those in the camera frame are related by:

$$\mathbf{X}_w = g_{wc}(\mathbf{X}_c)$$

and vice-versa $\mathbf{X}_c = g_{cw}\mathbf{X}_w$ where $g_{cw} = g_{wc}^{-1}$ is the inverse transformation, which maps coordinates in the world frame onto coordinates in the camera frame.

Transformations of coordinates between orthonormal reference frames are characterized by the rigid motion of the reference frame. Therefore, as discussed in Chapter 2, a rigid change of coordinates is described by the position of the origin of the new frame, and the orientation of the new frame relative to the old frame. Consider now the change of coordinates g_{wc}:

$$g_{wc} = (R_{wc}, T_{wc})$$

where $T_{wc} \in \mathbb{R}^3$ and R_{wc} is a rotation matrix. The change of coordinates of an arbitrary point is given by

$$\mathbf{X}_w = g_{wc}(\mathbf{X}_c) = R_{wc}\mathbf{X}_c + T_{wc}.$$

A more compact way of writing the above equation is to use the homogeneous representation,

$$\bar{\mathbf{X}} \doteq [\mathbf{X}^T, 1]^T \in \mathbb{R}^4, \quad \bar{g}_{wc} = \begin{bmatrix} R_{wc} & T_{wc} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}$$

so that

$$\bar{\mathbf{X}}_w \doteq \bar{g}_{wc}\bar{\mathbf{X}}_c = \begin{bmatrix} R_{wc} & T_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X}_c \\ 1 \end{bmatrix}.$$

The inverse transformation, that maps coordinates in the world frame onto coordinates in the camera frame, is given by

$$g_{cw} = (R_{cw}, T_{cw}) = (-R_{wc}^T T_{wc}, R_{wc}^T)$$

as it can be easily verified by:

$$\bar{g}_{cw}\bar{g}_{wc} = \bar{g}_{wc}\bar{g}_{cw} = I_{4\times 4}$$

which is the identity transformation, and therefore $\bar{g}_{cw} = \bar{g}_{wc}^{-1}$.

3.3.3 Ideal camera

Let us consider a generic point p, with coordinates $\mathbf{X}_w \in \mathbb{R}^3$ relative to the world reference frame. As we have just seen, the coordinates relative to the camera reference frame are given by $\mathbf{X}_c = R_{cw}\mathbf{X}_w + T_{cw} \in \mathbb{R}^3$. So far we have used subscripts to avoid confusion. From now on, for simplicity, we drop them by using the convention $R = R_{cw}$, $T = T_{cw}$ and $g = (R, T)$. In the following section we denote the coordinates of a point relative to the camera frame by $\mathbf{X} = [X, Y, Z]^T$ and the coordinates of the same point relative to the world frame by $\mathbf{X}_0$. $\mathbf{X}_0$ is also used to denote the coordinates of the point relative to the initial location of the camera moving frame. This is done often for convenience, since the choice of world frame is arbitrary and it is convenient to identify it with the camera frame at a particular time. Using this convention we have:

$$\mathbf{X} = g(\mathbf{X}_0) = R\mathbf{X}_0 + T. \tag{3.10}$$

Attaching a coordinate frame to the center of projection, with the optical axis aligned with the z-axis and adopting the ideal pin-hole camera model of Section 3.2.2 (see also Figure 3.6) the point of coordinates $\mathbf{X}$ is projected onto the point of coordinates

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} = -\frac{f}{Z}\begin{bmatrix} X \\ Y \end{bmatrix}.$$

where $[x, y]^T$ are coordinates expressed in a two-dimensional reference frame (the *retinal image frame*) centered at the principal point (the intersection between the optical axis and the image plane) with the x-axis and y-axis parallel to the X-axis and Y-axis respectively and f represents the focal length corresponding to the distance of the image plane from the center of projection. In homogeneous coordinates this relationship can be written as:

$$Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{3.11}$$

where $\bar{\mathbf{X}} \doteq [X, Y, Z, 1]^T$ is the representation of a 3-D point in homogeneous coordinates and $\bar{\mathbf{x}} \doteq [x, y, 1]^T$ are homogeneous (projective) coordinates of the point in the retinal plane. Since the Z-coordinate (or the depth of the point p) is usually unknown, we may simply denote it as an arbitrary positive scalar $\lambda \in \mathbb{R}_+$. Also notice that in the above equation we can decompose the matrix into

$$\begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Define two matrices:

$$A_f = \begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 4}.$$

Also notice that from the coordinate transformation we have for $\bar{\mathbf{X}} = [X, Y, Z, 1]^T$:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}. \tag{3.12}$$

To summarize, using the above notation, the geometric model for *an ideal camera* can be described as:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix},$$

or in matrix form

$$\boxed{\lambda \bar{\mathbf{x}} = A_f P \bar{\mathbf{X}} = A_f P \bar{g} \bar{\mathbf{X}}_0} \tag{3.13}$$

3.3.4 Camera with intrinsic parameters

The idea model of Equation (3.13) is specified relative to a very particular choice of reference frame, with Z axis along the optical axis etc. In practice, when one captures images with a digital camera, the coordinates of the optical axis or the optical center are not known, and neither are the units of measurements. Instead, one can typically specify the index (i, j) of a particular pixel, relative to the top-left corner of the image. Therefore, in order to render the model (3.13) usable, we need to specify its relationship with the pixel array.

The first step consists of specifying the units along the x and y axes: if x and y are specified in terms of metric units (e.g., millimeters), and x_s, y_x are scaled version that correspond to the index of a particular pixel, then the transformation from $\mathbf{x}$ coordinates to $\mathbf{x}_s$ coordinates can be described by a scaling matrix

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.14}$$

that depends on the size of the pixel (in metric units) along the x and y directions. In general, these may be different (i.e. the pixels can be rectangular, rather than square). However, x_s and y_s are still specified relative to the point of intersection of the optical axis with the image plane (also called the *principal point*), whereas the pixel index (i, j) is usually specified relative to the top-left corner, and is conventionally indicated by a positive number. Therefore, we need to translate the origin of the reference frame and invert its axis:

$$\begin{aligned} x_{im} &= -(x_s - o_x) \\ y_{im} &= -(y_s - o_y) \end{aligned}$$

where (o_x, o_y) are the coordinates (in pixels) of the principal point relative to the image reference frame, or where the Z-axis intersects the image plane. So the actual image coordinates are given by the vector $\mathbf{x}_{im} = [x_{im}, y_{im}]^T \in \mathbb{R}^2$ instead of the ideal image coordinates $\mathbf{x} = [x, y]^T$. The above steps can be written in a homogeneous coordinates in a matrix form in the following way:

$$\bar{\mathbf{x}}_{im} \doteq \begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \begin{bmatrix} -s_x & 0 & o_x \\ 0 & -s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.15}$$

where x_{im} and y_{im} are actual image coordinates in pixels. When $s_x = s_y$ the pixels are square. In case the pixels are not rectangular, a more general form of the scaling matrix can be considered:

$$S = \begin{bmatrix} s_x & s_\theta \\ 0 & s_y \end{bmatrix} \in \mathbb{R}^{2\times 2}$$

where s_θ is proportional to $\cot(\theta)$ of the angle between the image axes x and y. So then the transformation matrix in (3.15) takes the general form:

$$A_s = \begin{bmatrix} -s_x & -s_\theta & o_x \\ 0 & -s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}. \tag{3.16}$$

In many practical applications it is commonly assumed that $s_\theta = 0$.
In addition to the change of coordinate transformations, in the case of a large field of view, one can often observe image distortions along radial directions. The radial distortion effect is typically modeled as:

$$\begin{aligned} x &= x_d(1 + a_1 r^2 + a_2 r^4) \\ y &= y_d(1 + a_1 r^2 + a_2 r^4) \end{aligned}$$

where (x_d, y_d) are coordinates of the distorted points, $r^2 = x_d^2 + y_d^2$ and a_1, a_2 are then considered additional camera parameters. However, for simplicity, in this book we will assume that radial distortion has been compensated for. The reader can refer to [**?**] for details.
Now, combining the projection model from the previous section with the scaling and translation yields a more realistic model of a transformation between homogeneous coordinates of a 3-D point relative to the camera frame and homogeneous coordinates of its image expressed in terms of pixels:

$$\lambda \begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \begin{bmatrix} -s_x & -s_\theta & o_x \\ 0 & -s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

Notice that in the above equation, the effect of a real camera is in fact carried through two stages. The first stage is a standard perspective projection with respect to a *normalized coordinate system* (as if the focal length $f = 1$). This is characterized by the standard projection matrix $P = [I_{3\times 3}, 0]$. The second stage is an additional transformation (on the so obtained image $\mathbf{x}$) which depends on parameters of the camera such as the focal length f, the scaling factors s_x, s_y and s_θ and the center offsets o_x, o_y. The second transformation is obviously characterized by the combination of the two matrices A_s and A_f

$$A = A_s A_f = \begin{bmatrix} -s_x & -s_\theta & o_x \\ 0 & -s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Such a decoupling allows us to write the projection equation in the following way:

$$\lambda \bar{\mathbf{x}}_{im} = AP\bar{\mathbf{X}} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \quad (3.17)$$

The 3×3 matrix A collects all parameters that are "intrinsic" to the particular camera, and are therefore called *intrinsic parameters*; the matrix P represents the perspective projection. The matrix A is usually called the *intrinsic parameter matrix* or simply the *calibration matrix* of the camera. When A is known, the normalized coordinates $\mathbf{x}$ can be obtained from the pixel coordinates $\mathbf{x}_{im}$ by simple inversion of A:

$$\lambda \bar{\mathbf{x}} = \lambda A^{-1} \bar{\mathbf{x}}_{im} = P\bar{\mathbf{X}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \quad (3.18)$$

The information about the matrix A can be obtained through the process of camera calibration described in Chapter 6.

The normalized coordinate system corresponds to the ideal pinhole camera model with the image plane located in front of the center of projection and the focal length f equal to 1. Given this geometric interpretation, the individual entries of the matrix A correspond to:

$1/s_x$ size of the horizontal pixel in meters [m/pixel],
$1/s_y$ size of the vertical pixel in meters [m/pixel],
$\alpha_x = s_x f$ size of the focal length in horizontal pixels [pixel],
$\alpha_y = s_y f$ size of the focal length in vertical pixels [pixel],
α_x/α_y aspect ratio σ.

To summarize, the overall geometric relationship between 3-D coordinates $\mathbf{X}_0 = [X_0, Y_0, Z_0]^T$ relative to the world frame and their corresponding image coordinates $\mathbf{x}_{im} = [x_{im}, y_{im}]^T$ (in pixels) depends on the rigid body displacement between the camera frame and world frame (also called *extrinsic calibration parameters*), an ideal projection and the camera intrinsic parameters. The overall geometry of the image formation model is therefore captured by the following equation:

$$\lambda \begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix},$$

or in matrix form

$$\boxed{\lambda \bar{\mathbf{x}}_{im} = AP\bar{\mathbf{X}} = AP\bar{g}\bar{\mathbf{X}}_0} \quad (3.19)$$

3.3.5 Spherical projection

The pin-hole camera model outlined in the previous section considers planar imaging surfaces. An alternative imaging surface which is commonly considered is that of a sphere. This choice is partly motivated by retina shapes often encountered in biological systems. For spherical projection, we simply choose the imaging surface to be the unit sphere: $\mathbb{S}^2 = \left\{\mathbf{X} \in \mathbb{R}^3 \mid \|\mathbf{X}\| = 1\right\}$. Then, the spherical projection is defined by the map π_s from $\mathbb{R}^3$ to $\mathbb{S}^2$:

$$\pi_s : \mathbb{R}^3 \to \mathbb{S}^2, \quad \mathbf{X} \mapsto \mathbf{X} = \frac{\mathbf{X}}{\|\mathbf{X}\|}.$$

Similarly to the case of perspective projection, the relationship between the coordinates of 3-D points and their image projections can be expressed as

$$\lambda \bar{\mathbf{x}}_{im} = AP\bar{\mathbf{X}} = AP\bar{g}\bar{\mathbf{X}}_0 \tag{3.20}$$

where the scale $\lambda = Z$ in case of perspective projection becomes $\lambda = \sqrt{X^2 + Y^2 + Z^2}$ in case of spherical projection. Therefore, mathematically, the perspective projection and spherical projection are exactly equivalent to each other. The only difference is the unknown scale λ takes different values.

3.3.6 Approximate camera models

The most commonly used approximation to the perspective projection model is the so called *orthographic projection*. The light rays in the orthographic model travel along the lines parallel to the optical axis. The relationship between image points and 3-D points in this case is particularly simple: $x = X$; $y = Y$. So the geometric model for an *"orthographic camera"* can be expressed as:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{3.21}$$

or simply in a matrix form

$$\boxed{\mathbf{x} = P_2\mathbf{X}} \tag{3.22}$$

where $P_2 = [I_{2\times 2}, 0] \in \mathbb{R}^{2\times 3}$.

Orthographic projection is a good approximation to perspective projection when the variation of the depth between the viewed points is much smaller then the distance of the points from the image plane. In case the points viewed lie on a plane which is parallel to the image plane, the image of the points is essentially a scaled version of the original. This scaling can be explicitly incorporated into the orthographic projection model, leading to so called *weak-perspective* model. In such a case, the relationship between

image points and 3-D points is:

$$x = f\frac{X}{\bar{Z}}, \quad y = f\frac{Y}{\bar{Z}}$$

where $\bar{Z}$ is the average distance of the points viewed by the camera. This model is appropriate for the case when all points lie in the frontoparallel plane then the scaling factor corresponds to the distance of the plane from the origin. Denoting the scaling factor $s = \frac{f}{\bar{Z}}$ we can express the *weak-perspective camera model* (scaled orthographic) as:

$$\begin{bmatrix} x \\ y \end{bmatrix} = s \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.23}$$

or simply in a matrix form

$$\boxed{\mathbf{x} = sP_2\mathbf{X}} \tag{3.24}$$

These approximate projection models often lead to simplified and efficient algorithms for estimation of unknown structure and displacement of the cameras and are especially suitable for applications when the assumptions of the model are satisfied [**?**].

3.4 Summary

The perspective projection was introduced as a model of the image formation process. In the case of idealized projection (when the intrinsic parameters of the camera system are known) the coordinates of the image points are related to their 3D counterparts by an unknown scale λ

$$\lambda\bar{\mathbf{x}} = \bar{\mathbf{X}}.$$

In the absence of camera parameters the perspective projection has been augmented by an additional transformation A which captures the intrinsic parameters of the camera system and yields following relationship between image quantities and its 3D counterparts.

$$\lambda\bar{\mathbf{x}}_{im} = AP\bar{\mathbf{X}} = AP\bar{g}\bar{\mathbf{X}}_0$$

3.5 Exercises

1. Show that any point on a line through p projects onto the same coordinates (equation (3.4)).
2. Show how perspective projection approximates orthographic projection when the scene occupies a volume whose diameter is small

compared to its distance from the camera. Characterize other conditions under which the two projection models produce similar results (equal in the limit).

3. Consider a thin round lens imaging a plane parallel to the lens at a distance d from the focal plane. Determine the region of this plane that contributes to the image I at the point $\mathbf{x}$. (Hint: consider first a one-dimensional imaging model, then extend to a two-dimensional image).

4. **Perspective projection vs. orthographic projection**
It is common sense that, with a perspective camera, one can not tell an object from another object which is exactly *twice as big but twice as far*. This is a classic ambiguity introduced by perspective projection. Please use the ideal camera model to explain why this is true. Is the same also true for orthographic projection? Please explain.

5. **Vanishing points**
A straight line in the 3-D world is projected onto a straight line in the image. The projections of two parallel lines intersect in the image at so-called *vanishing point*.

   a) Show that projections of parallel lines in the image intersect at a point.
   b) Compute, for a given family of parallel lines, where in the image the vanishing point will be.
   c) When does the vanishing point of the lines in the image lie at infinity (i.e. they do not intersect)?

6. **Field of view**
An important parameter of the imaging system is the *field of view* (FOV). The field of view is twice the angle between the optical axis (z-axis) and the end of the retinal plane (CCD array). Imagine that you have a camera system with focal length 16mm, and retinal plane (CCD array) is $(16mm \times 12mm)$ and that you digitizer samples your imaging surface 500×500 pixels in each direction.

   a) Compute the FOV.
   b) Write down the relationship between the image coordinate and a point in 3D world expressed in the camera coordinate system.
   d) Describe how is the size of FOV related to focal length and how it affects the resolution in the image.
   e) Write a MATLAB program, which simulates the geometry of the projection process; given an object (3D coordinates of the object in the camera (or world) frame, create an image of that object. Experiment with changing the parameters of the imaging system.

7. **Calibration matrix**
   Please compute the calibration matrix A which represents the transformation from image I to I' as shown in Figure 3.8. Note that, from the definition of calibration matrix, you need to use homogeneous coordinates to represent points on the images. Suppose that the resulting image I' is further digitized into an array of 480×640 pixels and the intensity value of each pixel is quantized to an integer in $[0, 255]$. Then how many *different* digitized images one can possibly get from such a process?

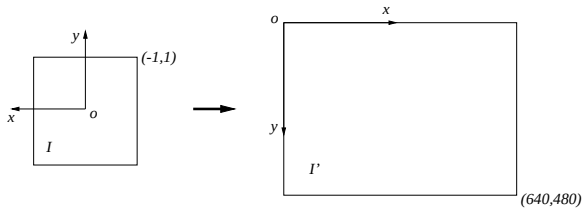

Figure 3.8. Transformation of a normalized image into pixel coordinates.

Chapter 4
Image primitives and correspondence

The previous chapter described how images are formed depending on the position of the camera, the geometry of the environment and the light distribution. In this chapter we begin our journey of understanding how to "undo" this process and process measurements of light in order to infer the geometry of the environment.

The geometry of the environment and the light distribution combine to give an image in a way that is inextricable: from a single image, in lack of prior information, we cannot tell light and geometry apart. One way to see that is to think of an image as coming from a certain object with a certain light distribution: one can always construct a different object and a different light distribution that would give rise to the same image. One example is the image itself! It is an object different from the true scene (it is flat) that gives rise to the same image (itself). In this sense, images are "illusions:" they are objects different than the true one, but that generate on the retina of the viewer the same stimulus (or "almost" the same, as we will see) that the original scene would.

However, if instead of looking at one single image we look at two or more images of the same scene taken for instance from different vantage points, then we can hope to be able to extract geometric information about the scene. In fact, each image depends upon the (changing) camera position, but they all depend upon the same geometry and light distribution, assuming that the scene has not changed between snapshots. Consider, for instance, the case of a scene and an photograph of the scene. When viewed from different viewpoints, the original scene will appear different depending upon the depth of objects within: distant objects will move little and

near objects will move mode. If, on the other hand, we move in front of a photograph of the scene, every points will have the same image motion.

This intuitive discussion suggests that, in order to be able to extract geometric information about the environment, we first have to be able to establish how points "move" from one image to the next. Or, more in general, we have to establish "which points corresponds to which" in different images of the same scene. This is called the *correspondence problem*. The correspondence problem is at the heart of the process of converting measurements of light into estimates of geometry.

In its full generality, this problem cannot be solved. If the light distribution and the geometry of the scene can change arbitrarily from one image to the next, there is no way to establish which points corresponds to which in different images. For instance, if we take a white marble sphere rotating, we have no way to tell which point corresponds to which on the sphere since each point looks the same. If instead we take a still mirror sphere and move the light, even though points are still (and therefore points correspond to themselves), their appearance changes from one image to the next. Even if light and geometry do not change, if objects have anisotropic reflectance properties, so that they change their appearance depending on the viewpoint, establishing correspondence can be very hard.

In this chapter we will study under what conditions the correspondence problem can be solved, and can be solved easily. We will start with the most naive approach, that consists in considering the image irradiance $I(\mathbf{x})$ as a "label" attached to each pixel $\mathbf{x}$. Although sound in principle, this approach leads to nowhere due to a variety of factors. We will then see how to extend the "label" idea to more and more complex labels, and what assumptions on the shape of objects different labels impose.

4.1 Correspondence between images

Given a projection of a 3D point in an image, the goal of establishing the correspondence, is to find the projection of the same 3D point in another view. In order to understand and tackle this difficult problem, we need to relate the abstract notion of a "point" to some measurable image quantities.

Consider one image, denoted by I_1, represented as a function defined on a compact two-dimensional region Ω (the sensor) taking values (irradiance) in the positive reals, as we discussed in the previous chapter:

$$\begin{aligned} I_1 : \Omega \subset \mathbb{R}^2 &\rightarrow \mathbb{R}_+ \\ \mathbf{x} &\mapsto I(\mathbf{x}). \end{aligned}$$

Under the simplifying assumptions of Chapter 3, the irradiance $I_1(\mathbf{x})$ is obtained by integrating the radiant energy in space along the ray $\lambda\mathbf{x}$. If the scene only contains opaque objects, then only one point along the projection

ray contributes to the irradiance. This point has coordinates $\mathbf{X} \in \mathbb{R}^3$, corresponding to a particular value of λ determined by the first intersection of the ray with a visible surface: $\lambda \mathbf{x} = \mathbf{X}$. Therefore, $I_1(\mathbf{x}) = E(\lambda \mathbf{x}) = E(\mathbf{X})$, where $\lambda \in \mathbb{R}_+$ represents the distance of the point along the projection ray and $E(\mathbf{X}) : \mathbb{R}^3 \longrightarrow \mathbb{R}_+$ is the radiant distribution of the object at $\mathbf{X}$. This is the fundamental *irradiance equation*:

$$I_1(\mathbf{x}) = E(\lambda \mathbf{x}) = E(\mathbf{X}). \tag{4.1}$$

Now suppose that a different image of the same scene becomes available, I_2, for instance one taken from a different vantage point. Naturally,

$$I_2 : \Omega \subset \mathbb{R}^2 \longrightarrow \mathbb{R}_+; \mathbf{x} \mapsto I_2(\mathbf{x}).$$

However, I_2 will in general be different from I_1. The first step in establishing correspondence is to understand how such a difference occurs. We distinguish between changes that occur in the image domain Ω and changes that occur in the value I_2.

4.1.1 Transformations in the image domain

Let us assume, for now, that we are imaging empty space except for a point with coordinates $\mathbf{X} \in \mathbb{R}^3$ that emits light with the same energy in all directions. This is a simplifying assumption that we will relax in the next section. If I_1 and I_2 are images of the same scene, they must satisfy the same irradiance equation (4.1). In general, however, the projection rays from the point $\mathbf{X}$ onto the two sensors will be different in the two images, due to the difference in the viewpoints, which can be characterized by as a displacement $(R, T) \in SE(3)$:

$$I_2(\mathbf{x}_2) = E(\mathbf{x}_2 \lambda_2) = E(\mathbf{X}) = E(R\mathbf{x}_1 \lambda_1 + T) \doteq I_1(\mathbf{x}_1) \tag{4.2}$$

Under these admittedly restrictive assumptions, the correspondence problem consists of establishing the relationship between $\mathbf{x}_1$ and $\mathbf{x}_2$. One may think of I_1 and I_2 as "labels" associated to $\mathbf{x}_1$ and $\mathbf{x}_2$, and therefore match points $\mathbf{x}_i$ based on their labels I_i. Equation (4.2) therefore shows that the point $\mathbf{x}_1$ in image I_1 corresponds to the point $\mathbf{x}_2$ in image I_2 if

$$\mathbf{x}_2 = h(\mathbf{x}_1) = (R\mathbf{x}_1 \lambda_1(\mathbf{X}) + T)/\lambda_2(\mathbf{X}) \tag{4.3}$$

where we have emphasized the fact that the scales $\lambda_i,\ i = 1, 2$ depend upon the coordinates of the point in space $\mathbf{X}$. Therefore, under the admittedly restrictive conditions of this section, a model of the deformation between two images of the same scene is given by:

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1)). \tag{4.4}$$

The function h describes the "image motion" that we have described informally in the header of this chapter. In order to make it more suggestive

of the motion of individual pixels, we could write h as

$$h(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{X}) \tag{4.5}$$

where the fact that h depends upon the shape of the scene is made explicitly in the argument of $\mathbf{u}$. Note that the dependence of $h(\mathbf{x})$ on the position of the point $\mathbf{X}$ comes through the scales λ_1, λ_2. In general, therefore, h is a function in an infinite-dimensional space (the space of surfaces), and solving for image correspondence is as difficult as estimating the shape of visible objects.

While the reader reflects on how inappropriate it is to use the brightness value of the image at a point as a "label" to match pixels across different images, we reflect on the form of the image motion h in some interesting special cases. We leave it as an exercise to the reader to prove the statements below.

Translational image model Consider the case where each image point moves with the same motion, $\mathbf{u}(\mathbf{X}) = \text{const}$, or $h(\mathbf{x}) = \mathbf{x} + \mathbf{u} \; \forall \; \mathbf{x} \in \Omega$ where $\mathbf{u} \in \mathbb{R}^2$. This model is correct only if the scene is flat and parallel to the plane, far enough from it, and the viewer is moving slowly in a direction parallel to the image. However, if instead of considering the whole image we consider a small window around a point $\mathbf{x}$, $W(\mathbf{x}) = \{\tilde{\mathbf{x}} \mid \|\mathbf{x} - \tilde{\mathbf{x}}\| \leq \epsilon\}$, then a simple translational model could be a reasonable approximation provided that the window is not an image of an occluding boundary (why?). Correspondence based on this model is at the core of most optical flow and feature tracking algorithms. We will adopt the one of Lucas and Kanade [**?**] as the prototypical example of a model in this class.

Affine image deformation Consider the case where $h(\mathbf{x}) = A\mathbf{x} + \mathbf{d}$ where $A \in \mathbb{R}^{2\times 2}$ and $\mathbf{d} \in \mathbb{R}^2$. This model is a good approximation for small planar patches parallel to the image plane moving under an arbitrary translation and rotation about the optical axis, and modest rotation about an axis parallel to the image plane. Correspondence based on this model has been addressed by Shi and Tomasi [**?**].

Now, let us see why attaching scalar labels to pixels is not a good idea.

4.1.2 Transformations of the intensity value

The basic assumption underlying the derivation in the previous section is that each point with coordinates $\mathbf{X}$ results in the same measured irradiance in both images, as in Equation (4.2). In practice, this assumption is unrealistic due to a variety of factors.

First, measuring light intensity consists of counting photons, a process that is intrinsically subject to uncertainty. In addition, the presence of the atmosphere, the material of the lens, the manufacturing of the sensor

all inject noise in the reading of light intensity. As a first approximation, one could lump all sources of uncertainty into an additive noise term n. This noise is often described statistically as a Poisson random variable (when emphasizing the nature of the counting process and enforcing the non-negativity constraints on irradiance), or as a Gaussian random variable (when emphasizing the concurrence of multiple independent sources of uncertainty). Therefore, equation (4.4) must be modified to take into account changes in the intensity value, in addition to the deformation of the domain:

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1)) + n(h(\mathbf{x}_1)). \tag{4.6}$$

More fundamental departures from the model (4.4) occur when one considers that points that are visible from one vantage point may become occluded from another. *Occlusions* could be represented by a factor multiplying I_2 that depends upon the shape of the surface being imaged:

$$I_1(\mathbf{x}_1) = f_o(\mathbf{X}, \mathbf{x}) I_2(h(\mathbf{x}_1)) + n(h(\mathbf{x}_1)). \tag{4.7}$$

For instance, for the case where only one point on the surface is emitting light, $f_o(\mathbf{X}, \mathbf{x}) = 1$ when the point is visible, and $f_o(\mathbf{X}, \mathbf{x}) = 0$ when not. The equation above should make very clear the fact that associating the label I_1 to the point $\mathbf{x}_1$ is not a good idea, since the value of I_1 depends upon the noise n and the shape of the surfaces in space $\mathbf{X}$, which we cannot control.

There is more: in most natural scenes, object do not emit light of their own but, rather, they reflect ambient light in a way that depends upon the properties of the material. Even in the absence of occlusions, different materials may scatter or reflect light by different amounts in different directions. A limit case consist of material that scatters light uniformly in all directions, such as marble and opaque matte surfaces, for which $f_o(\mathbf{X}, \mathbf{x})$ could simply be multiplied by a factor $\alpha(\mathbf{X})$ independent of the viewpoint $\mathbf{x}$. Such materials are called *Lambertian* and the function $\alpha(\mathbf{X})$ defined on the surface of visible objects is called *albedo*. However, for many other surfaces, for instance the other extreme case of a perfectly *specular* material, light rays hitting the surface are reflected by different amounts in different directions. Images of specular objects depend upon the light distribution of the ambient space, which includes light sources as well as every other object through inter-reflections. In general, few materials exhibit perfect Lambertian or specular reflection, and even more complex reflection models, such as translucent or anisotropic materials, are commonplace in natural and man-made scenes.

The reader can therefore appreciate that the situation can get very complicated even for relatively simple objects. Consider, for instance, the example of the marble sphere and the mirror sphere in the header of this chapter. Indeed, in the most general case an image depends upon the (unknown) light distribution, material properties, shape and pose of objects in

space, and images alone – no matter how many – do not provide sufficient information to recover all unknowns.

So how do we proceed? How can we establish correspondence despite the fact that labels are ambiguous and that point correspondence cannot be established for general scenes?

The second question can be easily dismissed by virtue of modesty: in order to recover a model of the scene, it is not necessary to establish the correspondence for every single point on a collection of images. Rather, in order to recover the pose of cameras and a simple model of an environment, correspondence of a few fiducial points is sufficient. It will be particularly clever if we could choose such fiducial points, of "features" in such a way as to guarantee that they are easily put in correspondence.

The first question can be addressed by proceeding as often when trying to counteract the effects of noise: integrating. Instead of considering Equation (4.2) in terms of points on an image, we can consider it defining correspondence in terms of *regions*. This can be done by integrating each side on a window $W(\mathbf{x})$ around each point $\mathbf{x}$, and using the equation to characterize the correspondence at $\mathbf{x}$. Another way of thinking of the same procedure is to associate to each pixel $\mathbf{x}$ not just the scalar label $I(\mathbf{x})$, but instead a "vector label"

$$l(\mathbf{x}) \doteq \{I(\tilde{\mathbf{x}}) \mid \tilde{\mathbf{x}} \in W(\mathbf{x}) \subset \Omega\}.$$

This way, rather than trying to match values, in order to establish correspondence, we will need to match vectors, or "regions."

Of course, if we considered the transformation of each point in the region as independent, $h(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{X})$, we would be back to ground zero, having to figure out two unknowns for each pixel. However, if we make the assumption that the whole region $W(\mathbf{x})$ undergoes the same motion, for instance $h(\tilde{\mathbf{x}}) = \tilde{\mathbf{x}} + \mathbf{u} \ \forall \ \tilde{\mathbf{x}} \in W(\mathbf{x})$, then the problem may become easier to solve. At this stage the reader should go back to consider under what assumptions a window undergoes a pure translation, and when it may be more appropriate to model the motion of the region as an affine deformation, or when neither approximation is appropriate.

In the remainder of this chapter we will explore in greater detail how to match regions based on simple motion models, such as pure translation or affine deformations. Notice that since we are comparing the appearance of regions, a notion of "discrepancy" or "matching cost" must be agreed upon.

4.2 Photometric and geometric features

In the previous section we have seen that, in the most general case, finding correspondence between points in two images of a scene with arbitrary photometry and geometry is impossible. Therefore, it makes sense to try

to characterize the locations in the image where such correspondence can be established.

One way to do that, as we have suggested in the previous section, is to associate each *geometric entity*, for instance a *point*, with a *support region*, that is a neighborhood of $\mathbf{x}$: $W(\mathbf{x}) \subset \Omega \mid \mathbf{x} \in W(\mathbf{x})$, and characterize the nature of the point in terms of the values of the image in the support region, $\{I(\tilde{\mathbf{x}}) \mid \tilde{\mathbf{x}} \in W(\mathbf{x})\}$. Depending on the particular irradiance pattern I, not all label sets can be matched. Consider, for instance, two images of a white wall. Clearly $I_1(\mathbf{x}_1) = I_2(\mathbf{x}_2)\ \forall \mathbf{x}_1, \mathbf{x}_2$, and therefore equation (4.4) does not define a correspondence function h.

As we have anticipated, rather than trying to match each pixel, we are going to opportunistically define as "feature points" those for which the correspondence problem can be solved. The first step is to choose a class of transformations, h, that depends upon a set of parameters α. For instance, $\alpha = \mathbf{u}$ for the translational model, and $\alpha = \{A, \mathbf{b}\}$ for the affine model. With an abuse of notation we indicate the dependency of h from the parameters as $h(\alpha)$. We can then define a pixel $\mathbf{x}$ to be a *feature point* if there exists a neighborhood $W(\mathbf{x})$ such that the following equations

$$I_1(\mathbf{x}) = I_2(h(\mathbf{x}, \alpha))\ \forall\ \mathbf{x} \in W(\mathbf{x}) \tag{4.8}$$

uniquely determine the parameters α. From the example of the white wall, it is intuitive that such conditions would require that I_1 and I_2 have non-zero gradient. In the sections to follow we will derive an explicit form of this condition for the case of translational model. Notice also that, due to noise, there may exist no parameters α for which the equation above is satisfied. Therefore, we may have to solve the equation above as an optimization problem. After choosing a discrepancy measure in Ω, $\phi(I_1, I_2)$, we can seek for

$$\hat{\alpha} = \arg\min_{\alpha} \phi(I_1(\mathbf{x}), I_2(h(\mathbf{x}, \alpha))). \tag{4.9}$$

Similarly, one may define a *feature line* as line segment with a support region and a collection of labels such that the orientation and normal displacement of the transformed line can be uniquely determined from the equation above.

In the next two sections we will see how to efficiently solve the problem above for the case where α are translation parameters or affine parameters.

Comment 4.1. *The definition of feature points and lines allows us to abstract our discussion from pixels and images to abstract entities such as points and lines. However, as we will discuss in later chapters, this separation is more conceptual than factual. Indeed, all the constraints among geometric entities that we will derive in Chapters 8, 9, 10, and 11 can be rephrased in terms of constraints on the irradiance values on collections of images.*

4.3 Optical flow and feature tracking

4.3.1 Translational model

Consider the translational model described in the previous sections, where

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1)) = I_2(\mathbf{x}_1 + \mathbf{u}). \tag{4.10}$$

If we consider the two images as being taken from infinitesimally close vantage points, we can write a differential version of the purely translational model as a constraint. In order to make the notation more suggestive, we use a time index t, so that $I_1(\mathbf{x}_1) \doteq I(\mathbf{x}(t_1), t_1)$. With this notation, assuming infinitesimal motion, Equation (4.10) can be re-written as

$$I(\mathbf{x}(t), t) = I(h(\mathbf{x}(t) + \mathbf{u}(t), t + dt) \tag{4.11}$$

where we have taken the liberty of calling $t_1 \doteq t$ and $t_2 \doteq t+dt$ to emphasize the fact that motion is infinitesimal. In the limit where $dt \to 0$, the equation can be written as

$$\nabla I(\mathbf{x}(t), t)^T \mathbf{u} + I_t(\mathbf{x}(t), t) = 0 \tag{4.12}$$

where

$$\nabla I(\mathbf{x}, t) \doteq \begin{bmatrix} \frac{\partial I}{\partial x}(\mathbf{x}, t) \\ \frac{\partial I}{\partial y}(\mathbf{x}, t) \end{bmatrix}, \quad \text{and} \quad I_t(\mathbf{x}, t) \doteq \frac{\partial I}{\partial t}(\mathbf{x}, t). \tag{4.13}$$

Another way of writing the equation is in terms of the total derivative with respect to time,

$$\frac{dI(x(t), y(t), t)}{dt} = 0. \tag{4.14}$$

since for $I(x(t), y(t), t)$ we have

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{4.15}$$

which is identical to (4.12) once we let $\mathbf{u} \doteq [u_x, u_y]^T = [\frac{dx}{dt}, \frac{dy}{dt}]^T \in \mathbb{R}^2$.

This equation is called the *"image brightness constancy constraint"*. Depending on where the constraint is evaluated, this equation can be used to compute what is called *optical flow*, or to *track photometric features* in a sequence of moving images.

Before we delve into the study of optical flow and feature tracking, notice that (4.12), if computed at each point, only provides one equation for two unknowns ($\mathbf{u}$). It is only when the equation is evaluated at each point in a region $\tilde{x} \in W(\mathbf{x})$, and the motion is assumed to be constant in the region, that the equation provides enough constraints on $\mathbf{u}$.

Optical flow and the aperture problem

We start by rewriting Equation (4.12) in a more compact form as

$$\boxed{\nabla I^T \mathbf{u} + I_t = 0} \tag{4.16}$$

For convenience we omit the time from $(x(t), y(t))$ in $I(x(t), y(t), t)$ and write simply $I(x, y, t)$.

There are two ways to exploit this constraint, corresponding to an "Eulerian" and "Lagrangian" approach to the problem of determining image motion. When we fix our attention at a particular image location $\bar{\mathbf{x}}$ and use (4.16) to compute the velocity of "particles flowing" through that pixel, $\mathbf{u}(\bar{\mathbf{x}}, t)$ is called *optical flow*. When instead the attention is on a particular particle $\mathbf{x}(t)$, and (4.16) is computed at the location $\mathbf{x}(t)$ as it moves through the image domain, we refer to the computation of $\mathbf{u}(\mathbf{x}(\bar{t}), \bar{t})$ as *feature tracking*. Optical flow and feature tracking are obviously related by $\mathbf{x}(t + dt) = \mathbf{x}(t) + \mathbf{u}(\mathbf{x}(t), t)dt$. The only difference, at the conceptual level, is where the vector $\mathbf{u}(\mathbf{x}, t)$ is computed: in optical flow it is computed at a fixed location on the image, whereas in feature tracking it is computed at the point $\mathbf{x}(t)$.

The brightness constancy constraint captures the relationship between the image velocity $\mathbf{u}$ of an image point (x, y) and spatial and temporal derivatives $\nabla I, I_t$ which are directly measurable. As we have already noticed, the equation provides a single constraint for two unknowns $\mathbf{u} = [u_x, u_y]^T$. From the linear algebraic point of view there are infinitely many solutions $\mathbf{u}$ which satisfy this equation. All we can compute is the projection of the actual optical flow vector to the direction of image gradient ∇I. This component is also referred to as *normal flow* and can be thought of as a minimum norm vector $\mathbf{u}_n \in \mathbb{R}^2$ which satisfies the brightness constancy constraint. It is given by a projection of the true motion field $\mathbf{u}$ into gradient direction and its magnitude is given by

$$\mathbf{u}_n \doteq \frac{\nabla I^T \mathbf{u}}{\|\nabla I\|} \cdot \frac{\nabla I}{\|\nabla I\|} = -\frac{I_t}{\|\nabla I\|} \cdot \frac{\nabla I}{\|\nabla I\|}. \tag{4.17}$$

Direct consequence of this observation is so called *aperture problem* which can be easily visualized. For example consider viewing locally an area of the image as it undergoes motion (see Figure 4.1). In spite of the fact that the dark square moved between the two consecutive frames diagonally, observing the motion purely through the window (aperture) we can observe only changes in the horizontal direction and may assume that the observed pattern moved arbitrarily along the direction of the edge.

The aperture problem is yet another manifestation of the fact that we cannot solve the correspondence problem by looking at brightness values at a point. Instead, as we have anticipated, we will be looking at brightness values in a region around each point. The assumption that motion is constant within the region is commonly known as "local constancy." It states

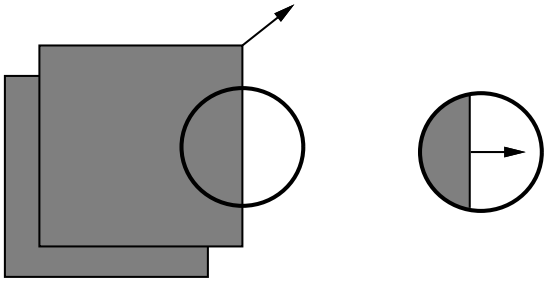

Figure 4.1. In spite of the fact that the dark square moved diagonally between the two consecutive frames, observing purely the cut-out patch we can only observe motion in horizontal direction and we may assume that the observed pattern moved arbitrarily along the direction of the edge.

that over a small window centered at the image location $\mathbf{x}$ the optical flow is the same for all points in a window $W(\mathbf{x})$. This is equivalent to assuming a purely translational deformation model:

$$h(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\lambda\mathbf{x}) = \mathbf{x} + \mathbf{u}(\lambda\mathbf{x}) \quad \text{for all } \mathbf{x} \in W(\mathbf{x}) \tag{4.18}$$

Hence the in order to compute the image velocity $\mathbf{u}$, we seek the image velocity which is most consistent with all the point constraints. Due to the effect of noise in the model (4.6), this now over-constrained system may not have a solution. Hence the matching process is formulated as a minimization of the following quadratic error function based on the brightness constancy constraint:

$$E_b(\mathbf{u}) = \sum_{W(x,y)} (\nabla I^T(x,y,t) \cdot \mathbf{u}(x,y) + I_t(x,y,t))^2 \tag{4.19}$$

where the subscript b indicates brightness constancy. To obtain a linear least-squares estimate of $\mathbf{u}(x,y)$ at each image location, we compute the derivative with respect to $\mathbf{u}$ of the error function $E_b(\mathbf{u})$:

$$\nabla E_b(\mathbf{u}) = 2 \sum_{W(x,y)} \nabla I(\nabla I^T \cdot \mathbf{u} + I_t) = 2 \sum_{W(x,y)} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \mathbf{u} + \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}$$

For $\mathbf{u}$ that minimizes E_b, it is necessary that $\nabla E_b(\mathbf{u}) = 0$. This yields:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \mathbf{u} + \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} = 0 \tag{4.20}$$

or in a matrix form:

$$G\mathbf{u} + \mathbf{b} = \mathbf{0}. \tag{4.21}$$

Solving this equation (if G is invertible) gives the least-squares estimate of image velocity:

$$\boxed{\mathbf{u} = -G^{-1}\mathbf{b}} \tag{4.22}$$

Note however that the matrix G is not guaranteed to be invertible. If the intensity variation in a local image window varies only along one dimension e.g., $I_x = 0$ or $I_y = 0$) or vanishes ($I_x = 0$ and $I_y = 0$), then G is *not* invertible. These singularities are frequently referred to as an *aperture* and *blank wall problem*, respectively. Based on these observations we see that it is the local properties of image irradiance in the window $W(\mathbf{x})$ which determine the ill-posedness of the problem.

Since it seems that the correspondence problem can be solved, under the brightness constancy assumption, for points $\mathbf{x}$ where $G(\mathbf{x})$ is invertible, it is convenient to *define* such points as "feature points" at least according to the quadratic criterion above.

The discrete-time equivalent of the above derivation is known as the "sum of squared differences" (SSD) algorithm. The SSD approach considers an image window W centered at a location (x, y) at time t and other candidate locations $(x + dx, y + dy)$ in the image at time $t + dt$, where the point could have moved between two frames. The goal is to find a displacement (dx, dy) at a location in the image (x, y) which minimizes the SSD criterion:

$$SSD(dx, dy) = \sum_{W(x,y)} (I(x, y, t) - I(x + dx, y + dy, t + dt))^2 \qquad (4.23)$$

where the summation ranges over the image window $W(x, y)$ centered at a feature of interest. Comparing to the error function (4.19), an advantage of the SSD criterion is that in principle we no longer need to compute derivatives of $I(x, y, t)$. One alternative is for computing the displacement is to enumerate the function at each location and choose the one which gives the minimum error. This formulation is due to Lucas and Kanade [LK81] and was originally proposed in the context of stereo algorithms and was later refined by Tomasi and Kanade [TK92] in more general feature tracking context.

4.3.2 Affine deformation model

Assuming that a whole region is moving with purely translational motion is reasonable only for very particular cases of scene structure and camera motion, as we have anticipated in the previous section. This is therefore too rigid a model, at the opposite end of the spectrum of assuming that each pixel can move independently.

In between these two extreme cases one may adopt richer models that account for more general scenes and motion. A commonly adopted model is that of affine deformation of image regions that support point features: $I_1(\mathbf{x}) = I_2(h(\mathbf{x}))$, where the function h has the following form:

$$h(\mathbf{x}) = A\mathbf{x} + \mathbf{d} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \mathbf{x} + \begin{bmatrix} a_5 \\ a_6 \end{bmatrix}. \qquad (4.24)$$

Similarly as in the pure translation model (4.10) we can formulate brightness constancy constraint for this more general 6-parameter affine model:

$$I(\mathbf{x}, t) = I(A\mathbf{x} + \mathbf{d}, t + dt). \tag{4.25}$$

Enforcing the above assumption over a region of the image we can estimate the unknown affine parameters A and $\mathbf{d}$, by integrating the above constraint for all the points in the region $W(x, y)$

$$E_a(A, \mathbf{d}) = \sum_{W(\mathbf{x})} (I(\mathbf{x}, t) - I(A\mathbf{x} + \mathbf{d}, t + dt))^2 \tag{4.26}$$

where the subscript a indicates the affine deformation model. By solving the above minimization problem one can obtain a linear-least squares estimate of the affine parameters A and $\mathbf{d}$ directly from the image measurements of spatial and temporal gradients. The solution has been first derived by Shi and Tomasi in [**?**].

4.4 Feature detection algorithms

In the previous sections we have seen how to compute the displacement or affine deformation of a photometric feature, and we have distinguished the case where the computation is performed at a fixed set of locations (optical flow) from the case where point features are tracked over time (feature tracking). One issue we have not addressed in this second case is how to initially select the points to be tracked. However, we have hinted in various occasions at the possibility of selecting as "feature points" the locations that allow to easily solve the correspondence problem. In this section we make this more precise by giving an numerical algorithm to select such features.

As the reader may have noticed, the description of any of those feature points relies on knowing the gradient of the image. Hence before we can give out any numerical algorithm for feature selection, we need to study how to compute image gradient in an accurate and robust way.

4.4.1 Computing image gradient

Neglecting for the moment the discrete nature of digital images, conceptually the image gradient $\nabla I(x, y) = [I_x, I_y]^T \in \mathbb{R}^2$ is simply given by the two partial derivatives:

$$I_x(x, y) = \frac{\partial I}{\partial x}(x, y), \quad I_y(x, y) = \frac{\partial I}{\partial y}(x, y). \tag{4.27}$$

While the notion of derivative is well defined for smooth functions, additional steps need to be taken when computing the derivatives of digital images.

The most straightforward and commonly used approximation of the gradient by first order differences of the raw images usually results in a rough approximation of ∇I. Due to the presence of noise, digitization error as well as the fact that only discrete samples of function $I(x, y)$ are available, a common practice is to *smooth* $I(x, y)$ before computing first differences to approximate the gradient. Smoothing can be accomplished by convolution of the image with a smoothing filter. Suitable low-pass filter which attenuates the high-frequency noise and at the same time can be serve as some sort of interpolating function is for instance a 2-D Gaussian function:

$$g_\sigma(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x^2+y^2)/2\sigma^2} \tag{4.28}$$

where the choice σ is related to the effective size of the window over which the intensity function is averaged. Since the Gaussian filter is separable, the convolution can be accomplished in two separate steps, first convolving the image in horizontal and vertical direction with a 1-D Gaussian kernel. The smoothed image then is:

$$\tilde{I}(x, y) = \int_u \int_v I(u, v) g_\sigma(x - u) g_\sigma(y - v)\ dudv. \tag{4.29}$$

Figures 4.2 and 4.3 demonstrate the effect of smoothing a noisy image by a convolution with the Gaussian function. Since convolution is linear, the

Figure 4.2. An image "Lena" corrupted by white random noises.

Figure 4.3. Smoothed image by a convolution with the Gaussian function.

operations of smoothing and computing the gradient can be interchanged in order, so that taking derivatives of the smoothed function $\tilde{I}$ is equivalent to convolving the original image I with a kernel that is the derivative of g_σ. Hence we can compute an approximation of the gradient ∇I according

to the following equations:

$$I_x(x,y) = \int_u \int_v I(u,v) g'_\sigma(x-u) g_\sigma(y-v)\, dudv \tag{4.30}$$

$$I_y(x,y) = \int_u \int_v I(u,v) g_\sigma(x-u) g'_\sigma(y-v)\, dudv \tag{4.31}$$

where $g'_\sigma(x) = \frac{-x}{\sqrt{2\pi\sigma^3}} e^{-x^2/2\sigma^2}$ is the derivative of the Gaussian function. Of course, in principle, linear filters other than the Gaussian can be designed to smooth the image or compute the gradient. Of course, over digital images, we cannot perform continuous integration. So the above convolution is commonly approximated by a finite summation over an region Ω of interest:

$$I_x(x,y) = \sum_{(u,v)\in\Omega} I(u,v) g'_\sigma(x-u) g_\sigma(y-v) \tag{4.32}$$

$$I_y(x,y) = \sum_{(u,v)\in\Omega} I(u,v) g_\sigma(x-u) g'_\sigma(y-v) \tag{4.33}$$

Ideally Ω should be chosen to be the entire image. But in practice, one can usually choose its size to be big enough so that the values of the Gaussian kernel are negligibly small outside of the region.

4.4.2 Line features: edges

By an edge in an image, we typically refer to a collection of pixels aligned along a certain direction such that norm of the gradient vector ∇I is high in the orthogonal direction. This simple idea results the following important algorithm

Algorithm 4.1 (Canny edge detector). *Given an image $I(x,y)$, follow the steps to detect if a given pixel (x,y) is an edge:*

- *set a threshold $\tau > 0$ and standard deviation $\sigma > 0$ for the Gaussian function,*
- *compute the gradient vector $\nabla I = [I_x, I_y]^T$ according to*

$$\boxed{\begin{array}{rcl} I_x(x,y) & = & \sum_{u,v} I(u,v) g'_\sigma(x-u) g_\sigma(y-v) \\ I_y(x,y) & = & \sum_{u,v} I(u,v) g_\sigma(x-u) g'_\sigma(y-v) \end{array}} \tag{4.34}$$

- *if $\|\nabla I(x,y)\|^2 = \nabla I^T \nabla I$ is a local maximum along the gradient and larger than the prefixed threshold τ, then mark it as an edge pixel.*

Figures 4.4 and 4.5 demonstrate edge pixels detected by the Canny edge detector on a gray-level image. Edges are important features which give away some local geometric information of the image. However, as we mentioned in previous sections, they are typically not good features to track due to the aperture problem. Hence in order to select features which are

Figure 4.4. Original image.

Figure 4.5. Edge pixels from the Canny edge detectors.

more robust to track or match, point features, or the so-called "corners", are more favored.

4.4.3 Point features: corners

The solution of the correspondence problem for the case of pure translation relied on inverting the matrix G made of the spatial gradients of the image (4.22). For G to be invertible, the region must have non-trivial gradient along two independent directions, resembling therefore "corner" structures in the images, as shown in Figure 4.6. So the existence of a corner point

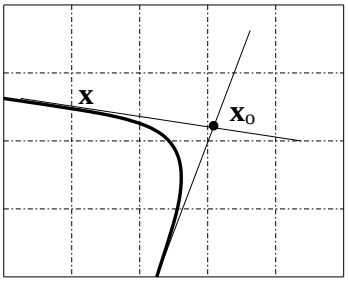

Figure 4.6. A corner feature $\mathbf{x}_o$ is the virtual intersection of local edges.

$\mathbf{x}_o = [x_o, y_o] \in \mathbb{R}^2$ means that over the window $W(x_o, y_o)$ the following minimization has a solution

$$\min_{\mathbf{x}_o} E_c(\mathbf{x}_o) \doteq \sum_{\mathbf{x} \in W(\mathbf{x}_o)} \left(\nabla I^T(\mathbf{x})(\mathbf{x} - \mathbf{x}_o)\right)^2 \tag{4.35}$$

where $\nabla I(\mathbf{x})$ is the gradient calculated at $\mathbf{x} = [x, y]^T \in W(x_o, y_o)$. It is then direct to check that the existence of a local minima for this error

function is equivalent to the summation of outer product of the gradients, i.e.

$$\sum_{\mathbf{x}\in W(\mathbf{x}_o)} \nabla I(\mathbf{x}) \nabla I^T(\mathbf{x}) \tag{4.36}$$

is a non-singular matrix. In the simplest implementations, the partial derivatives are approximated by convolution with the derivative of the Gaussian function. Overall summation after approximation yields a matrix G

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{4.37}$$

If the σ_{min}, denoting the smallest singular value of G, is above specified threshold τ we say that the point (x_0, y_0) is a feature point. If the feature-window has constant brightness, i.e. both singular values of G are zero, it cannot be localized in another image. When one of the singular values is close to zero, the brightness varies in a single direction only. In order to be able to solve the correspondence problem, a window needs to have a significant gradient along two independent directions. We will elaborate on these issues in the context of feature tracking. After this definition, it is easy to devise an algorithm to extract feature points (or corners)

Algorithm 4.2 (Corner detector). *Given an image $I(x, y)$, follow the steps to detect if a given pixel (x, y) is a corner feature:*

- *set a threshold $\tau \in \mathbb{R}$ and a window W of fixed size,*
- *for all pixels in the window W around (x, y) compute the matrix*

$$\boxed{G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}} \tag{4.38}$$

- *if the smallest singular value $\sigma_{min}(G)$ is bigger than the prefixed threshold τ, then mark the pixel as a feature (or corner) point.*

Although we have used the word "corner", the reader should observe that the test above only guarantees that the irradiance function I is "changing enough" in two independent directions within the window of interest. In other words, the window contains "sufficient texture". Besides using the norm (inner product) or outer product of the gradient ∇I to detect edge or corner, criteria for detecting both edge and corner together have also been explored. One of them is the well-known Harris corner and edge detector [HS88]. The main idea is to threshold the following quantity

$$C(G) = \det(G) + k \cdot \text{trace}^2(G) \tag{4.39}$$

where different choices of the parameter $k \in \mathbb{R}$ may result in either an edge detector or corner detector. To see this, let assume the two eigenvalues

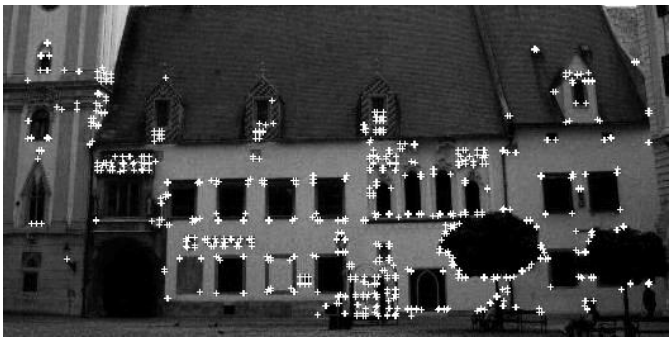

Figure 4.7. An example of the response of the Harris feature detector using 5×5 integration window and parameter $k = 0.04$.

(which in this case coincide with the singular values) of G are σ_1, σ_2. Then

$$C(G) = \sigma_1\sigma_2 + k(\sigma_1 + \sigma_2)^2 = (1 + 2k)\sigma_1\sigma_2 + k(\sigma_1^2 + \sigma_2^2). \tag{4.40}$$

Note that if $k > 0$ and either one of the eigenvalues is large, so will be $C(G)$. That is, both distinctive edge and corner features will more likely pass the threshold. If $k < 0$, then both eigenvalues need to be big enough to make $C(G)$ pass the threshold. In this case, the corner feature is favored. Simple thresholding operation often does not yield satisfactory results and leads to a detection of too many corners, which are not well localized. Partial improvements can be obtained by search for the local minima in the regions, where the response of the detector was high. Alternatively, more sophisticated techniques can be used, which utilize the edge detection techniques and indeed search for the high curvature points of the detected contours.

4.5 Compensating for photometric factors

4.6 Exercises

1. **Motion model**. Consider measuring image motion $h(\mathbf{x})$ and noticing that $h(\mathbf{x}) = \mathbf{x} + \mathbf{u}\ \forall\ \mathbf{x} \in \Omega$, i.e. each point on the image translates by the same amount $\mathbf{u}$. What particular motion (R, T) and 3D structure $\mathbf{X}$ must the scene have to satisfy this model?
2. Repeat the exercise above for an affine motion model, $h(\mathbf{x}) = A\mathbf{x} + \mathbf{u}$.
3. Repeat the exercise above for a projective motion model, $h(\mathbf{x}) = H\mathbf{x}$, in homogeneous coordinates.

4. **Eigenvalues of sum of squared differences**
   Given a set of vectors $u_1, \dots, u_m \in \mathbb{R}^n$, prove that all eigenvalues of the matrix

$$G = \sum_{i=1}^{m} u_i u_i^T \in \mathbb{R}^{n \times n} \tag{4.41}$$

   are non-negative. This concludes that the eigenvalues of G are the same as the singular values of G. (Note: you may take it for granted that all the eigenvalues are real since G is a real symmetric matrix.)

5. **Implementation of the corner detector**
   Implement a version of the corner detector introduced in class using MATLAB. Mark the most distinctive, say 20 to 50 (you get to choose the exact number), corner features of the image "Lena". Both your MATLAB codes and results must be turned in. (If you insist on coding in C or C++, it is fine too. But you must include clear instructions on how to execute your files.)
   After you are done, you may try to play with it. Here are some suggestions:

   - Identify and compare practical methods to choose the threshold τ or other parameters. Evaluate the choice by altering the level of brightness, saturation and contrast in the image.
   - In practice, you like to select only one pixel around a corner feature. Devise a method to choose the "best" pixel (or sub-pixel location) within a window, instead of every pixel above the threshold.
   - Devise some quality measure for feature points and sort them according to such measure. Note that the threshold only have "soft" control on the number of features selected. With such a quality measure, however, you can specify any number of features you want.
   - Surf the web and find two images of the same scene, try to match the features selected by your corner detector using the naive SSD criterion (how does it work?)

6. **Sub-pixel iteration**

   Both the linear and the affine model can be refined by pursuing sub-pixel iterations as well as by using multi-scale deformation models that allow handling larger deformations. In order to achieve sub-pixel accuracy, implement the following iteration

   - $\hat{d}^0 = G^{-1} e$
   - $\hat{d}^{i+1} = G^{-1} e^{i+1}$

   where we define

- $e^0 \doteq e$
- $e^{i+1} \doteq \int_W \nabla I(I - I_t)_{(u+d_1^i, v+d_2^i, t)} du dv$.

At each step $(u+d_1^i, v+d_2^i)$ is in general not on the pixel grid, so that it is necessary to interpolate the brightness values to obtain image intensity at that location. Matrix G is the discrete version of what we defined to be the SSD in the first section of this chapter.

7. **Multi-scale implementation**

   One problem common to all differential techniques is that they fail as the displacement across frames is bigger than a few pixels. One possible way to overcome this inconvenience is to use a coarse-to-fine strategy:

   - build a pyramid of images by smoothing and sub-sampling the original images (see for instance [BA83])
   - select features at the desired level of definition and then propagate the selection up the pyramid
   - track the features at the coarser level
   - propagate the displacement to finer resolutions and use that displacement as an initial step for the sub-pixel iteration described in the previous section.

   The whole procedure can be very slow for a full-resolution image if implemented on conventional hardware. However, it is highly parallel computation, so that the image could be sub-divided into regions which are then assigned to different processors. In many applications, where a prediction of the displacement of each feature is available, it is possible to process only restricted regions of interest within the image.

Part II

Geometry of pairwise views

Chapter 5
Reconstruction from two calibrated views

This chapter introduces the basic geometry of reconstruction of points in 3-D space from image measurements made from two different (calibrated) camera viewpoints. We then introduce a simple algorithm to recover the 3-D position of such points from their 2-D views. Although the algorithm is primarily conceptual, it illustrates the basic principles that will be used in deriving both more sophisticated algorithms as well as algorithms for the more general case of reconstruction from multiple views which will be discussed in Part III of the book.

The framework for this chapter is rather simple: Assume that we are given two views of a set of N feature points, which we denote formally as $\mathcal{I}$. The unknowns are the position of the points in 3-D space, denoted $\mathcal{Z}$ and the relative pose of the cameras, denoted $\mathcal{G}$, we will in this Chapter first derive constraints between these quantities. These constraints take the general form (one for each feature point)

$$f_i(\mathcal{I}, \mathcal{G}, \mathcal{Z}) = 0, \quad i = 1, \dots, N \tag{5.1}$$

for some functions f_i. The central problem to be addressed in this Chapter is to start from this system of nonlinear equations and attempt, if possible, to solve it to recover the unknowns $\mathcal{G}$ and $\mathcal{Z}$. We will show that it is indeed possible to recover the unknowns.

The equations in (5.1) are nonlinear and no closed-form solution is known at this time. However, the functions f_i have a very particular structure that allows us to eliminate the parameters $\mathcal{Z}$ that contain information about the 3-D structure of the scene and be left with a constraint on $\mathcal{G}$ and $\mathcal{I}$ alone, which is known as the *epipolar constraint*. As we shall see in Section 5.1,

the epipolar constraint has the general form

$$h_j(\mathcal{I}, \mathcal{G}) = 0, \quad j = 1, \dots, N. \tag{5.2}$$

Interestingly enough, it can be solved for $\mathcal{G}$ *in closed form*, as we shall show in Section 5.2.

The closed-form algorithm, however, is not suitable for use in real images corrupted by noise. In Section 5.3, we discuss how to modify it so as to minimize the effect of noise. When the two views are taken from infinitesimally close vantage points, the basic geometry changes in ways that we describe in Section 5.4.

5.1 The epipolar constraint

We begin by assuming that we are given two images of the same scene taken from two distinct vantage points. Using the tools developed in Chapter 4, we can identify corresponding points in the two images. Under the assumption that the scene is *static* (there are no moving objects) and that the *brightness constancy constraint* is satisfied (Section 4.3), corresponding points are images of the same point in space. Therefore, if we call $\mathbf{x}_1, \mathbf{x}_2$ the (homogeneous) coordinates of corresponding points on the image, these two points are related by a precise geometric relationship that we describe in this section.

5.1.1 Discrete epipolar constraint and the essential matrix

Following the notation of Chapter 3, each camera is represented by an orthonormal reference frame and can therefore be described as a change of coordinates relative to an inertial reference frame. Without loss of generality, we can assume that the inertial frame corresponds to one of the two cameras, while the other is positioned and oriented according to $g = (R, T) \in SE(3)$. Let $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ be the homogeneous coordinates of the projection of the same point p onto the two cameras. If we call $\mathbf{X}_1 \in \mathbb{R}^3$ and $\mathbf{X}_2 \in \mathbb{R}^3$ the 3-D coordinates of the point p relative to the two camera frames, they are related by a rigid body motion

$$\mathbf{X}_2 = R\mathbf{X}_1 + T$$

which can be written in terms of the images $\mathbf{x}_i$'s and the depths λ_i's as

$$\lambda_2 \mathbf{x}_2 = R\lambda_1 \mathbf{x}_1 + T.$$

In order to eliminate the depths λ_i's in the preceding equation, multiply both sides by $\widehat{T}$ to obtain

$$\lambda_2 \widehat{T}\mathbf{x}_2 = \widehat{T}R\lambda_1 \mathbf{x}_1.$$

Since the vector $\widehat{T}\mathbf{x}_2 = T \times \mathbf{x}_2$ is perpendicular to the vector $\mathbf{x}_2$, the inner product $\langle \mathbf{x}_2, \widehat{T}\mathbf{x}_2 \rangle = {\mathbf{x}_2}^T \widehat{T}\mathbf{x}_2$ is zero. This implies that the quantity $\mathbf{x}_2^T \widehat{T} R \lambda_1 \mathbf{x}_1$ is also zero. Since $\lambda_1 > 0$, the two image points $\mathbf{x}_1, \mathbf{x}_2$ satisfy the following constraint:

$$\boxed{\mathbf{x}_2^T \widehat{T} R \mathbf{x}_1 = 0.} \tag{5.3}$$

A geometric explanation for this constraint is immediate from Figure 6.1. The coordinates of the point p in the first camera $\mathbf{X}_1$, its coordinates in the second camera $\mathbf{X}_2$ and the vector connecting the two optical centers form a triangle. Therefore, the three vectors joining these points to the origin lie in the same plane and their triple product, which measures the volume they span is zero. This is true for the coordinates of the points $\mathbf{X}_i,\ i = 1, 2$ as well as for the homogeneous coordinates of their projection $\mathbf{x}_i,\ i = 1, 2$ since $\mathbf{X}_i$ and $\mathbf{x}_i$ (as two vectors) point in the same direction. The constraint (5.3) is just the triple product written in the reference frame of camera 2. The plane determined by the two centers of projection o_1, o_2 and the point p is called *epipolar plane*, the projection of the center of one camera onto the image plane of another camera $\mathbf{e}_i,\ i = 1, 2$ is called *epipole*, and the constraint (5.3) is called *epipolar constraint* or *essential constraint*. The matrix $E \doteq \widehat{T}R \in \mathbb{R}^{3\times 3}$ which captures the relative orientation between the two cameras is called the *essential matrix*.

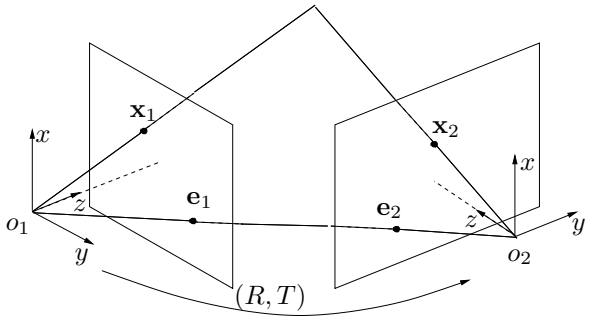

Figure 5.1. Two projections $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ of a 3-D point p from two vantage points. The relative Euclidean transformation between the two vantage points is given by $(R, T) \in SE(3)$. The vectors $\mathbf{x}_1, \mathbf{x}_2$ and T are coplanar, and therefore their triple product (5.3) must be zero.

5.1.2 Elementary properties of the essential matrix

The matrix $E = \widehat{T}R \in \mathbb{R}^{3\times 3}$ in equation (5.3) contains information about the relative position T and orientation $R \in SO(3)$ between the two cameras. Matrices of this form belong to a very particular set of matrices in $\mathbb{R}^{3\times 3}$

which we call the *essential space* and denote by $\mathcal{E}$

$$\mathcal{E} := \left\{ \widehat{T}R \,\middle|\, R \in SO(3), T \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{3\times 3}.$$

Before we study the structure of the space of essential matrices, we introduce a very useful lemma from linear algebra.

Lemma 5.1 (The hat operator). *If $T \in \mathbb{R}^3$, $A \in SL(3)$ and $T' = AT$, then $\widehat{T} = A^T \widehat{T'} A$.*

Proof. Since both $A^T\widehat{(\cdot)}A$ and $\widehat{A^{-1}(\cdot)}$ are linear maps from $\mathbb{R}^3$ to $\mathbb{R}^{3\times 3}$, one may directly verify that these two linear maps agree on the basis $[1,0,0]^T, [0,1,0]^T$ or $[0,0,1]^T$ (using the fact that $A \in SL(3)$ implies that $\det(A) = 1$). □

The following theorem, due to Huang and Faugeras [HF89], captures the algebraic structure of essential matrices:

Theorem 5.1 (Characterization of the essential matrix). *A non-zero matrix $E \in \mathbb{R}^{3\times 3}$ is an essential matrix* if and only if *E has a singular value decomposition (SVD): $E = U\Sigma V^T$ with*

$$\Sigma = diag\{\sigma, \sigma, 0\}$$

for some $\sigma \in \mathbb{R}_+$ and $U, V \in SO(3)$.

Proof. We first prove the necessity. By definition, for any essential matrix E, there exists (at least one pair) $(R,T), R \in SO(3), T \in \mathbb{R}^3$ such that $\widehat{T}R = E$. For T, there exists a rotation matrix R_0 such that $R_0 T = [0, 0, \|T\|]^T$. Denote this vector $a \in \mathbb{R}^3$. Since $\det(R_0) = 1$, we know $\widehat{T} = R_0^T \widehat{a} R_0$ from Lemma 5.1. Then $EE^T = \widehat{T}RR^T\widehat{T}^T = \widehat{T}\widehat{T}^T = R_0^T\widehat{a}\widehat{a}^T R_0$. It is direct to verify that

$$\widehat{a}\widehat{a}^T = \begin{bmatrix} 0 & -\|T\| & 0 \\ \|T\| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & \|T\| & 0 \\ -\|T\| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \|T\|^2 & 0 & 0 \\ 0 & \|T\|^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

So the singular values of the essential matrix $E = \widehat{T}R$ are $(\|T\|, \|T\|, 0)$. In general, in the SVD of $E = U\Sigma V^T$, U and V are unitary matrices, that is matrices whose columns are orthonormal, but whose determinants can be ± 1. We still need to prove that $U, V \in SO(3)$ (i.e. have determinant $+1$) to establish the theorem. We already have $E = \widehat{T}R = R_0^T \widehat{a} R_0 R$. Let $R_Z(\theta)$ be the matrix which represents a rotation around the Z-axis (or the X_3-axis) by an angle of θ radians, i.e. $R_Z(\theta) = e^{\widehat{e_3}\theta}$ with $e_3 = [0,0,1]^T \in \mathbb{R}^3$. Then

$$R_Z\left(+\frac{\pi}{2}\right) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then $\widehat{a} = R_Z(+\frac{\pi}{2})R_Z^T(+\frac{\pi}{2})\widehat{a} = R_Z(+\frac{\pi}{2})\ \mathrm{diag}\{\|T\|, \|T\|, 0\}$. Therefore

$$E = \widehat{T}R = R_0^T R_Z\left(+\frac{\pi}{2}\right)\ \mathrm{diag}\{\|T\|, \|T\|, 0\}R_0 R.$$

So in the SVD of $E = U\Sigma V^T$, we may choose $U = R_0^T R_Z(+\frac{\pi}{2})$ and $V^T = R_0 R$. By construction, both U and V are rotation matrices.

We now prove the sufficiency. If a given matrix $E \in \mathbb{R}^{3\times 3}$ has a SVD: $E = U\Sigma V^T$ with $U, V \in SO(3)$ and $\Sigma = \mathrm{diag}\{\sigma, \sigma, 0\}$. Define $(R_1, T_1) \in SE(3)$ and $(R_2, T_2) \in SE(3)$ to be

$$\begin{cases} (\widehat{T}_1, R_1) &= (UR_Z(+\frac{\pi}{2})\Sigma U^T, UR_Z^T(+\frac{\pi}{2})V^T), \\ (\widehat{T}_2, R_2) &= (UR_Z(-\frac{\pi}{2})\Sigma U^T, UR_Z^T(-\frac{\pi}{2})V^T) \end{cases} \tag{5.4}$$

It is now easy to verify that $\widehat{T}_1 R_1 = \widehat{T}_2 R_2 = E$. Thus, E is an essential matrix. □

Given a rotation matrix $R \in SO(3)$ and a rotation vector $T \in \mathbb{R}^3$, it is immediate to construct an essential matrix $E = \widehat{T}R$. The inverse problem, that is how to reconstruct T and R from a given essential matrix E, is less obvious. Before we show how to solve it in Theorem 5.2, we need the following preliminary lemma.

Lemma 5.2. *Consider an arbitrary non-zero skew symmetric matrix $\widehat{T} \in so(3)$ with $T \in \mathbb{R}^3$. If, for a rotation matrix $R \in SO(3)$, $\widehat{T}R$ is also a skew symmetric matrix, then $R = I$ or $e^{\widehat{u}\pi}$ where $u = \frac{T}{\|T\|}$. Further, $\widehat{T}e^{\widehat{u}\pi} = -\widehat{T}$.*

Proof. Without loss of generality, we assume T is of unit length. Since $\widehat{T}R$ is also a skew symmetric matrix, $(\widehat{T}R)^T = -\widehat{T}R$. This equation gives

$$R\widehat{T}R = \widehat{T}. \tag{5.5}$$

Since R is a rotation matrix, there exists $\omega \in \mathbb{R}^3, \|\omega\| = 1$ and $\theta \in \mathbb{R}$ such that $R = e^{\widehat{\omega}\theta}$. Then, (5.5) is rewritten as $e^{\widehat{\omega}\theta}\widehat{T}e^{\widehat{\omega}\theta} = \widehat{T}$. Applying this equation to ω, we get $e^{\widehat{\omega}\theta}\widehat{T}e^{\widehat{\omega}\theta}\omega = \widehat{T}\omega$. Since $e^{\widehat{\omega}\theta}\omega = \omega$, we obtain $e^{\widehat{\omega}\theta}\widehat{T}\omega = \widehat{T}\omega$. Since ω is the only eigenvector associated to the eigenvalue 1 of the matrix $e^{\widehat{\omega}\theta}$ and $\widehat{T}\omega$ is orthogonal to ω, $\widehat{T}\omega$ has to be zero. Thus, ω is equal to either $\frac{T}{\|T\|}$ or $-\frac{T}{\|T\|}$, i.e. $\omega = \pm u$. R then has the form $e^{\widehat{\omega}\theta}$, which commutes with $\widehat{T}$. Thus from (5.5), we get

$$e^{2\widehat{\omega}\theta}\widehat{T} = \widehat{T}. \tag{5.6}$$

According to *Rodrigues' formula* [MLS94], we have

$$e^{2\widehat{T}\theta} = I + \widehat{\omega}\sin(2\theta) + \widehat{\omega}^2(1 - \cos(2\theta)).$$

(5.6) yields

$$\widehat{\omega}^2\sin(2\theta) + \widehat{\omega}^3(1 - \cos(2\theta)) = 0.$$

Since $\widehat{\omega}^2$ and $\widehat{\omega}^3$ are linearly independent (Lemma 2.3 in [MLS94]), we have $\sin(2\theta) = 1 - \cos(2\theta) = 0$. That is, θ is equal to $2k\pi$ or $2k\pi + \pi$, $k \in \mathbb{Z}$. Therefore, R is equal to I or $e^{\widehat{\omega}\pi}$. Now if $\omega = u = \frac{T}{\|T\|}$ then, it is direct from the geometric meaning of the rotation $e^{\widehat{\omega}\pi}$ that $e^{\widehat{\omega}\pi}\widehat{T} = -\widehat{T}$. On the other hand if $\omega = -u = -\frac{T}{\|T\|}$ then it follows that $e^{\widehat{\omega}}\widehat{T} = -\widehat{T}$. Thus, in any case the conclusions of the lemma follows. □

The following theorem shows how to extract rotation and translation from an essential matrix as given in closed-form in equation (5.7) at the end of the theorem.

Theorem 5.2 (Pose recovery from the essential matrix). *There exist exactly two relative poses (R, T) with $R \in SO(3)$ and $T \in \mathbb{R}^3$ corresponding to a non-zero essential matrix $E \in \mathcal{E}$.*

Proof. Assume that $(R_1, T_1) \in SE(3)$ and $(R_2, p_2) \in SE(3)$ are both solutions for the equation $\widehat{T}R = E$. Then we have $\widehat{T}_1 R_1 = \widehat{T}_2 R_2$. It yields $\widehat{T}_1 = \widehat{T}_2 R_2 R_1^T$. Since $\widehat{T}_1, \widehat{T}_2$ are both skew symmetric matrices and $R_2 R_1^T$ is a rotation matrix, from the preceding lemma, we have that either $(R_2, T_2) = (R_1, T_1)$ or $(R_2, T_2) = (e^{\widehat{u}_1 \pi} R_1, -T_1)$ with $u_1 = T_1/\|T_1\|$. Therefore, given an essential matrix E there are exactly *two* pairs (R, T) such that $\widehat{T}R = E$. Further, if E has the SVD: $E = U\Sigma V^T$ with $U, V \in SO(3)$, the following formulas give the two distinct solutions

$$\boxed{\begin{array}{ll} (\widehat{T}_1, R_1) = & (UR_Z(+\frac{\pi}{2})\Sigma U^T, UR_Z^T(+\frac{\pi}{2})V^T), \\ (\widehat{T}_2, R_2) = & (UR_Z(-\frac{\pi}{2})\Sigma U^T, UR_Z^T(-\frac{\pi}{2})V^T). \end{array}} \tag{5.7}$$

□

5.2 Closed-form reconstruction

In the previous section, we have seen that images of corresponding points are related by the epipolar constraint, which involves the unknown relative pose between the cameras. Therefore, given a number of corresponding points, we could use the epipolar constraints to try to recover camera pose. In this section, we show a simple closed-form solution to this problem. It consists of two steps: First a matrix E is recovered from a number of epipolar constraints, then relative translation and orientation is extracted from E. However, since the matrix E recovered using correspondence data in the epipolar constraint may not be an essential matrix it needs to be projected into the space of essential matrices prior to applying the formula of equation (5.7).

Although the algorithm that we will propose is mainly conceptual and does not work well in the presence of large noise or uncertainty, it is impor-

tant in its illustration of the geometric structure of the space of essential matrices. We will deal with noise and uncertainty in Section 5.3.

5.2.1 The eight-point linear algorithm

Let $E = \widehat{T}R$ be the essential matrix associated with the epipolar constraint (5.3). When the entries of this 3×3 matrix are denoted as

$$E = \begin{bmatrix} e_1 & e_2 & e_3 \\ e_4 & e_5 & e_6 \\ e_7 & e_8 & e_9 \end{bmatrix} \tag{5.8}$$

and arrayed in a vector, which we define to be the *essential vector* $\mathbf{e} \in \mathbb{R}^9$, we have

$$\mathbf{e} = [e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9]^T \in \mathbb{R}^9.$$

Since the epipolar constraint $\mathbf{x}_2^T E \mathbf{x}_1 = 0$ is linear in the entries of E, we can rewrite it as a linear equation in the entries of e, namely, $\mathbf{a}^T \mathbf{e} = 0$ for some $\mathbf{a} \in \mathbb{R}^9$. The vector $\mathbf{a} \in \mathbb{R}^9$ is a function of the two corresponding image points, $\mathbf{x}_1 = [x_1, y_1, z_1]^T \in \mathbb{R}^3$ and $\mathbf{x}_2 = [x_2, y_2, z_2]^T \in \mathbb{R}^3$ and is given by

$$\mathbf{a} = [x_2x_1, x_2y_1, x_2z_1, y_2x_1, y_2y_1, y_2z_1, z_2x_1, z_2y_1, z_2z_1]^T \in \mathbb{R}^9. \tag{5.9}$$

The epipolar constraint (5.3) can then be rewritten as the inner product of $\mathbf{a}$ and $\mathbf{e}$

$$\mathbf{a}^T \mathbf{e} = 0.$$

Now, given a set of corresponding image points $(\mathbf{x}_1^j, \mathbf{x}_2^j)$, $j = 1, \ldots, n$, define a matrix $A \in \mathbb{R}^{n \times 9}$ associated with these measurements to be

$$A = [\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^n]^T \tag{5.10}$$

where the j^{th} row $\mathbf{a}^j$ is constructed from each pair $(\mathbf{x}_1^j, \mathbf{x}_2^j)$ using (6.21). In the absence of noise, the essential vector $\mathbf{e}$ has to satisfy

$$A\mathbf{e} = 0. \tag{5.11}$$

This linear equation may now be solved for the vector $\mathbf{e}$. For the solution to be unique (up to scale, ruling out the trivial solution $\mathbf{e} = 0$), the rank of the matrix $A \in \mathbb{R}^{9 \times n}$ needs to be exactly eight. This should be the case given $n \geq 8$ "ideal" corresponding points. In general, however, since correspondences may be noisy there may be no solution to (5.11). In such a case, one may choose e to minimize the function $\|A\mathbf{e}\|^2$, i.e. choose $\mathbf{e}$ is the eigenvector of $A^T A$ corresponding to its smallest eigenvalue. Another condition to be cognizant of is when the rank of A is less than 8, allowing for multiple solutions of equation (5.11). This happens when the feature points are not in "general position", for example when they all lie in a plane. Again in the presence of noise when the feature points are in general

position there are multiple small eigenvalues of $A^T A$, corresponding to the lack of conditioning of the data.

However, even in the absence of noise, for a vector $\mathbf{e}$ to be the solution of our problem, it is not sufficient that it is the null space of A. In fact, it has to satisfy an additional constraint, in that its matrix form E must belong to the space of essential matrices. Enforcing this structure in the determination of the null space of A is difficult. Therefore, as a first cut, we could first estimate the null space of A *ignoring the internal structure of* E, obtaining a matrix possibly not belong to the essential space, and then *orthogonally projecting* the matrix thus obtained onto the essential space. The following theorem says precisely what this projection is.

Theorem 5.3 (Projection onto the essential space). *Given a real matrix $F \in \mathbb{R}^{3\times 3}$ with a SVD: $F = U\,diag\{\lambda_1, \lambda_2, \lambda_3\}V^T$ with $U, V \in SO(3)$, $\lambda_1 \geq \lambda_2 \geq \lambda_3$, then the essential matrix $E \in \mathcal{E}$ which minimizes the error $\|E - F\|_f^2$ is given by $E = U\,diag\{\sigma, \sigma, 0\}V^T$ with $\sigma = (\lambda_1 + \lambda_2)/2$. The subscript f indicates the Frobenius norm.*

Proof. For any fixed matrix $\Sigma = \text{diag}\{\sigma, \sigma, 0\}$, we define a subset $\mathcal{E}_\Sigma$ of the essential space $\mathcal{E}$ to be the set of all essential matrices with SVD of the form $U_1\Sigma V_1^T$, $U_1, V_1 \in SO(3)$. To simplify the notation, define $\Sigma_\lambda = \text{diag}\{\lambda_1, \lambda_2, \lambda_3\}$. We now prove the theorem by two steps:

Step 1: We prove that for a fixed Σ, the essential matrix $E \in \mathcal{E}_\Sigma$ which minimizes the error $\|E - F\|_f^2$ has a solution $E = U\Sigma V^T$ (not necessarily unique). Since $E \in \mathcal{E}_\Sigma$ has the form $E = U_1\Sigma V_1^T$, we get

$$\|E - F\|_f^2 = \|U_1\Sigma V_1^T - U\Sigma_\lambda V^T\|_f^2 = \|\Sigma_\lambda - U^T U_1 \Sigma V_1^T V\|_f^2.$$

Define $P = U^T U_1, Q = V^T V_1 \in SO(3)$ which have the forms

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}, \quad Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}. \tag{5.12}$$

Then

$$\|E - F\|_f^2 \;\; = \;\; \|\Sigma_\lambda - U^T U_1 \Sigma V_1^T V\|_f^2 = tr(\Sigma_\lambda^2) - 2tr(P\Sigma Q^T \Sigma_\lambda) + tr(\Sigma^2).$$

Expanding the second term, using $\Sigma = \text{diag}\{\sigma, \sigma, 0\}$ and the notation p_{ij}, q_{ij} for the entries of P, Q, we have

$$tr(P\Sigma Q^T \Sigma_\lambda) = \sigma(\lambda_1(p_{11}q_{11} + p_{12}q_{12}) + \lambda_2(p_{21}q_{21} + p_{22}q_{22})).$$

Since P, Q are rotation matrices, $p_{11}q_{11} + p_{12}q_{12} \leq 1$ and $p_{21}q_{21} + p_{22}q_{22} \leq 1$. Since Σ, Σ_λ are fixed and $\lambda_1, \lambda_2 \geq 0$, the error $\|E - F\|_f^2$ is minimized when $p_{11}q_{11} + p_{12}q_{12} = p_{21}q_{21} + p_{22}q_{22} = 1$. This can be achieved when P, Q are of the general form

$$P = Q = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Obviously $P = Q = I$ is one of the solutions. That implies $U_1 = U, V_1 = V$.

