UNIVERSITÀ DEGLI STUDI DI TRIESTE

Dottorato di Ricerca in Ingegneria dell'Informazione

XI Ciclo

THREE-DIMENSIONAL VISION FOR STRUCTURE AND MOTION ESTIMATION

Andrea Fusiello

November 1998

Supervisor:

Vito Roberto, University of Udine, IT

External Supervisor:

Emanuele Trucco, Heriot-Watt University, UK

Co-ordinator:

Giuseppe Longo, University of Trieste, IT

Andrea Fusiello
Dip. di Matematica e Informatica
Università di Udine
Via delle Scienze 206
I-33100 Udine, Italy
e-mail: fusiello@dimi.uniud.it
http://www.dimi.uniud.it/~fusiello

This thesis was composed at the University of Udine using LaTeX, a typesetting system that employs TeX as its formatting engine. The mathematics is set in a new typeface called AMS Euler, designed by Hermann Zapf for the American Mathematical Society. The text is set in Computer Modern, the standard LaTeX typeface, designed by D.E. Knuth.

Abstract

This thesis addresses *computer vision techniques estimating geometric properties of the 3-D world from digital images.* Such properties are essential for object recognition and classification, mobile robots navigation, reverse engineering and synthesis of virtual environments.

In particular, this thesis describes the modules involved in the computation of the structure of a scene given some images, and offers original contributions in the following fields.

Stereo pairs rectification. A novel rectification algorithm is presented, which transform a stereo pair in such a way that corresponding points in the two images lie on horizontal lines with the same index. Experimental tests prove the correct behavior of the method, as well as the negligible decrease of the accuracy of 3-D reconstruction if performed from the rectified images directly.

Stereo matching. The problem of computational stereopsis is analyzed, and a new, efficient stereo matching algorithm addressing robust disparity estimation in the presence of occlusions is presented. The algorithm, called SMW, is an adaptive, multi-window scheme using left-right consistency to compute disparity and its associated uncertainty. Experiments with both synthetic and real stereo pairs show how SMW improves on closely related techniques for both accuracy and efficiency.

Features tracking. The Shi-Tomasi-Kanade feature tracker is improved by introducing an *automatic* scheme for rejecting spurious features, based on robust outlier diagnostics. Experiments with real and synthetic images confirm the improvement over the original tracker, both qualitatively and quantitatively.

Uncalibrated vision. A review on techniques for computing a three-dimensional model of a scene from a single moving camera, with unconstrained motion and unknown parameters is presented. The contribution is to give a critical, unified view of some of the most promising techniques. Such review does not yet exist in the literature.

3-D motion. A robust algorithm for registering and finding correspondences in two sets of 3-D points with significant percentages of missing data is proposed. The method, called RICP, exploits LMedS robust estimation to withstand the effect of outliers. Experimental comparison with a closely related technique, ICP, shows RICP's superior robustness and reliability.

Riassunto

Questa tesi, intitolata **Visione Tridimensionale per la Stima di Struttura e Moto**, tratta di tecniche di Visione Artificiale per la stima delle proprietà geometriche del mondo tridimensionale a partire da immagini numeriche. Queste proprietà sono essenziali per il riconoscimento e la classificazione di oggetti, la navigazione di veicoli mobili autonomi, il *reverse engineering* e la sintesi di ambienti virtuali.
In particolare, saranno descritti i moduli coinvolti nel calcolo della struttura della scena a partire dalle immagini, e verranno presentati contributi originali nei seguenti campi.

Rettificazione di immagini steroscopiche. Viene presentato un nuovo algoritmo per la rettificazione, il quale trasforma una coppia di immagini stereoscopiche in maniera che punti corrispondenti giacciano su linee orizzontali con lo stesso indice. Prove sperimentali dimostrano il corretto comportamento del metodo, come pure la trascurabile perdita di accuratezza nella ricostruzione tridimensionale quando questa sia ottenuta direttamente dalle immagini rettificate.

Calcolo delle corrispondenze in immagini stereoscopiche. Viene analizzato il problema della stereovisione e viene presentato un un nuovo ed efficiente algoritmo per l'identificazione di coppie di punti corrispondenti, capace di calcolare in modo robusto la disparità stereoscopica anche in presenza di occlusioni. L'algoritmo, chiamato SMW, usa uno schema multi-finestra adattativo assieme al controllo di coerenza destra-sinistra per calcolare la disparità e l'incertezza associata. Gli esperimenti condotti con immagini sintetiche e reali mostrano che SMW sortisce un miglioramento in accuratezza ed efficienza rispetto a metodi simili .

Inseguimento di punti salienti. L'inseguitore di punti salienti di Shi-Tomasi-Kanade viene migliorato introducendo uno schema automatico per lo scarto di punti spuri basato sulla diagnostica robusta dei campioni periferici (*outliers*). Gli esperimenti con immagini sintetiche e reali confermano il miglioramento rispetto al metodo originale, sia qualitativamente che quantitativamente.

Ricostruzione non calibrata. Viene presentata una rassegna ragionata dei metodi per la ricostruzione di un modello tridimensionale della scena, a partire da una telecamera che si muove liberamente e di cui non sono noti i parametri interni. Il contributo consiste nel fornire una visione critica e unificata delle più recenti tecniche. Una tale rassegna non esiste ancora in letterarura.

Moto tridimensionale. Viene proposto un algoritmo robusto per registrate e calcolare le corrispondenze in due insiemi di punti tridimensionali nei quali vi sia un numero significativo di elementi mancanti. Il metodo, chiamato RICP, sfrutta la stima robusta con la Minima Mediana dei Quadrati per eliminare l'effetto dei campioni periferici. Il confronto sperimentale con una tecnica simile, ICP, mostra la superiore robustezza e affidabilità di RICP.

Ringraziamenti

Devo a Bruno Caprile se ho deciso di intraprendere l'impervia strada della ricerca: egli è stato per me un maestro, mi ha insegnata ad amarla e mi ha dimostrato, con l'esempio, cosa fosse l'integrità scientifica e morale.

Per quanto riguarda questa tesi, il ringraziamento più grande va ad Emanuele Trucco, che mi ha costantemente seguito, incoraggiato, sorretto e guidato, sia dal punto di vista scientifico, che da quello umano.

Vincenzo Isaia ed Emanuele Trucco hanno letto le bozze della tesi, ed hanno contribuito con preziosi commenti.

Ringrazio Agostino Dovier, Marino Miculan, Stefano Mizzaro e Vittorio Murino per l'amicizia e per l'aiuto che mi hanno prestato in varie occasioni.

Tutti coloro che mi sono vicini hanno dovuto, in ragione della loro vicinanza, sopportare i momenti di cattivo umore, gli sfoghi e le assenze, sia fisiche che spirituali. A loro, e segnatamente a Paola, Vincenzo ed Egle, chiedo scusa e dedico questa tesi.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Among all sensing capabilities, vision has long been recognized as the one with the highest potential. Many biological systems use it as their most powerful way of gathering information about the environment, and relatively cheap and high-quality visual sensors can be connected to computers easily and reliably.

The achievements of biological visual systems are formidable: they record a band of electromagnetic radiation and use it to gain knowledge about surrounding objects that emit and reflect it. The effort to replicate biological vision exactly is maybe pointless; on the other hand, "airplanes do not have feathers". However, trying to emulate *some* of its functions is a practicable but challenging task [28, 33].

The processes involved in visual perception are usually separated into low-level and high-level [152]. *Low-level vision* is associated with the extraction of certain physical properties of the environment, such as depth, 3-D shape, object boundaries. They are typically spatially uniform and relatively independent of the task at hand, or of the knowledge associated with specific objects. *High-level vision*, in contrast, is concerned with problems such as the extraction of shape properties and spatial relations, and with object recognition and classification. High-level vision processes are usually applied to selected portions of the image, and depend on the goal of the computation and the knowledge related to specific objects.

Low-level Computer Vision can be thought of as inverse Computer Graphics [125, 40]. Computer Graphics is the generation of images by computer starting from abstract descriptions of a scene and a knowledge of the laws of image formation. Low-level Computer Vision is the process of obtaining descriptions of objects from images and a knowledge of the laws of image formation. Yet, graphics is a feed-forward process, a many-to-one activity, whereas (low level) Computer Vision is an inverse problem [115], involving a one-to-many mapping. When a scene is observed, a 3-D environment is compressed into a 2-D image, and a considerable amount of information is lost.

1.1 Scope and motivations

Computer Vision is therefore a very demanding engineering challenge, that involves many interacting components for the analysis of color, depth, motion, shape and texture of objects, and the use of visual information for recognition, navigation and manipulation. I will deal in this thesis with some of these aspects only, the scope of this thesis being the low-level processes related to the *extraction of geometric properties of the 3-D world from digital images.* The most important property is *shape*, being the dominant cue used by high-level vision processes (such as object recognition and classification) [152]. Moreover, 3-D geometric properties are essential for tasks such as mobile robots navigation, reverse engineering, and synthesis of virtual environments.

1.2 Synopsis

This thesis presents techniques for extracting 3-D descriptions of a scene from images. Depending on the information available about the acquisition process, different techniques are applicable. I will start from those assuming the maximum amount of knowledge possible, and move on to techniques relaxing this assumption to increasing degrees.

I endeavored to make this dissertation self-contained. Hence Chapter 2 is devoted to introducing the fundamental laws of image formation. An image is the projection of the 3-D space onto a 2-D array, and it contains two types of visual cues: *geometric* and *radiometric.* The former are related to the position of image points, the latter to their brightness. In this work I will deal mainly with the geometric aspect of Computer Vision, and to this purpose the geometric camera model will be described in detail.

In Chapters 3 and 4 I will address the *structure from stereo* problem: given two pictures of a scene taken with a *calibrated* rig of two cameras, and a set of matched points, which are all images of points located in the scene, reconstruct the 3-D coordinates of the points.

In Chapter 3 I will discuss the geometric issues of structure from stereo. First, I will describe a simple, linear *calibration* algorithm, that is, a procedure for measuring the camera's *extrinsic parameters* (i.e., its position and pose) and its *intrinsic parameters* (i.e., its internal characteristics). In photogrammetry, camera calibration is divided into the *exterior orientation problem* and the *interior orientation problem.* Second, a linear *triangulation* technique will be described, which allows one to actually reconstruct the 3-D coordinates of the points. Then, I will concentrate on the *epipolar geometry*, i.e., the relationship between corresponding points in the two images, and in particular on *rectification*, an operation meant to obtain a simple epipolar geometry for any calibrated stereo pair. The main original contribution of this chapter is to introduce a linear rectification algorithm for general, unconstrained stereo rigs.

In Chapter 4 I will address the problem of matching points, that is detecting pairs of points in the two images that are projection of the same points in the scene, in order to produce disparity maps, which are directly connected to 3-D positions in space. I propose a novel stereo matching algorithm, called SMW (Symmetric Multi-Window) addressing robust disparity estimation in the presence of occlusions.

In Chapter 5 and 6 and I will address the *structure from motion* problem: given several views of a scene taken with a moving camera with known intrinsic parameters, and given a set of matched points, recover camera's motion and scene structure. Compared to the structure from stereo problem, here we have a single moving camera instead of a calibrated rig of two cameras, and the extrinsic parameters (i.e., the relative camera displacements) are missing. The output reconstruction differs from the true (or absolute) reconstruction by a similarity transformation, composed by a rigid displacement (due to the arbitrary choice of the world reference frame) plus a a uniform change of scale (due to depth-speed ambiguity). This is called a *Euclidean* reconstruction.

Chapter 5 is devoted to study the problem of estimating the motion of the cameras, assuming that correspondences between points in consecutive frames are given. This is known in photogrammetry as the *relative orientation problem.*

In Chapter 6 I will address the problem of computing correspondences by tracking point features in image sequences. The main original contribution of this chapter is to extend existing tracking techniques by introducing a robust scheme for rejecting spurious features. This is done by employing a simple and efficient outlier rejection rule, called X84.

In Chapter 7 another bit of a-priori information is removed, and the intrinsic parameters are assumed unknown: the only information that can be exploited is contained in the video sequence itself. Starting from two-view correspondences only, one can still compute a *projective* reconstruction of the scene points, that differ from the true one (Euclidean) by an unknown projective transformation. Assuming that the unknown intrinsic parameters are *constant*, the rigidity of camera motion can be used to recover the intrinsic parameters, hence falling back to the case of structure from motion again. This process is called *autocalibration.* Very recently, new methods have been proposed which directly upgrade the *projective* structure to the Euclidean structure, by exploiting all the available constraints. This is the idea of *stratification.* The contribution of this chapter is to give a unified view of some of the most promising techniques. Such unified, comparative discussion has not yet been presented in the literature.

Finally, Chapter 8 addresses the *3-D motion problem*, where the points correspondences and the motion parameters between two sets of 3-D points are to be recovered. This is used to register 3-D measures obtained with different algorithms for structure recovery or different depth measuring devices, related by an unknown rigid transformation. The existence of missing points in the two sets makes the problem difficult. The contribution here is a robust algorithm, RICP, based on Least Median of Squares regression, for registering and finding correspondences in sets of 3-D points with significant percentages of missing data.

Figure 1 represents the layout of this thesis at a glance. The process described is *image in – structure out.* Depending on the amount of information available, the output structure is related in a different way with the true (absolute) structure. Each rectangle represent a module, that will be described in the section or chapter reported close to it. In summary, the modules are:

- calibration (exterior and interior orientation) (Section 3.2) ;
- triangulation (Section 3.3);
- rectification (Section 3.5);
- stereo matching (Chapter 4);
- motion analysis (relative orientation) (Chapter 5);
- feature tracking (Chapter 6);
- projective reconstruction (Section 7.4);
- autocalibration (Section 7.6);
- stratification (Section 7.7);
- 3-D motion (absolute orientation) (Chapter 8).

Demonstrations and source code for most of the original algorithms proposed here are available from the author's WWW page: http://www.dimi.uniud.it/~fusiello.

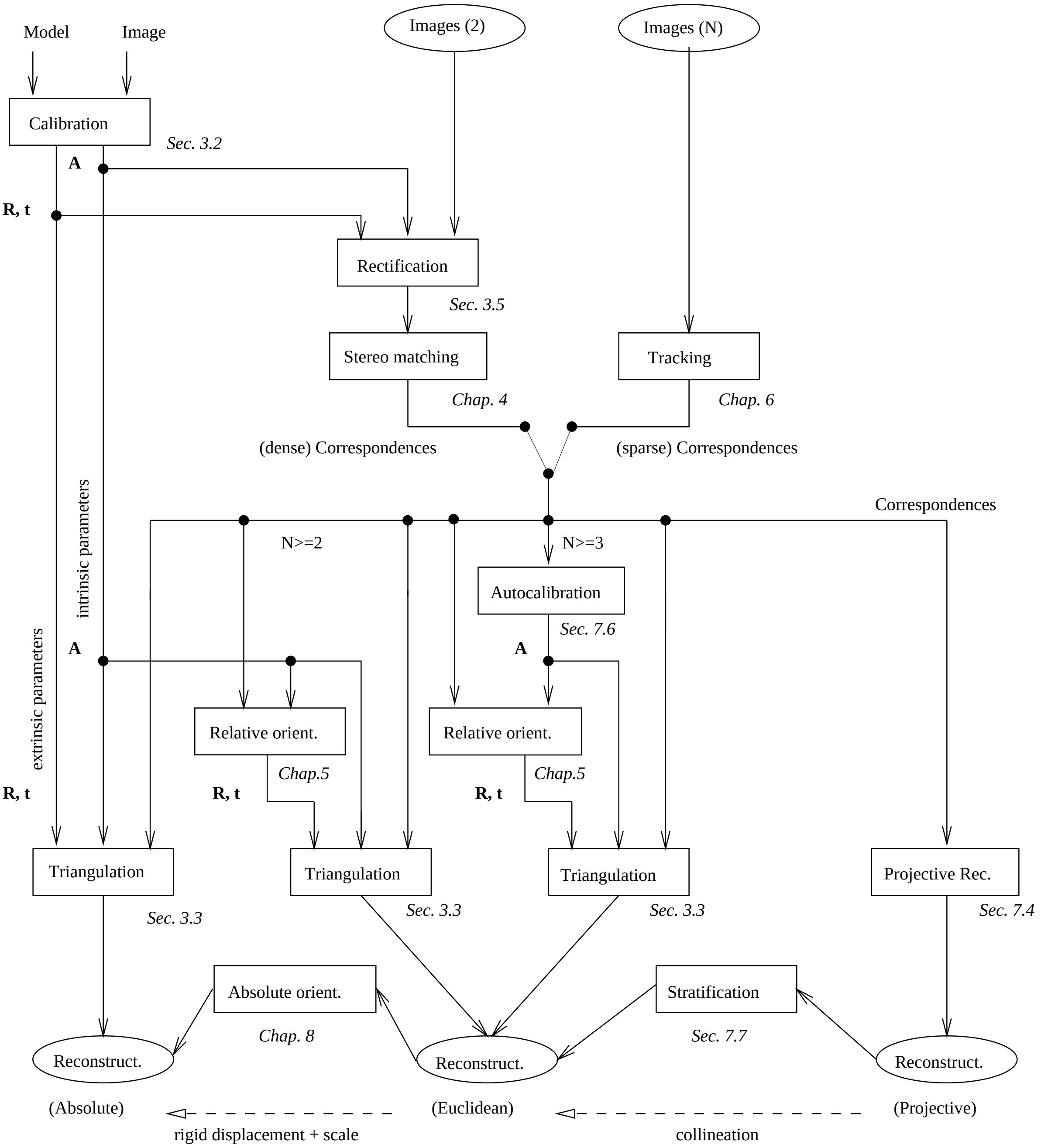

Figure 1: Thesis layout at a glance. **A** represents the intrinsic parameters, $\mathbf{R}, \mathbf{t}$ represent the extrinsic parameters, N is the number of images. Each rectangle represent a module, with the section where it is described close to it.