Step 2: From Step 1, we only need to minimize the error function over the matrices of the form $U\Sigma V^T \in \mathcal{E}$ where Σ may vary. The minimization problem is then converted to one of minimizing the error function

$$\|E - F\|_f^2 = (\lambda_1 - \sigma)^2 + (\lambda_2 - \sigma)^2 + (\lambda_3 - 0)^2.$$

Clearly, the σ which minimizes this error function is given by $\sigma = (\lambda_1 + \lambda_2)/2$. □

As we have already pointed out, the epipolar constraint only allows for the recovery of the essential matrix up to a scale (since the epipolar constraint of 5.3 is homogeneous in E, it is not modified by multiplying by any non-zero constant). A typical choice to address this ambiguity is to choose E to have its non-zero singular values to be 1, which corresponds to unit translation, that is, $\|T\| = 1$. Therefore, if E is the essential matrix recovered from image data, we can normalize it at the outset by replacing it with $\frac{E}{\|E\|}$. Note that the sign of the essential matrix is also arbitrary. According to Theorem 5.2, each normalized essential matrix gives two possible poses. So, in general, we can only recover the pose up to four solutions.[1] The overall algorithm, which is due to Longuet-Higgins [LH81], can then be summarized as follows

Algorithm 5.1 (The eight-point algorithm). *For a given set of image correspondences* $(\mathbf{x}_1^j, \mathbf{x}_2^j)$, $j = 1, \ldots, n$ *($n \geq 8$), this algorithm finds* $(R, T) \in SE(3)$ *which solves*

$$\mathbf{x}_2^{jT} \widehat{T} R \mathbf{x}_1^j = 0, \quad j = 1, \ldots, n.$$

1. **Compute a first approximation of the essential matrix**
   Construct the $A \in \mathbb{R}^{n \times 9}$ *from correspondences* $\mathbf{x}_1^j$ *and* $\mathbf{x}_2^j$ *as in (6.21), namely.*

   $$\mathbf{a}^j = [x_2^j x_1^j, x_2^j y_1^j, x_2^j z_1^j, y_2^j x_1^j, y_2^j y_1^j, y_2^j z_1^j, z_2^j x_1^j, z_2^j y_1^j, z_2^j z_1^j]^T \in \mathbb{R}^9.$$

   Find the vector $\mathbf{e} \in \mathbb{R}^9$ *of unit length such that* $\|A\mathbf{e}\|$ *is minimized as follows: compute the SVD* $A = U_A \Sigma_A V_A^T$ *and define* $\mathbf{e}$ *to be the* 9^{th} *column of* V_A*. Rearrange the 9 elements of* $\mathbf{e}$ *into a square* 3×3 *matrix* E *as in (5.8). Note that this matrix will in general* not *be an essential matrix.*

2. **Project onto the essential space**
   Compute the Singular Value Decomposition of the matrix E *recovered from data to be*

   $$E = U\, diag\{\sigma_1, \sigma_2, \sigma_3\} V^T$$

[1] Three of them can be eliminated once imposing the positive depth constraint.

where $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$ and $U, V \in SO(3)$. In general, since E may not be an essential matrix, $\sigma_1 \neq \sigma_2$ and $\sigma_3 > 0$. Compute its projection onto the essential space as $U\Sigma V^T$, where $\Sigma = diag\{1, 1, 0\}$.

3. **Recover displacement from the essential matrix**
Define the diagonal matrix Σ to be Extract R and T from the essential matrix as follows:

$$R = UR_Z^T\left(\pm\frac{\pi}{2}\right)V^T, \quad \widehat{T} = UR_Z\left(\pm\frac{\pi}{2}\right)\Sigma U^T.$$

Remark 5.1 (Infinitesimal viewpoint change). *It is often the case in applications that the two views described in this chapter are taken by a moving camera rather than by two static cameras. The derivation of the epipolar constraint and the associated eight-point algorithm does not change, as long as the two vantage points are distinct. In the limit that the two viewpoints come infinitesimally close, the epipolar constraint takes a distinctly different form called the continuous epipolar constraint which we will introduce in Section 5.4.*

Remark 5.2 (Enough parallax). *In the derivation of the epipolar constraint we have implicitly assumed that $E \neq 0$, which allowed us to derive the eight-point algorithm where the epipolar matrix is normalized to $\|E\| = 1$. Due to the structure of the essential matrix, $E = 0 \Leftrightarrow T = 0$. Therefore, the eight-point algorithm requires that $T \neq 0$. The translation vector T is also known as "parallax". In practice, due to noise, the algorithm will likely return an answer even when there is no parallax. However, in that case the estimated direction of translation will be meaningless. Therefore, one needs to exercise caution to make sure that there is "enough parallax" for the algorithm to operate correctly.*

Remark 5.3 (Positive depth constraint). *Since both E and $-E$ satisfy the same set of epipolar constraints, they in general give rise to $2 \times 2 = 4$ possible solutions to (R, T). However, this does on pose a potential problem because there is only one of them guarantees that the depths of the 3-D points being observed by the camera are all* positive. *That is, in general, three out of the four solutions will be physically impossible and hence may be discarded.*

Remark 5.4 (General position requirement). *In order for the above algorithm to work properly, the condition that the given 8 points are in "general position" is very important. It can be easily shown that if these points form certain degenerate configurations, the algorithm will fail. A case of some practical importance is when all the points happen to lie on the same 2-D plane in $\mathbb{R}^3$. We will discuss the geometry for image features from a plane in a later chapter (in a more general setting of multiple views). Nonetheless, we encourage the reader to solve the Exercise (7) at the end of this chapter that illustrates some basic ideas how to explicitly take*

into account such coplanar information and modify the essential constraint accordingly.

5.2.2 Euclidean constraints and structure reconstruction

The eight-point algorithm just described uses as input a set of eight or more point correspondences and returns the relative pose (rotation and translation) between the two cameras up a an arbitrary scale $\gamma \in \mathbb{R}^+$. Relative pose and point correspondences can then be used to retrieve the position of the points in 3-D space by recovering their depths relative to each camera frame.

Consider the basic rigid body equation, where the pose (R, T) has been recovered, with the translation T defined up to the scale γ. The usual rigid body motion, written in terms of the images and the depths, is given by

$$\lambda_2^j \mathbf{x}_2^j = \lambda_1^j R \mathbf{x}_1^j + \gamma T, \quad 1 \leq j \leq n. \tag{5.13}$$

Notice that, since (R, T) are known, the equations given by (5.13) are linear in both the structural scales λ's and the motion scales γ's and could be therefore easily solved. In the presence of noise, these scales can be retrieved by solving a standard linear least-squares estimation problem, for instance in the least-squares sense. Arrange all the unknown scales in (5.13) into an *extended scale vector* $\vec{\lambda}$

$$\vec{\lambda} = [\lambda_1^1, \ldots, \lambda_1^n, \lambda_2^1, \ldots, \lambda_2^n, \gamma]^T \in \mathbb{R}^{2n+1}.$$

Then all the constraints given by (5.13) can be expressed as a single linear equation

$$M\vec{\lambda} = 0 \tag{5.14}$$

where $M \in \mathbb{R}^{3n \times (2n+1)}$ is a matrix depending on (R, T) and $\{(\mathbf{x}_1^j, \mathbf{x}_2^j)\}_{j=1}^n$. For the linear equation (5.14) to have a unique solution, the matrix M needs to have a rank $2n$. In the absence of noise, the matrix M is generically of rank $2n$ if enough points are used[2], and it has a one-dimensional null space. Hence the equation (5.14) determines all the unknown scales up to a single universal scaling. The linear least-squares estimate of $\vec{\lambda}$ is simply the eigenvector of $M^T M$ which corresponds to the smallest eigenvalue. However, the obtained reconstruction may suffer from certain inconsistency that the 3-D locations of a point recovered from different vantage points do not necessarily coincide. In the next section, we address this issue and, at the mean time, address the issue of optimal reconstruction in the presence of noise.

[2]disregarding, of course, configurations of points which are not in "general position".

5.3 Optimal reconstruction

[3]

The eight-point algorithm to reconstruct camera pose, as described in the previous section, assumes that *exact* point correspondences are given. In the presence of noise in the correspondence, we have hinted at possible ways of estimating the essential matrix by solving a least-squares problem. In this section, we describe a more principled approach to minimize the effects of noise in the reconstruction. It requires solving an optimization problem on a differentiable manifold. Rather than develop the machinery to solve such optimization problems in this section, we will only formulate the problem here and describe an algorithm at a conceptual level. Details of the algorithm as well as more on optimization on manifolds are described in Appendix C.

As we have seen in Chapter 6, a calibrated camera can be described as a plane perpendicular to the Z-axis at a distance of 1 unit from the origin. Therefore, coordinates of image points $\mathbf{x}_1$ and $\mathbf{x}_2$ are of the form $[x, y, 1]^T \in \mathbb{R}^3$. In practice, we cannot measure the actual coordinates but rather their noisy versions

$$\tilde{\mathbf{x}}_1^j = \mathbf{x}_1^j + w_1^j, \quad \tilde{\mathbf{x}}_2^j = \mathbf{x}_2^j + w_2{}^j, \quad j = 1, \dots, n \tag{5.15}$$

where $\mathbf{x}_1^j$ and $\mathbf{x}_2^j$ are the "ideal" image coordinates and $w_1^j = [w_1{}_1^j,\ w_1{}_2^j,\ 0]^T$ and $w_2^j = [w_2{}_1^j,\ w_2{}_2^j,\ 0]^T$ are localization errors in the correspondence. Notice that it is the (unknown) ideal image coordinates $\mathbf{x}_i^j,\ i = 1, 2$ that satisfy the epipolar constraint $\mathbf{x}_2^{jT} \widehat{T} R \mathbf{x}_1^j = 0$, and *not* the (measured) noisy ones $\tilde{\mathbf{x}}_i^j,\ i = 1, 2$. One could think of the ideal coordinates as a "model" that depends upon the unknown parameters (R, T), and w_i^j as the discrepancy between the model and the measurements: $\tilde{\mathbf{x}}_i^j = \mathbf{x}_i^j(R, T) + w_i^j$. Therefore, one seeks the parameters (R, T) that minimize the discrepancy from the data, i.e. w_i^j. It is clear that, in order to proceed, we need to define a discrepancy criterion.

The simplest choice is to assume that w_i^j are unknown errors and to minimize their norm; for instance, the standard Euclidean two-norm $\|w\|_2 \doteq \sqrt{w^T w}$. Indeed, minimizing the squared norm is equivalent and often results in simpler algorithms: $\hat{R}, \hat{T} = \arg\min_{R,T} \phi(R, T)$ where

$$\phi(R, T) \doteq \sum_{i,j} \|w_i^j\|_2^2 = \sum_{i,j} \|\tilde{\mathbf{x}}_i^j - \mathbf{x}_i^j(R, T)\|_2^2.$$

This corresponds to a *"least-squares" criterion* (LS). Notice that the unknowns are constrained to live in a non-linear space: $R \in SO(3)$ and $T \in \mathbb{S}^2$, the unit sphere, after normalization due to the unrecoverable global scale factor, as discussed in the eight-point algorithm.

[3] This section may be skipped at a first reading.

Alternatively, one may assume that w_i^j are samples of a stochastic process with a certain, known distribution that depends upon the unknown parameters (R,T), $w_i^j \sim p(w|R,T)$ and maximize the (log)-likelihood function with respect to the parameters: $\hat{R},\hat{T} = \arg\max_{R,T} \phi(R,T)$, where

$$\phi(R,T) \doteq \sum_{i,j} \log p((\tilde{\mathbf{x}}_i^j - \mathbf{x}_i^j)|R,T).$$

This corresponds to a *"maximum likelihood" criterion* (ML).

If prior information about the unknown parameters (R,T) is available, for instance, their prior density $p(R,T)$, one may combine it with the log-likelihood function to obtain a posterior density $p(R,T|\tilde{\mathbf{x}}_j^j)$ using Bayes' rule, and seek $\hat{R},\hat{T} = \arg\max_{R,T} \phi(R,T)$, where

$$\phi \doteq p(R,T|\tilde{\mathbf{x}}_i^j \ \forall\ i,j)$$

This choice corresponds to a *"maximum a-posteriori" criterion* (MAP). Although conceptually more "principled", this choice requires defining a probability density on the space of camera pose $SO(3)\times\mathbb{S}^2$ which has a non-trivial Riemannian structure. Not only is this prior not easy to obtain, but also its description is beyond the scope of this book and will therefore not be further explored.

Notice that both the ML and LS criteria in the end reduce to optimization problems on the manifold $SO(3)\times\mathbb{S}^2$. The discussion in this section can be applied to ML as well as to LS, although for simplicity we restrict our discussion to LS. Now that we have chosen a criterion, the least-squares norm, we can pose the problem of optimal reconstruction by collecting all available constraints as follows: given $\tilde{\mathbf{x}}_i^j, \quad i=1,2,\ j=1,\dots,n$, find

$$\hat{R},\hat{T} = \arg\min \sum_{j=1}^{n}\sum_{i=1}^{2} \|w_i^j\|_2^2$$

subject to

$$\begin{cases} \tilde{\mathbf{x}}_i^j = \mathbf{x}_i^j + w_i{}^j, \quad i=1,2, j=1,\dots,n \\ \mathbf{x}_2^{jT}\widehat{T}R\mathbf{x}_1^j = 0, \quad j=1,\dots,n \\ \mathbf{x}_1^{jT}e_3 = 1, \quad j=1,\dots,n \\ \mathbf{x}_2^{jT}e_3 = 1, \quad j=1,\dots,n \\ R\in SO(3) \\ T\in\mathbb{S}^2 \end{cases} \tag{5.16}$$

Using Lagrange multipliers $\lambda^j,\gamma^j,\eta^j$, we can convert the above minimization problem to an unconstrained minimization problem over $R\in$

$SO(3), T \in \mathbb{S}^2$, $\mathbf{x}_i^j, \mathbf{x}_2^j$, $\lambda^j, \gamma^j, \eta^j$ with the constraints of equation (5.16) as

$$\min \sum_{j=1}^{n} \|\tilde{\mathbf{x}}_1^j - \mathbf{x}_1^j\|^2 + \|\tilde{\mathbf{x}}_2^j - \mathbf{x}_2^j\|^2 + \lambda^j \mathbf{x}_2^{jT} \widehat{T} R \mathbf{x}_1^j + \gamma^j (\mathbf{x}_1^{jT} e_3 - 1) + \eta^j (\mathbf{x}_2^{jT} e_3 - 1). \tag{5.17}$$

The necessary conditions for the existence of a minimum are:

$$\begin{aligned} 2(\tilde{\mathbf{x}}_1^j - \mathbf{x}_1^j) + \lambda^j R^T \widehat{T}^T \mathbf{x}_2^j + \gamma^j e_3 = 0 \\ 2(\tilde{\mathbf{x}}_2^j - \mathbf{x}_2^j) + \lambda^j \widehat{T} R \mathbf{x}_1^j + \eta^j e_3 = 0 \end{aligned}$$

Simplifying the necessary conditions, we obtain:

$$\begin{cases} \mathbf{x}_1^j &= \tilde{\mathbf{x}}_1^j - \frac{1}{2}\lambda^j \widehat{e}_3^T \widehat{e}_3 R^T \widehat{T}^T \mathbf{x}_2^j \\ \mathbf{x}_2^j &= \tilde{\mathbf{x}}_2^j - \frac{1}{2}\lambda^j \widehat{e}_3^T \widehat{e}_3 \widehat{T} R \mathbf{x}_1^j \\ \mathbf{x}_2^{jT} \widehat{T} R \mathbf{x}_1^j &= 0 \end{cases} \tag{5.18}$$

where λ^j is given by:

$$\lambda^j = \frac{2(\mathbf{x}_2^{jT} \widehat{T} R \tilde{\mathbf{x}}_1^j + \tilde{\mathbf{x}}_2^{jT} \widehat{T} R \mathbf{x}_1^j)}{\mathbf{x}_1^{jT} R^T \widehat{T}^T \widehat{e}_3^T \widehat{e}_3 \widehat{T} R \mathbf{x}_1^j + \mathbf{x}_2^{jT} \widehat{T} R \widehat{e}_3^T \widehat{e}_3 R^T \widehat{T}^T \mathbf{x}_2^j} \tag{5.19}$$

or

$$\lambda^j = \frac{2\mathbf{x}_2^{jT} \widehat{T} R \tilde{\mathbf{x}}_1^j}{\mathbf{x}_1^{jT} R^T \widehat{T}^T \widehat{e}_3^T \widehat{e}_3 \widehat{T} R \mathbf{x}_1^j} = \frac{2\tilde{\mathbf{x}}_2^{jT} \widehat{T} R \mathbf{x}_1^j}{\mathbf{x}_2^{jT} \widehat{T} R \widehat{e}_3^T \widehat{e}_3 R^T \widehat{T}^T \mathbf{x}_2^j}. \tag{5.20}$$

Substituting (5.18) and (5.19) into the least-squares cost function of equation (5.17, we obtain:

$$\phi(R, T, \mathbf{x}_1^j, \mathbf{x}_2^j) = \sum_{j=1}^{n} \frac{(\mathbf{x}_2^{jT} \widehat{T} R \tilde{\mathbf{x}}_1^j + \tilde{\mathbf{x}}_2^{jT} \widehat{T} R \mathbf{x}_1^j)^2}{\|\widehat{e}_3 \widehat{T} R \mathbf{x}_1^j\|^2 + \|\mathbf{x}_2^{jT} \widehat{T} R \widehat{e}_3^T\|^2} \tag{5.21}$$

If one uses instead (5.18) and (5.20), we get:

$$\phi(R, T, \mathbf{x}_1^j, \mathbf{x}_2^j) = \sum_{j=1}^{n} \frac{(\tilde{\mathbf{x}}_2^{jT} \widehat{T} R \mathbf{x}_1^j)^2}{\|\widehat{e}_3 \widehat{T} R \mathbf{x}_1^j\|^2} + \frac{(\mathbf{x}_2^{jT} \widehat{T} \tilde{\mathbf{x}}_1^j)^2}{\|\mathbf{x}_2^{jT} \widehat{T} R \widehat{e}_3^T\|^2}. \tag{5.22}$$

Geometrically, both expressions can be interpreted as distances of the image points $\tilde{\mathbf{x}}_1^j$ and $\tilde{\mathbf{x}}_2^j$ from the epipolar lines specified by $\mathbf{x}_1^j, \mathbf{x}_2^j$ and (R, T), as shown in Figure 5.2. We leave the verification of this to the reader as an exercise.

These expressions for ϕ can finally be minimized with respect to (R, T) as well as $\mathbf{x}_i^j$. In doing so, however, one has to make sure that the optimization proceeds on the space $R \in SO(3)$ and $T \in \mathbb{S}^2$ are enforced. In Appendix C we discuss methods of minimizing function defined on the space $SO(3) \times \mathbb{S}^2$, which can be used to minimize $\phi(R, T, \mathbf{x}_i^j)$ once $\mathbf{x}_i^j$ are known. Since $\mathbf{x}_i^j$ are *not* known, one can set up an alternating minimization scheme where an initial approximation of $\mathbf{x}_i^j$ is used to estimate an approximation of (R, T) which is used, in turn, to update the estimates of $\mathbf{x}_i^j$. It can be shown that

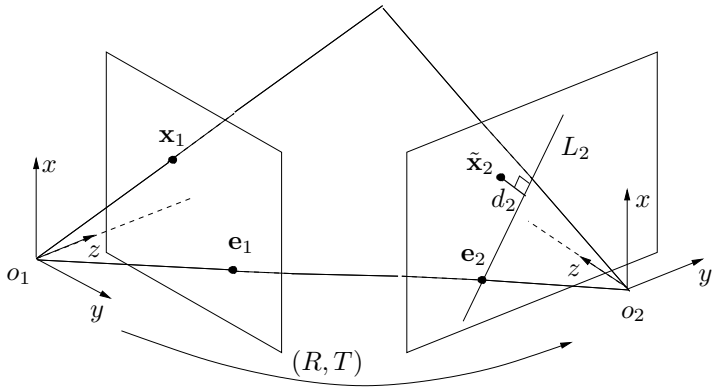

Figure 5.2. Two noisy image points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$. L_2 is the so-called *epipolar line* which is the intersection of the second image plane with the plane formed by the first image point $\mathbf{x}_1$ and the line connecting the two camera centers o_1, o_2. The distance d_2 is the geometric distance between the second image point $\tilde{\mathbf{x}}_2$ and the epipolar line in the second image plane. Symmetrically, one can define a similar geometric distance d_1 in the first image plane.

each such iteration decreases the cost function, and therefore convergence to a local extremum is guaranteed since the cost function is bounded below by zero.

Algorithm 5.2 (Optimal triangulation).

1. **Initialization**
   Initialize $\mathbf{x}_1^j(R,T)$ *and* $\mathbf{x}_2^j(R,T)$ *as* $\tilde{\mathbf{x}}_1^j$ *and* $\tilde{\mathbf{x}}_2^j$ *respectively.*

2. **Motion estimation**
   Update (R,T) *by minimizing* $\phi(R,T,\mathbf{x}_1^j(R,T),\mathbf{x}_2^j(R,T))$ *given by (5.21) or (5.22).*

3. **Structure triangulation**
   Solve for $\mathbf{x}_1^j(R,T)$ *and* $\mathbf{x}_2^j(R,T)$ *which minimize the objective function* ϕ *with respect to a fixed* (R,T) *computed from the previous step.*

4. **Return** *to Step 2 until the decrement in the value of* ϕ *is below a threshold.*

In step 3 above, for a fixed (R,T), $\mathbf{x}_1^j(R,T)$ and $\mathbf{x}_2^j(R,T)$ can be computed by minimizing the distance $\|\mathbf{x}_1^j - \tilde{\mathbf{x}}_1^j\|^2 + \|\mathbf{x}_2^j - \tilde{\mathbf{x}}_2^j\|^2$ for each pair of image points. Let $t_2^j \in \mathbb{R}^3$ be the normal vector (of unit length) to the (epipolar) plane spanned by $(\mathbf{x}_2^j, T)$. Given such a t_2^j, $\mathbf{x}_1^j$ and $\mathbf{x}_2^j$ are

determined by:

$$\mathbf{x}_1^j(t_1^j) = \frac{\widehat{e}_3 t_1^j t_1^{jT} \widehat{e}_3^T \tilde{\mathbf{x}}_1^j + \widehat{t_1^j}^T \widehat{t_1^j} e_3}{e_3^T \widehat{t_1^j}^T \widehat{t_1^j} e_3}, \quad \mathbf{x}_2^j(t_2^j) = \frac{\widehat{e}_3 t_2^j t_2^{jT} \widehat{e}_3^T \tilde{\mathbf{x}}_2^j + \widehat{t_2^j}^T \widehat{t_2^j} e_3}{e_3^T \widehat{t_2^j}^T \widehat{t_2^j} e_3},$$

where $t_1^j = R^T t_2^j \in \mathbb{R}^3$. Then the distance can be explicitly expressed as:

$$\|\mathbf{x}_2^j - \tilde{\mathbf{x}}_2^j\|^2 + \|\mathbf{x}_1^j - \tilde{\mathbf{x}}_1^j\|^2 \quad = \quad \|\tilde{\mathbf{x}}_2^j\|^2 + \frac{t_2^{jT} A^j t_2^j}{t_2^{jT} B^j t_2^j} + \|\tilde{\mathbf{x}}_1^j\|^2 + \frac{t_1^{jT} C^j t_1^j}{t_1^{jT} D^j t_1^j},$$

where $A^j, B^j, C^j, D^j \in \mathbb{R}^{3\times 3}$ are defined by:

$$\begin{array}{rclcl} A^j & = & I - (\widehat{e}_3 \tilde{\mathbf{x}}_2^j \tilde{\mathbf{x}}_2^{jT} \widehat{e}_3^T + \widehat{\mathbf{x}_2^j} \widehat{e}_3 + \widehat{e}_3 \widehat{\mathbf{x}_2^j}), & B^j = \widehat{e}_3^T \widehat{e}_3 \\ C^j & = & I - (\widehat{e}_3 \tilde{\mathbf{x}}_1^j \tilde{\mathbf{x}}_1^{jT} \widehat{e}_3^T + \widehat{\mathbf{x}_1^j} \widehat{e}_3 + \widehat{e}_3 \widehat{\mathbf{x}_1^j}), & D^j = \widehat{e}_3^T \widehat{e}_3 \end{array} . \tag{5.23}$$

Then the problem of finding $\mathbf{x}_1^j(R,T)$ and $\mathbf{x}_2^j(R,T)$ becomes one of finding t_2^{j*} which minimizes the function of a sum of two *singular Rayleigh quotients*:

$$\min_{t_2^{jT} T = 0, t_2^{jT} t_2^j = 1} V(t_2^j) \quad = \quad \frac{t_2^{jT} A^j t_2^j}{t_2^{jT} B^j t_2^j} + \frac{t_2^{jT} R C^j R^T t_2^j}{t_2^{jT} R D^j R^T t_2^j}. \tag{5.24}$$

This is an optimization problem on the unit circle $\mathbb{S}^1$ in the plane orthogonal to the vector T (therefore, geometrically, motion and structure recovery from n pairs of image correspondences is an optimization problem on the space $SO(3) \times \mathbb{S}^2 \times \mathbb{T}^n$ where $\mathbb{T}^n$ is an n-torus, i.e. an n-fold product of $\mathbb{S}^1$). If $N_1, N_2 \in \mathbb{R}^3$ are vectors such that T, N_1, N_2 form an orthonormal basis of $\mathbb{R}^3$, then $t_2^j = \cos(\theta) N_1 + \sin(\theta) N_2$ with $\theta \in \mathbb{R}$. We only need to find θ^* which minimizes the function $V(t_2^j(\theta))$. From the geometric interpretation of the optimal solution, we also know that the global minimum θ^* should lie between two values: θ_1 and θ_2 such that $t_2^j(\theta_1)$ and $t_2^j(\theta_2)$ correspond to normal vectors of the two planes spanned by $(\tilde{\mathbf{x}}_2^j, T)$ and $(R\tilde{\mathbf{x}}_1^j, T)$ respectively (if $\tilde{\mathbf{x}}_1^j, \tilde{\mathbf{x}}_2^j$ are already triangulated, these two planes coincide). The problem now becomes a simple bounded minimization problem for a scalar function and can be efficiently solved using standard optimization routines (such as "fmin" in Matlab or the Newton's algorithm).

5.4 Continuous case

As we pointed out in Section 5.1, the limit case where the two viewpoints are infinitesimally close requires extra attention. From the practical standpoint, this case is relevant to the analysis of a video stream where the camera motion is slow relative to the sampling frequency. In this section, we follow the steps of the previous section by giving a parallel derivation

of the geometry of points in space as seen from a moving camera, and deriving a conceptual algorithm for reconstructing camera motion and scene structure. In light of the fact that the camera motion is slow relative to the sampling frequency we will treat the motion of the camera as continuous. While the derivations proceed in parallel it is an important point of caution to the reader that there are some subtle differences.

5.4.1 Continuous epipolar constraint and the continuous essential matrix

Let us assume that camera motion is described by a smooth (i.e. continuously differentiable) trajectory $g(t) = (R(t), T(t)) \in SE(3)$ with body velocities $(\omega(t), v(t)) \in se(3)$ as defined in Chapter 2 For a point $p \in \mathbb{R}^3$, its coordinates as a function of time $\mathbf{X}(t)$ satisfy

$$\dot{\mathbf{X}}(t) = \widehat{\omega}(t)\mathbf{X}(t) + v(t). \tag{5.25}$$

From now on, for convenience, we will drop the time-dependency from the notation. The image of the point p taken by the camera is the vector $\mathbf{x}$ which satisfies $\lambda\mathbf{x} = \mathbf{X}$. Denote the velocity of the image point $\mathbf{x}$ by $\mathbf{u} = \dot{\mathbf{x}} \in \mathbb{R}^3$. $\mathbf{u}$ is also called *image motion field*, which under the brightness constancy assumption discussed in Chapter 3 can be approximated by the *optical flow*.

Consider now the inner product of the vectors in (5.25) with the vector $(v \times \mathbf{x})$

$$\dot{\mathbf{X}}^T(v \times \mathbf{x}) = (\widehat{\omega}\mathbf{X} + v)^T(v \times \mathbf{x}) = \mathbf{X}^T\widehat{\omega}^T\widehat{v}\mathbf{x}. \tag{5.26}$$

Further note that

$$\dot{\mathbf{X}} = \dot{\lambda}\mathbf{x} + \lambda\dot{\mathbf{x}} \quad \text{and} \quad \mathbf{x}^T(v \times \mathbf{x}) = 0.$$

Using this in equation (5.26), we have

$$\lambda\dot{\mathbf{x}}^T\widehat{v}\mathbf{x} - \lambda\mathbf{x}^T\widehat{\omega}^T\widehat{v}\mathbf{x} = 0.$$

When $\lambda \neq 0$, we obtain a constraint that plays the role of the epipolar constraint for the case of continuous-time images, in the sense that it does not depend upon the position of the points in space, but only on their projection and the motion parameters:

$$\mathbf{u}^T\widehat{v}\mathbf{x} + \mathbf{x}^T\widehat{\omega}\widehat{v}\mathbf{x} = 0. \tag{5.27}$$

Before proceeding, we state a lemma that will become useful in the remainder of this section.

Lemma 5.3. *Consider the matrices $M_1, M_2 \in \mathbb{R}^{3\times3}$. Then $\mathbf{x}^T M_1\mathbf{x} = \mathbf{x}^T M_2\mathbf{x}$ for all $\mathbf{x} \in \mathbb{R}^3$ if and only if $M_1 - M_2$ is a skew symmetric matrix, i.e. $M_1 - M_2 \in so(3)$.*

We leave the proof of this lemma as an exercise. Following the lemma, for any skew symmetric matrix $M \in \mathbb{R}^{3\times 3}$, $\mathbf{x}^T M \mathbf{x} = 0$. Since $\frac{1}{2}(\widehat{\omega}\widehat{v} - \widehat{v}\widehat{\omega})$ is a skew symmetric matrix, $\mathbf{x}^T \frac{1}{2}(\widehat{\omega}\widehat{v} - \widehat{v}\widehat{\omega})\mathbf{x} = 0$. If we define the *symmetric epipolar component* to be the following matrix

$$s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega}) \quad \in \mathbb{R}^{3\times 3}.$$

then we have

$$\mathbf{u}^T \widehat{v}\mathbf{x} + \mathbf{x}^T s \mathbf{x} = 0. \tag{5.28}$$

This equation suggests some redundancy in the continuous epipolar constraint (5.27). Indeed, the matrix $\widehat{\omega}\widehat{v}$ can only be recovered up to its symmetric component $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$[4]. This structure is substantially different from the discrete case, and it cannot be derived as a first-order approximation of the essential matrix $\widehat{T}R$. In fact a naive discretization would lead to a matrix of the form $\widehat{v}\widehat{\omega}$, whereas in the true continuous case we have to deal with only its symmetric component $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$! The set of interest in this case is the space of 6×3 matrices of the form

$$\mathcal{E}' = \left\{ \left[\begin{array}{c} \widehat{v} \\ \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega}) \end{array} \right] \middle| \omega, v \in \mathbb{R}^3 \right\} \quad \subset \mathbb{R}^{6\times 3}$$

which we call the *continuous essential space*. A matrix in this space is called a *continuous essential matrix*. Note that the continuous epipolar constraint (5.28) is homogeneous in the linear velocity v. Thus v may be recovered only up to a constant scale. Consequently, in motion recovery, we will concern ourselves with matrices belonging to the *normalized continuous essential space* with v normalized to be 1:

$$\mathcal{E}'_1 = \left\{ \left[\begin{array}{c} \widehat{v} \\ \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega}) \end{array} \right] \middle| \omega \in \mathbb{R}^3, v \in \mathbb{S}^2 \right\} \quad \subset \mathbb{R}^{6\times 3}.$$

5.4.2 Properties of continuous essential matrices

The skew-symmetric part of a continuous essential matrix simply corresponds to the velocity v. The characterization of the (normalized) essential matrix only focuses on its part $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. We call the space of all the matrices of this form the *symmetric epipolar space*

$$\mathcal{S} = \left\{ \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega}) \middle| \omega \in \mathbb{R}^3, v \in \mathbb{S}^2 \right\} \subset \mathbb{R}^{3\times 3}.$$

[4] This redundancy is the very reason why different forms of the continuous epipolar constraint exist in the literature [ZH84, PG98, VF95, May93, BCB97], and, accordingly, various approaches have been proposed to recover ω and v (see [TTH96]).

A matrix in this space is called a *symmetric epipolar component*. The motion estimation problem is now reduced to the one of *recovering the velocity* (ω, v) *with* $\omega \in \mathbb{R}^3$ *and* $v \in \mathbb{S}^2$ *from a given symmetric epipolar component* s.

The characterization of symmetric epipolar components depends on a characterization of matrices of the form $\widehat{\omega}\widehat{v} \in \mathbb{R}^{3\times 3}$, which is given in the following lemma. We define the matrix $R_Y(\theta)$ to be the rotation around the Y-axis by an angle $\theta \in \mathbb{R}$, i.e. $R_Y(\theta) = e^{\widehat{e}_2\theta}$ with $e_2 = [0, 1, 0]^T \in \mathbb{R}^3$.

Lemma 5.4. *A matrix* $Q \in \mathbb{R}^{3\times 3}$ *has the form* $Q = \widehat{\omega}\widehat{v}$ *with* $\omega \in \mathbb{R}^3$, $v \in \mathbb{S}^2$ *if and only if*

$$Q = -VR_Y(\theta)\,diag\{\lambda, \lambda\cos(\theta), 0\}V^T \tag{5.29}$$

for some rotation matrix $V \in SO(3)$. *the positive scalar* $\lambda = \|\omega\|$ *and* $\cos(\theta) = \omega^T v/\lambda$.

Proof. We first prove the necessity. The proof follows from the geometric meaning of $\widehat{\omega}\widehat{v}$ for any vector $q \in \mathbb{R}^3$,

$$\widehat{\omega}\widehat{v}q = \omega \times (v \times q).$$

Let $b \in \mathbb{S}^2$ be the unit vector perpendicular to both ω and v. Then, $b = \frac{v\times\omega}{\|v\times\omega\|}$ (if $v \times \omega = 0$, b is not uniquely defined. In this case, ω, v are parallel and the rest of the proof follows if one picks any vector b orthogonal to v and ω). Then $\omega = \lambda\exp(\widehat{b}\theta)v$ (according this definition, θ is the angle between ω and v, and $0 \le \theta \le \pi$). It is direct to check that if the matrix V is defined to be

$$V = (e^{\widehat{b}\frac{\pi}{2}}v, b, v),$$

then Q has the given form (5.29).

We now prove the sufficiency. Given a matrix Q which can be decomposed into the form (5.29), define the orthogonal matrix $U = -VR_Y(\theta) \in O(3)$.[5] Let the two skew symmetric matrices $\widehat{\omega}$ and $\widehat{v}$ given by the formulae

$$\widehat{\omega} = UR_Z(\pm\frac{\pi}{2})\Sigma_\lambda U^T, \quad \widehat{v} = VR_Z(\pm\frac{\pi}{2})\Sigma_1 V^T \tag{5.30}$$

where $\Sigma_\lambda = \mathrm{diag}\{\lambda, \lambda, 0\}$ and $\Sigma_1 = \mathrm{diag}\{1, 1, 0\}$. Then

$$\begin{aligned}
\widehat{\omega}\widehat{v} &= UR_Z(\pm\frac{\pi}{2})\Sigma_\lambda U^T VR_Z(\pm\frac{\pi}{2})\Sigma_1 V^T \\
&= UR_Z(\pm\frac{\pi}{2})\Sigma_\lambda(-R_Y^T(\theta))R_Z(\pm\frac{\pi}{2})\Sigma_1 V^T \\
&= U\mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\}V^T \\
&= Q.
\end{aligned} \tag{5.31}$$

Since ω and v have to be, respectively, the left and the right zero eigenvectors of Q, the reconstruction given in (5.30) is unique. □

[5] $O(3)$ represents the space of all orthogonal matrices (of determinant ± 1.)

The following theorem reveals the structure of the symmetric epipolar component.

Theorem 5.4 (Characterization of the symmetric epipolar component). *A real symmetric matrix $s \in \mathbb{R}^{3\times 3}$ is a symmetric epipolar component if and only if s can be diagonalized as $s = V\Sigma V^T$ with $V \in SO(3)$ and*

$$\Sigma = diag\{\sigma_1, \sigma_2, \sigma_3\}$$

with $\sigma_1 \geq 0, \sigma_3 \leq 0$ and $\sigma_2 = \sigma_1 + \sigma_3$.

Proof. We first prove the necessity. Suppose s is a symmetric epipolar component, there exist $\omega \in \mathbb{R}^3, v \in \mathbb{S}^2$ such that $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. Since s is a symmetric matrix, it is diagonalizable, all its eigenvalues are real and all the eigenvectors are orthogonal to each other. It then suffices to check that its eigenvalues satisfy the given conditions.

Let the unit vector b and the rotation matrix V be the same as in the proof of Lemma 5.4, so are θ and γ. Then according to the lemma, we have

$$\widehat{\omega}\widehat{v} = -VR_Y(\theta)\mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\}V^T.$$

Since $(\widehat{\omega}\widehat{v})^T = \widehat{v}\widehat{\omega}$, it yields

$$s = \frac{1}{2}V\left(-R_Y(\theta)\mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\} - \mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\}R_Y^T(\theta)\right)V^T.$$

Define the matrix $D(\lambda, \theta) \in \mathbb{R}^{3\times 3}$ to be

$$\begin{aligned} D(\lambda,\theta) &= -R_Y(\theta)\mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\} - \mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\}R_Y^T(\theta) \\ &= \lambda\begin{bmatrix} -2\cos(\theta) & 0 & \sin(\theta) \\ 0 & -2\cos(\theta) & 0 \\ \sin(\theta) & 0 & 0 \end{bmatrix}. \end{aligned}$$

Directly calculating its eigenvalues and eigenvectors, we obtain that $D(\lambda, \theta)$ is equal to

$$R_Y(\frac{\theta-\pi}{2})\mathrm{diag}\left\{\lambda(1-\cos(\theta)), -2\lambda\cos(\theta), \lambda(-1-\cos(\theta))\right\}R_Y^T(\frac{\theta-\pi}{2}) \tag{5.32}$$

Thus $s = \frac{1}{2}VD(\lambda, \theta)V^T$ has eigenvalues

$$\left\{\frac{1}{2}\lambda(1-\cos(\theta)), \quad -\lambda\cos(\theta), \quad \frac{1}{2}\lambda(-1-\cos(\theta))\right\}, \tag{5.33}$$

which satisfy the given conditions.

We now prove the sufficiency. Given $s = V_1\mathrm{diag}\{\sigma_1, \sigma_2, \sigma_3\}V_1^T$ with $\sigma_1 \geq 0, \sigma_3 \leq 0$ and $\sigma_2 = \sigma_1 + \sigma_3$ and $V_1^T \in SO(3)$, these three eigenvalues uniquely determine $\lambda, \theta \in \mathbb{R}$ such that the σ_i's have the form given in

(5.33)

$$\begin{cases} \lambda &= \sigma_1 - \sigma_3, & \lambda \geq 0 \\ \theta &= \arccos(-\sigma_2/\lambda), & \theta \in [0, \pi] \end{cases}$$

Define a matrix $V \in SO(3)$ to be $V = V_1 R_Y^T \left(\frac{\theta}{2} - \frac{\pi}{2}\right)$. Then $s = \frac{1}{2} V D(\lambda, \theta) V^T$. According to Lemma 5.4, there exist vectors $v \in \mathbb{S}^2$ and $\omega \in \mathbb{R}^3$ such that

$$\widehat{\omega}\widehat{v} = -V R_Y(\theta) \mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\} V^T.$$

Therefore, $\frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega}) = \frac{1}{2} V D(\lambda, \theta) V^T = s$. □

Figure 5.3 gives a geometric interpretation of the three eigenvectors of the symmetric epipolar component s for the case when both ω, v are of unit length. Theorem 5.4 was given as an exercise problem in Kanatani

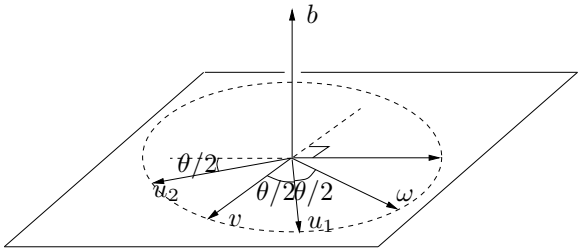

Figure 5.3. Vectors u_1, u_2, b are the three eigenvectors of a symmetric epipolar component $\frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. In particular, b is the normal vector to the plane spanned by ω and v, and u_1, u_2 are both in this plane. u_1 is the average of ω and v. u_2 is orthogonal to both b and u_1.

[Kan93] but it has never been really exploited in the literature for designing algorithms. The constructive proof given above is important in this regard, since it gives an explicit decomposition of the symmetric epipolar component s, which will be studied in more detail next.

Following the proof of Theorem 5.4, if we already know the eigenvector decomposition of a symmetric epipolar component s, we certainly can find at least one solution (ω, v) such that $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. We now discuss uniqueness, i.e. how many solutions exist for $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$.

Theorem 5.5 (Velocity recovery from the symmetric epipolar component). *There exist exactly four 3-D velocities (ω, v) with $\omega \in \mathbb{R}^3$ and $v \in \mathbb{S}^2$ corresponding to a non-zero $s \in \mathcal{S}$.*

Proof. Suppose (ω_1, v_1) and (ω_2, v_2) are both solutions for $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. Then we have

$$\widehat{v}_1\widehat{\omega}_1 + \widehat{\omega}_1\widehat{v}_1 = \widehat{v}_2\widehat{\omega}_2 + \widehat{\omega}_2\widehat{v}_2. \tag{5.34}$$

From Lemma 5.4, we may write

$$\begin{cases} \widehat{\omega}_1\widehat{v}_1 &= -V_1 R_Y(\theta_1)\mathrm{diag}\{\lambda_1, \lambda_1\cos(\theta_1), 0\}V_1^T \\ \widehat{\omega}_2\widehat{v}_2 &= -V_2 R_Y(\theta_2)\mathrm{diag}\{\lambda_2, \lambda_2\cos(\theta_2), 0\}V_2^T. \end{cases} \tag{5.35}$$

Let $W = V_1^T V_2 \in SO(3)$, then from (5.34)

$$D(\lambda_1, \theta_1) = W D(\lambda_2, \theta_2) W^T. \tag{5.36}$$

Since both sides of (5.36) have the same eigenvalues, according to (5.32), we have

$$\lambda_1 = \lambda_2, \quad \theta_2 = \theta_1.$$

We can then denote both θ_1 and θ_2 by θ. It is immediate to check that the only possible rotation matrix W which satisfies (5.36) is given by $I_{3\times 3}$ or

$$\begin{bmatrix} -\cos(\theta) & 0 & \sin(\theta) \\ 0 & -1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad or \quad \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & -1 & 0 \\ -\sin(\theta) & 0 & -\cos(\theta) \end{bmatrix}.$$

From the geometric meaning of V_1 and V_2, all the cases give either $\widehat{\omega}_1\widehat{v}_1 = \widehat{\omega}_2\widehat{v}_2$ or $\widehat{\omega}_1\widehat{v}_1 = \widehat{v}_2\widehat{\omega}_2$. Thus, according to the proof of Lemma 5.4, if (ω, v) is one solution and $\widehat{\omega}\widehat{v} = U\mathrm{diag}\{\lambda, \lambda\cos(\theta), 0\}V^T$, then all the solutions are given by

$$\begin{cases} \widehat{\omega} &= UR_Z(\pm\frac{\pi}{2})\Sigma_\lambda U^T, \quad \widehat{v} = VR_Z(\pm\frac{\pi}{2})\Sigma_1 V^T; \\ \widehat{\omega} &= VR_Z(\pm\frac{\pi}{2})\Sigma_\lambda V^T, \quad \widehat{v} = UR_Z(\pm\frac{\pi}{2})\Sigma_1 U^T \end{cases} \tag{5.37}$$

where $\Sigma_\lambda = \mathrm{diag}\{\lambda, \lambda, 0\}$ and $\Sigma_1 = \mathrm{diag}\{1, 1, 0\}$. □

Given a non-zero continuous essential matrix $E \in \mathcal{E}'$, according to (5.37), its symmetric component gives four possible solutions for the 3-D velocity (ω, v). However, in general only one of them has the same linear velocity v as the skew symmetric part of E. Hence, compared to the discrete case where there are two 3-D motions (R, T) associated to an essential matrix, the velocity (ω, v) corresponding to a continuous essential matrix is unique. This is because, in the continuous case, the so-called *twisted-pair ambiguity* which occurs in discrete case and is caused by a 180° rotation of the camera around the translation direction (see Maybank [May93]), is now avoided.

5.4.3 The eight-point linear algorithm

Based on the preceding study of the continuous essential matrix, this section describes an algorithm to recover the 3-D velocity of the camera from a set of (possibly noisy) optical flows.

Let $E = \begin{bmatrix} \widehat{v} \\ s \end{bmatrix} \in \mathcal{E}_1'$ with $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$ be the essential matrix associated with the continuous epipolar constraint (5.28). Since the submatrix $\widehat{v}$ is skew symmetric and s is symmetric, they have the following

form

$$\widehat{v} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \quad s = \begin{bmatrix} s_1 & s_2 & s_3 \\ s_2 & s_4 & s_5 \\ s_3 & s_5 & s_6 \end{bmatrix}. \tag{5.38}$$

Define the (continuous) *essential vector* $\mathbf{e} \in \mathbb{R}^9$ to be

$$\mathbf{e} = [v_1, v_2, v_3, s_1, s_2, s_3, s_4, s_5, s_6]^T. \tag{5.39}$$

Define a vector $\mathbf{a} \in \mathbb{R}^9$ associated to optical flow $(\mathbf{x}, \mathbf{u})$ with $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3, \mathbf{u} = [u_1, u_2, u_3]^T \in \mathbb{R}^3$ to be[6]

$$\mathbf{a} = [u_3 y - u_2 z, u_1 z - u_3 x, u_2 x - u_1 y, x^2, 2xy, 2xz, y^2, 2yz, z^2]^T. \tag{5.40}$$

The continuous epipolar constraint (5.28) can be then rewritten as

$$\mathbf{a}^T \mathbf{e} = 0.$$

Given a set of (possibly noisy) optical flow vectors $(\mathbf{x}^j, \mathbf{u}^j)$, $j = 1, \ldots, n$ generated by the same motion, define a matrix $A \in \mathbb{R}^{n \times 9}$ associated to these measurements to be

$$A = [\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^n]^T \tag{5.41}$$

where $\mathbf{a}^j$ are defined for each pair $(\mathbf{x}^j, \mathbf{u}^j)$ using (5.40). In the absence of noise, the essential vector $\mathbf{e}$ has to satisfy

$$A\mathbf{e} = 0. \tag{5.42}$$

In order for this equation to have a unique solution for $\mathbf{e}$, the rank of the matrix A has to be eight. Thus, *for this algorithm, the optical flow vectors of at least eight points are needed to recover the 3-D velocity, i.e.* $n \geq 8$, although the minimum number of optical flows needed is actually 5 (see Maybank [May93]).

When the measurements are noisy, there might be no solution of $\mathbf{e}$ for $A\mathbf{e} = 0$. As in the discrete case, one may approximate the solution my minimizing the error function $\|A\mathbf{e}\|^2$.

Since the continuous essential vector $\mathbf{e}$ is recovered from noisy measurements, the symmetric part s of E directly recovered from $\mathbf{e}$ is not necessarily a symmetric epipolar component. Thus one can not directly use the previously derived results for symmetric epipolar components to recover the 3-D velocity. Similarly to what we have done for the discrete case, we can first estimate the symmetric matrix s, and then project it onto the space of symmetric epipolar components.

Theorem 5.6 (Projection to the symmetric epipolar space). *If a real symmetric matrix* $F \in \mathbb{R}^{3 \times 3}$ *is diagonalized as* $F = V\, diag\{\lambda_1, \lambda_2, \lambda_3\} V^T$

[6] For perspective projection, $z = 1$ and $u_3 = 0$ thus the expression for $\mathbf{a}$ can be simplified.

with $V \in SO(3)$, $\lambda_1 \geq 0, \lambda_3 \leq 0$ and $\lambda_1 \geq \lambda_2 \geq \lambda_3$, then the symmetric epipolar component $E \in \mathcal{S}$ which minimizes the error $\|E-F\|_f^2$ is given by $E = V\,diag\{\sigma_1, \sigma_2, \sigma_2\}V^T$ with

$$\sigma_1 = \frac{2\lambda_1 + \lambda_2 - \lambda_3}{3}, \quad \sigma_2 = \frac{\lambda_1 + 2\lambda_2 + \lambda_3}{3}, \quad \sigma_3 = \frac{2\lambda_3 + \lambda_2 - \lambda_1}{3}. \tag{5.43}$$

Proof. Define $\mathcal{S}_\Sigma$ to be the subspace of $\mathcal{S}$ whose elements have the same eigenvalues $\Sigma = \text{diag}\{\sigma_1, \sigma_2, \sigma_3\}$. Thus every matrix $E \in \mathcal{S}_\Sigma$ has the form $E = V_1 \Sigma V_1^T$ for some $V_1 \in SO(3)$. To simplify the notation, define $\Sigma_\lambda = \text{diag}\{\lambda_1, \lambda_2, \lambda_3\}$. We now prove this theorem by two steps.

Step 1: We prove that the matrix $E \in \mathcal{S}_\Sigma$ which minimizes the error $\|E-F\|_f^2$ is given by $E = V\Sigma V^T$. Since $E \in \mathcal{S}_\Sigma$ has the form $E = V_1 \Sigma V_1^T$, we get

$$\|E - F\|_f^2 \;=\; \|V_1 \Sigma V_1^T - V \Sigma_\lambda V^T\|_f^2 = \|\Sigma_\lambda - V^T V_1 \Sigma V_1^T V\|_f^2.$$

Define $W = V^T V_1 \in SO(3)$ and W has the form

$$W = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}. \tag{5.44}$$

Then

$$\begin{aligned} \|E - F\|_f^2 &= \|\Sigma_\lambda - W \Sigma W^T\|_f^2 \\ &= tr(\Sigma_\lambda^2) - 2tr(W \Sigma W^T \Sigma_\lambda) + tr(\Sigma^2). \end{aligned} \tag{5.45}$$

Substituting (5.44) into the second term, and using the fact that $\sigma_2 = \sigma_1 + \sigma_3$ and W is a rotation matrix, we get

$$\begin{aligned} tr(W \Sigma W^T \Sigma_\lambda) &= \sigma_1(\lambda_1(1 - w_3^2) + \lambda_2(1 - w_6^2) + \lambda_3(1 - w_9^2)) \\ &+ \sigma_3(\lambda_1(1 - w_1^2) + \lambda_2(1 - w_4^2) + \lambda_3(1 - w_7^2)) \end{aligned} \tag{5.46}$$

Minimizing $\|E - F\|_f^2$ is equivalent to maximizing $tr(W \Sigma W^T \Sigma_\lambda)$. From (5.46), $tr(W \Sigma W^T \Sigma_\lambda)$ is maximized if and only if $w_3 = w_6 = 0$, $w_9^2 = 1$, $w_4 = w_7 = 0$ and $w_1^2 = 1$. Since W is a rotation matrix, we also have $w_2 = w_8 = 0$ and $w_5^2 = 1$. All possible W give a unique matrix in $\mathcal{S}_\Sigma$ which minimizes $\|E - F\|_f^2$: $E = V \Sigma V^T$.

Step 2: From step one, we only need to minimize the error function over the matrices which have the form $V \Sigma V^T \in \mathcal{S}$. The optimization problem is then converted to one of minimizing the error function

$$\|E - F\|_f^2 = (\lambda_1 - \sigma_1)^2 + (\lambda_2 - \sigma_2)^2 + (\lambda_3 - \sigma_3)^2$$

subject to the constraint

$$\sigma_2 = \sigma_1 + \sigma_3.$$

The formula (5.43) for $\sigma_1, \sigma_2, \sigma_3$ are directly obtained from solving this minimization problem. □

Remark 5.1. *For symmetric matrices which do not satisfy conditions $\lambda_1 \geq 0$ or $\lambda_3 \leq 0$, one may simply choose $\lambda_1' = max(\lambda_1, 0)$ or $\lambda_3' = min(\lambda_3, 0)$.*

Finally, we can outline an eigenvalue decomposition-based algorithm for estimating 3-D velocity from optical flow, which serves as a continuous counterpart of the eight-point algorithm given in Section 5.1.

Algorithm 5.3 (The continuous eight-point algorithm). *For a given set of images and optical flow vectors $(\mathbf{x}^j, \mathbf{u}^j)$, $j = 1, \ldots, n$, this algorithm finds $(\omega, v) \in SE(3)$ which solves*

$$\mathbf{u}^{jT} \widehat{v} \mathbf{x}^j + \mathbf{x}^{jT} \widehat{\omega} \widehat{v} \mathbf{x}^j = 0, \quad j = 1, \ldots, n.$$

1. **Estimate essential vector**
   Define a matrix $A \in \mathbb{R}^{n \times 9}$ whose j^{th} row is constructed from $\mathbf{x}^j$ and $\mathbf{u}^j$ as in (5.40). Use the SVD to find the vector $\mathbf{e} \in \mathbb{R}^9$ such that $A\mathbf{e} = 0$: $A = U_A \Sigma_A V_A^T$ and $\mathbf{e} = V(:, 9)$. Recover the vector $v_0 \in \mathbb{S}^2$ from the first three entries of $\mathbf{e}$ and a symmetric matrix $s \in \mathbb{R}^{3\times 3}$ from the remaining six entries as in (5.39).[7]

2. **Recover the symmetric epipolar component**
   Find the eigenvalue decomposition of the symmetric matrix s

   $$s = V_1 \, diag\{\lambda_1, \lambda_2, \lambda_3\} V_1^T$$

   with $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Project the symmetric matrix s onto the symmetric epipolar space $\mathcal{S}$. We then have the new $s = V_1 \, diag\{\sigma_1, \sigma_2, \sigma_3\} V_1^T$ with

   $$\sigma_1 = \frac{2\lambda_1 + \lambda_2 - \lambda_3}{3}, \quad \sigma_2 = \frac{\lambda_1 + 2\lambda_2 + \lambda_3}{3}, \quad \sigma_3 = \frac{2\lambda_3 + \lambda_2 - \lambda_1}{3};$$

3. **Recover velocity from the symmetric epipolar component**
   Define

   $$\begin{cases} \lambda &= \sigma_1 - \sigma_3, \quad \lambda \geq 0, \\ \theta &= \arccos(-\sigma_2/\lambda), \quad \theta \in [0, \pi]. \end{cases}$$

   Let $V = V_1 R_Y^T \left(\frac{\theta}{2} - \frac{\pi}{2}\right) \in SO(3)$ and $U = -V R_Y(\theta) \in O(3)$. Then the four possible 3-D velocities corresponding to the matrix s are given by

   $$\begin{cases} \widehat{\omega} &= U R_Z(\pm\frac{\pi}{2}) \Sigma_\lambda U^T, \quad \widehat{v} = V R_Z(\pm\frac{\pi}{2}) \Sigma_1 V^T \\ \widehat{\omega} &= V R_Z(\pm\frac{\pi}{2}) \Sigma_\lambda V^T, \quad \widehat{v} = U R_Z(\pm\frac{\pi}{2}) \Sigma_1 U^T \end{cases}$$

   where $\Sigma_\lambda = diag\{\lambda, \lambda, 0\}$ and $\Sigma_1 = diag\{1, 1, 0\}$;

[7] In order to guarantee v_0 to be of unit length, one needs to "re-normalize" $\mathbf{e}$, i.e. to multiply $\mathbf{e}$ with a scalar such that the 3-D vector determined by the first three entries of $\mathbf{e}$ is of unit length.

4. **Recover velocity from the continuous essential matrix**
From the four velocities recovered from the matrix s in step 3, choose the pair (ω^*, v^*) *which satisfies*

$$v^{*T} v_0 = \max_i v_i^T v_0.$$

Then the estimated 3-D velocity (ω, v) *with* $\omega \in \mathbb{R}^3$ *and* $v \in \mathbb{S}^2$ *is given by*

$$\omega = \omega^*, \quad v = v_0.$$

Remark 5.2. *Since both* $E, -E \in \mathcal{E}_1'$ *satisfy the same set of continuous epipolar constraints, both* $(\omega, \pm v)$ *are possible solutions for the given set of optical flows. However, as in the discrete case, one can get rid of the ambiguous solution by enforcing the "positive depth constraint".*

In situations where the motion of the camera is partially constrained, the above linear algorithm can be further simplified. The following example illustrates how it can be done.

Example 5.1 (Kinematic model of an aircraft). *This example shows how to utilize the so called nonholonomic constraints (see Murray, Li and Sastry [MLS94]) to simplify the proposed linear motion estimation algorithm in the continuous case. Let* $g(t) \in SE(3)$ *represent the position and orientation of an aircraft relative to the spatial frame, the inputs* $\omega_1, \omega_2, \omega_3 \in \mathbb{R}$ *stand for the rates of the rotation about the axes of the aircraft and* $v_1 \in \mathbb{R}$ *the velocity of the aircraft. Using the standard homogeneous representation for g (see Murray, Li and Sastry [MLS94]), the kinematic equations of the aircraft motion are given by*

$$\dot{g} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 & v_1 \\ \omega_3 & 0 & -\omega_1 & 0 \\ -\omega_2 & \omega_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} g$$

where ω_1 *stands for pitch rate,* ω_2 *for roll rate,* ω_3 *for yaw rate and* v_1 *the velocity of the aircraft. Then the 3-D velocity* (ω, v) *in the continuous epipolar constraint (5.28) has the form* $\omega = [\omega_1, \omega_2, \omega_3]^T, v = [v_1, 0, 0]^T$*. For the algorithm given above, we here have extra constraints on the symmetric matrix* $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$*:* $s_1 = s_5 = 0$ *and* $s_4 = s_6$*. Then there are only four different essential parameters left to determine and we can re-define the essential parameter vector* $\mathbf{e} \in \mathbb{R}^4$ *to be* $\mathbf{e} = [v_1, s_2, s_3, s_4]^T$*. Then the measurement vector* $\mathbf{a} \in \mathbb{R}^4$ *is to be* $\mathbf{a} = [u_3 y - u_2 z, 2xy, 2xz, y^2 + z^2]^T$*. The continuous epipolar constraint can then be rewritten as*

$$\mathbf{a}^T \mathbf{e} = 0.$$

If we define the matrix A from $\mathbf{a}$ *as in (5.41), the matrix* $A^T A$ *is a* 4×4 *matrix rather than a* 9×9 *one. For estimating the velocity* (ω, v)*, the dimensions of the problem is then reduced from 9 to 4. In this special case, the minimum number of optical flow measurements needed to guarantee a*

unique solution of $\mathbf{e}$ *is reduced to* 3 *instead of* 8. *Furthermore, the symmetric matrix* s *recovered from* $\mathbf{e}$ *is automatically in the space* $\mathcal{S}$ *and the remaining steps of the algorithm can thus be dramatically simplified. From this simplified algorithm, the angular velocity* $\omega = [\omega_1, \omega_2, \omega_3]^T$ *can be fully recovered from the images. The velocity information can then be used for controlling the aircraft.*

As in the discrete case, the linear algorithm proposed above is not optimal since it does not enforce the structure of the parameter space during the minimization. Therefore, the recovered velocity does not necessarily minimize the originally chosen error function $\|A\mathbf{e}(\omega, v)\|^2$ on the space $\mathcal{E}_1'$. Again like in the discrete case, we have to assume that translation is not zero. If the motion is purely rotational, then one can prove that there are infinitely many solutions to the epipolar constraint related equations. We leave this as an exercise to the reader.

5.4.4 Euclidean constraints and structure reconstruction

As in the discrete case, the purpose of exploiting Euclidean constraints is to reconstruct the scales of the motion and structure. From the above linear algorithm, we know we can only recover the linear velocity v up to an arbitrary scale. Without loss of generality, we may assume the velocity of the camera motion is $(\omega, \eta v)$ with $\|v\| = 1$ and $\eta \in \mathbb{R}$. By now, only the scale η is unknown to us. Substituting $\mathbf{X}(t) = \lambda(t)\mathbf{x}(t)$ into the equation

$$\dot{\mathbf{X}}(t) = \widehat{\omega}\mathbf{X}(t) + \eta v(t),$$

we obtain for the image $\mathbf{x}^j$ of each point $p^j \in \mathbb{E}^3, j = 1, \ldots, n$,

$$\dot{\lambda}^j\mathbf{x}^j + \lambda^j\dot{\mathbf{x}}^j = \hat{\omega}(\lambda^j\mathbf{x}^j) + \eta v \quad \Leftrightarrow \quad \dot{\lambda}^j\mathbf{x}^j + \lambda^j(\dot{\mathbf{x}}^j - \hat{\omega}\mathbf{x}^j) - \eta v = 0. \tag{5.47}$$

As one may expect, in the continuous case, the scale information is then encoded in $\lambda, \dot{\lambda}$ for the location of the 3-D point, and $\eta \in \mathbb{R}^+$ for the linear velocity v. Knowing $\mathbf{x}, \dot{\mathbf{x}}, \omega$ and v, these constraints are all linear in $\lambda^j, \dot{\lambda}^j, 1 \leq j \leq n$ and η. Also, if $\mathbf{x}^j, 1 \leq j \leq n$ are linearly independent of v, i.e. the feature points do not line up with the direction of translation, it can be shown that these linear constraints are not degenerate hence the unknown scales are determined up to a universal scale. We may then arrange all the unknown scales into a single vector $\vec{\lambda}$

$$\vec{\lambda} = [\lambda^1, \ldots, \lambda^n, \dot{\lambda}^1, \ldots, \dot{\lambda}^n, \eta]^T \in \mathbb{R}^{2n+1}.$$

For n optical flows, $\vec{\lambda}$ is a $2n+1$ dimensional vector. (5.47) gives $3n$ (scalar) linear equations. The problem of solving $\vec{\lambda}$ from (5.47) is usually overdetermined. It is easy to check that in the absence of noise the set of equations given by (5.47) uniquely determines $\vec{\lambda}$ if the configuration is non-critical. We can therefore write all the equations in a matrix form

$$M\vec{\lambda} = 0$$

with $M \in \mathbb{R}^{3n\times(2n+1)}$ a matrix depending on ω, v and $\{(\mathbf{x}^j, \dot{\mathbf{x}}^j)\}_{j=1}^n$. Then, in the presence of noise, the linear least-squares estimate of $\vec{\lambda}$ is simply the eigenvector of $M^T M$ corresponding to the smallest eigenvalue.

Notice that the rate of scales $\{\dot{\lambda}^j\}_{j=1}^n$ are also estimated. Suppose we have done the above recovery for a time interval, say (t_0, t_f), then we have the estimate $\vec{\lambda}(t)$ as a function of time t. But $\vec{\lambda}(t)$ at each time t is only determined up to an arbitrary scale. Hence $\rho(t)\vec{\lambda}(t)$ is also a valid estimation for any positive function $\rho(t)$ defined on (t_0, t_f). However, since $\rho(t)$ is multiplied to both $\lambda(t)$ and $\dot{\lambda}(t)$, their ratio

$$r(t) = \dot{\lambda}(t)/\lambda(t)$$

is independent of the choice of $\rho(t)$. Notice $\frac{d}{dt}(\ln\lambda) = \dot{\lambda}/\lambda$. Let the logarithm of the structural scale λ to be $y = \ln\lambda$. Then a time-consistent estimation $\lambda(t)$ needs to satisfy the following ordinary differential equation, which we call the *dynamic scale ODE*

$$\dot{y}(t) = r(t).$$

Given $y(t_0) = y_0 = \ln(\lambda(t_0))$, solve this ODE and obtain $y(t)$ for $t \in [t_0, t_f]$. Then we can recover a consistent scale $\lambda(t)$ given by

$$\lambda(t) = \exp(y(t)).$$

Hence (structure and motion) scales estimated at different time instances now are all relative to the same scale at time t_0. Therefore, in the continuous case, we are also able to recover all the scales as functions of time up to a universal scale. The reader must be aware that the above scheme is only *conceptual*. In practice, the ratio function $r(t)$ would never be available for all time instances in $[t_0, t_f]$.

Comment 5.1 (Universal scale ambiguity). *In both the discrete and continuous cases, in principle, the proposed schemes can reconstruct both the Euclidean structure and motion up to a universal scale.*

5.5 Summary

The seminal work of Longuet-Higgins [LH81] on the characterization of the so called *epipolar constraint*, has enabled the decoupling of the structure and motion problems and led to the development of numerous linear and nonlinear algorithms for motion estimation from two views, see [Fau93, Kan93, May93, WHA93] for overviews.

5.6 Exercises

1. **Linear equation**
   Solve $x \in \mathbb{R}^n$ from the following equation
   $$Ax = b$$
   where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. In terms of conditions on the matrix A and vector b, describe: When does a solution exist or not exist, and when is the solution unique or not unique? In case the solution is not unique, describe the whole solution set.

2. **Properties of skew symmetric matrix**
   (a) Prove Lemma 5.1.
   (b) Prove Lemma 5.3.

3. **A rank condition for epipolar constraint**
   Show that
   $$\mathbf{x}_2^T \widehat{T} R \mathbf{x}_1 = 0$$
   if and only if
   $$\operatorname{rank}[\widehat{\mathbf{x}_2} R \mathbf{x}_1 \ \widehat{\mathbf{x}_2} T] \leq 1.$$

4. **Rotational motion**
   Assume that camera undergoes pure rotational motion, i.e. it rotates around its center. Let $R \in SO(3)$ be the rotation of the camera and $\omega \in so(3)$ be the angular velocity. Show that, in this case, we have:
   (a) Discrete case: $\mathbf{x}_2^T \widehat{T} R \mathbf{x}_1 \equiv 0, \quad \forall T \in \mathbb{R}^3$;
   (b) Continuous case: $\mathbf{x}^T \widehat{\omega} \widehat{v} \mathbf{x} + \mathbf{u}^T \widehat{v} \mathbf{x} \equiv 0, \quad \forall v \in \mathbb{R}^3$.

5. **Projection to $O(3)$**
   Given an arbitrary 3×3 matrix $M \in \mathbb{R}^{3 \times 3}$ with positive singular values, find the orthogonal matrix $R \in O(3)$ such that the error $\|R - M\|_f^2$ is minimized. Is the solution unique? Note: Here we allow $\det(R) = \pm 1$.

6. **Geometric distance to epipolar line**
   Given two image points $\mathbf{x}_1, \tilde{\mathbf{x}}_2$ with respect to camera frames with their relative motion (R, T), show that the geometric distance d_2 defined in Figure 5.2 is given by the formula
   $$d_2^2 = \frac{(\tilde{\mathbf{x}}_2^T \widehat{T} R \mathbf{x}_1)^2}{\|\widehat{e}_3 \widehat{T} R \mathbf{x}_1\|^2}$$
   where $e_3 = [0, 0, 1]^T \in \mathbb{R}^3$.

7. **Planar scene and homography**
   Suppose we have a set of n fixed coplanar points $\{p^j\}_{j=1}^n \subset \mathbf{P}$, where

P denotes a plane. Figure 5.4 depicts the geometry of the camera frame relative to the plane. Without loss of generality, we further

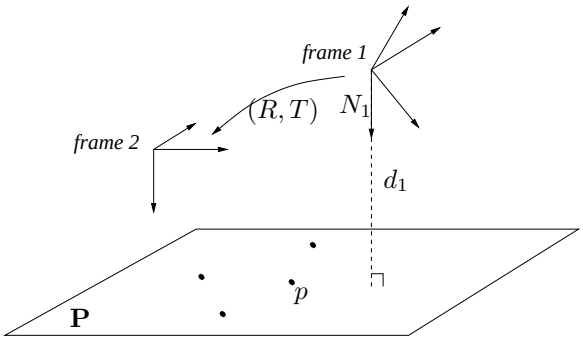

Figure 5.4. Geometry of camera frames 1 and 2 relative to a plane $\mathbf{P}$.

assume that the focal length of the camera is 1. That is, if $\mathbf{x}$ is the image of a point $p \in \mathbf{P}$ with 3-D coordinates $\mathbf{X} = [X_1, X_2, X_3]^T$, then

$$\mathbf{x} = \mathbf{X}/X_3 = [X_1/X_3, X_2/X_3, 1]^T \in \mathbb{R}^3.$$

Follow the following steps to establish the so-called *homography* between two images of the plane $\mathbf{P}$:

(a) Verify the following simple identity

$$\widehat{\mathbf{x}}\mathbf{X} = 0. \tag{5.48}$$

(b) Suppose that the $(R, T) \in SE(3)$ is the rigid body transformation from frame 1 to 2. Then the coordinates $\mathbf{X}_1, \mathbf{X}_2$ of a fixed point $p \in \mathbf{P}$ relative to the two camera frames are related by

$$\mathbf{X}_2 = \left(R + \frac{1}{d_1} T N_1^T\right) \mathbf{X}_1 \tag{5.49}$$

where d_1 is the perpendicular distance of camera frame 1 to the plane $\mathbf{P}$ and $N_1 \in \mathbb{S}^2$ is the unit surface normal of $\mathbf{P}$ relative to camera frame 1. In the equation, the matrix $H = (R + \frac{1}{d_1} T N_1^T)$ is the so-called *homography matrix* in the computer vision literature. It represents the transformation from $\mathbf{X}_1$ to $\mathbf{X}_2$.

(c) Use the above identities to show that: Given the two images $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ of a point $p \in \mathbf{P}$ with respect to camera frames 1 and 2 respectively, they satisfy the constraint

$$\widehat{\mathbf{x}_2}(R + \frac{1}{d_1} T N_1^T)\mathbf{x}_1 = 0. \tag{5.50}$$

(d) Prove that in order to solve uniquely (up to a scale) H from the above equation, one needs the (two) images of at least 4 points in $\mathbf{P}$ in general position.

8. **Singular values of the homography matrix**
Prove that any matrix of the form $H = R + uv^T$ with $R \in SO(3)$ and $u, v \in \mathbb{R}^3$ must have a singular value 1. (Hint: prove that the matrix $uv^T + vu^T + vu^T uv^T$ has a zero eigenvalue by constructing the corresponding eigenvector.)

9. **The symmetric component of the outer product of two vectors**
Suppose $u, v \in \mathbb{R}^3$, and $\|u\|^2 = \|v\|^2 = \alpha$. If $u \neq v$, the matrix $D = uv^T + vu^T \in \mathbb{R}^{3\times 3}$ has eigenvalues $\{\lambda_1, 0, \lambda_3\}$, where $\lambda_1 > 0$, and $\lambda_3 < 0$. If $u = \pm v$, the matrix D has eigenvalues $\{\pm 2\alpha, 0, 0\}$.

10. **The continuous homography matrix**
The continuous version of the homography matrix H introduced above is $H' = \widehat{\omega} + \frac{1}{d_1} v N_1^T$ where $\omega, v \in \mathbb{R}^3$ are the angular and linear velocities of the camera respectively. Suppose that you are given a matrix M which is also known to be of the form $M = H' + \lambda I$ for some $\lambda \in \mathbb{R}$. But you are not told the actual value of λ. Prove that you can uniquely recover H' from M and show how.

11. **Implementation of the SVD-based pose estimation algorithm**
Implement a version of the three step pose estimation algorithm for two views. Your MATLAB codes are responsible for:

- Initialization: Generate a set of N (≥ 8) 3-D points; generate a rigid body motion (R, T) between two camera frames and project (the coordinates of) the points (relative to the camera frame) onto the image plane correctly. Here you may assume the focal length is 1. This step will give you corresponding images as input to the algorithm.
- Motion Recovery: using the corresponding images and the algorithm to compute the motion $(\tilde{R}, \tilde{T})$ and compare it to the ground truth (R, T).

After you get the correct answer from the above steps, here are a few suggestions for you to play with the algorithm (or improve it):

- A more realistic way to generate these 3-D points is to make sure that they are all indeed "in front of" the image plane before and after the camera moves. If a camera has a field of view, how to make sure that all the feature points are shared in the view before and after.

- Systematically add some noises to the projected images and see how the algorithm responds. Try different camera motions and different layouts of the points in 3-D.
- Finally, to fail the algorithm, take all the 3-D points from some plane in front of the camera. Run the codes and see what do you get (especially with some noises on the images).

12. **Implementation of the eigenvalue-decomposition based velocity estimation algorithm**
Implement a version of the four step velocity estimation algorithm for optical flows. Your MATLAB codes are responsible for:
    - Initialization: Choose a set of N (≥ 8) 3-D points and a rigid body velocity (ω, v). Correctly obtain the image $\mathbf{x}$ and compute the image velocity $\mathbf{u} = \dot{\mathbf{x}}$ – you need to figure out how to compute $\mathbf{u}$ from (ω, v) and $\mathbf{X}$. Here you may assume the focal length is 1. This step will give you images and their velocities as input to the algorithm.
    - Motion Recovery: Use the algorithm to compute the motion $(\tilde{\omega}, \tilde{v})$ and compare it to the ground truth (ω, v).

Chapter 6
Camera calibration and self-calibration

Chapter 3 described the image-formation process and showed that camera geometry depends upon a number of parameters such as the position of the principal point, focal length, so called intrinsic parameters of the camera. Previous chapter demonstrated that in case the internal parameters of the camera are available, given two views of the scene and number of corresponding points in both views, the Euclidean structure of the scene as well as displacements between the views can be recovered. This chapter will again strive for the recovery of the Euclidean structure of the world and camera pose in the absence of knowledge of internal parameters of the camera.

The natural starting point is the development of techniques for estimating the internal parameters of the camera and reducing the problem to the one described in the previous chapter. Hence we first review the most common technique for camera calibration, by using a known object (a so-called "calibration rig"). Since this technique requires a specific calibration object it is suitable only for laboratory environments, where both the camera taking the images as well as the calibration object are available.

The second class of techniques for estimating camera parameters does not have this requirement, and involves estimating intrinsic parameters as well as scene structure and camera pose directly from image measurements; hence, it is called camera *self-calibration*. As we will see, given the richness and the difficulty of the problem, there are several avenues to pursue. We first focus on the so-called *intrinsic approach* to the problem of self-calibration which leads to analysis, algorithms and sufficient and necessary conditions for the recovery of the *all* the unknowns based on in-

trinsic constraints only. For the cases when the conditions are not satisfied we will characterize the associated ambiguities and suggest practical ways to resolve them.

If we could measure *metric* coordinates of points on the image plane (as opposed to *pixel* coordinates), we could model the projection as we have done in previous chapters:

$$\lambda \mathbf{x} = Pg\mathbf{X} \tag{6.1}$$

where $P = [I,\ 0]$ and $g \in SE(3)$ is the pose of the camera in the world reference frame. In practice, however, calibration parameters are not known ahead of time and, therefore, a more realistic model of the geometry of the image formation process takes the form

$$\lambda \mathbf{x}' = APg\mathbf{X} \tag{6.2}$$

where $A \in \mathbb{R}^{3\times 3}$ is the *camera calibration matrix* that collects the unknown parameters[1]

$$A = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{6.3}$$

In this chapter we show how to calibrate the camera, that is to recover both A (the *intrinsic* parameters) and g (the *extrinsic* parameters). Once the intrinsic parameters are recovered, we can convert equation (6.2) to a calibrated model by multiplying both sides by A^{-1}. In this chapter, we indicate pixel coordinates with a prime superscript $\mathbf{x}'$, whereas metric coordinates are indicated by $\mathbf{x}$.

We distinguish between two calibration methods: in classical calibration we are given images of a known object (a calibration rig); in *self-calibration* we attempt to recover intrinsic parameters together with camera pose and scene reconstruction without any calibration rig. The former is simpler and more robust, and several software package are available. However, often one has no control on how the images are captured and, therefore, auto-calibration is the only viable option. Most of the chapter, therefore, is concerned with self-calibration.

6.1 Calibration with a rig

Consider an object with distinct point-features whose coordinates relative to a fixed reference frame are known. We call this object a *calibration rig*. Typical choices for calibration rigs are checkerboard patterns mounted on one or more planes (Figure **??**) . Then the coordinates of each corner can

[1] In Section 6.3 we relax the restriction that A be upper triangular.

be easily specified relative to, say, the top-left corner of the checkerboard. Let us call these coordinates $\mathbf{X}$. The change of coordinates between the reference frame on the calibration rig and the camera frame is denoted by g and the calibration matrix by A, so that when we image the calibration rig we measure pixel coordinates $\mathbf{x}'$ of points on calibration grid, which satisfy

$$\lambda \mathbf{x}' = APg\mathbf{X} = \tilde{P}\mathbf{X}.$$

This can be written for each point on the calibration rig as:

$$\lambda \begin{bmatrix} x^i \\ y^i \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix} \begin{bmatrix} X^i \\ Y^i \\ Z^i \\ 1 \end{bmatrix} \tag{6.4}$$

For clarity we omit the $'$s from the pixel coordinates of image points in the above equation. Similarly as in the previous chapters, we can eliminate λ from the above equation, by multiplying both sides by $\widehat{\mathbf{x}}'$. This yields three equations, from which only two are independent. Hence for each point we have following two constraints:

$$\begin{aligned} x^i(\mathbf{p}_3^T\mathbf{X}) &= \mathbf{p}_1^T\mathbf{X} \\ y^i(\mathbf{p}_3^T\mathbf{X}) &= \mathbf{p}_2^T\mathbf{X} \end{aligned} \tag{6.5}$$

Unlike previous chapters, now X^i, Y^i and Z^i are known, and so are x^i, y^i. We can therefore stack all the unknowns p_{ij} into a vector and re-write the equations above as a system of linear equations

$$M\mathbf{p} = 0$$

where

$$\mathbf{p} = [p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{34}]^T \in \mathbb{R}^{12}.$$

A linear (sub-optimal) estimate of $\mathbf{p}$ can then be obtained by minimizing the linear least-squares criterion

$$\min \|M\mathbf{p}\|^2 \quad \text{subject to} \quad \|\mathbf{p}\|^2 = 1 \tag{6.6}$$

without taking into account the structure of the unknown vector $\mathbf{p}$. If we denote the (nonlinear) map from $\mathbf{X}$ to $\mathbf{x}'$ as:

$$\pi(\mathbf{X}, \tilde{P}) \doteq \mathbf{x}',$$

the estimate can be further refined by a nonlinear minimization of the following objective function

$$\min_{\tilde{P}} \sum_i \|\mathbf{x}'^i - \pi(\mathbf{X}^i, \tilde{P})\|^2.$$

After obtaining an estimate of the projection matrix $\tilde{P} = \tilde{P} = A[R \mid T] = [AR|AT]$ we can factor the first 3×3 sub-matrix into the calibration matrix

$A \in \mathbb{R}^{3\times 3}$ (in its upper triangular form) and rotation matrix $R \in SO(3)$ using a routine QR decomposition

$$AR = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix},$$

which yields an estimate of the intrinsic parameters in the calibration matrix A and of the rotational component of the extrinsic parameter. Estimating translation completes the calibration procedure:

$$T = A^{-1} \begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix}.$$

Several free software packages are available to perform calibration with a rig. Most also include compensation for radial distortion and other lens artifacts that we do not address here. In Chapter **??** we walk the reader through use of one of these packages in a step-by-step procedure to calibrate a camera.

6.2 The fundamental matrix

In the absence of a calibration grid, one view is clearly not sufficient to determine all the unknowns. Hence, we revisit here the intrinsic geometric relationship between two views, the concept of epipolar geometry introduced in the previous. In order to obtain the relationship between the two uncalibrated views related by the displacement (R, T), consider the first the rigid body motion between two views:

$$\lambda_2 \mathbf{x}_2 = R\lambda \mathbf{x}_1 + T$$

where $\mathbf{x} = \lambda \mathbf{X}$. Multiplying both sides by calibration matrix A:

$$\lambda_2 A\mathbf{x}_2 = AR\lambda \mathbf{x}_1 + AT \quad \Leftrightarrow \quad \lambda_2 \mathbf{x}'_2 = ARA^{-1}\lambda_1 \mathbf{x}'_1 + T' \tag{6.7}$$

where $\mathbf{x}'$ denotes a pixel coordinate of a point $\mathbf{x}$, such $\mathbf{x}' = A\mathbf{x}$ and $T' = AT$. In order to obtain an intrinsic constraint in terms of image measurements only, similarly as in the calibrated case, we can eliminate the unknown depth by multiplying both sides of (6.7) by $(\mathbf{x}'_2 \times T')$ yielding the following constraint:

$${\mathbf{x}'_2}^T \widehat{T}' ARA^{-1} \mathbf{x}'_1 = 0. \tag{6.8}$$

Alternatively, the transformation from the calibrated space to the uncalibrated given by $\mathbf{x} = A^{-1}\mathbf{x}'$, can be directly applied to epipolar constraint by substituting for $\mathbf{x}$:

$$\mathbf{x}_2^T \widehat{T} R \mathbf{x}_1 = 0 \quad \Leftrightarrow \quad {\mathbf{x}'_2}^T A^{-T} \widehat{T} R A^{-1} \mathbf{x}'_1 = 0 \tag{6.9}$$

, consider the first just the rigid body motion equation and multiply both sides by A

$$\lambda_2 A\mathbf{x}_2 = AR\lambda\mathbf{x}_1 + AT \quad \Leftrightarrow \quad \lambda_2\mathbf{x}_2' = ARA^{-1}\lambda_1\mathbf{x}_1' + T' \tag{6.10}$$

where $\mathbf{x} = \lambda\mathbf{X}, \mathbf{x}' = A\mathbf{x}$ and $T' = AT$. In order to obtain an intrinsic constraint in terms of image measurements only, similarly as in the calibrated case, we can eliminate the unknown depth by multiplying both sides of (6.7) by $(\mathbf{x}_2' \times T')$ yielding the following constraint:

$$\mathbf{x}_2'^T \widehat{T}' ARA^{-1}\mathbf{x}_1' = 0. \tag{6.11}$$

Alternatively the transformation from the calibrated space to the uncalibrated given by $\mathbf{x} = A^{-1}\mathbf{x}'$, can be directly applied to epipolar constraints by substituting for $\mathbf{x}$:

$$\mathbf{x}_2^T \widehat{T} R\mathbf{x}_1 = 0 \quad \Leftrightarrow \quad \mathbf{x}_2'^T A^{-T}\widehat{T}RA^{-1}\mathbf{x}_1' \tag{6.12}$$

The above cases both correspond to the uncalibrated version of the epipolar constraint:

$$\boxed{\mathbf{x}_2'^T F\mathbf{x}_1' = 0} \tag{6.13}$$

The two corresponding image points are related by the matrix $F = A^{-T}\widehat{T}RA^{-1} = \widehat{T}'ARA^{-1} \in \mathbb{R}^{3\times 3}$ is called *fundamental matrix*. When $A = I$, the fundamental matrix is an essential matrix $\widehat{T}R$. In order to reconcile two different forms of the fundamental matrix note that for $T \in \mathbb{R}^3$ and $A \in SL(3)$[2], we have $A^T\widehat{T}A = \widehat{A^{-1}T}$, which enables us to show that two different forms of fundamental matrix are equivalent:

$$\boxed{F = A^{-T}\widehat{T}RA^{-1} = A^{-T}\widehat{T}A^{-1}ARA^{-1} = \widehat{T'}ARA^{-1}} \tag{6.14}$$

We will study these different forms more closely in Section 6.3 and show that the second form turns out to be more convenient to use for camera self-calibration. Before getting more insights into the theory of self-calibration, let us have a look at some geometric intuition behind the fundamental matrix, its algebraic properties and algorithm for its recovery.

6.2.1 Geometric characterization of the fundamental matrix

Geometrically the fundamental matrix maps points in the first view into the lines in in the second view (or vice versa):

$$\mathbf{x}_2'^T F\mathbf{x}_1' = \mathbf{x}_2'^T \mathbf{l}_2 = 0$$

Notice that $\mathbf{l}_2$ defines a line implicitly as the collection of points that solve $\mathbf{l}_2^T\mathbf{x} = 0$, where the vector $\mathbf{l}_2$ is the normal to the line. From the first

[2]We will justify this generalization in the next section.

form of fundamental matrix, the part related to translation $T' = AT \in \mathbb{R}^3$ is called the *epipole*. The epipole is the point where the *baseline* (the line joining the centers of projection of the two cameras) intersect the image plane in the second (first) view and can be computed as the left (right) null space of the fundamental matrix $F(F^T)$. It can be easily verified that:

$$F^T T_2' = F T_1' = 0$$

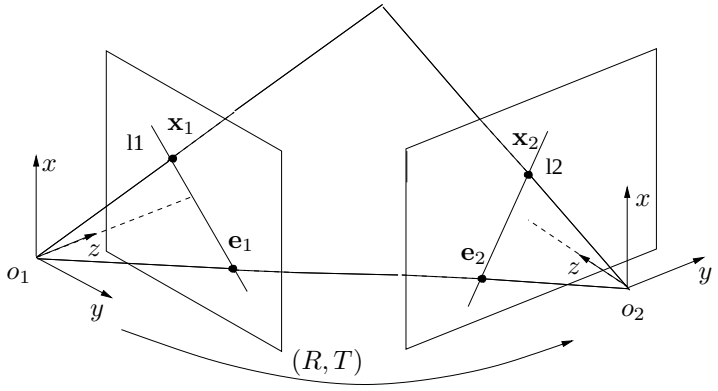

Figure 6.1. Two projections $\mathbf{x}_1', \mathbf{x}_2' \in \mathbb{R}^3$ of a 3-D point p from two vantage points. The relative Euclidean transformation between the two vantage points is given by $(R, T) \in SE(3)$.

Remark 6.1. *Characterization of the fundamental matrix. A non-zero matrix $F \in \mathbb{R}^{3\times 3}$ is a fundamental matrix if F has a singular value decomposition (SVD): $E = U\Sigma V^T$ with*

$$\Sigma = diag\{\sigma_1, \sigma_2, 0\}$$

for some $\sigma_1, \sigma_2 \in \mathbb{R}_+$.

The characterization of the fundamental matrix in terms of its SVD reflects the fact that the F is rank deficient $\det(F) = 0$. This can be simply observed by noticing that F is a product of a skew symmetric matrix $\widehat{T}'$ or rank 2 and a matrix $ARA^{-1} \in \mathbb{R}^{3\times 3}$ of rank 3. Similarly to the essential matrix we can see how can we factorize F into a skew symmetric matrix $\widehat{\mathbf{m}}$ and nonsingular matrix M:

$$(\widehat{\mathbf{m}}, M) \quad = \quad (UR_Z(\pm\frac{\pi}{2})diag\{1, 1, 0\}U^T, UR_Z^T(\pm\frac{\pi}{2})\Sigma V^T) \quad (6.15)$$

Note that given the above factorization and replacing Σ by matrix $\tilde{\Sigma}$:

$$\tilde{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ \alpha & \beta & \gamma \end{bmatrix} \quad (6.16)$$

then we have:

$$(\widehat{\mathbf{m}}, M) \quad = \quad (UR_Z(\pm\frac{\pi}{2})diag\{1,1,0\}U^T, UR_Z^T(\pm\frac{\pi}{2})\tilde{\Sigma}V^T) \quad (6.17)$$

give the rise to the same fundamental matrix, for arbitrary choice of parameters α, β, γ.

So, as we can see in the uncalibrated case, while the fundamental matrix is uniquely determined from point correspondences, the factorization of F into a skew-symmetric part and a non-singular matrix is not unique. There is a three-parameter family of transformations consistent with the point correspondences, which gives rise to the same fundamental matrix. This fact can be observed directly from the epipolar constraint by noticing that:

$$\mathbf{x}_2'^T \widehat{T}' ARA^{-1}\mathbf{x}_1' \;=\; \mathbf{x}_2'^T \widehat{T}'(ARA^{-1} + T'\mathbf{v}^T)\mathbf{x}_1' \;=\; 0. \qquad (6.18)$$

for an arbitrary vector $\mathbf{v}$, since $\widehat{T}'(T'\mathbf{v}^T) = 0$.

Given the above ambiguities present in the uncalibrated case, additional constraints have to be sought in order to be able to recover the Euclidean structure, pose of the cameras as well as information about intrinsic camera parameters. In the remainder of this chapter we will present two strategies for resolving these ambiguities: an *intrinsic approach* - where we exploit additional intrinsic constraints between image measurements in the uncalibrated case, and a *stratified approach* where we exploit knowledge of motion, partial structure and/or calibration. The two approaches will be reconciled in the last section of this chapter. Before we proceed in getting more insight and exploring different types of additional constraints, we outline an algorithm for the recovery of the fundamental matrix.

6.2.2 The eight-point linear algorithm

Similarly to the calibrated case, the fundamental matrix can be recovered from the measurements using linear techniques only, hence we will obtain a version of the 8-point algorithm for the uncalibrated case. Since F is uniquely determined by point correspondences between two views and geometrically represents a mapping of a point in one view into an epipolar line in another, it plays an important role in the process of establishing correspondences between two views. Given the fundamental matrix $F \in \mathbb{R}^{3\times 3}$ with the entries as:

$$F = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \qquad (6.19)$$

and a corresponding vector $\mathbf{f} \in \mathbb{R}^9$, we have

$$\mathbf{f} = [f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9]^T \in \mathbb{R}^9.$$

Since the epipolar constraint $\mathbf{x}_2^T F \mathbf{x}_1 = 0$ is linear in the entries of $\mathbf{f}$, we can rewrite it as:

$$\mathbf{a}^T \mathbf{f} = 0 \tag{6.20}$$

Where $\mathbf{a}$ is given by

$$\mathbf{a} = [x'_2 x'_1, x'_2 y'_1, x'_2 z'_1, y'_2 x'_1, y'_2 y'_1, y'_2 z'_1, z'_2 x'_1, z'_2 y'_1, z'_2 z'_1]^T \in \mathbb{R}^9. \tag{6.21}$$

Given a set of corresponding points and forming the matrix of measurements $A = [\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^n]^T$, $\mathbf{f}$ can be obtained as the minimizing solution of the following least-squares objective function $\|Af\|^2$. Such solution corresponds to the eigenvector associated with the smallest eigenvalue of $A^T A$. In the context of the above mentioned linear algorithm, since the point correspondences are in image coordinates, the individual entries of the matrix A are may be unbalanced and affect the conditioning of $A^T A$. An improvement can be achieved by a normalizing transform T_1 and T_2 in respective images, which makes the measurements of zero mean and unit-variance [**?**]. By the same arguments as in the calibrated case, the linear least-squares solution is not an optimal one, and it is typically followed by a nonlinear refinement step. More details on the nonlinear refinement strategies are described [HZ00].

6.3 Basics of uncalibrated geometry

Before we delve into algorithms to recover camera calibration, we point out an important observation that makes the geometry of uncalibrated cameras simpler to understand.

So far we have considered points living in the three-dimensional Euclidean space, and shown how to reconstruct spatial properties such as distances and angles. Distances and angles can be written in terms of the inner product[3] $\langle u, v \rangle \doteq u^T v$. If one were to define a different inner product, for instance $\langle u, v \rangle \doteq u^T \Omega v$ for some positive definite symmetric matrix Ω, then the notion of distances and angles would change, and one could think of a "distorted" or "uncalibrated" world. Nevertheless, all reconstruction algorithms described in the previous chapters, written in terms of this new inner product, would yield a reconstruction of the correct camera pose and scene structure in the distorted space. Only in the particular case where $\Omega = I$ would the reconstruction be physically correct.

The basic observation here is that an *uncalibrated camera* with calibration matrix A *imaging points in a calibrated world* is equivalent to a *calibrated camera imaging points in an uncalibrated world* governed by an inner product $\langle u, v \rangle_\Omega \doteq u^T \Omega v$. Furthermore, $\Omega = A^{-T} A^{-1}$.

[3] Definitions and properties of inner products can be found in Appendix A.

To see this, let $\mathbb{E}^3$ denote the three-dimensional Euclidean space, that is $\mathbb{R}^3$ with its standard inner product $\langle u, v\rangle \doteq u^T v$. Let instead $\mathbb{E}^{3'}$ denote the "uncalibrated" space, that is $\mathbb{R}^3$ with the inner product $\langle u, v\rangle_\Omega \doteq u^T \Omega v$. The map from the calibrated to the uncalibrated space is simply

$$\begin{aligned} \psi : \mathbb{R}^3 &\rightarrow \mathbb{R}^3 \\ \mathbf{X} &\mapsto \mathbf{X}' = A\mathbf{X} \end{aligned}$$

where $\mathbf{X} \in \mathbb{R}^3$ and $\mathbf{X}' \in \mathbb{R}^3$ represent the three-dimensional coordinates of the points $p \in \mathbb{E}^3$ and $p' = \psi(p) \in \mathbb{E}^{3'}$ respectively, and $A \in \mathbb{R}^{3\times 3}$ is the matrix representing the linear map. The map ψ induces a transformation of the inner product as follows

$$\langle \psi^{-1}(u), \psi^{-1}(v)\rangle = u^T A^{-T} A^{-1} v \doteq \langle u, v\rangle_{A^{-T}A^{-1}}, \quad \forall u, v \in T\mathbb{E}^3 \tag{6.22}$$

where $T\mathbb{E}^3$ is the space of vectors. Therefore, $\Omega = A^{-T}A^{-1}$. It is then clear that calibrating a camera, i.e. finding A, is equivalent to calibrating the space, i.e. finding its inner product Ω.[4] To see that, call $SL(3)$ the (group) of non-singular 3×3 matrices, and $\mathbb{K} \subset SL(3)$ the subset consisting of all upper-triangular matrices. That is, any matrix $A \in \mathbb{K}$ has the form

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix}. \tag{6.23}$$

Note that if A is upper-triangular, so is A^{-1}. Clearly, there is a one-to-one correspondence between $\mathbb{K}$ and the set of all upper-triangular matrices of the form given in (6.23); also, the equation $\Omega = A^{-T}A^{-1}$ gives a finite-to-one correspondence between $\mathbb{K}$ and the set of all 3×3 symmetric matrices with determinant 1 (by the *Cholesky factorization*) (see Appendix **??**). Usually, only one of the upper-triangular matrices corresponding to the same symmetric matrix has a physical interpretation as the intrinsic parameters of a camera. Thus, if the calibration matrix A does have the form given by (6.23), the self-calibration problem is equivalent to the problem of recovering the matrix Ω, i.e. the inner product of the uncalibrated space. Now let us consider a more general case where the uncalibrated camera is characterized by an arbitrary matrix $A \in SL(3)$. A has a QR-decomposition

$$A = QR, \quad Q \in \mathbb{K}, R \in SO(3). \tag{6.24}$$

Then $A^{-1} = R^T Q^{-1}$ and the associated symmetric matrix $\Omega = A^{-T}A^{-1} = Q^{-T}Q^{-1}$. In general, if $A = BR$ with $A, B \in SL(3), R \in SO(3)$, the $A^{-T}A^{-1} = B^{-T}B^{-1}$. That is A and B induce the same inner product on the uncalibrated space. In this case, we say that matrices A and B are

[4] The symmetric matrix Ω can be also interpreted as a conic in complex projective space, known as the absolute conic. Since our discussion does not involve any projective space, we do not further pursue that interpretation here.

equivalent. The quotient space $SL(3)/SO(3)$ can be called the *intrinsic parameter space* and will be further discussed in this section.

We now show that, without knowing camera motion and scene structure, the matrix $A \in SL(3)$ can only be recovered up to an equivalence class $\bar{A} \in SL(3)/SO(3)$. To see this, suppose that $B \in SL(3)$ is another matrix in the same equivalence class as A. Then $A = BR_0$ for some $R_0 \in SO(3)$. The coordinate transformation from time t_0 to t yields

$$\begin{aligned} & A\mathbf{X}(t) = AR(t)\mathbf{X}(t_0) + AT(t) \\ \Leftrightarrow \quad & BR_0\mathbf{X}(t) = BR_0R(t)R_0^T R_0\mathbf{X}(t_0) + BR_0T(t). \end{aligned} \tag{6.25}$$

From equation (6.25) it is clear that an uncalibrated camera with calibration matrix A undergoing the motion $(R(t), T(t))$ observing the point $p \in \mathbb{E}^3$ can never be distinguished from an uncalibrated camera with calibration matrix B undergoing the motion $(R_0R(t)R_0^T, R_0T(t))$ observing the point $R_0(p) \in \mathbb{E}^3$. The effect of R_0 is nothing but a *rotation* of the overall configuration space.

Therefore, without knowing camera motion and scene structure, the matrix A associated with an uncalibrated camera can only be recovered up to an equivalence class $\bar{A}$ in the space $SL(3)/SO(3)$. The subgroup $\mathbb{K}$ of all upper-triangular matrices in $SL(3)$ is one representation of such a space, as is the space of 3×3 symmetric positive definite matrices with determinant 1. Thus, $SL(3)/SO(3)$ does provide an intrinsic geometric interpretation for the unknown camera parameters. In general, the problem of camera self-calibration is then equivalent to the problem of recovering the symmetric matrix $\Omega = A^{-T}A^{-1}$, from which the upper-triangular representation of the intrinsic parameters can be easily obtained from the *Cholesky factorization*. The coordinate transformation in the uncalibrated space is given by

$$\begin{aligned} & A\mathbf{X}(t) = AR(t)\mathbf{X}(t_0) + AT(t) \\ \Leftrightarrow \quad & \mathbf{X}'(t) = AR(t)A^{-1}\mathbf{X}'(t_0) + T'(t) \end{aligned} \tag{6.26}$$

where $\mathbf{X}' = A\mathbf{X}$ and $T' = AT$. In homogeneous coordinates, the transformation group on the uncalibrated space is given by

$$G' = \left\{ \left[\begin{array}{cc} ARA^{-1} & T' \\ 0 & 1 \end{array} \right] \,\middle|\, T' \in \mathbb{R}^3, R \in SO(3) \right\} \subset \mathbb{R}^{4\times 4} \tag{6.27}$$

If the motion of a calibrated camera in the uncalibrated space is given by $g'(t) \in G', t \in \mathbb{R}$, the homogeneous coordinates of a point $p' \in \mathbb{E}^3$ satisfy

$$\left[\begin{array}{c} \mathbf{X}'(t) \\ 1 \end{array} \right] = \left[\begin{array}{cc} AR(t)A^{-1} & T'(t) \\ 0 & 1 \end{array} \right] \left[\begin{array}{c} \mathbf{X}'(t_0) \\ 1 \end{array} \right]. \tag{6.28}$$

From the ideal camera model, the image of the point p' with respect to such a camera (i.e. with an identity calibration matrix) is given by

$$\lambda(t)\mathbf{x}'(t) = P\mathbf{X}'(t) \tag{6.29}$$

where $\mathbf{X}'(t)$ is in homogeneous representation (so for the equation below). It is direct to check that the image $\mathbf{x}(t)$ is the same as the image of $p = \psi^{-1}(p') \in \mathbb{E}^3$ with respect to the uncalibrated camera, i.e. we have

$$\boxed{\lambda(t)\mathbf{x}'(t) = AP\mathbf{X}(t) = P\mathbf{X}'(t)} \tag{6.30}$$

Hence we have established a duality which can be roughly stated as "an uncalibrated camera in a calibrated Euclidean space is equivalent to a calibrated camera in an uncalibrated space". It seems that all we have done so far was converting the camera calibration problem to another one which is no easier. Nevertheless, as we will soon see, understanding of this duality in fact will give us some very useful insight when we study camera self-calibration. Note now that given the above introduction we can revisit the two different forms of the the fundamental matrix:

$$\boxed{F = A^{-T}\widehat{T}RA^{-1} = A^{-T}\widehat{T}A^{-1}ARA^{-1} = \widehat{T'}ARA^{-1}} \tag{6.31}$$

The first commonly encountered form $F = A^{-T}\widehat{T}RA^{-1}$ is the fundamental matrix of an uncalibrated camera in a calibrated space and the second form $F = \widehat{T'}ARA^{-1}$ is in fact the essential matrix of a calibrated camera in a uncalibrated space.

6.3.1 Kruppa's equations

In this section we pursue the analogy of an uncalibrated camera in a calibrated space with a calibrated camera in an uncalibrated space further, in order to derive intrinsic constraints on the calibration of the camera that can be inferred from a collection of fundamental matrices. In order to do so, we define two "copies" of the three-dimensional Euclidean space. The "calibrated" Euclidean space $\mathbb{E}_3$, with the usual inner product $\langle u, v\rangle \doteq u^T v$, and the "uncalibrated" Euclidean space $\mathbb{E}^3$ with the "uncalibrated" inner product $\langle u, v\rangle_S$ that ultimately will be related to the calibration of the camera. We remind the reader that the Euclidean space $\mathbb{E}_3$ is represented in coordinates by $\mathbb{R}^3$, endowed with the group of transformation $g = (T, R) \in SE(3)$, that acts on points of $\mathbb{E}_3$ via

$$g : \mathbb{R}^3 \to \mathbb{R}^3 \; ; \quad p \mapsto Rp + T. \tag{6.32}$$

Such a group induces an action on *vectors* v, which in this context are represented by the difference of two points $v = p_2 - p_1$, by

$$g_* : T\mathbb{R}^3 \sim \mathbb{R}^3 \to T\mathbb{R}^3 \sim \mathbb{R}^3 \; ; \quad v \mapsto Rv. \tag{6.33}$$

We make a point in distinguishing the action of (column) vectors from the action of *covectors*, or row vectors:

$$g^* : T\mathbb{R}^{3^*} \sim \mathbb{R}^3 \to T\mathbb{R}^{3^*} \sim \mathbb{R}^3 \; ; \quad v^T \mapsto v^T R \tag{6.34}$$

since this will be useful later. We equip $\mathbb{E}_3$ with the usual inner product

$$\langle , \rangle_S : T\mathbb{R}^3 \times T\mathbb{R}^3 \mapsto \mathbb{R} \; ; \quad (u, v) \mapsto \langle u, v \rangle \doteq u^T v \tag{6.35}$$

and we verify that such an inner product is invariant under the action of the group, as $\langle Ru, Rv \rangle = u^T R^T R v = u^T v = \langle u, v \rangle$. The inner product operates on covectors exactly in the same way: $\langle u^T, v^T \rangle \doteq u^T v$, so that the group action on covectors is also invariant: $\langle u^T R, v^T R \rangle = u^T R R^T v = u^T v = \langle u^T, v^T \rangle$. If we are given three (column) vectors, u, v, w, it is easy to verify that their volume, i.e. the determinant of the matrix having them as columns, is also invariant under the group action:

$$\det[Ru, Rv, Rw] = \det(R) \det[u, v, w] = \det[u, v, w]. \tag{6.36}$$

Now consider a transformation of the Euclidean space $\mathbb{E}_3$ via a (nonsingular) linear map into another "copy" of the Euclidean space, which we call $\mathbb{E}^3$ (it will soon be clear why we want to distinguish the two copies)

$$A : \mathbb{E}_3 \to \mathbb{E}^3 \; ; \quad p \mapsto Ap = q. \tag{6.37}$$

Such a map induces a group transformation on $\mathbb{E}^3$ by

$$h : \mathbb{R}^3 \to \mathbb{R}^3 \; ; \quad q \mapsto ARA^{-1}q + AT \tag{6.38}$$

which we represent as $h = (U, ARA^{-1})$ where

$$U = AT \in \mathbb{R}^3. \tag{6.39}$$

Such a group acts on vectors via

$$h_* : T\mathbb{R}^3 \to T\mathbb{R}^3 \; ; \quad u \mapsto ARA^{-1}u \tag{6.40}$$

and on covectors via

$$h^* : T\mathbb{R}^3 \to T\mathbb{R}^3 \; ; \quad u^T \mapsto u^T ARA^{-1}. \tag{6.41}$$

The bilinear, positive definite form that is invariant under the action of h, and plays the role of the inner product, is given by

$$\langle , \rangle_S : T\mathbb{R}^3 \times T\mathbb{R}^3 \to \mathbb{R} \; ; \quad (u, v) \mapsto \langle u, v \rangle_S \doteq u^T A^{-T} A^{-1} v \tag{6.42}$$

so that we have $\langle ARA^{-1}u, ARA^{-1}v \rangle_S = u^T A^{-T} R^T A^T A^{-T} A^{-1} ARA^{-1} v$ $= u^T A^{-T} A^{-1} v = \langle u, v \rangle_S$. The above bilinear form is determined uniquely by its values on the elements of the basis, i.e. the elements of the matrix $S \doteq A^{-T} A^{-1}$, so that we have

$$\langle u, v \rangle_S \doteq u^T S v \tag{6.43}$$

The bilinear form above acts on covectors via

$$\langle , \rangle_S : T\mathbb{R}^{3^*} \times T\mathbb{R}^{3^*} \to \mathbb{R} \; ; \quad (u^T, v^T) \mapsto \langle u^T, v^T \rangle_S \doteq u^T A A^T v \tag{6.44}$$

which is invariant with respect to the group action on covectors, since

$$\langle u^T ARA^{-1}, v^T ARA^{-1} \rangle_S = u^T A A^T v = \langle u^T, v^T \rangle_S \tag{6.45}$$

$\langle u^T, v^T \rangle_S \doteq u^T S^{-1} v$ which is different from $\langle u, v \rangle_S$; this shows that we cannot quite treat row and column vectors alike. The difference in the value of the inner product on vectors and covectors of $\mathbb{E}_3$ was not visible on $\langle , \rangle$ since in that case $S = I$. If we assume that the transformation A has determinant 1 (we can do this simply by scaling A, since we have assumed it is nonsingular, so $\det(A) \neq 0$) then, given three vectors u, v, w, their volume is invariant under the group action, as

$$\det[ARA^{-1}u, ARA^{-1}v, ARA^{-1}w] = \det(ARA^{-1}) \det[u, v, w] = \det[u, v, w]. \tag{6.46}$$

and on covectors via

$$h^* : T\mathbb{R}^3 \to T\mathbb{R}^3 \; ; \quad u^T \mapsto u^T ARA^{-1}. \tag{6.47}$$

With this machinery in our toolbox, we turn to the problem of camera self-calibration.

Uncalibrated camera, or uncalibrated space?

In our discussion of camera calibration, we have modeled a moving point, viewed with an uncalibrated camera, as

$$\begin{cases} p(t) = Rp_0 + T \\ \mathbf{x}(t) = \lambda^{-1} Ap(t) \end{cases} \tag{6.48}$$

where $p \in \mathbb{E}_3$ and A is a non-singular, upper-triangular matrix that is defined up to a scale. When $A = I$, we say that the camera is calibrated. Let us now apply the transformation A to the space $\mathbb{E}_3$ so that if we let $q = Ap$, we obtain

$$\begin{cases} q(t) = ARA^{-1}q_0 + U \\ \mathbf{x}(t) = \lambda^{-1} q(t). \end{cases} \tag{6.49}$$

Therefore, *an uncalibrated camera moving in the Euclidean space $\mathbb{E}_3$ with a rigid motion $g = (T, R)$ is equivalent to a calibrated camera moving in the transformed space $\mathbb{E}^3$ with a motion $h = (U, ARA^{-1})$*. In a calibrated system, represented by

$$\begin{cases} p(t) = Rp_0 + T \\ \mathbf{x}(t) = \lambda^{-1} p(t) \end{cases} \tag{6.50}$$

the Epipolar constraint expresses the fact that p, p_0 and T are coplanar, and therefore the volume they determine is zero: $\det[p(t), Rp_0, T] = 0$. We write this condition in terms of the inner product as

$$\langle p(t), Qp_0 \rangle = 0 \tag{6.51}$$

where we define the Essential matrix $Q \doteq \widehat{T}R \in TSO(3)$. The same volume constraint, in the uncalibrated space, is given by

$$\langle q(t), Fq_0 \rangle_S = 0 \tag{6.52}$$

where the matrix F, called the *Fundamental matrix*, is given by

$$F \doteq \widehat{U}ARA^{-1} \tag{6.53}$$

The importance of the above constraints is that, since they are homogeneous, we can substitute readily $\mathbf{x} = \lambda^{-1}p$ for p in the calibrated space, and $\mathbf{x} = \lambda^{-1}q$ for q in the uncalibrated one. Since $\mathbf{x}$ can be measured from images, the above equations give constraints either on the Essential matrix Q, in the calibrated case, or on the Fundamental matrix F, in the uncalibrated case. A more common expression of the fundamental matrix is given by $A^{-T}QA^{-1}$, which can be derived using the fact that

$$\widehat{AT} = A^{-T}\widehat{T}A^{-1} \tag{6.54}$$

so that

$$F = \widehat{AT}ARA^{-1} = A^{-T}\widehat{T}RA^{-1} = A^{-T}QA^{-1}. \tag{6.55}$$

Note that the vector $U = AT$, also called the *epipole*, is the (right) nullspace of the F matrix, which can be easily (i.e. linearly) computed from image measurements. As in the previous section, let A be a generic, non-singular matrix $A \in SL(3)$ and write its Q-R decomposition as $A = BR_0$. We want to make sure that the fundamental matrix does not depend on R_0: $F = (BR_0)^{-T}\widehat{T}R(BR_0)^{-1} = B^{-T}R_0\widehat{T}R_0^TR_0RR_0^TB^{-1}$. Therefore, we have that $F = A^{-T}\widehat{T}RA^{-1} = B^{-T}\widehat{\tilde{T}}\tilde{R}B^{-1}$, where $\tilde{T} = R_0T$ and $\tilde{R} = R_0RR_0^T$. In other words, as we expected, we cannot distinguish a point moving with (T, R) with calibration matrix A from one moving with $(\tilde{T}, \tilde{R})$ with calibration matrix B.

Enforcing the properties of the inner product: Kruppa equations

Since the Fundamental matrix can be recovered only up to a scale, let us assume, for the moment, that $\|U\| = \|AT\| = 1$. The following derivation is taken from [**?**]. Let $R_0 \in SO(3)$ be a rotation matrix such that

$$U = R_0e_3 \tag{6.56}$$

where $e_3 = [0\ 0\ 1]^T$. Note that at least one such matrix always exists (in geometric terms, one would say that this is a consequence of $SO(3)$ acting transitively on the sphere $\mathcal{S}^2$). Consider now the matrix $D = R_0FR_0^T$. A convenient expression for it is given by

$$D \doteq R_0FR_0^T = \widehat{e}_3(R_0A)^TR(R_0A)^{-T} = \begin{bmatrix} d_1^T \\ d_2^T \\ 0 \end{bmatrix}. \tag{6.57}$$

This expression comes from

$$\begin{aligned} D &= R_0 A^{-T} \widehat{T} R A^{-1} R_0^T \\ &= R_0 A^{-T} \widehat{A^{-1} R_0} e_3 R A^{-1} R_0^T = (R_0 A)^{-T} \widehat{(R_0 A)^{-1}} e_3 R (R_0 A)^{-1}. \end{aligned}$$

Since e_3 is in the (left) null space of D, we can arrange the rows so that the last one is zero. The remaining two columns are given by

$$\begin{cases} d_1^T = -e_2^T R_0 A R (R_0 A)^{-1} \\ d_2^T = e_1^T R_0 A R (R_0 A)^{-1}. \end{cases} \tag{6.58}$$

Now, the crucial observation comes from the fact that the covectors d_1^T and d_2^T are obtained through a transformation of the covectors e_1^T and e_2^T through the action of the group $\tilde{h} = (\tilde{U}, \tilde{A} R \tilde{A}^{-1})$, as indicated in equation (6.47), with $\tilde{A} = R_0 A$. Therefore, the inner product $\langle u^T, v^T \rangle_S = u^T \tilde{A} \tilde{A}^T v = u^T A A^T v = u^T S^{-1} v$ must be preserved, as indicated in Equation (6.45);

$$\begin{cases} \langle d_1^T, d_2^T \rangle_S = \langle e_1^T, e_2^T \rangle_S \\ \langle d_1^T, d_1^T \rangle_S = \langle e_2^T, e_2^T \rangle_S \\ \langle d_2^T, d_2^T \rangle_S = \langle e_1^T, e_1^T \rangle_S. \end{cases} \tag{6.59}$$

So far we have relied on the assumption that $\|AT\| = 1$. In general, that is not the case, so the transformed vectors must be scaled by the norm of AT. Taking this into account, we can write the conservation of the inner product as

$$\begin{cases} d_1^T S^{-1} d_2 = \lambda^{-2} e_1^T S^{-1} e_2 \\ d_1^T S^{-1} d_1 = \lambda^{-2} e_2^T S^{-1} e_2 \\ d_2^T S^{-1} d_2 = \lambda^{-2} e_1^T S^{-1} e_1 \end{cases} \tag{6.60}$$

where $\lambda^{-1} \doteq \|AT\|$. In order to eliminate the dependency on λ, we can consider ratios of inner products. We obtain therefore 2 independent constraints on the 6 independent components of the inner product matrix S which are called *Kruppa equations*.

$$\frac{\langle d_1^T, d_2^T \rangle_S}{\langle e_1^T, e_2^T \rangle_S} = \frac{\langle d_1^T, d_1^T \rangle_S}{\langle e_2^T, e_2^T \rangle_S} = \frac{\langle d_2^T, d_2^T \rangle_S}{\langle e_1^T, e_1^T \rangle_S}. \tag{6.61}$$

Given 3 or more fundamental matrices, taken from 3 or more sets of images using the same camera, Kruppa equations can be - in principle - solved for the calibration matrix S. We say "in principle" because in practice Kruppa equations are non-linear and quite sensitive to noise, as we will discuss later. The above derivation entails a choice of a matrix R_0 that satisfied (6.56). It is possible to avoid this arbitrary choice noticing that the fundamental matrix itself is acted upon as a covector. Therefore, without a choice of R_0, one can write, as in [**?**], a matrix version of Kruppa equations as

$$F S^{-1} F^T = -\lambda^{-2} \widehat{U} S^{-1} \widehat{U}. \tag{6.62}$$

where $\lambda^{-1} = \|U\|$ and $U = AT$ is in the null space of the fundamental matrix F. Note that the above derivation did not involve any projective geometry, nor complex loci such as the "Absolute conic", as traditionally done following [**?**].

Necessary and sufficient conditions for self-calibration

The above derivation is based on the assumption that the fundamental matrix F is normalized, i.e. $\|T'\| = 1$. However, since the epipolar constraint is homogeneous in the fundamental matrix F, it can only be determined up to an arbitrary scale. Suppose λ is the length of the vector $T' \in \mathbb{R}^3$ in $F = \widehat{T'}ARA^{-1}$. Consequently, the vectors ξ_1 and ξ_2 are also scaled by the same λ. Then the ratio between the left and right hand-side quantities in each equation of (**??**) is equal to λ^2. This reduces to two constraints on Ω^{-1}, known as *Kruppa's equations* after their discoverer in 1913

$$\boxed{\lambda^2 = \frac{\xi_1^T\Omega^{-1}\xi_1}{\eta_2^T\Omega^{-1}\eta_2} = \frac{\xi_2^T\Omega^{-1}\xi_2}{\eta_1^T\Omega^{-1}\eta_1} = \frac{\xi_1^T\Omega^{-1}\xi_2}{\eta_1^T\Omega^{-1}\eta_2}} \tag{6.63}$$

Equation (6.63) reveals the geometric meaning of the Kruppa ratio λ^2: it is the square of the length of the vector T' in the fundamental matrix F. This discovery turns out to be quite useful when we later discuss the renormalization of Kruppa's equations. Ideally, each fundamental matrix provides *at most two* algebraic constraints on Ω^{-1}, assuming that the two equations in (6.63) happen to be algebraically independent. Since the symmetric matrix Ω has five degrees of freedom, in general *at least three* fundamental matrices are needed to uniquely determine Ω. Nevertheless, as we will soon see, this is *not* the case for many special camera motions.

Comment 6.1. *One must be aware that solving Kruppa's equations for camera calibration is not equivalent to the camera self-calibration problem in the sense that there may exist solutions of Kruppa's equations which are not solutions of a "valid" self-calibration. Given a non-critical set of camera motions, the associated Kruppa equations do not necessarily give enough constraints to solve for the calibration matrix A. See Section 6.4.3 for a more detailed account.*

The above derivation of Kruppa's equations is straightforward, but the expression (6.63) depends on a particular choice of the rotation matrix R_0. However, there is an even simpler way to get an equivalent expression for Kruppa's equations in a matrix form. Given a normalized fundamental matrix $F = \widehat{T'}ARA^{-1}$, it is then straightforward to check that $\Omega^{-1} = AA^T$ must satisfy the following equation

$$F\Omega^{-1}F^T = \widehat{T'}\Omega^{-1}\widehat{T'}^T. \tag{6.64}$$

We call this equation the *normalized matrix Kruppa equation*. It is readily seen that this equation is equivalent to (**??**). If F is not normalized and is

scaled by $\lambda \in \mathbb{R}$, i.e. $F = \lambda \widehat{T'} A R A^{-1}$,[5] we then have the *matrix Kruppa equation*

$$\boxed{F\Omega^{-1}F^T = \lambda^2 \widehat{T'}\Omega^{-1}\widehat{T'}^T} \tag{6.65}$$

This equation is equivalent to the scalar version given by (6.63) and is independent of the choice of the rotation matrix R_0. Algebraic properties of Kruppa's equations have been extensively studied in the Computer Vision literature (see e.g. [MF92, ZF96]). However, we here are more concerned with conditions on dependency among the two Kruppa equations obtained from one fundamental matrix. Knowing such conditions, we may tell in practice whether a given set of Kruppa's equations suffice to guarantee a unique solution for calibration. As we will show in this chapter, for very rich class of camera motions which commonly occur in many practical applications, Kruppa's equations will become degenerate. Moreover, since Kruppa's equations (6.63) or (6.65) are nonlinear in Ω^{-1}, any self-calibration algorithms based on directly solving these equations would either be computationally expensive or have multiple local minima [Bou98, LF97]. So a more detailed study of Kruppa's equations is necessary: We need to know under what conditions they may have problems with such as degeneracy, and under what conditions they can be simplified and most importantly leads to linear solutions.

6.4 Self-calibration from special motions and chirality

In this section we discuss some special cases that deserve attention because they make self-calibration either simpler, or not possible at all. Note that subjecting the camera to special motions in order to simplify the calibration procedure requires access to the camera during the capture. Therefore, this technique can only be applied in a controlled scenario. At that point, however, one might as well use a rig, which makes the calibration simpler. For this reason, the reader can skip this section unless he/she is interested in gaining insight into Kruppa's equations and the practice of camera calibration.

6.4.1 Pure rotational motion

Given a fundamental matrix $F = \widehat{T'} A R A^{-1}$ with T' of unit length, the normalized matrix Kruppa equation (6.64) can be rewritten in the following

[5] Without loss of generality, from now on, we always assume $\|T'\| = 1$.

way

$$\widehat{T'}(\Omega^{-1} - ARA^{-1}\Omega^{-1}A^{-T}R^TA^T)\widehat{T'}^T = 0. \tag{6.66}$$

According to this form, if we define $C = ARA^{-1}$, a linear map, called the Lyapunov map, $\sigma : \mathbb{R}^{3\times3} \to \mathbb{R}^{3\times3}$ as $\sigma : X \mapsto X - CXC^T$, and a linear map $\tau : \mathbb{R}^{3\times3} \to \mathbb{R}^{3\times3}$ as $\tau : Y \mapsto \widehat{T'}Y\widehat{T'}^T$, then the solution Ω^{-1} of equation (6.66) is exactly the (symmetric real) kernel of the composition map

$$\tau \circ \sigma : \quad \mathbb{R}^{3\times3} \xrightarrow{\sigma} \mathbb{R}^{3\times3} \xrightarrow{\tau} \mathbb{R}^{3\times3}. \tag{6.67}$$

This interpretation of Kruppa's equations clearly decomposes the effects of the rotational and translational components of motion: if there is no translation, i.e. $T = 0$, then there is no map τ; if the translation is non-zero, the kernel is enlarged due to the composition with map τ. In general, the kernel of σ is only two-dimensional. A more precise statement is given by the lemma below:

Lemma 6.1 (A special Lyapunov map). *Given a rotation matrix R not of the form $e^{\widehat{u}k\pi}$ for some $k \in \mathbb{Z}$ and some $u \in \mathbb{R}^3$ of unit length, the symmetric real kernel associated with the map σ defined above is two-dimensional. Otherwise, the symmetric real kernel is four-dimensional if k is odd and six-dimensional if k is even.*

the proof of this lemma follows directly from the properties of a Lyapunov map [CD91]. According to this lemma, with two general rotations, the matrix Ω hence the camera calibration A can be uniquely determined. Let us see how this can be done from three images taken by a purely rotating camera. For a purely rotating camera, the coordinate transformation is

$$\mathbf{X}(t) = R(t)\mathbf{X}(t_0) \tag{6.68}$$

Corresponding image pairs $(\mathbf{x}'_1, \mathbf{x}'_2)$ then satisfy:

$$\lambda_2\mathbf{x}'_2 = ARA^{-1}\lambda_1\mathbf{x}'_1, \tag{6.69}$$

for some scales $\lambda_1, \lambda_2 \in \mathbb{R}_+$. Then the image correspondences satisfy a special version of epipolar constraint:

$$\widehat{\mathbf{x}'_2}ARA^{-1}\mathbf{x}'_1 = 0 \tag{6.70}$$

From this linear equation, in general, 4 pair of image correspondences uniquely determine the matrix $C := ARA^{-1} \in \mathbb{R}^{3\times3}$.[6] Recall that $\Omega^{-1} = AA^T$. It is then direct to check that:

$$\Omega^{-1} - C\Omega^{-1}C^T = 0. \tag{6.71}$$

[6]Note that from the equation we can only solve ARA^{-1} up to an arbitrary scale. But we also know that $\det(ARA^{-1})$ must be 1.

That is, Ω^{-1} has to be in the symmetric real kernel $\mathrm{SRKer}(L)$ of the Lyapunov map:

$$\begin{aligned} L : \mathbb{C}^{3\times 3} &\rightarrow \mathbb{C}^{3\times 3} \\ X &\mapsto X - CXC^T. \end{aligned} \tag{6.72}$$

As for how many matrices of the form $C_i := AR_iA^{-1}$ with $R_i \in SO(3)$ may determine the calibration matrix A, we have the following results as a corollary to Lemma 6.1:

Theorem 6.1 (Two rotational motions determine calibration). *Given two matrices $C_i = AR_iA^{-1} \in ASO(3)A^{-1}, i = 1, 2$ where $R_i = e^{\widehat{u}_i\theta_i}$ with $\|u_i\| = 1$ and θ_i's not equal to $k\pi, k \in \mathbb{Z}$, then $SRKer(L_1) \cap SRKer(L_2)$ is one-dimensional if and only if u_1 and u_2 are linearly independent.*

Proof. The necessity is obvious: if two rotation matrices R_1 and R_2 have the same eigenvectors hence $\mathrm{SRKer}(L_1) = \mathrm{SRKer}(L_2)$ where $L_i : X \mapsto X - C_iXC_i^T, i = 1, 2$. We now only need to prove the sufficiency. We may assume u_1 and u_2 are the two rotation axes of R_1 and R_2 respectively and are linearly independent. Since rotation angles of both R_1 and R_2 are not $k\pi$, both $\mathrm{SRKer}(L_1)$ and $\mathrm{SRKer}(L_2)$ are two-dimensional according to the Lemma 6.1. Since u_1 and u_2 are linearly independent, the matrices $Au_1u_1^TA^T$ and $Au_2u_2^TA^T$ are linearly independent and are in $\mathrm{SRKer}(L_1)$ and $\mathrm{SRKer}(L_2)$ respectively. Thus $\mathrm{SRKer}(L_1)$ is not fully contained in $\mathrm{SRKer}(L_2)$ because it already contains both $Au_2u_2^TA^T$ and $\Omega^{-1} = AA^T$ and could not possibly contains another independent $Au_1u_1^TA^T$. Hence their intersection $\mathrm{SRKer}(L_1) \cap \mathrm{SRKer}(L_2)$ has at most 1 dimension. $X = \Omega^{-1}$ is then the only solution in the intersection. □

According to this theorem, a simple way to calibrate an uncalibrated camera is to rotate it about two different axes. The self-calibration problem in this case becomes completely linear and a unique solution is also guaranteed with the conditions given above. However, in practice, one usually does not know where the center of a camera is and "rotating" the camera does not necessarily guarantee that the rotation is about the exact optical center. So the above algorithm can only be approximately applied given that practical conditions are close enough. This will certainly cause certain errors in the calibration matrix A finally estimated. In a general case with translational motion, however, the symmetric real kernel of the composition map $\tau \circ \sigma$ is usually three-dimensional hence the conditions for a unique calibration is much more complicated. Furthermore, as we will see in the next section, the dimension of the kernel is not always 3. In many cases of practical importance, this kernel may become four-dimensional instead, which in fact correspond to certain degeneracy of Kruppa's equations. Solutions for the unnormalized Kruppa's equations are even more complicated due to the unknown scale λ. Nevertheless, the following lemma shows that

the conditions on uniqueness depend only on the camera motion which somehow may simplify the situation a little bit:

Lemma 6.2. *Given a fundamental matrix $F = \widehat{T'}ARA^{-1}$ with $T' = AT$, a symmetric matrix $X \in \mathbb{R}^{3\times 3}$ is a solution of $FXF^T = \lambda^2\widehat{T'}X\widehat{T'}^T$ if and only if $Y = A^{-1}XA^{-T}$ is a solution of $EYE^T = \lambda^2\widehat{T}Y\widehat{T}^T$ with $E = \widehat{T}R$.*

The proof of this lemma is simply algebraic. This simple lemma, however, states a very important fact: given a set of fundamental matrices $F_i = \widehat{T_i'}AR_iA^{-1}$ with $T_i' = AT_i, i = 1, \ldots, n$, there is a one-to-one correspondence between the set of solutions of the equations

$$F_iXF_i^T = \lambda_i^2\widehat{T_i'}X\widehat{T_i'}^T, \quad i = 1, \ldots, n$$

and the set of solutions of the equations

$$E_iYE_i^T = \lambda_i^2\widehat{T}_iY\widehat{T}_i^T, \quad i = 1, \ldots, n$$

where $E_i = \widehat{T}_iR_i$ are essential matrices associated to the given fundamental matrices. Note that these essential matrices are determined only by the camera motion. Therefore, the conditions of uniqueness of the solution of Kruppa's equations only depend on the camera motion. Our next task is then to study *how* the solutions of Kruppa's equations depend on the camera motion and under what conditions the solutions can be simplified.

6.4.2 Translation perpendicular or parallel to rotation

From the derivation of Kruppa's equations (6.63) or (6.65), we observe that the reason why they are nonlinear is that we do not usually know the scale λ. It is then helpful to know under what conditions the matrix Kruppa equation will have the same solutions as the normalized one, i.e. with λ set to 1. Here we will study two special cases for which we are able to know directly what the missing λ is. The fundamental matrix can then be *renormalized* and we can therefore solve the camera calibration from the normalized matrix Kruppa equations, which are linear! These two cases are when the rotation axis is *parallel* or *perpendicular* to the translation. That is, if the motion is represented by $(R, T) \in SE(3)$ and the unit vector $u \in \mathbb{R}^3$ is the axis of $R \in SO(3)$,[7] then the two cases are when u is parallel or perpendicular to T. As we will soon see, these two cases are of great theoretical importance: Not only does the calibration algorithm become linear, but it also reveals certain subtleties of Kruppa's equations and explains when the nonlinear Kruppa equations are most likely to become ill-conditioned.

[7] R can always be written of the form $R = e^{\widehat{u}\theta}$ for some $\theta \in [0, \pi]$ and $u \in \mathbb{S}^2$.

Lemma 6.3. *Consider a camera motion $(R,T) \in SE(3)$ where $R = e^{\widehat{u}\theta}$, $\theta \in (0,\pi)$ and the axis $u \in \mathbb{R}^3$ is parallel or perpendicular to T. If $\gamma \in \mathbb{R}$ and positive definite matrix Y are a solution to the matrix Kruppa equation: $\widehat{T}RYR^T\widehat{T}^T = \gamma^2\widehat{T}Y\widehat{T}^T$ associated to the essential matrix $\widehat{T}R$, then we must have $\gamma^2 = 1$. Consequently, Y is a solution of the normalized matrix Kruppa equation: $\widehat{T}RYR^T\widehat{T}^T = \widehat{T}Y\widehat{T}^T$.*

Proof. Without loss of generality we assume $\|T\| = 1$. For the parallel case, let $x \in \mathbb{R}^3$ be a vector of unit length in the plane spanned by the column vectors of $\widehat{T}$. All such x lie on a unit circle. There exists $x_0 \in \mathbb{R}^3$ on the circle such that $x_0^T Y x_0$ is maximum. We then have $x_0^T RYR^T x_0 = \gamma^2 x_0^T Y x_0$, hence $\gamma^2 \leq 1$. Similarly, if we pick x_0 such that $x_0^T Y x_0$ is minimum, we have $\gamma^2 \geq 1$. Therefore, $\gamma^2 = 1$. For the perpendicular case, since the columns of $\widehat{T}$ span the subspace which is perpendicular to the vector T, the eigenvector u of R is in this subspace. Thus we have: $u^T RYR^T u = \gamma^2 u^T Y u \Rightarrow u^T Y u = \gamma^2 u^T Y u$. Hence $\gamma^2 = 1$ if Y is positive definite. □

Combining Lemma 6.3 and Lemma 6.2, we immediately have:

Theorem 6.2 (Renormalization of Kruppa's equations). *Consider an unnormalized fundamental matrix $F = \widehat{T'}ARA^{-1}$ where $R = e^{\widehat{u}\theta}$, $\theta \in (0,\pi)$ and the axis $u \in \mathbb{R}^3$ is parallel or perpendicular to $T = A^{-1}T'$. Let $e = T'/\|T'\| \in \mathbb{R}^3$. Then if $\lambda \in \mathbb{R}$ and a positive definite matrix Ω are a solution to the matrix Kruppa equation: $F\Omega^{-1}F^T = \lambda^2\widehat{e}\Omega^{-1}\widehat{e}^T$, we must have $\lambda^2 = \|T'\|^2$.*

This theorem claims that, for the two types of special motions considered here, there is no solution for λ in Kruppa's equation (6.65) besides the true scale of the fundamental matrix. Hence we can decompose the problem into finding λ first and then solving for Ω or Ω^{-1}. The following theorem allows to directly compute the scale λ in the two special cases for a given fundamental matrix:

Theorem 6.3 (Renormalization of the fundamental matrix). *Given an unnormalized fundamental matrix $F = \lambda\widehat{T'}ARA^{-1}$ with $\|T'\| = 1$, if $T = A^{-1}T'$ is parallel to the axis of R, then λ^2 is $\|F^T\widehat{T'}F\|$, and if T is perpendicular to the axis of R, then λ is one of the two non-zero eigenvalues of $F^T\widehat{T'}$.*

Proof. Note that, since $\widehat{T'}^T\widehat{T'}$ is a projection matrix to the plane spanned by the column vectors of $\widehat{T'}$, we have the identity $\widehat{T'}^T\widehat{T'}\widehat{T'} = \widehat{T'}$. First we prove the parallel case. It can be verified that, in general, $F^T\widehat{T'}F = \lambda^2\widehat{AR^TT}$. Since the axis of R is parallel to T, we have $R^TT = T$ hence $F^T\widehat{T'}F = \lambda^2\widehat{T'}$. For the perpendicular case, let $u \in \mathbb{R}^3$ be the axis of R. By assumption $T = A^{-1}T'$ is perpendicular to u. Then there exists $v \in \mathbb{R}^3$ such that $u = \widehat{T}A^{-1}v$. Then it is direct to check that $\widehat{T'}v$ is the eigenvector of $F^T\widehat{T'}$ corresponding to the eigenvalue λ. □

Then for these two types of special motions, the associated fundamental matrix can be immediately normalized by being divided by the scale λ. Once the fundamental matrices are normalized, the problem of finding the calibration matrix Ω^{-1} from the normalized matrix Kruppa equations (6.64) becomes a simple *linear* one. A normalized matrix Kruppa equation in general imposes *three* linearly independent constraints on the unknown calibration matrix given by (**??**). However, this is *no longer the case* for the special motions that we are considering here.

Theorem 6.4 (Degeneracy of the normalized Kruppa equations). *Consider a camera motion $(R, T) \in SE(3)$ where $R = e^{\widehat{u}\theta}$ has the angle $\theta \in (0, \pi)$. If the axis $u \in \mathbb{R}^3$ is parallel or perpendicular to T, then the normalized matrix Kruppa equation: $\widehat{T}RYR^T\widehat{T}^T = \widehat{T}Y\widehat{T}^T$ imposes only two linearly independent constraints on the symmetric matrix Y.*

Proof. For the parallel case, by restricting Y to the plane spanned by the column vectors of $\widehat{T}$, it yields a symmetric matrix $\tilde{Y}$ in $\mathbb{R}^{2\times 2}$. The rotation matrix $R \in SO(3)$ restricted to this plane is a rotation $\tilde{R} \in SO(2)$. The normalized matrix Kruppa equation is then equivalent to $\tilde{Y} - \tilde{R}\tilde{Y}\tilde{R}^T = 0$. Since $0 < \theta < \pi$, this equation imposes exactly two constraints on the three-dimensional space of 2×2 symmetric real matrices. The identity $I_{2\times 2}$ is the only solution. Hence the normalized Kruppa equation imposes exactly two linearly independent constraints on Y. For the perpendicular case, since u is in the plane spanned by the column vectors of $\widehat{T}$, there exist $v \in \mathbb{R}^3$ such that (u, v) form an orthonormal basis of the plane. Then the normalized matrix Kruppa equation is equivalent to

$$\widehat{T}RYR^T\widehat{T}^T = \widehat{T}Y\widehat{T}^T \quad \Leftrightarrow \quad (u,v)^T RYR^T(u,v) = (u,v)^T Y(u,v).$$

Since $R^T u = u$, the above matrix equation is equivalent to two equations

$$v^T RYu = v^T Yu, \quad v^T RYR^T v = v^T Yv. \tag{6.73}$$

These are the only two constraints on Y imposed by the normalized Kruppa equation. □

According to this theorem, although we can renormalize the fundamental matrix when rotation axis and translation are parallel or perpendicular, we only get two independent constraints from the resulting (normalized) Kruppa equation corresponding to a single fundamental matrix, i.e. one of the three linear equations in (**??**) depends on the other two. So degeneracy does occur to normalized Kruppa equations. Hence for these motions, in general, we still need three such fundamental matrices to uniquely determine the unknown calibration. What happens to the unnormalized Kruppa equations? If we do not renormalize the fundamental matrix and directly use the unnormalized Kruppa equations (6.63) to solve for calibration, the two nonlinear equations in (6.63) may become algebraically dependent. The following corollary shows that this is at least true for the case when the

translation is perpendicular to the rotation axis (e.g. a planar or orbital motion):

Corollary 6.1. *Consider a camera motion $(R,T) \in SE(3)$ where $R = e^{\widehat{u}\theta}$ has the angle $\theta \in (0,\pi)$. If the axis $u \in \mathbb{R}^3$ is perpendicular to T, then the unnormalized matrix Kruppa equation: $\widehat{T}RYR^T\widehat{T}^T = \gamma^2\widehat{T}Y\widehat{T}^T$ imposes only one algebraically independent constraints on the symmetric matrix Y.*

Proof. From the proof of Theorem 6.4, we know that any other linear constraint on Y imposed by the normalized Kruppa equations must be a "linear combination" the two given in (6.73)

$$\alpha v^T RYu + \beta v^T RYR^T v = \alpha v^T Yu + \beta v^T Yv, \quad \alpha, \beta \in \mathbb{R}.$$

Hence after eliminating γ, the unnormalized matrix Kruppa equation gives

$$\frac{v^T Yu}{v^T RYu} = \frac{v^T Yv}{v^T RYR^T v} = \frac{\alpha v^T Yu + \beta v^T Yv}{\alpha v^T RYu + \beta v^T RYR^T v}.$$

The first equation obviously implies the second regardless of values of α, β. □

Therefore, when the translation is perpendicular to the rotation axis, one can only get one (unnormalized Kruppa) constraint, as opposed to the expected two, on the unknown calibration Ω^{-1} from each fundamental matrix. However, the above arguments do not always apply to the parallel case. But our study provides a more precise picture about how many independent (linear or algebraic) constraints that one may get at most in different situations. This is summarized in Table 6.1. Although, mathematically, motion involving translation either parallel or perpendicular to the rotation is only a zero-measure subset of $SE(3)$, they are very commonly encountered in applications: Many images sequences are usually taken by moving the camera around an object in trajectory composed of *planar motion* or *orbital motion*, in which case the rotation axis and translation direction are likely perpendicular to each other. Our analysis shows that, for these types of motions, even if a unique calibration may exist from the given data,

Table 6.1. Maximum numbers of independent constraints given by Kruppa's equations and the angle $\phi \in [0,\pi)$ between the rotation and translation.

Cases	Type of Constraints	# of Constraints on Ω^{-1}
$(\phi \neq 0, \frac{\pi}{2})$	Unnormalized	≤ 2
	Normalized	≤ 3
$(\phi = 0)$	Unnormalized	≤ 2
	Normalized	≤ 2
$(\phi = \frac{\pi}{2})$	Unnormalized	≤ 1
	Normalized	≤ 2

a self-calibration algorithm based on directly solving Kruppa's equations (6.63) is likely to be ill-conditioned [Bou98]. To intuitively demonstrate the practical significance of our results, we give an example in Figure 6.2. Our analysis reveals that in these cases, it is crucial to renormalize Kruppa's equation using Theorem 6.4: once the fundamental matrix or Kruppa's equations are renormalized, not only is one more constraint recovered, but we also obtain linear constraints (normalized Kruppa's equations).

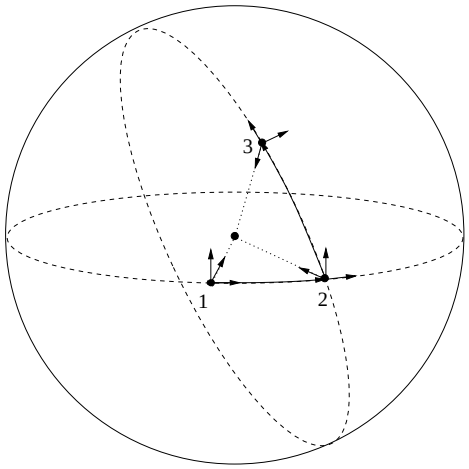

Figure 6.2. Two consecutive orbital motions with independent rotations: The camera optical axis always pointing to the center of the globe. Even if all pairwise fundamental matrices among the three views are considered, one only gets at most $1 + 1 + 2 = 4$ effective constraints on the camera intrinsic matrix if one uses the three unnormalized matrix Kruppa equations. At least one more view is need if one wishes to uniquely determine the unknown calibration. However, using renormalization instead, we may get back to $2 + 2 + 2 \geq 5$ constraints from only the given three views.

Comment 6.2 (Solutions of the normalized Kruppa equations). *Claims of Theorem 6.4 run contrary to the claims of Propositions B.5 hence B.9 in [ZF96]: In Proposition B.5 of [ZF96], it is claimed that the solution space of the normalized Kruppa's equations when the translation is parallel or perpendicular to the rotation axis is* two *or* three*-dimensional. In Theorem 6.4, we claim that the solution space is always* four*-dimensional. Theorem 6.4 does not cover the case when the rotation angle θ is π. However, if one allows the rotation to be π, the solutions of normalized Kruppa's equations are even more complicated. For example, we know $\widehat{T}e^{\widehat{u}\pi} = -\widehat{T}$ if u is of unit length and parallel to T (see [MKS00]). Therefore, if $R = e^{\widehat{u}\pi}$, the corresponding normalized Kruppa equation is completely degenerate and imposes no constraints at all on the calibration matrix.*

Comment 6.3 (Number of solutions). *Although Theorem 6.3 claims that for the perpendicular case λ is one of the two non-zero eigenvalues of $F^T\widehat{T'}$, unfortunately, there is no way to tell which one is the correct one – simulations show that it could be either the larger or smaller one. Therefore, in a numerical algorithm, for given $n \geq 3$ fundamental matrices, one needs to consider all possible 2^n combinations. According to Theorem 6.2, in the noise-free case, only one of the solutions can be positive definite, which corresponds to the true calibration.*

6.4.3 Calibration with chirality

It can be shown that if the scene is *rich enough*, then two general motions with rotation around different axes already determine a unique Euclidean solution for camera motion, calibration and scene structure (which will be explained later on). However, the two Kruppa equations obtained from these two motions will only give us *at most four* constraints on Ω which has *five* degrees of freedom. We hence need to know what information is missing from Kruppa's equations. Stated alternatively, can we get extra independent constraints on Ω from the fundamental matrix other than Kruppa's equations? The proof of Theorem 6.3 suggests another equation can be derived from the fundamental matrix $F = \lambda\widehat{T'}ARA^{-1}$ with $\|T'\| = 1$. Since $F^T\widehat{T'}F = \lambda^2\widehat{AR^TT}$, we can obtain the vector $\alpha = \lambda^2 AR^TT = \lambda^2 AR^TA^{-1}T'$. Then it is obvious that the following equation for $\Omega = A^{-T}A^{-1}$ holds

$$\alpha^T\Omega\alpha = \lambda^4 {T'}^T\Omega T'. \tag{6.74}$$

Notice that this is a constraint on Ω, not like Kruppa's equations which are constraints on Ω^{-1}. Combining Kruppa's equations given in (6.63) with (6.74) we have

$$\lambda^2 = \frac{\xi_1^T\Omega^{-1}\xi_1}{\eta_2^T\Omega^{-1}\eta_2} = \frac{\xi_2^T\Omega^{-1}\xi_2}{\eta_1^T\Omega^{-1}\eta_1} = \frac{\xi_1^T\Omega^{-1}\xi_2}{\eta_1^T\Omega^{-1}\eta_2} = \sqrt{\frac{\alpha^T\Omega\alpha}{{T'}^T\Omega T'}}. \tag{6.75}$$

Is the last equation algebraically independent of the two Kruppa equations? Although it seems to be quite different from Kruppa's equations, it is in fact dependent on them. This can be shown either numerically or using simple algebraic tools such as Maple. Thus, it appears that our effort to look for extra *independent* constraints on A from the fundamental matrix has failed.[8] In the following, we will give an explanation to this by showing that not all Ω which satisfy the Kruppa equations may give *valid* Euclidean reconstructions of both the camera motion and scene structure. The extra

[8] Nevertheless, extra *implicit* constraints on A may still be obtained from other algebraic facts. For example, the so-called *modulus constraints* give three implicit constraints on A by introducing three extra unknowns, for more details see [PG99].

constraints which are missing in Kruppa's equations are in fact captured by the so-called *chirality constraint*, which was previously studied in [Har98]. We now give a clear and concise description between the relationship of the Kruppa equations and chirality.

Theorem 6.5 (Kruppa's equations and chirality). *Consider a camera with calibration matrix I and motion (R, T). If $T \neq 0$, among all the solutions $Y = A^{-1}A^{-T}$ of Kruppa's equation $EYE^T = \lambda^2 \widehat{T} Y \widehat{T}^T$ associated to $E = \widehat{T}R$, only those which guarantee $ARA^{-1} \in SO(3)$ may provide a* valid Euclidean reconstruction *of both camera motion and scene structure in the sense that any other solution pushes some plane $N \subset \mathbb{R}^3$ to the plane at infinity, and feature points on different sides of the plane N have different signs of recovered depth.*

Proof. The images $\mathbf{x}_2, \mathbf{x}_1$ of any point $p \in \mathbb{R}^3$ satisfy the coordinate transformation

$$\lambda_2 \mathbf{x}_2 = \lambda_1 R \mathbf{x}_1 + T.$$

If there exists $Y = A^{-1}A^{-T}$ such that $EYE^T = \lambda^2 \widehat{T} Y \widehat{T}^T$ for some $\lambda \in \mathbb{R}$, then the matrix $F = A^{-T}EA^{-1} = \widehat{T'}ARA^{-1}$ is also an essential matrix with $T' = AT$, that is, there exists $\tilde{R} \in SO(3)$ such that $F = \widehat{T'}\tilde{R}$ (see [May93] for an account of properties of essential matrices). Under the new calibration A, the coordinate transformation is in fact

$$\lambda_2 A\mathbf{x}_2 = \lambda_1 ARA^{-1}(A\mathbf{x}_1) + T'.$$

Since $F = \widehat{T'}\tilde{R} = \widehat{T'}ARA^{-1}$, we have $ARA^{-1} = \tilde{R} + T'v^T$ for some $v \in \mathbb{R}^3$. Then the above equation becomes: $\lambda_2 A\mathbf{x}_2 = \lambda_1 \tilde{R}(A\mathbf{x}_1) + \lambda_1 T' v^T (A\mathbf{x}_1) + T'$. Let $\beta = \lambda_1 v^T (A\mathbf{x}_1) \in \mathbb{R}$, we can further rewrite the equation as

$$\lambda_2 A\mathbf{x}_2 = \lambda_1 \tilde{R} A\mathbf{x}_1 + (\beta + 1)T'. \tag{6.76}$$

Nevertheless, with respect to the solution A, the reconstructed images $A\mathbf{x}_1, A\mathbf{x}_2$ and $(\tilde{R}, T')$ must also satisfy

$$\gamma_2 A\mathbf{x}_2 = \gamma_1 \tilde{R} A\mathbf{x}_1 + T' \tag{6.77}$$

for some scale factors $\gamma_1, \gamma_2 \in \mathbb{R}$. Now we prove by contradiction that $v \neq 0$ is impossible for a valid Euclidean reconstruction. Suppose that $v \neq 0$ and we define the plane $N = \{p \in \mathbb{R}^3 | v^T p = -1\}$. Then for any $p = \lambda_1 A\mathbf{x}_1 \in N$, we have $\beta = -1$. Hence, from (6.76), $A\mathbf{x}_1, A\mathbf{x}_2$ satisfy $\lambda_2 A\mathbf{x}_2 = \lambda_1 \tilde{R} A\mathbf{x}_1$. Since $A\mathbf{x}_1, A\mathbf{x}_2$ also satisfy (6.77) and $T' \neq 0$, both γ_1 and γ_2 in (6.77) must be ∞. That is, the plane N is "pushed" to the plane at infinity by the solution A. For points not on the plane N, we have $\beta + 1 \neq 0$. Comparing the two equations (6.76) and (6.77), we get $\gamma_i = \lambda_i/(\beta + 1), i = 1, 2$. Then for a point in the far side of the plane N, i.e. $\beta + 1 < 0$, the recovered depth scale γ is negative; for a point! in the near side of N, i.e. $\beta + 1 > 0$, the recovered depth scale γ is positive. Thus, we must have that $v = 0$. □

Comment 6.4 (Quasi-affine reconstruction). *Theorem 6.5 essentially implies the chirality constraints studied in [Har98]. According to the above theorem, if only finitely many feature points are measured, a solution of the calibration matrix A which may allow a valid Euclidean reconstruction should induce a plane N not cutting through the convex hull spanned by all the feature points and camera centers. Such a reconstruction is referred as quasi-affine in [Har98].*

It is known from Theorem 6.1 that, in general, two matrices of the form ARA^{-1} determine a unique camera calibration A. Thus, following Theorem 6.5, *in principle*, a camera calibration can be *uniquely* determined by two independent rotations regardless of translation if enough (usually infinitely many) feature points are available. An intuitive example is provided in Figure 6.3. The significance of Theorem 6.5 is that it explains why we get only

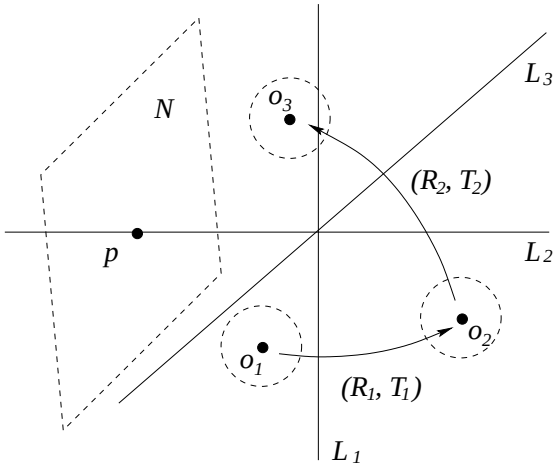

Figure 6.3. A camera undergoes two motions (R_1, T_1) and (R_2, T_2) observing a rig consisting of three straight lines L_1, L_2, L_3. Then the camera calibration is uniquely determined as long as R_1 and R_2 have independent rotation axes and rotation angles in $(0, \pi)$, regardless of T_1, T_2. This is because, for any invalid solution A, the associated plane N (see the proof of Theorem 6.5) must intersect the three lines at some point, say p. Then the reconstructed depth of point p with respect to the solution A would be infinite (points beyond the plane N would have negative recovered depth). This gives us a criteria to exclude all such invalid solutions.

two constraints from one fundamental matrix even in the two special cases when Kruppa's equations can be renormalized – extra ones are imposed by the structure, not the motion. The theorem also resolves the discrepancy between Kruppa's equations and the necessary and sufficient condition for a unique calibration: Kruppa's equations, although convenient to use, do not provide sufficient conditions for a valid calibration which allows a valid

Euclidean reconstruction of both the camera motion and scene structure. However, the fact given in Theorem 6.5 is somewhat difficult to harness in algorithms. For example, in order to exclude invalid solutions, one needs feature points on or beyond the plane N.[9] Alternatively, if such feature points are not available, one may first obtain a *projective reconstruction* and then use the so-called *absolute quadric constraints* to calibrate the camera [Tri97]. However, in such a method, the camera motion needs to satisfy a more restrictive condition than requiring only two independent rotations, i.e. it cannot be *critical* in the sense specified in [Stu97].

6.5 Calibration from continuous motion

[10] So far, we have mainly considered camera self-calibration when the motion of the camera is discrete – positions of the camera are specified as discrete points in $SE(3)$. In this section, we study its continuous version. Define the angular velocity $\widehat{\omega} = \dot{R}(t)R^T(t) \in so(3)$ and linear velocity $v = -\widehat{\omega}T(t) + \dot{T}(t) \in \mathbb{R}^3$ and. Let $v' = Av \in \mathbb{R}^3$, $\omega' = A\omega \in \mathbb{R}^3$. Differentiating the equation (6.26) with respect to time t, we obtain:

$$\dot{\mathbf{X}}' = A\widehat{\omega}A^{-1}\mathbf{X}' + v' \tag{6.78}$$

where $\mathbf{X}' = A\mathbf{X}$.

Uncalibrated continuous epipolar geometry

By the general case we mean that both the angular and linear velocities ω and v are non-zero. Note that $\mathbf{X}' = \lambda\mathbf{x}$ yields $\dot{\mathbf{X}}' = \dot{\lambda}\mathbf{x} + \lambda\dot{\mathbf{x}}$. Then (6.78) gives an uncalibrated version of the continuous epipolar constraint

$$\boxed{\dot{\mathbf{x}}^T A^{-T}\widehat{v}A^{-1}\mathbf{x} + \mathbf{x}^T A^{-T}\widehat{\omega}\widehat{v}A^{-1}\mathbf{x} = 0} \tag{6.79}$$

This will still be called the *continuous epipolar constraint*. As before, let $s \in \mathbb{R}^{3\times 3}$ to be $s = \frac{1}{2}(\widehat{\omega}\widehat{v} + \widehat{v}\widehat{\omega})$. Define the *continuous fundamental matrix* $F \in \mathbb{R}^{6\times 3}$ to be:

$$F = \begin{pmatrix} A^{-T}\widehat{v}A^{-1} \\ A^{-T}sA^{-1} \end{pmatrix}. \tag{6.80}$$

F can therefore be estimated from as few as eight optical flows $(\mathbf{x}, \dot{\mathbf{x}})$ from (6.79). Note that $\widehat{v'} = A^{-T}\widehat{v}A^{-1}$ and $\widehat{\omega'} = A^{-T}\widehat{\omega}A^{-1}$. Applying

[9] Some possible ways of harnessing the constraints provided by chirality have been discussed in [Har98]. Basically they give *inequality* constraints on the possible solutions of the calibration.

[10] This section may be skipped at a first reading

this property of the hat operator $\widehat{(\cdot)}$ repeatedly, we obtain

$$\begin{aligned} A^{-T}sA^{-1} &= \frac{1}{2}A^{-T}(\widehat{\omega}\widehat{v}+\widehat{v}\widehat{\omega})A^{-1} \\ &= \frac{1}{2}(A^{-T}\widehat{\omega}A^{T}\widehat{v'}+\widehat{v'}A\widehat{\omega}A^{-1}) = \frac{1}{2}(\widehat{\omega'}\Omega^{-1}\widehat{v'}+\widehat{v'}\Omega^{-1}\widehat{\omega'}). \end{aligned}$$

Then the continuous epipolar constraint (6.79) is equivalent to

$$\dot{\mathbf{x}}^T\widehat{v'}\mathbf{x}+\mathbf{x}^T\frac{1}{2}(\widehat{\omega'}\Omega^{-1}\widehat{v'}+\widehat{v'}\Omega^{-1}\widehat{\omega'})\mathbf{x}=0. \tag{6.81}$$

Suppose $\Omega^{-1} = BB^T$ for another $B \in SL(3)$, then $A = BR_0$ for some $R_0 \in SO(3)$. We have

$$\begin{aligned} & \dot{\mathbf{x}}^T\widehat{v'}\mathbf{x}+\mathbf{x}^T\frac{1}{2}(\widehat{\omega'}\Omega^{-1}\widehat{v'}+\widehat{v'}\Omega^{-1}\widehat{\omega'})\mathbf{x}=0 \\ \Leftrightarrow\quad & \dot{\mathbf{x}}^T\widehat{v'}\mathbf{x}+\mathbf{x}^T\frac{1}{2}(\widehat{\omega'}BB^T\widehat{v'}+\widehat{v'}BB^T\widehat{\omega'})\mathbf{x}=0 \\ \Leftrightarrow\quad & \dot{\mathbf{x}}^TB^{-T}\widehat{R_0v}B^{-1}\mathbf{x}+\mathbf{x}^TB^{-T}\widehat{R_0\omega}\widehat{R_0v}B^{-1}\mathbf{x}=0. \end{aligned} \tag{6.82}$$

Comparing to (6.79), one cannot tell the camera A with motion (ω, v) from the camera B with motion $(R_0\omega, R_0v)$. Thus, like the discrete case, without knowing the camera motion the calibration can only be recovered in the space $SL(3)/SO(3)$, i.e. only the symmetric matrix Ω^{-1} hence Ω can be recovered.

However, unlike the discrete case, the matrix Ω cannot be fully recovered in the continuous case. Since $\Omega^{-1} = AA^T$ is a symmetric matrix, it can be diagonalized as

$$\Omega^{-1} = R_1^T\Sigma R_1, \quad R_1 \in SO(3) \tag{6.83}$$

where $\Sigma = \mathrm{diag}\{\sigma_1, \sigma_2, \sigma_3\}$. Then let $\omega'' = R_1\omega'$ and $v'' = R_1v'$. Applying the property of the hat operator, we have

$$\begin{aligned} \widehat{v'} &= R_1^T\widehat{v''}R_1 \\ \frac{1}{2}(\widehat{\omega'}\Omega^{-1}\widehat{v'}+\widehat{v'}\Omega^{-1}\widehat{\omega'}) &= R_1^T\frac{1}{2}(\widehat{\omega''}\Sigma\widehat{v''}+\widehat{v''}\Sigma\widehat{\omega''})R_1. \end{aligned} \tag{6.84}$$

Thus the continuous epipolar constraint (6.79) is also equivalent to:

$$(R_1\dot{\mathbf{x}})^T\widehat{v''}(R_1\mathbf{x})+(R_1\mathbf{x})^T\frac{1}{2}(\widehat{\omega''}\Sigma\widehat{v''}+\widehat{v''}\Sigma\widehat{\omega''})(R_1\mathbf{x})=0. \tag{6.85}$$

From this equation, one can see that there is no way to tell a camera A with $AA^T = R_1^T\Sigma R_1$ from a camera $B = R_1A$. Therefore, only the diagonal matrix Σ can be recovered as camera parameters if both the scene structure and camera motion are unknown. Note that Σ is in $SL(3)$ hence $\sigma_1\sigma_2\sigma_3 = 1$. The singular values only have two degrees of freedom. Hence we have:

Theorem 6.6 (Singular values of calibration matrix from continuous motion). *Consider a camera with an unknown calibration matrix*

$A \in SL(3)$ undergoing an unknown motion (ω, v). Then only the eigenvalues of AA^T, i.e. singular values of A, can be recovered from the bilinear continuous epipolar constraint.

If we define that two matrices in $SL(3)$ are equivalent if and only if they have the same singular values. The intrinsic parameter space is then reduced to the space $SL(3)/\sim$ where $\sim$ represents this equivalence relation. The fact that only two camera parameters can be recovered was known to Brooks *et al.* [BCB97]. They have also shown how to do self-calibration for matrices A with only two unknown parameters using the above continuous method.

Comment 6.5. *It is a little surprising to see that the discrete and continuous cases have some significant difference for the first time, especially knowing that in the calibrated case these two cases have almost exactly parallel sets of theory and algorithms. We believe that this has to do with the map:*

$$\begin{aligned} \gamma_A : \mathbb{R}^{3\times 3} &\rightarrow \mathbb{R}^{3\times 3} \\ B &\mapsto ABA^T \end{aligned}$$

where A is an arbitrary matrix in $\mathbb{R}^{3\times 3}$. Let $so(3)$ be the Lie algebra of $SO(3)$. The restricted map $\gamma_A \mid_{so(3)}$ is an endomorphism while $\gamma_A \mid_{SO(3)}$ is not. Consider $\gamma_A \mid_{so(3)}$ to be the first order approximation of $\gamma_A \mid_{SO(3)}$. Then the information about the calibration matrix A does not fully show up until the second order term of the map γ_A. This also somehow explains why in the discrete case the constraints that we can get for A (essentially Kruppa's equations) are in general nonlinear.

Comment 6.6. *From the above discussion, if one only uses the (bilinear) continuous epipolar constraint, at most two intrinsic parameters of the calibration matrix A can be recovered. However, it is still possible that the full information about A can be recovered from multilinear constraints on the higher order derivatives of optical flow. A complete list of such constraints will be given later on when we study multiple view geometry. For now, a curious reader may refer to [Åst96].*

Pure rotational motion

Since full calibration is not possible in the general case when translation is present, we need to know if it is possible in some special case. The only case left is when there is only rotational motion, i.e. the linear velocity v is always zero. In this case the continuous fundamental matrix is no longer well defined. However from the equation (6.78) we have:

$$\dot{\mathbf{X}}' = A\widehat{\omega}A^{-1}\mathbf{X}' \quad \Rightarrow \quad \dot{\lambda}\mathbf{x} + \lambda\dot{\mathbf{x}} = A\widehat{\omega}\lambda A^{-1}\mathbf{x} \quad \Rightarrow \quad \widehat{\mathbf{x}}\dot{\mathbf{x}} = \widehat{\mathbf{x}}A\widehat{\omega}A^{-1}\mathbf{x} \tag{6.86}$$

This is a special version of the continuous epipolar constraint and it gives two independent constraints on the matrix $C := A\widehat{\omega}A^{-1}$ for each $(\mathbf{x}, \dot{\mathbf{x}})$.

Given $n \geq 4$ optical flow measurements $\{(\mathbf{x}_i, \dot{\mathbf{x}}_i)\}_{i=1}^n$, one may uniquely determine the matrix $A\widehat{\omega}A^{-1}$ by solving a linear equation:

$$Mc = b \tag{6.87}$$

where $M \in \mathbb{R}^{2n\times 9}$ is a matrix function of $\{(\mathbf{x}_i, \dot{\mathbf{x}}_i)\}_{i=1}^n$, $b \in \mathbb{R}^9$ is a vector function of $\widehat{\mathbf{x}}_i\dot{\mathbf{x}}_i$'s and $c \in \mathbb{R}^9$ is the 9 entries of $C = A\widehat{\omega}A^{-1}$. The unique solution is given by the following lemma:

Lemma 6.4. *If $\omega \neq 0$, then $A\widehat{\omega}A^{-1} = C_L - \gamma I$ where $C_L \in \mathbb{R}^{3\times 3}$ is the matrix corresponding to the least-squares solution of the equation $Mc = b$ and γ is the unique real eigenvalue of C_L.*

The proof is straightforward. Then the self-calibration problem becomes how to recover $\Omega = A^{-T}A^{-1}$ or $\Omega^{-1} = AA^T$ from matrices of the form $A\widehat{\omega}A^{-1}$. Without loss of generality, we may assume ω is of unit length.[11] Notice that this problem then becomes a continuous version of the discrete pure rotational case. Since $C = A\widehat{\omega}A^{-1} \in \mathbb{R}^{3\times 3}$, it is direct to check that

$$C\Omega^{-1} + \Omega^{-1}C^T = 0.$$

That is, $\Omega^{-1} = AA^T$ has to be in the kernel of the Lyapunov map:

$$\begin{aligned} L : \mathbb{C}^{3\times 3} &\rightarrow \mathbb{C}^{3\times 3} \\ X &\mapsto CX + XC^T \end{aligned} \tag{6.88}$$

If $\omega \neq 0$, the eigenvalues of $\widehat{\omega}$ have the form $0, i\alpha, -i\alpha$ with $\alpha \in \mathbb{R}$. Let the corresponding eigenvectors are $\omega, u, \bar{u} \in \mathbb{C}^3$. According to Callier and Desoer [CD91], the kernel of the map L has three dimensions and is given by:

$$\mathrm{Ker}(L) = \mathrm{span}\{X_1 = A\omega\omega^*A^T, X_2 = Auu^*A^T, X_3 = A\bar{u}\bar{u}^*A^T\}. \tag{6.89}$$

As in the discrete case, the symmetric real X is of the form $X = \beta X_1 + \gamma(X_2 + X_3)$ for some $\beta, \gamma \in \mathbb{R}$. That is the symmetric real kernel of L is only two-dimensional. We denote this space as $\mathrm{SRKer}(L)$. We thus have:

Lemma 6.5. *Given a matrix $C = A\widehat{\omega}A^{-1}$ with $\|\omega\| = 1$, the symmetric real kernel associated with the Lyapunov map $L : CX + XC^T$ is two-dimensional.*

Similar to the discrete case we have:

Theorem 6.7 (Two continuous rotational motions determine calibration). *Given matrices $C_i := A\widehat{\omega}_i A^{-1} \in \mathbb{R}^{3\times 3}, i = 1, \ldots, n$ with $\|\omega_i\| = 1$. The symmetric matrix $\Omega = A^{-T}A^{-1} \in SL(3)$ is uniquely determined if and only if at least* two *of the n vectors $\omega_i, i = 1, \ldots, n$ are linearly independent.*

[11] If not, the norm of the two non-zero eigenvalues of $A\widehat{\omega}A^{-1}$ is exactly the length of ω.

6.6 Three stage stratification

6.6.1 Projective reconstruction

The approach outlined in the previous section utilized the intrinsic constraints (expressed in terms of image measurements or quantities directly computable from them, i.e. F and T') for the recovery of the metric of the uncalibrated space Ω. The sufficient and necessary conditions for successful self-calibration stated that at least three views are needed, such that they are related by nonzero translations T_1, T_2 and rotations R_1, R_2 are around two linearly independent axes.

In the following section we will look at some cases where sufficient and necessary conditions are not satisfied and explore some additional avenues. Let us first revisit the epipolar geometry of two views in the uncalibated case. As we have already discussed at the beginning of this chapter, the fundamental matrix cannot be uniquely factorized into a skew-symmetric matrix and nonsingular matrix, due to the fact that there is a family of transformations parameterized by $\mathbf{v}$, which gives rise to the same fundamental matrix:

$$\mathbf{x}_2'^T \widehat{T}' A R A^{-1} \mathbf{x}_1' \;=\; \mathbf{x}_2'^T \widehat{T}' (A R A^{-1} + T' \mathbf{v}^T) \mathbf{x}_1' \;=\; 0. \tag{6.90}$$

since $\widehat{T}'(T'\mathbf{v}^T) = 0$.

Below we demonstrate that these possible decompositions of F yield projection matrices, which are related to each other by non-singular matrices $H \in \mathcal{GL}(4)$. The following theorem is due to Hartley [HZ00]. Without loss of generality we assume that the projection matrix associated with the first camera is $P_1 = [I, 0]$.

Theorem 6.8 (Projective reconstruction). *Let F be the fundamental matrix and (P_1, P_2) and $(\tilde{P}_1, \tilde{P}_2)$ are two possible pairs of projection matrices, yielding the same fundamental matrix F. Then there exists a nonsingular transformation $H \in \mathcal{GL}(4)$ such that $P_1 = H\tilde{P}_1$ and $P_2 = H\tilde{P}_2$.*

Suppose that there are two possible decompositions of F such that $F = \widehat{\mathbf{a}}A$ and $F = \widehat{\mathbf{b}}B$, then $\mathbf{b} = \lambda\mathbf{a}$ and $B = \lambda^{-1}(A + \mathbf{a}\mathbf{v}^T)$. Since $\widehat{\mathbf{a}}A = \widehat{\mathbf{b}}B$, then $\widehat{\mathbf{a}}(\lambda B - A) = 0$, so $\lambda B - A = \mathbf{a}\mathbf{v}^T$, hence $B = \lambda^{-1}(A + \mathbf{a}\mathbf{v}^T)$. What remains to be verified that given the projections matrices $P = [A, \mathbf{a}]$ and $\tilde{P} = [\lambda^{-1}(A + \mathbf{a}\mathbf{v}^T), \lambda\mathbf{a}]$ they are indeed related by the matrix H, such that $PH = \tilde{P}$, with H:

$$H = \begin{bmatrix} \lambda^{-1} I & 0 \\ \lambda^{-1}\mathbf{v}^T & \lambda \end{bmatrix}. \tag{6.91}$$

Canonical decomposition of F

The previous argument demonstrated that there is an entire 4 parameter family of transformations parameterized by $(\lambda, \mathbf{v})$, consistent with F. In order to be able to proceed with the reconstruction we need a systematic way to choose a representative from this family. Such a canonical decomposition of F consistent with a set of measurements has the following form:

$$P_1 = [I, 0] \text{ and } P_2 = [\widehat{T}'F, T'].$$

This assumes that the structure is expressed in the coordinate frame of the first camera with the projection matrix $P_1 = [I, 0] \in \mathbb{R}^{3\times 4}$ and projection matrix P_2 capturing the displacement between two views then becomes $P_2 = [\widehat{T}'F, T']$. Note that T' is the epipole in the second view, $F^T T' = 0$ and can be directly computed from F^T as its null space; hence, this decomposition can be obtained directly from image correspondences (by computing F) and is consistent with the fundamental matrix F. The image measurements and the canonical projection matrices are now linearly related to the unknown structure $\mathbf{X}$:

$$\begin{array}{rcccc} \lambda_1 \mathbf{x}_1' & = & P_1\mathbf{X} & = & [I, 0]\mathbf{X} \\ \lambda_2 \mathbf{x}_2' & = & P_2\mathbf{X} & = & [\widehat{T}'F, T']\mathbf{X} \end{array} \tag{6.92}$$

The presence of the ambiguity $H \in \mathcal{GL}(4)$ in the choice of the projection matrices reflects the fact that structure can be recovered given the choice of P_1 and P_2. Since the projection matrices P_1, P_2 are related to the actual displacement by an unknown transformation $H \in \mathcal{GL}(4)$, then the ambiguity associated with the Euclidean structure $\mathbf{X}$ is characterized by the inverse of transformation H^{-1}:

$$\begin{array}{rcccc} \lambda_1 \mathbf{x}_1' & = & P_1 H H^{-1}\mathbf{X} & = & \tilde{P}_1\mathbf{X}_p \\ \lambda_2 \mathbf{x}_2' & = & P_2 H H^{-1}\mathbf{X} & = & \tilde{P}_2\mathbf{X}_p \end{array} \tag{6.93}$$

The recovered structure $\mathbf{X}_p = H^{-1}\mathbf{X}$ is so called *projective structure*. The relationship in equation (6.92) enables us to proceed with the recovery of $\mathbf{X}_p$. Eliminating unknown scales λ_1 and λ_2 by multiplying both sides of equation by $\widehat{\mathbf{x}}_1'$ and $\widehat{\mathbf{x}}_2'$ in equation 6.92 we obtain three equations per point, only two of which are linearly independent. Hence two corresponding points $\mathbf{x}_1'$ and $\mathbf{x}_2'$ yield four independent constraints on $\mathbf{X}_p$

$$\begin{array}{rcl} x_1^i(\mathbf{p}_3^{1T}\mathbf{X}_p) & = & \mathbf{p}_1^{1T}\mathbf{X}_p \\ y_1^i(\mathbf{p}_3^{1T}\mathbf{X}_p) & = & \mathbf{p}_2^{1T}\mathbf{X}_p \\ x_2^i(\mathbf{p}_3^{2T}\mathbf{X}_p) & = & \mathbf{p}_1^{2T}\mathbf{X}_p \\ y_2^i(\mathbf{p}_3^{2T}\mathbf{X}_p) & = & \mathbf{p}_2^{2T}\mathbf{X}_p \end{array} \tag{6.94}$$

where $P_1 = [\mathbf{p}_1^1, \mathbf{p}_2^1, \mathbf{p}_3^1]^T$ and $P_2 = [\mathbf{p}_1^2, \mathbf{p}_2^2, \mathbf{p}_3^2]^T$ are the projection matrices described in terms of their rows. Structure can be then recovered as a linear least squares solution to the system of homogeneous equations

$$A\mathbf{X}_p = 0$$

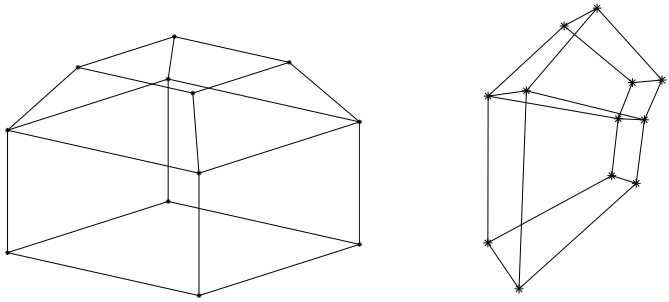

Figure 6.4. The true Euclidean Structure $\mathbf{X}$ of the scene and the projective structure $\mathbf{X}_p$ obtained by the algorithm described above.

Similarly to the calibrated case, this linear reconstruction is suboptimal in the presence of noise.