Chapter 2

Imaging and Camera Model

Computer Vision techniques use images to obtain information about the scene. In order to do that, we have to understand the process of image formation (*imaging*). In this chapter we will introduce a model for this process and, in more detail, a geometric model for the camera upon which all the other chapters rely.

2.1 Fundamentals of imaging

A computer vision device works by gathering light emitted or reflected from objects in the scene and creating a 2-D image. The questions that a model for the imaging process needs to address is "which scene point project to which pixel (*projective geometry*) and what is the color (or the brightness) of that pixel (*radiometry*)?".

2.1.1 Perspective projection

The simplest geometrical model of imaging is the *pinhole camera*.
Let P be a point in the scene, with coordinates $(\mathsf{X}, \mathsf{Y}, \mathsf{Z})$ and P' be its projection on the image plane, with coordinates $(\mathsf{X}', \mathsf{Y}', \mathsf{Z}')$. If f is the distance from the pinhole to the image plane, then by similar triangles, we can derive the following equations:

$$\frac{-\mathsf{X}'}{\mathsf{f}} = \frac{\mathsf{X}}{\mathsf{Z}} \quad \text{and} \quad \frac{-\mathsf{Y}'}{\mathsf{f}} = \frac{\mathsf{Y}}{\mathsf{Z}} \tag{1}$$

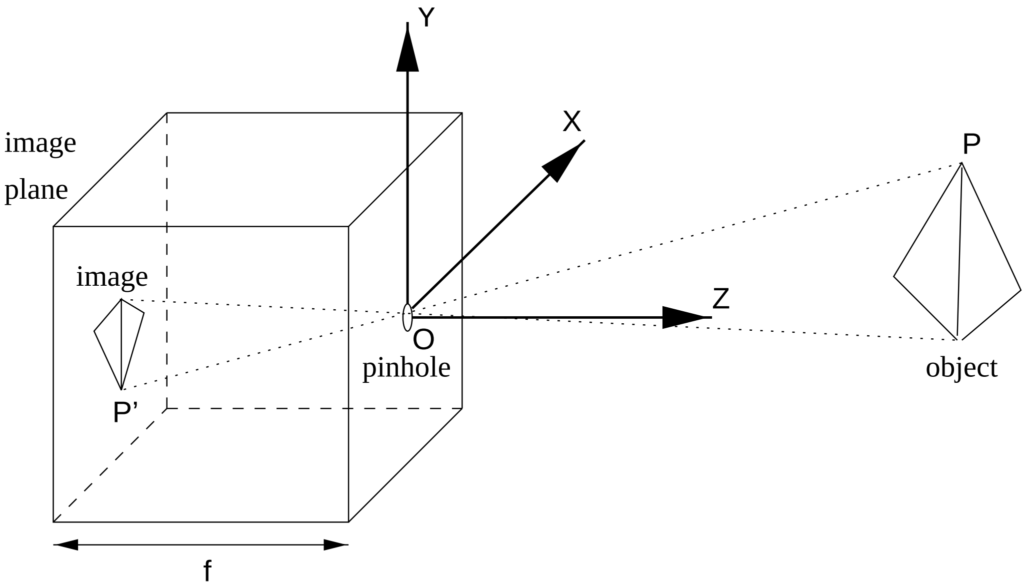

Figure 2: The pinhole camera.

hence

$$\begin{cases} X' = \dfrac{-fX}{Z} \\ Y' = \dfrac{-fY}{Z} \\ Z' = -f \quad . \end{cases} \tag{2}$$

These equations define an image formation process known as *perspective projection*, or *central projection*. Perspective was introduced in painting by L. B. Alberti [1] in 1435, as a technique for making accurate depictions of three-dimensional scenes.

The process is non-linear, owing to the division by Z. Note that the image is inverted, both left-right and up-down, with respect to the scene, as indicated in the equations by the negative signs. Equivalently, we can imagine to put the projection plane at a distance f *in front* of the pinhole, thereby obtaining a non-inverted image.

If the object is relatively shallow compared to its average distance from the camera, we can approximate perspective projection by *scaled orthographic projection* or *weak perspective*. The idea is the following. If the depth Z of the points on the object varies in the range $Z_0 \pm \Delta Z$, with $\Delta Z/Z_0 << 1$, then the perspective scaling factor f/Z can be approximated by a constant f/Z_0. Leonardo da Vinci recommended to use this approximation when $\Delta Z/Z_0 < 1/10$. Then (2) become:

$$X' = \frac{-f}{Z_0} X \qquad Y' = \frac{-f}{Z_0} Y \tag{3}$$

This is the composition of an orthographic projection and a uniform scaling by f/Z_0.

2.1.2 Optics

In the pinhole camera, for each scene point, there is only one light ray that reaches the image plane. A normal lens is actually much wider than a pinhole, which is necessary to collect more light. The drawback is that not all the scene can be in sharp focus at the same time. It is customary to approximate any complex optical systems with a *thin lens*. A thin lens has the following basic properties (refer to Figure 3):

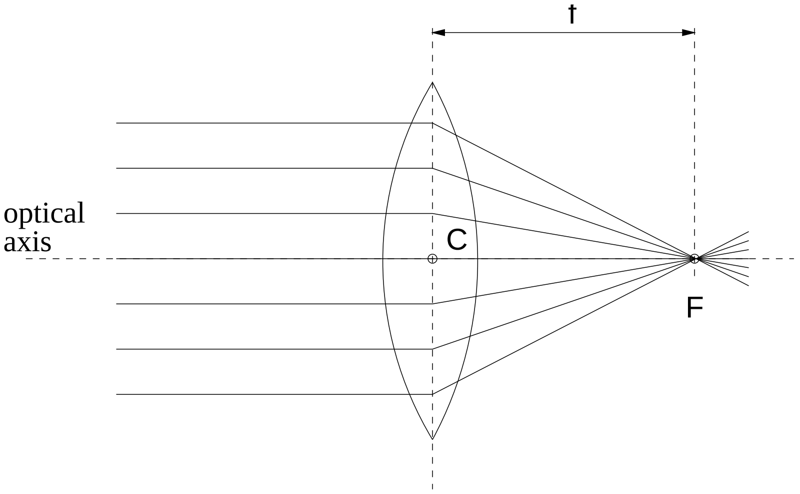

Figure 3: Thin lens.

1. any ray entering the lens parallel to the axis on one side goes through the *focus* F on the other side;

2. any ray going through the lens center C is not deflected.

The distance from the focus F to the lens center C is the *focal length*. It depends on the curvature of both sides of the lens and on the refraction index of the material.

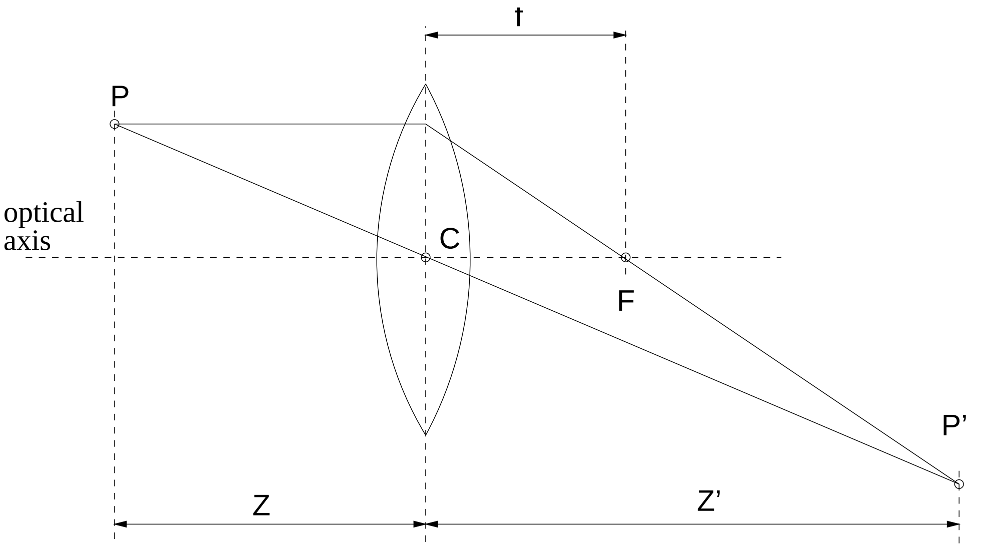

Figure 4: Construction of the image of a point.

Let P be a point in the scene; its image P′ can be obtained, thanks to the two properties of thin lenses, by the intersection of two special rays going through P: the ray parallel to the optical axis and the ray going through C (Figure 4).

Thanks to this construction and by similar triangles, we obtain the *thin lens equation*:

$$\frac{1}{Z} + \frac{1}{Z'} = \frac{1}{f} \quad . \tag{4}$$

The image of a scene point with depth (distance from the lens center) Z will be imaged in sharp focus at a distance Z′ from the lens center, which depends also on the focal length f. As the photosensitive elements in the image plane (rods and cones in the retina, silver halides crystals in photographic films, solid state electronic circuits in digital cameras) have a small but finite dimension, given a choice of Z′, scene points with depth in a range around Z will be in sharp focus. This range is referred as the *depth of field*.

In order to focus objects at different distances, the lens in the eye of vertebrates changes shape, whereas the lens in a camera moves in the Z direction.

2.1.3 Radiometry

The perceived brightness $I(p)$ of a small area p in the image is proportional to the amount of light directed toward the camera by the surface patch S_p that project to p. This in turn depends on the reflectance properties of S_p, the type and position of light sources.

Reflectance is the property of a surface describing the way it reflects incident light. It can be described by taking the ratio of the *radiance*[1] (L) and *irradiance* (E), for each illuminant direction (θ_e, ϕ_e) and each viewing angle (θ_l, ϕ_l), obtaining the *Bidirectional Reflectance Distribution Function* (BRDF):

$$\mathsf{BRDF}(\theta_l, \phi_l, \theta_e, \phi_e) = \frac{L(\theta_l, \phi_l)}{E(\theta_e, \phi_e)} \quad . \tag{5}$$

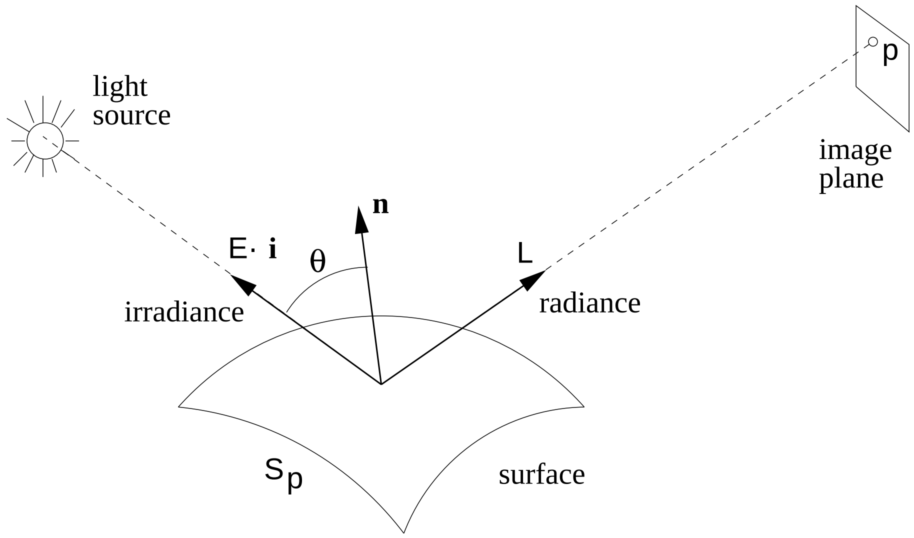

Figure 5: Radiometry of image formation.

Ideally, the light reflected from an object is characterized as being either diffusely or specularly reflected.

Specularly reflected light is reflected from the outer surface of the object. The energy of reflected light is concentrated primarily in a particular direction, such that the reflected and the incident rays are in the same plane and the angle of reflection is equal to the angle of incidence. This is the behavior of a perfect mirror.

[1] The *radiance* (irradiance) of a surface is the power per unit area of emitted (incident) light radiation. The *irradiance* of a surface is the power per unit area of incident light radiation.

Diffused light has been absorbed and re-emitted. The BRDF for a perfect diffusor is given by the well-known Lambert's law:

$$L = \rho E \cos \theta \tag{6}$$

where L is the radiance in S_p, E is the irradiance (the intensity of the light source), ρ is the *albedo*, which varies from 0 (black) to 1 (white), and θ is the angle between the light direction **i** and the surface normal **n** (refer to Figure 5). In the real world objects exhibit a combination of diffuse and specular properties.

In a simplified model of the photometry of image formation it is always assumed that the radiation leaving the surface S_p is equal to the radiation incident in p (no losses), hence the *brightness* $I(p)$ is given by:

$$I(p) = L(S_p). \tag{7}$$

2.1.4 Digital images

A digital image acquisition system consists of three hardware components: a *viewing camera*, a *frame grabber* and a *host computer* (Figure 6).

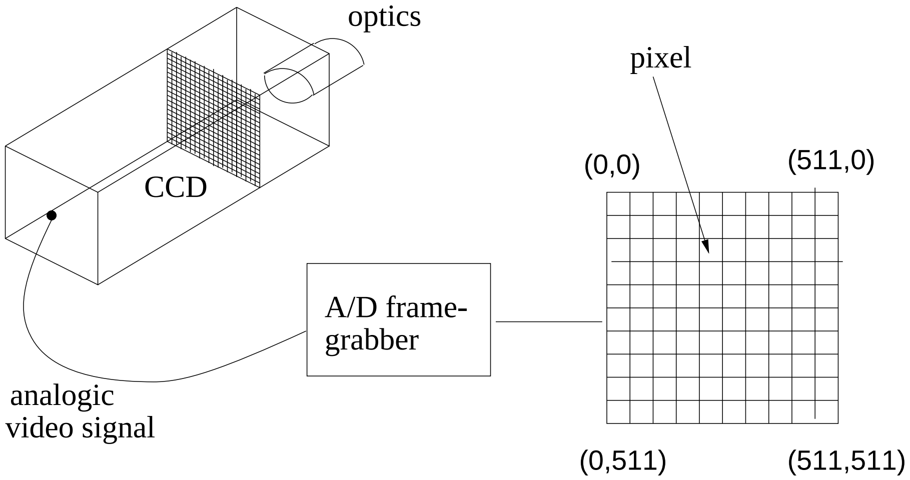

Figure 6: Digital image acquisition system.