In order to upgrade the projective structure $\mathbf{X}_p$ to Euclidean structure $\mathbf{X}$, we have to recover the unknown transformation $H \in \mathbb{R}^{4\times4}$, such that:

$$\mathbf{X} = H\mathbf{X}_p \tag{6.95}$$

This can be done in several steps as the remainder of this chapter outlines. The projective transformation in equation (6.91) characterizes the ambiguity relating the choice of projection matrices P and $\tilde{P}$. Even when we can exactly decompose F into parts related to rotation ARA^{-1} and translation AT and use it for reconstruction, the structure obtained in such way, is related to the original Euclidean structure by an unknown matrix A. Hence the projective transformation H upgrading projective structure $\mathbf{X}_p$ to Euclidean structure $\mathbf{X}$ has to resolve all these unknowns. For this process it is instrumental to consider following decomposition of the transformation H:

$$H = H_e H_a H_p = \begin{bmatrix} sR & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & 0 \\ \mathbf{v}^T & 1 \end{bmatrix} \tag{6.96}$$

The first part H_e depends simply on the choice of coordinate system and scale a factor, the second part H_a corresponds to so called affine upgrade, and is directly related to the knowledge of intrinsic parameters of the camera and the third transformation is so called projective upgrade H_p, which transforms points lying on a particular plane $[\mathbf{v}^T 1]\mathbf{X}_p = 0$ to the points at infinity. This basic observation sets the stage for the so called *stratified approach* to Euclidean reconstruction and self-calibration. In such setting the projective structure $\mathbf{X}_p$ is computed first, followed by computation of H_p and H_a and gradual upgrade of $\mathbf{X}_p$ to to $\mathbf{X}_a$ and x$\mathbf{X}$ in the following

way:

$$\mathbf{X} = H_a \mathbf{X}_a \quad \text{and} \quad \mathbf{X}_a = H_p \mathbf{X}_p$$

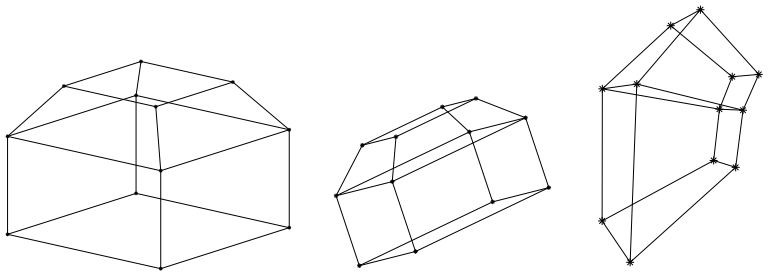

Figure 6.5. Pictorial demonstration of the stratified approach: Euclidean Structure $\mathbf{X}$, Affine Structure $\mathbf{X}_a$ and Projective structure $\mathbf{X}_p$ as described above.

There is a large variety of ways how to compute the individual upgrade transformations H_p and H_a using both intrinsic constraints or prior knowledge of scene structure and/or partial knowledge of calibration parameters. Some commonly used practical choices will be review in the next section.

6.6.2 Affine reconstruction

Imposing constraints

As we outlined in the previous section in case of general motion and absence of any other knowledge about structure and calibration matrix A, all we can compute the fundamental matrix F and projective transformation $\mathbf{X}_p$. In order to upgrade the projective structure to affine structure we first need to find the transformation H_p, such that $\mathbf{X}_a = H_p \mathbf{X}_p$

$$H_p = \begin{bmatrix} I & 0 \\ \mathbf{v}^T & 1 \end{bmatrix} \tag{6.97}$$

Note that the H_p has a special property that all points which satisfy the constraint $[\mathbf{v}, 1]^T \mathbf{X}_p = 0$, will be mapped to the points $[X, Y, Z, 0]^T$, with the last coordinate equal to 0. These points have a special geometric meaning: namely they corresponds to the intersections of parallel lines in 3D space; the so called *points at infinity*. While the points at infinity, do not correspond to any physical points in Euclidean space $\mathbb{E}_3$, due to the effects of perspective projection, the images of intersections of parallel lines correspond to perfectly valid points in the image plane, namely the *vanishing points*. Given the projective coordinates of at least three vanishing points

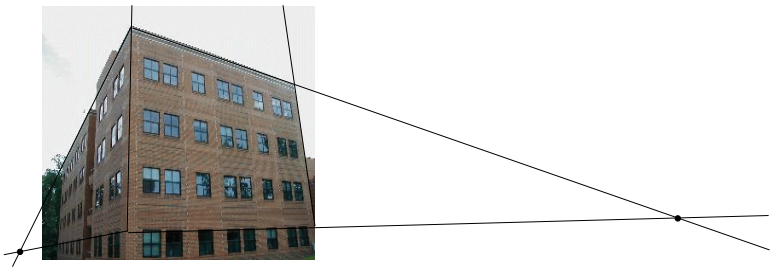

Figure 6.6. An example of vanishing lines and associated vanishing directions.

$\mathbf{X}_p^1, \mathbf{X}_p^2, \mathbf{X}_p^3$ obtained in the projective reconstruction step, we can compute the unknown vector $\mathbf{v}$, which corresponds to the plane in projective space, where all vanishing points lie.

$$[\mathbf{v}, 1]^T \mathbf{X}_p^i = 0$$

Once $\mathbf{v}$ has been determined the projective upgrade H_p is defined and the projective structure $\mathbf{X}_p$ can be upgraded to $\mathbf{X}_a$.

$$\mathbf{X}_a = \begin{bmatrix} I & 0 \\ \mathbf{v}^T & 1 \end{bmatrix} \mathbf{X}_p \tag{6.98}$$

Hence the projective upgrade can be computed when images of at least three vanishing points are available.

The situation is not completely lost, in case the above structure constraints are not available. Pollefeys in [**?**] proposed a technique, where additional so called modulus constraints on $\mathbf{v}$ are imposed, which capture the fact that $\mathbf{v}$, needs to be chosen in a way that $A + T'\mathbf{v}^T$ is a rotation matrix. This approach requires a solution to a set of fourth order polynomial equations and multiple views (four) are necessary, for the solution to be well conditioned. In Hartley [**?**] previously introduced chirality constraints are used to determine possible range of values for $\mathbf{v}$, where the optimum is achieved using linear programming approach.

Direct affine reconstruction

There are several ways how to achieve *affine reconstruction* $\mathbf{X}_a$, which will bring is in the context of stratified approach one step closer to the Euclidean reconstruction. One commonly used assumption in order to obtain affine reconstruction is to assume scaled orthographic camera projection model [**?**, TK92].

When general perspective projection model is used and the motion is pure translation the affine structure can be recovered directly.

Pure translation

From two views related by pure translation only, we have:

$$\begin{aligned} \lambda_2 A\mathbf{x}_2 &= \lambda_1 A\mathbf{x}_1 + AT \\ \lambda_2 \mathbf{x}_2' &= \lambda_1 \mathbf{x}_1' + T' \end{aligned} \tag{6.99}$$

$T' = AT$ can be computed directly from the epipolar constraint:

$$\mathbf{x}'^T_2 \widehat{T}' \mathbf{x}_1' = 0$$

Once T' has been computed, note that the unknown scales are now linear functions of known quantities and can be computed directly from the equation (6.100) as

$$\lambda_1 = \frac{(\widehat{\mathbf{x}}_2' \mathbf{x}_1')^T \widehat{\mathbf{x}}_2' T}{\|\widehat{\mathbf{x}}_2' \mathbf{x}_1'\|^2}$$

One can easily verified that the structure $\mathbf{X}_a$ obtained in this manner is indeed related to the Euclidean structure $\mathbf{X}$ by some unknown general affine transformation:

$$H_a = \begin{bmatrix} sR & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \tilde{A} & T \\ 0 & 1 \end{bmatrix} \tag{6.100}$$

6.6.3 Euclidean reconstruction

The upgrade from the affine $\mathbf{X}_a$ to Euclidean structure $\mathbf{X}$ requires knowledge of the metric S of the uncalibrated space. The presence of affine structure however brings us to the familiar territory. Suppose that we have two views of an affine structure available, which has been obtained by one of the methods in the previous section:

$$\begin{aligned} \lambda_1 \mathbf{x}_1' &= M_1 \mathbf{X}_a + \mathbf{m}_1 \\ \lambda_2 \mathbf{x}_2' &= M_2 \mathbf{X}_a + \mathbf{m}_2 \end{aligned} \tag{6.101}$$

where the projection matrices $P_1 = [M_1, \mathbf{m}_1]$ and $P_2 = [M_2, \mathbf{m}_2]$ can be computed using the same techniques as in the case of calibration with known object. Combining the above two equations, it can be shown that

$$\lambda_2 \mathbf{x}_2' = M\lambda_1 \mathbf{x}_1' + \mathbf{m} \tag{6.102}$$

where $M = M_2 M_1^{-1}$. Comparing the above equations with the familiar rigid body equations in the uncalibrated world:

$$\lambda_2 \mathbf{x}_2' = \lambda_1 ARA^{-1} \mathbf{x}_1' + T' \tag{6.103}$$

we can conclude that $M = ARA^{-1}$. This is exactly the case which we studied thoroughly in the pure rotation case for self-calibration, where we fully characterized the space of possible solutions for S given, two views related by $M = ARA^{-1}$. Both the results as well as the algorithms for this approach, reduce to a pure rotation case.

More details

0. More views are needed 1. Compute S 2. Cholesky 3. Get A.

Structure constraints

Natural additional constraints we can employ in order to solve for the unknown metric S are constraints on the structure; in particular distances between points and angles between lines. Angles are distances are directly related to notion of the inner product in the uncalibrated space and hence when known, can be used to solve for S. Commonly encountered constraint in man-made environments are orthogonality constraints between sets of lines in 3D. These are reflected in the image plane in terms of angles between vanishing directions, for example $\mathbf{v}_1, \mathbf{v}_2$ in the image plane, which correspond to a set of orthogonal lines in 3D have to satisfy the following constraint:

$$\mathbf{v}_1^T S^{-1} \mathbf{v}_2 = 0 \tag{6.104}$$

since the inner product for orthogonal lines is 0. Note since S^{-1} is a symmetric matrix with six degrees of freedom, we need at least five pairs of perpendicular lines to solve for S^{-1}.

More details

Point of difference between S, S^{-1} etc, how to get A.

Constraints on A

Computation of S^{-1} and consequently A can be simplified when some of the intrinsic parameters are known. The most commonly used assumptions are zero skew and known aspect ratios. In such case one can demonstrate that S^{-1} can be parameterized by fewer then six parameters and hence smaller number of constraints is necessary.

6.7 Summary

6.8 Exercises

1. **Lyapunov maps**
   Let $\{\lambda\}_{i=1}^n$ be the (distinct) eigenvalues of $A \in \mathbb{C}^{n\times n}$. Find *all* the n^2 eigenvalues and eigenvectors of the linear maps:

   (a) $L_1 : X \in \mathbb{C}^{n\times n} \mapsto A^T P - PA \in \mathbb{C}^{n\times n}$.
   (b) $L_2 : P \in \mathbb{C}^{n\times n} \mapsto A^T P A - P \in \mathbb{C}^{n\times n}$.

   (Hint: construct the eigenvectors of L_1 and L_2 from left eigenvectors $u_i, i = 1, \ldots, n$ and right eigenvectors $v_j, j = 1, \ldots, n$ of A.)

2. **A few identities associated to a fundamental matrix**
   Given a fundamental matrix $F = \widehat{T'}ARA^{-1}$ with $A \in SL(3)$, $T' = AT \in \mathbb{R}^3$, $R \in SO(3)$, besides the well-known Kruppa equation, we have the following

   (a) $F^T\widehat{T'}F = \widehat{AR^TT}$.

   (b) $ARA^{-1} - \widehat{T'}^T F = T'v^T$ for some $v \in \mathbb{R}^3$.

   Note that the first identity showed in the proof of Theorem 3.3 (in handout 7); the second identity in fact can be used to derive the so-called *modulus constraints* on calibration matrix A (See Hartley and Zissermann, page 458). (Hint: (a) is easy; for (b), the key step is to show that the matrix on the left hand side is of rank 1, for which you may need to use the fact that $\widehat{T'}^T\widehat{T'}$ is a projection matrix.)

3. **Invariant conic under a rotation**
   For the equation

$$S - RSR^T = 0, \quad R \in SO(3),\ S^T = S \in \mathbb{R}^{3\times 3},$$

   prove Lemma 7.3.1 for the case that rotation angle of R is between 0 and π. What are the two typical solutions for S? (Hint: You may need eigenvectors of the rotation matrix R. Properties of Lyapunov map from the previous homework should give you enough ideas how to find all possible S from eigenvectors (constructed from those of R) of the Lyapunov map $L : S \mapsto S - RSR^T$.) Since a symmetric real matrix S is usually used to represent a conic $x^TSx = 1$, S which satisfies the above equation obviously gives a conic that is invariant under the rotation R.

Part III

Geometry of multiple views

Chapter 7

Introduction to multiple view reconstruction

In previous chapters we have studied the geometry of points in space as seen from two views. There, the epipolar constraint plays a crucial role in that it captures the relationship between corresponding points and camera configuration without reference to the position of such points in space. Such relationship is typically referred to as *intrinsic*. The epipolar constraint gives a complete answer to the question of what the intrinsic relationship between corresponding points in two views are. It is then natural to ask this question in the context of an arbitrary number of views: *what is the relationship between corresponding points in m views?* Not only is this question crucial to the understanding of the geometry of points in space as seen from a number of views, but – like in the two-view case – the constraints will be used to derive algorithms for reconstructing camera configuration and, ultimately, the 3-D position of points.

The search for the m-view analogue of the epipolar constraint has been an active research area for almost two decades. It was realized early on [**?**, SA90] that the relationship between multiple views of the same point or line can be characterized by *multi-linear* constraints. Like the epipolar constraint is *bilinear*, i.e. it is linear in the coordinates of the point in one view having fixed the coordinates in the other, one can show that there exist m-linear constraints between corresponding points in m images. Consequently, the study of multiple view geometry has involved multi-dimensional linear operators, or tensors, with as many indices as views. There is now a substantial literature on the properties of so-called *tri-focal* and *quadri-focal tensors* that involves the use of sophisticated algebraic geometric tools, such as the Grassmann-Caley algebra [**?**]. In the interest of simplicity and com-

pleteness, this book takes an approach that is substantially different from the current literature, in that it only involves linear algebra and matrix manipulations. We will show that all the constraints between corresponding points in m images can be written as constraints on the rank of certain matrices. In order to put this approach into a historic context, in the exercises at the end of this chapter we will explore the relation between these rank conditions and multi-linear tensors.

Roughly speaking, the extant multi-linear analysis is adequate if point or line features are to be considered individually. In the next four chapters, we will however see clearly that if also incidence relations among multiple point and line features are considered, the multi-linear analysis is no longer sufficient. Nonetheless, the rank condition to be introduced will provide us a simple and unifying tool that gives a complete characterization of *all* intrinsic constraints among multiple images. More specifically, we will formally establish the following facts:

1. For perspective projection from $\mathbb{R}^3$ to $\mathbb{R}^2$, intrinsic algebraic and geometric relationships among multiple images of (multiple) points and lines (with incidence relations among themselves) can always be reduced to two categories: multi-linear relations among pairwise or triple-wise images; nonlinear relations among triple-wise or quadruple-wise images.
2. For perspective projection from $\mathbb{R}^n$ to $\mathbb{R}^k$ $(k < n)$, intrinsic algebraic and geometric relationships among multiple images of hyperplanes (with incidence relations among themselves) can be uniformly described by certain rank conditions, from which all kinds of algebraic constraints can be easily instantiated.

In addition to the theoretical development, we will demonstrate how to provide global geometric analysis for multiple images of points, lines, planes, and incidence relations among them based on the rank condition. Conceptual algorithms will be given for tasks such as feature matching, image transfer, and structure from motion. These algorithms demonstrate the possibility of accomplishing these tasks by simultaneously using all images of all features and all incidence relations. We leave more practical algorithmic issues such as optimality and real-time constraint to chapters in Part IV.

7.1 Basic notions: pre-image and co-image of point and line

To set the stage, we recall the notation from Chapter 3: consider a generic point $p \in \mathbb{E}^3$ in the space. The homogeneous coordinates of p relative to a fixed world coordinate frame are denoted as $\mathbf{X} \doteq [X, Y, Z, 1]^T \in \mathbb{R}^4$.

Then the (perspective) image $\mathbf{x}(t) \doteq [x(t), y(t), z(t)]^T \in \mathbb{R}^3$ of p, taken by a moving camera at time t satisfies the relationship

$$\lambda(t)\mathbf{x}(t) = A(t)Pg(t)\mathbf{X}, \tag{7.1}$$

where $\lambda(t) \in \mathbb{R}_+$ is the (unknown) depth of the point p relative to the camera frame, $A(t) \in SL(3)$ is the camera calibration matrix (at time t), $P = [I, 0] \in \mathbb{R}^{3\times 4}$ is the standard (perspective) projection matrix and $g(t) \in SE(3)$ is the coordinate transformation from the world frame to the camera frame at time t. In equation (7.1), $\mathbf{x}, \mathbf{X}$ and g are all in *homogeneous representation*. Suppose the transformation g is specified by its rotation $R \in SO(3)$ and translation $T \in \mathbb{R}^3$, then the homogeneous representation of g is simply

$$g = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \quad \in \mathbb{R}^{4\times 4}. \tag{7.2}$$

Notice that equation (7.1) is also equivalent to:

$$\lambda(t)\mathbf{x}(t) = [A(t)R(t)\ A(t)T(t)]\mathbf{X}. \tag{7.3}$$

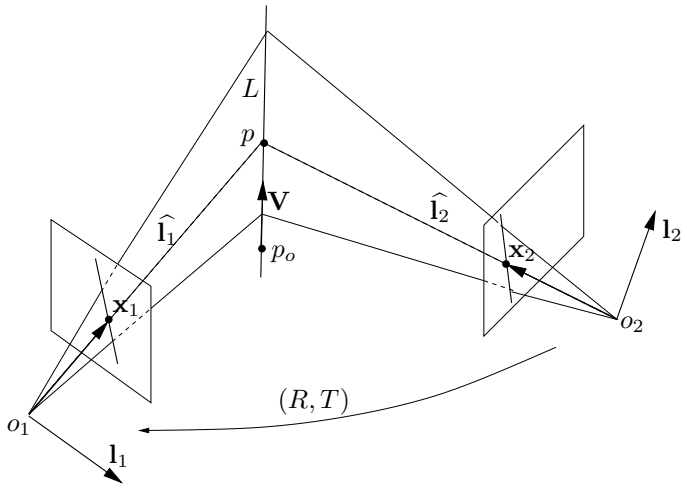

Figure 7.1. Images of a point p on a line L. Planes extended from the images $\widehat{\mathbf{l}_1}, \widehat{\mathbf{l}_2}$ should intersect at the line L in 3-D. Lines extended from the image points $\mathbf{x}_1, \mathbf{x}_2$ intersect at the point p. $\mathbf{l}_1, \mathbf{l}_2$ are the two co-images of the line.

Now suppose a point p lying on a straight line $L \subset \mathbb{E}^3$, as shown in Figure 10.1. The line L can be defined by a collection of points in $\mathbb{E}^3$ that can be described (in homogeneous coordinates) as:

$$L \doteq \{\mathbf{X} \mid \mathbf{X} = \mathbf{X}_o + \mu\mathbf{V},\ \mu \in \mathbb{R}\} \quad \subset \mathbb{R}^4, \tag{7.4}$$

where $\mathbf{X}_o = [X_o, Y_o, Z_o, 1]^T \in \mathbb{R}^4$ are coordinates of a base point p_o on this line and $\mathbf{V} = [V_1, V_2, V_3, 0]^T \in \mathbb{R}^4$ is a non-zero vector indicating the direction of the line. Then the *image* of the line L at time t is simply the collection of images $\mathbf{x}(t)$ of all points $p \in L$. It is clear that all such $\mathbf{x}(t)$'s span a plane in $\mathbb{R}^3$, as shown in Figure 10.1. For future development, we need a formal definition of the notion "pre-image":

Definition 7.1 (Pre-image). *The pre-image of a point or a line in 3-D is defined to be the subspace (in the camera coordinate frame) spanned by the homogeneous coordinates of the image point or points of the line in the image plane.*

So in the case of a point, its pre-image is a one-dimensional subspace, i.e. a line defined by the vector $\mathbf{x}$. Any non-zero vector on this line is a valid representative for this line. It uniquely determines the image of the point in the image plane: the intersection of the line with the image plane, as shown in Figure 10.1. In the case of a line, the pre-image is a plane as shown in Figure 10.1. Any two linearly independent vectors in this plane can be used to represent this plane. The plane uniquely determines the image of the line in the image plane: the intersection of the plane with the image plane. Notice that, any two points on the pre-image of a point give the same image in the image plane, so do any two lines on the pre-image of a line. So the pre-image is really the largest set of 3-D points or lines which have the same image in the image plane.

Since a pre-image is a subspace, we can also specify it by its maximum orthogonal supplementary subspace. For instance, it is usually more convenient to specify a plane by its normal vector. This leads to the notion of "co-image":

Definition 7.2 (Co-image). *The co-image of a point or a line in 3-D is defined to be the maximum orthogonal supplementary subspace othogonal to its pre-image (in the camera coordinate frame).*

Since the pre-image of a line L is a plane, we can denote its co-image (at time t) by the plane's normal vector $\mathbf{l}(t) = [a(t), b(t), c(t)]^T \in \mathbb{R}^3$. If $\mathbf{x}$ is the image of a point p on this line, then $\mathbf{l}$ satisfies the following equation:

$$\mathbf{l}(t)^T \mathbf{x}(t) = \mathbf{l}(t)^T A(t) P g(t) \mathbf{X} = 0. \tag{7.5}$$

Recall that we use $\widehat{u} \in \mathbb{R}^{3\times 3}$ to denote the skew-symmetric matrix associated to a vector $u \in \mathbb{R}^3$ whose column (or row) vectors span the plane orthogonal to the vector u. Then the column (or row) vectors of the matrix $\widehat{\mathbf{l}}$ span the pre-image of the line L, i.e. they span the plane which is *orthogonal* to $\mathbf{l}$.[1] This is illustrated in Figure 10.1. Similarly, if $\mathbf{x}$ is the image of

[1] In fact, there is some redundancy using $\widehat{\mathbf{l}}$ to describe the plane: the three column (or row) vectors in $\widehat{\mathbf{l}}$ are not linearly independent. They only span a two-dimensional space. However, we here use it anyway to simplify the notation.

a point p, its co-image (the plane orthogonal to $\mathbf{x}$) is given by the column (or row) vectors of the matrix $\widehat{\mathbf{x}}$. So from now on, we will use the following notations to represent the (pre-)image or co-image of a point or a line:

$$\begin{array}{llll} \text{image of a point:} & \mathbf{x} \in \mathbb{R}^3, & \text{co-image of a point:} & \widehat{\mathbf{x}} \in \mathbb{R}^{3\times 3}, \\ \text{image of a line:} & \widehat{\mathbf{l}} \in \mathbb{R}^{3\times 3}, & \text{co-image of a line:} & \mathbf{l} \in \mathbb{R}^3. \end{array} \tag{7.6}$$

Notice that we always have $\widehat{\mathbf{x}} \cdot \mathbf{x} = 0$ and $\widehat{\mathbf{l}} \cdot \mathbf{l} = 0$. Since (pre-)image and co-image are equivalent representation of the same geometric entity, sometimes for simplicity (and by abuse of language) we may simply refer to either one as "image" if its meaning is clear from the context.

In a realistic situation, we usually obtain "sampled" images of $\mathbf{x}(t)$ or $\mathbf{l}(t)$ at some time instances: $t_1, t_2, \ldots, t_m$. For simplicity we denote:

$$\lambda_i = \lambda(t_i), \quad \mathbf{x}_i = \mathbf{x}(t_i), \quad \mathbf{l}_i = \mathbf{l}(t_i), \quad \Pi_i = A(t_i)Pg(t_i). \tag{7.7}$$

We will call the matrix Π_i the *projection matrix* relative to the i^{th} camera frame. The matrix Π_i is then a 3×4 matrix which relates the i^{th} image of the point p to its world coordinates $\mathbf{X}$ by:

$$\lambda_i \mathbf{x}_i = \Pi_i \mathbf{X} \tag{7.8}$$

and the i^{th} co-image of the line L to its world coordinates $(\mathbf{X}_o, \mathbf{V})$ by:

$$\mathbf{l}_i^T \Pi_i \mathbf{X}_o = \mathbf{l}_i^T \Pi_i \mathbf{V} = 0, \tag{7.9}$$

for $i = 1, \ldots, m$. If the point is actually lying on the line, we further have a relationship between the image of the point and the co-image of the line:

$$\mathbf{l}_i^T \mathbf{x}_i = 0, \tag{7.10}$$

for $i = 1, \ldots, m$. For convenience, we will use $R_i \in \mathbb{R}^{3\times 3}$ to denote the first three columns of the projection matrix Π_i, and $T_i \in \mathbb{R}^3$ for the last column. Note that (R_i, T_i) are not necessarily the actual camera rotation and translation unless the camera is fully calibrated, i.e. $A(t_i) \equiv I$. In any case, the matrix R_i is always of full rank 3.

7.2 Preliminaries

We first observe that in above equations the unknowns, λ_i's, $\mathbf{X}$, $\mathbf{X}_o$ and $\mathbf{V}$, which encode the information about location of the point p or the line L are not intrinsically available from the images. By eliminating these unknowns from the equations we obtain the remaining intrinsic relationships between $\mathbf{x}, \mathbf{l}$ and Π only, i.e. between the image measurements and the camera configuration. Of course there are many different, but algebraically equivalent ways for elimination of these unknowns. This has in fact resulted in different kinds (or forms) of multilinear (or multifocal) constraints that exist in the computer vision literature. Our goal here should be a more *systematic*

way of eliminating all the above unknowns that results in a *complete* set of conditions and a clear geometric characterization of all constraints.

To this end, we can re-write the above equations in matrix form as

$$\begin{bmatrix} \mathbf{x}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{x}_m \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} \Pi_1 \\ \Pi_2 \\ \vdots \\ \Pi_m \end{bmatrix} \mathbf{X} \tag{7.11}$$

which, after defining

$$\mathcal{I} \doteq \begin{bmatrix} \mathbf{x}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{x}_m \end{bmatrix}, \quad \vec{\lambda} \doteq \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix}, \quad \Pi \doteq \begin{bmatrix} \Pi_1 \\ \Pi_2 \\ \vdots \\ \Pi_m \end{bmatrix}, \tag{7.12}$$

becomes

$$\boxed{\mathcal{I}\vec{\lambda} = \Pi\mathbf{X}.} \tag{7.13}$$

which is just a matrix version of (7.8). For obvious reasons, we call $\vec{\lambda} \in \mathbb{R}^{3m}$ the *depth scale vector*, and $\Pi \in \mathbb{R}^{3m\times 4}$ the *projection matrix* associated to the *image matrix* $\mathcal{I} \in \mathbb{R}^{3m\times m}$. The reader will notice that, with the exception of $\mathbf{x}_i$'s, everything else in this equation is unknown! Solving for the depth scales and the projection matrices directly from such equations is by no means straightforward. Like we did in the two-view case, therefore, we will decouple the recovery of the camera configuration matrices Π_i's from recovery of scene structure, λ_i's and $\mathbf{X}$.

In order for equation (7.13) to be satisfied, it is necessary for the columns of $\mathcal{I}$ and Π to be linearly dependent. In other words, the matrix

$$N_p \doteq [\Pi,\ \mathcal{I}] = \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & \cdots & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \Pi_m & 0 & \cdots & 0 & \mathbf{x}_m \end{bmatrix} \in \mathbb{R}^{3m\times(m+4)} \tag{7.14}$$

must have a non-trivial null-space. Hence

$$\boxed{\operatorname{rank}(N_p) \leq m + 3.} \tag{7.15}$$

In fact, from equation (7.13) it is immediate to see that the vector $u \doteq [\mathbf{X}^T, -\vec{\lambda}^T]^T \in \mathbb{R}^{m+4}$ is in the null space of the matrix N_p, i.e. $N_p u = 0$.

Similarly, for line features the following matrix:

$$N_l \doteq \begin{bmatrix} \mathbf{l}_1^T\Pi_1 \\ \mathbf{l}_2^T\Pi_2 \\ \vdots \\ \mathbf{l}_m^T\Pi_m \end{bmatrix} \in \mathbb{R}^{m\times 4} \tag{7.16}$$

must satisfy the rank condition

$$\text{rank}(N_l) \leq 2 \tag{7.17}$$

since it is clear that the vectors $\mathbf{X}_o$ and $\mathbf{V}$ are both in the null space of the matrix N_l due to (7.9). In fact, any $\mathbf{X} \in \mathbb{R}^4$ in the null space of N_l represents the homogeneous coordinates of some point lying on the line L, and vice versa.

The above rank conditions on N_p and N_l are merely the starting point. There is some difficulty in using them directly since: 1. the lower bounds on their rank are not yet clear; 2. their dimensions are high and hence the above rank conditions contain a lot of redundancy. This apparently obvious observation is the basis for the characterization of all constraints among corresponding point and line in multiple views. To put our future development in historic context, in the rest of this chapter we will concentrated on only the pairwise and triple-wise images.

7.3 Pairwise view geometry revisited

Note that the condition that the $m + 4$ columns of $\mathcal{I}$ and Π be linearly dependent does not involve the 3-D position of the points, just its images and camera configurations. Therefore, for the particular case of $m = 2$ views, one would like to make sure that this condition implies the epipolar constraint. Since $3m = m + 4 = 6$, the rank condition (7.15) is equivalent to

$$\det(N_p) = \det \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 \end{bmatrix} = 0. \tag{7.18}$$

Since the determinant is linear in the elements of the matrix, the equation above is clearly bilinear in $\mathbf{x}_1, \mathbf{x}_2$. It is an easy exercise to show that any bilinear function $\mathbf{x}_1, \mathbf{x}_2$ can be written as

$$\mathbf{x}_2^T F \mathbf{x}_1 \tag{7.19}$$

for some matrix $F \in \mathbb{R}^{3\times 3}$. In our case, the entries of F are the 4×4 minors of the matrix

$$\Pi = \begin{bmatrix} \Pi_1 \\ \Pi_2 \end{bmatrix} \in \mathbb{R}^{6\times 4}.$$

For instance, for the case of two calibrated cameras, the matrix F is exactly the essential matrix.[2] To see this, without loss of generality, let us assume

[2] In the tensorial jargon, the epipolar constraint is sometimes referred to as the *bilinear constraint*, and the essential or fundamental matrix F as the *bifocal tensor*. Mathematically, F can be interpreted as a covariant 2-tensor whose contraction with $\mathbf{x}_1$ and $\mathbf{x}_2$ is zero.

that the first camera frame coincides with the world frame and the camera motion between the two views is (R, T). Then we have

$$\Pi_1 = [I,\ 0], \quad \Pi_2 = [R,\ T] \quad \in \mathbb{R}^{3\times 4}. \tag{7.20}$$

The determinant of the matrix $N_p = [\Pi\ \mathcal{I}]$ yields

$$\begin{aligned} & \det \begin{bmatrix} I & 0 & \mathbf{x}_1 & 0 \\ R & T & 0 & \mathbf{x}_2 \end{bmatrix} = \det \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & T & R\mathbf{x}_1 & \mathbf{x}_2 \end{bmatrix} \\ = \quad & \det [T, R\mathbf{x}_1, \mathbf{x}_2] = \det [\mathbf{x}_2, T, R\mathbf{x}_1]\,. \end{aligned}$$

For the last step, we use the fact from linear algebra that $\det[v_1, v_2, v_3] = v_1^T(v_2 \times v_3)$. This yields

$$\det [\mathbf{x}_2, T, R\mathbf{x}_1] = \mathbf{x}_2^T \widehat{T} R \mathbf{x}_1.$$

Therefore, we obtain

$$\boxed{\det(N_p) = 0 \quad \Leftrightarrow \quad \mathbf{x}_2^T \widehat{T} R \mathbf{x}_1 = 0} \tag{7.21}$$

which says that, for the two-view case, the rank condition (7.15) is equivalent to the epipolar constraint. The advantage of the rank condition is that it generalizes to multiple views in a straightforward manner, unlike the epipolar constraint. A similar derivation can be followed for the uncalibrated case by just allowing $R \in \mathbb{R}^3$ to be general and *not* a rotation matrix. The resulting matrix F simply becomes the fundamental matrix.

However, for two (co-)images $\mathbf{l}_1, \mathbf{l}_2$ of a line L, the above condition (7.17) for the matrix N_l becomes

$$\operatorname{rank}(N_l) = \operatorname{rank} \begin{bmatrix} \mathbf{l}_1^T \Pi_1 \\ \mathbf{l}_2^T \Pi_2 \end{bmatrix} \leq 2. \tag{7.22}$$

But here N_l is a 2×4 matrix which automatically has a rank less and equal to 2. Hence there is essential no intrinsic constraints on two images of a line. This is not so surprising since for arbitrarily given two vectors $\mathbf{l}_1, \mathbf{l}_2 \in \mathbb{R}^3$, they represent two planes relative to the two camera frames respectively. These two planes will always intersect at some line in 3-D, and $\mathbf{l}_1, \mathbf{l}_2$ can be interpreted as two images of this line. A natural question hence arises: are there any non-trivial constraints for images of a line in more than 2 views? The answer will be yes which to some extent legitimizes the study of triple-wise images. Another advantage from having more than 2 views is to avoid some obvious degeneracy inherent in the pairwise view geometry. As demonstrated later in Figure 8.2, there are cases when all image points satisfy pairwise epipolar constraint but they do not correspond to a unique 3-D point. But as we will establish rigorously in the next chapter, constraints among triple-wise views can eliminate such degenerate cases very effectively.

In the next section, for the purpose of example, we show how to derive all multilinear constraints between corresponding images in $m = 3$ views.

We however do not pursue this direction any further for $m \geq 4$ views, since we will see in Chapters 8, 9, and 10 that there are no more multilinear constraints among more than 4 views! However, a much more subtle picture of the constraints among multiple images of points and lines cannot be revealed untill we study those more general cases.

7.4 Triple-wise view geometry

For three views of a point, we have $3m = 9 > m + 4 = 7$. The matrix

$$N_p = \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 & 0 \\ \Pi_3 & 0 & 0 & \mathbf{x}_3 \end{bmatrix} \in \mathbb{R}^{9\times 7} \tag{7.23}$$

then has more rows than columns. Therefore, the rank condition (7.15) implies that all 7×7 sub-matrices are singular. As we explore in the exercises, the only irreducible minor of N_p must contain at least two rows from each image. For example, if we choose all three rows of the first image and the $\{1, 2\}^{th}$ rows from the second and the $\{1, 2\}^{th}$ rows from the third, we obtain a 7×7 sub-matrix of N_p as

$$N_{033} = \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & 0 \\ \Pi_2^1 & 0 & x_{21} & 0 \\ \Pi_2^2 & 0 & x_{22} & 0 \\ \Pi_3^1 & 0 & 0 & x_{31} \\ \Pi_3^2 & 0 & 0 & x_{32} \end{bmatrix} \in \mathbb{R}^{7\times 7} \tag{7.24}$$

where the subscript rst of N_{rst} indicates that the r^{th}, s^{th} and t^{th} rows of images $1, 2$ and 3 respectively are omitted from the original matrix N_p when N_{rst} is formed. The rank condition (7.15) implies that

$$\det(N_{033}) = 0 \tag{7.25}$$

and, as in the two-view case, the determinant is linear in each of the $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$. The above equation, written in tensor form, is called the *trifocal tensor*; we explore its properties in the exercises. Here, instead, we derive its expression directly from the rank condition.

Let us demonstrate this by choosing the first camera frame to be the reference frame. That gives the three projection matrices $\Pi_i, i = 1, 2, 3$

$$\Pi_1 = [I, 0], \quad \Pi_2 = [R_2, T_2], \quad \Pi_3 = [R_3, T_3] \quad \in \mathbb{R}^{3\times 4}, \tag{7.26}$$

where, in the case of uncalibrated cameras, $R_i \in \mathbb{R}^{3\times 3}, i = 2, 3$ may not be rotation matrices. The matrix N_p is then

$$N_p = \begin{bmatrix} I & 0 & \mathbf{x}_1 & 0 & 0 \\ R_2 & T_2 & 0 & \mathbf{x}_2 & 0 \\ R_3 & T_3 & 0 & 0 & \mathbf{x}_3 \end{bmatrix} \in \mathbb{R}^{9\times 7}. \tag{7.27}$$

This matrix should satisfy the rank condition. After a column manipulation (which eliminates $\mathbf{x}_1$ from the first row), it is easy to see that N_p has the same rank as the matrix

$$N_p' = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ R_2 & T_2 & R_2\mathbf{x}_1 & \mathbf{x}_2 & 0 \\ R_3 & T_3 & R_3\mathbf{x}_1 & 0 & \mathbf{x}_3 \end{bmatrix} \in \mathbb{R}^{9\times 7}. \tag{7.28}$$

For this matrix to drop rank, its sub-matrix

$$N_p'' = \begin{bmatrix} T_2 & R_2\mathbf{x}_1 & \mathbf{x}_2 & 0 \\ T_3 & R_3\mathbf{x}_1 & 0 & \mathbf{x}_3 \end{bmatrix} \in \mathbb{R}^{6\times 4} \tag{7.29}$$

must drop rank. Multiplying the following matrix

$$D = \begin{bmatrix} \mathbf{x}_2^T & 0 \\ \widehat{\mathbf{x}_2} & 0 \\ 0 & \mathbf{x}_3^T \\ 0 & \widehat{\mathbf{x}_3} \end{bmatrix} \in \mathbb{R}^{8\times 6} \tag{7.30}$$

by N_p'' yields

$$DN_p'' = \begin{bmatrix} \mathbf{x}_2^T T_2 & \mathbf{x}_2^T R_2\mathbf{x}_1 & \mathbf{x}_2^T\mathbf{x}_2 & 0 \\ \widehat{\mathbf{x}_2}T_2 & \widehat{\mathbf{x}_2}R_2\mathbf{x}_1 & 0 & 0 \\ \mathbf{x}_3^T T_3 & \mathbf{x}_3^T R_3\mathbf{x}_1 & 0 & \mathbf{x}_3^T\mathbf{x}_3 \\ \widehat{\mathbf{x}_3}T_3 & \widehat{\mathbf{x}_3}R_3\mathbf{x}_1 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{8\times 4}. \tag{7.31}$$

Since D is of full rank 6, we have

$$\operatorname{rank}(N_p'') = \operatorname{rank}(DN_p'').$$

Hence the original matrix N_p is rank deficient if and only if the following sub-matrix of DN_p''

$$M_p = \begin{bmatrix} \widehat{\mathbf{x}_2}T_2 & \widehat{\mathbf{x}_2}R_2\mathbf{x}_1 \\ \widehat{\mathbf{x}_3}T_3 & \widehat{\mathbf{x}_3}R_3\mathbf{x}_1 \end{bmatrix} \in \mathbb{R}^{6\times 2} \tag{7.32}$$

is rank deficient. In order to conclude, we need the following fact, whose proof is left to the reader as an exercise:

Lemma 7.1. *Given any* non-zero *vectors* $a_i, b_i \in \mathbb{R}^3, i = 1,\dots n$, *the following matrix*

$$\begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_n & b_n \end{bmatrix} \in \mathbb{R}^{3n\times 2} \tag{7.33}$$

is rank deficient if and only if $a_i b_j^T - b_i a_j^T = 0$ *for all* $i, j = 1,\dots,n$.

Apply this to the matrix M in (7.32), we have

$$\boxed{\operatorname{rank}(N_p) < 7 \quad \Rightarrow \quad \widehat{\mathbf{x}_2}(T_2\mathbf{x}_1^T R_3^T - R_2\mathbf{x}_1 T_3^T)\widehat{\mathbf{x}_3} = 0} \tag{7.34}$$

Note that this is a matrix equation and it gives a total of $3 \times 3 = 9$ scalar trilinear equations in $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$. Equation (**??**) is the analogous of the epipolar constraint for three views.

Comment 7.1. *The derivation of the trilinear constraints shows that (7.34) implies bilinear constraints, since the two vectors* $\widehat{\mathbf{x}_2}T_2, \widehat{\mathbf{x}_2}R_2\mathbf{x}_1$ *being linearly dependent implies* $\mathbf{x}_2^T\widehat{T_2}R_2\mathbf{x}_1 = 0$ *(we leave this as an exercise to the reader). The trilinear constraint, however, does not imply the bilinear ones when, for example,* $\widehat{\mathbf{x}_3}T_3 = \widehat{\mathbf{x}_3}R_3\mathbf{x}_1 = 0$*. That corresponds to the case when the pre-image* p *lies on the line through optical centers* o_1, o_3*. In that case, the epipolar constraint between the first and second images is not implied by the trilinear constraint above but still implied by the condition* $\text{rank}(M_p) \leq 1$*. That pairwise bilinear constraints algebraically imply all the trilinear constraints is also true (except for some rare degenerate configurations), although much harder to prove (see Chapter 8).*

However, for three (co-)images $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ of a line L, the rank condition (7.17) for the matrix N_l becomes

$$\text{rank}(N_l) = \text{rank}\begin{bmatrix}\mathbf{l}_1^T\Pi_1\\ \mathbf{l}_2^T\Pi_2\\ \mathbf{l}_3^T\Pi_3\end{bmatrix} \leq 2. \tag{7.35}$$

Here N_l is a 3×4 matrix and the above rank condition is no longer void and it indeed imposes certain non-trivial constraints on the quantities involved. In the case we choose the first view to be the reference, i.e. $\Pi_1 = [I, 0]$, we have

$$\begin{bmatrix}\mathbf{l}_1^T\Pi_1\\ \mathbf{l}_2^T\Pi_2\\ \mathbf{l}_3^T\Pi_3\end{bmatrix} = \begin{bmatrix}\mathbf{l}_1^T & 0\\ \mathbf{l}_2^TR_2 & \mathbf{l}_2^TT_2\\ \mathbf{l}_3^TR_3 & \mathbf{l}_2^TT_3\end{bmatrix}. \tag{7.36}$$

Similar to the point case, we conduct simple matrix (column) manipulation as below

$$\begin{bmatrix}\mathbf{l}_1^T & 0\\ \mathbf{l}_2^TR_2 & \mathbf{l}_2^TT_2\\ \mathbf{l}_3^TR_3 & \mathbf{l}_3^TT_3\end{bmatrix}\begin{bmatrix}\mathbf{l}_1 & \widehat{\mathbf{l}_1} & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}\mathbf{l}_1^T\mathbf{l}_1 & 0 & 0\\ \mathbf{l}_2^TR_2\mathbf{l}_1 & \mathbf{l}_2^TR_2\widehat{\mathbf{l}_1} & \mathbf{l}_2^TT_2\\ \mathbf{l}_3^TR_3\mathbf{l}_1 & \mathbf{l}_3^TR_3\widehat{\mathbf{l}_1} & \mathbf{l}_3^TT_3\end{bmatrix}. \tag{7.37}$$

Therefore, $\text{rank}(N_l) \leq 2$ is equivalent to the following matrix

$$M_l = \begin{bmatrix}\mathbf{l}_2^TR_2\widehat{\mathbf{l}_1} & \mathbf{l}_2^TT_2\\ \mathbf{l}_3^TR_3\widehat{\mathbf{l}_1} & \mathbf{l}_3^TT_3\end{bmatrix} \quad \in \mathbb{R}^{2\times 4} \tag{7.38}$$

has a rank less and equal to 1. That is the two row vectors of M_l are linearly dependent. In case $\mathbf{l}_2^TT_2, \mathbf{l}_3^TT_3$ are non-zero, this linear dependency can be written as

$$\boxed{\mathbf{l}_3^T(T_3\mathbf{l}_2^TR_2 - R_3\mathbf{l}_2^TT_2)\widehat{\mathbf{l}_1} = 0.} \tag{7.39}$$

This gives a total of $1 \times 3 = 3$ scalar trilinear equations in $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$.

Although the two types of trilinear constraints (7.34) and (7.39) are expressed in terms of point and line respectively, they in fact describe the same type of relationship among three images of a point on a line. It is easy to see that columns (or rows) of $\widehat{\mathbf{x}}$ are co-images of lines passing through the point, and columns (or rows) of $\widehat{\mathbf{l}}$ are images of points lying on the line (see Exercise 1). Hence, each scalar equation from (7.34) and (7.39) simply falls into the following type of equations:

$$\boxed{\mathbf{l}_2^T (T_2 \mathbf{x}_1^T R_3^T - R_2 \mathbf{x}_1 T_3^T) \mathbf{l}_3 = 0} \tag{7.40}$$

where $\mathbf{l}_2, \mathbf{l}_3$ are respectively co-images of two lines that pass through the same point whose image in the first view is $\mathbf{x}_1$. Here, $\mathbf{l}_2$ and $\mathbf{l}_3$ do not have to correspond to the coimages of the same line in 3-D. This is illustrated in Figure 7.2.

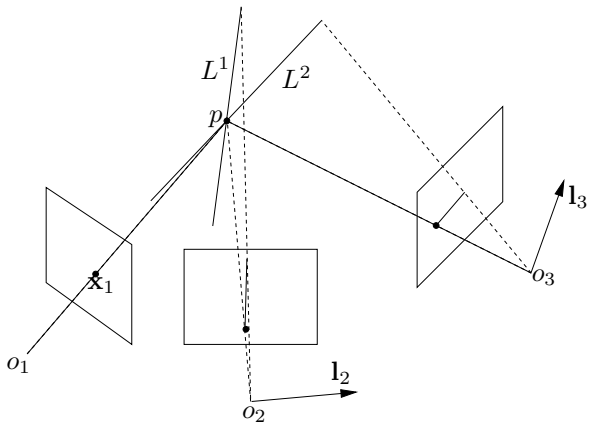

Figure 7.2. Images of two lines L^1, L^2 intersecting at a point p. Planes extended from the image lines might not intersect at the same line in 3-D. But $\mathbf{x}_1, \mathbf{l}_2, \mathbf{l}_3$ satisfy the trilinear relationship.

The above trilinear relation (7.40) plays a similar role for triple-wise view geometry as the fundamental matrix for pairwise view geometry. [3] Here the fundamental matrix must be replaced by the notion of *trifocal tensor*, traditionally denoted as $\mathcal{T}$. Like the fundamental matrix, the trifocal tensor depends (nonlinearly) on the motion paramters R and T's, but in a more complicated way. The above trilinear relation can then be written formally

[3] A more detailed study of the relationship between the trilinear constraints and the bilinear epipolar constraints is given in the next Chapter.

as the *interior product* of the tensor $\mathcal{T}$ with the three vectors $\mathbf{x}_1, \mathbf{l}_2, \mathbf{l}_3$:

$$\mathcal{T}(\mathbf{x}_1, \mathbf{l}_2, \mathbf{l}_3) = \sum_{i,j,k=1}^{3,3,3} \mathcal{T}(i,j,k)\mathbf{x}_1(i)\mathbf{l}_2(j)\mathbf{l}_3(k) = 0. \tag{7.41}$$

Intuitively, the tensor $\mathcal{T}$ consists of $3 \times 3 \times 3$ entries since the above trilinear relation has as many coefficients to be determined. Like the fundamental matrix which only has 7 free parameters ($\det(F) = 0$), there are in fact only 18 free parameters in $\mathcal{T}$. If one is only interested in recovering $\mathcal{T}$ linearly from triple-wise images as did we for the fundamental matrix from pairwise ones, by ignoring their internal dependency, we need at least 26 equations of the above type (7.41) to recover all the coefficients $\mathcal{T}(i,j,k)$'s up to a scale. We hence needs at least 26 the triplets $(\mathbf{x}_1, \mathbf{l}_2, \mathbf{l}_3)$ with correspondence specified by Figure 7.2. We leave as exercise at least how many point or line correspondences across three views are needed in order to have 26 linearly independent equations for solving $\mathcal{T}$.

7.5 Summary

7.6 Exercises

1. **Image and co-image of points and lines**
   Suppose p^1, p^2 are two points on the line L; and L^1, L^2 are two lines intersecting at the point p. Let $\mathbf{x}, \mathbf{x}^1, \mathbf{x}^2$ be the images of the points p, p^1, p^2 respectively and $\mathbf{l}, \mathbf{l}^1, \mathbf{l}^2$ be the co-images of the lines L, L^1, L^2 respectively.
   (a) Show that for some $\alpha, \beta \in \mathbb{R}$:
   $$\mathbf{l} = \alpha\widehat{\mathbf{x}^1}\mathbf{x}^2, \quad \mathbf{x} = \beta\widehat{\mathbf{l}^1}\mathbf{l}^2.$$
   (b) Show that for some $r, s, u, v \in \mathbb{R}^3$:
   $$\mathbf{l}^1 = \widehat{\mathbf{x}}u, \quad \mathbf{l}^2 = \widehat{\mathbf{x}}v, \quad \mathbf{x}^1 = \widehat{\mathbf{l}}r, \quad \mathbf{x}^2 = \widehat{\mathbf{l}}s.$$
   (c) Please draw a picture and convince yourself about the above relationships.

2. **Linear estimating of the trifocal tensor**

   (a) Show that the corresponding three images $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ of a point p in 3-D give rise to 4 linearly independent constraints of the type (7.41).
   (b) Show that the corresponding three co-images $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ of a line L in 3-D give rise to 2 linearly independent constraints of the type (7.41).

(c) Conclude that in general one needs n points and m lines across three views with $4n + 2m \geq 26$ to recover the trifocal tensor $\mathcal{T}$ up to a scale.

3. **Multi-linear function**
A function $f(\cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is called *bilinear* if $f(x, y)$ is linear in $x \in \mathbb{R}^n$ if $y \in \mathbb{R}^n$ is fixed, and vice versa. That is,

$$\begin{aligned} f(\alpha x_1 + \beta x_2, y) &= \alpha f(x_1, y) + \beta f(x_2, y), \\ f(x, \alpha y_1 + \beta y_2) &= \alpha f(x, y_1) + \beta f(x, y_2), \end{aligned}$$

for any $x, x_1, x_2, y, y_1, y_2 \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$. Show that any bilinear function $f(x, y)$ can be written as

$$f(x, y) = x^T M y$$

for some matrix $M \in \mathbb{R}^{n \times n}$. Notice that in the epipolar constraint equation

$$\mathbf{x}_2^T F \mathbf{x}_1 = 0,$$

the left hand side is a bilinear form in $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$. Hence epipolar constraint is also sometimes referred as *bilinear constraint*. Similarly we can define *trilinear* or *quadrilinear* functions. In the most general case, we can define a multi-linear function $f(x_1, \ldots, x_i, \ldots, x_m)$ which is linear in each $x_i \in \mathbb{R}^n$ if all other x_j's with $j \neq i$ are fixed. Such a function is called *m-linear*. In mathematics, another name for such multi-linear object is *tensor*. Try to describe, while we can use matrix to represent a bilinear function, how to represent a multi-linear function then?

4. **Minors of a matrix**
Suppose M is a $m \times n$ matrix with $m \geq n$. Then M is a "rectangular" matrix with m rows and n columns. We can pick arbitrary n (say $i_1^{th}, \ldots, i_n^{th}$) distinctive rows of M and form a $n \times n$ sub-matrix of M. If we denote such $n \times n$ sub-matrix as $M_{i_1, \ldots, i_n}$, its determinant $\det(M_{i_1, \ldots, i_n})$ is called a *minor* of M. Prove that the n column vectors are linearly dependent if and only if all the minors of M are identically zero, i.e.

$$\det(M_{i_1, \ldots, i_n}) = 0, \quad \forall i_1, i_2, \ldots, i_n \in [1, m].$$

By the way, totally how many different minors can you possibly gotten there? (Hint: For the "if and only if" problem, the "only if" part is easy; for the "if" part, proof by induction makes it easier.) This fact was used to derive the traditional multi-linear constraints in some computer vision literature.

5. **Linear dependency of two vectors**
Given any four *non-zero* vectors $a_1, \ldots, a_n, b_1, \ldots, b_n \in \mathbb{R}^3$, the

following matrix

$$\begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_n & b_n \end{bmatrix} \in \mathbb{R}^{3n\times 2} \tag{7.42}$$

is rank deficient if and only if $a_i b_j^T - b_i a_j^T = 0$ for all $i, j = 1, \ldots, n$. Explain what happens if some of the vectors are zero.

Chapter 8

Geometry and reconstruction from point features

In the previous chapter we have introduced a rank condition that, we claimed, captures all independent constraints between corresponding points in an arbitrary number of images. We have seen that, in the case of two views, the rank condition reduces to the epipolar constraint. Unlike the epipolar constraint, however, it can be used to derive useful constraints between correspondences in three views. In this chapter we carry out the rest of our program: we derive all independent constraints between correspondences in an arbitrary number of images. In the chapters that follow, we will gradually extend the theory to all primitive geometric features: points, lines, planes, as well as an arbitrary mixture of them.

8.1 Multiple views of a point

We first recall the notation used in Chapter 7: the generic point in space $p \in \mathbb{E}^3$ has coordinates $\mathbf{X} = [X, Y, Z, 1]^T$ relative to a fixed (inertial) coordinate frame. The coordinates of its projection onto the image plane of a moving camera are $\mathbf{x}_i = [x(t_i), y(t_i), 1]^T$ for $t_1, t_2, \ldots, t_m$; the two are related by

$$\lambda_i \mathbf{x}_i = A_i P g_i \mathbf{X}, \quad i = 1, \ldots, m \tag{8.1}$$

where $\lambda_i \in \mathbb{R}_+$ is the (unknown) depth of the point p relative to the camera frame, $A_i \in SL(3)$ is the camera calibration matrix (at time t_i), $P = [I, 0] \in \mathbb{R}^{3\times 4}$ is the constant projection matrix and $g(t) \in SE(3)$ is the coordinate transformation from the world frame to the camera frame at

time t_i. We call $\Pi_i \doteq A(t_i)Pg(t_i)$. Collecting all the indices, we can write (8.1) in matrix form as

$$\begin{array}{rcl} \mathcal{I}\vec{\lambda} & = & \Pi\mathbf{X} \end{array}$$

$$\begin{bmatrix} \mathbf{x}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{x}_m \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} \Pi_1 \\ \Pi_2 \\ \vdots \\ \Pi_m \end{bmatrix} \mathbf{X}. \tag{8.2}$$

$\vec{\lambda} \in \mathbb{R}^m$ is the *depth scale vector*, and $\Pi \in \mathbb{R}^{3m\times 4}$ is the *projection matrix* associated to the *image matrix* $\mathcal{I} \in \mathbb{R}^{3m\times m}$.

The condition for the above equation to have a solution can be written in terms of the following matrix in $\mathbb{R}^{3m\times(m+4)}$

$$N_p \doteq [\Pi,\, \mathcal{I}] = \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & \cdots & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \Pi_m & 0 & \cdots & 0 & \mathbf{x}_m \end{bmatrix} \tag{8.3}$$

where we use the subscript p to indicate that the matrix N_p is associated to a point feature. For $v = [\mathbf{X}^T, -\vec{\lambda}^T]^T \in \mathbb{R}^{m+4}$ to solve equation (8.2), N_p must have a non-trivial null space, that is, this $3m \times (m+4)$ matrix must be *rank-deficient*

$$\boxed{\text{rank}(N_p) \leq m+3} \tag{8.4}$$

Remark 8.1 (Null space of N_p). *Even though equations (8.2) and (8.4) are equivalent, if rank$(N_p) \leq m+2$ then equation $N_p v = 0$ has more than one solution. In the next section, we will show that rank(N_p) is either $m+2$ or $m+3$ and that the first case happens if and only if the point being observed and all the camera centers lie on the same line.*

Remark 8.2 (Positive depth). *Even if rank$(N_p) = m+3$, there is no guarantee that $\vec{\lambda}$ in (8.2) will have positive entries.*[1] *In practice, if the point being observed is always in front of the camera, then $\mathcal{I}$, Π and $\mathbf{X}$ in (8.2) will be such that the entries of $\vec{\lambda}$ are positive. Since the solution to $N_p v = 0$ is unique and $v \doteq [\mathbf{X}^T, -\vec{\lambda}^T]^T$ is a solution, then the last m entries of v have to be of the same sign.*

It is a basic fact of linear algebra that a matrix being rank-deficient can be expressed in terms of the determinants of certain submatrices being zero.

[1] In a projective setting, this is not a problem since all points on the same line through the camera center are equivalent. But it does matter if we want to choose appropriate reference frames such that the depth for the recovered 3-D point is physically meaningful, i.e. positive.

In particular, for $N_p \in \mathbb{R}^{3m\times m+4}$, $m \geq 2$, all submatrices of dimension $(m+4)\times(m+4)$ must be zero. Each determinant is a polynomial equation in the entries of the projection matrix Π and images $\mathbf{x}_1, \ldots, \mathbf{x}_m$ having the general form

$$f(\Pi, \mathbf{x}_1, \ldots, \mathbf{x}_m) = 0. \tag{8.5}$$

The polynomials $f(\cdot)$ are always linear in each $\mathbf{x}_i$, $i = 1, \ldots, m$. For the case of $m = 2, 3, 4$, they are known as *bilinear, trilinear and quadrilinear constraints* respectively. These constraints can be written in tensorial form using the so-called *bifocal, trifocal and quadrifocal tensors*.

As we have anticipated in Chapter 7, rather than using tensors and multi-indices, we are going to derive all independent constraints explicitly from the rank condition (8.4). We begin with manipulating the matrix N_p to arrive at a simpler rank condition in the next section.

8.2 The multiple view matrix and its rank

Without loss of generality, we may assume that the first camera frame is chosen to be the reference frame. [2] That gives the projection matrices $\Pi_i, i = 1, \ldots, m$

$$\Pi_1 = [I, 0],\ \ldots,\ \Pi_m = [R_m, T_m] \quad \in \mathbb{R}^{3\times 4}, \tag{8.6}$$

where $R_i \in \mathbb{R}^{3\times 3}, i = 2, \ldots, m$ represent the first three columns of Π_i and $T_i \in \mathbb{R}^3, i = 2, \ldots, m$ is the fourth column of Π_i. Although we have used the suggestive notation (R_i, T_i) here, they are not necessarily the actual rotation and translation. Only in the case when the camera is perfectly calibrated, i.e. $A_i = I$, does R_i correspond to actual camera rotation and T_i to translation.

With the above notation, we eliminate $\mathbf{x}_1$ from the first row of N_p using column manipulation. It is easy to see that N_p has the same rank as the following matrix in $\mathbb{R}^{3m\times(m+4)}$

$$\begin{bmatrix} I & 0 & 0 & 0 & \cdots & 0 \\ R_2 & T_2 & R_2\mathbf{x}_1 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ R_m & T_m & R_m\mathbf{x}_1 & 0 & 0 & \mathbf{x}_m \end{bmatrix} = \left[\begin{array}{c|c} I & 0 \\ \hline R_2 & \\ \vdots & N_p' \\ R_m & \end{array}\right].$$

[2] Depending on the context, the reference frame could be either an Euclidean, affine or projective reference frame. In any case, the projection matrix for the first image becomes $[I,\ 0] \in \mathbb{R}^{3\times 4}$.

Hence, the original N_p is rank-deficient if and only if the sub-matrix $N_p' \in \mathbb{R}^{3(m-1)\times(m+1)}$ is. Multiplying N_p' on the left by the following matrix[3]

$$D_p = \begin{bmatrix} \mathbf{x}_2^T & 0 & \cdots & 0 \\ \widehat{\mathbf{x}_2} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \mathbf{x}_m^T \\ 0 & \cdots & 0 & \widehat{\mathbf{x}_m} \end{bmatrix} \in \mathbb{R}^{4(m-1)\times 3(m-1)} \tag{8.7}$$

yields the following matrix $D_p N_p'$ in $\mathbb{R}^{4(m-1)\times(m+1)}$

$$\begin{bmatrix} \mathbf{x}_2^T T_2 & \mathbf{x}_2^T R_2 \mathbf{x}_1 & \mathbf{x}_2^T \mathbf{x}_2 & 0 & 0 & 0 \\ \widehat{\mathbf{x}_2} T_2 & \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & 0 & 0 & \ddots & 0 \\ \mathbf{x}_m^T T_m & \mathbf{x}_m^T R_m \mathbf{x}_1 & 0 & 0 & 0 & \mathbf{x}_m^T \mathbf{x}_m \\ \widehat{\mathbf{x}_m} T_m & \widehat{\mathbf{x}_m} R_m \mathbf{x}_1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Since D_p has full rank $3(m-1)$, we have $\text{rank}(N_p') = \text{rank}(D_p N_p')$. Hence the original matrix N_p is rank-deficient if and only if the following sub-matrix of $D_p N_p'$ is rank-deficient

$$M_p = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \mathbf{x}_1 & \widehat{\mathbf{x}_3} T_3 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m} R_m \mathbf{x}_1 & \widehat{\mathbf{x}_m} T_m \end{bmatrix} \in \mathbb{R}^{3(m-1)\times 2}. \tag{8.8}$$

We call M_p the *multiple view matrix* associated to a point feature p. More precisely, we have proved the following:

Theorem 8.1 (Rank equivalence condition for point features). *Matrices N_p and M_p satisfy*

$$\boxed{\textit{rank}(M_p) = \textit{rank}(N_p) - (m+2) \leq 1.} \tag{8.9}$$

Therefore, rank(N_p) is either $m+3$ or $m+2$, depending on whether rank(M_p) is 1 or 0, respectively.

Comment 8.1 (Geometric interpretation of M_p). *Notice that $\widehat{\mathbf{x}}_i T_i$ is the normal to the epipolar plane given by frames 1 and i, that is $\widehat{\mathbf{x}}_i R_i \mathbf{x}_1$. Therefore, the rank condition not only implies that these two normals are parallel (as obvious from the epipolar constraint) but also that the scale between these two possible normal vectors is the same for all frames. A more detailed geometric interpretation is given in the next section.*

[3]For $u \in \mathbb{R}^3$, we use $\widehat{u} \in \mathbb{R}^{3\times 3}$ to denote the skew symmetric generating the cross product, i.e. for any vector $v \in \mathbb{R}^3$, we have $\widehat{u}v = u \times v$; see Chapter 2.

Due to the rank equality (8.9) for M_p and N_p, we conclude that the rank-deficiency of M_p is equivalent to all multi-linear constraints that may arise among the m images. To see this more explicitly, notice that for M_p to be rank-deficient, it is necessary for the any pair of the vectors $\widehat{\mathbf{x}}_i T_i, \widehat{\mathbf{x}}_i R_i \mathbf{x}_1$ to be linearly dependent. This gives us the well-known bilinear (or epipolar) constraints

$$\mathbf{x}_i^T \widehat{T}_i R_i \mathbf{x}_1 = 0. \tag{8.10}$$

between the i^{th} and 1^{st} images. Hence the constraint $\text{rank}(M_p) \leq 1$ consistently generalizes the epipolar constraint (for 2 views) to arbitrary m views. Using Lemma 7.1. we can conclude that

$$\widehat{\mathbf{x}}_i (T_i \mathbf{x}_1^T R_j^T - R_i \mathbf{x}_1 T_j^T) \widehat{\mathbf{x}_j} = 0. \tag{8.11}$$

Note that this is a matrix equation and it gives a total of $3 \times 3 = 9$ scalar (trilinear) equations in terms of $\mathbf{x}_1, \mathbf{x}_i, \mathbf{x}_j$. These 9 equations are exactly the 9 trilinear constraints that one would obtain from the minors of N_p following the conventional derivation of trilinear constraints.

Notice that the multiple view matrix M_p being rank-deficient is equivalent to all its 2×2 minors having determinant zero. Since the 2×2 minors involve three images only, we conclude the following:

- There are *no additional* independent relationships among images among any four views. Hence, the so-called *quadrilinear constraints* do not impose any new constraints on the four images other than the trilinear and bilinear constraints. This is an indirect proof of the fact that the quadrifocal tensors can be factorized into trifocal tensors or bifocal tensors.[4]
- Trilinear constraints (8.11) in general implies bilinear constraints (8.10), except when $\widehat{\mathbf{x}}_i T_i = \widehat{\mathbf{x}}_i R_i \mathbf{x}_1 = 0$ for some i. This corresponds to a degenerate case in which the pre-image p lies on the line through optical centers o_1, o_i.

So far we have essentially given a much more simplified proof for the following facts regarding multilinear constraints among multiple images:

Theorem 8.2 (Linear relationships among multiple views of a point). *For any given m images corresponding to a point $p \in \mathbb{E}^3$ relative to m camera frames, that the matrix M_p is of rank no more than 1 yields the following:*

1. *Any algebraic constraint among the m images can be reduced to only those involving 2 and 3 images at a time. Formulae of these bilinear and trilinear constraints are given by (8.10) and (8.11) respectively.*

[4] Although we only proved it for the special case with $\Pi_1 = [I,\ 0]$, the general case differs from this special one only by a choice of a reference frame.

2. *For given m images of a point, all the triple-wise trilinear constraints algebraically imply all pairwise bilinear constraints, except in the degenerate case in which the pre-image p lies on the line through optical centers o_1, o_i for some i.*

Comment 8.2 (Rank condition v.s. multilinear constraints). *Our discussion implies that multilinear constraints are certainly necessary for the rank of matrix M_p (hence N_p) to be deficient. But, rigorously speaking, they are* **not** *sufficient. According to Lemma 7.1, for multilinear constraints to be equivalent to the rank condition, vectors in the M_p matrix need to be non-zero. This is not always true for certain degenerate cases, as mentioned above. Hence, the rank condition on M_p provides a more general framework for capturing* all *constraints among multiple images.*

8.3 Geometric interpretation of the rank condition

In the previous section, we have classified all algebraic constraints that may arise among m corresponding images of a point. We now know that the relationship among m images essentially boils down to that between 2 and 3 views at a time, characterized by the bilinear constraint (8.10) and trilinear constraint (8.11) respectively. But we have not yet explained what these equations mean and whether there is a simpler intuitive geometric explanation for all these algebraic relationships.

8.3.1 Uniqueness of the pre-image

Given 3 vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{R}^3$, if they are indeed images of some 3-D point p with respect to the three camera frames – as shown in Figure 8.1 – they should automatically satisfy both the bilinear and trilinear constraints, e.g.

$$\begin{aligned} &\text{bilinear:} \quad \mathbf{x}_2^T \widehat{T}_2 R_2 \mathbf{x}_1 = 0, \quad \mathbf{x}_3^T \widehat{T}_3 R_3 \mathbf{x}_1 = 0, \\ &\text{trilinear:} \quad \widehat{\mathbf{x}_2}(T_2 \mathbf{x}_1^T R_3^T - R_2 \mathbf{x}_1 T_3^T)\widehat{\mathbf{x}_3} = 0. \end{aligned}$$

Now the inverse problem: If the three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ satisfy either the bilinear constraints or trilinear constraints, are they necessarily images of some single point in 3-D, the so-called *pre-image*?

Let us first study whether bilinear constraints are sufficient to determine a unique pre-image in 3-D. For the given three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$, suppose that they satisfy three pairwise epipolar constraints

$$\mathbf{x}_2^T F_{21} \mathbf{x}_1 = 0, \quad \mathbf{x}_3^T F_{31} \mathbf{x}_1 = 0, \quad \mathbf{x}_3^T F_{32} \mathbf{x}_2 = 0, \tag{8.12}$$

with $F_{ij} = \widehat{T_{ij}} R_{ij}$ the fundamental matrix between the i^{th} and j^{th} images. Note that each image (as a point on the image plane) and the corresponding optical center uniquely determine a line in 3-D that passes through

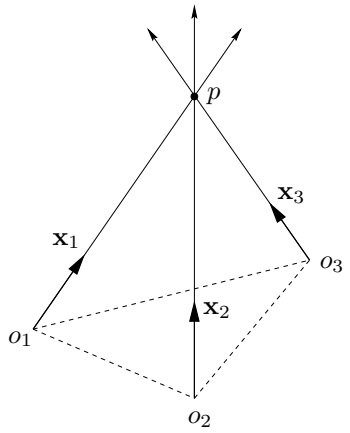

Figure 8.1. Three rays extended from the three images $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ intersect at one point p in 3-D, the pre-image of $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$.

them. This gives us a total of three lines. Geometrically, the three epipolar constraints simply imply that each pair of the three lines are coplanar. So when do three pairwise coplanar lines intersect at exactly one point in 3-D? If these three lines are not coplanar, the intersection is uniquely determined, so is the pre-image. If all of them do lie on the same plane, such a unique intersection is not always guaranteed. As shown in Figure 8.2, this may occur when the lines determined by the images lie on the plane spanned by the three optical centers o_1, o_2, o_3, the so-called *trifocal plane*, or when the three optical centers lie on a straight line regardless of the images, the so-called *rectilinear motion*. The first case is of less prac-

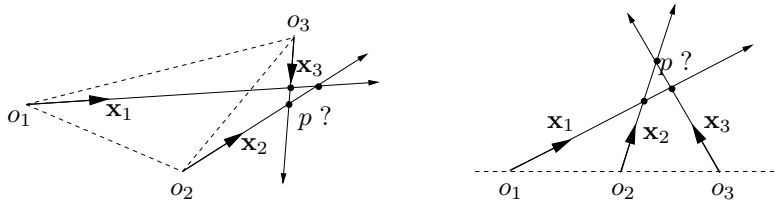

Figure 8.2. Two cases when the three lines determined by the three images $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ lie on the same plane, in which case they may not necessarily intersect at a unique point p.

tical effect since 3-D points generically do not lie on the trifocal plane. The second case is more important: regardless of what 3-D feature points one chooses, pairwise epipolar constraints alone do not provide sufficient constraints to determine a unique 3-D point from any given three image vectors. In such a case, extra constraints need to be imposed on the three images in order to obtain a unique pre-image.

Would trilinear constraints suffice to salvage the situation? The answer to this is yes and let us show why. Given any three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{R}^3$, suppose they satisfy the trilinear constraint equation

$$\widehat{\mathbf{x}_2}(T_2\mathbf{x}_1^T R_3^T - R_2\mathbf{x}_1 T_3^T)\widehat{\mathbf{x}_3} = 0.$$

In order to determine $\mathbf{x}_3$ uniquely (up to a scale) from this equation, we need the matrix

$$\widehat{\mathbf{x}_2}(T_2\mathbf{x}_1^T R_3^T - R_2\mathbf{x}_1 T_3^T) \in \mathbb{R}^{3\times 3}$$

to be of exact rank 1. The only case that $\mathbf{x}_3$ is undetermined is when this matrix is rank 0, that is

$$\widehat{\mathbf{x}_2}(T_2\mathbf{x}_1^T R_3^T - R_2\mathbf{x}_1 T_3^T) = 0.$$

That is

$$\text{range}(T_2\mathbf{x}_1^T R_3^T - R_2\mathbf{x}_1 T_3^T) \subset \text{span}\{\mathbf{x}_2\}. \tag{8.13}$$

If T_3 and $R_3\mathbf{x}_1$ are linearly independent, then (8.13) holds if and only if the vectors $R_2\mathbf{x}_1, T_2, \mathbf{x}_2$ are linearly dependent. This condition simply means that the line associated to the first image $\mathbf{x}_1$ coincide with the line determined by the optical centers o_1, o_2.[5] If T_3 and $R_3\mathbf{x}_1$ are linearly dependent, $\mathbf{x}_3$ is determinable since $R_3\mathbf{x}_1$ lies on the line determined by the optical centers o_1, o_3. Hence we have shown, that $\mathbf{x}_3$ cannot be uniquely determined from $\mathbf{x}_1, \mathbf{x}_2$ by the trilinear constraint if and only if

$$\widehat{T_2}R_2\mathbf{x}_1 = 0, \quad \text{and} \quad \widehat{T_2}\mathbf{x}_2 = 0. \tag{8.14}$$

Due to the symmetry of the trilinear constraint equation, $\mathbf{x}_2$ is not uniquely determined from $\mathbf{x}_1, \mathbf{x}_3$ by the trilinear constraint if and only if

$$\widehat{T_3}R_3\mathbf{x}_1 = 0, \quad \text{and} \quad \widehat{T_3}\mathbf{x}_3 = 0. \tag{8.15}$$

We still need to show that these three images indeed determine a unique pre-image in 3-D if either one of the images can be determined from the other two by the trilinear constraint. This is obvious. Without loss of generality, suppose it is $\mathbf{x}_3$ that can be uniquely determined from $\mathbf{x}_1$ and $\mathbf{x}_2$. Simply take the intersection $p' \in \mathbb{E}^3$ of the two lines associated to the first two images and project it back to the third image plane – such intersection exists since the two images satisfy epipolar constraint.[6] Call this image $\mathbf{x}_3'$. Then $\mathbf{x}_3'$ automatically satisfies the trilinear constraint. Hence $\mathbf{x}_3' = \mathbf{x}_3$ due to its uniqueness. Therefore, p' is the 3-D point p where all the three lines intersect in the first place. As we have argued before, the trilinear constraint (8.11) actually implies bilinear constraint (8.10). Therefore, the 3-D pre-image p is uniquely determined if either $\mathbf{x}_3$ can be determined from

[5] In other words, the pre-image point p lies on the epipole between the first and second camera frames.

[6] If these two lines are parallel, we take the intersection in the plane at infinity.

$\mathbf{x}_1, \mathbf{x}_2$ or $\mathbf{x}_2$ can be determined from $\mathbf{x}_1, \mathbf{x}_3$. So we have in fact proved the following known fact:

Lemma 8.1 (Properties of bilinear and trilinear constraints). *Given three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{R}^3$ and three camera frames with distinct optical centers. If the three images satisfy pairwise epipolar constraints*

$$\mathbf{x}_i^T \widehat{T_{ij}} R_{ij} \mathbf{x}_j = 0, \quad i, j = 1, 2, 3,$$

a unique pre-image p is determined except when the three lines associated to the three images are coplanar. If these vectors satisfy all triple-wise trilinear constraints[7]

$$\widehat{\mathbf{x}_j}(T_{ji}\mathbf{x}_i^T R_{ki}^T - R_{ji}\mathbf{x}_i T_{ki}^T)\widehat{\mathbf{x}_k} = 0, \quad i, j, k = 1, 2, 3,$$

then they determine a unique pre-image $p \in \mathbb{E}^3$ except when the three lines associated to the three images are collinear.

The two cases (which are essentially one case) in which the bilinear constraints may become degenerate are shown in Figure 8.2. Figure 8.3 shows the only case in which trilinear constraint may become degenerate. In sim-

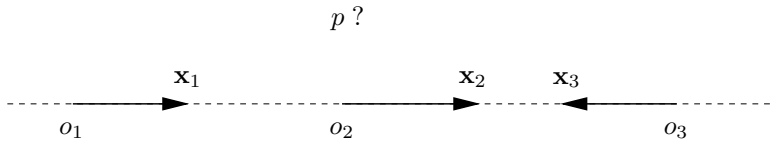

Figure 8.3. If the three images and the three optical centers lie on a straight line, any point on this line is a valid pre-image that satisfies all the constraints.

ple terms, "bilinear fails for coplanar and trilinear fails for collinear". For more than 3 views, in order to check the uniqueness of the pre-image, one needs to apply the above lemma to every pairwise or triple-wise views. The possible number of combinations of degenerate cases make it very hard to draw any consistent conclusion. However, in terms of the rank condition on the multiple view matrix, Lemma 8.1 can be generalized to multiple views in a much more concise and unified way:

Theorem 8.3 (Uniqueness of the pre-image). *Given m vectors on the image planes with respect to m camera frames, they correspond to the same point in the 3-D space if the rank of the M_p matrix relative to any of the camera frames is 1. If its rank is 0, the point is determined up to the line where all the camera centers must lie on.*

Hence both the largest and smallest singular values of M_p have meaningful geometric interpretation: the smallest being zero is necessary for

[7] Although there seem to be total of nine possible (i, j, k), there are in fact only three different trilinear constraints due to the symmetry in the trilinear constraint equation.

a unique pre-image; the largest being zero is necessary for a non-unique pre-image.

8.3.2 Geometry of the multiple view matrix

Now we are ready to discover the interesting geometry that the matrix M_p really captures. From the equation

$$\lambda_i \mathbf{x}_i = \lambda_1 R_i \mathbf{x}_1 + T_i, \quad i = 2, \ldots, m, \tag{8.16}$$

it is direct to see that

$$M_p \begin{bmatrix} \lambda_1 \\ 1 \end{bmatrix} = 0. \tag{8.17}$$

So the coefficient that relates the two columns of the M_p matrix is simply the depth of the point p in 3-D space with respect to the center of the first camera frame (the reference).[8] Hence, from the M_p matrix alone, we may know the distance from the point p to the camera center o_1. One can further prove that for any two points of the same distance from o_1, there always exist a set of camera frames for each point such that their images give exactly the same M_p matrix.[9] Hence we can interpret M_p as a map from a point in 3-D space to a scalar

$$M_p : p \in \mathbb{R}^3 \mapsto d \in \mathbb{R}_+,$$

where $d = \|p - o_1\|$. This map is certainly surjective but not injective. Points that may give rise to the same M_p matrix hence lie on a sphere $\mathbb{S}^2$ centered around the camera center o_1. The above scalar d is exactly the radius of the sphere. We may summarize our discussion into the following theorem:

Theorem 8.4 (Geometry of the multiple view matrix M_p). *The matrix M_p associated to a point p in 3-D maps this point to a unique scalar d. This map is surjective but not injective and two points may give rise to the same M_p matrix if and only if they are of the same distance to the center o_1 of the reference camera frame. That distance is given by the coefficients which relate the two columns of M_p.*

Therefore, knowing M_p (i.e. the distance d), we know that p must lie on a sphere of radius $d = \|p - o_1\|$. Given three camera frames, we can choose either camera center as the reference, then we essentially have three M_p matrix for each point. Each M_p matrix gives a sphere around each camera center in which the point p should stay. The intersection of two such spheres in 3-D space is generically a circle, as shown in Figure 8.4. Then one would

[8] Here we implicitly assume that the image surface is a sphere with radius 1. If it is a plane instead, statements below should be interpreted accordingly.

[9] We omit the detail of the proof here for simplicity.

imagine that, in general, the intersection of all three spheres determines the 3-D location of the point p up to two solutions, unless all the camera centers lie on the same line as o_1, o_2 (i.e. except for the rectilinear motion). In the next chapter which studies rank condition for a line, we further show

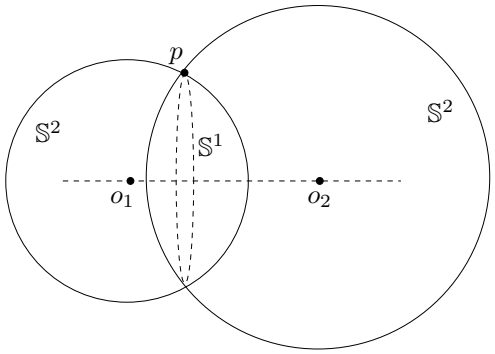

Figure 8.4. The rank-deficient matrices M_p of a point relative to two distinct camera centers determine the point p up to a circle.

that there are some profound relationships between the M_p matrix for a point and that for a line.

8.4 Applications of the rank condition

The rank condition of the multiple view matrix M_p allows us to use all the constraints among multiple images simultaneously for purposes such as feature matching or motion recovery, without specifying a particular set of pairwise or triple-wise frames. Ideally, one would like to formulate the entire problem of reconstructing 3-D motion and structure as one optimizing some global objective function subject to the rank condition. However, such an approach usually relies on very difficult nonlinear optimization methods. Instead, we here divide the overall problem into a few subproblems: matching features assuming known motion, and estimating motion assuming known matched features.

8.4.1 Correspondence

Notice that $M_p \in \mathbb{R}^{3(m-1)\times 2}$ being rank-deficient is equivalent to the determinant of $M_p^T M_p \in \mathbb{R}^{2\times 2}$ being zero

$$\det(M_p^T M_p) = 0. \tag{8.18}$$

In general $M_p^T M_p$ is a function of the projection matrix Π and images $\mathbf{x}_1, \ldots, \mathbf{x}_m$. If Π is known and we like to test if given m vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m \in$

$\mathbb{R}^3$ indeed satisfy all the constraints that m images of a single 3-D preimage should, we only need to test if the above determinant is zero. A more numerically robust algorithm would be:

Algorithm 8.1 (Multiple view matching test). *Suppose the projection matrix Π associated to m camera frames are given. Then for given m vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^3$,*

1. *Compute the matrix $M_p \in \mathbb{R}^{3(m-1)\times 2}$ according to (8.8).*
2. *Compute second eigenvalue λ_2 of $M_p^T M_p$;*
3. *If $\lambda_2 \leq \epsilon$ for some pre-fixed threshold, the m image vectors match.*

8.4.2 Reconstruction

Now suppose that m images $\mathbf{x}_1^i, \ldots, \mathbf{x}_m^i$ of n points p^i, $i = 1, \ldots, n$ are given and we want to use them to estimate the unknown projection matrix Π. The rank condition of the M_p matrix can be written as

$$\alpha^i \begin{bmatrix} \widehat{\mathbf{x}_2^i} T_2 \\ \widehat{\mathbf{x}_3^i} T_3 \\ \vdots \\ \widehat{\mathbf{x}_m^i} T_m \end{bmatrix} + \begin{bmatrix} \widehat{\mathbf{x}_2^i} R_2 \mathbf{x}_1^i \\ \widehat{\mathbf{x}_3^i} R_3 \mathbf{x}_1^i \\ \vdots \\ \widehat{\mathbf{x}_m^i} R_m \mathbf{x}_1^i \end{bmatrix} = 0 \in \mathbb{R}^{3(m-1)\times 1}, \tag{8.19}$$

for proper $\alpha^i \in \mathbb{R}, i = 1, \ldots n$.

From (8.1) we have $\lambda_j^i \mathbf{x}_j^i = \lambda_1^i R_j \mathbf{x}_1^i + T_j$. Multiplying by $\widehat{\mathbf{x}_j^i}$ we obtain $\widehat{\mathbf{x}_j^i}(R_j \mathbf{x}_1^i + T_j/\lambda_1^i) = 0$. Therefore, $\alpha^i = 1/\lambda_1^i$ can be interpreted as the inverse of the depth of point p^i with respect to the first frame. The set of equations in (8.19) is equivalent to finding vectors $\vec{R}_j = [r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}]^T \in \mathbb{R}^9$ and $\vec{T}_j = T_j \in \mathbb{R}^3$, $j = 2, \ldots, m$, such that

$$P_j \begin{bmatrix} \vec{T}_j \\ \vec{R}_j \end{bmatrix} = \begin{bmatrix} \alpha^1 \widehat{\mathbf{x}_j^1} & \widehat{\mathbf{x}_j^1} * {\mathbf{x}_1^1}^T \\ \alpha^2 \widehat{\mathbf{x}_j^2} & \widehat{\mathbf{x}_j^2} * {\mathbf{x}_1^2}^T \\ \vdots & \vdots \\ \alpha^n \widehat{\mathbf{x}_j^n} & \widehat{\mathbf{x}_j^n} * {\mathbf{x}_1^n}^T \end{bmatrix} \begin{bmatrix} \vec{T}_j \\ \vec{R}_j \end{bmatrix} = 0 \in \mathbb{R}^{3n}, \tag{8.20}$$

where $A * B$ is the *Kronecker product* of A and B. It can be shown that if α^i's are known, the matrix P_j is of rank 11 if more than $n \geq 6$ points in general position are given. In that case, the kernel of P_j is unique, and so is (R_j, T_j).

Euclidean reconstruction

For simplicity, we here assume that the camera is perfectly calibrated. Therefore, is $A(t) = I$, R_i is a rotation matrix in $SO(3)$ and T_i is a trans-

lation vector in $\mathbb{R}^3$. Given the first two images of (at least) eight points in general configuration, $T_2 \in \mathbb{R}^3$ and $R_2 \in SO(3)$ can be estimated using the classic *eight point algorithm*. The equation given by the first row in (8.19) implies $\alpha^i \widehat{\mathbf{x}_2^i} T_2 = -\widehat{\mathbf{x}_2^i} R_2 \mathbf{x}_1^i$, whose least squares solution up to scale (recall that T_2 is recovered up to scale from the eight point algorithm) is given by

$$\alpha^i = -\frac{(\widehat{\mathbf{x}_2^i} T_2)^T \widehat{\mathbf{x}_2^i} R_2 \mathbf{x}_1^i}{||\widehat{\mathbf{x}_2^i} T_2||^2}, \quad i = 1, \ldots, n. \tag{8.21}$$

These values of α^i can therefore be used to initialize the equation (8.19). Multiplying on the left the j-th block of (8.19) by T_j^T yields the epipolar constraints ${\mathbf{x}_j^i}^T \widehat{T_j} R_j \mathbf{x}_1^i = 0$. We know that in general the solution (R_j, T_j) of these equations is unique, with T_j recovered up to scale. Hence the solution to (8.19) is unique and we can then recover from (8.19) the scale of each T_j up to a single scale for all the T_j's (recall that the α^i's were computed up to scale). Since the α^i's are known, (8.20) becomes a set of linear equations on $\vec{T}_j$ and $\vec{R}_j$, whose solution can be described as follows.