The camera is composed by the optical system – which we approximate with a thin lens – and by a CCD (*Charged Coupled Device*) array that constitute the image plane. This can be regarded as a $n \times m$ grid of rectangular photosensitive cells (typically, a CCD array is 1×1 cm and is composed by about 5×10^5 elements), each of them converting the incident light energy into a voltage. The output of the CCD is an analog electric signal, obtained by scanning the photo-sensors by lines and reading the cell's voltage.

This video signal is sent to a device called *frame grabber*, where it is digitized into a 2-D rectangular array of $N \times M$ (typically, 512×512) integer values and stored in a memory buffer. The entries of the array are called *pixel* (*pict*ure *el*ements). We will henceforth denote by $I(u, v)$ the image value (brightness) at the pixel u, v (row v, column u).

The host computer acquires the image by transferring it from the frame buffer to its internal memory. Typical transfer rates are about 25 Hz (1 frame every 40 ms).

The dimensions of the CCD array ($n \times m$) are not necessarily the same as the dimension of the image (array of $N \times M$ pixels): this implies that the position of a point in the image plane is different if measured in CCD elements or in pixels (the latter is what we can measure from images). There is a scale factor relating the two measures:

$$u_{\text{pix}} = \frac{n}{N} u_{\text{CCD}} \tag{8}$$

$$v_{\text{pix}} = \frac{m}{M} v_{\text{CCD}} \tag{9}$$

It is customary to assume that the CCD array is composed by $N \times M$ rectangular elements, whose size is called the *effective pixel size* (measured in $\mathrm{m} \cdot \mathrm{pixel}^{-1}$).

The process of sampling the image plane and transforming the image in digital format, performed by digital image acquisition system, is called *pixelization*.

2.2 Camera model

In this section we will give a more detailed description of the geometric model of the pinhole camera. In particular, following [33], we will draw the mathematical relationship between the 3-D coordinates of a scene point and the coordinates of its projection onto the image plane.

A pinhole camera is modeled by its *optical center* C and its *retinal plane* (or *image plane*) $\mathcal{R}$. A 3-D point W is projected into an image point M given by the intersection of $\mathcal{R}$ with the line containing C and W (Figure 7). The line containing C and orthogonal to $\mathcal{R}$ is called the *optical axis* (the Z axis in Figure 7) and its intersection with $\mathcal{R}$ is the *principal point*. The distance between C and $\mathcal{R}$ is the *focal distance* (note that, since in our model C is behind $\mathcal{R}$, real cameras will have negative focal distance).

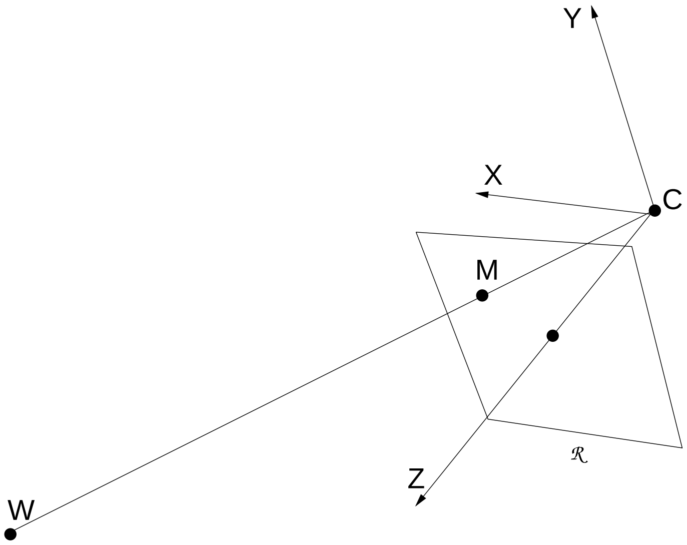

Figure 7: The pinhole camera model, with the *camera reference frame* (X,Y,Z) depicted.

Let us introduce the following reference frames (Figure 8):

- the *world reference frame* x,y,z is an arbitrary 3-D reference frame, in which the position of 3-D points in the scene are expressed, and can be measured directly.
- the *image reference frame* u,v is the coordinate system in which the position of pixels in the image are expressed.
- the *camera standard reference frame* X,Y,Z, is a 3-D frame attached to the camera, centered in C, with the Z axis coincident with the optical axis, X parallel to u and Y parallel to v.

Let us consider first a very special case, in which the world reference frame is taken coincident with the camera reference frame, the focal distance is 1, the effective pixel size is 1, and the u,v reference frame is centered in the principal point.

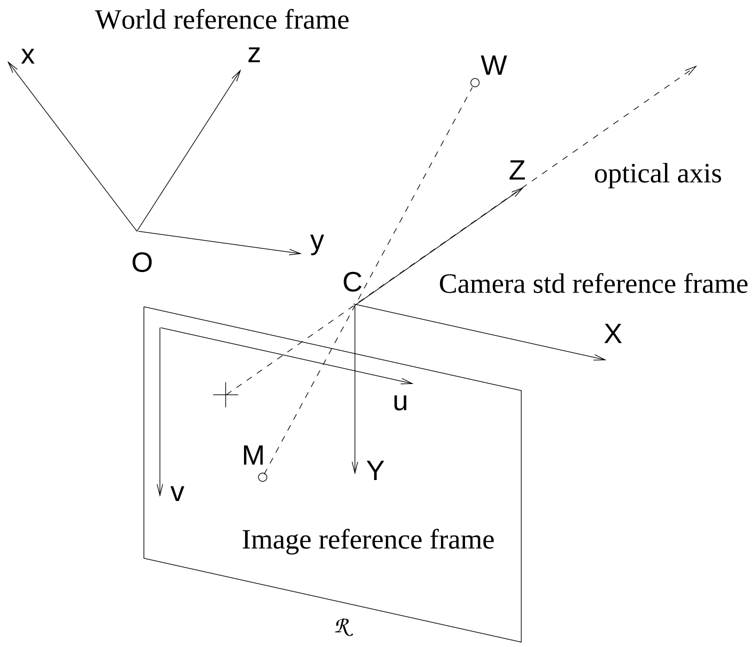

Figure 8: Reference frames.

Let $\mathbf{w} = (x, y, z)$ the coordinates of W in the world reference frame and $\mathbf{m}$ the coordinates of M in the image plane (in pixels). From simple geometrical considerations, as we did in Section 2.1.1, we obtain the following relationship:

$$\frac{1}{z} = \frac{u}{x} = \frac{v}{y} \tag{10}$$

that is

$$\begin{cases} u = \dfrac{1}{z}\,x \\ v = \dfrac{1}{z}\,y \end{cases} . \tag{11}$$

This is the *perspective projection.* The mapping from 3-D coordinates to 2-D coordinates is clearly non-linear; using homogeneous coordinates, instead, it becomes linear. Homogeneous coordinates are simply obtained by adding an arbitrary component to the usual Cartesian coordinates (see Appendix A). Cartesian coordinates can be obtained by dividing each homogeneous component by the last one and removing the “1” in last position. Therefore, there is a one to many correspondence between Cartesian and homogeneous coordinates. Homogeneous coordinates can represent the usual Euclidean points plus the points at infinity, which are points with the last component equal to zero, that does not have a Cartesian counterpart.

Let

$$\tilde{\mathbf{m}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{w}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{12}$$

be the homogeneous coordinates of M and W respectively. We will henceforth use the superscript ˜ to denote homogeneous coordinates. The projection equation, in this simplified case, writes:

$$\begin{bmatrix} \kappa u \\ \kappa v \\ \kappa \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{13}$$

Note that the value of κ is equal to the third coordinate of the W, which – in this special reference frame – coincides with the distance of the point to the plane XY. Points with $\kappa = 0$ are projected to infinity. They lie on the plane $\mathcal{F}$ parallel to $\mathcal{R}$ and containing C, called the *focal plane*.

Hence, in homogeneous coordinates, the projection equation writes

$$\kappa \tilde{\mathbf{m}} = \tilde{\mathbf{P}} \tilde{\mathbf{w}}. \tag{14}$$

or,

$$\tilde{\mathbf{m}} \simeq \tilde{\mathbf{P}} \tilde{\mathbf{w}}. \tag{15}$$

where $\simeq$ means "equal up to an arbitrary scale factor".

The matrix $\tilde{\mathbf{P}}$ represent the geometric model of the camera, and is called the *camera matrix* or *perspective projection matrix* (PPM). In this very special case we have

$$\tilde{\mathbf{P}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [\mathbf{I}|\mathbf{0}] .$$

2.2.1 Intrinsic parameters

In a more realistic model of the camera, the retinal plane is placed *behind* the projection center at a certain distance f. Projection equations become

$$\begin{cases} u = \frac{-f}{z} x \\ v = \frac{-f}{z} y \quad , \end{cases} \tag{16}$$

where f is the *focal distance* in meters.

Moreover, pixelization must be taken into account, by introducing a translation of the principal point and a scaling of u and v axis:

$$\begin{cases} u = k_u \frac{-f}{z} x + u_0 \\ v = k_v \frac{-f}{z} y + v_0 \quad , \end{cases} \tag{17}$$

where (u_0, v_0) are the coordinates of the principal point, k_u (k_v) is the inverse of the effective pixel size along the horizontal (vertical) direction, measured in $\text{pixel} \cdot \text{m}^{-1}$.

After these changes, the PPM writes:

$$\tilde{\mathbf{P}} = \begin{bmatrix} -fk_u & 0 & u_0 & 0 \\ 0 & -fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \mathbf{A}[\mathbf{I}|\mathbf{0}] \tag{18}$$

where

$$\mathbf{A} = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{19}$$

If the CCD grid is not rectangular, u and v are not orthogonal; if θ is the angle they form, then the matrix $\mathbf{A}$ becomes:

$$\mathbf{A} = \begin{bmatrix} -fk_u & fk_u \cot\theta & u_0 \\ 0 & -fk_v / \sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{20}$$

Hence, the matrix $\mathbf{A}$ has – in general – the following form:

$$\mathbf{A} = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} , \tag{21}$$

where $\alpha_u = -fk_u$, $\alpha_v = -fk_v/\sin\theta$ are the focal lengths in horizontal and vertical pixels, respectively, and $\gamma = fk_u \cot\theta$ is the *skew factor.* The parameters $\alpha_u, \alpha_v, \gamma, u_0$, and v_0 are called *intrinsic parameters*.

Normalized coordinates

It is possible to undo the pixelization by pre-multiplying the pixel coordinates by the inverse of $\mathbf{A}$, obtaining the so called *normalized coordinates*, giving the position of a point on the retinal plane, *measured in meters*:

$$\tilde{\mathbf{p}} = \mathbf{A}^{-1}\tilde{\mathbf{m}}. \tag{22}$$

The homogeneous normalized coordinates of a point in the retinal plane can be interpreted (see Appendix A) as a 3-D vector centered in C and pointing toward the point on the retinal plane, whose equation is $z = 1$. This vector, of which only the direction is important, is called *ray vector*.

2.2.2 Extrinsic parameters

Let us now change the world reference system, which was taken as coincident with the camera standard reference frame. The rigid transformation that brings the camera reference frame onto the new world reference frame encodes the camera's position and orientation. This transformation is defined in terms of the 3×3 rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$. If $\mathbf{w}_{\text{std}}$ and $\mathbf{w}_{\text{new}}$ are the Cartesian coordinates of the scene point in these two frames, we have:

$$\mathbf{w}_{\text{std}} = \mathbf{R}\mathbf{w}_{\text{new}} + \mathbf{t}. \tag{23}$$

Using homogeneous coordinates the latter rewrites:

$$\tilde{\mathbf{w}}_{\text{std}} = \mathbf{G}\tilde{\mathbf{w}}_{\text{new}} \tag{24}$$

where

$$\mathbf{G} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{25}$$

The PPM yielded by this reference change is the following:

$$\tilde{\mathbf{P}} = \mathbf{A}[\mathbf{I}|\mathbf{0}]\mathbf{G} = \mathbf{A}[\mathbf{R}|\mathbf{t}] = [\mathbf{A}\mathbf{R}|\mathbf{A}\mathbf{t}]. \tag{26}$$

The three entries of the translation vector $\mathbf{t}$ and the three parameters[2] that encodes $\mathbf{R}$ are the *extrinsic parameters*.

Since $\tilde{\mathbf{w}}_{\text{new}} = \mathbf{G}^{-1}\tilde{\mathbf{w}}_{\text{std}}$, with

$$\mathbf{G}^{-1} = \begin{bmatrix} \mathbf{R}^{\top} & -\mathbf{R}^{\top}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{27}$$

the columns of $\mathbf{R}^{\top}$ are the coordinates of the axis of the standard reference frame relative to the world reference frame and $-\mathbf{R}^{\top}\mathbf{t}$ is the position of the optical center C in the world reference frame.

2.2.3 Some properties of the PPM

Let us write the PPM as

$$\tilde{\mathbf{P}} = \left[\begin{array}{c|c} \mathbf{q}_1^{\top} & q_{14} \\ \mathbf{q}_2^{\top} & q_{24} \\ \mathbf{q}_3^{\top} & q_{34} \end{array}\right] = [\mathbf{Q}|\tilde{\mathbf{q}}]. \tag{28}$$

Projection in Cartesian coordinates

From (14) we obtain by substitution:

$$\begin{bmatrix} \kappa u \\ \kappa v \\ \kappa \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^{\top}\mathbf{w} + q_{14} \\ \mathbf{q}_2^{\top}\mathbf{w} + q_{24} \\ \mathbf{q}_3^{\top}\mathbf{w} + q_{34} \end{bmatrix} \tag{29}$$

Hence, the perspective projection in Cartesian coordinates writes

$$\begin{cases} u = \dfrac{\mathbf{q}_1^{\top}\mathbf{w} + q_{14}}{\mathbf{q}_3^{\top}\mathbf{w} + q_{34}} \\ v = \dfrac{\mathbf{q}_2^{\top}\mathbf{w} + q_{24}}{\mathbf{q}_3^{\top}\mathbf{w} + q_{34}}. \end{cases} \tag{30}$$

Optical center

The *focal plane* (the plane XY in Figure 7) is parallel to the retinal plane and contains the optical center. It is the locus of the points projected to infinity, hence its equation

[2] A rotation in the 3-D space can be parameterized by means of the three Euler angles, for example.

is $\mathbf{q}_3^\top \mathbf{w} + q_{34} = 0$. The two planes defined by $\mathbf{q}_1^\top \mathbf{w} + q_{14} = 0$ and $\mathbf{q}_2^\top \mathbf{w} + q_{24} = 0$ intersect the retinal plane in the vertical and horizontal axis of the retinal coordinates, respectively. The optical center C is the intersection of these three planes, hence its coordinates $\mathbf{c}$ are the solution of

$$\tilde{\mathbf{P}} \begin{bmatrix} \mathbf{c} \\ 1 \end{bmatrix} = \mathbf{0}, \tag{31}$$

then

$$\mathbf{c} = -\mathbf{Q}^{-1}\tilde{\mathbf{q}}. \tag{32}$$

From the latter a different way of writing $\tilde{\mathbf{P}}$ is obtained:

$$\tilde{\mathbf{P}} = [\mathbf{Q}| - \mathbf{Q}\mathbf{c}]. \tag{33}$$

Optical ray

The *optical ray* associated to an image point M is the locus of the points that are projected to M, $\{\mathbf{w} : \tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{w}}\}$, i.e., the line MC. A point on the optical ray of M is the optical center, that belongs to *every* optical ray; another point on the optical ray of M is the *point at infinity*, of coordinates

$$\begin{bmatrix} \mathbf{Q}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix},$$

indeed:

$$\tilde{\mathbf{P}} \begin{bmatrix} \mathbf{Q}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix} = \mathbf{Q}\mathbf{Q}^{-1}\tilde{\mathbf{m}} = \tilde{\mathbf{m}}.$$

The parametric equation of the optical ray is therefore (in projective coordinates):

$$\tilde{\mathbf{w}} = \begin{bmatrix} \mathbf{c} \\ 1 \end{bmatrix} + \lambda \begin{bmatrix} \mathbf{Q}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix} \quad \lambda \in \mathbb{R}. \tag{34}$$

In Cartesian coordinates, it re-writes:

$$\mathbf{w} = \mathbf{c} + \lambda \mathbf{Q}^{-1}\tilde{\mathbf{m}} \quad \lambda \in \mathbb{R}. \tag{35}$$

Factorization of the PPM

The camera is modeled by its perspective projection matrix $\tilde{\mathbf{P}}$, which has the form (26), in general. Vice versa, a PPM can be decomposed, using the QR factorization, into the product

$$\tilde{\mathbf{P}} = \mathbf{A}[\mathbf{R}|\mathbf{t}] = [\mathbf{A}\mathbf{R}|\mathbf{A}\mathbf{t}]. \tag{36}$$

Indeed, given $\tilde{\mathbf{P}} = [\mathbf{Q}|\tilde{\mathbf{q}}]$, by comparison with (36) we obtain $\mathbf{Q} = \mathbf{A}\mathbf{R}$, that is $\mathbf{Q}^{-1} = \mathbf{R}^{-1}\mathbf{A}^{-1}$. Let $\mathbf{Q}^{-1} = \mathbf{U}\mathbf{B}$ be the QR factorization of $\mathbf{Q}^{-1}$, where $\mathbf{U}$ is orthogonal and $\mathbf{B}$ is upper triangular. Hence $\mathbf{R} = \mathbf{U}^{-1}$ and $\mathbf{A} = \mathbf{B}^{-1}$. Moreover $\mathbf{t} = \mathbf{A}^{-1}\tilde{\mathbf{q}} = \mathbf{B}\tilde{\mathbf{q}}$.