Let $\tilde{T}_j \in \mathbb{R}^3$ and $\tilde{R}_j \in \mathbb{R}^{3\times 3}$ be the (unique) solution of (8.20). Such a solution is obtained as the eigenvector of P_j associated to the smallest singular value. Let $\tilde{R}_j = U_j S_j V_j^T$ be the SVD of $\tilde{R}_j$. Then the solution of (8.20) in $\mathbb{R}^3 \times SO(3)$ is given by:

$$T_j = \frac{\det(U_j V_j^T)}{\sqrt[3]{\det(S_j)}} \tilde{T}_j \in \mathbb{R}^3, \tag{8.22}$$

$$R_j = \det(U_j V_j^T)\, U_j V_j^T \in SO(3). \tag{8.23}$$

In the presence of noise, solving for α^i using only the first two frames may not necessarily be the best thing to do. Nevertheless, this arbitrary choice of α^i allows us to compute all the motions (R_j, T_j), $j = 2, \ldots, m$. Given all the motions, the least squares solution for α^i from (8.19) is given by

$$\alpha^i = -\frac{\sum_{j=2}^m (\widehat{\mathbf{x}_j^i} T_j)^T \widehat{\mathbf{x}_j^i} R_j \mathbf{x}_1^i}{\sum_{j=2}^m ||\widehat{\mathbf{x}_j^i} T_j||^2}, \; i = 1, \ldots, n. \tag{8.24}$$

Note that these α^i's are the same as those in (8.21) if $m = 2$. One can then recompute the motion given this new α^i's, until the error in α is small enough.

We then have the following linear algorithm for multiple view motion and structure estimation:

Algorithm 8.2 (Multiple view six-eight point algorithm). *Given m images $\mathbf{x}_1^i, \ldots, \mathbf{x}_m^i$ of n points p^i, $i = 1, \ldots, n$, estimate the motions (R_j, T_j), $j = 2, \ldots, m$ as follows:*

1. *Initialization: $k = 0$*

   (a) *Compute* (R_2, T_2) *using the eight point algorithm for the first two views.*
   (b) *Compute* $\alpha_k^i = \alpha^i/\alpha^1$ *from (8.21).*

2. *Compute* $(\tilde{R}_j, \tilde{T}_j)$ *as the eigenvector associated to the smallest singular value of* P_j, $j = 2, \ldots, m$.

3. *Compute* (R_j, T_j) *from (8.22) and (8.23) for* $j = 2, \ldots, m$.

4. *Compute the new* $\alpha^i = \alpha_{k+1}^i$ *from (8.24). Normalize so that* $\alpha_{k+1}^1 = 1$.

5. *If* $||\alpha_k - \alpha_{k+1}|| > \epsilon$, *for a pre-specified* $\epsilon > 0$, *then* $k = k+1$ *and goto 2. Else stop.*

The camera motion is then $(R_j, T_j), j = 2, \ldots, m$ and the structure of the points (with respect to the first camera frame) is given by the converged depth scalar $\lambda_1^i = 1/\alpha^i, i = 1, \ldots, n$. There are a few notes for the proposed algorithm:

1. It makes use of all multilinear constraints simultaneously for motion and structure estimation. (R_j, T_j)'s seem to be estimated using pairwise views only but that is not exactly true. The computation of the matrix P_j depends on all α^i each of which is in turn estimated from the M_p^i matrix involving all views. The reason to set $\alpha_{k+1}^1 = 1$ is to fix the universal scale. It is equivalent to putting the first point at a distance of 1 to the first camera center.

2. It can be used either in a batch fashion or a recursive one: initializes with two views, recursively estimates camera motion and automatically updates scene structure when new data arrive.

3. It may effectively eliminate the effect of occluded feature points - if a point is occluded in a particular image, simply drop the corresponding group of three rows in the M_p matrix without affecting the condition on its rank.

Remark 8.3 (Euclidean, affine or projective reconstruction). *We must point out that, although the algorithm seems to be proposed for the calibrated camera (Euclidean) case only, it works just the same if the camera is weakly calibrated (affine) or uncalibrated (projective). The only change needed is at the initialization step. In order to initialize* α^i*'s, either an Euclidean, affine or projective choice for* (R_2, T_2) *needs to be specified. This is where our knowledge of the camera calibration enters the picture. In the worst case, i.e. when we have no knowledge on the camera at all, any choice of a projective frame will do. In that case, the relative transformation among camera frames and the scene structure are only recovered up to a projective transformation. Once that is done, the rest of the algorithm runs exactly the same.*

8.5 Experiments

In this section, we show by simulations the performance of the proposed algorithm. We compare motion and structure estimates with those of the eight point algorithm for two views and then we compare the estimates as a function of the number of frames.

8.5.1 Setup

The simulation parameters are as follows: number of trials is 1000, number of feature points is 20, number of frames is 3 or 4 (unless we vary it on purpose), field of view is 90^o, depth variation is from 100 to 400 units of focal length and image size is 500×500. Camera motions are specified by their translation and rotation axes. For example, between a pair of frames, the symbol XY means that the translation is along the X-axis and rotation is along the Y-axis. If n such symbols are connected by hyphens, it specifies a sequence of consecutive motions. The ratio of the magnitude of translation $||T||$ and rotation θ, or simply the T/R *ratio*, is compared at the center of the random cloud scattered in the truncated pyramid specified by the given field of view and depth variation (see Figure 8.5). For each simulation, the amount of rotation between frames is given by a proper angle and the amount of translation is then automatically given by the T/R ratio. We always choose the amount of total motion such that all feature points will stay in the field of view for all frames. In all simulations, independent Gaussian noise with std given in pixels is added to each image point. Error measure for rotation is $\arccos\left(\frac{tr(R\tilde{R}^T)-1}{2}\right)$ in degrees where $\tilde{R}$ is an estimate of the true R. Error measure for translation is the angle between T and $\tilde{T}$ in degrees where $\tilde{T}$ is an estimate of the true T.

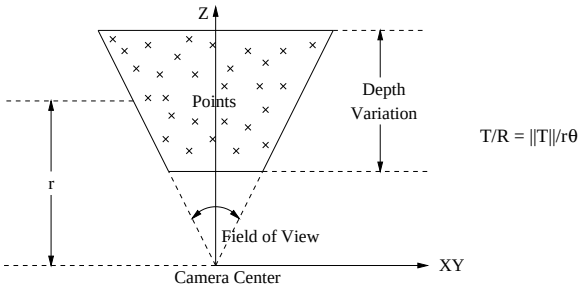

Figure 8.5. Simulation setup

8.5.2 Comparison with the 8 point algorithm

Figure 8.6 plots the errors of rotation estimates and translation estimates compared with results from the standard 8-point linear algorithm. Figure 8.7 shows a histogram of the relative translation scale for a noise level of 3 pixels as well as the error on the estimation of the structure. As we see, the multiple view linear algorithm not only generalizes but also outperforms the well-known eight point linear algorithm for two views. This is because the algorithm implicitly uses the estimated structure of the scene for the estimation of the motion, while the eight point algorithm does not.

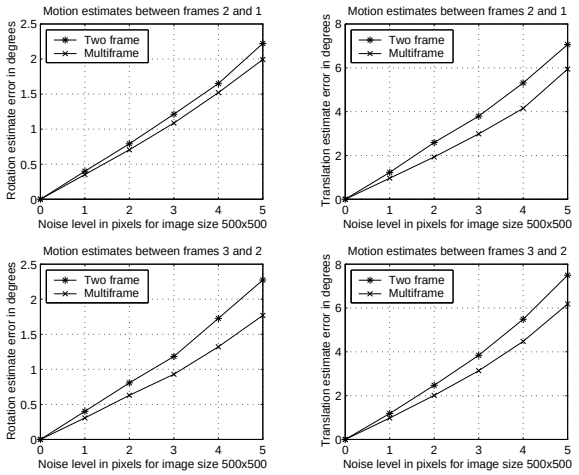

Figure 8.6. Motion estimate error comparison between 8 point algorithm and multiple view linear algorithm.

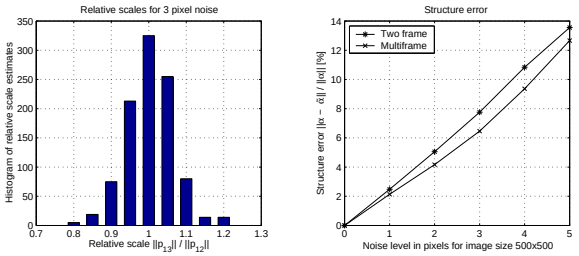

Figure 8.7. Relative scale and structure estimate error comparison. Motion is XX-YY and relative scale is 1.

8.5.3 Error as a function of the number of frames

In this simulation, we analyze the effect of the number of frames on motion and structure estimates. In principle, motion estimates should not necessarily improve, since additional frames introduce more data as well as more unknowns. However, structure estimates should improve, because additional data is available to estimate the same structure. We consider 7 frames with motion XX-YY-$X(XY)$-ZY-$(XY)Z$-$(YZ)(YZ)$ and plot the estimation errors for the first pair of frames as a function of the number of frames. As expected, that rotation estimates and relative translation scales approximately independent on the number of frames. Translation estimates with three frames are better than those with two frames, but there is no significant improvement for more than three frames. Finally, structure estimates in general improve with the number of frames.

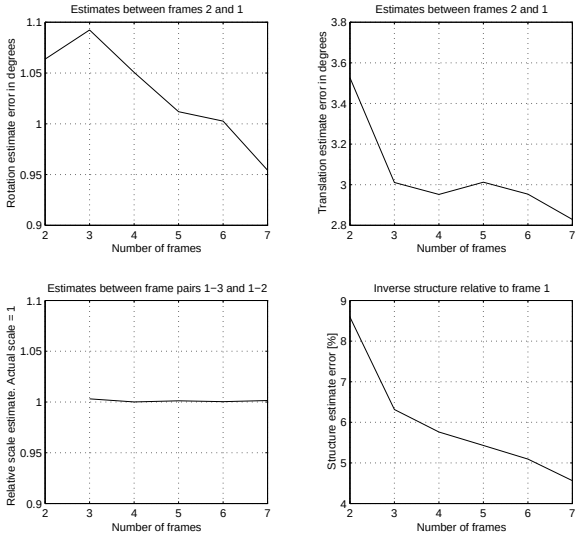

Figure 8.8. Estimation error as a function of the number of frames. Noise level is 3 pixels and relative scale is 1.

8.5.4 Experiments on real images

We now consider an indoor sequence with the camera undergoing rectilinear motion. We compare the performance of the proposed multiple view linear algorithm against the conventional eight point algorithm for two views and the multiple view nonlinear algorithm in [VMHS01]. In order to work with real images, we need to calibrate the camera, track a set of feature points and establish their correspondences across multiple frames. We calibrated the camera from a set of planar feature points using Zhang's technique

[Zha98]. For feature tracking and correspondence, we adapted the algorithm from [ZDFL95].

The multiple view nonlinear algorithm is initialized with estimates from the eight-point linear algorithm. Since the translation estimates of the linear algorithm are given up to scale only, for the multiple view case an initialization of the relative scale between consecutive translations is required. This is done by triangulation since the directions of the translations are known. For example, the relative scale between T_{21} and T_{32} is $\sin(\beta)/\sin(\gamma)$ where β is the angle between T_{31} and $R_{21}T_{21}$ and γ is the angle between T_{23} and $R_{13}T_{13}$. Recall in the multiple view case the vector of translations $\mathcal{T}$ is recovered up to one single scale. The estimated motion is then compared with the ground truth data. Error measures are the same as in the previous section.

We use 4 images of an indoor scene, with the motion of the camera in a straight line (rectilinear motion) along the Z-axis (see Figure 8.9). The relative scales between consecutive translations are 2:1 and 1:2, respectively. Even though the motion is rectilinear, relative scales still can be initialized by triangulation, because image measurements are noisy.

Figure 8.10 shows the error between the motion and structure estimates and ground truth. It can be observed that the proposed linear algorithm is able to recover the correct motion and structure, giving better results in most of the cases.

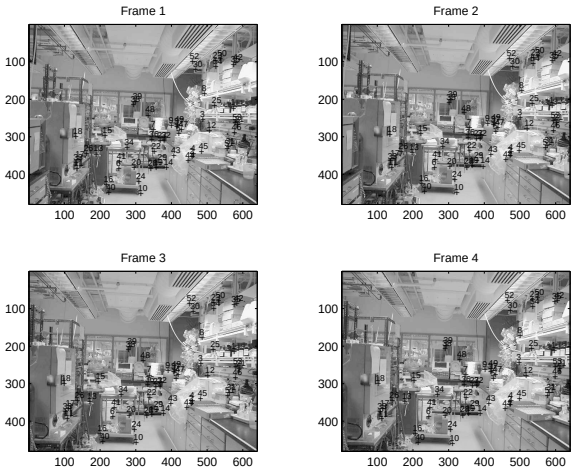

Figure 8.9. Indoor rectilinear motion image sequence

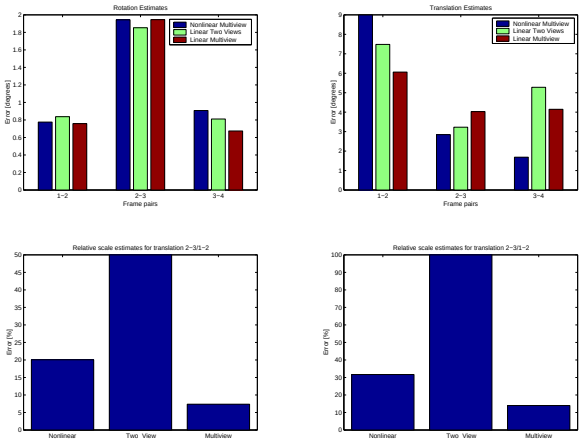

Figure 8.10. Motion and structure estimates for indoor rectilinear motion image sequence

8.6 Summary

8.7 Exercises

1. **Rectilinear motion**
   Use the rank condition of the M_p matrix to show that if the camera is calibrated and only translating on a straight line, then the relative translational scale between frames cannot be recovered from pairwise bilinear constraints, but it can be recovered from trilinear constraints. (Hint: use $R_i = I$ to simplify the constraints.)

2. **Pure rotational motion**
   In fact the rank condition on the matrix M_p is so fundamental that it works for the pure rotational motion case too. Please show that if there is no translation, the rank condition on the matrix M_p is equivalent to the constraints that we may get in the case of pure rotational motion

   $$\widehat{\mathbf{x}}_i R_i \mathbf{x}_1 = 0. \tag{8.25}$$

   (Hint: you need to convince yourself that in this case the rank of N_p is no more than $m + 2$.)

3. **Points in the plane at infinity**
   Show that the multiple view matrix M_p associated to m images of a point p in the plane at infinity satisfies the rank condition: $\text{rank}(M_p) \leq 1$. (Hint: the homogeneous coordinates for a point p at infinity are of the form $\mathbf{X} = [X, Y, Z, 0]^T \in \mathbb{R}^4$.)

Chapter 9

Geometry and reconstruction from line features

Our discussion so far has been based on the assumption that point correspondence is available in a number $m \geq 2$ of images. However, as we have seen in Chapter 4, image correspondence is subject to the aperture problem, that causes the correspondence to be undetermined along the direction of brightness edges. One may therefore consider matching edges, rather than individual points. In particular, man-made scenes are abundant with straight lines features that can be matched in different views using a variety of techniques. Therefore, assuming that we are given correspondence between lines in several images, how can we exploit them to recover camera pose as well as the 3-D structure of the environment? In order to answer this question, we need to study the geometry of line correspondences in multiple views. Fortunately, as we see in this chapter, most of the concept described in the last chapter for the case of points carry through with lines. Therefore, we follow the derivation in Chapter 8 to derive all the independent constraints between line correspondences.

9.1 Multiple views of a line

There are several equivalent ways to represent a line in space, L. In terms of homogeneous coordinates, the line L can be described as the subset of $\mathbb{R}^3$ which satisfies

$$L = \{\mathbf{X} \mid \mathbf{X} = \mathbf{X}_o + \mu v, \mu \in \mathbb{R}\} \quad \subset \mathbb{R}^3,$$

where $\mathbf{X}_o = [X_o, Y_o, Z_o, 1]^T \in \mathbb{R}^4$ is a base point on this line, $v = [v_1, v_2, v_3, 0]^T \in \mathbb{R}^4$ is a non-zero vector indicating the direction of the line, as shown in Figure 9.1. Hence, in homogeneous coordinates, we can specify the line by the pair of vectors $(\mathbf{X}_o, v)$. Also, the line L can be viewed as a 1-dimensional hyperplane in $\mathbb{R}^3$ and can be described as

$$L = \{\mathbf{X} \mid \bar{\Pi}\mathbf{X} = 0\} \quad \subset \mathbb{R}^3,$$

where $\bar{\Pi} \in \mathbb{R}^{2\times 4}$ is a 2×4 matrix of rank 2 and X_o, v are both in its kernel. To distinguish these different but equivalent representations, we call the later as the *co-representation*. Clearly, a co-representation specifies a geometric subspace by its orthogonal supplement. Notice that neither description is unique. For example, any point on the line L can be the base point $\mathbf{X}_o$; or one can choose $\bar{\Pi}$ to be an $n \times 4$ matrix as long as it is of rank 2 and has the same kernel.

Similarly, for an image line in the image plane $\mathbb{R}^2$, we can describe it in either way. In this chapter, for simplicity, we will mainly use the co-representation: an image point $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$ lies on an image line which is represented by a vector $\mathbf{l} = [a, b, c]^T \in \mathbb{R}^3$ if it satisfies

$$\mathbf{l}^T\mathbf{x} = 0.$$

This is illustrated in Figure 9.1. We will call $\mathbf{l}$ the *co-image* of the line

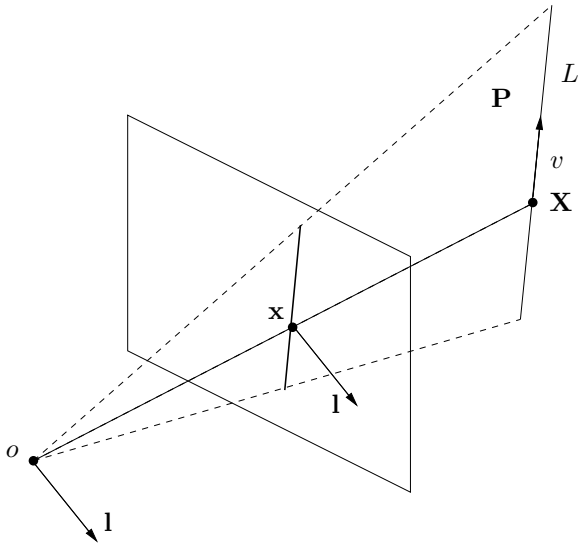

Figure 9.1. Image of a line L. The intersection of $\mathbf{P}$ and the image plane represents the physical projection of L. But it is more conviniently represented by a normal vector $\mathbf{l}$ of the plane $\mathbf{P}$.

L. We will discuss in more detail the definition of image and co-image in

Chapter 11. Using this notation, a co-image $\mathbf{l}(t) = [a(t), b(t), c(t)]^T \in \mathbb{R}^3$ of a $L \subset \mathbb{E}^3$ taken by a moving camera satisfies the following equation

$$\mathbf{l}(t)^T \mathbf{x}(t) = \mathbf{l}(t)^T A(t) P g(t) \mathbf{X} = 0, \tag{9.1}$$

where $\mathbf{x}(t)$ is the image of the point $\mathbf{X} \in L$ at time t, $A(t) \in SL(3)$ is the camera calibration matrix (at time t), $P = [I, 0] \in \mathbb{R}^{3\times 4}$ is the constant projection matrix and $g(t) \in SE(3)$ is the coordinate transformation from the world frame to the camera frame at time t. Note that $\mathbf{l}(t)$ is the normal vector of the plane formed by the optical center o and the line L (see Figure 9.1). Since the above equation holds for any point $\mathbf{X}$ on the line L, it yields

$$\mathbf{l}(t)^T A(t) P g(t) \mathbf{X}_o = \mathbf{l}(t)^T A(t) P g(t) v = 0. \tag{9.2}$$

In the above equations, all $\mathbf{x}, \mathbf{X}$ and g are in homogeneous representation.

In practice one only measures images at sample times $t_1, t_2, \ldots, t_m$. For simplicity we denote

$$\mathbf{l}_i = \mathbf{l}(t_i), \quad \Pi_i = A(t_i) P g(t_i). \tag{9.3}$$

The matrix Π_i is then a 3×4 matrix which relates the i^{th} image of the line L to its world coordinates $(\mathbf{X}_o, v)$ by

$$\boxed{\mathbf{l}_i^T \Pi_i \mathbf{X}_o = \mathbf{l}_i^T \Pi_i v = 0} \tag{9.4}$$

for $i = 1, \ldots, m$. In the above equations, everythin except $\mathbf{l}_i$'s is unknown. Solving for Π_i's and L (i.e. $(\mathbf{X}_o, v)$) simultaneously from these equations is extremely difficult. A natural way to simplify the task is to exploit the rank deficiency condition from the following equations

$$N_l \mathbf{X}_o = 0, \quad N_l v = 0, \tag{9.5}$$

where

$$N_l = \begin{bmatrix} \mathbf{l}_1^T \Pi_1 \\ \mathbf{l}_2^T \Pi_2 \\ \vdots \\ \mathbf{l}_m^T \Pi_m \end{bmatrix} \in \mathbb{R}^{m\times 4}. \tag{9.6}$$

Hence the rank of N_l must be

$$\boxed{\text{rank}(N_l) \leq 2.} \tag{9.7}$$

9.2 The multiple view matrix and its rank

Without loss of generality, we may assume that the first camera frame is chosen to be the reference frame. That gives the projection matrices $\Pi_i, i = 1, \ldots, m$ the general form

$$\Pi_1 = [I, 0], \quad \ldots, \quad \Pi_m = [R_m, T_m] \quad \in \mathbb{R}^{3\times 4}, \tag{9.8}$$

where, as before, $R_i \in \mathbb{R}^{3\times 3}, i = 2, \ldots, m$ is the first three columns of Π_i and $T_i \in \mathbb{R}^3, i = 2, \ldots, m$ is the fourth column of Π_i. Notice that if the camera is calibrated, R_i corresponds to the actual camera rotation and T_i to the translation. Now the matrix N_l becomes

$$N_l = \begin{bmatrix} \mathbf{l}_1^T & 0 \\ \mathbf{l}_2^T R_2 & \mathbf{l}_2^T T_2 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m & \mathbf{l}_m^T T_m \end{bmatrix} \in \mathbb{R}^{m\times 4}. \tag{9.9}$$

This matrix should have a rank of no more than 2. Multiplying N_l on the right by the following matrix

$$D_l = \begin{bmatrix} \widehat{\mathbf{l}_1} & \mathbf{l}_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 5} \tag{9.10}$$

yields

$$N_l' = \begin{bmatrix} 0 & \mathbf{l}_1^T \mathbf{l}_1 & 0 \\ \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} & \mathbf{l}_2^T R_2 \mathbf{l}_1 & \mathbf{l}_2^T T_2 \\ \vdots & \vdots & \vdots \\ \mathbf{l}_m^T R_m \widehat{\mathbf{l}_1} & \mathbf{l}_m^T R_m \mathbf{l}_1 & \mathbf{l}_m^T T_m \end{bmatrix} \in \mathbb{R}^{m\times 5}. \tag{9.11}$$

Then since D_l is of full rank 4, it yields

$$\operatorname{rank}(N_l') = \operatorname{rank}(N_l) \leq 2.$$

Obviously, this is true if and only if the following sub-matrix of N_l'

$$M_l = \begin{bmatrix} \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1} & \mathbf{l}_3^T T_3 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m \widehat{\mathbf{l}_1} & \mathbf{l}_m^T T_m \end{bmatrix} \in \mathbb{R}^{(m-1)\times 4} \tag{9.12}$$

has rank no more than one. We call the matrix M_l the *multiple view matrix* associated to a line feature L. Hence we have proved the following:

Theorem 9.1 (Rank deficiency equivalence condition). *For the two matrices N_l and M_l, we have*

$$\boxed{\textit{rank}(M_l) = \textit{rank}(N_l) - 1 \leq 1.} \tag{9.13}$$

Therefore rank(N_l) is either 2 or 1, depending on whether rank(M_l) is 1 or 0, respectively.

Comment 9.1 (Constraint on rotation from M_l). *One may notice that for the matrix M_l to be of rank 1, it is necessary that the first three columns are of rank 1. This imposes constraints on the camera rotation R_i's only.*

The rank condition certainly implies all trilinear constraints among the given m images of the line. To see this more explicitly, notice that for $\text{rank}(M_l) \leq 1$, it is necessary for any pair of row vectors of M_l to be linearly dependent. This gives us the well-known trilinear constraints

$$\boxed{\mathbf{l}_j^T T_j \mathbf{l}_i^T R_i \widehat{\mathbf{l}_1} - \mathbf{l}_i^T T_i \mathbf{l}_j^T R_j \widehat{\mathbf{l}_1} = 0} \tag{9.14}$$

among the first, i^{th} and j^{th} images. Hence the constraint $\text{rank}(M_l) \leq 1$ is a natural generalization of the trilinear constraint (for 3 views) to arbitrary m views since when $m = 3$ it is equivalent to the trilinear constraint for lines, except for some rare degenerate cases, e.g., $\mathbf{l}_i^T T_i = 0$ for some i.

It is easy to see from the rank of matrix M_l that there will be no more independent relationship among either pairwise or quadruple-wise image lines. Trilinear constraints are the only non-trivial ones for all the images that correspond to a single line in 3-D. So far we have essentially proved the following facts regarding multilinear constraints among multiple images of a line:

Theorem 9.2 (Linear relationships among multiple views of a line). *For any given m images corresponding to a line L in $\mathbb{E}^3$ relative to m camera frames, the rank-deficient matrix M_l implies that any algebraic constraints among the m images can be reduced to only those among 3 images at a time, characterized by the so-called trilinear constraints (9.14).*

Although trilinear constraints are necessary for the rank of matrix M_l (hence N_l) to be 1, they are *not* sufficient. For equation (9.14) to be non-trivial, it is required that the entry $\mathbf{l}_i^T T_i$ in the involved rows of M_l need to be non-zero. This is not always true for certain degenerate cases - such as the when line is parallel to the translational direction. The rank condition on M_l, therefore, provides a more complete account of the constraints among multiple images and it avoids artificial degeneracies that could be introduced by using algebraic equations.

9.3 Geometric interpretation of the rank condition

In the previous section, we have classified the algebraic constraints that may arise among m corresponding images of a line. We now know that the relationships among m images somehow boil down to those among 3 views at a time, characterized by the trilinear constraints (9.14). We here explain what these equations mean and give a simple intuitive geometric interpretation for all these algebraic relationships.

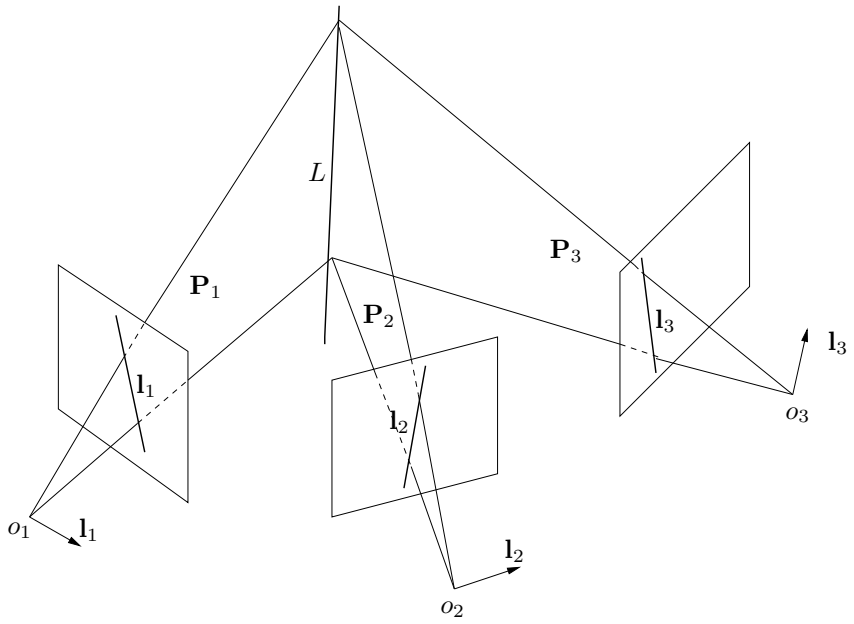

Figure 9.2. Three planes extended from the three images $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ intersect at one line L in 3-D, the pre-image of $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$.

9.3.1 Uniqueness of the pre-image

Given 3 vectors $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3 \in \mathbb{R}^3$, if they are indeed images of some line L in 3-D with respect to the three camera frames as shown in Figure 9.2, they should automatically satisfy the trilinear constraints, e.g.,

$$\mathbf{l}_2^T T_2 \mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1} - \mathbf{l}_3^T T_3 \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} = 0.$$

Like we did for points, we now ask the inverse question: if the three vectors $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ satisfy the trilinear constraints, are they necessarily images of some single line in 3-D, the so-called *pre-image*? As shown in Figure 9.3, we denote the planes formed by the optical center o_i of the i^{th} frame and image line $\mathbf{l}_i$ to be $\mathbf{P}_i$, $i = 1, 2, 3$. Denote the intersection line between $\mathbf{P}_1$ and $\mathbf{P}_2$ as L_2 and the intersection line between $\mathbf{P}_1$ and $\mathbf{P}_3$ as L_3. As pointed out at the beginning, $\mathbf{l}_i \in \mathbb{R}^3$ is also the normal vector of $\mathbf{P}_i$. Then without loss of generality, we can assume that $\mathbf{l}_i$ is the unit normal vector of plane $\mathbf{P}_i$, $i = 1, 2, 3$ and the trilinear constraint still holds. Thus, $-\mathbf{l}_i^T T_i = d_i$ is the distance from o_1 to the plane $\mathbf{P}_i$ and $(\mathbf{l}_i^T R_i)^T = R_i^T \mathbf{l}_i$ is the unit normal vector of $\mathbf{P}_i$ expressed in the 1^{th} frame. Furthermore, $(\mathbf{l}_i^T R_i \widehat{\mathbf{l}_1})^T$ is a vector parallel to L_i with length being $\sin(\theta_i)$, where $\theta_i \in [0, \pi]$ is the angle between the planes $\mathbf{P}_1$ and $\mathbf{P}_i$, $i = 2, 3$. Therefore, in the general case, the trilinear constraint implies two things: first, as $(\mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1})^T$ is linear dependent of $(\mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1})^T$, L_2 is parallel to L_3. Secondly, as $d_2 \sin(\theta_3) =$

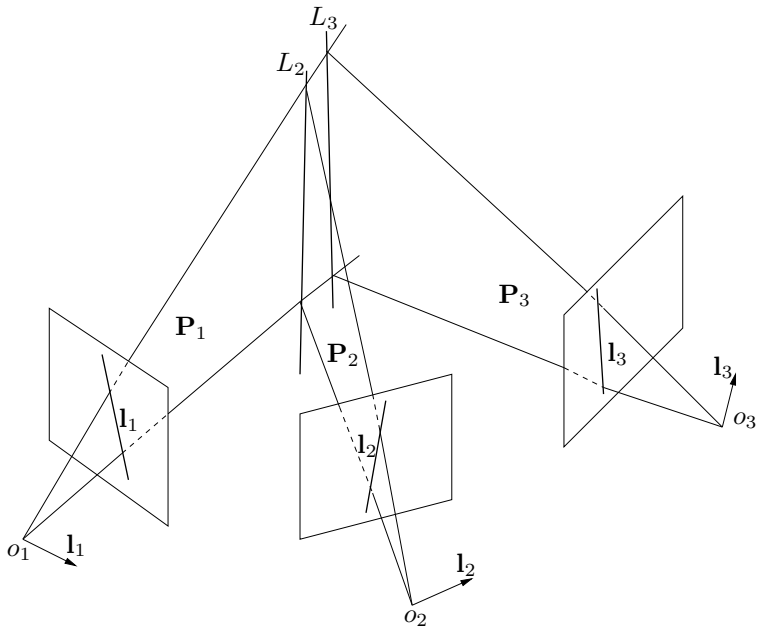

Figure 9.3. Three planes extended from the three images $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ intersect at lines L_2 and L_3, which actually coincides.

$d_3 \sin(\theta_2)$, the distance from o_1 to L_2 is the same with the distance from o_1 to L_3. They then imply that L_2 coincides with L_3, or in other words, the line L in 3-D space is uniquely determined. However, we have degeneracy when $\mathbf{P}_1$ coincides with $\mathbf{P}_2$ and $\mathbf{P}_3$. In that case, $d_2 = d_3 = 0$ and $(\mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1})^T = (\mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1})^T = 0_{3\times 1}$. There are infinite number of lines in $\mathbf{P}_1 = \mathbf{P}_2 = \mathbf{P}_3$ that generate the same set of images $\mathbf{l}_1$, $\mathbf{l}_2$ and $\mathbf{l}_3$. The case when $\mathbf{P}_1$ coincides with only $\mathbf{P}_2$ or only $\mathbf{P}_3$ is more tricky. For example, if $\mathbf{P}_1$ coincides with $\mathbf{P}_2$ but not with $\mathbf{P}_3$, then $d_2 = 0$ and $(\mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1})^T = 0_{3\times 1}$. However, if we re-index the images (frames), then we still can obtain a non-trivial trilinear constraint, from which L can be deduced as the intersection line between $\mathbf{P}_1$ and $\mathbf{P}_3$. So we have in fact proved the following fact:

Lemma 9.1 (Properties of trilinear constraints for lines). *Given three camera frames with distinct optical centers and any three vectors $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3 \in \mathbb{R}^3$ that represents three lines on each image plane. If the three image lines satisfy trilinear constraints*

$$\mathbf{l}_j^T T_{ji} \mathbf{l}_k^T R_{ki} \widehat{\mathbf{l}_i} - \mathbf{l}_k^T T_{ki} \mathbf{l}_j^T R_{ji} \widehat{\mathbf{l}_i} = 0, \quad i, j, k = 1, 2, 3$$

a unique pre-image L is determined except when the three planes defined respectively by the centers o_i's of the camera and the vectors $\mathbf{l}_i$'s as their

normals all coincide with each other. For this only degenerate case, the matrix M_l becomes zero. [1]

For more than 3 views, in order to check the uniqueness of the pre-image, one needs to apply the above lemma to every triplet of views. The possible combinations of degenerate cases make it very hard to draw any consistent conclusion. However, in terms of the rank condition on the multiple view matrix, the lemma can be generalized to multiple views in a much more concise and unified form:

Theorem 9.3 (Uniqueness of the pre-image). *Given m vectors in $\mathbb{R}^3$ representing lines on the image planes with respect to m camera frames, they correspond to the same line in the 3-D space if the rank of the matrix M_l relative to any of the camera frames is 1. If its rank is 0 (i.e. the matrix M_l is zero), then the line is determined up to a plane on which all the camera centers must lie.*

The proof follows directly from Theorem 9.1. So the case that the line in 3-D shares the same plane as all the centers of the camera frames is the only degenerate case when one will not be able to determine the exact 3-D location of the line from its multiple images. As long as the camera frames have distinct centers, the set of lines that are coplanar with these centers is of only zero measure. Hence, roughly speaking, trilinear constraints among images of a line rarely fail in practice. Even the rectilinear motion will not pose a problem as long as enough number of lines in 3-D are observed, the same as in the point case. On the other hand, the theorem also suggests a criteria to tell from the matrix M_l when a degenerate configuration is present: exactly when the largest singular value of the M_l matrix (with respect to any camera frame) is close to zero.

9.3.2 Geometry of the multiple view matrix

Now we can have better understanding of the geometric meaning of the matrix M_l. If $\mathbf{l}_1, \ldots, \mathbf{l}_m$ are the m images of one line L in m different views, and without loss of generality, $\mathbf{l}_i$'s are unit vectors, then $R_i^T \mathbf{l}_i$ is the normal vector of the plane $\mathbf{P}_i$ formed by L and the optical center o_i of the i^{th} frame expressed in the 1^{th} frame. With $-\mathbf{l}_i^T T_i$ being the distance from o_1 to $\mathbf{P}_i$, $[\mathbf{l}_i{}^T R_i \quad \mathbf{l}_i^T T_i] X = 0$, $X = [x, y, z, 1] \in \mathbb{R}^4$ is the function of plane $\mathbf{P}_i$. Besides, since $\mathbf{l}_1, R_2^T \mathbf{l}_2, \ldots, R_m^T \mathbf{l}_m$ are all perpendicular to L, they are coplanar. Hence $(\mathbf{l}_i^T R_i \widehat{\mathbf{l}_1})^T$ is parallel to the vector along the line L, i.e. $[\mathbf{l}_i^T R_i \widehat{\mathbf{l}_1}, 0]^T \in \mathbb{R}^4$ is proportional to the vector v which defines the line L in 3-D. Since M_l has rank 1, if we view each row of M_l as homogeneous coordinates of some point in 3-D, then all the rows in fact defines a unique

[1] Here we use subscripts ji to indicate that the related transformation is from the i^{th} frame to the j^{th}.

point in 3-D. This point is not necessarily on the image plane though. If we call this point p, then the vector defined by $p - o_1$ is obviously parallel to the original line L in 3-D. We also call it v. Therefore the so-defined M_l matrix in fact gives us a map from lines in 3-D to vectors in 3-D:

$$M_l : L \subset \mathbb{R}^3 \mapsto v \in \mathbb{R}^3.$$

This map is certainly not injective but surjective. That is, from M_l matrix alone, one will not be able to recover the exact 3-D location of the line L, but it will give us most of the information that we need to know about its images. Moreover, the vector gives the direction of the line, and the norm $\|v\|$ is exactly the ratio:

$$\|v\| = \sin(\theta_i)/d_i, \quad \forall i = 2, \ldots, m.$$

Roughly speaking, the farther away is the line L from o_1, the smaller this ratio is. In fact, one can show that the family of parallel lines in 3-D that all map to the same vector v form a cylinder centered around o_1. The above ratio is exactly the inverse of the radius of such a cylinder. We may summarize our discussion into the following theorem:

Theorem 9.4 (Geometry of the matrix M_l). *The matrix M_l associated to a line L in 3-D maps this line to a unique vector v in 3-D. This map is surjective and two lines are mapped to the same vector if and only if they are: 1. parallel to each other and 2. of the same distance to the center o of the reference camera frame. That distance is exactly $1/\|v\|$.*

Therefore, knowing M_l (i.e. the vector v), we know that L must lie on a circle of radius $r = 1/\|v\|$, as shown in Figure 9.4. So if we can obtain the M_l

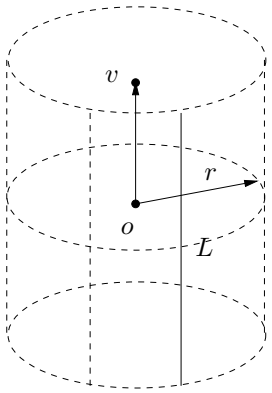

Figure 9.4. An equivalent family of parallel lines that give the same M_l matrix.

matrix for L with respect to another camera center, we get two families of parallel lines lying on two cylinders. In general, these two cylinders intersect at two lines, unless the camera centers all lie on a line parallel to the line

L. Hence, two M_l matrices (for the same line in 3-D) with respect to two distinct camera centers determine the line L up to two solutions. One would imagine that, in general, a third M_l matrix respect to a third camera center will then uniquely determine the 3-D location of the line L.

9.3.3 Relationships between rank conditions for line and point

It is interesting to explore the relationship between the rank conditions derived for points in Chapter 8 and the rank conditions for lines in this chapter. Let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be the m images of a point p in 3-D space, and $(R_i, T_i) \in \mathbb{R}^{3\times 4}$be the corresponding transformation from the i^{th} camera frame to the first, $i = 2, \ldots, m$. Denote

$$M_p = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & \widehat{\mathbf{x}_2} T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m} R_m \mathbf{x}_1 & \widehat{\mathbf{x}_m} T_m \end{bmatrix} \in \mathbb{R}^{3(m-1)\times 2}.$$

Then according to the rank condition for point in Chapter 8, we have

$$\text{rank}(M_p) \leq 1.$$

The apparent similarities between both the rank conditions and the forms of M_l and M_p are expected due to the geometric duality between lines and point. In the 3-D space, a point can be uniquely determined by two lines, while a line can be uniquely determined by two points. So our question now is that if the two set of rank conditions can be derived from each other based on the geometric duality.

First we show that we can derive the rank condition for line from rank condition for point. Let p_1 and p_2 be two distinct points on a line L in the 3-D space. Denote the m images of p_1, p_2 under m views to be $\mathbf{x}_1^1, \ldots, \mathbf{x}_m^1$ and $\mathbf{x}_1^2, \ldots, \mathbf{x}_m^2$, respectively. Hence, the i^{th} $(i = 1, \ldots, m)$ view of L can be expressed as $\mathbf{l}_i = \widehat{\mathbf{x}_i^2}\mathbf{x}_i^1$ or $\widehat{\mathbf{l}_i} = \mathbf{x}_i^2\mathbf{x}_i^{1T} - \mathbf{x}_i^1\mathbf{x}_i^{2T}$, $i = 1, \ldots, m$. From the rank deficiency of M_p^1 and M_p^2, we have $\widehat{\mathbf{x}_i^1} R_i \mathbf{x}_1^1 = \alpha \widehat{\mathbf{x}_i^1} T_i$ and $\widehat{\mathbf{x}_i^2} R_i \mathbf{x}_1^2 = \beta \widehat{\mathbf{x}_i^2} T_i$ for some $\alpha, \beta \in \mathbb{R}$ and $i = 1, \ldots, m$. This gives

$$\mathbf{l}_i^T R_i \widehat{\mathbf{l}_1} = -\mathbf{x}_i^{1T} \widehat{\mathbf{x}_i^2} T_i (\alpha \mathbf{x}_1^{2T} - \beta \mathbf{x}_1^{1T}), \quad \mathbf{l}_i^T T_i = -\mathbf{x}_i^{1T} \widehat{\mathbf{x}_i^2} T_i. \tag{9.15}$$

which means that each row of M_l is spanned by the same vector $[(\alpha \mathbf{x}_1^{2T} - \beta \mathbf{x}_1^{1T}), 1]^T \in \mathbb{R}^4$. Therefore,

$$\text{rank}(M_p^1) \leq 1 \;\&\; \text{rank}(M_p^2) \leq 1 \quad \Rightarrow \quad \text{rank}(M_l) \leq 1.$$

The inverse direction of this duality is not so straightforward. The reason is obvious: the duality is not totally symmetric. In the 3-D space, any distinct two points can determine a line. However, not any two lines may intersect at one point unless they are coplanar. Hence, in order to prove the inverse, an additional coplanar condition for the two lines in space should be imposed. This will be investigated in the next chapter. But the matrices M_p and M_l

already reveal some interesting *duality* between the camera center o and the point p in 3-D. For example, if the camera center moves on a straight line (the rectilinear motion), from the M_p matrix associated to a point p, the 3-D location of the point p can only be determined up to a circle. On the other hand, if the camera center is fixed but the point p can move on a straight line L, from the M_l matrix associated to the line L, the 3-D location of this line can only be determined up to a circle too. Mathematically speaking, matrices M_p and M_l define an equivalence relationship for points and lines in the 3-D space, respectively. They both group points and lines according to their distance to the center of the reference camera frame. Numerically, the sensitivity for M_p and M_l as rank-deficient matrices depends very much on such distance. Roughly speaking, the farther away is the point or line, the more sensitive the matrix is to noise or disturbance. Hence, in practice, one may view M_p and M_l as a natural "metric" for the quality of the measurements associated to the multiple images of a 3-D point or line.

9.4 Applications of the rank condition

The rank condition on the matrix M_l allows us to use all the constraints among multiple images of a line feature simultaneously without specifying a set of triple-wise frames. That is, it makes it possible to use all the data simultaneously for purposes such as feature matching or recovering camera configuration from multiple images of lines. Many natural algorithms are suggested by the simple form of the M_l matrix. These algorithms run in exact parallel as those outlined for the case of point features.

Ideally, one likes to formulate the entire problem of reconstructing 3-D motion and structure as one optimizing some global objective function[2] subject to the rank condition. However, such an approach usually relies on very difficult nonlinear optimization methods. Instead, we here divide the overall problem into a few subproblems: matching features assuming known motion, and estimating motion assuming known matched features. Sections 4.1 and 4.2 treat these two subproblems respectively.

9.4.1 Matching

In general we like to test if given m vectors $\mathbf{l}_1, \ldots, \mathbf{l}_m \in \mathbb{R}^3$ with known Π indeed satisfy all the constraints that m images of a single 3-D line (pre-image) should. There are two ways of performing this. One is based on the fact that M_l matrix has rank no more than 1. Another is based on the geometric interpretation of M_l matrix.

[2]Such as the *reprojection error* in image.

Since $\text{rank}(M_l) \leq 1$, the 4×4 matrix $M_l^T M_l$ also has rank no more than 1. Hence, ideally with given $\mathbf{l}_1, \ldots, \mathbf{l}_m$ and Π, the eigenvalues of $M_l^T M_l$ should satisfy $\lambda_1 \geq 0$, $\lambda_2 = \lambda_3 = \lambda_4 = 0$. A more numerically robust algorithm would be:

Algorithm 9.1 (Multiple view matching test). *Suppose the projection matrix Π associated to m camera frames are given. Then for given m vectors $\mathbf{l}_1, \ldots, \mathbf{l}_m \in \mathbb{R}^3$,*

1. *Compute the matrix $M_l \in \mathbb{R}^{m \times 4}$ according to (9.12).*
2. *Compute second largest eigenvalue λ_2 of the 4×4 matrix $M_l^T M_l$;*
3. *If $\lambda_2 \leq \epsilon_1$ for some pre-fixed threshold ϵ_1, then we say the m image vectors match. Otherwise discard it as outliers.*

9.4.2 Reconstruction

Now suppose that $m(\geq 3)$ images $\mathbf{l}_1^j, \ldots, \mathbf{l}_m^j$ of n lines L^j, $j = 1, \ldots, n$ are given and we want to use them to estimate the unknown projection matrix Π. From the rank condition of the M_l matrix, we know that the kernel of M_l should have 3 dimensions. Denote M_l^j to be the M_l matrix for the j^{th} line, $j = 1, \ldots, n$. Since ${\mathbf{l}_i^j}^T R_i \widehat{\mathbf{l}_1^j}$ is a vector parallel to L^j, then for any two linear independent vectors $u^j, w^j \in \mathbb{R}^3$ lying in the plane perpendicular to L^j, $\left[\begin{smallmatrix} u^j \\ 0 \end{smallmatrix}\right]$ and $\left[\begin{smallmatrix} w^j \\ 0 \end{smallmatrix}\right]$ are two base vectors in the kernel of M_l^j. Let $\left[\begin{smallmatrix} \alpha^j \\ 1 \end{smallmatrix}\right] \in \ker(M_l^j)$ be a vector orthogonal to both $\left[\begin{smallmatrix} u^j \\ 0 \end{smallmatrix}\right]$ and $\left[\begin{smallmatrix} w^j \\ 0 \end{smallmatrix}\right]$. Then

$$\alpha^j = [\alpha_1^j, \alpha_2^j, \alpha_3^j]^T = k^j \widehat{u^j} w^j, \tag{9.16}$$

for some $k^j \in \mathbb{R}$. It is direct to see that α^j is a vector parallel to the line L^j. It is in fact pointing to the opposite direction of the vector v^j associated to the line L^j and has a norm equal to the distance from the line L^j to the center of the reference camera frame. Thus for the i^{th} view, $i = 2, \ldots, m$, we define matrices:

$$P_i \doteq \begin{bmatrix} {\mathbf{l}_i^1}^T & {\alpha^1}^T(-\widehat{\mathbf{l}_1^1} * {\mathbf{l}_i^1}^T) \\ \vdots & \vdots \\ {\mathbf{l}_i^n}^T & {\alpha^n}^T(-\widehat{\mathbf{l}_1^n} * {\mathbf{l}_i^n}^T) \\ 0 & {u^1}^T(-\widehat{\mathbf{l}_1^1} * {\mathbf{l}_i^1}^T) \\ \vdots & \vdots \\ 0 & {u^n}^T(-\widehat{\mathbf{l}_1^n} * {\mathbf{l}_i^n}^T) \\ 0 & {w^1}^T(-\widehat{\mathbf{l}_1^1} * {\mathbf{l}_i^1}^T) \\ \vdots & \vdots \\ 0 & {w^n}^T(-\widehat{\mathbf{l}_1^n} * {\mathbf{l}_i^n}^T) \end{bmatrix} \in \mathbb{R}^{3n \times 12}, \tag{9.17}$$

where $*$ is the *Kronecker* product. Then we have

$$P_i \begin{bmatrix} \vec{T}_i \\ \vec{R}_i \end{bmatrix} = 0, \tag{9.18}$$

where $\vec{T}_i = T_i$ and $\vec{R}_i = [r_{11}, r_{21}, r_{31}, r_{12}, r_{22}, r_{32}, r_{13}, r_{23}, r_{33}]^T$, with r_{kl} being the $(kl)^{th}$ entry of R_i, $k, l = 1, 2, 3$. It can be shown that if α^j, u^j, w^j's are known, the matrix P_i is of rank 11 if more than $n \geq 12$ lines in general position are given. In that case, the kernel of P_i is unique, and so is (R_i, T_i). Hence if we know α^j for each $j = 1, \ldots, n$, then we can estimate (R_i, T_i) by performing singular value decomposition (SVD) on P_i. In practice, the initial estimations of α^j, u^j, w^j's may be noisy and only depend on local data, so we can use iteration method. First, we get estimates for (R_i, T_i)'s, then we use the estimated motions to re-calculate α^j's, and iterate in this way till the algorithm converges. However, currently there are several ways to update α^j, u^j, w^j's in each iteration. Initialization of α^j, u^j, w^j and the overall algorithm are described in the next section.

Euclidean reconstruction

Here we illustrate the overall algorithm for the case when the camera is assumed to be calibrated. That is $A(t) = I$ hence the projection matrix $\Pi_i = [R_i, T_i]$ represents actual Euclidean transformation between camera frames. We will discuss later on what happens if this assumption is violated.

Given the first three views of (at least) 12 lines in the 3-D space. Let M_{l3}^j be the matrix formed by the first 3 columns of M_l^j associated to the first three views. From M_{l3}^j, we can already estimate $u^j, w^j \in \ker(M_{l3}^j)$, where u^j, w^j are defined above. So after we obtain an initial estimation of $(\tilde{R}_i, \tilde{T}_i)$ for $i = 2, 3$, we can calculate $\tilde{M}_{l3}^j$ and compute the SVD of it such that $\tilde{M}_{l3}^j = U_2 S_2 V_2^T$, then pick $\tilde{u}^j, \tilde{w}^j$ being the 2^{nd} and 3^{rd} columns of V_2 respectively. Then the estimation for α^j is $\tilde{\alpha}^j = k^j(\tilde{u}^j \times \tilde{w}^j)$ for some $k^j \in \mathbb{R}$ to be determined. k can be estimated by least squares method such that

$$k^j = -\frac{\sum_{i=2}^m (\mathbf{l}_i^{j\,T} \tilde{T}_i)(\mathbf{l}_i^{j\,T} \tilde{R}_i \widehat{\mathbf{l}_1^j} \widehat{\tilde{u}^j} \tilde{w}^j)}{\sum_{i=2}^m (\mathbf{l}_i^{j\,T} \tilde{R}_i \widehat{\mathbf{l}_1^j} \widehat{\tilde{u}^j} \tilde{w}^j)^2}. \tag{9.19}$$

Using the estimated α^j, u^j, w^j, we can now compute the matrix P_i from (9.17). By performing SVD on P_i such that $P_i = USV^T$, we then pick the last column of V to obtain estimates respectively for T_i and R_i as $\bar{T}_i \in \mathbb{R}^3$ and $\bar{R}_i \in \mathbb{R}^{3\times 3}$ for $i = 2, \ldots, m$. Since $\bar{R}_i$ is not guaranteed to be a rotation matrix, in order to project $\bar{R}_i$ onto $SO(3)$, we can then do SVD on $\bar{R}_i$ such

that $\bar{R}_i = U_i S_i V_i^T$, so the estimates for $R_i \in SO(3)$ and $T_i \in \mathbb{R}^3$ are

$$\tilde{T}_i = \frac{\text{sign}(\det(U_i V_i^T))}{\sqrt[3]{\det(S_i)}} \bar{T}_i \in \mathbb{R}^3, \tag{9.20}$$

$$\tilde{R}_i = \text{sign}(\det(U_i V_i^T)) U_i V_i^T \in SO(3). \tag{9.21}$$

The α^j, u^j, w^j can then be re-estimated from the full matrix M_l^j computed from the motion estimates $(\tilde{R}_i, \tilde{T}_i)$.

Based on above arguments, we can summarize our algorithm as the following:

Algorithm 9.2 (Structure and motion from multiple views of lines). *Given a set of m images $\mathbf{l}_1^j, \ldots, \mathbf{l}_m^j$ of n lines L^j, $j = 1, \ldots, n$, we can estimate the motions (R_i, T_i), $i = 2, \ldots, m$ as the following:*

1. *Initialization*
   (a) *Set step counter $s = 0$.*
   (b) *Compute initial estimates for (R_i, T_i), $i = 2, 3$ for the first three views.*
   (c) *Compute initial estimates of α_s^j, u_s^j, w_s^j for $j = 1, \ldots, n$ from the first three views.*
2. *Compute P_i from (9.17) for $i = 2, \ldots, m$, and obtain $(\bar{R}_i, \bar{T}_i)$ from the eigenvector associated to its smallest singular value.*
3. *Compute $(\tilde{R}_i, \tilde{T}_i)$ from (9.20) and (9.21), $i = 2, \ldots, m$.*
4. *Compute $\alpha_{s+1}^j, u_{s+1}^j, w_{s+1}^j$ based on the M^j matrix calculated by using $(\tilde{R}_i, \tilde{T}_i)$. Normalize α_{s+1}^j so that $\|\alpha_{s+1}^1\| = 1$.*
5. *If $\|\alpha_{s+1} - \alpha_s\| < \epsilon$ for some threshold ϵ, then stop, else set $s = s + 1$, and goto 2.*

There are several notes for the above algorithm:

1. From (9.17) we can see that in order to get a unique (R_i, T_i) from SVD on P_i, we want the rank of P_i to be 11, this requires that we have at least 12 pairs of line correspondences.
2. There are several ways for the initial estimation of α^j's. There is an nonlinear algorithm for estimating trifocal tensor given by Hartley *et. al.* [HZ00]. Although linear algorithms for three views of line features also exist [WHA93], they usually require at least 13 lines matched across three views. In practice, one may instead use the two view 8 point algorithm to initialize the first three views.
3. The way that α^j is re-estimated from M_l^j is not unique. It can also be recovered from the rows of M_l^j, from the relationship between α^j and v mentioned above. The reason to set $\|\alpha_{s+1}^1\|$ to be 1 in Step 4

is to fix the universal scale. It is equivalent to putting the first line at a distance of 1 to the first camera center.

4. There is an interesting characteristic of the above algorithm: (R_i, T_i) seem to be estimated using pairwise views only. But it is not exactly true. The computation of the matrix P_i depends on all α^j, u^j, w^j each of which is estimated from the M_l^j matrix for all views.

Remark 9.1 (Euclidean, affine or projective reconstruction). *We must point out that, although it seems to be proposed for the calibrated camera (Euclidean) case only, the algorithm works just the same if the camera is weakly calibrated (affine) or uncalibrated (projective). The only change needed is at the initialization step. In order to initialize α^j's, either a Euclidean, affine or projective choice for (R_2, T_2) and (R_3, T_3) needs to be specified. This is where our knowledge of the camera calibration enters the picture. In the worst case, i.e. when we have no knowledge on the camera at all, any choice of a projective frame will do. In that case, the relative transformation among camera frames and the scene structure are only recovered up to a projective transformation of choice. Once that is done, the rest of the algorithm runs exactly the same.*

9.5 Experiments

In this section, we show in simulation the performance of the above algorithm. We test it on two different scenarios: sensitivity of motion and structure estimates with respect to the level of noise on the line measurements; and the effect of number of frames on motion and structure estimates.

9.5.1 Setup

The simulation parameters are as follows: number of trials is 500, number of feature lines is typically 20, number of frames is 4 (unless we vary it on purpose), field of view is 90^o, depth variation is from 100 to 400 units of focal length and image size is 500×500. Camera motions are specified by their translation and rotation axes. For example, between a pair of frames, the symbol XY means that the translation is along the X-axis and rotation is along the Y-axis. If n such symbols are connected by hyphens, it specifies a sequence of consecutive motions. The ratio of the magnitude of translation $||T||$ and rotation θ, or simply the T/R *ratio*, is compared at the center of the random cloud scattered in the truncated pyramid specified by the given field of view and depth variation. For each simulation, the amount of rotation between frames is given by a proper angle and the amount of translation is then automatically given by the T/R ratio. We

always choose the amount of total motion such that all feature (lines) will stay in the field of view for all frames. In all simulations, each image line is perturbed by independent noise with a standard deviation given in degrees. Error measure for rotation is $\arccos\left(\frac{tr(R\tilde{R}^T)-1}{2}\right)$ in degrees where $\tilde{R}$ is an estimate of the true R. Error measure for translation is the angle between T and $\tilde{T}$ in degrees where $\tilde{T}$ is an estimate of the true T. Error measure for structure is approximately measured by the angle between α and $\tilde{\alpha}$ where $\tilde{\alpha}$ is an estimate of the true α.

Algorithm 2 requires that we initialize with the first three views. In simulations below, we initialize the motion among the first three views using motion estimates from the linear algorithm for point features (see the previous chapter). The error in initial motion estimates correspond to an increasing level of noise on image points from 0 to 5 pixels. While the line measurements are perturbed by an increasing level of random angle from 0 to 2.5 degrees, the motion for the first three views are initialized with a corresponding level of noisy estimates. Of course, one may run existing algorithms based on trilinear constraint [WHA93] for line features and initialize the first three views. But motion estimation from line measurements alone is much more sensitive to degenerate configurations - any line coplanar with the translation gives essential no information on the camera motion nor its own 3-D structure. On the other hand, point features give more stable initial estimates.

9.5.2 Motion and structure from four frames

Figures 9.5 and 9.6 plot respectively the motion and structure estimate errors versus the level of noises added on the line measurements. The motion sequence is XX-YY-$X(XY)$, where (XY) means a direction half between the X-axis and Y-axis. Unlike point features, the quality of line feature measurements depends heavily on the camera motion. Among the 20 randomly generated 3-D lines, those which are coplanar with the translation will give little information on the camera motion or its 3-D location. These "bad" measurements typically contribute to a larger error in estimates. Their effect is quite noticeable in simulations and sometimes even causes numerical instability. By examining the multiple view matrix M_l associated to each line, we believe in the future we will be able to find good criteria to eliminate the "bad" lines hence improve the motion and structure estimates. Even so, the current algorithm is still able to converge to reasonably good estimates for the uninitialized motion between the fourth frame and the first.

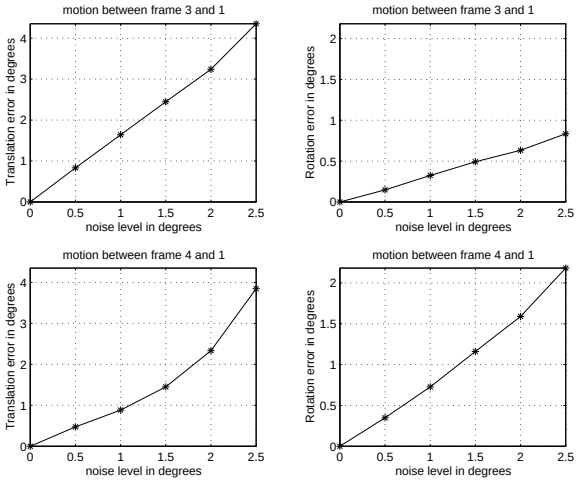

Figure 9.5. Motion estimate error from four views. The number of trials is 500. T/R ratio is 1. The amount of rotation between frames is 20^o.

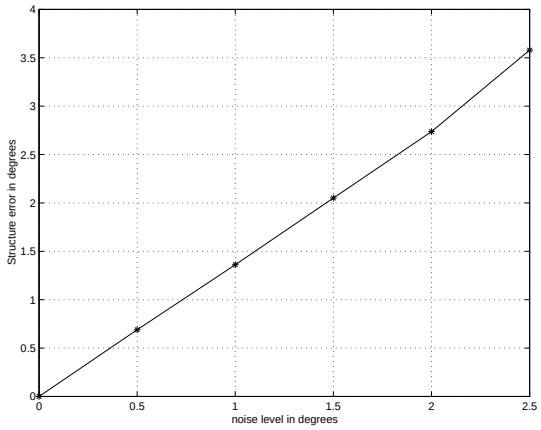

Figure 9.6. Structure estimate error from four views. The number of trials is 500. T/R ratio is 1. The amount of rotation between frames is 20^o.

9.5.3 Error as a function of number of frames

Figures 9.7 and 9.8 plot respectively the motion and structure estimate errors versus the number of frames. The noise level on the image line measurements is 1^o. The motion sequence is an orbital motion around the set of 20 random lines generated at a distance between 200 and 400 units of focal length away from the camera centers. The amount of rotation is incrementally 15^o between frames and a corresponding amount of translation is used to generate the orbital motion. From Figure 9.8, we see that the structure estimates improve while the number of frames increases, which is

expected. In fact, the error converges to the given level of noise on the line measurements. However, according to Figure 9.7, motion estimates do not necessarily improve with an increase number of frames since the number of motion parameters increase linearly with the number of frames. In fact, we see that after a few frames, additional frames will have little effect on the previous motion estimates.

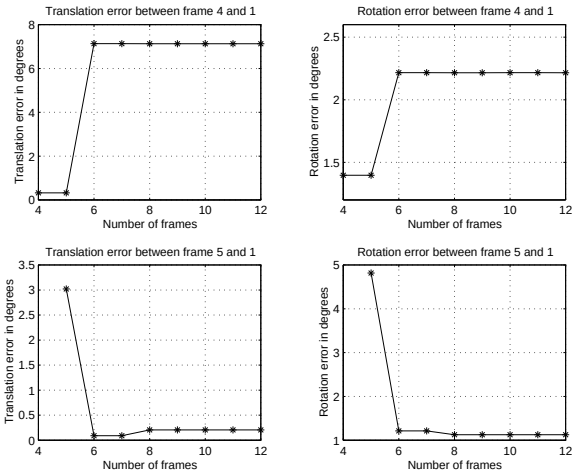

Figure 9.7. Motion (between frames 4-1 and 5-1) estimate error versus number of frames for an orbital motion. The number of trials is 500.

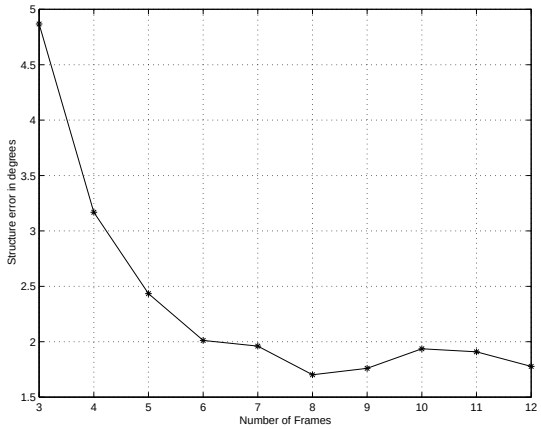

Figure 9.8. Structure estimate error versus number of frames for an orbital motion. The number of trials is 500.

Comment 9.2. *From our experience with the simulations, we want to mention a few things about the proposed linear algorithm for line features:*

- *Degenerate configuration is a more severe problem for line features than point features. Eliminating bad lines based on the multiple view matrix M_l will improve the estimate significantly.*
- *The rank 1 condition for the four column M_l is numerically less stable than that for the two column M_p for point features. Better numerical techniques for imposing the rank condition on M_l are worth investigating.*
- *The algorithm requires at least 12 lines which is twice as many as that required by the algorithm using point features. Increase the number of line features, motion estimates will improve but not the structure.*
- *In the future, we need to further investigate how these algorithms perform if a mixture of point and line features are used, or additional constraints such as coplanar are imposed on the features.*

9.6 Summary

9.7 Exercises

Chapter 10

Geometry and reconstruction with incidence relations

The previous two chapters described the constraints involving corresponding points or lines. In this chapter we address the case when correspondence of points *and* lines is available. This results in a formal condition that has all rank conditions discussed in previous chapters as special cases. We also discuss how to enforce the constraint that all geometric features lie on a plane, which is a special case of considerable practical importance.

10.1 Image and co-image of a point and line

Let $\mathbb{E}^3$ denote the three-dimensional Euclidean space and $p \in \mathbb{E}^3$ denote a point in the space. The homogeneous coordinates of p relative to a fixed world coordinate frame are denoted as $\mathbf{X} \doteq [X, Y, Z, 1]^T \in \mathbb{R}^4$. From previous chapters, we know that the (perspective) image $\mathbf{x}(t) \doteq [x(t), y(t), z(t)]^T \in \mathbb{R}^3$ of p, taken by a moving camera at time t satisfies the following relationship:

$$\lambda(t)\mathbf{x}(t) = A(t)Pg(t)\mathbf{X}. \tag{10.1}$$

Now suppose that p is lying on a straight line $L \subset \mathbb{E}^3$, as shown in Figure 10.1. We know that the line L can be defined by a collection of points in $\mathbb{E}^3$ that can be described (in homogeneous coordinates) as:

$$L \doteq \{\mathbf{X} \mid \mathbf{X} = \mathbf{X}_o + \mu v,\ \mu \in \mathbb{R}\} \quad \subset \mathbb{R}^4, \tag{10.2}$$

where $\mathbf{X}_o = [X_o, Y_o, Z_o, 1]^T \in \mathbb{R}^4$ are coordinates of a base point p_o on this line and $v = [v_1, v_2, v_3, 0]^T \in \mathbb{R}^4$ is a non-zero vector indicating the

direction of the line. Then the *image* of the line L at time t is simply the collection of images $\mathbf{x}(t)$ of all points $p \in L$. It is clear that all such $\mathbf{x}(t)$'s span a plane in $\mathbb{R}^3$, as shown in Figure 10.1. The projection of the line is simply the intersection of this plane with the image plane. Usually it is more convenient to specify a plane by its normal vector, denoted as $\mathbf{l}(t) = [a(t), b(t), c(t)]^T \in \mathbb{R}^3$. We call this vector $\mathbf{l}$ the *co-image* of the line L, which satisfies the following equation:

$$\mathbf{l}(t)^T \mathbf{x}(t) = \mathbf{l}(t)^T A(t) P g(t) \mathbf{X} = 0 \tag{10.3}$$

for any image $\mathbf{x}(t)$ of any point p on the line L. Notice that the column (or row) vectors of the matrix $\widehat{\mathbf{l}}$ span the (physical) image of the line L, i.e. they span the plane which is *orthogonal* to $\mathbf{l}$.[1] This is illustrated in Figure 10.1. Similarly, if $\mathbf{x}$ is the image of a point p, its co-image (the plane orthogonal to $\mathbf{x}$) is given by the matrix $\widehat{\mathbf{x}}$. So in this chapter, we will use the following notation:

$$\begin{array}{llll} \text{Image of a point:} & \mathbf{x} \in \mathbb{R}^3, & \text{Co-image of a point:} & \widehat{\mathbf{x}} \in \mathbb{R}^{3\times 3}, \\ \text{Image of a line:} & \widehat{\mathbf{l}} \in \mathbb{R}^{3\times 3}, & \text{Co-image of a line:} & \mathbf{l} \in \mathbb{R}^3. \end{array} \tag{10.4}$$

Notice that we always have $\widehat{\mathbf{x}} \cdot \mathbf{x} = 0$ and $\widehat{\mathbf{l}} \cdot \mathbf{l} = 0$. Since image and co-image are equivalent representation of the same geometric entity, sometimes for simplicity (and by abuse of language) we may simply refer to either one as "image" if its meaning is clear from the context.

If sampled images of $\mathbf{x}(t)$ or $\mathbf{l}(t)$ are given at some time instances: $t_1, t_2, \ldots, t_m$, for simplicity we denote:

$$\lambda_i = \lambda(t_i), \quad \mathbf{x}_i = \mathbf{x}(t_i), \quad \mathbf{l}_i = \mathbf{l}(t_i), \quad \Pi_i = A(t_i) P g(t_i). \tag{10.5}$$

As before, the matrix Π_i is the projection matrix relative to the i^{th} camera frame. The matrix $\Pi_i \in \mathbb{R}^{3\times 4}$ then relates the i^{th} image of the point p to its world coordinates $\mathbf{X}$ by:

$$\lambda_i \mathbf{x}_i = \Pi_i \mathbf{X} \tag{10.6}$$

and the i^{th} co-image of the line L to its world coordinates $(\mathbf{X}_o, v)$ by:

$$\mathbf{l}_i^T \Pi_i \mathbf{X}_o = \mathbf{l}_i^T \Pi_i v = 0, \tag{10.7}$$

for $i = 1, \ldots, m$. If the point is actually lying on the line, we further have a relationship between the image of the point and the co-image of the line:

$$\mathbf{l}_i^T \mathbf{x}_i = 0, \tag{10.8}$$

for $i = 1, \ldots, m$.

In the above systems of equations, the unknowns, λ_i's, $\mathbf{X}$, $\mathbf{X}_o$ and v, which encode the information about 3-D location of the point p or the line

[1] In fact, there is some redundancy using $\widehat{\mathbf{l}}$ to describe the plane: the three column (or row) vectors in $\widehat{\mathbf{l}}$ are not linearly independent. They only span a two-dimensional space. However, we here use it anyway to simplify the notation.

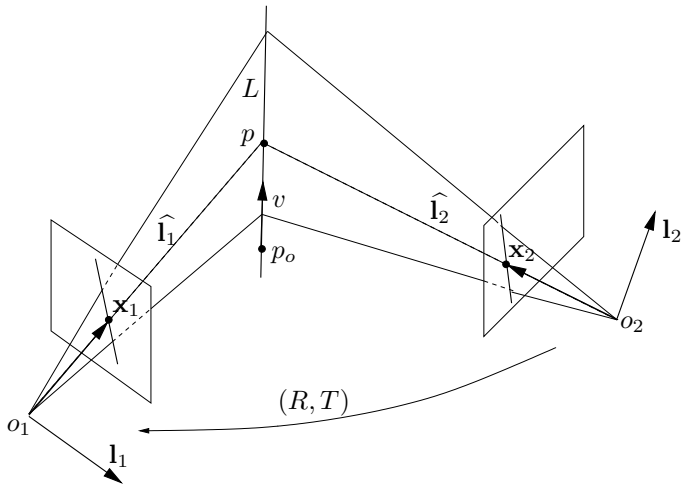

Figure 10.1. Images of a point p on a line L. Planes extended from the image lines $\widehat{\mathbf{l}_1}, \widehat{\mathbf{l}_2}$ should intersect at the line L in 3-D. Lines extended from the image points $\mathbf{x}_1, \mathbf{x}_2$ intersect at the point p.

L in $\mathbb{R}^3$ are not intrinsically available from images. By eliminating these unknowns from the equations we obtain the remaining intrinsic relationships between $\mathbf{x}, \mathbf{l}$ and Π only, i.e. between the image measurements and the camera configuration. Of course there are many different, but algebraically equivalent ways for elimination of these unknowns. This has in fact resulted in different kinds (or forms) of multilinear (or multifocal) constraints that exist in the computer vision literature. We here introduce a more *systematic* way of eliminating *all* the above unknowns that results in a *complete* set of conditions and a clear geometric characterization of *all* constraints.

10.2 Rank conditions for various incidence relations

As we have seen in the previous chapters, multiple images of a point or a line in $\mathbb{E}^3$ are governed by certain rank conditions. Such conditions not only concisely capture geometric constraints among multiple images, but also are the key to further reconstruction of the camera motion and scene structure. In this section, we will show that all basic incidence relationships among different features, i.e. *inclusion*, *intersection* or *restriction* of features, can also be fully captured by similar rank conditions. Since these relationships can be easily detected or verified in each image, such knowledge can be and should be exploited if a consistent reconstruction is sought.

10.2.1 Inclusion of features

Consider the situation when you observe a line $\mathbf{l}_1$ in the reference view but in remaining views, you observe images $\mathbf{x}_2, \ldots, \mathbf{x}_m$ of a feature point on the line – we say that this feature point is *included* by the line. To derive the constraints that such features have to satisfy, we start with the matrix N_p (which was introduced in Chapter 8) and left multiply it by the following matrix:

$$D_l' = \begin{bmatrix} \mathbf{l}_1^T & 0 \\ \widehat{\mathbf{l}_1} & 0 \\ 0 & I_{3(m-1)\times 3(m-1)} \end{bmatrix} \in \mathbb{R}^{(3m+1)\times 3m}. \tag{10.9}$$

We obtain:

$$D_l' N_p = \begin{bmatrix} \mathbf{l}_1^T & 0 & 0 & 0 & \cdots & 0 \\ \widehat{\mathbf{l}_1} & 0 & \widehat{\mathbf{l}_1}\mathbf{x}_1 & 0 & \cdots & 0 \\ R_2 & T_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ R_m & T_m & 0 & \cdots & 0 & \mathbf{x}_m \end{bmatrix} \in \mathbb{R}^{(3m+1)\times(m+4)}. \tag{10.10}$$

Since $\text{rank}(D_l') = 3m$, we have $\text{rank}(N_p) = \text{rank}(D_l' N_p) \leq m + 3$. Now left multiply $D_l' N_p$ by the the following matrix:

$$D_p' = \begin{bmatrix} I_{4\times 4} & 0 & 0 & \cdots & 0 \\ 0 & \widehat{\mathbf{x}_2} & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2^T & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & \widehat{\mathbf{x}_m} \\ 0 & \cdots & 0 & 0 & \mathbf{x}_m^T \end{bmatrix} \in \mathbb{R}^{(4m+1)\times(3m+1)}. \tag{10.11}$$

It is direct to verify that the rank of the resulting matrix $D_p' D_l' N_p$ is related to the rank of its sub-matrix:

$$N_p'' = \begin{bmatrix} \mathbf{l}_1^T & 0 \\ \widehat{\mathbf{x}_2} R_2 & \widehat{\mathbf{x}_2} T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m} R_m & \widehat{\mathbf{x}_m} T_m \end{bmatrix} \in \mathbb{R}^{(3m-2)\times 4} \tag{10.12}$$

by the expression $\text{rank}(N_p'') + m \leq \text{rank}(D_p' D_l' N_p) = \text{rank}(N_p)$. Now right multiplying N_p'' by:

$$\begin{bmatrix} \mathbf{l}_1 & \widehat{\mathbf{l}_1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 5} \tag{10.13}$$

yields:

$$\begin{bmatrix} \mathbf{l}_1^T\mathbf{l}_1 & 0 & 0 \\ \widehat{\mathbf{x}_2}R_2\mathbf{l}_1 & \widehat{\mathbf{x}_2}R_2\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2}T_2 \\ \vdots & \vdots & \vdots \\ \widehat{\mathbf{x}_m}R_m\mathbf{l}_1 & \widehat{\mathbf{x}_m}R_m\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_m}T_m \end{bmatrix} \in \mathbb{R}^{(3m-2)\times 5}. \tag{10.14}$$

We call its sub-matrix:

$$M_{lp} \doteq \begin{bmatrix} \widehat{\mathbf{x}_2}R_2\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2}T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m}R_m\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_m}T_m \end{bmatrix} \in \mathbb{R}^{[3(m-1)]\times 4} \tag{10.15}$$

the *multiple view matrix* for a point included by a line. Its rank is related to that of N_p by the expression:

$$\text{rank}(M_{lp}) \leq \text{rank}(N_p) - (m+1) \leq 2. \tag{10.16}$$

Since $\text{rank}(AB) \geq (\text{rank}(A) + \text{rank}(B) - n)$ for all $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{n\times k}$, we have $\text{rank}(\widehat{\mathbf{x}}_i R_i \widehat{\mathbf{l}_1}) \geq 1$. So we essentially have proven the following:

Lemma 10.1 (Rank condition with inclusion). *For multiple images of a point p on a line L, the multiple view matrix M_{lp} defined above satisfies*

$$\boxed{1 \leq \textit{rank}(M_{lp}) \leq \textit{rank}(N_p) - (m+1) \leq 2.} \tag{10.17}$$

The rank condition on M_{lp} then captures the incidence condition in which a line with co-image $\mathbf{l}_1$ includes a point p in $\mathbb{E}^3$ with images $\mathbf{x}_2, \ldots, \mathbf{x}_m$ with respect to $m-1$ camera frames. What kind of equations does this rank condition give rise to? Without loss of generality, let us look at the sub-matrix

$$M_{lp} = \begin{bmatrix} \widehat{\mathbf{x}_2}R_2\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2}T_2 \\ \widehat{\mathbf{x}_3}R_3\widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_3}T_3 \end{bmatrix} \in \mathbb{R}^{6\times 4}. \tag{10.18}$$

The rank condition implies that every 3×3 sub-matrix of M_{lp} has determinant zero. The first three rows are automatically of rank 2 and so are the last three rows and the first three columns.[2] Hence any non-trivial determinant consists of two and only two rows from either the first three or last three rows, and consists of two and only two columns from the first three columns. For example, if we choose two rows from the first three, it is direct to see that such a determinant is a polynomial of degree 5 on (the entries of) $\mathbf{l}_1, \mathbf{x}_2$ and $\mathbf{x}_3$ which is quadratic on $\mathbf{l}_1$ and also quadratic in

[2] This is due to the redundancy of using $\widehat{u}$ for the orthogonal supplement of $u \in \mathbb{R}^3$.

$\mathbf{x}_2$. The resulting equation is *not* multilinear in these vectors at all,[3] but it indeed imposes non-trivial constraints among these images (or co-images). The study of this type of constraints has been completely ignored in the past. In fact, if we have four images (in M_{lp}), one may take one row from the 2nd, 3rd and 4th images. The resulting equation would involve all four images. It is however not the "quadrilinear constraint" since the equation is always quadratic in $\mathbf{l}_1$. We may summarize the above discussion as follows:

Corollary 10.1 (Multiple view nonlinear constraints). *Given $m \geq 4$ images, multilinear constraints exist only up to triple-wise images. However, nonlinear constraints exist among triple-wise and quadruple-wise images.*

10.2.2 Intersection of features

Consider the situation in which, in the reference, you observe the image $\mathbf{x}_1$ of a point p, but in the remaining views, you observe co-images $\mathbf{l}_2, \mathbf{l}_3, \ldots, \mathbf{l}_m$ of lines that *intersect* at the point – in this case these co-images do not have to correspond to the same line in $\mathbb{E}^3$. There are many different, but equivalent ways to derive the constraints that such set of features has to satisfy. For simplicity, we here start with the matrix M_p (which was introduced in Chapter 8):

$$M_p = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \mathbf{x}_1 & \widehat{\mathbf{x}_3} T_3 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m} R_m \mathbf{x}_1 & \widehat{\mathbf{x}_m} T_m \end{bmatrix} \in \mathbb{R}^{[3(m-1)]\times 2}. \tag{10.19}$$

This matrix should have a rank of no more than 1. Since the point belongs to all the lines, we have

$$\mathbf{l}_i^T \mathbf{x}_i = 0, i = 1, \ldots, m.$$

Hence $\mathbf{l}_i \in \text{range}(\widehat{\mathbf{x}}_i)$. That is, there exist $u_i \in \mathbb{R}^3$ such that $\mathbf{l}_i = \widehat{\mathbf{x}}_i^T u_i, i = 1, \ldots, m$. Since $\text{rank}(M_p) \leq 1$, so should be the rank of following matrix (whose rows are simply linear combinations of those of M_p):

$$\begin{bmatrix} u_2^T \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & u_2^T \widehat{\mathbf{x}_2} T_2 \\ \vdots & \vdots \\ u_m^T \widehat{\mathbf{x}_m} R_m \mathbf{x}_1 & u_m^T \widehat{\mathbf{x}_m} T_m \end{bmatrix} = \begin{bmatrix} \mathbf{l}_2^T R_2 \mathbf{x}_1 & \mathbf{l}_2^T T_2 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m \mathbf{x}_1 & \mathbf{l}_m^T T_m \end{bmatrix} \in \mathbb{R}^{(m-1)\times 2}. \tag{10.20}$$

[3] Hence it is a constraint that is *not* in any of the multilinear (or multifocal) constraint lists previously studied in the computer vision literature. For a list of these multilinear constraints, see [HZ00].

We call the matrix

$$M_{pl} \doteq \begin{bmatrix} \mathbf{l}_2^T R_2 \mathbf{x}_1 & \mathbf{l}_2^T T_2 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m \mathbf{x}_1 & \mathbf{l}_m^T T_m \end{bmatrix} \quad \in \mathbb{R}^{(m-1)\times 2} \tag{10.21}$$

the *multiple view matrix* for lines intersecting at a point. Then we have proved:

Lemma 10.2 (Rank condition with intersection). *Given the image of a point p and multiple images of lines intersecting at p, the multiple view matrix M_{pl} defined above satisfies:*

$$\boxed{0 \leq \mathit{rank}(M_{pl}) \leq 1.} \tag{10.22}$$

The above rank condition on the matrix M_{pl} captures the incidence condition between a point and lines which intersect at the same point. It is worth noting that for the rank condition to be true, it is necessary that all 2×2 minors of M_{pl} be zero, i.e. the following constraints hold among arbitrary triplets of given images:

$$[\mathbf{l}_i^T R_i \mathbf{x}_1][\mathbf{l}_j^T T_j] - [\mathbf{l}_i^T T_i][\mathbf{l}_j^T R_j \mathbf{x}_1] = 0 \quad \in \mathbb{R}, \quad i, j = 2, \ldots, m. \tag{10.23}$$

These are exactly the well-known *point-line-line* relationships among three views [HZ00]. However, here $\mathbf{l}_i$ and $\mathbf{l}_j$ do not have to be co-images of the same line in $\mathbb{E}^3$. Their pre-images only have to intersect at the same point p. This undoubtly relaxed the restriction on the meaning of "corresponding" line features. Hence our results have extended the use of the point-line-line relationship. The rank of the above matrix M_{pl} is bounded by 1 is because a point image feature $\mathbf{x}_1$ of p is chosen in the first image. If we relax it to an image of a line that intersects at p, then we get a matrix which is similar to M_l defined in the previous chapter:

$$\tilde{M}_l \doteq \begin{bmatrix} \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} & \mathbf{l}_2^T T_2 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m \widehat{\mathbf{l}_1} & \mathbf{l}_m^T T_m \end{bmatrix} \quad \in \mathbb{R}^{(m-1)\times 4}. \tag{10.24}$$

However, here $\mathbf{l}_i$ in the i^{th} view can be the co-image of *any* line out of a family which intersect at a point p. It is then easy to derive from Lemma 10.1 and the proof of Lemma 10.2 that:

Corollary 10.2 (An intersecting family of lines). *Given m imasge of a family lines intersecting at a 3D point p, the matrix $\tilde{M}_l$ defined above satisfies:*

$$\boxed{1 \leq \mathit{rank}(\tilde{M}_l) \leq 2.} \tag{10.25}$$

We leave the proof as exercise to the reader. The reader also should be aware of the difference between this corollary and the Theorem 9.1: Here $\mathbf{l}_1, \ldots, \mathbf{l}_m$ do not have to be the coimages of a single 3D line, but instead each of them may randomly correspond to any line in an intersecting family. It is easy to see that the above rank condition imposes non-trivial constraints among up to four images, which conforms to what Corollary 10.1 says.

10.2.3 Features restricted to a plane

Another incidence condition commonly encountered in practice is that all features involved are actually lying on a plane, say $\mathbf{P}$, in $\mathbb{E}^3$. In general, a plane can be described by a vector $\pi = [a, b, c, d] \in \mathbb{R}^4$ such that the homogeneous coordinates $\mathbf{X}$ of any point p on this plane satisfy the equation:

$$\pi \mathbf{X} = 0. \tag{10.26}$$

Although we assume such a constraint on the coordinates $\mathbf{X}$ of p, in general we do not have to assume that we know π in advance.