Parameterization of the PPM

If we write

$$\mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad \text{and} \quad \mathbf{R} = \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix} \tag{37}$$

from (20) and (26) we obtain the following expression for $\tilde{\mathbf{P}}$ as a function of the intrinsic and extrinsic parameters

$$\tilde{\mathbf{P}} = \begin{bmatrix} \alpha_u \mathbf{r}_1^\top - \dfrac{\alpha_u}{\tan\theta}\mathbf{r}_2^\top + u_0\mathbf{r}_3^\top & \alpha_u t_1 - \dfrac{\alpha_u}{\tan\theta}\mathbf{r}_2^\top t_2 + u_0 t_3 \\ \dfrac{\alpha_v}{\sin\theta}\mathbf{r}_2^\top + v_0\mathbf{r}_3^\top & \dfrac{\alpha_v}{\sin\theta}t_2 + v_0 t_3 \\ \mathbf{r}_3^\top & t_3 \end{bmatrix} \tag{38}$$

In the hypothesis, usually verified in practice with good approximation, that $\theta = \pi/2$, we obtain:

$$\tilde{\mathbf{P}} = \begin{bmatrix} \alpha_u \mathbf{r}_1^\top + u_0\mathbf{r}_3^\top & \alpha_u t_1 + u_0 t_3 \\ \alpha_v \mathbf{r}_2^\top + v_0\mathbf{r}_3^\top & \alpha_v t_2 + v_0 t_3 \\ \mathbf{r}_3^\top & t_3 \end{bmatrix} \tag{39}$$

A generic PPM, defined up to a scale factor, must be normalized in such a way that $||\mathbf{q}_3|| = 1$ if it has to be parameterized as (38) or (39).

Projective depth

The parameter κ that appear in (14) is called *projective depth.* If the PPM is normalized with $\|\mathbf{q}_3\| = 1$, it is the distance of W from the focal plane (i.e., its depth). Indeed, from (29) and (38) we have:

$$\kappa = \mathbf{r}_3^\top \mathbf{w} + t_3. \tag{40}$$

Since $\tilde{\mathbf{w}}_{\text{std}} = \mathbf{G}\tilde{\mathbf{w}}_{\text{new}}$, κ is the third coordinate of the representation of W in the camera standard reference, hence just its distance from the focal plane.

Vanishing points

The perspective projection of a pencil of parallel lines in space is a pencil of lines in the image plane passing through a common point, called the *vanishing point.* Let us consider a line whose parametric equation is $\mathbf{w} = \mathbf{a} + \lambda\mathbf{n}$, where $\mathbf{n}$ is the direction. Its projection on the image plane has parametric equation:

$$\begin{cases} u = \dfrac{\mathbf{q}_1^\top(\mathbf{a}+\lambda\mathbf{n}) + q_{14}}{\mathbf{q}_3^\top(\mathbf{a}+\lambda\mathbf{n}) + q_{34}} \\ \\ v = \dfrac{\mathbf{q}_2^\top(\mathbf{a}+\lambda\mathbf{n}) + q_{24}}{\mathbf{q}_3^\top(\mathbf{a}+\lambda\mathbf{n}) + q_{34}} \end{cases}. \tag{41}$$

The vanishing point (u_∞, v_∞) is obtained by sending λ to infinity:

$$\begin{cases} u_\infty = \lim\limits_{\lambda\to\infty} \dfrac{\mathbf{q}_1^\top\mathbf{a} + \lambda\mathbf{q}_1^\top\mathbf{n} + q_{14}}{\mathbf{q}_3^\top\mathbf{a} + \lambda\mathbf{q}_3^\top\mathbf{n} + q_{34}} = \dfrac{\mathbf{q}_1^\top\mathbf{n}}{\mathbf{q}_3^\top\mathbf{n}} \\ \\ v_\infty = \lim\limits_{\lambda\to\infty} \dfrac{\mathbf{q}_2^\top\mathbf{a} + \lambda\mathbf{q}_2^\top\mathbf{n} + q_{24}}{\mathbf{q}_3^\top\mathbf{a} + \lambda\mathbf{q}_3^\top\mathbf{n} + q_{34}} = \dfrac{\mathbf{q}_2^\top\mathbf{n}}{\mathbf{q}_3^\top\mathbf{n}} \end{cases}. \tag{42}$$

2.3 Conclusions

An image is the projection of the 3-D space onto a 2-D array, and it contains two types of visual cues: geometric and radiometric. The former is related to the position of image points, the latter to their brightness. In this work we will deal mainly with the geometric aspect of Computer Vision, and to this purpose we described in detail the geometric model of the pinhole camera (the missing topics are covered for instance in [149]), that establishes the relationship between the world coordinates of a scene point and the image coordinates of its projection. From a geometrical standpoint, the camera is full modeled by a 3×4 matrix, in homogeneous coordinates. We described some useful properties of this matrix, that will be needed in the following chapters.

Chapter 3

Structure from Stereo

In this chapter and in the next one, we will address the following problem: given two pictures of a scene (a *stereo pair*) taken with a *calibrated* rig of two cameras, for which intrinsic and extrinsic parameters have been measured, and a set of matched points, which are all images of points located in the scene, reconstruct the 3-D coordinates of the points.

We will discuss here the geometrical issues of stereo reconstruction. The computation of corresponding points is treated in the next chapter.

After describing simple linear calibration and reconstruction algorithms, we will concentrate on the *epipolar geometry*, i.e., the relationship between corresponding points and in particular on *rectification*, an operation meant to insure a simple epipolar geometry for a stereo pair. The main original contribution of this chapter is to introduce a *linear rectification algorithm for general, unconstrained stereo rigs.* The algorithm takes the two perspective projection matrices of the original cameras, and computes a pair of rectifying projection matrices. We report tests proving the correct behavior of our method, as well as the negligible decrease of the accuracy of 3-D reconstruction if performed from the rectified images directly.

3.1 Introduction

The aim of *structure from stereo* [16, 30] is to reconstruct the 3-D geometry of a scene from two views, which we call *left* and *right*, taken by two pinhole cameras. Two distinct processes are involved: *correspondence* (or matching) and *reconstruction.*

The former estimates which points in the left and right images are projections of the same scene point (a *conjugate pair*). The 2-D displacement vector between conjugate points, when the two images are superimposed, is called *disparity*. Stereo matching will be addressed in the next chapter. Reconstruction (Section 3.3) recovers the full 3-D coordinates of points, using the estimated disparity and a model of the stereo rig, specifying the pose and position of each camera and its internal parameters. The measurement of camera model parameters is known as *calibration* (Section 3.2).

The coordinates of conjugate points are related by the so-called *epipolar geometry* (Section 3.4). Given a point in one image, its conjugate must belong to a line in the other image, called the epipolar line. Given a pair of stereo images, *rectification* determines a transformation of each image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes. The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras. The important advantage of rectification is that computing correspondences is made much simpler.

In Section 3.5 we present a novel algorithm for rectifying a calibrated stereo rig of unconstrained geometry and mounting general cameras. The only input required is the pair of perspective projection matrices (PPM) of the two cameras, which are estimated by calibration. The output is the pair of *rectifying* PPMs, which can be used to compute the rectified images. Reconstruction can also be performed directly from the rectified images and PPMs. Section 3.5.1 derive the algorithm for computing the rectifying PPMs and Section 3.5.2 expresses the rectifying image transformation in terms of PPMs. Section 3.5.3 gives the compact (20 lines), working MATLAB code for our algorithm. A formal proof of the correctness of our algorithm is given in Section 3.5.4. Section 3.5.5 reports tests on synthetic and real data. Section 3.6 is a brief discussion of our work.

A "rectification kit" containing code, examples data and instructions is available on line (http://www.dimi.uniud.it/~fusiello/rect.html).

3.2 Calibration

Calibration consist in *computing as accurately as possible the intrinsic and extrinsic parameters of the camera.* These parameters determine the way 3-D points project to image points. If enough correspondences between world points and image points are available, it is possible to compute camera intrinsic and extrinsic parameters by solving the perspective projection equation for the unknown parameters. In photogrammetry these two problem are known as *interior orientation* problem and *exterior orientation* problem[1] respectively. Some direct calibration methods cast the problem in terms of the camera parameters [38, 150, 22, 134], others solve for the unknown entries of $\tilde{\mathbf{P}}$ [33, 121]. They are equivalent since, as we already know, parameters can be factorized out from $\tilde{\mathbf{P}}$. In our experiments we used the algorithm (and the code) developed by L. Robert [121]. In this section we will describe a simple linear method for camera calibration, which, in practice, requires a subsequent non-linear iterative refinement, as in [121].

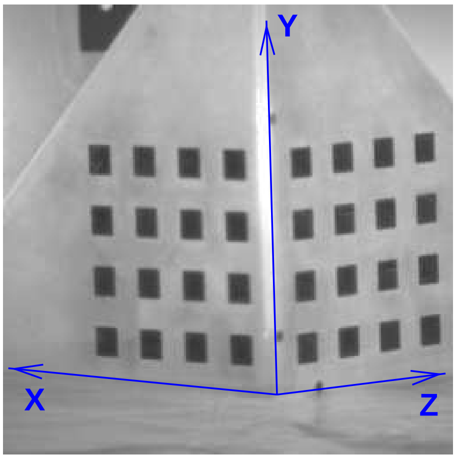

Figure 9: Picture of the calibration jig, with superimposed the world reference system.

[1] In particular the exterior orientation problem is relevant in the so-called CAD-based Vision [21], in which one has a model of an object, a camera with known intrinsic parameters and wants to recognize the image of the object by aligning it with the model [152]. One method to perform alignment is to estimate camera's pose, solving the exterior orientation problem, project the model accordingly, and then match the projection with the image to refine the estimate [88].

Linear-LS method

Given N reference points, not coplanar, each correspondence between an image point $\mathbf{m}_i = [u_i, v_i]^\top$, and a reference point $\mathbf{w}_i$ gives a pair of equations, derived from (30):

$$\begin{cases} (\mathbf{q}_1 - u_i\mathbf{q}_3)^\top \mathbf{w}_i + q_{14} - u_i q_{34} = 0 \\ (\mathbf{q}_2 - v_i\mathbf{q}_3)^\top \mathbf{w}_i + q_{24} - v_i q_{34} = 0 \end{cases} \tag{43}$$

The unknown PPM is composed by 12 elements, but being defined up to a scale factor (homogeneous coordinates) it depends on 11 parameters only. We can choose $q_{34} = 1$, thereby reducing the unknown to 11, obtaining the following two equations:

$$\begin{bmatrix} \mathbf{w}_i^\top & 1 & \mathbf{0} & 0 & -u_i\mathbf{w}_i^\top \\ \mathbf{0} & 0 & \mathbf{w}_i^\top & 1 & -v_i\mathbf{w}_i^\top \end{bmatrix} [\mathbf{q}_1^\top, q_{14}, \mathbf{q}_2^\top, q_{24}, \mathbf{q}_3^\top]^\top = \begin{bmatrix} u_i \\ v_i \end{bmatrix}. \tag{44}$$

For N points we obtain a linear system of 2N equations in 11 unknowns: 6 non coplanar points are sufficient. In practice more points are available, and one has to solve a linear least-squares problem. Singular Value Decomposition (SVD) can be used to solve the linear least-square problem $\mathbf{Lx} = \mathbf{b}$ (see [50]). Let $\mathbf{L} = \mathbf{UDV}^\top$ the SVD of $\mathbf{L}$. The least-squares solution is $\mathbf{b} = (\mathbf{VD}^+\mathbf{U}^\top)\mathbf{x}$ where $\mathbf{D}^+$ is constructed by substituting the non-zero elements of $\mathbf{D}$ with their inverse.

Please note that the PPM yielded by this method needs to be normalized with $||\mathbf{q}_3|| = 1$, if it has to be interpreted like (38).

The previous approach has the advantage of providing closed-form solution quickly, but the disadvantage that the criterion that is minimized does not have a geometrical interpretation. The quantity we are actually interested in minimizing is the distance in the image plane between the points and the reprojected reference points:

$$\chi = \sum_{i=1}^{n} \left\| \frac{\mathbf{q}_1^\top \mathbf{w}_i + q_{14}}{\mathbf{q}_3^\top \mathbf{w}_i + q_{34}} - u_i \right\|^2 + \left\| \frac{\mathbf{q}_2^\top \mathbf{w}_i + q_{24}}{\mathbf{q}_3^\top \mathbf{w}_i + q_{34}} - v_i \right\|^2. \tag{45}$$

This lead to a non-linear minimization, but results are more accurate, being less sensitive to noise.

Robert's calibration method [121] take a slightly different approach: the basic idea is to replace the distance by a criterion computed directly from the gray-level image, without extracting calibration points $\mathbf{m}_i$ explicitly. It proceeds by first computing a rough estimate of the projection matrix, then refining the estimate using a technique analogous to active contours [81]. The initialization stage use the linear-LS algorithm. It takes as input a set of 6 non-coplanar 3-D anchor points, and their 2-D images, obtained manually by a user who clicks their approximate position. The refinement stage requires a set of 3-D model points which should project in the image onto edge points. Using non-linear optimization over the camera parameters, the program maximize the image gradient at the position where the model points project.

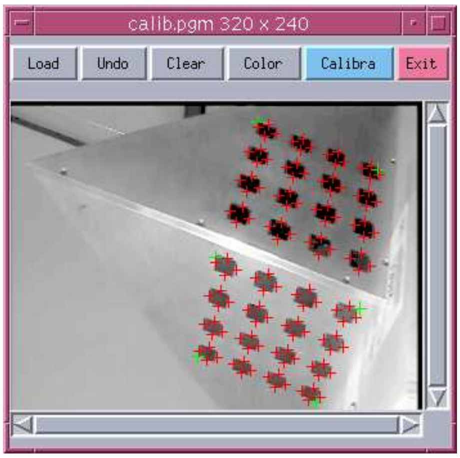

Figure 10: Screen shot of *Calibtool*, the interface to the calibration program. The user must simply select with the mouse six predefined points on the calibration pattern and then choose “Calibra”. The PPM is returned in a file.

3.3 Reconstruction

In the contest of structure from stereo, reconstruction consists in computing the Cartesian coordinates of 3-D points (structure) starting from a set of matched points in the image pair, and from known camera parameters. Given the PPMs of the two cameras and the coordinates of a pair of conjugate points, the coordinates of the world point of which they both are projection can be recovered by a simple linear algorithm. Geometrically, the process can be thought as intersecting the optical rays of the two image points, and for this reason it is sometimes called *triangulation*.

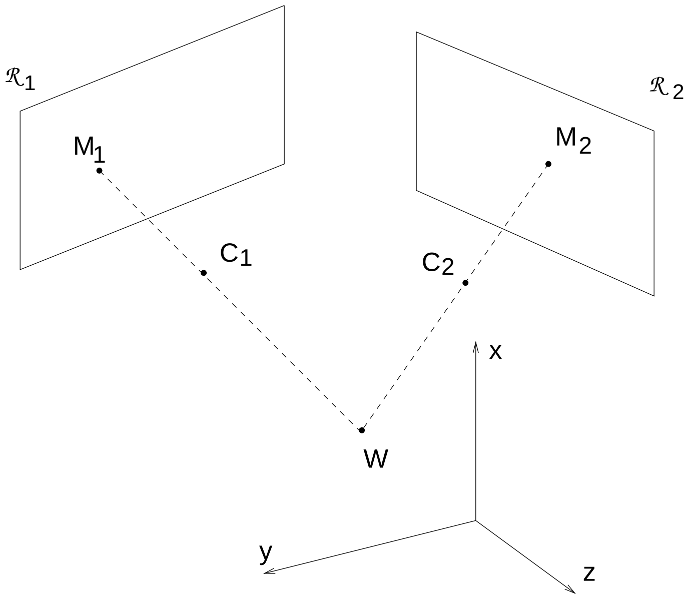

Figure 11: Triangulation.

Linear-Eigen method.

Let $\tilde{\mathbf{w}} = [x, y, z, t]^\top$ the sought coordinates of the world point[2], and let $\mathbf{m} = [u, v]^\top$ and $\mathbf{m}' = [u', v']^\top$ the image coordinates of a conjugate pair. Let

$$\tilde{\mathbf{P}} = \left[\begin{array}{c|c} \mathbf{q}_1^\top & q_{14} \\ \mathbf{q}_2^\top & q_{24} \\ \mathbf{q}_3^\top & q_{34} \end{array}\right] \text{ and } \tilde{\mathbf{P}}' = \left[\begin{array}{c|c} \mathbf{q}_1'^\top & q_{14}' \\ \mathbf{q}_2'^\top & q_{24}' \\ \mathbf{q}_3'^\top & q_{34}' \end{array}\right] \tag{46}$$

From (15) we obtain an homogeneous linear system of four equation in the unknown x, y, z, t:

$$\begin{bmatrix} (\mathbf{q}_1 - u\mathbf{q}_3)^\top + q_{14} - uq_{34} \\ (\mathbf{q}_2 - v\mathbf{q}_3)^\top + q_{24} - vq_{34} \\ (\mathbf{q}_1' - u'\mathbf{q}_3')^\top + q_{14}' - u'q_{34}' \\ (\mathbf{q}_2' - v'\mathbf{q}_3')^\top + q_{24}' - v'q_{34}' \end{bmatrix} \tilde{\mathbf{w}} = \mathbf{0}. \tag{47}$$

These equations defines $\tilde{\mathbf{w}}$ only up to a scale factor, i.e., the system matrix $\mathbf{L}$ is rank-deficient. In order to avoid the trivial solution $\tilde{\mathbf{w}} = \mathbf{0}$, we solve the following constrained minimization problem

$$\min \|\mathbf{L}\mathbf{w}\| \qquad \text{subject to } \|\mathbf{w}\| = 1, \tag{48}$$

[2] We use the parameter t instead of 1 as the homogeneous component of $\tilde{\mathbf{w}}$ in order to accommodate for points at infinity (in practice, far from the camera) that have $t = 0$.

whose solution is the unit eigenvector corresponding to the smallest eigenvalue of the matrix $\mathbf{L}^\top \mathbf{L}$. SVD can be used also to solve this problem. Indeed, if $\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ is the SVD of $\mathbf{L}$, then the solution is the column of $\mathbf{V}$ corresponding to the smallest singular value of $\mathbf{L}$.

As in the case of calibration, a cause of inaccuracy in this linear method is that the value being minimized ($||\mathbf{L}\mathbf{x}||$) has no geometric meaning. A minimization of a suitable cost function, like the error in the image plane, should be performed to achieve better accuracy [33, 64, 168]:

$$\chi = ||\mathbf{m} - \tilde{\mathbf{P}}\hat{\mathbf{w}}|| + ||\mathbf{m}' - \tilde{\mathbf{P}}'\hat{\mathbf{w}}'||. \tag{49}$$

where $\hat{\mathbf{w}}$ is the sought estimate of the coordinates of W. See [65] for a discussion about algebraic versus geometric error minimization in gometric Computer Vision.

3.4 Epipolar geometry

Let us consider a stereo rig composed by two pinhole cameras (Figure 12). Let C_1 and C_2 be the optical centers of the left and right cameras respectively. A 3-D point W is projected onto both image planes, to points M_1 and M_2, which constitute a conjugate pair. Given a point M_1 in the left image plane, its conjugate point in the right image is constrained to lie on a line called the *epipolar line* (of M_1). Since M_1 may be the projection of an arbitrary point on its optical ray, the epipolar line is the projection through C_2 of the optical ray of M_1. All the epipolar lines in one image plane pass through a common point (E_1 and E_2 respectively.) called the *epipole*, which is the projection of the conjugate optical center.

The fundamental matrix

Given two camera matrices, a world point of coordinates $\tilde{\mathbf{w}}_1$, is projected onto a pair of conjugate points of coordinates $\tilde{\mathbf{m}}_1$ and $\tilde{\mathbf{m}}_2$:

$$\begin{cases} \tilde{\mathbf{m}}_1 \simeq \tilde{\mathbf{P}}_1 \tilde{\mathbf{w}} \\ \tilde{\mathbf{m}}_2 \simeq \tilde{\mathbf{P}}_2 \tilde{\mathbf{w}}. \end{cases}$$

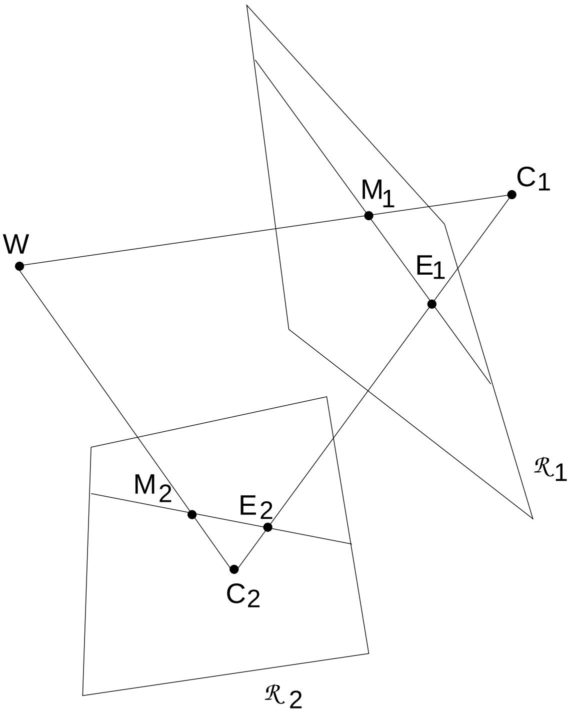

Figure 12: Epipolar geometry. The epipole E_1 of the first camera is the projection of the optical center C_2 of the second camera (and vice versa).

The equation of the epipolar line of $\tilde{\mathbf{m}}_1$ can be easily obtained by projecting the optical ray of $\tilde{\mathbf{m}}_1$

$$\tilde{\mathbf{w}} = \begin{bmatrix} \mathbf{c}_1 \\ 1 \end{bmatrix} + \lambda \begin{bmatrix} \mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1 \\ 0 \end{bmatrix} \tag{50}$$

with $\tilde{\mathbf{P}}_2$. From

$$\tilde{\mathbf{P}}_2 \begin{bmatrix} \mathbf{c}_1 \\ 1 \end{bmatrix} = \tilde{\mathbf{P}}_2 \begin{bmatrix} -\mathbf{Q}_1^{-1}\tilde{\mathbf{q}}_1 \\ 1 \end{bmatrix} = \tilde{\mathbf{q}}_2 - \mathbf{Q}_2\mathbf{Q}_1^{-1}\tilde{\mathbf{q}}_1 = \mathbf{e}_2 \tag{51}$$

and

$$\tilde{\mathbf{P}}_2 \begin{bmatrix} \mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1 \\ 0 \end{bmatrix} = \mathbf{Q}_2\mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1 \tag{52}$$

we obtain the equation of the epipolar line of $\tilde{\mathbf{m}}_1$:

$$\tilde{\mathbf{m}}_2 = \mathbf{e}_2 + \lambda \mathbf{Q}_2\mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1. \tag{53}$$

This is the equation of a line going through the points $\mathbf{e}_2$ (the epipole) and $\mathbf{Q}_2\mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1$. The collinearity of these two points and $\tilde{\mathbf{m}}_2$ is expressed in the projective plane by the triple product (see Appendix A):

$$\tilde{\mathbf{m}}_2^\top(\mathbf{e}_2 \wedge \mathbf{Q}_2\mathbf{Q}_1^{-1}\tilde{\mathbf{m}}_1) = 0, \tag{54}$$

which can be written in the more compact form

$$\tilde{\mathbf{m}}_2^\top \mathbf{F} \tilde{\mathbf{m}}_1 = 0, \tag{55}$$

by introducing the *fundamental matrix* $\mathbf{F}$:

$$\mathbf{F} = [\mathbf{e}_2]_\wedge \mathbf{Q}_2\mathbf{Q}_1^{-1}, \tag{56}$$

where $[\mathbf{e}_2]_\wedge$ is a skew-symmetric matrix acting as the external product[3] with $\mathbf{e}_2$. The fundamental matrix relates conjugate points; the role of left and right images is symmetrical, provided that we transpose $\mathbf{F}$:

$$\tilde{\mathbf{m}}_1^\top \mathbf{F}^\top \tilde{\mathbf{m}}_2 = 0. \tag{58}$$

Since $\det([\mathbf{e}_2]_\wedge) = 0$, the rank of $\mathbf{F}$ is in general 2. Moreover, $\mathbf{F}$ is defined up to a scale factor, because (55) is homogeneous. Hence it depends upon seven parameters. Indeed, it can be parameterized with the *epipolar transformation*, that is characterized by the projective coordinates of the epipoles (2×2) and by the three coefficients of the *homography* (see Appendix A) between the two pencils of epipolar lines [93].

3.5 Rectification

Given a pair of stereo images, *rectification* determines a transformation of each image plane such that pairs of conjugate epipolar lines become collinear and parallel

[3] It is well-known that the external product $\mathbf{t} \wedge \mathbf{x}$ can be written as a matrix vector product $[\mathbf{t}]_\wedge \mathbf{x}$, with

$$[\mathbf{t}]_\wedge = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{57}$$

to one of the image axes (usually the horizontal one). The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras. The important advantage of rectification is that computing correspondences is made simpler. Other rectification algorithm can be found in [5, 60, 123, 112].

When C_1 is in the focal plane of the right camera, the right epipole is at infinity, and the epipolar lines form a bundle of parallel lines in the right image. A very special case is when both epipoles are at infinity, that happens when the line $\mathsf{C}_1\mathsf{C}_2$ (the *baseline*) is contained in both focal planes, i.e., the retinal planes are parallel to the baseline (see Figure 13). Epipolar lines then form a bundle of parallel lines in both images. Any pair of images can be transformed so that epipolar lines are parallel and horizontal in each image. This procedure is called *rectification*.

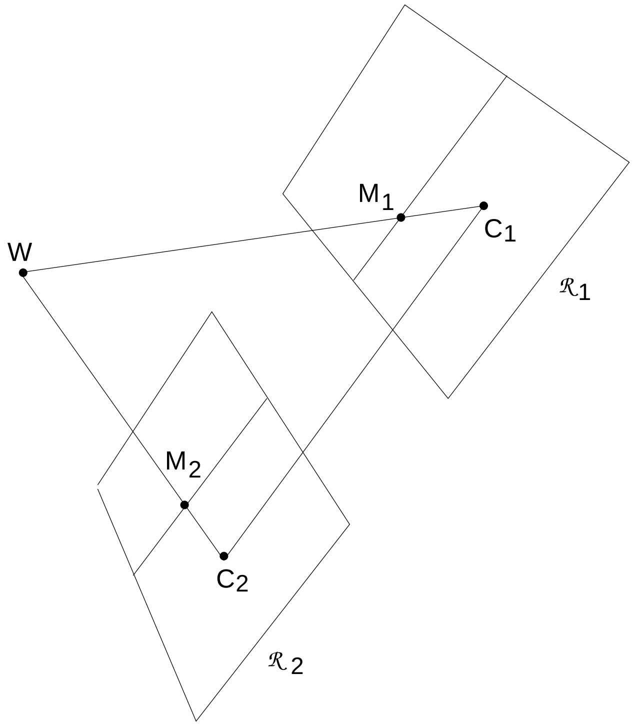

Figure 13: Rectified cameras. Image planes are coplanar and parallel to the baseline.

3.5.1 Rectification of camera matrices

We will assume that the stereo rig is *calibrated*, i.e., the old PPMs $\tilde{\mathbf{P}}_{o1}$ and $\tilde{\mathbf{P}}_{o2}$ are known. This assumption is not strictly necessary [60, 123], but leads to a simpler technique. The idea behind rectification is to define two new perspective matrices

$\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$, that preserve the optical centers and with the baseline contained in the focal planes. This ensures that epipoles are at infinity, hence epipolar lines are parallel. In addition, to have a proper rectification, it is required that epipolar lines are horizontal, and that corresponding points have the same vertical coordinate. We will formalize analytically this requirements in Section 3.5.4, where we also show that the algorithm given in the present section satisfies that requirements.

The new PPMs will have both the same orientation but different position. Positions (optical centers) are the same as the old cameras, whereas orientation changes because we rotate both cameras around the optical centers in such a way that focal planes becomes coplanar and contain the baseline.

In order to simplify the algorithm, the rectified PPMs will have also the same intrinsic parameters. The resulting PPMs will differ only in their optical centers. The new camera pair can be thought as a single camera translated along the X axis of its standard reference system. This intuitively satisfies the rectification requirements (formal proof in Section 3.5.4).

Let us think of the new PPMs in terms of their factorization. From (36) and (33):

$$\tilde{\mathbf{P}}_{n1} = \mathbf{A}[\mathbf{R} \mid -\mathbf{R}\ \mathbf{c}_1], \qquad \tilde{\mathbf{P}}_{n2} = \mathbf{A}[\mathbf{R} \mid -\mathbf{R}\ \mathbf{c}_2]. \tag{59}$$

The optical centers $\mathbf{c}_1$ and $\mathbf{c}_2$ are given by the old optical centers, computed with (32). The rotation matrix $\mathbf{R}$ is the same for both PPMs, and is computed as detailed below. The intrinsic parameters matrix $\mathbf{A}$ is also the same for both PPMs, but can be chosen arbitrarily (see MATLAB code, Figure 14). We will specify $\mathbf{R}$ by means of its row vectors

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix} \tag{60}$$

that are the X, Y and Z axes respectively, of the camera standard reference frame, expressed in world coordinates.

According to the previous geometric arguments, we take:

1. the new X axis parallel to the baseline: $\mathbf{r}_1 = (\mathbf{c}_1 - \mathbf{c}_2)/||\mathbf{c}_1 - \mathbf{c}_2||$

2. the new Y axis orthogonal to X (mandatory) and to $\mathbf{k}$: $\mathbf{r}_2 = \mathbf{k} \wedge \mathbf{r}_1$

3. the new Z axis orthogonal to XY (mandatory) : $\mathbf{r}_3 = \mathbf{r}_1 \wedge \mathbf{r}_2$

where **k** is an arbitrary unit vector, that fixes the position of the new Y axis in the plane orthogonal to X. We take it equal to the Z unit vector of the old left matrix, thereby constraining the new Y axis to be orthogonal to both the new X and the old left Z. The algorithm is given in more details in the MATLAB version, Figure 14.