Similarly, consider a line $L = \{\mathbf{X} \mid \mathbf{X} = \mathbf{X}_o + \mu v, \mu \in \mathbb{R}\}$. Then this line is in the plane $\mathbf{P}$ if and only if:

$$\pi \mathbf{X}_o = \pi v = 0. \tag{10.27}$$

For convenience, given a plane $\pi = [a, b, c, d]$, we usually define $\pi^1 = [a, b, c] \in \mathbb{R}^3$ and $\pi^2 = d \in \mathbb{R}$.

It turns out that, in order to take into account the planar restriction, we only need to change the definition of each multiple view matrix slightly, but all the rank conditions remain exactly the same. This is because in order to combine equations (10.6) and (10.7) with the planar constraints (10.26) and (10.27), we only need to change the definition of the matrices N_p and N_l to:

$$N_p \doteq \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & \cdots & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \Pi_m & 0 & \cdots & 0 & \mathbf{x}_m \\ \pi & 0 & \cdots & \cdots & 0 \end{bmatrix} \quad \text{and} \quad N_l \doteq \begin{bmatrix} \mathbf{l}_1^T \Pi_1 \\ \mathbf{l}_2^T \Pi_2 \\ \vdots \\ \mathbf{l}_m^T \Pi_m \\ \pi \end{bmatrix}.$$

Such modifications do not change the rank of N_p or N_l. Therefore, as before, we have $\text{rank}(N_p) \leq m + 3$ and $\text{rank}(N_l) \leq 2$. Then one can easily follow the previous proofs for all the rank conditions by carrying this extra row of (planar) constraint with the matrices and the rank conditions on the resulting multiple view matrices remain the same as before. We leave this as exercise for the reader (see Exercise ??). We here summarize the results as follows:

Corollary 10.3 (Rank conditions for coplanar features). *Given a point p and a line L lying on a plane $\mathbf{P}$ which is specified by the vector $\pi \in \mathbb{R}^4$, in Theorem 8.1 and Lemma 10.2, append the matrix $[\pi^1\mathbf{x}_1\ \pi^2]$ to the matrices M_p and M_{pl} respectively; in Theorem 9.1 and Lemma 10.1, append the matrix $[\pi^1\widehat{\mathbf{l}_1}\ \pi^2]$ to the matrices M_l and M_{lp} respectively. Then the rank conditions on the new matrices M_p, M_l, M_{lp} and M_{pl} remain the same as in Theorems 8.1, 9.1, and Lemmas 10.1 and 10.2.*

For example, the multiple view matrices M_p and M_l become:

$$M_p = \begin{bmatrix} \widehat{\mathbf{x}_2}R_2\mathbf{x}_1 & \widehat{\mathbf{x}_2}T_2 \\ \widehat{\mathbf{x}_3}R_3\mathbf{x}_1 & \widehat{\mathbf{x}_3}T_3 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m}R_m\mathbf{x}_1 & \widehat{\mathbf{x}_m}T_m \\ \pi^1\mathbf{x}_1 & \pi^2 \end{bmatrix}, \quad M_l = \begin{bmatrix} \mathbf{l}_2^T R_2\widehat{\mathbf{l}_1} & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3\widehat{\mathbf{l}_1} & \mathbf{l}_3^T T_3 \\ \vdots & \vdots \\ \mathbf{l}_m^T R_m\widehat{\mathbf{l}_1} & \mathbf{l}_m^T T_m \\ \pi^1\widehat{\mathbf{l}_1} & \pi^2 \end{bmatrix}. \tag{10.28}$$

Then the rank condition $\text{rank}(M_p) \leq 1$ implies not only the multilinear constraints as before, but also the following equations (by considering the sub-matrix consisting of the i^{th} group of three rows of M_p and its last row)

$$\widehat{\mathbf{x}_i}T_i\pi^1\mathbf{x}_1 - \widehat{\mathbf{x}_i}R_i\mathbf{x}_1\pi^2 = 0, \quad i = 2, \ldots, m. \tag{10.29}$$

When the plane $\mathbf{P}$ does not cross the camera center o_1, i.e. $\pi^2 \neq 0$, these equations give exactly the well-known *homography* constraints for planar image feature points

$$\widehat{\mathbf{x}_i}\left(R_i - \frac{1}{\pi^2}T_i\pi^1\right)\mathbf{x}_1 = 0 \tag{10.30}$$

between the 1^{st} and the i^{th} views. The matrix $H_i = \left(R_i - \frac{1}{\pi^2}T_i\pi^1\right)$ in the equation is the well-known *homography matrix* between the two views. Similarly from the rank condition on M_l, we can obtain homography in terms of line features

$$\mathbf{l}_i^T\left(R_i - \frac{1}{\pi^2}T_i\pi^1\right)\widehat{\mathbf{l}_1} = 0 \tag{10.31}$$

between the 1^{st} and the i^{th} views.

We know that on a plane $\mathbf{P}$, any two points determine a line and any two lines determine a point. This dual relationship is inherited in the following relationship between the rank conditions on M_p and M_l:

Corollary 10.4 (Duality between coplanar points and lines). *If the M_p matrices of two distinct points on a plane are of rank less than or equal to 1, then the M_l matrix associated to the line determined by the two points is of rank less than or equal to 1. On the other hand, if the M_l matrices of two distinct lines on a plane are of rank less than or equal to 1, then the M_p matrix associated to the intersection of the two lines is of rank less than or equal to 1.*

The proof is left as exercise (see Exercise ??). An immediate implication of this corollary is that given a set of feature points sharing the same 3-D plane, it really does not matter too much whether one uses the M_p matrices for points, or the M_l matrices for lines determined by pairwise points (in all the views). They essentially give exactly the same set of constraints. This is illustrated in Figure 10.2.

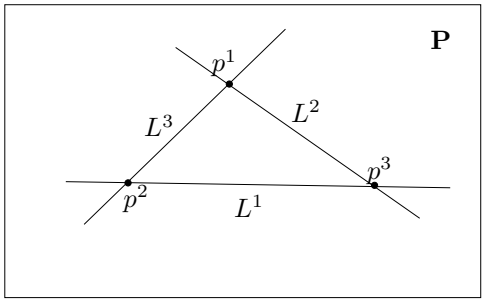

Figure 10.2. Duality between a set of three points and three lines in a plane $\mathbf{P}$: the rank conditions associated to p^1, p^2, p^3 are exactly equivalent those associated to L^1, L^2, L^3.

The above approach for expressing a planar restriction relies explicitly on the parameters π of the underlying plane $\mathbf{P}$ (which leads to the homography). There is however another intrinsic (but equivalent) way to express the planar restriction, by using combinations of the rank conditions that we have so far on point and line features. Since three points are always coplanar, at least four points are needed to make any planar restriction non-trivial. Suppose four feature points $p^1, \ldots, p^4$ are coplanar as shown in Figure 10.3, and their images are denoted as $\mathbf{x}^1, \ldots, \mathbf{x}^4$. The (virtual)

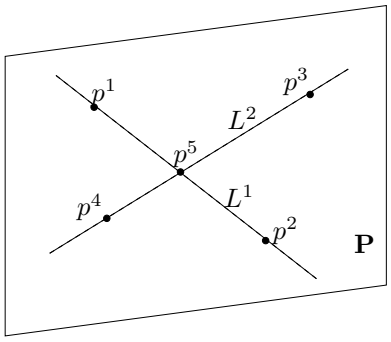

Figure 10.3. $p^1, \ldots, p^4$ are four points on the same plane $\mathbf{P}$, if and only if the two associated (virtual) lines L^1 and L^2 intersect at a (virtual) point p^5.

co-images $\mathbf{l}^1, \mathbf{l}^2$ of the two lines L^1, L^2 and the (virtual) image $\mathbf{x}^5$ of their

intersection p^5 can be uniquely determined by $\mathbf{x}^1, \ldots, \mathbf{x}^4$:

$$\mathbf{l}^1 = \widehat{\mathbf{x}^1}\mathbf{x}^2, \quad \mathbf{l}^2 = \widehat{\mathbf{x}^3}\mathbf{x}^4, \quad \mathbf{x}^5 = \widehat{\mathbf{l}^1}\mathbf{l}^2. \tag{10.32}$$

Then the coplanar constraint for $p^1, \ldots, p^4$ can be expressed in terms of the intersection relation between L^1, L^2 and p^5. If we use $\mathbf{l}_i^j$ to denote the i^{th} image of the line j, for $j = 1, 2$, and $i = 1, \ldots, m$, and $\mathbf{x}_i^j$ is defined similarly. According to the preceding Section 10.2.2, the coplanar constraint is equivalent to the following matrix:

$$\tilde{M}_{pl} = \begin{bmatrix} \mathbf{l}_2^{1T} R_2 \mathbf{x}_1^5 & \mathbf{l}_2^{1T} T_2 \\ \mathbf{l}_2^{2T} R_2 \mathbf{x}_1^5 & \mathbf{l}_2^{2T} T_2 \\ \vdots & \vdots \\ \mathbf{l}_m^{1T} R_m \mathbf{x}_1^5 & \mathbf{l}_m^{1T} T_m \\ \mathbf{l}_m^{2T} R_m \mathbf{x}_1^5 & \mathbf{l}_m^{2T} T_m \end{bmatrix} \in \mathbb{R}^{[2(m-1)]\times 2} \tag{10.33}$$

satisfying $\text{rank}(\tilde{M}_{pl}) \leq 1$ condition. Note that, unlike (10.29), this condition does not explicitly depends on the parameters π. We can get 2 more vitual points in a similar way from the given four coplanar points – from different pairs of virtual lines formed by these four points. Hence having (images of) 4 coplanar points is equivalent to having (images of) a total of 4 (real) + 3 (virtual) = 7 points. Algebraic equations that one may get from above rank conditions associated to all these points are essentially equivalent to the homography equations (10.29) (with π eliminated using the images of all four coplanar points). These equations, again, will not be multilinear in the given images $\mathbf{x}_i^j$, $j = 1, \ldots, 4, i = 1, \ldots, m$. Instead, they are typically *quadratic* in each given $\mathbf{x}_i^j$. Despite that, we here see again the effectiveness of using rank conditions to *intrinsically* express incidence relations in a unified fashion.

10.3 Rank conditions on the universal multiple view matrix

In preceding sections, we have seen that incidence conditions among multiple images of multiple features can usually be concisely expressed in terms of certain rank conditions on various types of the so-called multiple view matrix. In this section, we will demonstrate that all these conditions are simply special instantiations of a unified rank condition on a universal multiple view matrix.

For m images $\mathbf{x}_1, \ldots, \mathbf{x}_m$ of a point p on a line L with its m co-images $\mathbf{l}_1, \ldots, \mathbf{l}_m$, we define the following set of matrices:

$$\begin{aligned} D_i &\doteq \mathbf{x}_i \in \mathbb{R}^3 \quad \text{or} \quad \widehat{\mathbf{l}_i} \in \mathbb{R}^{3\times 3}, \\ D_i^\perp &\doteq \widehat{\mathbf{x}_i} \in \mathbb{R}^{3\times 3} \quad \text{or} \quad \mathbf{l}_i^T \in \mathbb{R}^3. \end{aligned}$$

Then, depending on whether the available (or chosen) measurement from the i^{th} image is the point feature $\mathbf{x}_i$ or the line feature $\mathbf{l}_i$, the D_i (or $D_i^\perp$) matrix is assigned its corresponding value. That choice is completely independent of the other D_j (or $D_j^\perp$) for $j \neq i$. The "dual" matrix $D_i^\perp$ can be viewed as the *orthogonal supplement* to D_i and it always represents a co-image (of a point or a line).[4] Using the above definition of D_i and $D_i^\perp$, we now formally define a *universal multiple view matrix:*

$$M \doteq \begin{bmatrix} D_2^\perp R_2 D_1 & D_2^\perp T_2 \\ D_3^\perp R_3 D_1 & D_3^\perp T_3 \\ \vdots & \vdots \\ D_m^\perp R_m D_1 & D_m^\perp T_m \end{bmatrix}. \tag{10.34}$$

Depending on the particular choice for each $D_i^\perp$ or D_1, the dimension of the matrix M may vary. But no matter what the choice for each individual $D_i^\perp$ or D_1 is, M will always be a valid matrix of certain dimension. Then after elimination of the unknowns λ_i's, $\mathbf{X}$, $\mathbf{X}_o$ and v in the system of equations in (10.6) and (10.7), we obtain:

Theorem 10.1 (Multiple view rank conditions). *Consider a point p lying on a line L and its images $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^3$ and co-images $\mathbf{l}_1, \ldots, \mathbf{l}_m \in \mathbb{R}^3$ relative to m camera frames whose relative configuration is given by (R_i, T_i) for $i = 2, \ldots, m$. Then for any choice of $D_i^\perp$ and D_1 in the definition of the multiple view matrix M, the rank of the resulting M belongs to and only to one of the following two cases:*

1. *If $D_1 = \widehat{\mathbf{l}_1}$ and $D_i^\perp = \widehat{\mathbf{x}_i}$ for some $i \geq 2$, then:*

$$\boxed{1 \leq \mathit{rank}(M) \leq 2.} \tag{10.35}$$

2. *Otherwise:*

$$\boxed{0 \leq \mathit{rank}(M) \leq 1.} \tag{10.36}$$

A complete proof of this theorem is a straight-forward combination and extension of Theorems 8.1, 9.1, Lemmas 10.1 and 10.2. Essentially, the above theorem gives a universal description of the incidence relation between a point and line in terms of their m images seen from m vantage points.

As a result of Theorem 10.1, all previously known and some additional unknown constraints among multiple images of point or line features are simply certain *instantiations* of the rank conditions of Theorem 10.1. The instantiations corresponding to case 2 are exactly the ones that give rise to

[4] In fact, there are many equivalent matrix representations for D_i and $D_i^\perp$. We choose $\widehat{\mathbf{x}_i}$ and $\widehat{\mathbf{l}_i}$ here because they are the simplest forms representing the orthogonal subspaces of $\mathbf{x}_i$ and $\mathbf{l}_i$ and also linear in $\mathbf{x}_i$ and $\mathbf{l}_i$ respectively.

the multilinear constraints in the literature. The instantiations corresponding to case 1, as we have seen before, give rise to constraints that are *not* necessarily multilinear. The completeness of Theorem 10.1 also implies that there would be *no* multilinear relationship among quadruple-wise views, even in the mixed feature scenario.[5] Therefore, quadrilinear constraints and quadrilinear tensors are clearly *redundant* for multiple view analysis. However, as we mentioned before (in Section 10.2.1), nonlinear constraints may still exist up to four views.

As we have demonstrated in the previous sections, other incidence conditions such as all features belonging to a plane in $\mathbb{E}^3$ can also be expressed in terms of the same set of rank conditions:

Corollary 10.5 (Planar features and homography). *Suppose that all features are in a plane and coordinates* $\mathbf{X}$ *of any point on it satisfy the equation* $\pi\mathbf{X} = 0$ *for some vector* $\pi^T \in \mathbb{R}^4$. *Denote* $\pi = [\pi^1, \pi^2]$ *with* ${\pi^1}^T \in \mathbb{R}^3, \pi^2 \in \mathbb{R}$. *Then simply append the matrix*

$$\begin{bmatrix} \pi^1 D_1 & \pi^2 \end{bmatrix} \tag{10.37}$$

to the matrix M *in its formal definition (10.34). The rank condition on the new* M *remains exactly the same as in Theorem 10.1.*

The rank condition on the new matrix M then implies *all* constraints among multiple images of these planar features, as well as the special constraint previously known as homography. Of course, the above representation is not intrinsic – it depends on parameters π that describe the 3-D location of the plane. Following the process in Section 10.2.3, the above corollary reduces to rank conditions on matrices of the type in (10.33), which in turn, give multi-quadratic constraints on the images involved.

Remark 10.1 (Features at infinity). *In Theorem 10.1, if the point* p *and the line* L *are in the plane at infinity* $\mathbb{P}^3 \setminus \mathbb{E}^3$, *the rank condition on the multiple view matrix* M *is just* the same. *Hence the rank condition extends to multiple view geometry of the entire projective space* $\mathbb{P}^3$, *and it does not discriminate between Euclidean, affine or projective space. In fact, the rank conditions are invariant under a much larger group of transformations: It allows any transformation that preserves all the incidence relations among a given set of features, these transformations* do not *even have to be linear.*

Remark 10.2 (Occlusion). *If any feature is occluded in a particular view, the corresponding row (or a group of rows) is simply omitted from* M*; or if only the point is occluded but not the entire line(s) on which the point lies, then simply replace the missing image of the point by the corresponding*

[5] In fact, this is quite expected: While the rank condition geometrically corresponds to the incidence condition that lines intersect at a point and that planes intersect at a line, incidence condition that three-dimensional subspaces intersect at a plane is a void condition in $\mathbb{E}^3$.

image(s) of the line(s). In either case, the overall rank condition on M remains unaffected*. In fact, the rank condition on M gives a very effective criterion to tell whether or not a set of (mixed) features indeed corresponds one to another. If the features are miss-matched, either due to occlusion or to errors while establishing correspondences, the rank condition will be violated.*

For clarity, we demonstrate using the following examples how to obtain these different types of constraints by instantiating M:

Example 10.1 (Point-point-point constraints). *Let us choose $D_1 = \mathbf{x}_1, D_2^\perp = \widehat{\mathbf{x}_2}, D_3^\perp = \widehat{\mathbf{x}_3}$. Then we get a multiple view matrix:*

$$M = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \mathbf{x}_1 & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \mathbf{x}_1 & \widehat{\mathbf{x}_3} T_3 \end{bmatrix} \in \mathbb{R}^{6\times 2}. \tag{10.38}$$

Then rank$(M) \leq 1$ gives:

$$[\widehat{\mathbf{x}_2} R_2 \mathbf{x}_1][\widehat{\mathbf{x}_3} T_3]^T - [\widehat{\mathbf{x}_3} R_3 \mathbf{x}_1][\widehat{\mathbf{x}_2} T_2]^T = 0 \quad \in \mathbb{R}^{3\times 3}. \tag{10.39}$$

The above equation give the point-point-point type of constraints on three images.

Example 10.2 (Line-line-line constraints). *Let us choose $D_1 = \widehat{\mathbf{l}_1}, D_2^\perp = \mathbf{l}_2^T, D_3^\perp = \mathbf{l}_3^T$. Then we get a multiple view matrix:*

$$M = \begin{bmatrix} \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1} & \mathbf{l}_3^T T_3 \end{bmatrix} \in \mathbb{R}^{2\times 4}. \tag{10.40}$$

Then rank$(M) \leq 1$ gives:

$$\left[\mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1}\right][\mathbf{l}_3^T T_3] - \left[\mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1}\right][\mathbf{l}_2^T T_2] = 0 \quad \in \mathbb{R}^3. \tag{10.41}$$

The above equation gives the line-line-line type of constraints on three images.

Example 10.3 (Point-line-line constraints). *Let us choose $D_1 = \mathbf{x}_1, D_2^\perp = \mathbf{l}_2^T, D_3^\perp = \mathbf{l}_3^T$. Then we get a multiple view matrix:*

$$M = \begin{bmatrix} \mathbf{l}_2^T R_2 \mathbf{x}_1 & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3 \mathbf{x}_1 & \mathbf{l}_3^T T_3 \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{10.42}$$

Then rank$(M) \leq 1$ gives

$$[\mathbf{l}_2^T R_2 \mathbf{x}_1][\mathbf{l}_3^T T_3] - [\mathbf{l}_3^T R_3 \mathbf{x}_1][\mathbf{l}_2^T T_2] = 0 \quad \in \mathbb{R}. \tag{10.43}$$

The above equation gives the point-line-line type of constraints on three images.

Example 10.4 (Line-point-point constraints). *Let us choose* $D_1 = \widehat{\mathbf{l}_1}, D_2^\perp = \widehat{\mathbf{x}_2}, D_3^\perp = \widehat{\mathbf{x}_3}$. *Then we get a multiple view matrix:*

$$M = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_3} T_3 \end{bmatrix} \in \mathbb{R}^{6\times 4}. \tag{10.44}$$

Then $\mathrm{rank}(M) \leq 2$ *implies that all* 3×3 *sub-matrices of* M *have determinant zero. These equations give the line-point-point type of constraints on three images.*

Similarly, other choices in $D_i^\perp$ and D_1 will give all possible types of constraints among any views of point and line features arbitrarily mixed.

10.4 Geometric interpretation of the rank conditions

Since there are practically infinitely many possible instantiations for the multiple view matrix, it is impossible to provide a geometric description for each one of them. Instead, we will discuss a few essential cases that will give the reader a clear idea about how the rank condition of the multiple view matrix works geometrically. In particular, we will demonstrate that a further drop of rank in the multiple view matrix M, can be clearly interpreted in terms of a corresponding geometric degeneracy. Understanding these representative cases would be sufficient for the reader to carry out a similar analysis to any other instantiation.

10.4.1 Case 2: $0 \leq \mathrm{rank}(M) \leq 1$

Let us first consider the more general case, i.e. case 2 in Theorem 10.1, when $\mathrm{rank}(M) \leq 1$. We will discuss case 1 afterwards. For case 2, there are only two interesting sub-cases, depending on the value of the rank of M, are:

$$a)\ \mathrm{rank}(M) = 1, \quad \text{and} \quad b)\ \mathrm{rank}(M) = 0. \tag{10.45}$$

Case a), when rank of M is 1, corresponds to the generic case for which, regardless of the particular choice of features in M, all these features satisfy the incidence condition. More explicitly, all the point features (if at least 2 are present in M) come from a unique 3-D point p, all the lines features (if at least 3 are present in M) come from a unique 3-D line L, and if both point and line features are present, the point p then must lie on the line L in 3-D. This is illustrated in Figure 10.4.

What happens if there are not enough point or line features present in M? If, for example, there is only one point feature $\mathbf{x}_1$ present in M_{pl}, then the rank of M_{pl} being 1 implies that the line L is uniquely determined by

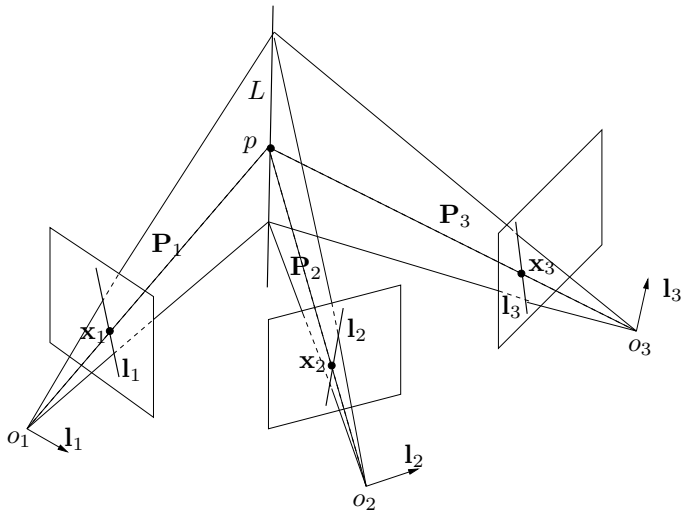

Figure 10.4. Generic configuration for the case rank$(M) = 1$. Planes extended from the (co-)images $\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3$ intersect at one line L in 3-D. Lines extended from the images $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ intersect at one point p. p must lie on L.

$\mathbf{l}_2, \ldots, \mathbf{l}_m$. Hence, the point p is determined by L and $\mathbf{x}_1$. On the other hand, if there is only one line feature present in some M, but more than two point features, L is then a family of lines lying on a plane and passing through the point p determined by the rest of point features in M.

Case *b*), when the rank of M is 0, implies that all the entries of M are zero. It is easy to verify that this corresponds to a set of degenerate cases in which the 3-D location of the point or the line cannot be uniquely determined from their multiple images (no matter how many), and the incidence condition between the point p and the line L no longer holds. In these cases, the best we can do is:

- When there are more than two point features present in M, the 3-D location of the point p can be determined up to a line which connects all camera centers (related to these point features);

- When there are more than three line features present in M, the 3-D location of the line L can be determined up to the plane on which all related camera centers must lie;

- When both point and line features are present in M, we can usually determine the point p up to a line (connecting all camera centers related to the point features) which lies on the same plane on which the rest of the camera centers (related to the line features) and the line L must lie.

Let us demonstrate this last case on a concrete example. Suppose the number of views is $m = 6$ and we choose the matrix M to be:

$$M = \begin{bmatrix} \mathbf{l}_2^T R_2 \mathbf{x}_1 & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3 \mathbf{x}_1 & \mathbf{l}_3^T T_3 \\ \mathbf{l}_4^T R_4 \mathbf{x}_1 & \mathbf{l}_4^T T_4 \\ \widehat{\mathbf{x}_5} R_5 \mathbf{x}_1 & \widehat{\mathbf{x}_5} T_5 \\ \widehat{\mathbf{x}_6} R_6 \mathbf{x}_1 & \widehat{\mathbf{x}_6} T_6 \end{bmatrix} \in \mathbb{R}^{9\times 2}. \tag{10.46}$$

The geometric configuration of the point and line features corresponding to the condition $\text{rank}(M) = 0$ is illustrated in Figure 10.5. But notice that, among all the possible solutions for L and p, if they both happen to be at infinity, the incidence condition then would hold for all the images involved.

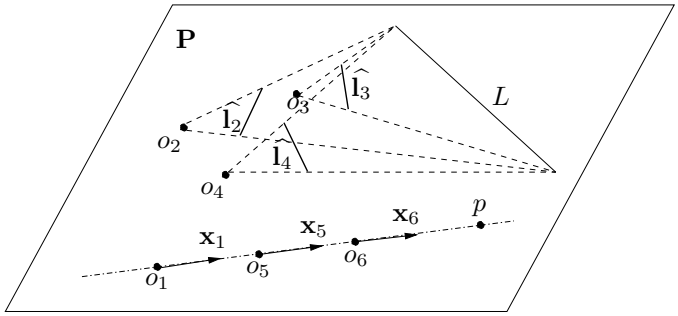

Figure 10.5. A degenerate configuration for the case $\text{rank}(M) = 0$: a point-line-line-line-point-point scenario. From the given rank condition, the line L could be any where on the plane spanned by all the camera centers; the point p could be any where on the line through o_1, o_5, o_6.

10.4.2 Case 1: $1 \leq \textit{rank}(M) \leq 2$

We now discuss case 1 in Theorem 10.1, when $\text{rank}(M) \leq 2$. In this case, the matrix M must contain at least one sub-matrix of the type:

$$\begin{bmatrix} \widehat{\mathbf{x}}_i R_i \widehat{\mathbf{l}}_1 & \widehat{\mathbf{x}}_i T_i \end{bmatrix} \in \mathbb{R}^{3\times 4}, \tag{10.47}$$

for some $i \geq 2$. It is easy to verify that such a sub-matrix can never be zero, hence the only possible values for the rank of M are:

$$a)\ \text{rank}(M) = 2, \quad \text{and} \quad b)\ \text{rank}(M) = 1. \tag{10.48}$$

Case a), when the rank of M is 2, corresponds to the generic cases for which the incidence condition among the features is effective. The essential

example here is the matrix M_{lp} given in (10.15):

$$M_{lp} = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2} T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m} R_m \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_m} T_m \end{bmatrix} \in \mathbb{R}^{[3(m-1)]\times 4}. \tag{10.49}$$

If $\text{rank}(M_{lp}) = 2$, it can be shown that the point p is only determined up to the plane specified by o_1 and $\mathbf{l}_1$ but all the point features $\mathbf{x}_2, \ldots, \mathbf{x}_m$ correspond to the same point p. The line L is only determined up to this plane, but the point p does not have to be on this line. This is illustrated in Figure 10.6. Beyond M_{lp}, if there are more than two line features present

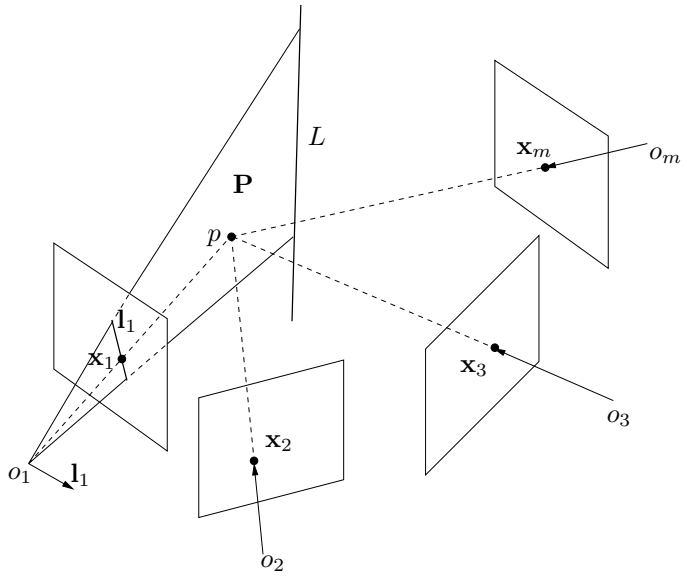

Figure 10.6. Generic configuration for the case $\text{rank}(M_{lp}) = 2$.

in some M, the point p then must lie on every plane associated to every line feature. Hence p must be on the intersection of these planes. Notice that, even in this case, adding more rows of line features to M will not be enough to uniquely determine L in 3-D. This is because the incidence condition for multiple line features requires that the rank of the associated matrix M_l be 1.[6] If we only require rank 2 for the overall matrix M, the line can be determined only up to a family of lines – intersections of the planes associated to all the line features – which all should intersect at the same point p.

[6]Matrix M_l is obtained from M by extracting the line measurements only.

Case b), when the rank of M is 1, corresponds to a set of degenerate cases for which the incidence relationship between the point p and the line L will be violated. For example, it is direct to show that M_{lp} is of rank 1 if and only if all the vectors $R_i^{-1}\mathbf{x}_i, i = 2, \ldots, m$ are parallel to each other and they are all orthogonal to $\mathbf{l}_1$, and $R_i^{-1}T_i, i = 2, \ldots, m$ are also orthogonal to $\mathbf{l}_1$. That means all the camera centers lie on the same plane specified by o_1 and $\mathbf{l}_1$ and all the images $\mathbf{x}_2, \ldots, \mathbf{x}_m$ (transformed to the reference camera frame) lie on the same plane and are parallel to each other. For example, suppose that $m = 5$ and choose M to be:

$$M = \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_3} T_3 \\ \widehat{\mathbf{x}_4} R_4 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_4} T_4 \\ \mathbf{l}_5^T R_5 \widehat{\mathbf{l}_1} & \mathbf{l}_5^T T_5 \end{bmatrix} \in \mathbb{R}^{10\times 4}. \tag{10.50}$$

The geometric configuration of the point and line features corresponding to the condition $\text{rank}(M) = 1$ is illustrated in Figure 10.7.

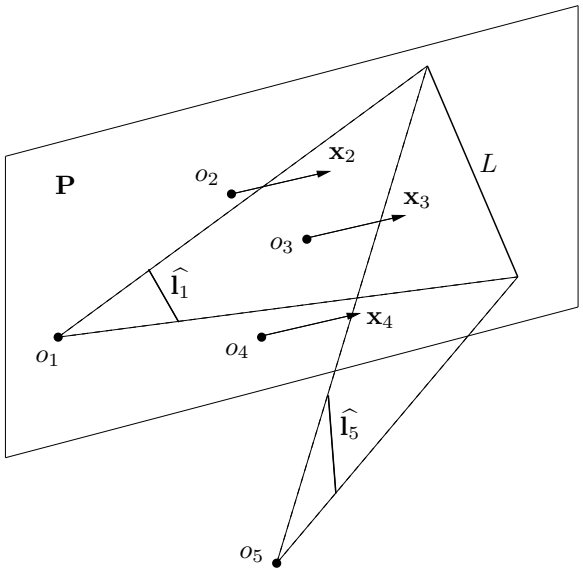

Figure 10.7. A degenerate configuration for the case $\text{rank}(M) = 1$: a line-point-point-point-line scenario.

Notice that in this case, we no longer have an incidence condition for the point features. However, one can view them as if they intersected at a point p at infinity. In general, we no longer have the incidence condition between the point p and the line L, unless both the point p and line L are in the plane at infinity in the first place. But since the rank condition is effective for line features, the incidence condition for all the line features still holds.

To summarize the above discussion, we see that the rank conditions indeed allow us to carry out meaningful global geometric analysis on the relationship among multiple point and line features for arbitrarily many views. There is no doubt that this extends existing methods based on multifocal tensors that can only be used for analyzing up to three views at a time. Since there is yet no systematic way to extend triple-wise analysis to multiple views, the multiple view matrix seems to be a more natural tool for multiple-view analysis. Notice that the rank conditions implies all previously known multilinear constraints, but multilinear constraints do not necessarily imply the rank conditions. This is because the use of algebraic equations may introduce certain artificial degeneracies that make a global analysis much more complicated and sometimes even intractable. On the other hand, the rank conditions have no problem in characterizing all the geometrically meaningful degeneracies in a multiple-view mixed-feature scenario. All the degenerate cases simply correspond to a further drop of rank for the multiple view matrix.

10.5 Applications of the rank conditions

The unified formulation of the constraints among multiple images in terms of the rank conditions, allows us to tackle many problems in multiple view geometry *globally*. More specifically, one can use these rank conditions to test whether or not a given set of features (points or lines) indeed satisfy certain incidence relations in $\mathbb{E}^3$. For example, whether a given set of point and line features indeed *correspond* one to another depends on whether all associated rank conditions are satisfied. The rank values can be used not only to detect outliers but also degenerate configurations. The rank conditions also allow us to *transfer* multiple features to a new view by using all features and incidence conditions among them simultaneously, without resorting to the 3-D structure of the scene. Hence the multiple view matrix provides the geometric basis for any *view synthesis* or *image based rendering* techniques. Most importantly, as we will demonstrate in the following section, the rank conditions naturally suggest new ways of formulating the problem of *structure from motion* recovery from multiple views.

For simplicity, we here only demonstrate the use of the theory on the structure from motion recovery problem. Essentially, the mathematical problem associated to structure from motion is how to find the camera configuration (R_i, T_i)'s such that all the rank conditions hold simultaneously for a given set of features, including both point and line features. Incidence relations among the given points and lines can now be explicitly taken into account when a global and consistent recovery of motion and structure takes place. To demonstrate how the multiple view matrix natu-

rally suggests a solution, let us consider image of a cube as shown in Figure 10.8. Let the j^{th} corner p^j be the intersection of the three edges L^{1j}, L^{2j}

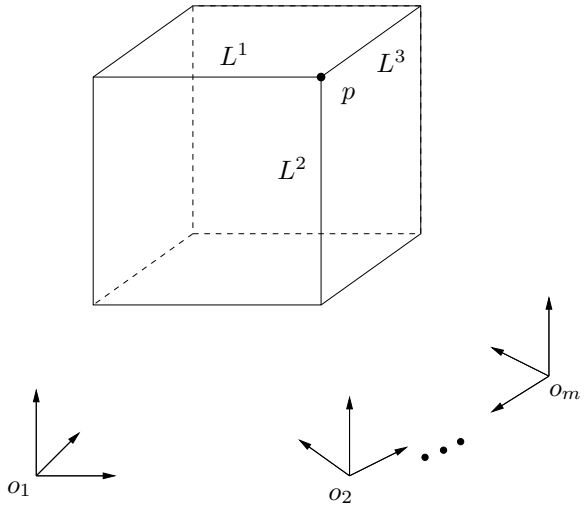

Figure 10.8. A standard cube. The three edges L^1, L^2, L^3 intersect at the corner p. The coordinate frames indicate that m images are taken at these vantage points.

and L^{3j}, $j = 1, \ldots, 8$. From the m images of the cube, we have the multiple view matrix M^j associated to p^j:

$$M^j = \begin{bmatrix} \widehat{\mathbf{x}_2^j} R_2 \mathbf{x}_1^j & \widehat{\mathbf{x}_2^j} T_2 \\ \mathbf{l}_2^{1jT} R_2 \mathbf{x}_1^j & \mathbf{l}_2^{1jT} T_2 \\ \mathbf{l}_2^{2jT} R_2 \mathbf{x}_1^j & \mathbf{l}_2^{2jT} T_2 \\ \mathbf{l}_2^{3jT} R_2 \mathbf{x}_1^j & \mathbf{l}_2^{3jT} T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}_m^j} R_m \mathbf{x}_1^j & \widehat{\mathbf{x}_m^j} T_m \\ \mathbf{l}_m^{1jT} R_m \mathbf{x}_1^j & \mathbf{l}_m^{1jT} T_m \\ \mathbf{l}_m^{2jT} R_m \mathbf{x}_1^j & \mathbf{l}_m^{2jT} T_m \\ \mathbf{l}_m^{3jT} R_m \mathbf{x}_1^j & \mathbf{l}_m^{3jT} T_m \end{bmatrix} \in \mathbb{R}^{[6(m-1)] \times 2} \tag{10.51}$$

where $\mathbf{x}_i^j \in \mathbb{R}^3$ means the image of the j^{th} corner in the i^{th} view and $\mathbf{l}_i^{kj} \in \mathbb{R}^3$ means the image of the k^{th} edge associated to the j^{th} corner in the i^{th} view. Theorem 10.1 says that $\text{rank}(M^j) = 1$. One can verify that $\alpha^j = [\lambda_1^j, 1]^T \in \mathbb{R}^2$ is in the kernel of M^j. In addition to the multiple images $\mathbf{x}_1^j, \mathbf{x}_2^j, \mathbf{x}_3^j$ of the j^{th} corner p^j itself, the extra rows associated to the line features $\mathbf{l}_i^{kj}, k = 1, 2, 3, i = 1, \ldots, m$ also help to determine the depth scale λ_1^j.

We can already see one advantage of the rank condition: It can simultaneously handle multiple incidence conditions associated to the same feature.[7] In principle, by using (10.33) or Corollary 10.5, one can further take into account that the four vertices and edges on each face are coplanar.[8] Since such incidence conditions and relations among points and lines occur frequently in practice, especially for man-made objects, such as buildings and houses, the use of multiple view matrix for mixed features is going to improve the quality of the overall reconstruction by explicitly taking into account all incidence relations among features of various types.

In order to estimate α^j we need to know the matrix M^j, i.e. we need to know the motion $(R_2, T_2), \ldots, (R_m, T_m)$. From the geometric meaning of $\alpha^j = [\lambda_1^j, 1]^T$, α^j can be solved already if we know only the motion (R_2, T_2) between the first two views, which can be initially estimated using the standard 8 point algorithm. Knowing α^j's, the equations:

$$M^j \alpha^j = 0, \quad j = 1, \ldots, 8 \tag{10.52}$$

become linear in $(R_2, T_2), \ldots, (R_m, T_m)$. We can use them to solve for the motions (again). Define the vectors:

$$\vec{R}_i = [r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}]^T \in \mathbb{R}^9$$

and $\vec{T}_i = T_i \in \mathbb{R}^3$, $i = 2, \ldots, m$. Solving (10.52) is then equivalent to finding the solution to the following equations for $i = 2, \ldots, m$:

$$P_i \begin{bmatrix} \vec{R}_i \\ \vec{T}_i \end{bmatrix} = \begin{bmatrix} \lambda_1^1 \widehat{\mathbf{x}_i^1} * \mathbf{x}_1^{1T} & \widehat{\mathbf{x}_i^1} \\ \lambda_1^1 \mathbf{l}_i^{11T} * \mathbf{x}_1^{1T} & \mathbf{l}_i^{11T} \\ \lambda_1^1 \mathbf{l}_i^{21T} * \mathbf{x}_1^{1T} & \mathbf{l}_i^{21T} \\ \lambda_1^1 \mathbf{l}_i^{31T} * \mathbf{x}_1^{1T} & \mathbf{l}_i^{31T} \\ \vdots & \vdots \\ \lambda_1^8 \widehat{\mathbf{x}_i^8} * \mathbf{x}_1^{8T} & \widehat{\mathbf{x}_i^8} \\ \lambda_1^8 \mathbf{l}_i^{18T} * \mathbf{x}_1^{8T} & \mathbf{l}_i^{18T} \\ \lambda_1^8 \mathbf{l}_i^{28T} * \mathbf{x}_1^{8T} & \mathbf{l}_i^{28T} \\ \lambda_1^8 \mathbf{l}_i^{38T} * \mathbf{x}_1^{8T} & \mathbf{l}_i^{38T} \end{bmatrix} \begin{bmatrix} \vec{R}_i \\ \vec{T}_i \end{bmatrix} = 0 \ \in \mathbb{R}^{48}, \tag{10.53}$$

where $A * B$ is the *Kronecker product* of A and B. In general, if we have more than 6 feature points (here we have 8) or equivalently 12 feature lines, the rank of the matrix P_i is 11 and there is a unique solution to $(\vec{R}_i, \vec{T}_i)$.

Let $\tilde{T}_i \in \mathbb{R}^3$ and $\tilde{R}_i \in \mathbb{R}^{3\times 3}$ be the (unique) solution of (10.53) in matrix form. Such a solution can be obtained numerically as the eigenvector of P_i

[7] In fact, any algorithm extracting point feature essentially relies on exploiting local incidence condition on multiple edge features. The structure of the M matrix simply reveals a similar fact within a larger scale.

[8] We here omit doing it for simplicity.

associated to the smallest singular value. Let $\tilde{R}_i = U_i S_i V_i^T$ be the SVD of $\tilde{R}_i$. Then the solution of (10.53) in $\mathbb{R}^3 \times SO(3)$ is given by:

$$T_i = \frac{\text{sign}(\det(U_i V_i^T))}{\sqrt[3]{\det(S_i)}} \tilde{T}_i \in \mathbb{R}^3, \tag{10.54}$$

$$R_i = \text{sign}(\det(U_i V_i^T))\, U_i V_i^T \in SO(3). \tag{10.55}$$

We then have the following linear algorithm for motion and structure estimation from three views of a cube:

Algorithm 10.1 (Motion and structure from mixed features). *Given m $(= 3)$ images $\mathbf{x}_1^j, \ldots, \mathbf{x}_m^j$ of $n(= 8)$ points p^j, $j = 1, \ldots, n$ (as the corners of a cube), and the images $\mathbf{l}_i^{kj}, k = 1, 2, 3$ of the three edges intersecting at p^j, estimate the motions (R_i, T_i), $i = 2, \ldots, m$ as follows:*

1. *Initialization: $s = 0$*
   (a) *Compute (R_2, T_2) using the 8 point algorithm for the first two views [LH81].*
   (b) *Compute $\alpha_s^j = [\lambda_1^j / \lambda_1^1, 1]^T$ where λ_1^j is the depth of the j^{th} point relative to the first camera frame.*
2. *Compute $(\tilde{R}_i, \tilde{T}_i)$ as the eigenvector associated to the smallest singular value of P_i, $i = 2, \ldots, m$.*
3. *Compute (R_i, T_i) from (10.54) and (10.55) for $i = 2, \ldots, m$.*
4. *Compute new $\alpha_{s+1}^j = \alpha^j$ from (10.52). Normalize so that $\lambda_{1,s+1}^1 = 1$.*
5. *If $||\alpha_s - \alpha_{s+1}|| > \epsilon$, for a pre-specified $\epsilon > 0$, then $s = s + 1$ and goto 2. Else stop.*

The camera motion is then the converged $(R_i, T_i), i = 2, \ldots, m$ and the structure of the points (with respect to the first camera frame) is the converged depth scalar $\lambda_1^j, j = 1, \ldots, n$.

We have a few comments on the proposed algorithm:

1. The reason to set $\lambda_{1,s+1}^1 = 1$ is to fix the universal scale. It is equivalent to putting the first point at a relative distance of 1 to the first camera center.
2. Although the algorithm is based on the cube, considers only three views, and utilizes only one type of multiple view matrix, it can be easily generalized to any other objects and arbitrarily many views whenever incidence conditions among a set of point features and line features are present. One may also use the rank conditions on different types of multiple view matrices provided by Theorem 10.1.
3. The above algorithm is a straightforward modification of the algorithm proposed for the pure point case given in Chapter 8. All the measurements of line features directly contribute to the estimation of

the camera motion and the structure of the points. Throughout the algorithm, there is no need to initialize or estimate the 3-D structure of lines.

The reader must be aware that the above algorithm is only *conceptual* (and naive in many ways). It by no means suggests that the resulting algorithm would work better in practice than some existing algorithms in every situation. The reason is, there are many possible ways to impose the rank conditions and each of them, although maybe algebraically equivalent, can have dramatically different numerical stability and sensitivity. To make the situation even worse, under different conditions (e.g., long baseline or short baseline), correctly imposing these rank conditions does require different numerical recipes.[9] A systematic characterization of numerical stability of the rank conditions remains largely open at this point. It is certainly the next logical step for future research.

10.6 Simulation results

Given the above conceptual algorithm the following section presents simulation results in order to justify our intuition behind the suggested global approach to structure from motion recovery. While at this stage we do not make any optimality claims, due to the linear nature of the proposed algorithms, we will demonstrate the performance and dependency of the algorithm on types of features in the presence of noise.

The simulation parameters are as follows: the camera's field of view is 90^o, image size is 500×500, everything is measured in units of the focal length of the camera, and features typically are suited with a depth variation is from 100 to 400 units of focal length away from the camera center, i.e. they are located in the truncated pyramid specified by the given field of view and depth variation (see Figure 8.5). Camera motions are specified by their translation and rotation axes. For example, between a pair of frames, the symbol XY means that the translation is along the X-axis and rotation is along the Y-axis. If we have a sequence of such symbols connected by hyphens, it specifies a sequence of consecutive motions. We always choose the amount of total motion, so that all feature points will stay in the field of view for all frames. In all simulations, independent Gaussian noise with a standard deviation (std) given in pixels is added to each image point, and each image line is perturbed in a random direction of a random angle with a corresponding standard deviation given in degrees.[10] Error measure for rotation is $\arccos\left(\frac{tr(R\tilde{R}^T)-1}{2}\right)$ in degrees where $\tilde{R}$ is an estimate of the true

[9] This is true even for the standard 8 point algorithm in the two view case.

[10] Since line features can be measured more reliably than point features, lower noise level is added to them in simulations.

R. Error measure for translation is the angle between T and $\tilde{T}$ in degrees where $\tilde{T}$ is an estimate of the true T. Error measure for the scene structure is the percentage of $\|\alpha - \tilde{\alpha}\|/\|\alpha\|$ where $\tilde{\alpha}$ is an estimate of the true α.

We apply the algorithm to a scene which consists of (four) cubes. Cubes are good objects to test the algorithm since the relationships between their corners and edges are easily defined and they represent a fundamental structure of many objects in real-life. It is certainly a first step to see how the multiple view matrix based approach is able to take into account point and line features as well as their inter-relationships to facilitate the overall recovery. The length of the four cube edges are 30, 40, 60 and 80 units of focal length, respectively. The cubes are arranged so that the depth of their corners ranges from 75 to 350 units of focal length. The three motions (relative to the first view) are an XX-motion with -10 degrees rotation and 20 units translation, a YY-motion with 10 degrees rotation and 20 units translation and another YY-motion with -10 degrees rotation and 20 units translation, as shown in Figure 10.9.

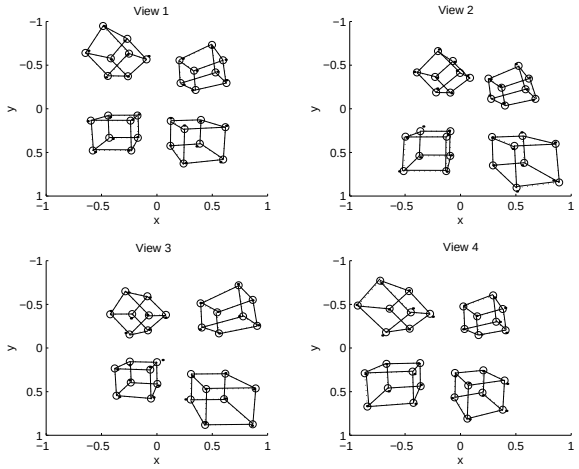

Figure 10.9. Four views of four 3-D cubes in (normalized) image coordinates. The circle and the dotted lines are the original images, the dots and the solid lines are the noisy observations under 5 pixels noise on point features and 0.5 degrees noise on line features.

We run the algorithm for 1000 trials with the noise level on the point features from 0 pixel to 5 pixels and a corresponding noise level on the line features from 0 degree to 1 degrees. Relative to the given amount of translation, 5 pixels noise is rather high because we do want to compare how all the algorithms perform over a large range of noise levels. The results of the motion estimate error and structure estimate error are given in Figure 10.10 and 10.11 respectively. The "Point feature only" algorithm essentially uses the multiple view matrix M in (10.52) without all the

rows associated to the line features; and the "Mixed features" algorithm uses essentially the same M as in (10.52). Both algorithms are initialized by the standard 8 point algorithm. The "Mixed features" algorithm gives a significant improvement in all the estimates as a result of the use of both point and line features in the recovery. Also notice that, at a high noise levels, even though the 8 point algorithm gives rather off initialization values, the two iterative algorithms manage to converge back to reasonable estimates.

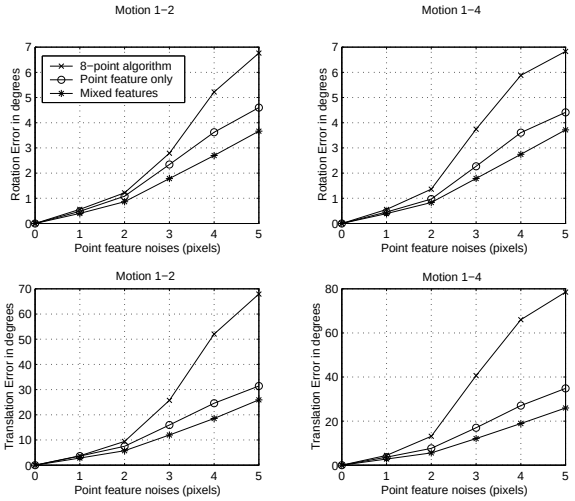

Figure 10.10. Motion estimates error versus level of noises. "Motion x-y" means the estimate for the motion between image frames x and y. Since the results are very much similar, we only plotted "Motion 1-2" and "Motion 1-4".

10.7 Summary

10.8 Exercises

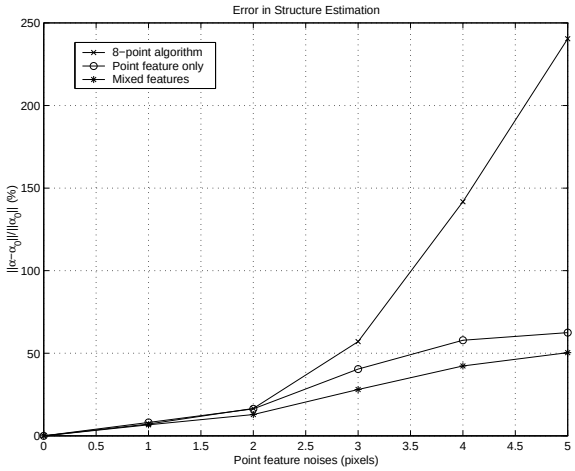

Figure 10.11. Structure estimates error versus level of noises. Here α_0 represents the true structure and α the estimated.

Chapter 11

Multiple view geometry in high dimensional space

A rather common problem that people typically encounter in computer vision, robotics, medical imaging, signal processing, multi-channel communications and even biophysics is to reconstruct (the "shape" and "location" of) a high-dimensional "object" or "signal" from its lower-dimensional "projections" (e.g., images of a building taken by a Kodak camera). Due to physical or technological limitations, it is usually much more economic or convenient to obtain low-dimensional measurements rather than to try to measure the object directly in the high-dimensional space where it resides. It is already well-known to us now that, under proper conditions, if enough two-dimensional images or views of a three-dimensional object are given, a full reconstruction of both the shape and location of the object is possible, without even knowing where these images were taken by the camera. It is then reasonable to expect that this should also be true for projections in spaces of other (typically higher) dimensions. To a large extent, this chapter shows how to systematically make this generalization, as well as to demonstrate potential uses of such a generalized theory to problems beyond the scope of classic multiple view geometry in computer vision.

While the multiple view geometry developed so far typically applies to the situation that the scene is static and only the camera is allowed to move, the generalized theory would no longer have such limitation. It is easy to show that, if a scene contains independently moving objects, we usually can embed the problem into a space of higher dimensional linear space, with a point in the high-dimensional space representing both the location and velocity of a moving point in the physical three-dimensional world. In this chapter, our focus is on a complete characterization of algebraic and

geometric constraints that govern multiple k-dimensional images of hyperplanes in an n-dimensional space. We show that these constraints can again be captured through the rank condition techniques. Such techniques also simultaneously capture all types of geometric relationships among hyperplanes, such as inclusion, intersection and restriction. The importance of this study is at least two-fold: 1. In many applications, objects involved are indeed multi-faceted (polygonal) and their shape can be well modeled (or approximated) as a combination of hyperplanes; 2. In some cases, there is not enough information or it is not necessary to locate the exact location of points in a high-dimensional space and instead, we may still be interested in identifying them up to some hyperplane. As we will point out later, for the special case $n = 3$, the results naturally reduce to what is known so far for points, lines and planes in preceding chapters. Since reconstruction is not the main focus of this chapter, the reader is referred to other chapters for how to use these constraints to develop algorithms for various reconstruction purposes.

11.1 Projection in high dimensional space

In this section, we introduce notation and basic ingredients for generalizing multiple view geometry from $\mathbb{R}^3$ to $\mathbb{R}^n$. More importantly, we will carefully single out assumptions that are needed to make such generalization well-posed. Despite these assumptions, the formulation remains general enough to encompass almost all types of projections of practical importance (e.g., perspective or orthographic projections) and a large class of transformations between vantage points (e.g., Euclidean, affine or projective transformations).

11.1.1 Image formation

An *abstract camera* is a pair consisting of a k-dimensional hyperplane $I^k \subseteq \mathbb{R}^n$ $(k < n)$ and a point o outside I^k. The point o is called the *camera center*.

Given an abstract camera (I^k, o) we can find a point $o' \in I^k$ be such that $\vec{oo'}$ is orthogonal to $\vec{xo'}$ for all $x \in I^k$. In fact this point o' is the one that minimizes the value of $\|ox\|$ for all $x \in I^k$. The point o' is called the *image center*. By associating a coordinate system to I^k with its origin at o', we have an *image plane* (or more correctly an *image space*). Without loss of generality, we usually assume the distance $\|oo'\|$ is normalized to 1.[1]

[1] In computer vision, the distance $\|oo'\|$ is the focal length of the camera. Since there is always a universal scaling ambiguity, typically a metric is chosen such that this distance is normalized.

If we define the world coordinate system in $\mathbb{R}^n$ with its origin at o. Then for any point $p \in \mathbb{R}^n$, the coordinate $x = [x_1, x_2, \ldots, x_k]^T$ of its image in the image plane is obtained by:

$$\lambda x = PX, \tag{11.1}$$

where $X = [X_1, X_2, \ldots, X_n]^T \in \mathbb{R}^n$ is the coordinate of p, matrix $P \in \mathbb{R}^{k\times n}$, $\lambda = \lambda(X)$ is a scale factor which is a function of X. We need the following two assumptions:

1. Matrix P has full rank, i.e. $\text{rank}(P) = k$. Otherwise, if $\text{rank}(P) = k' < k$, it simply becomes the projection from $\mathbb{R}^n$ to $\mathbb{R}^{k'}$.
2. λ is a first order function of X, i.e. $\lambda = \vec{\alpha}^T X + c$, where $\vec{\alpha} = [\alpha_1, \ldots, \alpha_n]^T \in \mathbb{R}^n$ is a
constant vector and $c \in \mathbb{R}$ is a constant.

Note if $X = 0$, then we define $x = 0$. The image is not defined for $X \in \mathbb{R}^n$ such that $X \neq 0$ and $\lambda(X) = 0$. For example, in the perspective projection from 3D space to 2D image, $P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, $\vec{\alpha} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and $c = 0$. The image is not defined for points on the plane $X_3 = 0$ except the origin $X = 0$. For the orthographic projection from 3D space to 2D image, matrix P is still the same while $\vec{\alpha} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ and $c = 1$. The image is defined for every point in $\mathbb{R}^3$.

For any d-dimensional ($0 \leq d < n$) hyperplane $H^d \subseteq \mathbb{R}^n$ with equation $A^T X + b = 0$ ($A \in \mathbb{R}^{n\times(n-d)}$, $b \in \mathbb{R}^{n-d}$), its image is a $q(\leq d)$-dimensional hyperplane S^q in the image plane with equation $A_I^T x + b_I = 0$ ($A_I \in \mathbb{R}^{k\times(k-q)}$, $b_I \in \mathbb{R}^{k-q}$). Then we define the *preimage* of S^q to be a hyperplane F with equation $A_F^T X + b_F = 0$, where $A_F^T = A_I^T P + b_I \vec{\alpha}^T$ and $b_F = cb_I$. Hence $F \subseteq \mathbb{R}^n$ has dimension $n - k + q$. It is the largest set in $\mathbb{R}^n$ that can give rise to the same image as H^d does and $H^d \subseteq F$. So a basic relationship is:

$$n - k \geq d - q \geq 0. \tag{11.2}$$

Figure 11.1 illustrates a special case when $n = 3, k = 1$ and $d = q = 0$.

11.1.2 Motion of the camera and homogeneous coordinates

A coordinate frame $X = [X_1, X_2, \ldots, X_n]^T$ (for the space) is associated to the camera, whose origin is the camera center o. We define the *motion* of the camera to be a transformation which maps one camera coordinate frame to another (probably with a different origin) in $\mathbb{R}^n$. It is obvious that such a transformation is an affine transformation which can be represented by a matrix $R \in \mathbb{R}^{n\times n}$ and a vector $T \in \mathbb{R}^n$ such that

$$Y = RX + T \tag{11.3}$$

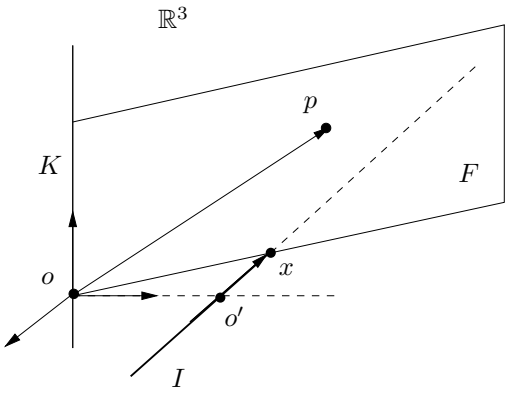

Figure 11.1. An image x of a point $p \in \mathbb{R}^3$ under a perspective projection from $\mathbb{R}^3$ to $\mathbb{R}^1(\doteq I)$. K is the subspace orthogonal to the subspace spanned by o and I. The plane F is the preimage of x, which is the subspace spanned by p and K.

where $X \in \mathbb{R}^n$ and $Y \in \mathbb{R}^n$ are coordinates of the same point p with respect to the two coordinate frames (cameras) respectively. For this map to be invertible, R must be invertible, i.e. $R \in GL(n, \mathbb{R})$, the general linear group.

In the above formulation it is more convenient to use the so-called *homogeneous coordinates* for both the point and its image. The homogeneous coordinate for point p is $\mathbf{X} = [X_1, \ldots, X_n, 1]^T$. Then its homogeneous coordinate in the new coordinate frame after motion of the camera is

$$\mathbf{Y} = g\mathbf{X}, \tag{11.4}$$

where $g = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$ belongs to $GL(n+1, \mathbb{R})$. The homogeneous coordinate of the image of p is $\mathbf{x} = [x_1, \ldots, x_k, 1]^T$. Let $P' = \begin{bmatrix} P & 0 \\ \vec{\alpha}^T & c \end{bmatrix} \in \mathbb{R}^{(k+1)\times(n+1)}$. Then we have

$$\lambda\mathbf{x} = P'\mathbf{X}. \tag{11.5}$$

Since $\text{rank}(P') \geq \text{rank}(P) = k$, so either $\text{rank}(P') = k+1$ or $\text{rank}(P') = k$.

In general, assume we have m views of the point. The motion for each view is $g_i = \begin{bmatrix} R_i & T_i \\ 0 & 1 \end{bmatrix}$, then the image in the i^{th} frame is

$$\lambda_i \mathbf{x}_i = P' g_i \mathbf{X} = \Pi_i \mathbf{X}, \quad i = 1, \ldots, m, \tag{11.6}$$

where $\Pi_i = P' g_i \in \mathbb{R}^{(k+1)\times(n+1)}$ and $\text{rank}(\Pi_i) = \text{rank}(P')$ since g_i is of full rank. It is typical in computer vision that in the above equations, except

for $\mathbf{x}_i$'s, everything else is unknown and subject to recovery.[2] It is easy to see that a unique recovery would be impossible if the matrix Π_i's could be arbitrary. Hence the key is to utilize the fact that Π_i's are from certain motion group and do have some nice internal structure.[3] Even so, it is still very difficult to recover $\Pi, \lambda, \mathbf{X}$ altogether directly from above equations. We first observe that, among the unknowns, both λ and $\mathbf{X}$ encode information about the location of the point p in $\mathbb{R}^n$, which are not intrinsically available from the images. Hence we would like to eliminate them from the equations. The remaining relationships would be between $\mathbf{x}$ and Π only, i.e. between the images and the camera configuration. These relationships, as we will soon see, are to be captured precisely by the rank conditions.

Remark 11.1. *Most results in this chapter essentially apply to any system of equations of the type:*

$$\lambda_i \mathbf{x}_i = \Pi_i \mathbf{X}, \quad i = 1, \ldots, m.$$

The internal structure of Π_i's seems to be less important if we do not talk about reconstruction or its geometric interpretation. This is also the reason why, as the rank condition being a purely algebraic result, it really does not matter whether the g in $\Pi = P'g$ is Euclidean, affine or projective. In fact, g may belong to any subgroup of $GL(n+1, \mathbb{R})$, each representing a special "multiple view geometry". The rank condition, to a large extent, is a concise way of expressing the above equations in terms of only $\mathbf{x}_i$'s and Π_i's.

11.1.3 Preimage and co-image

The fact that the dimension of image plane is lower than that of space implies that different points in space may be projected onto the same image point. Obviously, in image it is impossible to distinguish two points in space if their difference is in the kernel of the matrix P'. This leads to the important notion of "preimage". Intuitively, the preimage consists of the set of all points that may give rise to the same image. It plays a crucial role later in the derivation of the rank condition. Here we give it a precise definition as well as discuss its relation with the rank of the matrix P'.

The case when $\text{rank}(P') = k$

We first consider the situation that $\text{rank}(P') = k$. If $\text{rank}(P') = k$, we must have $\vec{\beta}^T P = \vec{\alpha}^T$ for some $\vec{\beta} \in \mathbb{R}^k$ and $c = 0$. However, this implies that

$$\lambda = \lambda \vec{\beta}^T x, \tag{11.7}$$

[2] The above equations are associated to only one feature point. We usually need images of *multiple* points in order to recover all the unknowns.

[3] We later will see this can also be relaxed when we allow Π to depend on time.

which further implies that $\lambda = 0$ or $\vec{\beta}^T x = 1$. Since the points (except the camera center) that make $\lambda = 0$ do not have well-defined image, we only need to consider the case of $\vec{\beta}^T x = 1$. This later equation implies that the real meaningful image plane is only a $(k-1)$-dimensional hyperplane in I^k. The whole problem reduces to one for projection from $\mathbb{R}^n$ to $\mathbb{R}^{k-1}$. Hence, without loss of generality, we only need to consider the case when $\text{rank}(P') = k+1$.

The case when $\text{rank}(P') = k+1$

First, we consider the case that $\text{rank}(P') = k+1$. Both the perspective projection and the orthographic projection belong to this case. For the d-dimensional hyperplane H^d mentioned above, its equation is then

$$A'^T \mathbf{X} = 0, \tag{11.8}$$

where $A' = \begin{bmatrix} A \\ b^T \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n-d)}$ and the equation of its image S^q is

$$A_I'^T \mathbf{x} = 0, \tag{11.9}$$

where $A'_I = \begin{bmatrix} A_I \\ b_I^T \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k-q)}$.

Note the set of $\mathbf{X} \in \mathbb{R}^{n+1}$ satisfying (11.8) is a subspace H^{d+1} of dimension $d+1$ and the original hyperplane H^d is the intersection of this set with the hyperplane in $\mathbb{R}^{n+1}$: $X_{n+1} = 1$. Hence, instead of studying the object in $\mathbb{R}^n$, we can study the corresponding object of one dimension higher in $\mathbb{R}^{n+1}$. Consequently, the image S^q becomes a subspace $S^{q+1} = \{\mathbf{x} \in \mathbb{R}^{k+1} : A_I'^T \mathbf{x} = 0\} \subseteq \mathbb{R}^{k+1}$. The space $\mathbb{R}^{k+1}$ can be interpreted as the space I^{k+1} formed by the camera center o and the image plane I^k, which we denote as *image space*. So S^q is the intersection of S^{q+1} with the image plane I^k in the image space I^{k+1}. Define the set $F = \{\mathbf{X} \in \mathbb{R}^{n+1} : A_I'^T P' \mathbf{X} = 0\}$ to be the *preimage* of S^{q+1}, then $H^{d+1} \subseteq F$ and F is the largest set in $\mathbb{R}^{n+1}$ that can give rise to the same image of H^{d+1}. If we consider the I^{k+1} as a subspace of $\mathbb{R}^{n+1}$, then S^{q+1} is the intersection of the image space with F or H^{d+1}. Since it is apparent that the space summation $F + I^{k+1} = \mathbb{R}^{n+1}$, the dimension of F is

$$\dim(F) = \dim(F \cap I^{k+1}) + \dim(F + I^{k+1}) - \dim(I^{k+1}) = (n-k) + q + 1. \tag{11.10}$$

If we take the motion into consideration, then $F = \{\mathbf{X} \in \mathbb{R}^{n+1} : A_I'^T P' g \mathbf{X} = 0\}$, which is the preimage expressed in the world frame. Since g is of full rank, the above dimensional property for F still holds. Notice that the dimension of the preimage F is in general larger than that of the subspace H^{d+1} which gives rise to the image S^q in the image plane in the first place. Hence in general, hyperplanes which may generate the same

image are not necessarily unique and may even have different dimensions. More specifically, such hyperplanes may be of dimensions $d = q, q+1, \ldots$, or $q+(n-k-1)$.[4] Nonetheless, if the camera and the object hyperplane are of general configuration, typically the dimension of the image of a hyperplane H^d is

$$q = \min\{d, k\}. \tag{11.11}$$

Since S^{q+1} is a subspace of the image space $I^{k+1} = \mathbb{R}^{k+1}$, it can be described by the span of a matrix $\mathbf{s} = [u_1, u_2, \ldots, u_{q+1}] \in \mathbb{R}^{(k+1)\times(q+1)}$ or by its maximum complementary orthogonal space (in I^{k+1}) which is spanned by $\mathbf{s}^\perp = [v_1, v_2, \ldots, v_{k-q}] \in \mathbb{R}^{(k+1)\times(k-q)}$ such that $(\mathbf{s}^\perp)^T\mathbf{s} = 0$. By abuse of language, we also call $\mathbf{s}$ the *image* and we call $\mathbf{s}^\perp$ its *co-image*. Of course the co-image of $\mathbf{s}^\perp$ is $\mathbf{s}$. $\mathbf{s}$ and $\mathbf{s}^\perp$ are simply two equivalent ways of describing the subspace S^{q+1} of the same image. For example, in the equation (11.9), A'_I can be viewed as the co-image of the hyperplane H^d. Figure 11.2 illustrates the image and co-image of a line under the perspective projection from $\mathbb{R}^3$ to $\mathbb{R}^2$.

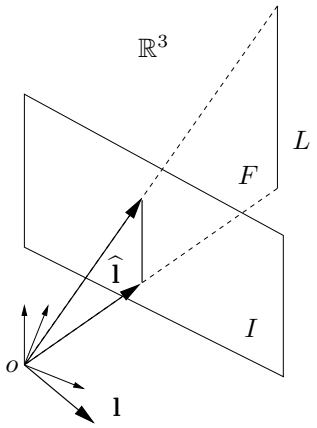

Figure 11.2. The image $\mathbf{s} = \widehat{\mathbf{l}}$ and co-image $\mathbf{s}^\perp = \mathbf{l}$ of a line L in $\mathbb{R}^3$. The three column vectors of the image $\widehat{\mathbf{l}}$ span the two-dimensional subspace (in the image space I^{2+1}) formed by o and L (in this case it coincides with the preimage F); and the column vector of the co-image $\mathbf{l}$ is the vector orthogonal to that subspace.

When P' is of rank $k+1$, there always exists a matrix $g \in GL(n+1, \mathbb{R})$ such that $P'g$ is of the standard form:

$$P'g = \begin{bmatrix} I_{(k+1)\times(k+1)} & 0_{(k+1)\times(n-k)} \end{bmatrix}. \tag{11.12}$$

[4]Here we assume that the hyperplanes do not cross the origin o. Otherwise, d can be $q + n - k$.

Hence, without loss of generality, we may assume that P' itself is of above form already. This is an assumption that holds algebraically. However, if we want to study the physical meaning of the matrix P', then two situations need to be distinguished. First is the case that we can find $k+1$ linear independent columns from the first n columns of P'. In this case the matrix g is an affine transformation. The second case is when we can only find at most k linear independent columns from the first n columns of P'. Then the matrix g is not an affine transformation. The perspective projection mentioned above is an example of the first case and the orthographic projection belongs to the second situation. Although the rank conditions to be derived do not discriminate against these projections, this difference however may dramatically simplify the reconstruction problem in the case of orthographic projection (see [TK92]).

11.1.4 Generic motion assumption

If we fix the location of a hyperplane $H^d \subseteq \mathbb{R}^n$ and move the camera around the hyperplane under the motion g, then in general the image of the hyperplane will have dimension $q = \min\{d, k\}$. We call the set of motions $\{g\}$ of the camera which make $q < \min\{d, k\}$ as *degenerate* and call all the other motions as *generic*. Then the set of degenerate motions typically only constitute a zero-measure set in a chosen (continuous) group of transformations. Also notice that, under a generic motion, for a hyperplane of dimension $d \geq k$, its image is of dimension k, which means it will occupy the whole image plane. In general we cannot extract any useful information from such an image (about either the location of the hyperplane or the motion of the camera). Therefore in this chapter, unless otherwise stated, we always assume:

1. Motions of the camera are generic with respect to all the hyperplane involved;
2. Hyperplanes whose images are considered always have dimension $d < k$.

Although the image of a hyperplane of dimension higher than or equal to k is considered as invalid, the presence of such a hyperplane may still affect the constraints on the images of other hyperplanes in it. We will discuss this in more detail in Section 11.3.4.

11.2 Rank condition on multiple images of a point

One basic type of hyperplane in $\mathbb{R}^3$ is a point (when $d = 0$). From Chapter 8, we know that multiple images of a point satisfy certain rank condition

(Theorem 8.1). Here we first study how to generalize this result to points in $\mathbb{R}^n$.

Take multiple, say m, images of the same point p with respect to multiple camera frames. We obtain a family of equations

$$\mathbf{x}_i \lambda_i = P' g_i \mathbf{X} = \Pi_i \mathbf{X}, \quad i = 1, \dots, m. \tag{11.13}$$

To simplify notation, let $\Pi_i = P' g_i = \begin{bmatrix} \bar{R}_i & \bar{T}_i \end{bmatrix} \in \mathbb{R}^{(k+1)\times(n+1)}$, where $\bar{R}_i \in \mathbb{R}^{(k+1)\times(k+1)}$ and $\bar{T}_i \in \mathbb{R}^{(k+1)\times(n-k)}$.[5] In the above equations, except for the $\mathbf{x}_i$'s, everything else is unknown and subject to recovery. However, solving for the λ_i's, Π_i's and $\mathbf{X}$ simultaneously from such equations is extremely difficult – at least nonlinear. A traditional way to simplify the task is to decouple the recovery of the matrices Π_i's from that of λ_i's and $\mathbf{X}$. Then the remaining relationship would be between the images $\mathbf{x}_i$'s and the camera configuration Π_i's only. Since such a relationship does not involve knowledge on the location of the hyperplane in $\mathbb{R}^n$, it is referred to as "intrinsic". It constitutes as the basis for any image-only based techniques. For that purpose, let us rewrite the system of equations (11.13) in a single matrix form

$$\mathcal{I}\vec{\lambda} = \Pi \mathbf{X} \tag{11.14}$$

$$\begin{bmatrix} \mathbf{x}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{x}_m \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} \Pi_1 \\ \Pi_2 \\ \vdots \\ \Pi_m \end{bmatrix} \mathbf{X}. \tag{11.15}$$

For obvious reasons, we will call $\vec{\lambda} \in \mathbb{R}^m$ the *scale vector*, and $\Pi \in \mathbb{R}^{[m(k+1)]\times(n+1)}$ the *projection matrix* associated to the *image matrix* $\mathcal{I} \in \mathbb{R}^{[m(k+1)]\times m}$.

In order to eliminate the unknowns $\vec{\lambda}$ and $\mathbf{X}$, we consider the following matrix in $\mathbb{R}^{[m(k+1)]\times(m+n+1)}$

$$N_p \doteq [\Pi,\ \mathcal{I}] = \begin{bmatrix} \Pi_1 & \mathbf{x}_1 & 0 & \cdots & 0 \\ \Pi_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \Pi_m & 0 & \cdots & 0 & \mathbf{x}_m \end{bmatrix} \in \mathbb{R}^{[m(k+1)]\times(m+n+1)}. \tag{11.16}$$

Apparently, the rank of N_p is no less than $k + m$ (assuming the generic configuration such that $\mathbf{x}_i$ is not zero for all $1 \le i \le m$). It is clear that there exists $v \doteq [\mathbf{X}^T, -\vec{\lambda}^T]^T \in \mathbb{R}^{m+n+1}$ in the null space of N_p. Then the equation (11.14) is equivalent to

$$m + k \le \operatorname{rank}(N_p) \le m + n \tag{11.17}$$

[5] Only in the special case that $k = n - 1$, we have $\bar{R}_i = R_i$ and $\bar{T}_i = T_i$.

as long as $m(k+1) \geq (m+n+1)$. In any case, the matrix N_p is rank deficient.

Multiplying N_p on the left by the following matrix

$$D_p = \begin{bmatrix} \mathbf{x}_1^T & 0 & \cdots & 0 \\ (\mathbf{x}_1^\perp)^T & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \mathbf{x}_m^T \\ 0 & \cdots & 0 & (\mathbf{x}_m^\perp)^T \end{bmatrix} \in \mathbb{R}^{[m(k+1)]\times[m(k+1)]}$$

yields the following matrix $D_p N_p$ in $\mathbb{R}^{[m(k+1)]\times(m+n+1)}$

$$D_p N_p = \begin{bmatrix} \mathbf{x}_1^T\bar{R}_1 & \mathbf{x}_1^T\bar{T}_1 & \mathbf{x}_1^T\mathbf{x}_1 & 0 & 0 & 0 \\ (\mathbf{x}_1^\perp)^T\bar{R}_1 & (\mathbf{x}_1^\perp)^T\bar{T}_1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & 0 & 0 & \ddots & 0 \\ \mathbf{x}_m^T\bar{R}_m & \mathbf{x}_m^T\bar{T}_m & 0 & 0 & 0 & \mathbf{x}_m^T\mathbf{x}_m \\ (\mathbf{x}_m^\perp)^T\bar{R}_m & (\mathbf{x}_m^\perp)^T\bar{T}_m & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Since D_p is of full rank $m(k+1)$, we have $\text{rank}(D_p N_p) = \text{rank}(N_p)$. It is then easy to see that the rank of N_p is related to the sub-matrix of $D_p N_p$

$$C_p = \begin{bmatrix} (\mathbf{x}_1^\perp)^T\bar{R}_1 & (\mathbf{x}_1^\perp)^T\bar{T}_1 \\ (\mathbf{x}_2^\perp)^T\bar{R}_2 & (\mathbf{x}_2^\perp)^T\bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T\bar{R}_m & (\mathbf{x}_m^\perp)^T\bar{T}_m \end{bmatrix} \in \mathbb{R}^{(mk)\times(n+1)} \tag{11.18}$$

by the following equation

$$k \leq \text{rank}(C_p) = \text{rank}(N_p) - m \leq n. \tag{11.19}$$

This rank condition can be further reduced if we choose the world coordinate frame to be the first camera frame, i.e. $[R_1, T_1] = [I, 0]$. Then $\bar{R}_1 = I_{(k+1)\times(k+1)}$ and $\bar{T}_1 = 0_{(k+1)\times(n-k)}$. Note that, algebraically, we do not lose any generality in doing so. Now the matrix C_p becomes

$$C_p = \begin{bmatrix} (\mathbf{x}_1^\perp)^T & 0 \\ (\mathbf{x}_2^\perp)^T\bar{R}_2 & (\mathbf{x}_2^\perp)^T\bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T\bar{R}_m & (\mathbf{x}_m^\perp)^T\bar{T}_m \end{bmatrix} \in \mathbb{R}^{(mk)\times(n+1)}. \tag{11.20}$$

Multiplying C_p on the right by the following matrix

$$D_{p1} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_1^\perp & 0 \\ 0 & 0 & I_{(n-k)\times(n-k)} \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$

yields the matrix

$$C_p D_{p1} = \begin{bmatrix} 0 & (\mathbf{x}_1^\perp)^T \mathbf{x}_1^\perp & 0 \\ (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{x}_1 & (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{x}_1^\perp & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{x}_1 & (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{x}_1^\perp & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{(mk)\times(n+1)}. \tag{11.21}$$

Note that D_{p1} is of full rank $n+1$ and $(\mathbf{x}_1^\perp)^T \mathbf{x}_1^\perp$ is of rank k. Let us define the so-called *multiple view matrix* M_p associated to a point p to be

$$M_p \doteq \begin{bmatrix} (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{x}_1 & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{x}_1 & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[(m-1)k]\times(n-k+1)}. \tag{11.22}$$

Then it is obvious that its rank is related to that of C_p by $\text{rank}(M_p) = \text{rank}(C_p) - k$. Therefore, we have:

$$\boxed{0 \leq \text{rank}(M_p) = \text{rank}(C_p) - k \leq n - k.} \tag{11.23}$$

Although the rank condition on M_p seems to be more compact than that on C_p, C_p depends evenly on $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ and M_p treats $\mathbf{x}_1$ differently from the others.

Example 11.1 (Epipolar constraint). *For a conventional camera, the dimension of the space is $n = 3$ and that of the image plane is $k = 2$. For two views and (R, T) being the relative motion between them, the above rank condition becomes the well-known* epipolar constraint*:*

$$0 \leq \textit{rank}\begin{bmatrix} \widehat{\mathbf{x}_2} R \mathbf{x}_1 & \widehat{\mathbf{x}_2} T \end{bmatrix} \leq 1 \quad \Leftrightarrow \quad \mathbf{x}_2^T R \widehat{T} \mathbf{x}_1 = 0.$$

11.3 Rank conditions on multiple images of hyperplanes

As we have seen in the previous section, multiple images of a point in $\mathbb{R}^n$ are governed by certain rank conditions. Such conditions not only concisely capture geometric constraints among multiple images, but also are the key to further reconstruction of the camera motion and scene structure. In this section, we generalize the rank conditions to hyperplanes in $\mathbb{R}^n$. Furthermore, we will also show that basic geometric relationships among different hyperplanes, i.e. *inclusion* and *intersection* of hyperplanes, can also be fully captured by such rank conditions. This in fact should be expected. As we will see later, roughly speaking, what the rank conditions indeed capture geometrically is the fact that the hyperplane (observed from different vantages points) must belong to the intersection of all preimages. Inclusion and intersection are properties of the same nature for hyperplanes – they are all

incidence conditions among hyperplanes. Furthermore, these relationships can be easily detected or verified in images and such knowledge can be and should be exploited if a consistent reconstruction is sought.

11.3.1 Multiple images of a hyperplane

As mentioned before, the image of a d-dimensional hyperplane H^d is a $q(\leq d)$-dimensional hyperplane in the image plane. It can be described as a $(q+1)$-dimensional subspace S^{q+1} formed by the hyperplane and the camera center o

$$\mathbf{s} \doteq [u_1, \ldots, u_{q+1}] \in \mathbb{R}^{(k+1)\times(q+1)}, \tag{11.24}$$

or its co-image

$$\mathbf{s}^\perp \doteq [v_1, \ldots, v_{k-q}] \in \mathbb{R}^{(k+1)\times(k-q)}. \tag{11.25}$$

Now given m images $\mathbf{s}_1, \ldots, \mathbf{s}_m$ (hence co-images $\mathbf{s}_1^\perp, \ldots, \mathbf{s}_m^\perp$) of H^d relative to m camera frames, let $q_i + 1$ be the dimension of $\mathbf{s}_i, i = 1, \ldots, m$, note for generic motions, $q_i = d, i = 1, \ldots, m$. We then obtain the following matrix

$$C_h = \begin{bmatrix} (\mathbf{s}_1^\perp)^T \bar{R}_1 & (\mathbf{s}_1^\perp)^T \bar{T}_1 \\ (\mathbf{s}_2^\perp)^T \bar{R}_2 & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d)]\times(n+1)}. \tag{11.26}$$

Let S^{d+1} be the subspace spanned by H^d and the origin of the world coordinate frame. Then it is immediately seen that S^{d+1} in its homogeneous representation is the kernel of C_h. In other words, the rank of C_h is

$$(k-d) \leq \text{rank}(C_h) \leq (n-d). \tag{11.27}$$

The lower bound comes from the fact $\text{rank}((\mathbf{s}_1^\perp)^T \Pi_1) = \text{rank}(\mathbf{s}_1^\perp) = k - d$. Without loss of generality, we choose the first camera frame to be the world frame, i.e. $\Pi_1 = [I\ 0]$. Then the matrix C_h becomes

$$C_h = \begin{bmatrix} (\mathbf{s}_1^\perp)^T & 0 \\ (\mathbf{s}_2^\perp)^T \bar{R}_2 & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d)]\times(n+1)}. \tag{11.28}$$

Multiplying C_h on the right by the following matrix

$$D_{h1} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_1^\perp & 0 \\ 0 & 0 & I_{(n-d)\times(n-d)} \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$

yields the following matrix

$$C_h D_{h1} = \begin{bmatrix} 0 & (\mathbf{s}_1^\perp)^T \mathbf{s}_1^\perp & 0 \\ (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{s}_1 & (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{s}_1^\perp & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{s}_1 & (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{s}_1^\perp & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d)]\times(n+1)}. \tag{11.29}$$

Note that $D_{p1} \in \mathbb{R}^{(n+1)\times(n+1)}$ is of full rank $n+1$ and $(\mathbf{s}_1^\perp)^T \mathbf{s}_1^\perp \in \mathbb{R}^{(k-d)\times(k-d)}$ is of rank $k-d$. Let us define the so-called *multiple view matrix* M_h associated to the hyperplane H^d to be

$$M_h \doteq \begin{bmatrix} (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{s}_1 & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{s}_1 & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[(m-1)(k-d)]\times(n-k+d+1)}. \tag{11.30}$$

Then it is obvious that its rank is related to that of C_h by $\text{rank}(M_h) = \text{rank}(C_h) - (k-d)$. Therefore, we have:

$$0 \leq \text{rank}(M_h) = \text{rank}(C_h) - (k-d) \leq (n-k). \tag{11.31}$$

Hence we obtain the following statement which generalizes (11.23):

Theorem 11.1 (Rank condition for multiple images of one hyperplane). *Given m images $\mathbf{s}_1, \ldots, \mathbf{s}_m$ and co-images $\mathbf{s}_1^\perp, \ldots, \mathbf{s}_m^\perp$ of a d-dimensional hyperplane H^d in $\mathbb{R}^n$, the associated multiple view matrix M_h defined above satisfies*

$$\boxed{0 \leq \textit{rank}(M_h) \leq (n-k).} \tag{11.32}$$

It is easy to see that, in the case that the hyperplane is a point, we have $d = 0$. Hence $0 \leq \text{rank}(M_h) \leq (n-k)$. So the result (11.23) given in Section 11.2 is trivially implied by this theorem. The significance of this theorem is that the rank of the matrix M_h defined above does *not* depend on the dimension of the hyperplane H^d. Its rank only depends on the difference between the dimension of the ambient space $\mathbb{R}^n$ and that of the image plane $\mathbb{R}^k$. Therefore, in practice, if many features of a scene are exploited for reconstruction purposes, it is possible to design algorithms that do not discriminate different features.

Example 11.2 (Trilinear constraints on three views of a line). *When $n = 3, k = 2, m = 3$ and $d = 1$, it gives the known result on three views of a line:*

$$\textit{rank} \begin{bmatrix} \mathbf{l}_2^T R_2 \widehat{\mathbf{l}_1} & \mathbf{l}_2^T T_2 \\ \mathbf{l}_3^T R_3 \widehat{\mathbf{l}_1} & \mathbf{l}_3^T T_3 \end{bmatrix} \leq 1 \tag{11.33}$$

where $\mathbf{l}_i \in \mathbb{R}^3, i = 1, 2, 3$ are the three coimages of the line, and (R_2, T_2) and (R_3, T_2) are the relative motions among the three views.