3.5.2 The rectifying transformation

In order to rectify – let's say – the left image, we need to compute the transformation mapping the image plane of $\tilde{\mathbf{P}}_{o1} = [\mathbf{Q}_{o1}|\tilde{\mathbf{q}}_{o1}]$ onto the image plane of $\tilde{\mathbf{P}}_{n1} = [\mathbf{Q}_{n1}|\tilde{\mathbf{q}}_{n1}]$. We will see that the sought transformation is the collinearity given by the 3×3 matrix $\mathbf{T}_1 = \mathbf{Q}_{n1}\mathbf{Q}_{o1}^{-1}$. The same result will apply to the right image.

For any 3-D point **w** we can write

$$\begin{cases} \tilde{\mathbf{m}}_{o1} = \tilde{\mathbf{P}}_{o1}\tilde{\mathbf{w}} \\ \tilde{\mathbf{m}}_{n1} = \tilde{\mathbf{P}}_{n1}\tilde{\mathbf{w}}. \end{cases} \tag{61}$$

According to (35) , the equations of the optical rays are the following (since rectification does not move the optical center)

$$\begin{cases} \mathbf{w} = \mathbf{c}_1 + \lambda_o \mathbf{Q}_{o1}^{-1}\tilde{\mathbf{m}}_{o1} \\ \mathbf{w} = \mathbf{c}_1 + \lambda_n \mathbf{Q}_{n1}^{-1}\tilde{\mathbf{m}}_{n1}; \end{cases} \tag{62}$$

Hence:

$$\tilde{\mathbf{m}}_{n1} = \lambda \mathbf{Q}_{n1}\mathbf{Q}_{o1}^{-1}\tilde{\mathbf{m}}_{o1}. \tag{63}$$

where λ is an arbitrary scale factor (it is an equality between homogeneous quantities). This is a clearer and more compact result than the one reported in [5].

The transformation $\mathbf{T}_1$ is then applied to the original left image to produce the rectified image, as in Figure 17. Note that the pixels (integer-coordinate positions) of the rectified image correspond, in general, to non-integer positions on the original image plane. Therefore, the gray levels of the rectified image are computed by bilinear interpolation.

```
function [T1,T2,Pn1,Pn2] = rectify(Po1,Po2)

% RECTIFY: compute  rectification matrices in homogeneous coordinate
%
%         [T1,T2,Pn1,Pn2] = rectify(Po1,Po2)  computes the rectified
%         projection matrices "Pn1" and "Pn2", and the transformation
%         of the retinal plane "T1" and  "T2" (in homogeneous coord.)
%         which perform rectification.  The arguments are the two old
%         projection  matrices "Po1" and "Po2".

%         Andrea Fusiello, MVL 1998 (fusiello@dimi.uniud.it)


% factorize old PPMs
[A1,R1,t1] = art(Po1);
[A2,R2,t2] = art(Po1);

% optical centers (unchanged)
c1 = - inv(Po1(:,1:3))*Po1(:,4);
c2 = - inv(Po2(:,1:3))*Po2(:,4);

% new x axis (= direction of the baseline)
v1 = (c1-c2);
% new y axes (orthogonal to new x and old z)
v2 = extp(R1(3,:)',v1);
% new z axes (no choice, orthogonal to baseline and y)
v3 = extp(v1,v2);

% new extrinsic parameters (translation unchanged)
R = [v1'/norm(v1)
     v2'/norm(v2)
     v3'/norm(v3)];

% new intrinsic parameters (arbitrary)
A = (A1 + A2)./2;
A(1,2)=0; % no skew

% new projection matrices
Pn1 = A * [R -R*c1 ];
Pn2 = A * [R -R*c2 ];

% rectifying image transformation
T1 = Pn1(1:3,1:3)* inv(Po1(1:3,1:3));
T2 = Pn2(1:3,1:3)* inv(Po2(1:3,1:3));

------------------------

function [A,R,t] = art(P)
% ART: factorize a PPM as  P=A*[R;t]

Q = inv(P(1:3, 1:3));
[U,B] = qr(Q);

R = inv(U);
t = B*P(1:3,4);
A = inv(B);
A = A ./A(3,3);
```

Figure 14: Working MATLAB code of the `rectify` function.

3.5.3 Summary of the Rectification algorithm

The Rectification algorithm can be summarized as follows:

- Given a stereo pair of images `I1,I2` and PPMs `Po1,Po2` (obtained by calibration);
- compute `[T1,T2,Pn1,Pn2] = rectify(Po1,Po2)` (see box);
- rectify images by applying `T1` and `T2`.

Reconstruction of 3-D position can be performed from the rectified images directly, using `Pn1,Pn2`.
The code of the algorithm, shown in Figure 14 is simple and compact, and the comments enclosed make it understandable without knowledge of matlab.

3.5.4 Rectification analysis

In this section we will (i) formulate analytically the rectification requirements, and (ii) prove that our algorithm yields PPMs $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ that satisfies such requirements.

Definition 3.1
A pair of PPMs $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ are said to be *rectified* if, for any point $\mathbf{m}_1 = (u_1, v_1)^\top$ in the left image, its epipolar line in the right image has equation $v_2 = v_1$, and, for any point $\mathbf{m}_2 = (u_2, v_2)^\top$ in the right image, its epipolar line in the left image has equation $v_1 = v_2$.

In the following, we shall write $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ as follows:

$$\tilde{\mathbf{P}}_{n1} = \left[\begin{array}{c|c} \mathbf{s}_1^\top & s_{14} \\ \mathbf{s}_2^\top & s_{24} \\ \mathbf{s}_3^\top & s_{34} \end{array}\right] = [\mathbf{S}|\tilde{\mathbf{s}}] \qquad \tilde{\mathbf{P}}_{n2} = \left[\begin{array}{c|c} \mathbf{d}_1^\top & d_{14} \\ \mathbf{d}_2^\top & d_{24} \\ \mathbf{d}_3^\top & d_{34} \end{array}\right] = [\mathbf{D}|\tilde{\mathbf{d}}]. \tag{64}$$

Proposition 3.2

Two perspective projection matrices $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ are rectified if and only if

$$\begin{cases} \mathbf{s}_1\mathbf{c}_2 + s_{14} \neq 0 \\ \mathbf{s}_2\mathbf{c}_2 + s_{24} = 0 \\ \mathbf{s}_3\mathbf{c}_2 + s_{34} = 0 \end{cases} \text{and} \begin{cases} \mathbf{d}_1\mathbf{c}_1 + d_{14} \neq 0 \\ \mathbf{d}_2\mathbf{c}_1 + d_{24} = 0 \\ \mathbf{d}_3\mathbf{c}_1 + d_{34} = 0 \end{cases} \tag{65}$$

and

$$\frac{\mathbf{s}_2\mathbf{w} + s_{24}}{\mathbf{s}_3\mathbf{w} + s_{34}} = \frac{\mathbf{d}_2\mathbf{w} + d_{24}}{\mathbf{d}_3\mathbf{w} + d_{34}}, \tag{66}$$

where $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ are written as in (64) and $\mathbf{c}_1$ and $\mathbf{c}_2$ are the respective optical center's coordinates.

Proof As we know, the epipolar line of $\tilde{\mathbf{m}}_2$ is the projection of its optical ray onto the left camera, hence its parametric equation writes:

$$\tilde{\mathbf{m}}_1 = \tilde{\mathbf{P}}_{n1}\begin{bmatrix}\mathbf{c}_2 \\ 1\end{bmatrix} + \tilde{\mathbf{P}}_{n1}\begin{bmatrix}\lambda\mathbf{D}^{-1}\tilde{\mathbf{m}}_2 \\ 0\end{bmatrix} = \tilde{\mathbf{e}}_1 + \lambda\mathbf{S}\mathbf{D}^{-1}\tilde{\mathbf{m}}_2 \tag{67}$$

where $\tilde{\mathbf{e}}_1$, the epipole, is the projection of the conjugate optical center $\mathbf{c}_2$: [4]

$$\tilde{\mathbf{e}}_1 = \tilde{\mathbf{P}}_{n1}\begin{bmatrix}\mathbf{c}_2 \\ 1\end{bmatrix} = \begin{bmatrix}\mathbf{s}_1\mathbf{c}_2 + s_{14} \\ \mathbf{s}_2\mathbf{c}_2 + s_{24} \\ \mathbf{s}_3\mathbf{c}_2 + s_{34}\end{bmatrix}. \tag{68}$$

The parametric equation of the epipolar line of $\tilde{\mathbf{m}}_2$ in image coordinates becomes:

$$\begin{cases} u = [\mathbf{m}_1]_1 = \dfrac{[\tilde{\mathbf{e}}_1]_1 + \lambda[\tilde{\mathbf{n}}]_1}{[\tilde{\mathbf{e}}1]_3 + \lambda[\tilde{\mathbf{n}}]_3} \\ v = [\mathbf{m}_1]_2 = \dfrac{[\tilde{\mathbf{e}}_1]_2 + \lambda[\tilde{\mathbf{n}}]_2}{[\tilde{\mathbf{e}}_1]_3 + \lambda[\tilde{\mathbf{n}}]_3} \end{cases} \tag{69}$$

where $\tilde{\mathbf{n}} = \mathbf{S}\mathbf{D}^{-1}\tilde{\mathbf{m}}_2$, and $[.]_i$ is the projection operator extracting the ith component from a vector.

Analytically, the direction of each epipolar line can be obtained by taking the derivative of the parametric equations (69) with respect to λ:

[4] In this section only, to improve readability, we omit the transpose sign in scalar products. All vector products are scalar products, unless otherwise noted.

$$\begin{cases} \dfrac{du}{d\lambda} = \dfrac{[\tilde{\mathbf{n}}]_1[\tilde{\mathbf{e}}_1]_3 - [\tilde{\mathbf{n}}]_3[\tilde{\mathbf{e}}_1]_1}{([\tilde{\mathbf{e}}_1]_3 + \lambda[\tilde{\mathbf{n}}]_3)^2} \\ \dfrac{dv}{d\lambda} = \dfrac{[\tilde{\mathbf{n}}]_2[\tilde{\mathbf{e}}_1]_3 - [\tilde{\mathbf{n}}]_3[\tilde{\mathbf{e}}_1]_2}{([\tilde{\mathbf{e}}_1]_3 + \lambda[\tilde{\mathbf{n}}]_3)^2} \end{cases}. \tag{70}$$

Note that the denominator is the same in both components, hence it does not affect the direction of the vector. The epipole is rejected to infinity when $[\tilde{\mathbf{e}}_1]_3 = 0$. In this case, the direction of the epipolar lines in the right image doesn't depend on $\mathbf{n}$ and all the epipolar lines becomes parallel to vector $[[\tilde{\mathbf{e}}_1]_1 \; [\tilde{\mathbf{e}}_1]_2]^\top$. The same holds, *mutatis mutandis*, for the left image.

Hence, epipolar lines are horizontal if and only if (65) holds. The vertical coordinate of conjugate points is the same in both image if and only if (66) holds, as can easily seen by plugging (64) into (30). □

PROPOSITION 3.3

The two camera matrices $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ produced by the RECTIFICATION algorithm are rectified.

Proof We shall prove that, if $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ are built according to the RECTIFICATION algorithm, then (65) and (66) hold.

From (59) we obtain

$$\begin{array}{lll} s_{14} = -\mathbf{s}_1\mathbf{c}_1 & d_{14} = -\mathbf{d}_1\mathbf{c}_2 & \mathbf{s}_1 = \mathbf{d}_1 \\ s_{24} = -\mathbf{s}_2\mathbf{c}_1 & d_{24} = -\mathbf{d}_2\mathbf{c}_2 & \mathbf{s}_2 = \mathbf{d}_2 \\ s_{34} = -\mathbf{s}_3\mathbf{c}_1 & d_{34} = -\mathbf{d}_3\mathbf{c}_2 & \mathbf{s}_3 = \mathbf{d}_3 \end{array} \tag{71}$$

From the factorization (36), assuming $\gamma = 0$, we obtain

$$\begin{bmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \\ \mathbf{s}_3^\top \end{bmatrix} = \mathbf{A}\mathbf{R} = \begin{bmatrix} \alpha_u \mathbf{r}_1^\top + u_0\mathbf{r}_3^\top \\ \alpha_v \mathbf{r}_2^\top + v_0\mathbf{r}_3^\top \\ \mathbf{r}_3^\top \end{bmatrix} \tag{72}$$

From the construction of $\mathbf{R}$, we have that $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ are mutually orthogonal and $\mathbf{r}_1 = \beta(\mathbf{c}_1 - \mathbf{c}_2)$ with $\beta = 1/||\mathbf{c}_1 - \mathbf{c}_2||$.

From all these facts, the following four identity are derived:

$$\begin{aligned}
\mathbf{s}_1(\mathbf{c}_1 - \mathbf{c}_2) &= \beta \mathbf{s}_1 \mathbf{r}_1 = \beta(\alpha_u \mathbf{r}_1 + u_0 \mathbf{r}_3)\mathbf{r}_1 = \beta(\alpha_u \mathbf{r}_1 \mathbf{r}_1 + u_0 \mathbf{r}_3 \mathbf{r}_1) = \beta \alpha_u \neq 0 && (73)\\
\mathbf{s}_2(\mathbf{c}_1 - \mathbf{c}_2) &= \beta \mathbf{s}_2 \mathbf{r}_1 = \beta(\alpha_v \mathbf{r}_2 + v_0 \mathbf{r}_3)\mathbf{r}_1 = \beta(\alpha_v \mathbf{r}_2 \mathbf{r}_1 + v_0 \mathbf{r}_3 \mathbf{r}_1) = 0 && (74)\\
\mathbf{s}_3(\mathbf{c}_1 - \mathbf{c}_2) &= \beta \mathbf{s}_3 \mathbf{r}_1 = \beta \mathbf{r}_3 \mathbf{r}_1 = 0 && (75)\\
\mathbf{s}_2 \wedge \mathbf{s}_3 &= \mathbf{s}_2 \wedge \mathbf{r}_3 = \lambda(\mathbf{r}_2 \wedge \mathbf{r}_3) = \lambda \mathbf{r}_1 && (76)
\end{aligned}$$

The parameter λ in (76) is scalar taking into account that $\mathbf{s}_2$ is a linear combination of $\mathbf{r}_2$ and $\mathbf{r}_3$.

Equation (65) follows easily from (73) (74)(75). Equation (66) is equivalent to

$$(\mathbf{s}_2 \mathbf{w} + s_{24})(\mathbf{d}_3 \mathbf{w} + d_{34}) = (\mathbf{s}_3 \mathbf{w} + s_{34})(\mathbf{d}_2 \mathbf{w} + d_{24}). \qquad (77)$$

Expanding, and using (74),(76) and properties of the external product we obtain

$$\begin{aligned}
-\mathbf{s}_2(\mathbf{c}_1 - \mathbf{c}_2)\mathbf{s}_3 \mathbf{w} + (\mathbf{s}_2 \mathbf{c}_1)(\mathbf{s}_3 \mathbf{c}_2) - (\mathbf{s}_2 \mathbf{c}_2)(\mathbf{s}_3 \mathbf{c}_1) &= \\
(\mathbf{s}_2 \mathbf{c}_1)(\mathbf{s}_3 \mathbf{c}_2) - (\mathbf{s}_2 \mathbf{c}_2)(\mathbf{s}_3 \mathbf{c}_1) &= \\
(\mathbf{s}_2 \wedge \mathbf{s}_3)(\mathbf{c}_1 \wedge \mathbf{c}_2) &= \qquad (78)\\
\lambda \mathbf{r}_1(\mathbf{c}_1 \wedge \mathbf{c}_2) &= \\
\lambda \beta(\mathbf{c}_1 - \mathbf{c}_2)(\mathbf{c}_1 \wedge \mathbf{c}_2) &= 0.
\end{aligned}$$

□

3.5.5 Experimental results

We ran tests to verify that the algorithm performed rectification correctly, and also to check that the accuracy of the 3-D reconstruction did not decrease when performed from the rectified images directly.

Correctness

The tests used both synthetic and real data. Each set of synthetic data consisted of a cloud of 3-D points and a pair of PPMs. For reasons of space, we report only two examples. Figure 16 shows the original and rectified images with a nearly rectified stereo rig: the camera translation was $-[100\ 2\ 3]$ mm and the rotation angles roll=1.5°, pitch=2°, yaw=1°. Figure 15 shows the same with a more general

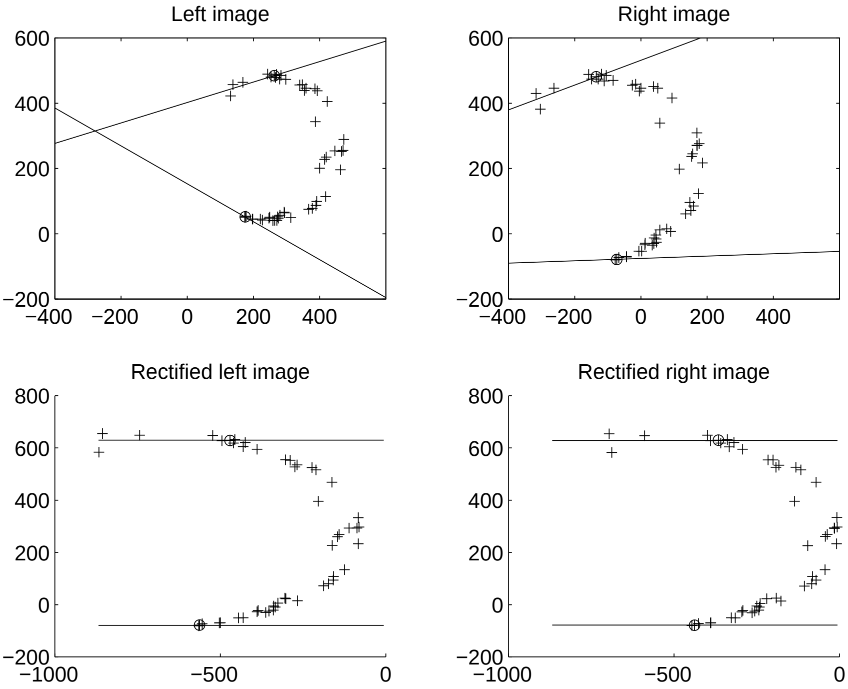

Figure 15: General synthetic stereo pair (top) and rectified pair (bottom). The figure shows the epipolar lines of the points marked with a circle in both images.

geometry: the camera translation was −[100 20 30] mm and the rotation angles roll=19° pitch=32° and yaw=5°.

Real-data experiments used calibrated stereo pairs, courtesy of INRIA-Syntim. We show the results obtained with a nearly rectified stereo rig (Figure 17) and with a more general stereo geometry (Figure 18). The right image of each pair shows three epipolar lines corresponding to the points marked by a cross in the left image. The pixel coordinates of the rectified images are not constrained to lie in any special part of the image plane, and an arbitrary translation were applied to both images to bring them in a suitable region of the plane; then the output images were cropped to the size of the input images. In the case of the "Sport" stereo pair (image size 768 × 576), we started from the following camera matrices:

$$\mathbf{P}_{o1} = \begin{bmatrix} 9.7655352e+02 & 5.3829220e+01 & -2.3984731e+02 & 3.8754954e+05 \\ 9.8498581e+01 & 9.3334472e+02 & 1.5747888e+02 & 2.4287923e+05 \\ 5.7902862e-01 & 1.1085118e-01 & 8.0773700e-01 & 1.1185149e+03 \end{bmatrix}$$

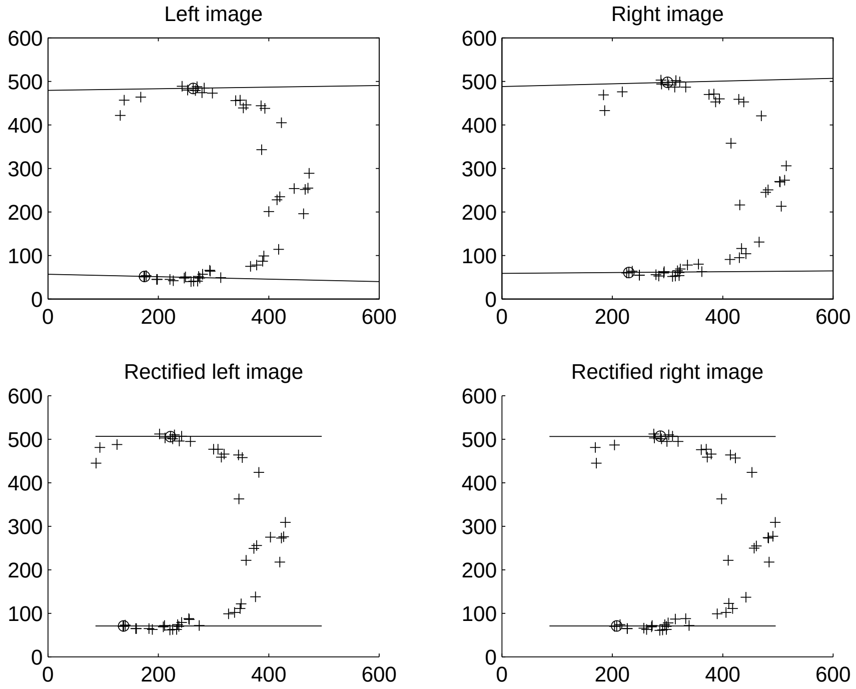

Figure 16: Nearly rectified synthetic stereo pair (top) and rectified pair (bottom). The figure shows the epipolar lines of the points marked with a circle in both images.

$$\mathbf{P}_{o2} = \begin{bmatrix} 9.7670272e+02 & 5.3761100e+01 & -2.4002435e+02 & 4.0034922e+04 \\ 9.8682765e+01 & 9.3104118e+02 & 1.5678255e+02 & 2.5173864e+05 \\ 5.7665530e-01 & 1.1413953e-01 & 8.0897550e-01 & 1.1743716e+03 \end{bmatrix}.$$

After adding the statement `A(1,3) = A(1,3) + 160` to the `rectify` program, to keep the rectified image in the center of the 768 × 576 window, we obtained the following rectified camera matrices:

$$\mathbf{P}_{n1} = \begin{bmatrix} 1.0431495e+03 & 7.4525523e+01 & -2.5850412e+02 & 4.1246428e+05 \\ 1.1652788e+02 & 9.3389317e+02 & 1.4105910e+02 & 2.3883586e+05 \\ 6.8550713e-01 & 1.1391110e-01 & 7.1909960e-01 & 1.1024013e+03 \end{bmatrix}$$

$$\mathbf{P}_{n2} = \begin{bmatrix} 1.0431495e+03 & 7.4525523e+01 & -2.5850412e+02 & 4.0698457e+04 \\ 1.1652788e+02 & 9.3389317e+02 & 1.4105910e+02 & 2.3883586e+05 \\ 6.8550713e-01 & 1.1391110e-01 & 7.1909960e-01 & 1.1024013e+03 \end{bmatrix}.$$

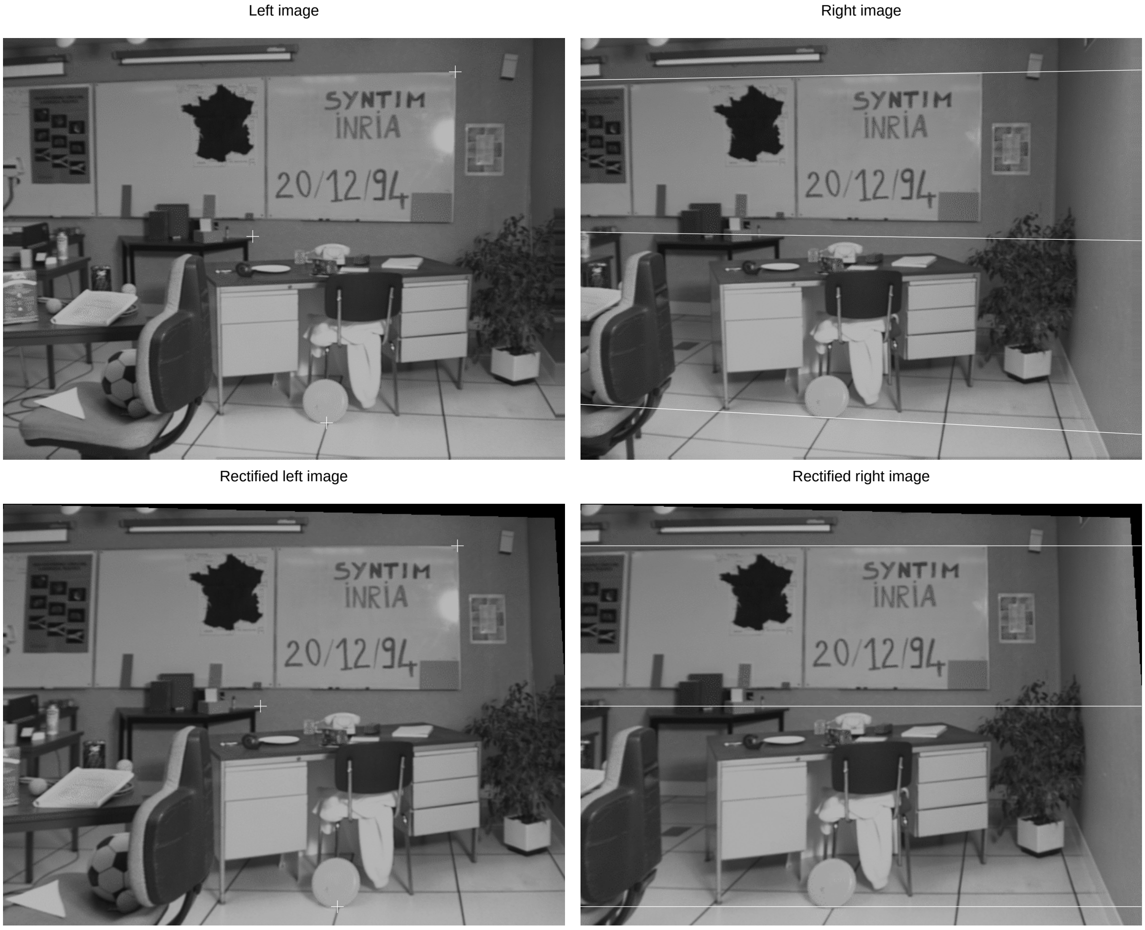

Figure 17: "Sport" stereo pair (top) and rectified pair (bottom). The right pictures plot the epipolar lines corresponding to the points marked in the left pictures.

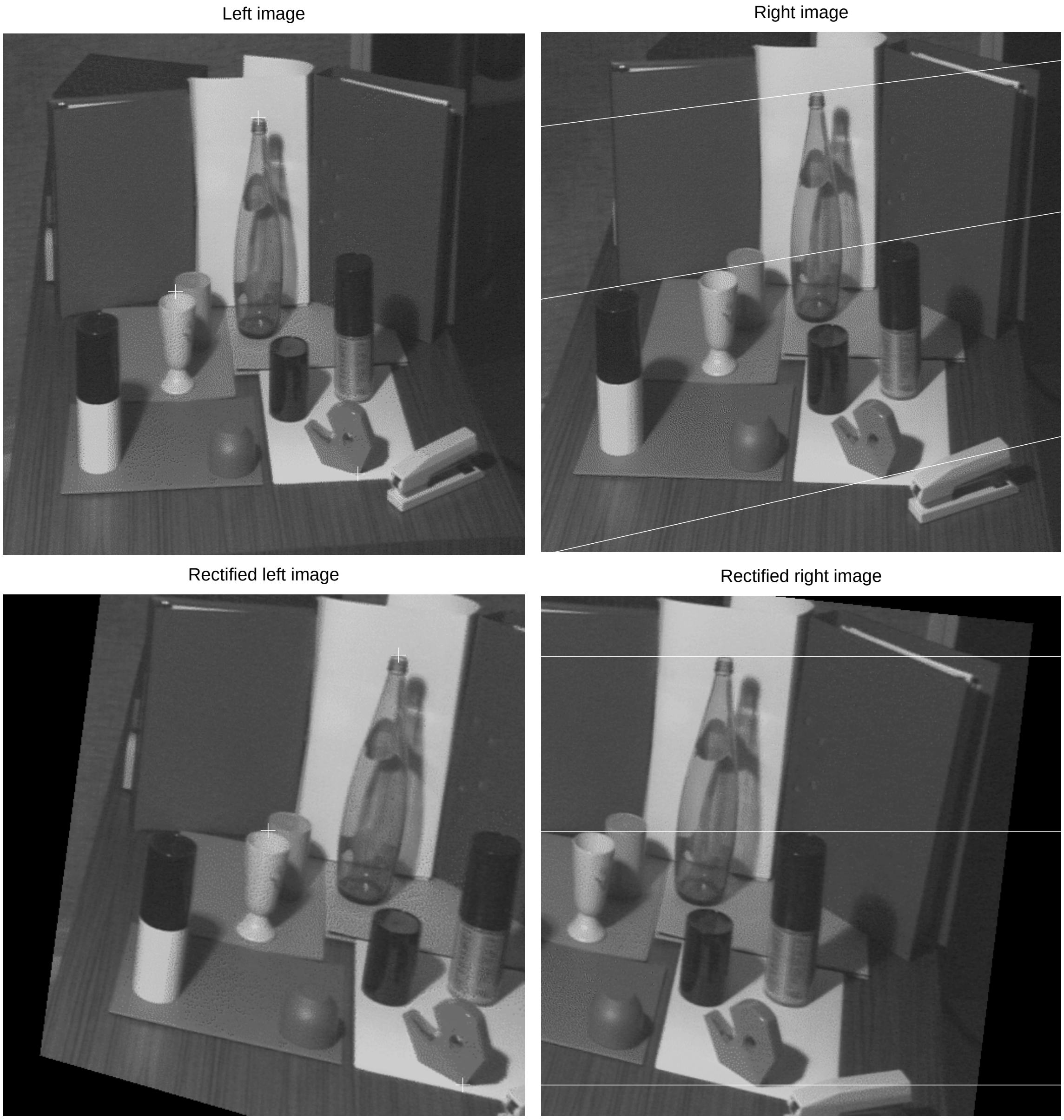

Figure 18: "Color" stereo pair (top) and rectified pair (bottom). The right pictures plot the epipolar lines corresponding to the points marked in the left pictures.

Accuracy

In order to evaluate the errors introduced by rectification on reconstruction, we compared the accuracy of 3-D reconstruction computed from original and rectified images. We used synthetic, noisy images of random clouds of 3-D points. Imaging errors were simulated by perturbing the image coordinates, and calibration errors by perturbing the intrinsic and extrinsic parameters, both with additive, Gaussian noise. Reconstruction were performed using the Linear-Eigen method, described in Section 3.3.

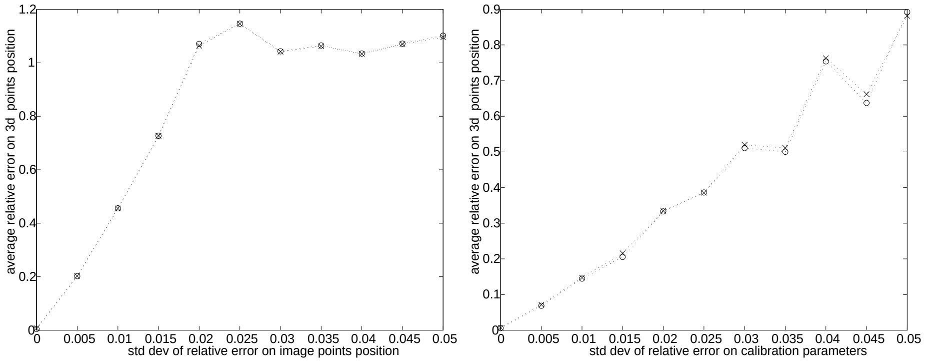

Figure 19: Reconstruction error vs noise levels in the image coordinates (left) and calibration parameters (right) for the general synthetic stereo pair. Crosses refer to reconstruction from rectified images, circles to reconstruction from unrectified images.