11.3.2 Inclusion of hyperplanes

Now let us investigate the relationships between two hyperplanes. Without loss of generality, we may assume $d_1 \geq d_2$ and assume that an H^{d_2} hyperplane belongs to an H^{d_1} hyperplane in $\mathbb{R}^n$: $H^{d_2} \subseteq H^{d_1}$. So the image of H^{d_1} contains that of H^{d_2}. And if $\mathbf{s} \in \mathbb{R}^{(k+1)\times(d_1+1)}$ is the image of H^{d_1} and $\mathbf{x} \in \mathbb{R}^{(k+1)\times(d_2+1)}$ is the image of H^{d_2} (relative to some camera frame), we have

$$\mathbf{x}^T \mathbf{s}^\perp = 0. \tag{11.34}$$

Two new cases may arise from this relationship:
Case One. Consider the matrix

$$C_h^1 = \begin{bmatrix} (\mathbf{s}_1^\perp)^T & 0 \\ (\mathbf{s}_2^\perp)^T \bar{R}_2 & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d_1)]\times(n+1)}. \tag{11.35}$$

According to (11.27), we have

$$k - d_1 \leq \operatorname{rank}(C_h^1) \leq n - d_1. \tag{11.36}$$

Multiplying C_h^1 on the right by the following matrix

$$D_{h1}^1 = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_1^\perp & 0 \\ 0 & 0 & I_{(n-k)\times(n-k)} \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$

yields the matrix

$$C_h^1 D_{h1}^1 = \begin{bmatrix} 0 & (\mathbf{s}_1^\perp)^T \mathbf{x}_1^\perp & 0 \\ (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{x}_1 & (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{x}_1^\perp & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{x}_1 & (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{x}_1^\perp & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d_1)]\times(n+1)}. \tag{11.37}$$

Let

$$M_h^1 \doteq \begin{bmatrix} (\mathbf{s}_2^\perp)^T \bar{R}_2 \mathbf{x}_1 & (\mathbf{s}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T \bar{R}_m \mathbf{x}_1 & (\mathbf{s}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[(m-1)(k-d_1)]\times(n-k+d_2)}. \tag{11.38}$$

Since $(\mathbf{s}_1^\perp)^T \mathbf{x}_1^\perp$ is of rank at least $k - d_1$, we must have

$$0 \leq \operatorname{rank}(M_h^1) \leq \operatorname{rank}(C_h^1) - (k - d_1) \leq n - k. \tag{11.39}$$

Case Two. Now we consider the matrix

$$C_h^2 = \begin{bmatrix} (\mathbf{x}_1^\perp)^T & 0 \\ (\mathbf{x}_2^\perp)^T \bar{R}_2 & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d_2)]\times(n+1)}. \tag{11.40}$$

According to (11.27), it has the property

$$\operatorname{rank}(C_h^2) \leq n - d_2. \tag{11.41}$$

Since $H^{d_2} \subseteq H^{d_1}$, we may choose $\mathbf{x}$ to be a sub-matrix of $\mathbf{s}$ and choose $\mathbf{s}^\perp$ to be a sub-matrix of $\mathbf{x}^\perp$. Hence, the following sub-matrix of C_h^2

$$\bar{C}_h^2 = \begin{bmatrix} (\mathbf{s}_1^\perp)^T & 0 \\ (\mathbf{x}_2^\perp)^T \bar{R}_2 & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[m(k-d_2)-(d_1-d_2)]\times(n+1)} \tag{11.42}$$

satisfies

$$k - d_1 \leq \operatorname{rank}(\bar{C}_h^2) \leq \operatorname{rank}(C_h^2) \leq n - d_2. \tag{11.43}$$

Now multiplying $\bar{C}_h^2$ on the right by the following matrix

$$D_{h1}^2 = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_1^\perp & 0 \\ 0 & 0 & I_{(n-k)\times(n-k)} \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$

yields the matrix

$$\bar{C}_h^2 D_{h1}^2 = \begin{bmatrix} 0 & (\mathbf{s}_1^\perp)^T \mathbf{s}_1^\perp & 0 \\ (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{s}_1 & (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{s}_1^\perp & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{s}_1 & (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{s}_1^\perp & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix}. \tag{11.44}$$

Let

$$M_h^2 \doteq \begin{bmatrix} (\mathbf{x}_2^\perp)^T \bar{R}_2 \mathbf{s}_1 & (\mathbf{x}_2^\perp)^T \bar{T}_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T \bar{R}_m \mathbf{s}_1 & (\mathbf{x}_m^\perp)^T \bar{T}_m \end{bmatrix} \in \mathbb{R}^{[(m-1)(k-d_2)]\times(n-k+d_1+1)}. \tag{11.45}$$

Since $(\mathbf{s}_1^\perp)^T \mathbf{s}_1^\perp$ is of full rank $(k - d_1)$, then we have

$$0 \leq \operatorname{rank}(M_h^2) = \operatorname{rank}(\bar{C}_h^2) - (k - d_1) \leq (n - k) + (d_1 - d_2). \tag{11.46}$$

Definition 11.1 (Formal multiple view matrix). *We define a* multiple view matrix M *as*

$$M \doteq \begin{bmatrix} (D_2^\perp)^T \bar{R}_2 D_1 & (D_2^\perp)^T \bar{T}_2 \\ (D_3^\perp)^T \bar{R}_3 D_1 & (D_3^\perp)^T \bar{T}_3 \\ \vdots & \vdots \\ (D_m^\perp)^T \bar{R}_m D_1 & (D_m^\perp)^T \bar{T}_m \end{bmatrix} \tag{11.47}$$

where the D_i's and $D_i^\perp$'s stand for images and co-images of some hyperplanes respectively. The actual values of D_i's and $D_i^\perp$'s are to be determined in context.

Therefore, we have proved the following statement which further generalizes Theorem 11.1:

Theorem 11.2 (Rank condition with inclusion). *Consider a d_2-dimensional hyperplane H^{d_2} belonging to a d_1-dimensional hyperplane H^{d_1} in $\mathbb{R}^n$. m images $\mathbf{x}_i \in \mathbb{R}^{(k+1)\times(d_2+1)}$ of the H^{d_2} and m images $\mathbf{s}_i \in \mathbb{R}^{(k+1)\times(d_1+1)}$ of the H^{d_1} relative to the i^{th} $(i = 1, \ldots, m)$ camera frame are given and the relative transformation from the 1^{st} camera frame to the i^{th} is (R_i, T_i) with $R_i \in \mathbb{R}^{n\times n}$ and $T_i \in \mathbb{R}^n, i = 2, \ldots, m$. Let the D_i's and $D_i^\perp$'s in the multiple view matrix M have the following values:*

$$\begin{cases} D_i^\perp \doteq \mathbf{x}_i^\perp \in \mathbb{R}^{(k+1)\times(k-d_2)} & \text{or} \quad \mathbf{s}_i^\perp \in \mathbb{R}^{(k+1)\times(k-d_1)}, \\ D_i \doteq \mathbf{x}_i \in \mathbb{R}^{(k+1)\times(d_2+1)} & \text{or} \quad \mathbf{s}_i \in \mathbb{R}^{(k+1)\times(d_1+1)}. \end{cases} \tag{11.48}$$

Then for all possible instances of the matrix M, we have the two cases:

1. Case one: *If $D_1 = \mathbf{s}_1$ and $D_i^\perp = \mathbf{x}_i^\perp$ for some $i \geq 2$, then*

$$\boxed{\textit{rank}(M) \leq (n-k) + (d_1 - d_2),}$$

2. Case two: *Otherwise,*

$$\boxed{0 \leq \textit{rank}(M) \leq n - k.}$$

Since $\text{rank}(AB) \geq (\text{rank}(A) + \text{rank}(B) - n)$ for all $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{n\times k}$, we have

$$\text{rank}[(D_i^\perp)^T \bar{R}_i D_1] \geq (d_1 - d_2)$$

if the matrix $\bar{R}_i \in \mathbb{R}^{(k+1)\times(k+1)}$ is full rank for some $i \geq 2$. So the rank condition for the case one can be improved with a tight lower bound

$$\boxed{(d_1 - d_2) \leq \text{rank}(M) \leq (n-k) + (d_1 - d_2).}$$

This theorem is very useful since it applies to many cases which are of practical importance.

Example 11.3 (Three views of a point on a line). *When $n = 3, k = 2, m = 3$ and $d_1 = 1, d_2 = 0$, it gives the known result in computer vision on three views of a point on a line:*

$$1 \leq \textit{rank} \begin{bmatrix} \widehat{\mathbf{x}_2} R_2 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_2} T_2 \\ \widehat{\mathbf{x}_3} R_3 \widehat{\mathbf{l}_1} & \widehat{\mathbf{x}_3} T_3 \end{bmatrix} \leq 2 \tag{11.49}$$

where $\mathbf{x}_i \in \mathbb{R}^3$ and $\mathbf{l}_i \in \mathbb{R}^3$ are the images and coimages of the point and line respectively. The rank condition is equivalent to what is known as the "line-point-point" relation.

The above theorem can also be easily generalized to any set of cascading hyperplanes:

$$H^{d_l} \subseteq H^{d_{l-1}} \subseteq \cdots \subseteq H^{d_1}$$

for some $l \in \mathbb{Z}_+$. We omit the details for simplicity.

11.3.3 Intersection of hyperplanes

Consider two hyperplanes H^{d_1} and H^{d_2} both containing a third hyperplanes

$$H^{d_3} \subseteq H^{d_1} \cap H^{d_2}.$$

Besides the rank conditions given by Theorem 11.2, some extra rank conditions hold for the mixed images of all three hyperplanes. Let $\mathbf{x}_i \in \mathbb{R}^{(k+1)\times(d_3+1)}$ be the images of the H^{d_3}, $\mathbf{r}_i \in \mathbb{R}^{(k+1)\times(d_2+1)}$ be those of the H^{d_2} and $\mathbf{s}_i \in \mathbb{R}^{(k+1)\times(d_1+1)}$ be those of the H^{d_1} relative to the i^{th} $(i = 1, \ldots, m)$ camera frame. Without loss of generality, let $\mathbf{x}_i$ be a sub-matrix of $\mathbf{r}_i$ and $\mathbf{s}_i$, and on the other hand, let $\mathbf{r}_i^\perp$ and $\mathbf{s}_i^\perp$ be sub-matrices of $\mathbf{x}_i^\perp$ for $i = 1, \ldots, m$. Then it is straight forward to prove (in a manner similar to the proofs in the above) the following rank conditions for these three hyperplanes.

Theorem 11.3 (Rank condition with intersection). *Consider hyperplanes H^{d_1}, H^{d_2}, and H^{d_3} with $H^{d_3} \subseteq H^{d_1} \cap H^{d_2}$. Given their m images relative to m camera frames as above, let the D_i's and $D_i^\perp$'s in the multiple view matrix M have the following values:*

$$\begin{cases} D_i^\perp \doteq \mathbf{x}_i^\perp, \ \mathbf{r}_i^\perp, \ \textit{or} \ \mathbf{s}_i^\perp, \\ D_1 \doteq \mathbf{x}_1. \end{cases} \tag{11.50}$$

Then we have:

$$\boxed{0 \leq \textit{rank}(M) \leq (n-k).}$$

It is straight forward to generalize this theorem to a hyperplane which is in the intersection of more than two hyperplanes:

$$H^{d_l} \subseteq H^{d_{l-1}} \cap \cdots \cap H^{d_1}$$

for some $l \in \mathbb{Z}_+$. Simply allow the matrix $D_i^\perp$ to take the co-image of any hyperplane involved and the rank condition on M always holds.

11.3.4 Restriction to a hyperplane

In practice, there are situations when all hyperplanes being observed belong to an ambient hyperplane in $\mathbb{R}^n$, and the location of the ambient hyperplane relative to the world reference frame is fixed. Here we no longer need the assumption that the dimension of this ambient hyperplane is lower than that of the image plane. It is not its image that we are interested in here. However, as in Sections 11.3.2 and 11.3.3, we still assume that all the other hyperplanes whose images are taken do have dimension lower than that of the image plane. This extra knowledge on the existence of such an ambient hyperplane is supposed to impose extra constraints on the multiple images of any hyperplane in the ambient hyperplane.

In general, a known d-dimensional hyperplane H^d in $\mathbb{R}^n$ can be described by a full-rank $(n-d)\times(n+1)$ matrix $\bar{\Pi}\in\mathbb{R}^{(n-d)\times(n+1)}$ such that any point $p\in H^d$ satisfies

$$\bar{\Pi}\mathbf{X}=0 \tag{11.51}$$

where $\mathbf{X}\in\mathbb{R}^{n+1}$ is the homogeneous coordinate of the point p. We call the matrix $\bar{\Pi}$ the *homogeneous representation* for H^d. Of course, such a representation is not unique – any two matrices $\bar{\Pi}_1$ and $\bar{\Pi}_2$ with the same kernel give rise to the same hyperplane. For convenience, we usually divide the $(n-d)\times(n+1)$ matrix $\bar{\Pi}$ into two parts:

$$\bar{\Pi}=[\bar{\Pi}^1,\bar{\Pi}^2],\quad \text{with}\quad \bar{\Pi}^1\in\mathbb{R}^{(n-d)\times(k+1)},\quad \bar{\Pi}^2\in\mathbb{R}^{(n-d)\times(n-k)}. \tag{11.52}$$

Definition 11.2 (Formal extended multiple view matrix). *We define a* multiple view matrix M *as*

$$M\doteq\begin{bmatrix}(D_2^\perp)^T\bar{R}_2D_1 & (D_2^\perp)^T\bar{T}_2\\ (D_3^\perp)^T\bar{R}_3D_1 & (D_3^\perp)^T\bar{T}_3\\ \vdots & \vdots\\ (D_m^\perp)^T\bar{R}_mD_1 & (D_m^\perp)^T\bar{T}_m\\ \bar{\Pi}^1D_1 & \bar{\Pi}^2\end{bmatrix} \tag{11.53}$$

where the D_i's and $D_i^\perp$'s stand for images and co-images of some hyperplanes respectively. The actual values of D_i's and $D_i^\perp$'s are to be determined in context.

Then, the rank conditions given in Theorems 11.2 and 11.3 need to be modified to the following two corollaries, respectively:

Corollary 11.1 (Rank condition with restricted inclusion). *Consider hyperplanes $H^{d_2}\subseteq H^{d_1}\subseteq H^d$ in $\mathbb{R}^n$, where $d_1,d_2<k$ and $d<n$. The hyperplane H^d is described by a matrix $\bar{\Pi}\in\mathbb{R}^{(n-d)\times(n+1)}$ (relative to the first camera frame). Then the rank conditions in Theorem 11.2 do not change at all when the multiple view matrix there is replaced by the extended multiple view matrix.*

Corollary 11.2 (Rank condition with restricted intersection). *Consider hyperplanes H^{d_1}, H^{d_2}, and H^{d_3} with $H^{d_3}\subseteq H^{d_1}\cap H^{d_2}$ which all belong to an ambient hyperplane H^d, where $d_1,d_2,d_3<k$ and $d<n$. The hyperplane H^d is described by a matrix $\bar{\Pi}\in\mathbb{R}^{(n-d)\times(n+1)}$ (relative to the first camera frame). Then the rank conditions in Theorem 11.3 do not change at all when the multiple view matrix there is replaced by the extended multiple view matrix.*

One can easily prove these corollaries by modifying the proofs for the theorems accordingly.

Example 11.4 (Planar homography). *Consider the standard perspective projection from $\mathbb{R}^3$ to $\mathbb{R}^2$. Suppose, as illustrated in Figure 11.3, all*

feature points lie on certain plane in $\mathbb{R}^3$*, which can be described by some vector* $\pi \in \mathbb{R}^4$*. Then the* M *matrix associated to two images* $\mathbf{x}_1, \mathbf{x}_2$ *of a point* p *is*

$$M = \begin{bmatrix} \widehat{\mathbf{x}_2} T & \widehat{\mathbf{x}_2} R \mathbf{x}_1 \\ \pi^2 & \pi^1 \mathbf{x}_1 \end{bmatrix} \tag{11.54}$$

where $\pi^1 \in \mathbb{R}^3, \pi^2 \in \mathbb{R}$ *and* $[\pi^1, \pi^2] = \pi$*. It is easy to show that rank*$(M) \leq 1$ *is equivalent to the equation*

$$\widehat{\mathbf{x}_2} \left(R - \frac{1}{\pi^2} T \pi^1 \right) \mathbf{x}_1 = 0. \tag{11.55}$$

This equation in computer vision is known as homography *between two views of a planar scene. The matrix in the middle* $H = \left(R - \frac{1}{\pi^2} T \pi^1\right) \in \mathbb{R}^{3\times 3}$ *is correspondingly called the* homography matrix*. Knowing the images of more than four points, this equation can be used to recover both the camera motion* (R, T) *and the plane* π*.*

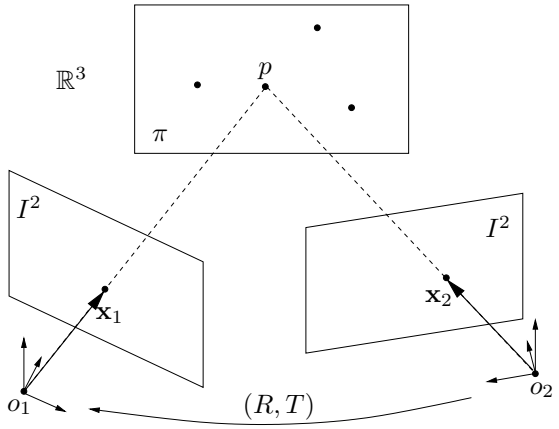

Figure 11.3. Two views of points on a plane $\pi \subset \mathbb{R}^3$. For a point $p \in \pi$, its two (homogeneous) images are the two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ with respect to the two vantage points.

11.4 Geometric interpretation of rank conditions

In the previous section, we have seen that multiple images of hyperplanes must satisfy certain algebraic conditions, the so-called rank conditions. In this section, we would like to address the opposite question: Given such an algebraic condition, what does it mean geometrically? More specifically, we like to study:

1. Given a multiple view matrix M for a hyperplane, can we characterize the hyperplane? If not, to what extent can we characterize it?

2. Given a multiple view matrix M and its rank, what can we tell about the geometric configuration of the camera motions and the hyperplanes involved?

11.4.1 Multiple view stereopsis in n-dimensional space

Given multiple images of a hyperplane $H^d \subset \mathbb{R}^n$, according to Section 11.3.1, we have $\text{rank}(M_h) = [\text{rank}(C_h) - (k-d)]$. Then the dimension of the kernel of C_h is $[(n+1) - \text{rank}(C_h)]$. Hence if $\text{rank}(M_h)$ reaches its upper bound $n-k$, i.e. $\text{rank}(C_h)$ reaches its upper bound $n-d$, the kernel of C_h has dimension $d+1$. From previous discussions we know that this kernel is exactly the homogeneous representation of H^d. Therefore, the upper bound of the rank means that the hyperplane can be determined uniquely. If $\text{rank}(C_h) = l < n-d$, then we can only determine the hyperplane H^d up to an $(n-l)$-dimensional hyperplane as the kernel of C_h.

The structure of the matrix C_h provides another way of interpreting the kernel of C_h. C_h is composed of m blocks of $(k-d) \times (n+1)$ matrices $\begin{bmatrix}(\mathbf{s}_i^\perp)^T \bar{R}_i & (\mathbf{s}_i^\perp)^T \bar{T}_i\end{bmatrix}$, $i = 1, \ldots, m$. The kernel of each block is the preimage F_i of $\mathbf{s}_i$. Hence the kernel of C_h is the intersection of F_i's. The dimension of F_i is $(n-k+d+1)$ for all i. Define $G_1 = F_1$, $G_i = G_{i-1} \cap F_i$ for $i = 2, \ldots, m$. Then $\dim(G_i) \leq \dim(G_{i-1})$ for $i = 2, \ldots, m$ and G_m is the kernel of C_h. For the motions $\Pi_1, \ldots, \Pi_m$, if $\dim(G_i) < \dim(G_{i-1})$ for $i = 2, \ldots, m$, then call the set of motions as *independent motions* and the set of views (or images) as *independent views*. Otherwise, we call the set of motions as *dependent motions*. In the following discussions, we only consider independent motions.

For any set of independent motions, the corresponding G_1 has dimension $(n-k+d+1)$. Then in order to determine uniquely H^d, it is desirable to have $\dim(G_m) = d+1$. Hence the reduction of dimension from G_1 to G_m is $n-k$. For $i > 1$, the dimension reduction for the i^{th} view is given by

$$\begin{aligned} 1 &\leq \dim(G_{i-1}) - \dim(G_i) = \dim(G_{i-1} + F_i) - \dim(F_i) \\ &\leq (n+1) - (n-k+d+1) = k-d. \end{aligned}$$

Thus, in order to determine uniquely H^d, we need at least $m = \lceil \frac{n-k}{k-d} \rceil + 1$ views with independent motions. However, this is the "optimal" case such that each view can induce maximum dimension reduction of G_i. In general, the number required to determine uniquely H^d can be larger than this number. The maximum number of independent views required is $m = (n-k+1)$ in which case each view only contributes to a one-dimensional reduction of G_i. For example, in the special case for point features, we have $d = 0$. Hence the minimum number of independent views required is

then $\lceil \frac{n-k}{k} \rceil + 1$. When $n = 3, k = 2$, this number reduces to 2 which is well-known for 3-D stereopsis.

11.4.2 A special case

In addition to above general statements regarding the geometry that rank conditions may imply, any particular instance of the multiple view matrix in fact encodes some special information about the geometric configuration of the camera motions and the hyperplanes involved. It is impractical to study all possible instances of the multiple view matrix. Here we demonstrate how this can be done for a special case when $k = n-1$.[6] All the rank conditions in Theorem 11.1, 11.2 and 11.3 are therefore simplified accordingly. We discuss their geometric implications.

An individual hyperplane

Let us first study the multiple view matrix in Theorem 11.1. So there are two sub-cases only for the rank:

$$\text{rank}(M_h) = 1, \quad \text{or} \quad \text{rank}(M_h) = 0. \tag{11.56}$$

Suppose its rank is 1 for an M_h associated to some d-dimensional hyperplane H^d in $\mathbb{R}^n$. Let $\mathbf{s}_i, i = 1, \ldots, m$ be the m images of H^d relative to m camera frames specified by the relative transformations $(R_i, T_i), i = 2, \ldots, m$ (from the 1^{st} frame to the i^{th}). Consider the matrix C_h defined in Section 11.3.1

$$C_h = \begin{bmatrix} (\mathbf{s}_1^\perp)^T & 0 \\ (\mathbf{s}_2^\perp)^T R_2 & (\mathbf{s}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T R_m & (\mathbf{s}_m^\perp)^T T_m \end{bmatrix} \in \mathbb{R}^{[m(n-d-1)] \times (n+1)}. \tag{11.57}$$

Then from the derivation in Section 11.3.1, we have the following rank inequality

$$(n-d) \geq \text{rank}(C_h) = \text{rank}(M_h) + (n-d-1) = (n-d). \tag{11.58}$$

Hence the rank of C_h is exactly $n-d$. It kernel is a $(d+1)$-dimensional subspace in the homogeneous space $\mathbb{P}^n$, which gives rise to the homogeneous coordinates (relative to the first camera frame) of all points on the hyperplane H^d. Hence C_h itself is a valid homogeneous representation for H^d, although it may consist of many redundant rows. In any case, when the rank of M_h is 1, the location of the hyperplane H^d (relative to the first camera frame) is determined uniquely.

[6] This is partly because when $n = 3, k = 3 - 1 = 2$, it is the classic case that we have studied in previous chapters.

Suppose the rank of M_h is 0, correspondingly the rank of the matrix C_h is $(n-d-1)$ instead of $(n-d)$. Then the kernel of the matrix C_h is of dimension $(d+2)$ in $\mathbb{P}^n$. This gives rise to a $(d+1)$-dimensional hyperplane H^{d+1} in $\mathbb{R}^n$. To see what this hyperplane is, let us define the matrix:

$$K_h = \begin{bmatrix} \mathbf{s}_1 & 0 \\ 0 & 1 \end{bmatrix} \quad \in \mathbb{R}^{(n+1)\times(d+2)}. \tag{11.59}$$

Then it is immediate that

$$\operatorname{rank}(M_h) = \operatorname{rank}(C_h K_h). \tag{11.60}$$

Then $\operatorname{rank}(M_h) = 0$ simply means that $\operatorname{range}(K_h) \subseteq \ker(C_h)$. Notice that the range of the matrix K_h is a $(d+2)$-dimensional subspace (relative to the first camera frame) in $\mathbb{P}^n$ which represents homogeneously the $(d+1)$-dimensional hyperplane spanned by H^d and the center of the first camera frame. Then the location of the originally observed H^d is only determined up to this H^{d+1} – its exact location within H^{d+1} cannot be known from its given m images. Since $\operatorname{rank}(M_h) = 0$, it also means the entire matrix M_h is zero, which further reveals some special relationships between all the camera frames and the images. It is easy to see that M_h is zero if and only if each camera center and the hyperplane H^d span the same $(d+1)$-dimensional hyperplane in $\mathbb{R}^n$.

Inclusion

We now study the multiple view matrix in Theorem 11.2. Here we know the two hyperplanes $H^{d_2} \subseteq H^{d_1}$ and would like to know what the rank condition implies about their general location relative to the camera frames. If the images involved in a particular multiple view matrix are all from the same hyperplane, there is essentially the "one hyperplane" case which has been discussed in the above section. The two interesting cases left are the two matrices M_h^1 and M_h^2 defined in Section 11.3.2:

$$M_h^1 \doteq \begin{bmatrix} (\mathbf{s}_2^\perp)^T R_2 \mathbf{x}_1 & (\mathbf{s}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T R_m \mathbf{x}_1 & (\mathbf{s}_m^\perp)^T T_m \end{bmatrix} \quad \in \mathbb{R}^{[(m-1)(n-d_1-1)]\times(d_2+2)},$$

$$M_h^2 \doteq \begin{bmatrix} (\mathbf{x}_2^\perp)^T R_2 \mathbf{s}_1 & (\mathbf{x}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T R_m \mathbf{s}_1 & (\mathbf{x}_m^\perp)^T T_m \end{bmatrix} \quad \in \mathbb{R}^{[(m-1)(n-d_2-1)]\times(d_1+2)}.$$

According to the rank conditions given in Theorem 11.2, we have:

$$0 \leq \operatorname{rank}(M_h^1) \leq 1, \quad (d_1 - d_2) \leq \operatorname{rank}(M_h^2) \leq 1 + (d_1 - d_2).$$

If $\operatorname{rank}(M_h^1) = 1$, then C_h^1 defined in Section 11.3.2 has $\operatorname{rank}(C_h^1) = n - d_1$. Hence the kernel of the matrix C_h^1 uniquely determines the hyperplane H^{d_1}

– it location relative to the first camera frame. Also since $\mathbf{s}_i^\perp$ is a sub-matrix of $\mathbf{x}_i^\perp$, $i = 1, \ldots, m$, we have:

$$\text{rank}\begin{bmatrix} (\mathbf{s}_2^\perp)^T R_2 \mathbf{x}_1 & (\mathbf{s}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{s}_m^\perp)^T R_m \mathbf{x}_1 & (\mathbf{s}_m^\perp)^T T_m \end{bmatrix} \quad \leq \quad \text{rank}\begin{bmatrix} (\mathbf{x}_2^\perp)^T R_2 \mathbf{x}_1 & (\mathbf{x}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T R_m \mathbf{x}_1 & (\mathbf{x}_m^\perp)^T T_m \end{bmatrix}.$$

So the rank of the second matrix must be 1, hence the location of the hyperplane H^{d_2} is uniquely determined according to the discussion in Section 11.4.2. However, if $\text{rank}(M_h^1) = 0$, we only know

$$(n - d_1 - 1) \leq \text{rank}(C_h^1) \leq (n - d_1). \tag{11.61}$$

In this case, nothing definite can be said about the uniqueness of H^{d_1} from the rank test on this particular multiple view matrix. On the other hand, the entire matrix M_h^1 being zero indeed reveals some degeneracy about the configuration: All the camera centers and H^{d_1} span a $(d_1 + 1)$-dimensional hyperplane in $\mathbb{R}^n$ since $(\mathbf{s}_i^\perp)^T T_i = 0$ for all $i = 2, \ldots, m$; the only case that the exact location of H^{d_1} is determinable (from this particular rank test) is when all the camera centers are actually in H^{d_1}. Furthermore, we are not able to say anything about the location of H^{d_2} in this case either, except that it is indeed in H^{d_1}.

Now for the matrix M_h^2, if $\text{rank}(M_h^2) = (d_1 - d_2 + 1)$, then the matrix C_h^2 defined in Section 11.3.2 has $\text{rank}(C_h^2) = n - d_2$. Hence the kernel of the matrix C_h^2 uniquely determines the hyperplane H^{d_2} – it location relative to the first camera frame. Or, simply use the following rank inequality (from comparing the numbers of columns):

$$\text{rank}\begin{bmatrix} (\mathbf{x}_2^\perp)^T R_2 \mathbf{s}_1 & (\mathbf{x}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T R_m \mathbf{s}_1 & (\mathbf{x}_m^\perp)^T T_m \end{bmatrix} \quad \leq \quad \text{rank}\begin{bmatrix} (\mathbf{x}_2^\perp)^T R_2 \mathbf{x}_1 & (\mathbf{x}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T R_m \mathbf{x}_1 & (\mathbf{x}_m^\perp)^T T_m \end{bmatrix} + (d_1 - d_2).$$

So the second matrix will have rank 1, hence the location of H^{d_2} is unique. However, in general we are not able to say anything about whether H^{d_1} is uniquely determinable from such a rank test, except that it contains H^{d_2}. Now, if $\text{rank}(M_h^2) = (d_1 - d_2)$, we only know

$$(n - d_2 - 1) \leq \text{rank}(C_h^2) \leq (n - d_2). \tag{11.62}$$

Then this particular rank test does not imply the uniqueness of the hyperplane H^{d_2}. From the matrix M_h^2 being of rank $(d_1 - d_2)$, we can still derive that all the camera centers and H^{d_1} span a $(d_1 + 1)$-dimensional hyperplane H^{d_1+1} since $(\mathbf{x}_i^\perp)^T T_i$ is linearly dependent of $(\mathbf{x}_i^\perp)^T R_i \mathbf{x}_1$ for $i = 2, \ldots, m$. So H^{d_1} is in general not determinable unless all the camera centers lie in H^{d_1}. If it is the special case that all the camera centers actually lie in H^{d_2}, the location of H^{d_2} is then determinable. If some camera centers are off the hyperplane H^{d_2} but still in the hyperplane H^{d_1+1}, we simply cannot

tell the uniqueness of H^{d_2} from this particular rank test. Examples for either scenario exist. Generally speaking, the bigger the difference between the dimensions d_1 and d_2, the less information we will get about H^{d_2} from the rank test on M_h^2 alone. Due to this reason, in practice, other types of multiple view matrices are preferable.

We summarize our discussion in this section in Table 11.1.

Matrix	rank(M_h^1)	rank(M_h^1)	rank(M_h^2)	rank(M_h^2)
Rank values	1	0	$d_1 - d_2 + 1$	$d_1 - d_2$
H^{d_1}	unique	$\subset H^{d_1+1}$	$\supseteq H^{d_2}$	$\subset H^{d_1+1}$
H^{d_2}	unique	$\subseteq H^{d_1}$	unique	$\subseteq H^{d_1}$
Camera centers	$\subset \mathbb{R}^n$	$\subset H^{d_1+1}$	$\subset \mathbb{R}^n$	$\subset H^{d_1+1}$

Table 11.1. Locations of hyperplanes or camera centers implied by each single rank test. Whenever it is well-defined, H^{d_1+1} is the (d_1+1)-dimensional hyperplane spanned by the H^{d_1} and all the camera centers.

Intersection

Finally, we study the multiple view matrix in Theorem 11.3. Here we know three hyperplanes are related by $H^{d_3} \subseteq H^{d_1} \cap H^{d_2}$. Let us first study the case $\text{rank}(M) = 1$. Since $\mathbf{r}_i^\perp$ and $\mathbf{s}_i^\perp$ are both sub-matrices of $\mathbf{x}_i^\perp$, $i = 1, \ldots, m$, we have:

$$\text{rank} \begin{bmatrix} (D_2^\perp)^T R_2 \mathbf{x}_1 & (D_2^\perp)^T T_2 \\ \vdots & \vdots \\ (D_m^\perp)^T R_m \mathbf{x}_1 & (D_m^\perp)^T T_m \end{bmatrix} \quad \leq \quad \text{rank} \begin{bmatrix} (\mathbf{x}_2^\perp)^T R_2 \mathbf{x}_1 & (\mathbf{x}_2^\perp)^T T_2 \\ \vdots & \vdots \\ (\mathbf{x}_m^\perp)^T R_m \mathbf{x}_1 & (\mathbf{x}_m^\perp)^T T_m \end{bmatrix}$$

where $D_i^\perp = \mathbf{x}_i^\perp$, $\mathbf{r}_i^\perp$, or $\mathbf{s}_i^\perp$. Therefore, the rank of the second matrix in the above rank inequality is 1. That is the hyperplane H^{d_3} can always be determined. As before, nothing really can be said about the other two hyperplanes except that they both contain H^{d_3}. Now if $\text{rank}(M) = 0$, then nothing really can be said about the locations of any hyperplane, except that all the camera centers and the hyperplanes H^{d_1}, H^{d_2} span a $(d_1 + d_2 + 1)$-dimensional hyperplane – a degenerate configuration. The analysis goes very much the same for any number of hyperplanes which all intersect at H^{d_3}.

11.4.3 Degenerate motions

In previous sections, we have assumed that the hyperplane H^d relative to the camera (and its image plane) is of general position. That essentially means that the hyperplane and its image have the same dimension: $q = d$ when $d \leq k$. This assumption has to a large extent simplified our analysis. However, in some rare degenerate situations, the image of a hyperplane could be of strictly lower dimension, i.e. $q < d$. Such a case is illustrated

by Figure 11.4. Degenerate situations happen in high dimensional spaces in a similar fashion. More specifically, let K be the orthogonal complement to the image space I^{k+1} in $\mathbb{R}^n$. Still use H^{d+1} to denote the subspace spanned by the hyperplane H^d and the camera center. Then we have $d = q + \dim(H^{d+1} \cap K)$. So a degenerate situation occurs whenever the intersection

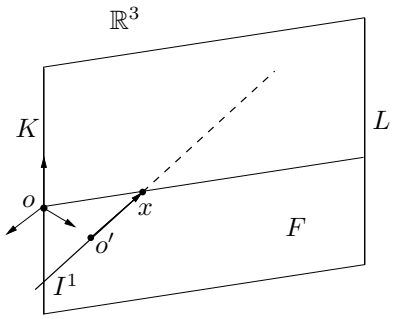

Figure 11.4. A degenerate case: the image of a line is a point under the perspective projection from $\mathbb{R}^3$ to $\mathbb{R}^1 (\doteq I^1)$. Let K be the subspace orthogonal to the plane spanned by o and I^1. Then this happens when L is parallel to K.

of K and H^{d+1} is non-trivial. A potential problem that may arise with a degenerate situation is that the dimension of the image of a hyperplane may vary from view to view. Using these degenerate images in the multiple view matrix may result in a looser bound on the rank conditions. Suppose the first image of H^d has a dimension $q_1 < d$. Then all the theorems about rank conditions in preceding sections need to be changed accordingly. For example, Theorem 11.1 becomes:

Corollary 11.3 (Rank condition with degenerate motions). *Given m images $\mathbf{s}_1, \ldots, \mathbf{s}_m$ and co-images $\mathbf{s}_1^\perp, \ldots, \mathbf{s}_m^\perp$ of a d-dimensional hyperplane H^d in $\mathbb{R}^n$, the associated multiple view matrix M_h defined above satisfies*

$$\boxed{0 \leq \mathit{rank}(M_h) \leq (n-k) - (d-q_1)} \tag{11.63}$$

where q_1 is the dimension of the hyperplane of the image for H^d with respect to the 1^{st} camera frame.

Notice that since $(n-k) - (d-q_1) < (n-k)$, the rank condition says that we may only determine the hyperplane H^d up to an H^{2d-q_1} subspace (with respect to the first view). This is expected since a degenerate view is chosen as the reference. If we choose a different (non-degenerate) frame as the reference, the rank may vary and gives a better description of the overall configuration.

In view of this, we see that the rank of the corresponding matrix C_h (defined during the derivation of M in previous sections) in general are more

symmetric with respect to all images. However, the rank of C_h typically depends on the dimension of the hyperplane involved.[7] That makes it much harder to use in practice. For example, if the goals is to reconstruct a scene using all available features (including hyperplanes of all dimensions), any algorithm based on C_h needs to deal with different hyperplanes separately. The use of the multiple view matrix M would be much more uniform, despite its possible trouble with degeneracy.

11.5 Applications of the generalized rank conditions

The rank conditions for various multiple-view matrices in the classic case when $n = 3$ and $k = 2$ have been studied recently. These results then become special cases for the general rank conditions developed in this chapter. For example, for the m images of a point in the 3-D space, the corresponding multiple-view matrix M_p then satisfies $0 \leq M_p \leq n - k = 1$. Similarly for the line feature in 3-D, the multiple-view matrix also satisfies $0 \leq M_l \leq n - k = 1$. As we have mentioned before, the rank condition is going to be useful for any problem that has a setting of projecting a high-dimensional signal to low-dimensional ones. We discuss here two of those problems in computer vision.

11.5.1 Multiple view constraints for dynamical scenes

One important application of the rank conditions is for the analysis of the dynamical scene, which means that the scene contains moving objects with independent motions. Let us first see how this can be done in general. The trajectory of a moving point p (with respect to the world coordinate frame) can be described by a function $X(t) : \mathbb{R} \to \mathbb{R}^3$. If the function is smooth and analytic, we may express it in terms of Taylor expansion:

$$X(t) = X(0) + \sum_{j=1}^{\infty} \frac{X^{(j)}(0)}{j!} t^j. \tag{11.64}$$

In practice, we may approximate the series up to N^{th} order terms:

$$X(t) \approx X(0) + \sum_{j=1}^{N} \frac{X^{(j)}(0)}{j!} t^j. \tag{11.65}$$

[7] The rank of M could depend on the difference of the hyperplanes in the case of hyperplane inclusion. But that only occurs in Case one (see Theorem 11.2) which is usually of least practical importance.

Notice that when $N = 1$, we essentially assume that points are moving in constant velocity (with respect to the world frame); when $N = 2$, points are moving with a constant acceleration (with a parabolic trajectory). Suppose the relative motion from the i^{th} camera frame (at time t_i) to the world frame is $g_i = (R_i, T_i) \in SE(3)$. In homogeneous coordinates, the i^{th} (perspective) image of this point (at time t_i) is then given by

$$\lambda_i \mathbf{x}_i = P' g_i \mathbf{X}(t_i) = P' g_i \left(\mathbf{X}(0) + \sum_{j=1}^{N} \frac{\mathbf{X}^{(j)}(0)}{j!} t_i^j \right) \tag{11.66}$$

where $P' \in \mathbb{R}^{3\times 4}$. If we define a new projection matrix to be

$$\Pi_i = [R_i t_i^N \ \ldots \ R_i t_i \ R_i, T_i] \in \mathbb{R}^{3\times(3N+4)} \tag{11.67}$$

and a new coordinate associated to the point p in $\mathbb{R}^{3n}$ to be

$$\bar{\mathbf{X}} = [X^{(N)}(0)^T \ \ldots \ X'(0)^T, \mathbf{X}^T]^T \in \mathbb{R}^{3N+4}, \tag{11.68}$$

then the above equation becomes

$$\lambda_i \mathbf{x}_i = \Pi_i \bar{\mathbf{X}}, \quad \bar{\mathbf{X}} \in \mathbb{R}^{3N+4}, \mathbf{x}_i \in \mathbb{R}^3. \tag{11.69}$$

This is indeed a multiple-view projection from $\mathbb{R}^{3(N+1)}$ to $\mathbb{R}^2$ (in its homogeneous coordinates). The rank conditions then give basic constraints that can be further used to segment these features or reconstruct their coordinates. The choice of the *time-base* $(t^N, \ldots, t, 1)$ in the construction of Π above is not unique. If we change the time-base to be $(\sin(t), \cos(t), 1)$, it would allow us to embed points with elliptic trajectory into a higher dimensional linear space.

As a simple example, let us study in more detail a scene consisting of point features with each moving at a constant velocity. According to above discussion, this can be viewed as a projection from a six-dimensional space to a two-dimensional one. The six-dimensional coordinate consists of both a point's 3-D location and its velocity. We are interested in the set of algebraic (and geometric) constraints that govern its multiple images. Since now we have a projection from $\mathbb{R}^6$ to $\mathbb{R}^2$, for given m images, the multiple view matrix M_p (as defined in (11.22)) associated to a point p has a dimension $2(m-1) \times 5$ and satisfies the rank condition $0 \leq \text{rank}(M_p) \leq 6 - 2 = 4$. Then any 5×5 minor (involving images from 4 views) of this matrix has determinant zero. These minors exactly correspond to the only type of constraints discovered in [**?**] (using tensor terminology). However, the rank conditions provide many other types of constraints that are independent of the above. Especially, we know each point moving on a straight line in 3D – from the constant velocity assumption, it is then natural to consider the incidence condition that a point lies on a line. According to Theorem 11.2, for m images of a point on a line, if we choose D_1 to be the image of the point and $D_i^\perp$ to be the co-image of the line, the multiple view matrix M has a dimension $(m-1) \times 5$. M has $\text{rank}(M) \leq 4$. It is direct to check

that constraints from its minor being zero involve up to 6 (not 4) views. If we consider the rank conditions for other types of multiple view matrix M defined in Theorem 11.2, different constraints among points and lines can be obtained. Therefore, to complete and correct the result in [**?**] (where only the types of projection $\mathbb{R}^n \to \mathbb{R}^2, n = 4, \ldots, 6$ were studied), we have in general

Corollary 11.4 (Multilinear constraints for $\mathbb{R}^n \to \mathbb{R}^k$). *For projection from $\mathbb{R}^n$ to $\mathbb{R}^k$, non-trivial algebraic constraints involve up to $(n-k+2)$ views. The tensor associated to the $(n-k+2)$-view relationship in fact induces all the other types of tensors associated to smaller numbers of views.*

The proof directly follows from the rank conditions. In the classic case $n = 3, k = 2$, this corollary reduces to the well-known fact in computer vision that irreducible constraints exist up to triple-wise views, and furthermore, the associated (tri-focal) tensor induces all (bi-focal) tensors (i.e. the essential matrix) associated to pairwise views. In the case $n = 6, k = 2$, the corollary is consistent to our discussion above. Of course, besides embedding (11.69) of a dynamic scene to a higher dimensional space $\mathbb{R}^{3(N+1)}$, knowledge on the motion of features sometime allows us to embed the problem in a lower dimensional space. These special motions have been studied in [**?**], and our results here also complete their analysis.

11.5.2 Multiple views of a kinematic chain

Maybe one of the most important applications of the rank conditions is to describe objects that can be modeled as kinematic chains. These objects include any linked rigid body (or so-called articulated body) such as a walking human being or a robot arm. The key idea here is that each link (of the body) can be very well represented as a moving line (a 1-dimensional hyperplane); and each joint can be represented as a moving point where links intersect. The rigidity of the motion essentially allows us to embed points on the body as a hyperplane into a high dimensional space. The dimension of the hyperplane is exactly the degree of freedoms of the links. To demonstrate how this can work, we consider the simplest case: two links joined by a *resolute joint* with the first one fixed to the ground through another resolute joint, as shown in Figure 11.5. Without loss of generality, suppose the joints lie in the xy-plane (with respect to the world coordinate frame). Let $e_1 = [1, 0, 0]^T \in \mathbb{R}^3$ be the unit vector along the x-axis and $e_3 = [0, 0, 1] \in \mathbb{R}^3$ along the z-axis; $\omega_1, \omega_2 \in \mathbb{R}$ be the angular velocities of the first and second joints respectively; and $l_1, l_2 \in \mathbb{R}$ be the lengths of the two links. Assume the location of the first joint is $X_o \in \mathbb{R}^3$. Assume at time $t = 0$, the two joints start from the initial state $\theta_1 = \theta_2 = 0$.

Then a point on the first link has the coordinates $X_1(t) = X_o + r_1 \exp(\widehat{e_3}\omega_1 t)e_1$ where $r_1 \in \mathbb{R}$ is the distance from the point to the first

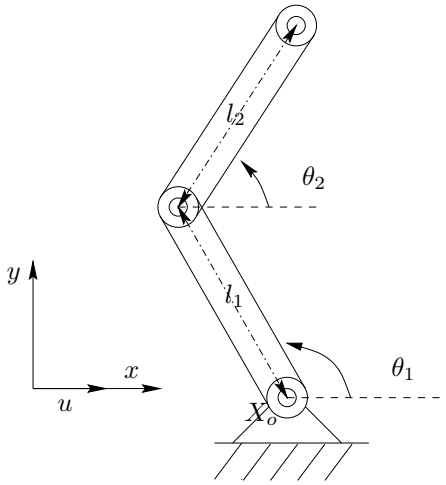

Figure 11.5. Two resolute joints.

joint. Using Rodrigues' formula we can rewrite this as

$$X_1(r_1,t) = [X_o + r_1(I + \widehat{e_3}^2)e_1] + [r_1\widehat{e_3}e_1]\cdot\sin(\omega_1 t) + [-r_1\widehat{e_3}^2 e_1]\cdot\cos(\omega_1 t).$$

Similarly for the second link, coordinates of a point with a distance $r_2 \in \mathbb{R}$ to the second joint are given by

$$X_2(r_2,t) = [X_1(l_1,t) + r_2(I + \widehat{e_3}^2)e_1] + [r_2\widehat{e_3}e_1]\cdot\sin(\omega_2 t) + [-r_2\widehat{e_3}^2 e_1]\cdot\cos(\omega_2 t).$$

If we choose the time-base $(\sin(\omega_1 t), \cos(\omega_1 t), \sin(\omega_2 t), \cos(\omega_2 t), 1)$ for the new projection matrix Π, we may embed the links into $\mathbb{R}^{15}$. Hence the projection of a point on the links is a projection from $\mathbb{R}^{15}$ to $\mathbb{R}^2$, and can be expressed in homogeneous coordinates:

$$\lambda(t)\mathbf{x}(t) = \Pi(t)\bar{\mathbf{X}}, \quad \bar{\mathbf{X}} \in \mathbb{R}^{16}, \mathbf{x} \in \mathbb{R}^3. \tag{11.70}$$

It is clear to see that points from each link lie on a one-dimensional hyperplane (a line) in the ambient space $\mathbb{R}^{15}$. These two links together span a two-dimensional hyperplane, unless $\omega_1 = \omega_2$ (i.e. the two links essentially move together as one rigid body). The rank conditions that we have for such a projection will be able to exploit the fact that the two links intersect at the second joint, and that points on the links belong to a two-dimensional hyperplane $Z = 0$ (w.r.t. the world coordinate frame).

In general, using similar techniques, we may embed a linked rigid body of multiple joints or links (maybe of other types) into a high dimensional space. If a linked rigid body consists of N links, in general we can embed its configuration into $\mathbb{R}^{O(6N)}$ – with the assumption of constant translation and rotation velocity. This makes perfect sense since each link potentially may introduce 6 degrees of freedom – three for rotation and three for translation. In principle, the rank conditions associated to the projection $\mathbb{R}^{O(6N)}$

to $\mathbb{R}^2$ then provide necessary constraints among links, joints. Recognizing the hyperplane to which all points belong tells what types of joints there are; and recovering the high-dimensional coordinates of each point allows to segment points to different links. For applications such as tracking walking human or controlling robot arms, resolute joints are to a large extent sufficient for the purpose. However, in the case of walking human, the joints are typically not fixed to the ground. Hence we need to model the base point X_o as a moving point too. Combining with results in the previous section, the resulting embedding is obvious.

11.6 Summary

In this chapter, we have studied algebraic and geometric relationships among multiple low-dimensional projections (also called images) of hyperplanes in a high-dimensional linear space. To a large extent, the work is a systematic generalization of classic multiple view geometry in computer vision to higher dimensional spaces. The abstract formulation of the projection model in this chapter allows us to treat uniformly a large class of projections that are normally encountered in many practical applications, such as perspective projection and orthographic projection. We have demonstrated in detail how incidence conditions among multiple images of multiple hyperplanes can be uniformly and concisely described in terms of certain rank conditions on the so-called multiple view matrix. These incidence conditions include: inclusion, intersection, and restriction (to a fixed hyperplane).

The significance of such rank conditions is multi-fold: 1. the multiple view matrix is the result of a systematic elimination of unknowns that are dependent on n-dimensional location of the hyperplanes involved. The rank conditions essentially describe incidence conditions in $\mathbb{R}^n$ in terms of only low dimensional images (of the hyperplanes) and relative configuration among the vantage points. These are the only constraints that are available for any reconstruction whose input data are the images only. 2. As an alternative to the multi-focal tensor techniques that are widely used in computer vision literature for studying multiple view geometry, the rank conditions provide a more complete, uniform and precise description of all the incidence conditions in multiple view geometry. They are therefore easier to use in reconstruction algorithms for multiple view analysis and synthesis. 3. A crucial observation is that the rank conditions are invariant under any transformations that preserve incidence relationships. Hence it does not discriminate against Euclidean, affine, projective or any other transformation groups, as long as they preserve incidence conditions of the hyperplanes involved. Since these transformations were traditionally studied separately in computer vision, the rank conditions provide a way

of unifying these efforts. 4. The generalization of multiple view geometry to high dimensional spaces has dramatically increased the range of potential applications outside computer vision. As we have shown, many computer vision problems that classic multiple view geometry is not able to handle can in fact be embedded as a multiple view geometry problem in a space of higher dimension. These generalized rank conditions can also be applied to many other problems in communications, signal processing and medical imaging, whenever the essential problem is to recover high dimensional objects from their low dimensional projections.

In this chapter, we have only categorized constraints in multiple view geometry based on the rank techniques. That is merely the first step to the goal of reconstructing high-dimensional objects from multiple (perspective) images. There is already a vast body of literature in computer vision showing (for the case of projection from $\mathbb{R}^3$ to $\mathbb{R}^2$) how to exploit some of these constraints for reconstruction purposes. Since most of the existing techniques deal with two or three images at a time, we believe that the rank conditions, which unify global constraints among multiple images, will eventually give rise to more global techniques for reconstruction. In the case of three-dimensional space, this has been shown to be true. However, at this point, very little is known about and very few algorithms are available for reconstruction from multiple views in a general n-dimensional space.

This chapter has essentially demonstrated the possibility of generalizing classic multiple view geometry from $\mathbb{R}^3$ to $\mathbb{R}^n$. However, there is yet another direction that a generalization could take: changing the structure of the underlying ambient space, say from Euclidean to non-Euclidean. That is to develop a version of multiple view geometry for spaces with non-zero curvature. Such a generalization typically requires a generalization of the concept of "perspective projection" (see [MS98]). In such a context, many intriguing mathematical questions may arise: how to study geometric relationships between an "object" in a general space (say a Riemannian manifold) and its multiple low-dimensional "images"; what are the corresponding conditions that would allow a full reconstruction of the object from its multiple images; and in general, how to develop efficient algorithms for such reconstruction purposes. We would like to leave these questions for future investigation.

Part IV

Reconstruction algorithms

Chapter 12

Batch reconstruction from multiple views

Chapter 13

Recursive estimation from image sequences

In this chapter we consider the problem of estimating structure and motion under the constraint of causal processing. In applications such as driving, docking, tracking, manipulation etc., a control task must be performed based on visual information up to the current time instant. Processing data in a batch would result in a delay (to first collect and then process the data) that makes the control task significantly harder. The problem of causally estimating structure and motion can be shown to not have a finite-dimensional optimal solution. Therefore, one has to resort to approximations. We do so in this chapter, where we describe a local nonlinear filter to recursively estimate structure and motion that has been implemented on commercial hardware for real-time operation.

13.1 Motion and shape estimation as a filtering problem

Consider an N-tuple of points in the three-dimensional Euclidean space $\mathbb{E}^3$ and their coordinates represented as a matrix

$$\mathbf{X} \doteq \left[\begin{array}{cccc} \mathbf{X}^1 & \mathbf{X}^2 & \dots & \mathbf{X}^N \end{array} \right] \in \mathbb{R}^{3 \times N} \tag{13.1}$$

and let them move under the action of a rigid motion between two adjacent time instants $g(t+1) = \exp(\widehat{\xi}(t))g(t); \quad \xi(t) \in se(3)$. We assume that we can measure the (noisy) projection

$$\mathbf{y}^i(t) = \pi \left(g(t)\mathbf{X}^i \right) + \mathbf{n}^i(t) \in \mathbb{R}^2 \quad \forall\, i = 1 \dots N \tag{13.2}$$

where we know the correspondence $\mathbf{y}^i \leftrightarrow \mathbf{X}^i$. Finally, by organizing the time-evolution of the configuration of points and their motion, we end up with a discrete-time, non-linear dynamical system:

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{X}(t) & \mathbf{X}(0) = \mathbf{X}_0 \in \mathbb{R}^{3\times N} \\ g(t+1) = \exp(\widehat{\xi}(t))g(t) & g(0) = g_0 \in SE(3) \\ \xi(t+1) = \xi(t) + \alpha(t) & \xi(0) = \xi_0 \in se(3) \\ \mathbf{y}^i(t) = \pi\left(g(t)\mathbf{X}^i(t)\right) + \mathbf{n}^i(t) & \mathbf{n}^i(t) \sim \mathcal{N}(0, \Sigma_n) \end{cases} \tag{13.3}$$

where $\sim \mathcal{N}(M, S)$ indicates a normal distribution with mean M and covariance S. In the above model, α is the relative acceleration between the viewer and the scene. If some prior modeling information is available (for instance when the camera is mounted on a vehicle or on a robot arm), this is the place to use it. Otherwise, a statistical model can be employed. In particular, one way to formalize the fact that no information is available is by modeling α as a Brownian motion process. This is what we do in this chapter[1]. In principle one would like - at least for this simplified formalization of SFM - to find the "optimal solution", that is the description of the state of the above system $\{\mathbf{X}, g, \xi\}$ given a sequence output measurements (correspondences) $\mathbf{y}^i(t)$ over an interval of time. Since the measurements are noisy and described using a stochasic model, the description of the state consists in its probability density conditioned on the measurements. We call an algorithm that delivers the conditional density of the state at time t *causally* (i.e. based upon measurements up to time t) the *optimal filter*.

13.2 Observability

To what extent can the 3-D shape and motion of a scene be reconstructed *causally* from measurements of the motion of its projection onto the sensor? This is the subject of this section, which we start by establishing some notation that will be used throughout the rest of the chapter.

Let a rigid motion $g \in SE(3)$ be represented by a translation vector $T \in \mathbb{R}^3$ and a rotation matrix $R \in SO(3)$, and let $\alpha \neq 0$ be a scalar. The *similarity group*, which we indicate by $g_\alpha \in SE(3) \times \mathbb{R}_+$ is the composition of a rigid motion and a scaling, which acts on points in $\mathbb{R}^3$ as follows:

$$g_\alpha(\mathbf{X}) = \alpha R\mathbf{X} + \alpha T. \tag{13.4}$$

We also define an action of g_α on $SE(3)$ as

$$g_\alpha(g') = \{\alpha RT' + \alpha T, RR'\} \tag{13.5}$$

[1]We wish to emphasize that this choice is not crucial towards the results of this chapter. Any other model would do, as long as the overall system is observable.

and an action on $se(3)$, represented by v and ω, as

$$g_\alpha(\xi) = \{\alpha v, \widehat{\omega}\}. \tag{13.6}$$

The similarity group, acting on an N-tuple of points in $\mathbb{R}^3$, generates an equivalence class:

$$[\mathbf{X}] = \{\mathbf{Y} \in \mathbb{R}^{3\times N} \mid \exists\, g_\alpha \mid \mathbf{Y} = g_\alpha \mathbf{X}\} \tag{13.7}$$

two configurations of points $\mathbf{X}$ and $\mathbf{Y} \in \mathbb{R}^{3\times N}$ are equivalent if there exists a similarity transformation g_α that brings one onto the other: $\mathbf{Y} = g_\alpha \mathbf{X}$. Such equivalence class in (13.7) is called a *fiber*, and the collection of all fibers is called a *fiber bundle*. Therefore, the similarity group organizes the space of N-tuples into a fiber bundle, which we call the *state-space bundle*: given a point $\mathbf{X}$ in $\mathbb{R}^{3\times N}$, it belongs to one and only one fiber. From any given point it is possible to move either along the fiber (via the similarity group) or across fibers. One element of each fiber is sufficient to represent it, since all other elements are just transformed versions of it via the similarity group. In order to obtain a representation of the whole bundle, however, we need a consistent way of choosing a representative for each fiber. This is called a *base* of the fiber bundle. We will show how to construct a base of the homogeneous space of coordinates of points under the similarity transformation in section 13.3. For more details on the geometric structure of fiber bundles the reader is referred to [GH86]; for the purpose of this chapter an intuitive notion of fiber bundle as the equivalence class of coordinates of points in $\mathbb{R}^3$ under the similarity transformation suffices.

Preliminaries

Consider a discrete-time nonlinear dynamical system of the general form

$$\begin{cases} x(t+1) = f(x(t)) & x(t_0) = x_0 \\ y(t) = h(x(t)) \end{cases} \tag{13.8}$$

and let $y(t; t_0, x_0)$ indicate the output of the system at time t, starting from the initial condition x_0 at time t_0. In this section we want to characterize the states x that can be reconstructed from the measurements y. Such a characterization depends upon the structure of the system f, h and not on the measurement noise, which is therefore assumed to be absent for the purpose of the analysis in this section.

Definition 13.1. *Consider a system in the form (13.8) and a point in the state-space x_0. We say that x_0 is* indistinguishable *from x_0' if $y(t; t_0, x_0') = y(t; t_0, x_0) \quad \forall\, t, t_0$. We indicate with $\mathcal{I}(x_0)$ the set of initial conditions that are indistinguishable from x_0.*

Definition 13.2. *We say that the system (13.8) is* observable up to a (group) transformation ψ *if*

$$\mathcal{I}(x_0) = [x_0] \doteq \{x_0' \mid \exists\ \psi \mid x_0' = \psi(x_0)\}. \tag{13.9}$$

As articulated in the previous section, the state-space can be represented as a *fiber bundle*: each fiber is an equivalence class of initial conditions that are indistinguishable. Clearly, from measurements of the output $y(t)$ over any period of time, it is possible to recover at most the equivalence class (fiber) where the initial condition belongs, that is $\mathcal{I}(x_0)$, but not x_0 itself. The only case when this is possible is if the system is observable up to the identity transformation. In this case we have $\mathcal{I}(x_0) = \{x_0\}$ and we say that the system is *observable*.

For a generic linear time-varying system of the form

$$\begin{cases} x(t+1) = F(t)x(t) & x(t_0) = x_0 \\ y(t) = H(t)x(t) \end{cases} \tag{13.10}$$

the *k-observability Grammian* is defined as

$$M_k(t) \doteq \sum_{i=t}^{t+k} \Phi_i^T(t) H^T(t) H(t) \Phi_i(t) \quad \forall\ i\ >\ t \tag{13.11}$$

where $\Phi_t(t) = I$ and $\Phi_i(t) \doteq F(i-1)\dots F(t)$. The following definition will come handy in section 13.4:

Definition 13.3. *We say that the system (13.10) is* uniformly observable *if there exist real numbers $m_1 > 0,\ m_2 > 0$ and an integer $k > 0$ such that $m_1 I \leq M_k(t) \leq m_2 I\ \forall\ t$.*

The following proposition revisits the well-known fact that, under constant velocity, structure and motion are observable up to a (global) similarity transformation.

Proposition 13.1. *The model (13.3) where the points $\mathbf{X}$ are in general position is observable up to a similarity transformation of $\mathbf{X}$ provided that $v_0 \neq 0$. In particular, the set of initial conditions that are indistinguishable from $\{\mathbf{X}_0, g_0, \xi_0\}$, where $g_0 = \{T_0, R_0\}$ and $e^{\hat{\xi}_0} = \{v_0, U_0\}$, is given by $\{\tilde{R}\mathbf{X}_0\alpha + \tilde{T}\alpha, \tilde{g}_0, \tilde{\xi}_0\}$, where $\tilde{g}_0 = \{T_0\alpha - R_0\tilde{R}^T\tilde{T}\alpha, R_0\tilde{R}^T\}$ and $e^{\hat{\tilde{\xi}}_0} = \{v_0\alpha, U_0\}$.*

Proof. Consider two initial conditions $\{\mathbf{X}_1, g_1, \xi_1\}$ and $\{\mathbf{X}_2, g_2, \xi_2\}$. For them to be indistinguishable we must have $\mathbf{y}(t) = \pi(g_1(t)\mathbf{X}_1(t)) = \pi(g_2(t)\mathbf{X}_2(t)) \quad \forall\ t \geq 0$. In particular, at time $t = 0$ this is equivalent to the existence of a diagonal matrix of scalings, $A(1)$, such that $g_1(0)\mathbf{X}_1(0) = (g_2(0)\mathbf{X}_2) \cdot A(1)$, where the operator $\cdot$ performs the scaling according to $(g\mathbf{X}) \cdot A \doteq RXA + TA$. Under the assumption of constant velocity, we have that $g(t) = e^{t\hat{\xi}} g(0)$, and therefore the group action

g only appears at the initial time. Consequently, we drop the time index and write simply g_1 and g_2 as points in $SE(3)$. At the generic time instant t, the indistinguishability condition can therefore be written as $e^{t\widehat{\xi}_1} g_1 \mathbf{X}_1 = (e^{t\widehat{\xi}_2} g_2 \mathbf{X}_2) \cdot A(t+1)$. Therefore, given $\mathbf{X}_2, g_2, \xi_2$, in order to find the initial conditions that are indistinguishable from it, we need to find $\mathbf{X}_1, g_1, \xi_1$ and $A(k), k \geq 1$ such that, after some substitutions, we have

$$\begin{cases} g_1 \mathbf{X}_1 = (g_2 \mathbf{X}_2) \cdot A(1) \\ e^{\widehat{\xi}_1} e^{(k-1)\widehat{\xi}_2} g_2 \mathbf{X}_2 = \left(e^{\widehat{\xi}_2} e^{(k-1)\widehat{\xi}_2} g_2 \mathbf{X}_2 \right) \cdot A(k+1) \quad k \geq 1. \end{cases} \tag{13.12}$$

Making the representation of $SE(3)$ explicit, we write the above conditions as

$$\begin{cases} R_1 \mathbf{X}_1 + \bar{T}_1 = (R_2 \mathbf{X}_2 + \bar{T}_2) A(1) \\ U_1 \tilde{\mathbf{X}}_k A(k) + \bar{v}_1 = U_2 \tilde{\mathbf{X}}_k A(k+1) + \bar{v}_2 A(k+1) \end{cases} \tag{13.13}$$

where we have defined $\tilde{\mathbf{X}}_k \doteq e^{(k-1)\widehat{\xi}_2} g_2 \mathbf{X}_2$ which, by the assumption of general position, is of full rank 3, and $\bar{v}$ denotes the rank-one matrix $\bar{v} \doteq v I_N^T$, where I_N is the N-dimensional vector of ones. We can rewrite the second of the equations above in a more enlightening way as follows:

$$\tilde{\mathbf{X}}_k A(k) A^{-1}(k+1) - U_1^T U_2 \tilde{\mathbf{X}}_k = U_1^T (\bar{v}_2 A(k+1) - \bar{v}_1) A^{-1}(k+1). \tag{13.14}$$

The $3 \times N$ matrix on the right hand-side has rank at most 2, while the left hand-side has rank 3, following the general-position conditions, unless $A(k)A^{-1}(k+1) = I$ and $U_1^T U_2 = I$, in which case it is identically zero. Therefore, both terms in the above equations must be identically zero. From $U_1^T U_2 = I$ we conclude that $U_1 = U_2$, while from $A(k)A^{-1}(k+1) = I$ we conclude that $A(k)$ is constant. However, the right hand-side imposes that $\bar{v}_2 A = \bar{v}_1$, or in vector form $v_2 a^T = v_1 I_N^T$ where $A = diag\{a\}$, which implies that $A = \alpha I$, i.e. a multiple of the identity. Now, going back to the first equation in (13.13), we conclude that $R_1 = R_2 \tilde{R}^T$, for any $\tilde{R} \in SO(3)$, $X_1 = (\tilde{R} X_0 + \tilde{T})\alpha$ for any $\tilde{T} \in \mathbb{R}^3$, and finally $T_1 = (T_2 - R_1 \tilde{R}^T \tilde{T})\alpha$, which concludes the proof. □

Remark 13.1. *The relevance of the above proposition for the practical estimation of structure from motion (where velocity is not necessarily constant) is that one can (in principle) solve the problem using the above model only when velocity varies slowly compared to the sampling frequency. If, however, some information on acceleration becomes available (as for instance if the camera is mounted on a support with some inertia), then the restriction on velocity can be lifted. This framework, however, will not hold if the data $y(t)$ are snapshots of a scene taken from sparse viewpoints, rather than a sequence of images taken ad adjacent time instants while the camera is moving in a continuous trajectory.*

The following proposition states that it is possible to make the model observable by fixing the direction of three points and one depth. It is closely

related to what some authors call invariance to "gauge transformations" [McL99], except that in our case we have to consider the *causality constraint*. When we interpret the state-space as a fiber bundle under the action of the similarity group, fixing the direction of three points and one depth identifies a base of the bundle, that is a point in the similarity group. Without loss of generality (i.e. modulo a re-ordering of the states) we will assume the indices of such three points to be 1, 2 and 3. We consider a point $\mathbf{X}$ as parameterized by its direction $\mathbf{y}$ and depth ρ, so that $\mathbf{X} = \mathbf{y}\rho$.

Proposition 13.2. *Given the direction of three non-collinear points, $\mathbf{y}^1, \mathbf{y}^2, \mathbf{y}^3$ and the scale of one point, $\rho^1 > 0$, and given vectors ϕ^i, $i = 1 \dots N$, the set of motions $g = \{T, R\} \in SE(3)$ and scales $\alpha \in \mathbb{R}$ such that*

$$\alpha R \mathbf{y}^i \rho^i + \alpha T = \phi^i \qquad \forall\ i = 1 \dots N \geq 3 \tag{13.15}$$

has measure zero.

Proof. Suppose that the statement holds for $N = 3$, then it holds for any $N > 3$, as any additional equation of the form $\phi^i = \alpha R \mathbf{y}^i \rho^i + \alpha T$ is linear in the variable $\mathbf{X}^i \doteq \mathbf{y}^i \rho^i$, and therefore can be solved uniquely. Since ${\mathbf{X}_3}^i = \rho^i$, the latter is uniquely determined, and so is $\mathbf{y}^i = \frac{\mathbf{X}^i}{\rho^i}$. Therefore, we only need to prove the statement for $N = 3$:

$$\begin{cases} \phi^1 = \alpha R \mathbf{y}^1 \rho^1 + \alpha T \\ \phi^2 = \alpha R \mathbf{y}^2 \rho^2 + \alpha T \\ \phi^3 = \alpha R \mathbf{y}^3 \rho^3 + \alpha T. \end{cases} \tag{13.16}$$

Solve the first equation for αT,

$$\alpha T = \phi^1 - \alpha R \mathbf{y}^1 \rho^1 \neq 0 \tag{13.17}$$

and substitute into the second and third equation to get

$$\begin{cases} \phi^2 - \phi^1 = \alpha R (\mathbf{y}^2 \rho^2 - \mathbf{y}^1 \rho^1) \\ \phi^3 - \phi^1 = \alpha R (\mathbf{y}^3 \rho^3 - \mathbf{y}^1 \rho^1). \end{cases} \tag{13.18}$$

The scale $\alpha > 0$ can be solved for as a function of the unknown scales ρ^2 and ρ^3

$$\alpha = \frac{\|\phi^2 - \phi^1\|}{\|\mathbf{y}^2 \rho^2 - \mathbf{y}^1 \rho^1\|} = \frac{\|\phi^3 - \phi^1\|}{\|\mathbf{y}^3 \rho^3 - \mathbf{y}^1 \rho^1\|} \tag{13.19}$$

(note that these expressions are always legitimate as a consequence of the non-collinearity assumption). After substituting α in equations (13.18), we get

$$\begin{cases} \frac{\phi^2 - \phi^1}{\|\phi^2 - \phi^1\|} = R \frac{\mathbf{y}^2 \rho^2 - \mathbf{y}^1 \rho^1}{\|\mathbf{y}^2 \rho^2 - \mathbf{y}^1 \rho^1\|} \\ \frac{\phi^3 - \phi^1}{\|\phi^3 - \phi^1\|} = R \frac{\mathbf{y}^3 \rho^3 - \mathbf{y}^1 \rho^1}{\|\mathbf{y}^3 \rho^3 - \mathbf{y}^1 \rho^1\|}. \end{cases} \tag{13.20}$$

In the above equations, the only unknowns are R, ρ^2 and ρ^3. Note that, while on the left hand-side there are two fixed unit-norm vectors, on the

right hand-side there are unit-norm vectors parameterized by ρ^2 and ρ^3 respectively. In particular, the right hand-side of the first equation in (13.20) is a vector on the unit circle of the plane spanned by $\mathbf{y}^1$ and $\mathbf{y}^2$, while the right hand-side of the second equation is a vector on the unit circle of the plane π_2 spanned by $\mathbf{y}^1$ and $\mathbf{y}^3$. By the assumption of non-collinearity, these two planes do not coincide. We write the above equation in a more compact form as

$$\begin{cases} {\phi_\nu}^1 = R\mu_{\rho^2} \\ {\phi_\nu}^2 = R\mu_{\rho^3}. \end{cases} \tag{13.21}$$

Now R must preserve the angle between ${\phi_\nu}^1$ and ${\phi_\nu}^2$, which we indicate as $\widehat{{\phi_\nu}^1{\phi_\nu}^2}$, and therefore μ_{ρ^2} and μ_{ρ^2} must be chosen accordingly. If $\widehat{{\phi_\nu}^1{\phi_\nu}^2} > \pi - \widehat{\pi_1\pi_2}$, no such choice is possible. Otherwise, there exists a one-dimensional interval set of ρ^1, ρ^2 for which one can find a rotation R that preserves the angle. However, R must also preserve the cross product, so that we have

$${\phi_\nu}^1 \times {\phi_\nu}^2 = (R\mu_{\rho^2}) \times R\mu_{\rho^3} = R\mu_{\rho^2} \times (R^T R\mu_{\rho^3}) = R(\mu_{\rho^2} \times \mu_{\rho^3}) \tag{13.22}$$

(note that the norm of the two cross products is the same as a consequence of the conservation of the inner product), and therefore ρ^2 and ρ^3 are determined uniquely, and so is R, which concludes the proof. □

13.3 Realization

In order to design a finite-dimensional approximation to the optimal filter, we need an observable realization of the original model. How to obtain it is the subject of this section.

Local coordinates

Our first step consists in characterizing the local-coordinate representation of the model (13.3). To this end, we represent $SO(3)$ locally in canonical exponential coordinates: let Ω be a three-dimensional real vector ($\Omega \in \mathbb{R}^3$); $\frac{\Omega}{\|\Omega\|}$ specifies the direction of rotation and $\|\Omega\|$ specifies the angle of rotation in radians. Then a rotation matrix can be represented by its exponential coordinates $\widehat{\Omega} \in so(3)$ such that $R \doteq \exp(\widehat{\Omega}) \in SO(3)$ as described in chapter 2. The three-dimensional coordinate $\mathbf{X}^i$ is represented by its projection onto the image plane $\mathbf{y}^i$ and its depth ρ^i, so that

$$\mathbf{y}^i \doteq \pi(\mathbf{X}^i) \doteq \begin{bmatrix} \frac{X_1^i}{X_3^i} \\ \frac{X_2^i}{X_3^i} \end{bmatrix} \qquad \rho^i = X_3^i. \tag{13.23}$$

Such a representation has the advantage of decomposing the uncertainty in the measured directions $\mathbf{y}$ (low) from the uncertainty in depth ρ (high). The model (13.3) in local coordinates is therefore

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 1 \dots N & \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 1 \dots N & \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + v(t) & & T(0) = T_0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t)) \exp(\widehat{\Omega}(t))) & & \Omega(0) = \Omega_0 \\ v(t+1) = v(t) + \alpha_v(t) & v(0) = v_0 & \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 & \\ \mathbf{y}^i(t) = \pi \left(\exp(\widehat{\Omega}(t)) \mathbf{y}_0^i(t) \rho^i(t) + T(t) \right) + \mathbf{n}^i(t) & & i = 1 \dots N. \end{cases} \tag{13.24}$$

The notation $Log_{SO(3)}(R)$ stands for Ω such that $R = e^{\widehat{\Omega}}$ and is computed by inverting Rodrigues' formula as described in chapter 2.

Minimal realization

In linear time-invariant systems one can decompose the state-space into an observable subspace and its (unobservable) complement. In the case of our system, which is nonlinear and observable up to a group transformation, we can exploit the bundle structure of the state-space to realize a similar decomposition: the base of the fiber bundle is observable, while individual fibers are not. Therefore, in order to restrict our attention to the observable component of the system, we only need to choose a base of the fiber bundle, that is a particular (representative) point on each fiber.

Proposition 13.2 suggests a way to render the model (13.24) observable by eliminating the states that fix the unobservable subspace.

Corollary 13.1. *The model*

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 4 \dots N & \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 2 \dots N & \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + v(t) & & T(0) = T_0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t)) \exp(\widehat{\Omega}(t))) & & \Omega(0) = \Omega_0 \\ v(t+1) = v(t) + \alpha_v(t) & v(0) = v_0 & \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 & \\ \mathbf{y}^i(t) = \pi \left(\exp(\widehat{\Omega}(t)) \mathbf{y}_0^i(t) \rho^i(t) + T(t) \right) + n^i(t) & & i = 1 \dots N. \end{cases} \tag{13.25}$$

which is obtained by eliminating $\mathbf{y}_0^1, \mathbf{y}_0^2, \mathbf{y}_0^3$ *and* ρ^1 *from the state of the model (13.24), is observable.*

Linearization and uniform observability of the minimal realization

Let $x \doteq [\mathbf{y}_0^{4\,T}, \ldots, \mathbf{y}_0^{N\,T}, \rho^2, \ldots, \rho^N, T^T, \Omega^T, v^T, \omega^T]^T$ be the state vector of a minimal realization, $\mathbf{e}_i$ the i-th canonical vector in $\mathbb{R}^3$ and define $Y^i(t) \doteq e^{\widehat{\Omega}(t)}\mathbf{y}_0^i(t)\rho^i(t) + T(t)$, $Z^i(t) \doteq \mathbf{e}_3^T Y^i(t)$. Let $F(t) \doteq \frac{\partial f(x)}{\partial x}$, $H(t) \doteq \frac{\partial \pi(x)}{\partial x}$ denote the linearization of the state and measurement equation in (**??**) respectively. Explicit expressions for the linearization are given in section 13.5. Here we just wish to remark that, in order for the linearization to be well-defined, we need to ensure that the depth of each point $Z^i(t)$ is strictly positive as well as bounded. This is a reasonable assumption since it corresponds to each visible point being in front of the camera at a finite distance. Therefore, we restrict our attention to motions that guarantee that this condition is verified.

Definition 13.4. *We say that a motion is admissible if $v(t)$ is not identically zero, i.e. there is an open set (a, b) such that $v(t) \neq 0$, $t \in (a, b)$ and the corresponding trajectory of the system (**??**) is such that $c \leq Z^i(t) \leq C$, $\forall\ i = 1, \ldots, N$, $\forall\ t > 0$ for some constants $c > 0$, $C < \infty$.*

Proposition 13.3. *Let $F(t) \doteq \frac{\partial f(x)}{\partial x}$, $H(t) \doteq \frac{\partial \pi(x)}{\partial x}$ denote the linearization of the state and measurement equation in (**??**) respectively; let $N > 5$ and assume the motion is admissible; then the linearized system is uniformly observable.*

Proof. Let $k = 2$; that there exists an $m_2 < \infty$ such that the Grammian $M_2(t) \leq m_2 I$ follows from the fact that $F(t)$ and $H(t)$ are bounded for all t, as can be easily verified provided that motion is admissible. We now need to verify that $M_2(t)$ is strictly positive definite for all t. To this end, it is sufficient to show that the matrix

$$U_2(t) \doteq \begin{bmatrix} H(t) \\ H(t+1)F(t) \end{bmatrix} \tag{13.26}$$

has full column rank equal to $3N + 5$ for all values of t, which can be easily verified whenever $N \geq 5$. □

13.4 Stability

Following the derivation in previous sections, the problem of estimating the motion, velocity and point-wise structure of the scene can be converted into the problem of estimating the state of the model (**??**). We propose to solve the task using a nonlinear filter, properly designed to account for the observability properties of the model. The implementation, which we report in detail in section 13.5, results in a sub-optimal filter. However, it is important to guarantee that the estimation error, while different from

zero, remains bounded. To streamline the notation, we represent the model (**??**) as

$$\begin{cases} x(t+1) = f(x(t)) + w(t) \quad w(t) \sim \mathcal{N}(0, \Sigma_{w_0}(t)), \\ y(t) = h(x(t)) + n(t) \quad n(t) \sim \mathcal{N}(0, \Sigma_{n_0}(t)). \end{cases} \tag{13.27}$$

The filter is described by a difference equation for the state $\hat{x}(t)$. We call the estimation error

$$\tilde{x}(t) \doteq x(t) - \hat{x}(t) \tag{13.28}$$

and its variance at time t $P(t)$. The initial conditions for the estimator are

$$\begin{cases} \hat{x}(0) = x_0 \\ P(0) = P_0 > 0 \end{cases} \tag{13.29}$$

and its evolution is governed by

$$\begin{cases} \hat{x}(t+1) = f(\hat{x}(t)) + K(t)[y(t+1) - h(\hat{x}(t))] \\ P(t+1) = \mathcal{R}\left(P(t), F(t), H(t), \Sigma_n, \Sigma_w\right) \end{cases} \tag{13.30}$$

where $\mathcal{R}$ denotes the usual Riccati equation which uses the linearization of the model $\{F, H\}$ computed at the current estimate of the state, as described in [**?**].

Note that we call Σ_{n_0}, Σ_{w_0} the variance of the measurement and model noises, and Σ_n, Σ_w the tuning parameters that appear in the Riccati equation. The latter are free for the designer to choose, as described in section 13.5.

Stability analysis

The aim of this section is to prove that the estimation error generated by the filter just described is bounded. In order to do so, we need a few definitions.

Definition 13.5. *A stochastic process $\tilde{x}(t)$ is said to be* exponentially bounded in mean-square *(or MS-bounded) if there are real numbers η, $\nu > 0$ and $0 < \theta < 1$ such that $E\|\tilde{x}(t)\|^2 \leq \eta\|\tilde{x}(0)\|^2\theta^t + \nu$ for all $t \geq 0$. $\tilde{x}(t)$ is said do be* bounded with probability one *(or bounded WP1) if $P[\sup_{t\geq 0} \|\tilde{x}(t)\| < \infty] = 1$.*

Definition 13.6. *The filter (13.27) is said to be* stable *if there exist positive real numbers ϵ and δ such that*

$$\|\tilde{x}(0)\| \leq \epsilon,\ \Sigma_n(t) \leq \delta I,\ \Sigma_w(t) \leq \delta I \implies \tilde{x}(t) \text{ is bounded.} \tag{13.31}$$

Depending on whether $x(t)$ is bounded in mean square or with probability one, we say that the filter is "MS-stable" *or* "stable WP1"*.*

We are now ready to state the core proposition of this section

Proposition 13.4. *Let the hypothesis of proposition 13.3 be fulfilled. Then the filter based on the model (??) is MS-stable and stable WP1.*

In order to prove the proposition, we need a result that follows directly from corollary 5.2 of [?]:

Lemma 13.1. *In the filter based on the model (??), let motion be admissible and $P_0 > 0$. Then there exist positive real numbers p_1 and p_2 such that*

$$p_1 I \leq P(t) \leq p_2 I \quad \forall\ t \geq 0. \tag{13.32}$$

Proof. The proof follows from corollary 5.2 of [?], using proposition 13.3 on the uniform observability of the linearization of (??). □

Now the proof for the proposition:

Proof. The proposition follows from theorem 3.1 in [?], making use in the assumptions of the boundedness of $F(t)$, $H(t)$, lemma 13.1 and the differentiability of f and g when $0 < \rho^i < \infty\ \forall\ i$. □

Non-minimal models

Most recursive schemes for causally reconstructing structure and motion available in the literature represent structure using only one state per point (either its depth in an inertial frame, or its inverse, or other variations on the theme). This corresponds to reducing the state of the model (??), with the states $\mathbf{y}_0^i$ substituted for the measurements $\mathbf{y}^i(0)$, which causes the model noise $n(t)$ to be non-zero-mean[2]. When the zero-mean assumption implicit in the use of the Kalman filter is violated, the filter settles at a severely biased estimate. In this case we say that the model is *sub-minimal*.

On the other hand, when the model is non-minimal – such is the case when we do not force it to evolve on a base of the state-space bundle – the filter is free to wonder on the non-observable space (i.e. along the fibers of the state-space bundle), therefore causing the explosion of the variance of the estimation error along the unobservable components.

13.5 Implementation

In the previous section we have seen that the model (??) is observable. Notice that in that model the index for $\mathbf{y}_0^i$ starts at 4, while the index for ρ^i starts at 2. This corresponds to choosing the first three points as reference for the similarity group and is necessary (and sufficient) for guaranteeing

[2]In fact, if we call $n^i(0)$ the error in measuring the position of the point i at time 0, we have that $E[n^i(t)] = n^i(0)\ \forall\ t$.

that the representation is minimal. In the model (**??**), we are free to choose the initial conditions Ω_0, T_0, which we will therefore let be $\Omega_0 = T_0 = 0$, thereby choosing the camera reference at the initial time instant as the world reference.

Partial autocalibration

As we have anticipated, the models proposed can be extended to account for changes in calibration. For instance, if we consider an imaging model with focal length f[3]

$$\pi_f(\mathbf{X}) = \frac{f \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}}{X_3} \tag{13.33}$$

where the focal length can change in time, but no prior knowledge on how it does so is available, one can model its evolution as a random walk

$$f(t+1) = f(t) + \alpha_f(t) \qquad \alpha_f(t) \sim \mathcal{N}(0, \sigma_f^2) \tag{13.34}$$

and insert it into the state of the model (13.3). As long as the overall system is observable, the conclusions reached in the previous section will hold. The following claim shows that this is the case for the model (13.34) above. Another imaging model proposed in the literature is the following [**?**]:

$$\pi_\beta(\mathbf{X}) = \frac{\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}}{1 + \beta X_3} \tag{13.35}$$

for which similar conclusions can be drawn.

Proposition 13.5. *Let $g = \{T, R\}$ and $\xi = \{v, \omega\}$. The model*

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{X}(t) & \mathbf{X}(0) = \mathbf{X}_0 \\ g(t+1) = e^{\widehat{\xi}} g(t) & g(0) = g_0 \\ \xi(t+1) = \xi(t) & \xi(0) = \xi_0 \\ f(t+1) = f(t) & f(0) = f_0 \\ \mathbf{y}(t) = \pi_f(g(t)\mathbf{X}(t)) \end{cases} \tag{13.36}$$

is observable up to the action of the group represented by $\tilde{T}, \tilde{R}, \alpha$ acting on the initial conditions.

Proof. Consider the diagonal matrix $F(t) = \text{diag}\{f(t),\ f(t),\ 1\}$ and the matrix of scalings $A(t)$ as in the proof in section 13.2. Consider then two

[3]This f is not to be confused with the generic state equation of the filter in section 13.6.

initial conditions $\{\mathbf{X}_1, g_1, \xi_1, f_1\}$ and $\{\mathbf{X}_2, g_2, \xi_2, f_2\}$. For them to be indistinguishable there must exist matrices of scalings $A(k)$ and of focus $F(k)$ such that

$$\begin{cases} g_1\mathbf{X}_1 = F(1)(g_2\mathbf{X}_2)\cdot A(1) \\ e^{\widehat{\xi}_1}e^{(k-1)\widehat{\xi}_1}g_1\mathbf{X}_1 = F(k+1)\left(e^{\widehat{\xi}_2}e^{(k-1)\widehat{\xi}_2}g_2\mathbf{X}_2\right)\cdot A(k+1) \quad k \geq 1. \end{cases} \tag{13.37}$$

Making the representation explicit we obtain

$$\begin{cases} R_1\mathbf{X}_1 + \bar{T}_1 = F(1)(R_2\mathbf{X}_2 + \bar{T}_2)A(1) \\ U_1F(k)\tilde{\mathbf{X}}_kA(k) + \bar{v}_1 = F(k+1)(U_2\tilde{\mathbf{X}}_k + \bar{v}_2)A(k+1) \end{cases} \tag{13.38}$$

which can be re-written as

$$\begin{aligned} &\tilde{\mathbf{X}}_kA(k)A^{-1}(k+1) - F^{-1}(k)U_1^TF(k+1)U_2\tilde{\mathbf{X}}_k \\ &= F(k)^{-1}U_1^T(F(k+1)\bar{v}_2A(k+1) - \bar{v}_1)A^{-1}(k+1). \end{aligned}$$

The two sides of the equation have equal rank only if it is equal to zero, which draws us to conclude that $A(k)A^{-1}(k+1) = I$, and hence A is constant. From $F^{-1}(k)U_1^TF(k+1)U_2 = I$ we get that $F(k+1)U_2 = U_1F(k)$ and, since $U_1, U_2 \in SO(3)$, we have that taking the norm of both sides $2f^2(k+1) + 1 = 2f^2(k) + 1$, where f must be positive, and therefore constant: $FU_2 = U_1F$. From the right hand side we have that $F\bar{v}_2A = \bar{v}_1$, from which we conclude that $A = \alpha I$, so that in vector form we have $v_1 = \alpha Fv_2$. Therefore, from the second equation we have that, for any f and any α, we can have

$$v_1 = \alpha Fv_2 \tag{13.39}$$

$$U_1 = FU_2F^{-1}. \tag{13.40}$$

However, from the first equation we have that $R_1\mathbf{X}_1 + T_1 = \alpha FR_2\mathbf{X}_2 + \alpha FT_2$, whence - from the general position conditions - we conclude that $R_1 = \alpha FR_2$ and therefore $F = I$. From that we have that $T_1 = \alpha FT_2 = \alpha T_2$ which concludes the proof. □

Remark 13.2. *The previous claim essentially implies that the realization remains minimal if we add into the model the focal parameter. Note that observability depends upon the structural properties of the model,* not *on the noise, which is therefore assumed to be zero for the purpose of the proof.*

Saturation

Instead of eliminating states to render the model observable, it is possible to design a nonlinear filter directly on the (unobservable) model (13.3) by *saturating* the filter along the unobservable component of the state space as we show in this section. In other words, it is possible to design the initial variance of the state of the estimator as well as its model error in such a

way that it will never move along the unobservable component of the state space.

As suggested in the previous section, one can saturate the states corresponding to $\mathbf{y}_0^1, \mathbf{y}_0^2, \mathbf{y}_0^3$ and ρ^1. We have to guarantee that the filter initialized at $\widehat{\mathbf{y}}_0, \widehat{\rho}_0, \widehat{g}_0, \widehat{\xi}_0$ evolves in such a way that $\widehat{\mathbf{y}}_0^1(t) = \widehat{\mathbf{y}}_0^1, \widehat{\mathbf{y}}_0^2(t) = \widehat{\mathbf{y}}_0^2, \widehat{\mathbf{y}}_0^3(t) = \widehat{\mathbf{y}}_0^3, \widehat{\rho}^1(t) = \widehat{\rho}_0^1$. It is simple, albeit tedious, to prove the following proposition.

Proposition 13.6. *Let $P_{\mathbf{y}^i}(0), P_{\rho^i}(0)$ denote the variance of the initial condition corresponding to the state $\mathbf{y}_0^i$ and ρ^i respectively, and $\Sigma_{\mathbf{y}^i}, \Sigma_{\rho^i}$ the variance of the model error corresponding to the same state, then*

$$\begin{cases} P_{\mathbf{y}^i}(0) = 0, \quad i = 1 \dots 3 \\ P_{\rho^1} = 0 \\ \Sigma_{y^i} = 0, \quad i = 1 \dots 3 \\ \Sigma_{\rho^1} = 0 \end{cases} \tag{13.41}$$

implies that $\widehat{\mathbf{y}}_0^i(t|t) = \widehat{\mathbf{y}}_0^i(0), \quad i = 1 \dots 3$, and $\widehat{\rho}^1(t|t) = \widehat{\rho}^1(0)$.

Pseudo-measurements

Yet another alternative to render the model observable is to add pseudo-measurement equations with zero error variance.

Proposition 13.7. *The model*

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 1 \dots N & \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 1 \dots N & \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + v(t) & & T(0) = 0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t)) \exp(\widehat{\Omega}(t))) & & \Omega(0) = 0 \\ v(t+1) = v(t) + \alpha_v(t) & v(0) = v_0 & \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & \omega(0) = \omega_0 & \\ \mathbf{y}^i(t) = \pi \left(\exp(\widehat{\Omega}(t)) \mathbf{y}^i(t) \rho^i(t) + T(t) \right) + n^i(t) & & i = 1 \dots N \\ \rho^1 = \psi_1 & & \\ \mathbf{y}^i(t) = \phi^i \quad i = 1 \dots 3, & & \end{cases} \tag{13.42}$$

where ψ_1 is an arbitrary (positive) constant and ϕ^i are three non-collinear points on the plane, is observable.

The implementation of an extended Kalman filter based upon the model (**??**) is straightforward. However, for the sake of completeness we report it in section 13.6. The only issue that needs to be dealt with is the disappearing and appearing of feature points, a common trait of sequences of images of natural scenes. Visible feature points may become occluded (and therefore their measurements become unavailable), or occluded points may

become visible (and therefore provide further measurements). New states must be properly initialized. Occlusion of point features do not cause major problems, unless the feature that disappears happens to be associated with the scale factor. This is unavoidable and results in a drift whose nature is explained below.

Occlusions

When a feature point, say $\mathbf{X}^i$, becomes occluded, the corresponding measurement $\mathbf{y}^i(t)$ becomes unavailable. It is possible to model this phenomenon by setting the corresponding variance to infinity or, in practice $\Sigma_{n^i} = MI_2$ for a suitably large scalar $M > 0$. By doing so, we guarantee that the corresponding states $\hat{\mathbf{y}}_0^i(t)$ and $\hat{\rho}^i(t)$ are not updated:

Proposition 13.8. *If* $\Sigma_{n^i} = \infty$*, then* $\hat{\mathbf{y}}_0^i(t+1) = \hat{\mathbf{y}}_0^i(t)$ *and* $\hat{\rho}^i(t+1) = \hat{\rho}^i(t)$.

An alternative, which is actually preferable in order to avoid useless computation and ill-conditioned inverses, is to eliminate the states $\hat{\mathbf{y}}_0^i$ and $\hat{\rho}^i$ altogether, thereby reducing the dimension of the state-space. This is simple due to the diagonal structure of the model (**??**): the states ρ^i, $\mathbf{y}_0^i$ are decoupled, and therefore it is sufficient to remove them, and delete the corresponding rows from the gain matrix $K(t)$ and the variance $\Sigma_w(t)$ for all t past the disappearance of the feature (see section 13.6).

When a new feature point appears, on the other hand, it is not possible to simply insert it into the state of the model, since the initial condition is unknown. Any initialization error will disturb the current estimate of the remaining states, since it is fed back into the update equation for the filter, and generates a spurious transient. We address this problem by running a separate filter in parallel for each point using the current estimates of motion from the main filter in order to reconstruct the initial condition. Such a "subfilter" is based upon the following model, where we assume that N_τ features appear at time τ:

$$\begin{cases} \mathbf{y}_\tau^i(t+1) = \mathbf{y}_\tau^i(t) + \eta_{y^i(t)} & \mathbf{y}_\tau^i(0) \sim \mathcal{N}(\mathbf{y}^i(\tau), \Sigma_{n^i}) \qquad t > \tau \\ \rho_\tau^i(t+1) = \rho_\tau^i(t) + \eta_{\rho^i(t)} & \rho^i(0) \sim \mathcal{N}(1, P_\rho(0)) \\ \mathbf{y}^i(t) = \pi \left(\exp(\widehat{\Omega}(t|t)) [\exp(\widehat{\Omega}(\tau|\tau))]^{-1} \left[\mathbf{y}_\tau^i(t) \rho_\tau^i(t) - T(\tau|\tau) \right] + T(t|t) \right) + n^i(t) \end{cases} \tag{13.43}$$

for $i = 1 \dots N_\tau$, where $\Omega(t|t)$ and $T(t|t)$ are the current best estimates of Ω and T, $\Omega(\tau|\tau)$ and $T(\tau|\tau)$ are the best estimates of Ω and T at $t = \tau$. In pracice, rather than initializing ρ to 1, one can compute a first approximation by triangulating on two adjacent views, and compute covariance of the initialization error from the covariance of the current estimates of motion. Several heuristics can be employed in order to decide when the estimate of the initial condition is good enough for it to be inserted into the main filter. The most natural criterion is when the variance of the estimation

error of ρ^i_τ in the subfilter is comparable with the variance of ρ^j_0 for $j \neq i$ in the main filter. The last step in order to insert the feature i into the main filter consists in bringing the coordinates of the new points back to the initial frame. This is done by

$$\mathbf{X}^i = \left[\exp(\widehat{\Omega}(\tau|\tau))\right]^{-1} \left[\mathbf{y}^i_\tau \rho^i_\tau - T(\tau|\tau)\right]. \tag{13.44}$$

Drift

The only case when losing a feature constitutes a problem is when it is used to fix the observable component of the state-space (in our notation, $i = 1, 2, 3$) [4]. The most obvious choice consists in associating the reference to any other visible point. This can be done by saturating the corresponding state and assigning as reference value the current best estimate. In particular, if feature i is lost at time τ, and we want to switch the reference index to feature j, we eliminate $\mathbf{y}^i_0,\ \rho^i$ from the state, and set the diagonal block of Σ_w and $P(\tau)$ with indices $3j-3$ to $3j$ to zero. Therefore, by proposition 13.6, we have that

$$\hat{\mathbf{y}}^j_0(\tau + t) = \hat{\mathbf{y}}^j_0(\tau) \quad \forall\, t > 0. \tag{13.45}$$

If $\hat{\mathbf{y}}^j_0(\tau)$ was equal to $\mathbf{y}^j_0$, switching the reference feature would have no effect on the other states, and the filter would evolve on the same observable component of the state-space determined by the reference feature i.

However, in general the difference $\tilde{\mathbf{y}}^j_0(\tau) \doteq \mathbf{y}^j_0(\tau) - \hat{\mathbf{y}}^j_0$ is a random variable with variance $\Sigma_\tau = P_{3j-3:3j-1,3j-3:3j-1}$. Therefore, switching the reference to feature j causes the observable component of the state-space to move by an amount proportional to $\tilde{\mathbf{y}}^j_0(\tau)$. When a number of switches have occurred, we can expect - on average - the state-space to move by an amount proportional to $\|\Sigma_\tau\|\#\text{switches}$. This is unavoidable. What we can do is at most try to keep the bias to a minimum by switching the reference to the state that has the lowest variance[5].

Of course, should the original reference feature i become available, one can immediately switch the reference to it, and therefore recover the original base and annihilate the bias.

---

[4] When the scale factor is not directly associated to one feature, but is associated to a function of a number of features (for instance the depth of the centroid, or the average inverse depth), then losing any of these features causes a drift.

[5] Just to give the reader an intuitive feeling of the numbers involved, we find that in practice the average lifetime of a feature is around 10-30 frames depending on illumination and reflectance properties of the scene and motion of the camera. The variance of the estimation error for $\mathbf{y}^i_0$ is in the order of 10^{-6} units of focal length, while the variance of ρ^i is in the order of 10^{-4} units for noise levels commonly encountered with commercial cameras.

13.6 Complete algorithm

The implementation of an approximate wide-sense nonlinear filter for the model (**??**) proceeds as follows:

Initialization

Choose the initial conditions

$$\begin{cases} \mathbf{y}_0^i = \mathbf{y}^i(0) \\ \rho_0^i = 1 \\ T_0 = 0 \\ \Omega_0 = 0 \\ v_0 = 0 \\ \omega_0 = 0 \end{cases} \qquad \forall\ i = 1 \dots N. \tag{13.46}$$

For the initial variance P_0, choose it to be block diagonal with blocks $\Sigma_{n^i}(0)$ corresponding to $\mathbf{y}_0^i$, a large positive number M (typically 100-1000 units of focal length) corresponding to ρ^i, zeros corresponding to T_0 and Ω_0 (fixing the inertial frame to coincide with the initial reference frame). We also choose a large positive number W for the blocks corresponding to v_0 and ω_0.

The variance $\Sigma_n(t)$ is usually available from the analysis of the feature tracking algorithm. We assume that the tracking error is independent in each point, and therefore Σ_n is block diagonal. We choose each block to be the covariance of the measurement $\mathbf{y}^i(t)$ (in the current implementation they are diagonal and equal to 1 pixel std.). The variance $\Sigma_w(t)$ is a design parameter that is available for tuning. We describe the procedure in section 13.6. Finally, set

$$\begin{cases} \hat{x}(0|0) \doteq [\mathbf{y}^{4T}_0, \dots \mathbf{y}^{NT}_0,\ \rho_0^2, \dots, \rho_0^N,\ T_0^T,\ \Omega_0^T,\ v_0^T,\ \omega_0^T]^T \\ P(0|0) = P_0. \end{cases} \tag{13.47}$$

Transient

During the first transient of the filter, we do not allow for new features to be acquired. Whenever a feature is lost, its state is removed from the model and its best current estimate is placed in a storage vector. If the feature was associated with the scale factor, we proceed as in section 13.5. The transient can be tested as either a threshold on the innovation, a threshold on the variance of the estimates, or a fixed time interval. We choose a combination with the time set to 30 frames, corresponding to one second of video.

The recursion to update the state x and the variance P proceed as follows: Let f and h denote the state and measurement model, so that

equation (**??**) can be written in concise form as

$$\begin{cases} x(t+1) = f(x(t)) + w(t) & w(t) \sim \mathcal{N}(0, \Sigma_w) \\ y(t) = h(x(t)) + n(t) & n(t) \sim \mathcal{N}(0, \Sigma_n) \end{cases} \tag{13.48}$$

We then have

Prediction:

$$\begin{cases} \hat{x}(t+1|t) = f(x(t|t)) \\ P(t+1|t) = F(t)P(t|t)F^T(t) + \Sigma_w \end{cases} \tag{13.49}$$

Update:

$$\begin{cases} \hat{x}(t+1|t+1) = \hat{x}(t+1|t) + L(t+1)\left(y(t+1) - h(\hat{x}(t+1|t))\right) \\ P(t+1|t+1) = \Gamma(t+1)P(t+1|t)\Gamma^T(t+1) + L(t+1)\Sigma_n(t+1)L^T(t+1). \end{cases} \tag{13.50}$$

Gain:

$$\begin{cases} \Gamma(t+1) \doteq I - L(t+1)H(t+1) \\ L(t+1) \doteq P(t+1|t)H^T(t+1)\Lambda^{-1}(t+1) \\ \Lambda(t+1) \doteq H(t+1)P(t+1|t)H^T(t+1) + \Sigma_n(t+1) \end{cases} \tag{13.51}$$

Linearization:

$$\begin{cases} F(t) \doteq \frac{\partial f}{\partial x}(\hat{x}(t|t)) \\ H(t+1) \doteq \frac{\partial h}{\partial x}(\hat{x}(t+1|t)) \end{cases} \tag{13.52}$$

Let $\mathbf{e}_i$ be the i-th canonical vector in $\mathbb{R}^3$ and define $Y^i(t) \doteq e^{\hat{\Omega}(t)}[\mathbf{y}_0^i(t)]\rho^i(t) + T(t)$, $Z^i(t) \doteq \mathbf{e}_3^T Y^i(t)$. The i-th block-row ($i = 1, \ldots, N$) $H_i(t)$ of the matrix $H(t)$ can be written as $H_i = \frac{\partial y^i}{\partial Y^i}\frac{\partial Y^i}{\partial x} \doteq \Pi_i \frac{\partial Y^i}{\partial x}$ where the time argument t has been omitted for simplicity of notation. It is easy to check that $\Pi_i = \frac{1}{Z^i}\begin{bmatrix} I_2 & -\pi(Y^i) \end{bmatrix}$ and

$$\frac{\partial Y^i}{\partial x} = \begin{bmatrix} \underbrace{\begin{bmatrix} 0 \ \ldots \ \frac{\partial Y^i}{\partial \mathbf{y}_0^1} \ \ldots \ 0 \end{bmatrix}}_{2N-6} & \underbrace{\begin{bmatrix} 0 \ \ldots \ \frac{\partial Y^i}{\partial \rho^i} \ \ldots \ 0 \end{bmatrix}}_{N-1} & \underbrace{\frac{\partial Y^i}{\partial T}}_{3} & \underbrace{\frac{\partial Y^i}{\partial \Omega}}_{3} & \underbrace{0}_{3} & \underbrace{0}_{3} \end{bmatrix}.$$

The partial derivatives in the previous expression are given by

$$\begin{cases} \frac{\partial Y^i}{\partial \mathbf{y}_0^i} = e^{\hat{\Omega}} \begin{bmatrix} I_2 \\ 0 \end{bmatrix} \rho^i \\ \frac{\partial Y^i}{\partial \rho^i} = e^{\hat{\Omega}} \begin{bmatrix} \mathbf{y}_0^i \end{bmatrix} \\ \frac{\partial Y^i}{\partial T} = I \\ \frac{\partial Y^i}{\partial \Omega} = \begin{bmatrix} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_1}\mathbf{y}_0^i\rho^i & \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_2}\mathbf{y}_0^i\rho^i & \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_3}\mathbf{y}_0^i\rho^i \end{bmatrix} \end{cases}$$

The linearization of the state equation involves derivatives of the logarithm function in SO(3) which is available as a Matlab function in the software

distribution [**?**] and will not be reported here. We shall use the following notation:

$$\frac{\partial Log_{SO(3)}(R)}{\partial R} \doteq \left[\begin{array}{cccc} \frac{\partial Log_{SO(3)}(R)}{\partial r_{11}} & \frac{\partial Log_{SO(3)}(R)}{\partial r_{21}} & \cdots & \frac{\partial Log_{SO(3)}(R)}{\partial r_{33}} \end{array} \right]$$

where r_{ij} is the element in position (i, j) of R. Let us denote $R \doteq e^{\hat{\omega}} e^{\hat{\Omega}}$; the linearization of the state equation can be written in the following form:

$$F \doteq \begin{bmatrix} I_{2N-6} & 0 & 0 & 0 & 0 & 0 \\ 0 & I_{N-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{\hat{\omega}} & 0 & I & \left[\frac{\partial e^{\hat{\omega}}}{\partial \omega_1} T \quad \frac{\partial e^{\hat{\omega}}}{\partial \omega_2} T \quad \frac{\partial e^{\hat{\omega}}}{\partial \omega_3} T \right] \\ 0 & 0 & 0 & \frac{\partial Log_{SO(3)}(R)}{\partial R} \frac{\partial R}{\partial \Omega} & 0 & \frac{\partial Log_{SO(3)}(R)}{\partial R} \frac{\partial R}{\partial \omega} \\ 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

where

$$\frac{\partial R}{\partial \Omega} = \left[\left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_1} \right)^{\vee} \quad \left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_2} \right)^{\vee} \quad \left(e^{\hat{\omega}} \frac{\partial e^{\hat{\Omega}}}{\partial \Omega_3} \right)^{\vee} \right]$$

and

$$\frac{\partial R}{\partial \omega} \doteq \left[\left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_1} e^{\hat{\Omega}} \right)^{\vee} \quad \left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_2} e^{\hat{\Omega}} \right)^{\vee} \quad \left(\frac{\partial e^{\hat{\omega}}}{\partial \omega_3} e^{\hat{\Omega}} \right)^{\vee} \right]$$

and the bracket $(\cdot)^{\vee}$ indicates that the content has been organized into a column vector.

Regime

Whenever a feature disappears, we simply remove it from the state as during the transient. However, after the transient a feature selection module works in parallel with the filter to select new features so as to maintain roughly a constant number (equal to the maximum that the hardware can handle in real time), and to maintain a distribution as uniform as possible across the image plane. We implement this by randomly sampling points on the plane, searching then around that point for a feature with enough brightness gradient (we use an SSD-type test [**?**]).

Once a new point-feature is found (one with enough contrast along two independent directions), a new filter (which we call a "subfilter") is initialized based on the model (13.43). Its evolution is given by

Initialization:

$$\begin{cases} \hat{\mathbf{y}}^i_\tau(\tau|\tau) = \mathbf{y}^i_\tau(\tau) \\ \hat{\rho}^i_\tau(\tau|\tau) = 1 \\ P_\tau(\tau|\tau) = \begin{bmatrix} \ddots & & & \\ & \Sigma_{n^i}(\tau) & & \\ & & \ddots & \\ & & & M \end{bmatrix} \end{cases} \tag{13.53}$$

Prediction:

$$\begin{cases} \hat{\mathbf{y}}^i_\tau(t+1|t) = \hat{\mathbf{y}}^i_\tau(t|t) \\ \hat{\rho}^i_\tau(t+1|t) = \hat{\rho}^i_\tau(t|t) \\ P_\tau(t+1|t) = P_\tau(t+1|t) + \Sigma_w(t) \end{cases} \qquad t > \tau \tag{13.54}$$

Update:

$$\begin{bmatrix} \hat{\mathbf{y}}^i_\tau(t+1|t+1) \\ \hat{\rho}_\tau(t+1|t+1) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{y}}^i_\tau(t+1|t) \\ \hat{\rho}_\tau(t+1|t) \end{bmatrix} +$$

$$L_\tau(t+1)\left(\mathbf{y}^i(t) - \pi(\exp(\widehat{\Omega}(t))\left[\exp(\widehat{\Omega}(\tau))\right]^{-1}\left[\mathbf{y}^i(t)\rho^i(t) - T(\tau)\right] + T(t))\right)$$

and P_τ is updated according to a Riccati equation in all similar to (13.50).

After a probation period, whose length is chosen according to the same criterion adopted for the main filter, the feature is inserted into the state using the transformation (13.44). The initial variance is chosen to be the variance of the estimation error of the subfilter.

Tuning

The variance $\Sigma_w(t)$ is a design parameter. We choose it to be block diagonal, with the blocks corresponding to $T(t)$ and $\Omega(t)$ equal to zero (a deterministic integrator). We choose the remaining parameters using standard statistical tests, such as the Cumulative Periodogram of Bartlett [**?**]. The idea is that the parameters in Σ_w are changed until the innovation process $\epsilon(t) \doteq y(t) - h(\hat{x}(t))$ is as close as possible to being white. The periodogram is one of many ways to test the "whiteness" of a stochastic process. In practice, we choose the blocks corresponding to $\mathbf{y}^i_0$ equal to the variance of the measurements, and the elements corresponding to ρ^i all equal to σ_ρ. We then choose the blocks corresponding to v and ω to be diagonal with element σ_v, and then we change σ_v relative to σ_ρ depending on whether we want to allow for more or less regular motions. We then change both, relative to the variance of the measurement noise, depending on the level of desired smoothness in the estimates.

Tuning nonlinear filters is an art, and this is not the proper venue to discuss this issue. Suffices to say that we have only performed the procedure

once and for all. We then keep the same tuning parameters no matter what the motion, structure and noise in the measurements.

Chapter 14

Step-by-step building of a 3-D model from images

In this chapter we present a step-by-step demonstration of how to build a 3-Dmodel from images. The sections in this chapter follow closely matlab routines that are available from the book's website.

Chapter 15

Extensions, applications and further research directions

This chapter highlights research directions of great practical importance that have not yet reached the point where robust algorithms are available.

Part V

Appendices

Appendix A
Basic facts from linear algebra

Since in computer vision, we will mostly deal with real vector spaces, we here introduce some important facts about linear algebra for real vector spaces.[1] $\mathbb{R}^n$ is then a natural model for a real vector space of n-dimension. If $\mathbb{R}^n$ is considered as a linear space, its elements, so-called vectors, are complete under two basic operations: scalar multiplication and vector summation. That is, given any two vectors $x, y \in \mathbb{R}^n$, and any two real scalars $\alpha, \beta \in \mathbb{R}$, we may obtain a new vector $z = \alpha x + \beta y$ in $\mathbb{R}^n$. Linear algebra mainly studies properties of the so-called *linear transformations* among different real vector spaces. Since such transformations can typically represented as matrices, linear algebra to a large extent studies the properties of matrices.

A.1 Linear maps and linear groups

A *linear transformation* from a real linear (vector) space $\mathbb{R}^n$ to $\mathbb{R}^m$ is defined as a map $L : \mathbb{R}^n \to \mathbb{R}^m$ such that

- $L(x + y) = L(x) + L(y)\ \forall, \quad x, y \in \mathbb{R}^n$
- $L(\alpha x) = \alpha L(x), \quad \forall x \in \mathbb{R}^n, \alpha \in \mathbb{R}.$

[1] We here do not intend to provide a formal introduction to linear algebra. Instead, we do assume that the reader has basic knowledge in linear algebra.

Clearly, with respect to the standard bases of $\mathbb{R}^n$ and $\mathbb{R}^m$, the map L can be *represented* by a matrix $A \in \mathbb{R}^{m \times n}$ such that

$$L(x) = Ax, \quad \forall x \in \mathbb{R}^n. \tag{A.1}$$

The set of all (real) $m \times n$ matrices are denoted as $\mathcal{M}(m, n)$. If viewed as a linear space, $\mathcal{M}(m, n)$ can be identified as the space $\mathbb{R}^{mn}$. By abuse of language, we sometimes simply refer to a linear map L by its representation (matrix) A.

If $n = m$, the set $\mathcal{M}(n, n) \doteq \mathcal{M}(n)$ forms an algebraic structure called *ring* (over the field $\mathbb{R}$). That is, matrices in $\mathcal{M}(n)$ are closed under matrix multiplication and summation: if A, B are two $n \times n$ matrices, so are $C = AB$ and $D = A + B$. If we consider the ring of all $n \times n$ matrices, its group of units $GL(n)$ – which consists of all $n \times n$ *invertible* matrices and is called the *general linear group* – can be identified with the set of invertible linear maps from $\mathbb{R}^n$ to $\mathbb{R}^n$:

$$L : \mathbb{R}^n \to \mathbb{R}^n;\ x \mapsto L(x) = Ax \mid A \in GL(n). \tag{A.2}$$

Now let us further consider $\mathbb{R}^n$ with its standard inner product structure. That is, given two vectors $x, y \in \mathbb{R}^n$, we define their *inner product* to be $\langle x, y \rangle = x^T y$. We say that a linear transformation A (from $\mathbb{R}^n$ to itself) is *orthogonal* if it preserves such inner product:

$$\langle Ax, Ay \rangle = \langle x, y \rangle, \quad \forall\ x, y \in \mathbb{R}^n. \tag{A.3}$$

The set of $n \times n$ orthogonal matrices forms the *orthogonal group* $O(n)$. If R is a matrix representative of an orthogonal transformation, expressed relative to an orthonormal reference frame, then it is easy to see that the orthogonal group is characterized as

$$O(n) = \{R \in GL(n) \mid R^T R = I\}. \tag{A.4}$$

The determinant of an orthogonal matrix can be ± 1. The subgroup of $O(n)$ with unit determinant is called the *special orthogonal group* $SO(n)$.

A.2 Gram-Schmidt orthonormalization

A matrix in $GL(n)$ has n independent rows (columns). A matrix in $O(n)$ has orthonormal rows (columns). The Gram-Schmidt procedure can be viewed as a map between $GL(n)$ and $O(n)$, for it transforms a nonsingular matrix into an orthonormal one. Call $\mathcal{L}_+(n)$ the subset of $GL(n)$ consisting of lower triangular matrices with positive elements along the diagonal. Such matrices form a subgroup of $GL(n)$. Then we have

Theorem A.1. *(Gram-Schmidt)* $\forall A \in GL(n)\ \exists! Q \in \mathcal{L}_+(n),\ R \in O(n)$ *such that*

$$A = QR \tag{A.5}$$

Proof. Contrary to convention elsewhere, within this proof all vectors stand for row vectors. That is, if v is an n-dimensional row vector, it is of the form: $v = [v_1, v_2, \ldots, v_n]$ which does not have a transpose on. Denote the i^{th} row vector of the given matrix A as a_i for $i = 1, \ldots, n$. The proof consists in constructing L and E iteratively from the row vectors a_i:

$$\begin{array}{lcll} q_1 & \doteq & a_1 & \longrightarrow \quad r_1 \doteq q_1/\|q_1\| \\ q_2 & \doteq & a_2 - \langle a_2, r_1\rangle r_1 & \longrightarrow \quad r_2 \doteq q_2/\|q_2\| \\ \vdots & \doteq & \vdots & \longrightarrow \quad \vdots \\ q_n & \doteq & a_n - \sum_{i=1}^{n-1} \langle a_{i+1}, r_i\rangle r_i & \longrightarrow \quad r_n \doteq q_n/\|q_n\| \end{array}$$

Then $R = [r_1^T \ldots r_n^T]^T$ and the matrix Q is obtained as

$$Q = \begin{bmatrix} \|q_1\| & 0 & \ldots & 0 \\ \langle a_2, r_1\rangle & \|q_2\| & \ldots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \langle a_2, r_1\rangle & \cdots & \langle a_n, r_{n-1}\rangle & \|q_n\| \end{bmatrix}$$

By construction R is orthogonal, i.e. $RR^T = R^T R = I$. □

Remark A.1. *Gram-Schmidt's procedure has the peculiarity of being* causal*, in the sense that the k-th column of the transformed matrix depends only upon rows with index $l \leq k$ of the original matrix. The choice of the name E for the orthogonal matrix above is not random. In fact we will view the Kalman filter as a way to perform a Gram-Schmidt orthonormalization on a peculiar Hilbert space, and the outcome E of the procedure is traditionally called the* innovation*.*

A.3 Symmetric matrices

Definition A.1. $S \in \mathbb{R}^{n\times n}$ *is* symmetric *iff* $S^T = S$.

Theorem A.2. *S is symmetric then*

1. *Let (v, λ) be eigenvalue-eigenvector pairs. If $\lambda_i \neq \lambda_j$ then $v_i \perp v_j$, i.e. eigenvectors corresponding to distinct eigenvalues are orthogonal.*
2. *$\exists$ n orthonormal eigenvectors of S, which form a basis for $\mathbb{R}^n$.*
3. *$S \geq 0$ iff $\lambda_i \geq 0$ $\forall i = 1, \ldots, n$, i.e. S is positive semi-definite iff all eigenvalues are non-negative.*
4. *if $S \geq 0$ and $\lambda_1 \geq \lambda_2 \cdots \lambda_n$ then $\max_{\|x\|_2=1}\langle x, Sx\rangle = \lambda_1$ and $\min_{\|x\|_2=1}\langle x, Sx\rangle = \lambda_n$.*

Remark A.2.

- *from point (3) of the previous theorem we see that if* $V = [v_1, v_2, \ldots, v_n]$ *is the matrix of all the eigenvectors, and* $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ *is the diagonal matrix of the corresponding eigenvalues, then we can write* $S = V\Lambda V^T$*; note that* V *is orthogonal.*
- *Proofs of the above claims are easy exercises.*

Definition A.2. *Let* $A \in \mathbb{R}^{m\times n}$*, then we define the induced 2-norm of* A *as an operator between* $\mathbb{R}^n$ *and* $\mathbb{R}^m$ *as*

$$\|A\| \doteq \max_{\|x\|_2=1} \|Ax\|_2^2 = \max_{\|x\|_2=1} \langle x, A^T Ax\rangle.$$

Remark A.3.

- *Similarly other induced operator norms on* A *can be defined starting from different norms on the domain and co-domain spaces on which* A *operates.*
- *let* A *be as above, then* A^TA *is clearly symmetric and positive semi-definite, so it can be diagonalized by a orthogonal matrix* V*. The eigenvalues, being non-negative, can be written as* σ_i^2*. By ordering the columns of* V *so that the eigenvalue matrix* Λ *has decreasing eigenvalues on the diagonal, we see, from point (e) of the previous theorem, that* $A^TA = V\,diag(\sigma_1^2, \ldots, \sigma_n^2)V^T$ *and* $\|A\|_2 = \sigma_1$*.*

A.4 Structure induced by a linear map

- Let A be an operator from a vector space $\mathbb{R}^n$ to a space $\mathbb{R}^m$
- Define the *null space* of A, denoted as $Nu(A)$ to be a subspace of $\mathbb{R}^n$ such that $Ax = 0$ if and only if $x \in Nu(A)$; define the *range* or span of A, denoted as $Ra(A)$ or span(A), to be a subspace of $\mathbb{R}^m$ such that $y = Ax$ for some $x \in \mathbb{R}^n$ if and only if $y \in Ra(A)$.
- Given a subspace S of $\mathbb{R}^n$, we define its *orthogonal supplement* to be the subspace $S^\perp \subseteq \mathbb{R}^n$ such that $x \in S^\perp$ if and only if $x^Ty = 0$ for all $y \in S$. We denote $\mathbb{R}^n = S \oplus S^\perp$.
- Then with respect any linear map A from $\mathbb{R}^n$ to $\mathbb{R}^m$, $\mathbb{R}^n$ can be decomposed as:

$$\mathbb{R}^n = Nu(A) \oplus Nu(A)^\perp$$

- And $\mathbb{R}^m$ can be decomposed as:

$$\mathbb{R}^m = Ra(A) \oplus Ra(A)^\perp.$$

Theorem A.3. *Let* A *is a linear map from* $\mathbb{R}^n$ *to* $\mathbb{R}^m$*; then*

a) $Nu(A)^\perp = Ra(A^T)$

b) $Ra(A)^{\perp} = Nu(A^T)$

c) $Nu(A^T) = Nu(AA^T)$

d) $Ra(A) = Ra(AA^T)$.

Proof. To prove $Nu(AA^T) = Nu(A^T)$, we have:

- $AA^Tx = 0 \Rightarrow \langle x, AA^Tx\rangle = \|A^Tx\|^2 = 0 \Rightarrow A^Tx = 0$, hence $Nu(AA^T) \subseteq Nu(A^T)$.
- $A^Tx = 0 \Rightarrow AA^Tx = 0$, hence $Nu(AA^T) \supseteq Nu(A^T)$.

To prove $Ra(AA^T) = Ra(A)$, we first need to prove that $\mathbb{R}^n$ is a direct sum of $Ra(A^T)$ and $Nu(A)$, i.e. part *a*) of the theorem. Part *b*) can then be proved similarly. We prove this by showing that a vector x is in $Nu(A)$ if and only if it is orthogonal to $Ra(A^T)$: $x \in Nu(A) \Leftrightarrow \langle A^Tx, y\rangle = 0, \forall y \in Ra(A^T) \Leftrightarrow \langle x, Ay\rangle = 0, \forall y$. Hence $Nu(A)$ is exactly the subspace which is orthogonal supplementary to $Ra(A^T)$ (sometimes denoted as $Ra(A^T)^{\perp}$). Therefore $\mathbb{R}^n$ is a direct sum of $Ra(A^T)$ and $Nu(A)$. Let $Img_A(S)$ denote the image of a subspace S under the map A. Then we have: $Ra(A) = Img_A(\mathbb{R}^n) = Img_A(Ra(A^T)) = Ra(AA^T)$ (in the second equality we used the fact that $\mathbb{R}^n$ is a direct sum of $Ra(A^T)$ and $Nu(A)$.) □

In fact the same result still holds even if the domain of the linear map A is replaced by an infinite dimensional linear space with an inner product (i.e. $\mathbb{R}^n$ is replaced by a Hilbert space). In that case, this theorem is also known as the *Finite Rank Operator Fundamental Lemma*.

A.5 The Singular Value Decomposition (SVD)

The SVD is a useful tool to capture essential features of a linear operator, such as the rank, range space, null space, induced norm etc. and to "generalize" the concept of "eigenvalue- eigenvector" pair. The computation of the SVD is numerically well-conditioned, so it makes sense to try to solve some typical linear problems as matrix inversions, calculation of rank, best 2-norm approximations, projections and fixed-rank approximations, in terms of the SVD of the operator.

A.5.1 Algebraic derivation

Given a matrix $A \in \mathbb{R}^{m\times n}$, the following theorem and its proof follow that in [CD91].

Theorem A.1 (Singular Value Decomposition). *Let $A \in \mathbb{R}^{m\times n}$ be a matrix of rank r. Then there exist matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$, and $\Sigma_1 \in \mathbb{R}^{r\times r}$ such that:*

1. *$V = [V_1 : V_2], V_1 \in \mathbb{R}^{n\times r}$, satisfies:*
   V is unitary, i.e. $V^TV = I_{n\times n}$;
   $Ra(V_1) = Ra(A^T)$, and the columns of V_1 form an orthonormal basis of $Ra(A^T)$;
   $Ra(V_2) = Nu(A)$, and the columns of V_2 form an orthonormal basis of $Nu(A)$;
   The columns of V form a complete orthonormal basis of eigenvectors of A^TA.

2. *$U = [U_1 : U_2], U_1 \in \mathbb{R}^{m\times r}$, satisfies:*
   U is unitary, i.e. $U^TU = I_{m\times m}$;
   $Ra(U_1) = Ra(A)$, and the columns of U_1 form an orthonormal basis of $Ra(A)$;
   $Ra(U_2) = Nu(A^T)$, and the columns of U_2 form an orthonormal basis of $Nu(A^T)$;
   The columns of U form a complete orthonormal basis of eigenvectors of AA^T.

3. *$\Sigma_1 = diag(\sigma_1, \sigma_2, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$ such that $\sigma_1 \geq \sigma_2 \geq \cdots, \geq \sigma_r > 0$.*
   $A \in \mathbb{R}^{m\times n}$ has a dyadic expansion:

$$A = U_1\Sigma_1V_1^T, \quad \text{or equivalently,} \quad A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

   where u_i, v_i are the columns of U_1 and V_1 respectively.

4. *$A \in \mathbb{R}^{m\times n}$ has a singular value decomposition (SVD):*

$$A = U\Sigma V^T, \quad \text{with} \quad \Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}_{m\times n}.$$

Proof. 1. $A \in \mathbb{R}^{m\times n}$ has rank r, hence the nonnegative (or, equivalently, positive semidefinite) matrix A^TA has rank r according to Theorem A.3. It has n nonnegative eigenvalues σ_i^2 ordered as:

$$\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_r^2 > 0 = \sigma_{r+1}^2 = \cdots = \sigma_n^2$$

to which corresponds a complete orthonormal eigenvector basis $(v_i)_{i=1}^n$ of A^TA. This family of vectors (in $\mathbb{R}^n$) form the columns of a unitary $n \times n$ matrix, say, V. From Theorem A.3, $Ra(A^TA) = Ra(A^T)$ and $Nu(A^TA) = Nu(A)$, the properties listed in 1 follow.

2. Define a diagonal matrix $\Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$. We then have $A^TAV_1 = V_1\Sigma_1^2$, hence $(AV_1\Sigma_1^{-1})^T(AV_1\Sigma_1^{-1}) = I_{r\times r}$. This defines a $m \times r$ matrix:

$$U_1 = AV_1\Sigma_1^{-1}. \tag{A.6}$$

Then $U_1^TU_1 = I_{r\times r}$. Since A^TA and AA^T both have exactly r nonzero eigenvalues, it follows that the columns of U_1 form an orthonormal basis for $Ra(AA^T)$ and $Ra(A)$. Thus the properties of U_1 listed in 2 hold. Now define

an $m \times (m-r)$ matrix U_2 with orthonormal columns which are orthogonal to columns of U_1. Then $U = [U_1 : U_2]$ is clearly an unitary matrix. From the proof of the theorem A.3, columns of U_2 form an orthonormal basis of $Nu(A^T)$ or $Nu(AA^T)$. Therefore, columns of U_2 are all the eigenvectors corresponding to the zero eigenvalue. Hence columns of U form a complete orthonormal basis of eigenvectors of AA^T. List 2 is then fully proven.

3. From the definition of U_1 in (A.6), we have:

$$A = U_1 \Sigma_1 V_1^T.$$

The dyadic expansion directly follows.

4. The singular value decomposition follows because:

$$A[V_1 : V_2] = [U_1 \Sigma_1 : 0] = [U_1 : U_2]\Sigma \quad \Rightarrow \quad A = U \Sigma V^T.$$

□

After we have gone through all the trouble proving this theorem, you must know that SVD has become a numerical routine available in many computational softwares such as MATLAB. Within MATLAB, to compute the SVD of a given $m \times n$ matrix A, simply use the command "$[U, S, V] = SVD(A)$" which returns matrices U, S, V satisfying $A = USV^T$ (where S represents Σ as defined above).

A.5.2 Geometric interpretation

Theorem A.4. *Let a square matrix $A \in \mathbb{R}^{n \times n} = U \Sigma V^T$, then A maps the unit sphere $\mathbb{S}^{n-1} \doteq \{x \in \mathbb{R}^n : \|x\|_2 = 1\}$ to an ellipsoid with half-axes $\sigma_i u_i$*

Proof. let x, y be such that $Ax = y$. $\{v_1, \ldots, v_n\}$ is an orthonormal basis for $\mathbb{R}^n$. With respect to such basis x has coordinates $[\langle v_1, x\rangle, \langle v_2, x\rangle, \ldots, \langle v_n, x\rangle]$. Idem for $\{u_i\}$. Let $y = \sum_{i=1}^n y_i u_i \rightarrow Ax = \sum_{i=1}^n \sigma_i u_i v_i^T x = \sum_{i=1}^n \sigma_i u_i \langle v_i, x\rangle = \sum_{i=1}^n y_i u_i = y$. Hence $\sigma_i \langle v_i, x\rangle = y_i$. Now $\|x\|_2^2 = \sum_{i=1}^n \langle v_i, x\rangle^2 = 1$, $\forall x \in \mathbb{S}^{n-1}$, from which we conclude $\sum_{i=1}^n y_i^2/\sigma_i^2 = 1$, which represents the equation of an ellipsoid with half-axes of length σ_i. □

A.5.3 Some properties of the SVD

The problems involving orthogonal projections onto invariant subspaces of A, as Linear Least Squares (LLSE) or Minimum Energy problems, are easily solved using the SVD.

Definition A.3 (Generalized (Moore-Penrose) Inverse). *Let $A \in \mathbb{R}^{m \times n}$, $A = U \Lambda V^T$ where Λ is the diagonal matrix with diagonal elements $(\lambda_1, \ldots \lambda_r, 0 \ldots 0)$; then define the generalized inverse of A to be*

$$A^\dagger = U \Lambda_{(r)}^{-1} V^T, \quad \textit{where} \quad \Lambda_{(r)}^{-1} = diag(\lambda_1^{-1}, \ldots, \lambda_r^{-1}, 0, \ldots, 0)$$

The so-defined generalized inverse has the following properties:

Theorem A.5.

- $AA^\dagger A = A$
- $A^\dagger AA^\dagger = A^\dagger$

The generalized inverse can be used to solve linear equations in general:

Theorem A.6 (Least squares solution of a linear systems). *Consider the problem $Ax = b$ with $A \in \mathbb{R}^{m\times n}$ of rank $p \leq min(m, n)$, then the solution $\hat{x}$ that minimizes $\|A\hat{x} - b\|$ is given by $\hat{x} = A^\dagger b$.*

One of the most important properties of the SVD has to deal with fixed-rank approximations of a given operator. Given A as an operator from a space X to a space Y of rank n, we want to find an operator B from the same spaces such that it has rank $p < n$ fixed and $\|A - B\|_F$ is minimal, where the F indicates the Frobenius norm (in this context it is the sum of the singular values).

If we had the usual 2-norm and we calculate the SVD of $A = U\Sigma V^T$, then by simply setting all the singular values but the first p to zero, we have an operator $B \doteq U\Sigma_{(p)}V^T$, where $\Sigma_{(p)}$ denotes a matrix obtained from Σ by setting to zero the elements on the diagonal after the p^{th}, which has exactly the same two norm of A and satisfies the requirement on the rank. It is not difficult to see the following result

Theorem A.7 (Fixed rank approximations). *Let A, B be defined as above, then $\|A - B\|_F = \sigma_{p+1}$. Furthermore such norm is the minimum achievable.*

Proof: easy exercise; follows directly from the orthogonal projection theorem and the properties of the SVD given above.

The following two results have something to do with the sensitivity of solving linear equation of the form $Ax = b$.

Proposition A.1 (Perturbations). *Consider a non-singular matrix $A \in \mathbb{R}^{n\times n}$ (if A is singular substitute its inverse by the moore-penrose pseudo-inverse). Let δA be a full-rank perturbation. Then*

- $|\sigma_k(A + \delta A) - \sigma_k(A)| \leq \sigma_1(\delta A), \quad \forall k = 1, \ldots, n$
- $\sigma_n(A\delta A) \geq \sigma_n(A)\sigma_n(\delta A)$
- $\sigma_1(A^{-1}) = \frac{1}{\sigma_n(A)}$

Proposition A.2 (Condition number). *Consider the problem $Ax = b$, and consider a "perturbed" full rank problem $(A + \delta A)(x + \delta x) = b$. Since $Ax = b$, then to first order approximation $\delta x = -A^\dagger \delta Ax$. Hence $\|\delta x\| \leq$*

$\|A^\dagger\|\|\delta A\|\|x\|$, *from which*

$$\frac{\|\delta x\|}{\|x\|} = \|A^\dagger\|\|A\|\frac{\|\delta A\|}{\|A\|} \doteq k(A)\frac{\|\delta A\|}{\|A\|}$$

where $k(A) = \|A^\dagger\|\|A\|$ *is called the condition number of* A. *It easy to see that* $k(A) = \sigma_1/\sigma_n$.

Appendix B
Least-square estimation and filtering

B.1 Linear least-variance estimators of random vectors

Let $T : \mathbb{R}^n \to \mathbb{R}^m;\ X \mapsto Y$ be a transformation acting between two spaces of random vectors with instances in $\mathbb{R}^m$ and $\mathbb{R}^n$ (the model generating the data). We are interested in building an estimator for the random vector X, given measurements of instances of the random vector Y. An estimator is a function $T^* : \mathbb{R}^m \to \mathbb{R}^n;\ Y \mapsto \hat{X} = T^*(Y)$, which solves an optimization problem of the form

$$\hat{T}^* \doteq \arg\min_{T \in \mathcal{T}} \mathcal{C}(X - T^*(Y))_{\mathcal{T}} \tag{B.1}$$

where $\mathcal{T}$ is a suitable chosen class of functions and $\mathcal{C}(\cdot)_{\mathcal{T}}$ some cost in the $X-$space.

We concentrate on one of the simplest possible choices, which correspond to *minimum variance affine* estimators:

$$\mathcal{T} \doteq \{A \in \mathbb{R}^{n \times m}; b \in \mathbb{R}^n \mid T^*(Y) = AY + b\} \tag{B.2}$$

$$\mathcal{C}(\cdot)_{\mathcal{T}} \doteq E\left[\|\cdot\|^2\right] \tag{B.3}$$

where the latter operator takes the expectation of the squared euclidean norm of the random vector Y. Therefore, we seek for

$$(\hat{A}, \hat{b}) \doteq \arg\min_{A,b} E\left[\|X - (AY + b)\|^2\right] \tag{B.4}$$

We call $\mu_X \doteq E[X]$ and $\Sigma_X \doteq E[XX^T]$, and similarly for Y. First notice that if $\mu_X = \mu_Y = 0$, then $\hat{b} = 0$. Therefore, consider the centered vectors $\bar{X} \doteq X - \mu_X$ and $\bar{Y} \doteq Y - \mu_Y$ and the reduced problem

$$\hat{A} \doteq \arg\min_A E\left[\|\bar{X} - A\bar{Y}\|^2\right]. \tag{B.5}$$

Now observe that

$$\begin{aligned} E\left[\|X - AY - b\|^2\right] &= E\left[\|A\bar{Y} - \bar{X} + (A\mu_X + b - \mu_Y)\|^2\right] \\ &= E\left[\|\bar{X} - A\bar{Y}\|^2\right] + \|A\mu_X + b - \mu_Y\|^2. \end{aligned} \tag{B.6}$$

Hence, if we assume for a moment that we have found $\hat{A}$ that solves the problem (B.5), then trivially

$$\hat{b} = \mu_X - \hat{A}\mu_Y \tag{B.7}$$

annihilates the second term of eq. (B.6).

Therefore, we will concentrate on the case $\mu_X = \mu_Y = 0$ without loss of generality.

B.1.1 Projections onto the range of a random vector

The set of all random variables Z_i defined on the same probability space, with *zero-mean* $E[Z_i] = 0$ and *finite variance* $\Sigma_{Z_i} < \infty$ is a *Hilbert space* with the inner-product given by

$$\langle Z_i, Z_j \rangle_{\mathcal{H}} \doteq \Sigma_{Z_i Z_j} = E[Z_i Z_j]. \tag{B.8}$$

In this space the notion of orthogonality corresponds to the notion of *uncorrelatedness*. The components of a random vector Y define a subspace of such Hilbert space:

$$\mathcal{H}(Y) = \text{span}\langle Y_1, \ldots, Y_m \rangle \tag{B.9}$$

where the span is intended over the reals.[1] We say that the subspace $\mathcal{H}(Y)$ is *full rank* if $\Sigma_Y = E[YY^T] > 0$.

The structure of a Hilbert space allows us to make use of the concept of *orthogonal projection* of a random variable onto the span of a random vector:

$$\begin{aligned} \hat{Z} = \text{pr}_{\langle \mathcal{H}(Y) \rangle}(X) &\Leftrightarrow \langle X - \hat{Z}, Z \rangle_{\mathcal{H}} = 0, \quad \forall\ Z \in \mathcal{H}(Y) \\ &\Leftrightarrow \langle X - \hat{Z}, Y_i \rangle_{\mathcal{H}} = 0, \quad \forall\ i = 1 \ldots n \qquad \text{(B.10)} \\ &\doteq \hat{E}[X|Y] \qquad \text{(B.11)} \\ &\doteq \hat{X}(Y) \qquad \text{(B.12)} \end{aligned}$$

[1] So $\mathcal{H}(Y)$ is the space of random variables which are linear combination of Y_i, $i = 1, \ldots, m$.

The notation $\hat{E}[X|Y]$ is often used for the projection of X over the span of Y [2].

B.1.2 Solution for the linear (scalar) estimator

Let $Z = AY$ be a linear estimator for the random *variable* $X \in \mathbb{R}$; $A \in \mathbb{R}^n$ is a row-vector, and $Y \in \mathbb{R}^m$ an m-dimensional column random vector. The least-square estimate $\hat{Z}$ is given by the choice of A that solves the following problem:

$$\hat{A} = \arg\min_A \|AY - X\|^2_{\mathcal{H}} \tag{B.13}$$

where $\|\cdot\|^2_{\mathcal{H}} = E\left[\|\cdot\|^2\right]$ is the norm induced by the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$.

Theorem B.1. *The solution $\hat{Z} = \hat{A}Y$ to the problem (B.13) exists, is unique and corresponds to the orthogonal projection of X onto the span of Y:*

$$\hat{Z} = \mathrm{pr}_{\langle\mathcal{H}(Y)\rangle}(X) \tag{B.14}$$

The proof is an easy exercise. In the following we report an explicit construction of the best estimator $\hat{A}$. From substituting the expression of the estimator onto the definition of orthogonal projection (B.12), we get

$$0 = \langle X - \hat{A}Y, Y_i\rangle_{\mathcal{H}} = E[(X - \hat{A}Y)Y_i] \tag{B.15}$$

which holds iff $E[XY_i] = \hat{A}E[YY_i]$, $\forall\ i = 1 \dots n$. In a row-vector notation we write

$$\begin{aligned} E[XY^T] &= \hat{A}E[YY^T] \\ \Sigma_{XY} &= \hat{A}\Sigma_Y \end{aligned} \tag{B.16}$$

which, provided that $\mathcal{H}(Y)$ is full rank, gives $\hat{A} = \Sigma_{XY}\Sigma_Y^{-1}$.

B.1.3 Affine least-variance estimator

Suppose we want to compute the best estimator of a zero-mean *random vector* X as a *linear* map of the zero-mean random vector Y. We just have to repeat the construction reported in the previous section for each component X_i of X, so that the rows $\hat{A}_{i.}$ of the matrix $\hat{A}$ are given by

$$\begin{aligned} \hat{A}_{1.} &= \Sigma_{X_iY}\Sigma_Y^{-1} \\ \vdots &= \vdots \\ \hat{A}_{n.} &= \Sigma_{X_nY}\Sigma_Y^{-1} \end{aligned} \tag{B.17}$$

[2] The resemblance with a conditional expectation is due to the fact that, in the presence of Gaussian random vectors such a projection is indeed the conditional expectation.

which eventually gives us

$$\hat{A} = \Sigma_{XY}\Sigma_Y^{-1}. \tag{B.18}$$

If now the vectors X and Y are not zero-mean, $\mu_X \neq 0$, $\mu_Y \neq 0$, we first transform it into a zero-mean problem by defining $\bar{Y} \doteq Y - \mu_Y$, $\bar{X} \doteq X - \mu_X$, then solve for the linear least-variance estimator $\hat{A} = \Sigma_{\bar{X}\bar{Y}}\Sigma_{\bar{Y}}^{-1} \doteq \Sigma_{XY}\Sigma_Y^{-1}$, and then substitute to get

$$\hat{Z} = \mu_X + \Sigma_{XY}\Sigma_Y^{-1}(Y - \mu_Y) \tag{B.19}$$

which is the least-variance affine estimator

$$\hat{Z} \doteq \hat{E}[X|Y] = \hat{A}Y + \hat{b} \tag{B.20}$$

where

$$\hat{A} = \Sigma_{XY}\Sigma_Y^{-1} \tag{B.21}$$

$$\hat{b} = \mu_X - \Sigma_{XY}\Sigma_Y^{-1}\mu_Y. \tag{B.22}$$

It is an easy exercise to compute the variance of the estimation error $\tilde{X} \doteq X - \hat{Z}$:

$$\Sigma_{\tilde{X}} = \Sigma_X - \Sigma_{XY}\Sigma_Y^{-1}\Sigma_{YX}. \tag{B.23}$$

If we interpret the variance of X as the "prior uncertainty", and the variance ot $\tilde{X}$ as the "posterior uncertainty", we may interpret the second term (which is positive semi-definite) of the above equation as a "decrease" of the uncertainty.

B.1.4 Properties and interpretations of the least-variance estimator

The variance of the estimation error in equation (B.23) is by construction the smallest that can be achieved *with an affine estimator*. Of course if we consider a broader class $\mathcal{T}$ of estimators, the estimation error can be further decreased, unless the model that generates the data T is itself affine:

$$Y = T(X) = FX + W. \tag{B.24}$$

In such a case, using the matrix inversion lemma [3], it is easy to compute the expression of the optimal (affine) estimator that depends only upon Σ_X, Σ_W and F:

$$\hat{Z} = \Sigma_X F^T(F\Sigma_X F^T + \Sigma_W)^{-1}Y \tag{B.25}$$

which achieves a variance of the estimation error equal to

$$\Sigma_{\tilde{X}} = \Sigma_X - \Sigma_X F^T(F\Sigma_X F^T + \Sigma_W)^{-1}F\Sigma_X. \tag{B.26}$$

[3] If A, B, C, D are real matrices of the appropriate dimensions with A and C invertible, then $(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA$.

Projection onto an orthogonal sum of subspaces

Let $Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$ be such that

$$\mathcal{H}(Y) = \mathcal{H}(Y_1) \oplus \mathcal{H}(Y_2). \tag{B.27}$$

We may now wonder what are the conditions under which

$$\hat{E}[X|Y] = \hat{E}[X|Y_1] + \hat{E}[X|Y_2]. \tag{B.28}$$

After an easy calculation one can see that the above is true iff $E[Y_1 Y_2^T] = 0$, which is to say when

$$\mathcal{H}(Y_1) \perp \mathcal{H}(Y_2) \tag{B.29}$$

Change of basis

Suppose that instead of measuring the instances of a random vector Y we measure another random vector Z which is related to Y via a change of basis: $Z = TY \mid T \in GL(m)$. If we call $\hat{E}[X|Y] = \hat{A}Y$, then it is immediate to see that

$$\begin{aligned} \hat{E}[X|Z] &= \Sigma_{XZ}\Sigma_Z^{-1} Z \\ &= \Sigma_{XY} T^T (T^{T^{-1}} \Sigma_Y T^{-1}) Z \\ &= \Sigma_{XY}\Sigma_Y^{-1} T^{-1} Z. \end{aligned} \tag{B.30}$$

Innovations

The linear least-variance estimator involves the computation of the inverse of the output covariance matrix Σ_Y. It may be interesting to look for changes of bases T that transform the output Y into $Z = TY$ such that $\Sigma_Z = I$. In such a case the optimal estimator is simply

$$\hat{E}[X|Z] = \Sigma_{XZ} Z. \tag{B.31}$$

Let us pretend for a moment that the components of the vector Y are samples of a process taken over time: $Y_i = y(i)$, and call $y^t = [Y_1, \ldots, Y_t]^T$ the history of the process up to time t. Each component (sample) is an element of the Hilbert space $\mathcal{H}$, which has a well-defined notion of orthogonality, and where we can apply Gram-Schmidt procedure in order to make the "vectors" $y(i)$ orthogonal (uncorrelated).

$$\begin{array}{lcll} v_1 & \doteq & y(1) & \longrightarrow \quad e_1 \doteq v_1/\|v_1\| \\ v_2 & \doteq & y(2) - \langle y(2), e_1 \rangle e_1 & \longrightarrow \quad e_2 \doteq v_2/\|v_2\| \\ \vdots & \doteq & \vdots & \longrightarrow \qquad \vdots \\ v_t & \doteq & y(t) - \sum_{i=1}^{t-1} \langle y(i), e_i \rangle e_i & \longrightarrow \quad e_t \doteq v_t/\|v_t\| \end{array} .$$

The process $\{e\}$, whose instances up to time t are collected into the vector $e^t = [e_1, \ldots, e_t]^T$ has a number of important properties:

1. The component of e^t are *orthonormal* in $\mathcal{H}$ (or equivalently $\{e\}$ is an uncorrelated process). This holds by construction.

2. The transformation from y to e is *causal*, in the sense that – if we represent it as a matrix L_t such that

$$y^t = L_t e^t \tag{B.32}$$

then $L_t \in \mathcal{L}_+$ is lower-triangular with positive diagonal. This follows from the Gram-Schmidt procedure.

3. The process $\{e\}$ is *equivalent* to $\{y\}$ in the sense that they generate the same span

$$\mathcal{H}(y^t) = \mathcal{H}(e^t). \tag{B.33}$$

This property follows from the fact that L_t is non-singular.

4. If we write $y^t = L_t e^t$ in matrix form as $Y = LE$, then $\Sigma_Y = LL^T$.

The meaning of the components of v, and the name *innovation*, comes from the fact that we can interpret

$$v_t \doteq y(t) - \hat{E}[y(t)|y^{t-1}] \tag{B.34}$$

as a *one-step prediction error*. The process e is a scaled version of v such that its variance is the identity.

We may now wonder whether each process $\{y\}$ has an innovation, and if so, whether it is unique. The following theorem, which is known as *Cholesky factorization theorem* or *Spectral Factorization theorem* depending upon the context, states the conditions:

Theorem B.2. *There exists a unique vector E which is causally equivalent to Y iff there exists a unique lower-triangular matrix L, called Choleski's factor, such that $\Sigma_Y = LL^T$.*

Remark B.1. *The Choleski factor can be interpreted as a "whitening filter", in the sense that it acts on the components of the vector Y in a causal fashion to make them uncorrelated.*

We may consider a two-step solution to the problem of finding the least-square filter: a "whitening step"

$$E = L^{-1}Y \tag{B.35}$$

where $\Sigma_E = I$, and a projection onto $\mathcal{H}(E)$:

$$\hat{X}(Y) = \Sigma_{XE} L^{-1} Y. \tag{B.36}$$

B.2 Linear least-variance estimator for stationary processes

In the previous section we have interpreted a column-vector as a collection of samples from a scalar random process, and computed the least-variance estimator by orthogonal projection. In this section we see how this plot generalizes to proper *stationary* processes. We consider only scalar processes for simplicity of notation, although all considerations can be extended to vector-valued processes.

Let us assume that $\{x(t)\} \in \mathbb{R}^n$ and $\{y(t)\} \in \mathbb{R}^m$ are (wide-sense) jointly stationary, i.e.

$$\Sigma_{xy}(t,s) \doteq E[x(t)y^T(s)] = \Sigma_{xy}(t-s). \tag{B.37}$$

Again, we restrict our attention to *linear* estimators of $\{x(t)\}$ given the measurements of $\{y(s);\ s \leq t\}$ *up to time* t. We denote the estimate by $\hat{x}(t|t)$. A linear estimator is described by a convolution kernel h such that

$$\hat{x}(t|t) = \sum_{k=-\infty}^{t} h(t,k)y(k). \tag{B.38}$$

The design of the least-variance estimator involves finding the kernel $\hat{h}$ such that the estimation error $\tilde{x}(t) \doteq x(t) - \hat{x}(t|t)$ has minimum variance. This is found, as in the previous sections for the static case, by imposing that the estimation error be orthogonal to the history of the process $\{y\}$ up to time t:

$$\begin{aligned} &\langle x(t) - \hat{x}(t|t), y(s)\rangle_{\mathcal{H}} = 0, \quad \forall\ s \leq t \\ &E[x(t)y^T(s)] - \sum_{k=-\infty}^{t} h(t,k)E[y(k)y^T(s)] = 0, \quad \forall\ s \leq t \end{aligned} \tag{B.39}$$

which is equivalent to

$$\Sigma_{xy}(t-s) = \sum_{k=-\infty}^{t} h(t,k)\Sigma_y(k-s). \tag{B.40}$$

The above is equivalent to a linear system with an infinite number of equations, and we will assume that it has a unique solution for H. Given that the processes involved are (jointly) stationary, and the convolution starts at $-\infty$, it can be easily seen that the kernel h is *time invariant*: $h(t,k) = h(t-k)$. Therefore the last equation is equivalent to

$$\Sigma_{xy}(t) = \sum_{s=0}^{\infty} h(s)\Sigma_y(t-s), \quad \forall t \geq 0 \tag{B.41}$$

which is called *Wiener-Hopf equation* and is exactly equivalent to the orthogonality conditions (B.16). In fact, if we $\mathcal{Z}$-transform the above

equation

$$S_{xy}(z) = H(z)S_y(z) \tag{B.42}$$

we have exactly the same expression as equation (B.16), which we could try to solve as

$$\hat{H}(z) = S_{xy}(z)S_y^{-1}(z) \tag{B.43}$$

provided that the spectral density S_y is invertible. This, however, is not quite the solution we are looking for. In fact, in order to be of any use, the estimator must be *causal* (it must not depend upon "future" samples of the process $\{y\}$) and *stable* (it must return a bounded estimate for bounded data). We can express these conditions by requiring

- causality: $h(t) = 0$, $\forall t < 0$ (or $H(z)$ analytic at ∞)
- stability: $H(z)$ analytic in $|z| \geq 1$ (or $h(t)$ square-summable).

One particular case is when the spectral density of $\{y\}$ is the identity (or equivalently $\{y\}$ is a white noise). Then $S_y = I$ and we could choose

$$h(t) = \begin{cases} \Sigma_{xy}(t), & t \geq 0 \\ 0, & t < 0. \end{cases} \tag{B.44}$$

This suggests us to try to *whiten* (or orthonormalize) the measurement process $\{y\}$ in a similar fashion to what we did in section B.1.4. Indeed we can state a theorem similar to B.2, which is known as the *spectral factorization theorem*:

Theorem B.3. *There exists a process $\{\tilde{e}\}$ such that $\mathcal{H}(\tilde{e}^t) = \mathcal{H}(y^t)$ and $\Sigma_{\tilde{e}}(t) = \Lambda\delta(t)$ iff there exists $W(z)$ stable and causal, with $W^{-1}(z)$ causal such that $S_y(z) = W(z)W(z^{-1})$.*

Remark B.2. *In words there exists a white process $\{\tilde{e}\}$ (called the* innovation*) which is causally equivalent to $\{y\}$ iff the spectral density of y has a causal, stable and minimum-phase spectral factor. If we re-scale $W(z)$ to $L(z) = W(z)W(\infty)^{-1}$, the innovation $\{e\}$ is re-normalized so that $\Sigma_e(t) = I\delta(t)$, and is called* normalized innovation*.*

We may at this point repeat the two-step construction of the least-variance estimator. First the "whitening step":

$$E(z) = L^{-1}(z)Y(z) \tag{B.45}$$

and then the *causal* part of the projection:

$$\hat{X}(Y) = \begin{cases} \Sigma_{xe}(t) * e(t), & t \geq 0 \\ 0, & t < 0 \end{cases} \tag{B.46}$$

where $*$ indicates the standard convolution. Equivalently, if we denote by $[S_{xe}(z)]_+$ the causal part of the $\mathcal{Z}$-transform of $\Sigma_{xe}(t)$, we can write

$$\hat{X}(Y)_{(z)} = [S_{xe}(z)]_+ E(z). \tag{B.47}$$

Since $S_{xe}(z) = S_{xy}(z)L(z^{-1})^{-1}$, the final expression of our linear, least-variance estimator is (in the $\mathcal{Z}$-domain) $\hat{x} = \hat{H}(z)y(z)$, where the kernel H is given by

$$\hat{H}(z) = \left[S_{xy}(z)L^{-1}(z^{-1})\right]_+ L^{-1}(z). \tag{B.48}$$

The corresponding filter is known as the *Wiener filter*. Again we can recover the meaning of the innovation as the one-step prediction error for the measurements: in fact, the best prediction of the process $\{y\}$, indicated with $\hat{y}(t|t-1)$, is defined as the projection of $y(t)$ onto the span of $\{y\}$ up to $t-1$, indicated with $\mathcal{H}_{t-1}(y)$. Such projection is therefore defined such that

$$y(t) = \hat{y}(t|t-1) + e(t) \tag{B.49}$$

where $e(t) \perp \mathcal{H}_{t-1}(y) \;=\; \mathcal{H}_{t-1}(e)$.

B.3 Linear, finite-dimensional stochastic processes

A linear, finite-dimensional stochastic process (LFDSP) is defined as the output of a linear, finite-dimensional dynamical system driven by white Gaussian noise. Let $A(t)$, $B(t)$, $C(t)$, $D(t)$ be time-varying matrices of suitable dimensions, $\{n(t)\} \in \mathcal{N}(0, I) \mid E[n(t)n^T(s)] = I\delta(t-s)$ a white, zero-mean Gaussian noise and x_0 a random vector which is uncorrelated with $\{n\}$: $E[x_0 n^T(t)] = 0, \;\forall\; t$. Then $\{y(t)\}$ is a LFDSP if there exists $\{x(t)\}$ such that

$$\begin{cases} x(t+1) = A(t)x(t) + B(t)n(t) & x(t_0) = x_0 \\ y(t) = C(t)x(t) + D(t)n(t) \end{cases} \tag{B.50}$$

We call $\{x\}$ the *state process*, $\{y\}$ the *output (or measurement) process*, and $\{n\}$ the *input (or driving) noise*. The time-evolution of the state process can be written as the orthogonal sum of the past history (prior to the initial condition), and the present history (from the initial condition until the present time)

$$x(t) = \Phi_{t_0}^t x_0 + \sum_{k=t_0}^{t-1} \Phi_{k+1}^t B(t)n(t) = \hat{E}[x(t)|\mathcal{H}(x^{t_0})] + \hat{E}[x(t)|x(t_0), \ldots, x(t-1)] \tag{B.51}$$

where Φ denotes a fundamental set of solutions, which is the flow of the differential equation

$$\begin{cases} \Phi(t+1, s) = A(t)\Phi(t, s) \\ \Phi(t, t) = I. \end{cases} \tag{B.52}$$

In the case of a time-invariant system $A(t) = A, \;\forall t$, then $\Phi(t, s) = A^{t-s}$.

Remark B.3. *As a consequence of the definitions, the orthogonality between the state and the input noise propagates up to the current time:*

$$n(t) \perp_{\mathcal{H}} x(s), \quad \forall s \leq t. \tag{B.53}$$

Moreover, the past history up to time s is alway summarized by the value of the state at that time (Markov property):

$$\hat{E}[x(t)|\mathcal{H}_s(x)] = \hat{E}[x(t)|x(s)] = \Phi(t,s)x(s), \quad \forall t \geq s. \tag{B.54}$$

B.4 Stationariety of LFDSP

In order to design the least-squares estimator as in the previous sections, we ask what are the conditions under which a LFDSP is stationary. The first restriction we require is that the system be time-invariant. The mean of the state process at time t is given by

$$\mu_x(t) = A^{t-t_0}\mu_{x_0} \tag{B.55}$$

while the covariance of the state-process

$$\Sigma_x(t,s) = A^{t-s}\Sigma_x(s) \tag{B.56}$$

evolves according to the following *Ljapunov equation*

$$\Sigma_x(s+1) = A\Sigma_x(s)A^T + BB^T. \tag{B.57}$$

The conditions for stationariety impose that $\sigma_x(t) = \text{const}$ and $\mu_x(t) = \text{const}$. It is easy to prove the following

Theorem B.4. *Let A be stable (have all eigenvalues in the unit complex circle), then $\Sigma_x(t-t_0) \to \bar{\Sigma}$, where $\bar{\Sigma} = \sum_{k=0}^{\infty} A^k BB^T A^{T^k}$ is the unique equilibrium solution of the above Ljapunov equation, and $\{x\}$ describes asymptotically a stationary process. If x_0 is such that $\Sigma_x(t_0) = \bar{\Sigma}$, then the process is stationary for all $t \geq t_0$.*

Remark B.4. *The condition of stability for A is sufficient, but not necessary for generating a stationary process. If, however, the pair (A,B) is completely controllable, so that the noise input affects all of the components of the state, then such a stability condition becomes also necessary.*

B.5 The linear Kalman filter

Suppose we are given a linear finite-dimensional process, which has a realization (A,B,C,D) as in equation (B.50). While we measure the (noisy) output $y(t)$ of such a realization, we do not have access to its state $x(t)$. The Kalman filter is a dynamical model that accepts as input the output of the process realization, and returns an estimate of its state that has the

property of having the least error variance. In order to derive the expression for the filter, we write the LFDSP as follows:

$$\begin{cases} x(t+1) = Ax(t) + v(t) & x(t_0) = x_0 \\ y(t) = Cx(t) + w(t) \end{cases} \tag{B.58}$$

where we have neglected the time argument in the matrices $A(t)$ and $C(t)$ (all considerations can be carried through for time-varying systems as well). $v(t) = Bn(t)$ is a white, zero-mean Gaussian noise with variance Q, $w(t) = Dn(t)$, also a white, zero-mean noise, has variance R, so that we could write

$$\begin{aligned} v(t) &= \sqrt{Q}n(t) \\ w(t) &= \sqrt{R}n(t) \end{aligned}$$

where n is a unit-variance noise. In general v and w will be correlated, and in particular we will call

$$S(t) = E[v(t)w^T(t)]. \tag{B.59}$$

We require that the initial condition x_0 be uncorrelated from the noise processes:

$$x_0 \perp \{v\}, \{w\}, \quad \forall\, t \tag{B.60}$$

The first step is to modify the above model so that the model error v is uncorrelated from the measurement error w.

Uncorrelating the model from the measurements

In order to uncorrelate the model error from the measurement error we can just substitute v with the complement of its projection onto the span of w. Let us call

$$\tilde{v}(t) = v(t) - \hat{E}[v(t)|H(w)] = v(t) - \hat{E}[v(t)|w(t)] \tag{B.61}$$

the last equivalence is due to the fact that w is a white noise. We can now use the results from section B.1 to conclude that

$$\tilde{v}(t) = v(t) - SR^{-1}w(t) \tag{B.62}$$

and similarly for the variance matrix

$$\tilde{Q} = Q - SR^{-1}S^T. \tag{B.63}$$

Substituting the expression of $v(t)$ into the model (B.58) we get

$$\begin{cases} x(t+1) = Fx(t) + SR^{-1}y(t) + \tilde{v} \\ y(t) = Cx(t) + w(t) \end{cases} \tag{B.64}$$

where $F = A - SR^{-1}C$. The model error $\tilde{v}$ in the above model is uncorrelated from the measurement noise w, and the cost is that we had to add an output-injection term $SR^{-1}y(t)$.

Prediction step

Suppose at some point in time we are given a current estimate for the state $\hat{x}(t|t)$ and a corresponding estimate of the variance of the model error $P(t|t) = E[\tilde{x}(t)\tilde{x}(t)^T]$ where $\tilde{x} = x - \hat{x}$. At the initial time t_0 we can take $\hat{x}(t_0|t_0) = x_0$ with some bona-fide variance matrix. Then it is immediate to compute

$$\hat{x}(t+1|t) = F\hat{x}(t|t) + SR^{-1}y(t) + \hat{E}[\tilde{v}(t)|H_t(y)] \tag{B.65}$$

where the last term is zero since $\tilde{v}(t) \perp x(s),\ \forall x \leq t$ and $\tilde{v}(t) \perp w(s),\ \forall s$ and therefore $\tilde{v}(t) \perp y(s),\ \forall s \leq t$. The estimation error is therefore

$$\tilde{x}(t+1|t) = F\tilde{x}(t|t) + \tilde{v}(t) \tag{B.66}$$

where the sum is an orthogonal sum, and therefore it is trivial to compute the variance as

$$P(t+1|t) = FP(t|t)F^T + \tilde{Q}. \tag{B.67}$$

Update step

Once a new measurement is acquired, we can update our prediction so as to take into account the new measurement. The update is defined as $\hat{x}(t+1|t+1) \doteq \hat{E}[x(t+1)|H_{t+1}(y)]$. Now, as we have seen in section B.1.4, we can decompose the span of the measurements into the orthogonal sum

$$H_{t+1}(y) = H_t(y) + \{e(t+1)\} \tag{B.68}$$

where $e(t+1) \doteq y(t+1) - \hat{E}[y(t+1)|H_t(y)]$ is the innovation process. Therefore, we have

$$\hat{x}(t+1|t+1) = \hat{E}[x(t+1)|H_t(y)] + \hat{E}[x(t+1)|e(t+1)] \tag{B.69}$$

where the last term can be computed using the results from section B.1:

$$\hat{x}(t+1|t+1) = \hat{x}(t+1|t) + L(t+1)e(t+1) \tag{B.70}$$

where $L(t+1) \doteq \Sigma_{xe}(t+1)\Sigma_e^{-1}(t+1)$ is called the *Kalman gain*. Substituting the expression for the innovation we have

$$\hat{x}(t+1|t+1) = \hat{x}(t+1|t) + L(t+1)\left(y(t+1) - C\hat{x}(t+1|t)\right) \tag{B.71}$$

from which we see that the update consists in a linear correction weighted by the Kalman gain.

Computation of the gain

In order to compute the gain $L(t+1) \doteq \Sigma_{xe}(t+1)\Sigma_e^{-1}(t+1)$ we derive an alternative expression for the innovation:

$$e(t+1) = y(t+1) - Cx(t+1) + Cx(t+1) - C\hat{x}(t+1|t) = w(t+1) + C\tilde{x}(t+1|t) \tag{B.72}$$

from which it is immediate to compute

$$\Sigma_{xe}(t+1) = P(t+1|t)C^T. \tag{B.73}$$

Similarly we can derive the variance of the innovation $\Lambda(t+1)$:

$$\Lambda(t+1) \doteq \Sigma_e(t+1) = CP(t+1|t)C^T + R \tag{B.74}$$

and therefore the Kalman gain is

$$L(t+1) = P(t+1|t)C^T\Lambda^{-1}(t+1). \tag{B.75}$$

Variance update

From the update of the estimation error

$$\tilde{x}(t+1|t+1) = \tilde{x}(t+1|t) - L(t+1)e(t+1) \tag{B.76}$$

we can easily compute the update for the variance. We first observe that $\tilde{x}(t+1|t+1)$ is by definition orthogonal to $H_{t+1}(y)$, while the correction term $L(t+1)e(t+1)$ is contained in the history of the innovation, which is by construction equal to the history of the process y: $H_{t+1}(y)$. Then it is immediate to see that

$$P(t+1|t+1) = P(t+1|t) - L(t+1)\Lambda(t+1)L^T(t+1). \tag{B.77}$$

The above equation is not convenient for computational purposes, since it does not guarantee that the updated variance is a symmetric matrix. An alternative form of the above that does guarantee symmetry of the result is

$$P(t+1|t) = \Gamma(t+1)P(t+1|t)\Gamma(t+1)^T + L(t+1)RL(t+1)^T \tag{B.78}$$

where $\Gamma(t+1) = I - L(t+1)C$. The last equation is in the form of a discrete Riccati equation (DRE).

Predictor equations

It is possible to combine the two steps above and derive a single model for the one-step predictor. We summarize the result as follows:

$$\begin{aligned} \hat{x}(t+1|t) &= A\hat{x}(t|t-1) + K_s(t)\left(y(t) - C\hat{x}(t|t-1)\right) && \text{(B.79)} \\ P(t+1|t) &= F\Gamma(t)P(t|t-1)\Gamma(t)^T F^T + FL(t)RL^T(t)F^T + Q && \text{(B.80)} \end{aligned}$$

where we have defined

$$\begin{aligned} K_s(t) &\doteq FL(t) + SR^{-1} && \text{(B.81)} \\ &= \left(AP(t|t-1)C^T + S\right)\Lambda^{-1}(t) && \text{(B.82)} \end{aligned}$$

B.6 Asymptotic properties

If we consider time-invariant models (all matrices A, C, Q, S, R are constant in time), we can study the asymptotic behavior of the estimator.

Remark B.5. *In particular, the dynamics of the estimator depends upon $P(t)$, the solution of the DRE of equation (B.78). We want such a solution to converge asymptotically to a small* but non-zero *value. In fact, $P = 0$ corresponds to a zero gain $K = 0$, which indicates that the filter does take into account the measurements. In such a case we say that the filter is* saturated.

We will not get into the details of the results of the asymptotic theory of Kalman filtering. We will only report the main results, which essentially says that if the realization of the LFDSP is minimal, then there exists a unique positive-definite fixed-point of the DRE, and the solution converges to the fixed-point asymptotically. Furthermore the dynamics of the estimation error is stable (even though the process may be unstable). The Kalman filter converges asymptotically to the Wiener filter described in section B.2.

Claim B.1. *If the pair (F, C) is detectable and $(F, \sqrt{Q})$ is stabilizable, then there exists a unique $P \,|P = P^T \geq 0$ fixed point of the DRE (B.78). Furthermore $P(t) \to P$ for all positive semi-definite $P(t_0)$ and $\Gamma = lim_{t\to\infty}\Gamma(t)$ is stable.*

We recall that a (F, C) being detectable means that the unobservable subspace is stable, as well as $(F, \sqrt{Q})$ being stabilizable means that the uncontrollable subspace is stable. The proof of the above claim, as well as other results on the asymptotic properties of the Kalman filter, can be found for instance in [Jaz70].

Appendix C

Basic facts from optimization

References

[Åst96] K. Åström. *Invariancy Methods for Points, Curves and Surfaces in Computational Vision*. PhD thesis, Department of Mathematics, Lund University, 1996.

[BA83] P. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communication*, 31:532–540, 1983.

[BCB97] M. J. Brooks, W. Chojnacki, and L. Baumela. Determining the ego-motion of an uncalibrated camera from instantaneous optical flow. *in press*, 1997.

[Bou98] S. Bougnoux. From projective to Euclidean space under any practical situation, a criticism of self-calibration. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pages 790–796, 1998.

[CD91] F. M. Callier and C. A. Desoer. *Linear System Theory*. Springer Texts in Electrical Engineering. Springer-Verlag, 1991.

[Fau93] O. Faugeras. *There-Dimensional Computer Vision*. The MIT Press, 1993.

[GH86] Guckenheimer and Holmes. *Nonlinear oscillations, dynamical systems and bifurcations of vector fields*. Springer Verlag, 1986.

[Har98] R. I. Hartley. Chirality. *International Journal of Computer Vision*, 26(1):41–61, 1998.

[HF89] T. Huang and O. Faugeras. Some properties of the E matrix in two-view motion estimation. *IEEE PAMI*, 11(12):1310–12, 1989.

[HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Conference*, pages 189–192, 1988.

[HZ00] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge, 2000.

[Jaz70] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. NY: Academic Press, 1970.

[Kan93] K. Kanatani. *Geometric Computation for Machine Vision*. Oxford Science Publications, 1993.

[LF97] Q.-T. Luong and O. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *IJCV*, 22(3):261–89, 1997.

[LH81] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[LK81] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[May93] S. Maybank. *Theory of Reconstruction from Image Motion*. Springer Series in Information Sciences. Springer-Verlag, 1993.

[McL99] P. F. McLauchlan. Gauge invariance in projective 3d reconstruction. In *IEEE Workshop on Multi-View Modeling and Analysis of Visual Scenes, Fort Collins, CO, June 1999*, 1999.

[MF92] S. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[MKS00] Y. Ma, J. Kovsecká, and S. Sastry. Linear differential algorithm for motion recovery: A geometric approach. *International Journal of Computer Vision*, 36(1):71–89, 2000.

[MLS94] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC press Inc., 1994.

[MS98] Y. Ma and S. Sastry. Vision theory in spaces of constant curvature. *Electronic Research Laboratory Memorandum, UC Berkeley*, UCB/ERL(M98/36), June 1998.

[PG98] J. Ponce and Y. Genc. Epipolar geometry and linear subspace methods: a new approach to weak calibration. *International Journal of Computer Vision*, 28(3):223–43, 1998.

[PG99] M. Pollefeys and L. Van Gool. Stratified self-calibration with the modulus constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):707–24, 1999.

[SA90] M. Spetsakis and Y. Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–184, 1990.

[Stu97] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, pages 1100–1105. IEEE Comput. Soc. Press, 1997.

[TK92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography. *Intl. Journal of Computer Vision*, 9(2):137–154, 1992.

[Tri97] B. Triggs. Autocalibration and the absolute quadric. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, 1997.

[TTH96] T. Y. Tian, C. Tomasi, and D. J. Heeger. Comparison of approaches to egomotion computation. In *Proceedings of 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 315–20, Los Alamitos, CA, USA, 1996. IEEE Comput. Soc. Press.

[VF95] T. Vieville and O. D. Faugeras. Motion analysis with a camera with unknown, and possibly varying intrinsic parameters. *Proceedings of Fifth International Conference on Computer Vision*, pages 750–756, June 1995.

[VMHS01] R. Vidal, Y. Ma, S. Hsu, and S. Sastry. Optimal motion estimation from multiview normalized epipolar constraint. In *ICCV'01*, Vancouver, Canada, 2001.

[WHA93] J. Weng, T. S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences*. Springer Verlag, 1993.

[ZDFL95] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119, 1995.

[ZF96] C. Zeller and O. Faugeras. Camera self-calibration from video sequences: the Kruppa equations revisited. *Research Report 2793, INRIA, France*, 1996.

[ZH84] X. Zhuang and R. M. Haralick. Rigid body motion and optical flow image. *Proceedings of the First International Conference on Artificial Intelligence Applications*, pages 366–375, 1984.

[Zha98] Z. Zhang. A flexible new technique for camera calibration. *Microsoft Technical Report MSR-TR-98-71*, 1998.

Glossary of notations

(R, T)	$R \in SO(3)$ the rotation; $T \in \mathbb{R}^3$ the translation
(ω, v)	$\omega \in \mathbb{R}^3$ the angular velocity; $v \in \mathbb{R}^3$ the linear velocity
$E \doteq \widehat{T}R$	the essential matrix
$L \in \mathbb{E}^3$	a generic (abstract) line in the Euclidean space
$\mathbb{E}^3$	three-dimensional Euclidean space
$\mathbf{X} \doteq [X, Y, Z, W]^T \in \mathbb{R}^4$	homogeneous coordinates of a point in $\mathbb{E}^3$
$\mathbf{X}^i$	coordinates of the i^{th} point
$\mathbf{X}^i_j$	coordinates of the i^{th} point with respect to the j^{th} (camera) coordinate frame, short version for $\mathbf{X}^i(t_j)$
$\mathbf{l} \doteq [a, b, c]^T$	homogeneous coordinates of the coimage of a line
$\mathbf{x} \doteq [x, y, z]^T \in \mathbb{R}^3$	homogeneous coordinates of the image of a point
$\mathbf{x}^i$	coordinates of the i^{th} image of a point
$\mathbf{x}^i_j$	coordinates of the i^{th} image with respect to the j^{th} (camera) coordinate frame, short version for $\mathbf{x}^i(t_j)$
$g \in SE(3)$	special Euclidean motion

$p \in \mathbb{E}^3$	a generic (abstract) point in the Euclidean space

Index