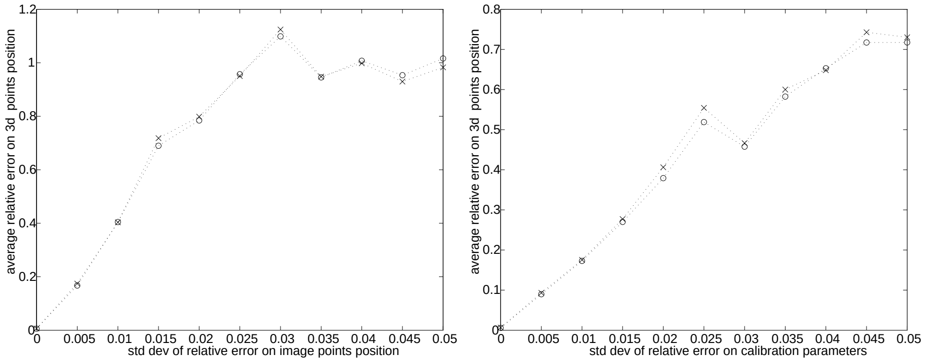

Figure 20: Reconstruction error vs noise levels in the image coordinates (left) and calibration parameters (right) for the nearly rectified synthetic stereo pair. Crosses refer to reconstruction from rectified images, circles to reconstruction from unrectified images.

Figures 19 and 20 show the average (over the set of points) relative error measured on 3-D point position, plotted against noise. Figure 19 shows the results for the stereo rig used in Figure 15, and Figure 20 for the one used in Figure 16. Each point plotted is an average over 100 independent trials. The abscissa is the standard deviation of the relative error on coordinates of image point or calibration parameters.

3.6 Conclusions

Given two images and a set of matched points, the 3-D coordinates of the corresponding world points can be reconstructed with a simple linear technique, if camera parameters (intrinsic and extrinsic) are known. The process of measuring camera parameters is called calibration. Epipolar geometry relates a point in one image with the set of possible matches in the other, which constitutes a line, called epipolar line. Matching is greatly simplified if the epipolar lines are parallel and horizontal in each image, i.e., if the images are rectified. In this chapter we have developed a simple and compact rectification algorithm. The correct behavior of the algorithm has been demonstrated with both synthetic and real images. Interestingly enough, reconstruction can be performed directly from the disparities of the rectified images, using the rectifying PPMs. Our tests show that this process does not introduces appreciable errors compared with reconstructing from the original images.

Chapter 4

Stereo Matching

In the previous chapter we assumed that we could identify *conjugate pairs*, that is to say, pairs of points in the two images that are projection of the same points in the scene. In this chapter we will address the problem of detecting conjugate pairs in stereo images. We propose a novel stereo matching algorithm, called SMW (Symmetric Multi-Window) addressing robust disparity estimation in the presence of occlusions. The algorithm is an adaptive, multi-window scheme using left-right consistency to compute disparity and its associated uncertainty. We demonstrate and discuss performances with both synthetic and real stereo pairs, and show how our results improve on those of closely related techniques for both accuracy and efficiency.

4.1 Introduction

Detecting conjugate pairs in stereo images is a challenging problem known as the *correspondence problem*, i.e., finding which points in the left and right images are projections of the same scene point (a *conjugate pair*).

Several factors make the correspondence problem difficult: (i) its inherent *ambiguity*, which requires the introduction of physical and geometric constraints; (ii)*occlusions*; (iii) *photometric distortions* and (iv) *figural distortion*. In Section 4.2 these factors are described, and the available constraints are introduced. Then, the existing methods are outlined.

In Section 4.3 we present a new Symmetric, Multi-Window algorithm (henceforth

SMW) for stereo matching, which addresses the problem mentioned in Section 4.1, and outperforms closely related methods. SMW's assumptions are clearly stated in Section 4.3.1. SMW is based on the Block Matching algorithm (Section 4.3.2); it employs an adaptive, multi-window scheme to cure distortions and yield accurate disparities (Section 4.3.3), associated to uncertainty estimates. Robustness in the presence of occlusions is achieved thanks to the *left-right consistency* constraint (Section 4.3.4). A consistent uncertainty estimation mechanism (Section 4.3.5) guarantees that the depth maps produced can be used by data fusion schemes like [148]. In Section 4.3.6 we give a pseudo-code summary of the SMW algorithm. A detailed experimental evaluation, including a comparison with similar methods reported in the literature, is reported in Section 4.4. Our results (stereo pairs and disparity maps) are available on the web (http://www.dimi.uniud.it/˜fusiello/demo-smw/smw.html) where the source code for the SMW algorithm can be downloaded as well.

4.2 The correspondence problem

The *correspondence problem* (or *matching problem*) can be regarded as a search problem, since for each element on the left image (a point, region, or generic feature), a similar element is to be found in the right one, according to a given similarity measure. The output of a stereo matching algorithm is a set of correspondences, or a *disparity map* that gives the disparity for some or all points of a reference image. To prevent ambiguous or false matches and avoid combinatorial explosion, the search space must be suitably constrained. Geometric, physical and photometric constraints imposed by both the observer (our stereo rig) and the scene, include the following.

Similarity constraint [51]. Left and right images of a given scene element are similar. This is often implicit.

Epipolar constraint (see Chapter 3). Given a point in the left image, the corresponding point must lie on a straight line (called *epipolar line*) in the right image. This constraint reduces the search space from two-dimensional to one-dimensional. It applies in every situation, provided that the epipolar geometry is known.

Smoothness constraint [96]. The distance of scene points from the cameras changes smoothly almost everywhere, thereby limiting the allowable disparity gradient. This fails, obviously, at depth discontinuities.

Uniqueness constraint [96]. Each image element has one and only one conjugate. This fails if transparent objects are presents or in the presence of occlusions.

Ordering constraint [7]. If point $\mathbf{m}_1$ in the one image matches point $\mathbf{m}'_1$ in the other image, then the corresponding of a point $\mathbf{m}_2$ that lies at the right (left) of $\mathbf{m}_1$ must lie at the right (left) of $\mathbf{m}'_1$. This constraint hold for points belonging on the surface of an opaque object. It fails at region known as *forbidden zone* (See Figure 21).

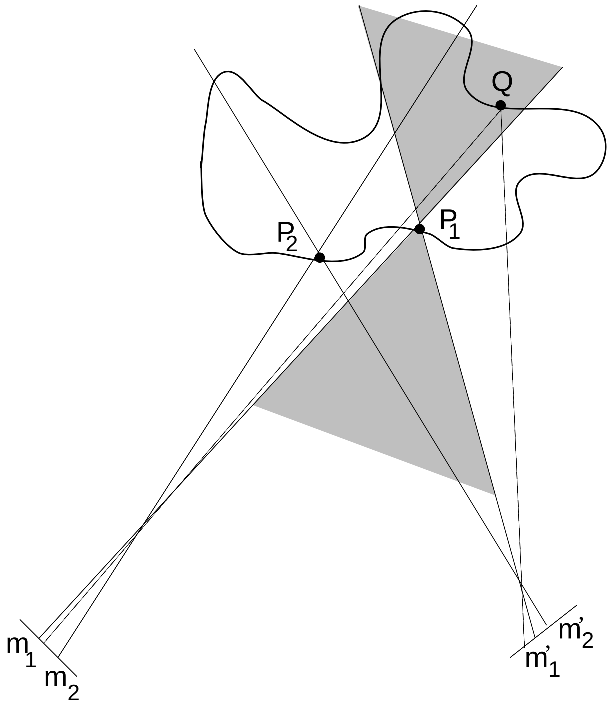

Figure 21: Ordering constraint. Point Q , which lies behind an opaque object, violates the ordering constraint. The shaded region is the forbidden zone of P_1.

Major problems affecting machine stereo arise because the scene is viewed from two different viewpoints, which is also the key feature of stereo. The larger the baseline the more severe these effects, which include the following.

Occlusions. Since the two images of the scene are slightly different, there are elements that are imaged only in one camera. Hence, there are image points without a corresponding, or, stated in other words, not all points in one image belongs to a conjugate pair.

Photometric distortion. A typical assumption is that the perceived intensity of a surface patch does not depend on the viewing direction: light source is a point at infinity and the surfaces are *Lambertian* (see Chapter 2). This is not true in general, and the same world point takes different intensities in each view.

Figural distortion. Owing to perspective projection, the same object appears different when projected in the left and right images.

4.2.1 Matching techniques

The techniques adopted for the stereo matching problem can be classified along two dimensions: the kind of image element considered for matching *(What to match)*, and the techniques to compute matching *(How to match)*. In addition, one can be interested in the computational schemes adopted, especially when biological plausibility is of concern [15] .

What to match

Let us address the first issue. Some algorithms [26, 96] match individual pixels, i.e., the atomic elements in an image. More robust methods, called *area-based*, perform matching between gray levels of image patches (*windows*), by computing some form of similarity or correlation. The disparity may then be computed for every pixel [35, 41], for the centers of the windows [90, 162], or for selected points of interest [56].
Since gray-levels are not identical in the two image, some problems arise with matching raw intensities. They can be overcome by considering the output of a bandpass filter, usually a Laplacian of Gaussian (LoG) filter [108]. One could also compute the response of a bank of filters at a given image point, which defines a vector characterizing the local structure of the image [78, 157]. A similar vector is estimated

on the other image, in order to compute matching.

Matching image features is generally more robust; the related class of algorithms is called *feature-based*. In the present context, the term “features” indicates physically meaningful cues, such as edges [97, 52, 116, 7, 110], segments (collinear connected edges) [102], and corners (where two edges cross) [8]. Features can be extracted by bandpass filters, derivative operators or *ad hoc* non-linear operators.

The local phase of the image signal – computed via Fourier or Gabor transforms – has also been used for matching [77, 76, 67]. As disparity should be less than one pixel to avoid aliasing (according to the sampling theorem, or the “quarter cycle limit” [96]), a multi-resolution scheme should be employed.

How to match

We now come to the second question: Once chosen the elements to be matched, how to perform matching?

Correlation techniques consist in finding the amount of shifting that yields the maximum similarity score between the left and the right elements. Although several correlation measures have been proposed, the *Sum of Squared Differences* (SSD) measure is regarded as a reasonable choice [2, 41, 35, 111, 74, 79, 47]. Recently,a new approach based the local ordering of intensities have been presented [163] with promising results.

With *relaxation-based methods* the elements are joined by weighted links; the initial weights are iteratively updated by propagating constraints, until some equilibrium configuration is reached [96, 97, 52, 116, 8].

Dynamic programming techniques adopt a cost function, that embeds the constraints and is minimized to get the best set of matches [110, 7, 74, 47, 11, 26]. The solution is a curve in the match space [26, 47] or the disparity space [74]. Usually, the cost functional is derived using Bayesian reasoning [47, 11, 26].

A novel approach to matching consists in representing image scan lines by means of *intrinsic curves* [139], i.e, the paths followed by a descriptor vector as the scan line is traversed from left to right. Intrinsic curves are invariant to image displacements, and this property is exploited to compute matching.

Computational schemes

As far as the computational scheme is concerned, algorithms can be classified into *cooperative*, *coarse-to-fine* and *feed-forward* (see [15] for more details).

Cooperative models, pioneered by Marr and Poggio [96], exploit the properties of recurrent nets, which perform relaxation to a minimum energy configuration.

In *coarse-to-fine models*, the disparities computed at different spatial scales are fused to compute the final disparity estimate. In biological vision, coarse-to-fine models identify a special class of algorithms using multiple spatial filters that simulate receptive fields [97, 108]. In machine vision, this paradigm is applicable to any scheme, in order to get scale independence and data redundancy [84]. It is mandatory only with phase-based methods.

Whereas the cooperative and the coarse-to-fine techniques require cooperative feedback or sequential disparity processing over the spatial scales, the *feed-forward scheme* [162] operates in one shot, like most of the machine stereo algorithms.

For further details on machine stereo, the reader can consult the book [53] or the surveys in [16, 30]; a review on human computational stereo is given in [15].

4.3 A new area-based stereo algorithm

In this section we present our new, efficient stereo algorithm addressing robust disparity estimation in the presence of occlusions. The algorithm is an adaptive, multiwindow scheme using left-right consistency to compute disparity and its associated uncertainty.

4.3.1 Assumptions

With no loss of generality, we assume that conjugate pairs lie along raster lines, that is, the stereo pair has been *rectified* (Section 3.5) to achieve parallel and horizontal epipolar lines in each image.

We also assume that the image intensities $I(x, y)$ of corresponding points in the two images are the same. If this is not true, the images can be *normalized* by a simple algorithm [26] that computes the parameters α, β of the gray-level *global* transformation

$$I_l(x, y) = \alpha I_r(x, y) + \beta \qquad \forall(x, y)$$

by fitting a straight line to the plot of the left cumulative histogram versus the right cumulative histogram. This normalization fails if images are taken from viewpoints too far apart.

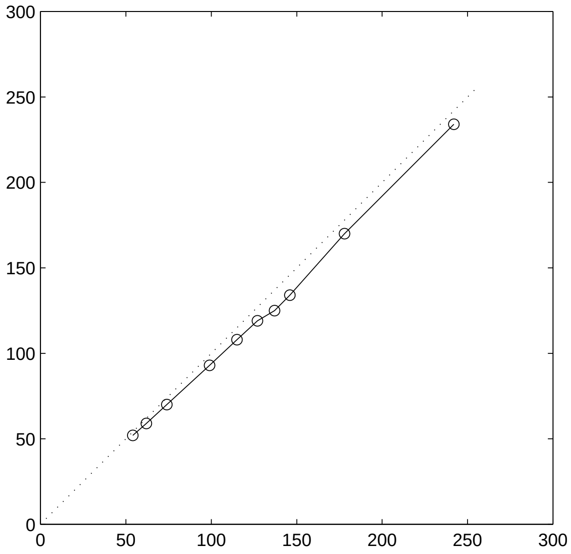

Figure 22: Ten percentile points from “Shrub” histograms.

4.3.2 The Block Matching algorithm

The basic structure of the *block matching* algorithm can be outlined as follows. For each pixel in the image chosen as *reference* (e.g., the left one, I_l), similarity scores are computed by comparing a fixed, small window centered on the pixel to a window in the other image (here, I_r), shifting along the raster line. Windows are compared through the *normalized SSD* measure, that quantifies the difference between intensity patterns:

$$C(x, y, d) = \frac{\sum_{(\xi,\eta)} [I_l(x+\xi, y+\eta) - I_r(x+\xi + d, y+\eta)]^2}{\sqrt{\sum_{(\xi,\eta)} I_l(x+\xi, y+\eta)^2 \sum_{(\xi,\eta)} I_r(x+\xi+d, y+\eta)^2}} \tag{79}$$

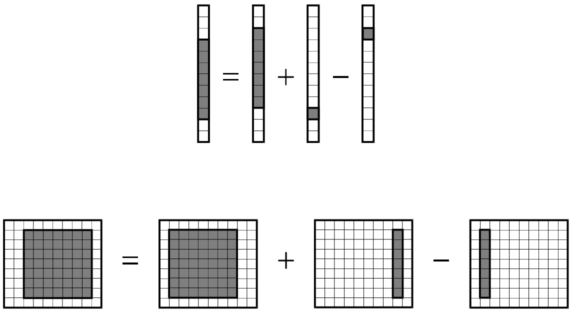

Figure 23: Efficient implementation of correlation.

where $\xi \in [-n, n]$, $\eta \in [-m, m]$. The disparity estimate for pixel (x, y) is the one that minimizes the SSD error:

$$d_o(x, y) = \arg\min_d C(x, y, d). \tag{80}$$

Sub-pixel accuracy can be achieved, for instance, by fitting a parabola to the SSD error function $C(d)$ in the neighborhood of the minimum d_0 [2]:

$$s(x, y) = \frac{1}{2} \frac{C(x, y, d_o{-}1) - C(x, y, d_o{+}1)}{C(x, y, d_o{-}1){-}2C(x, y, d_o){+}C(x, y, d_o{+}1)} \tag{81}$$

The Simple Block Matching (henceforth SBM) algorithm is reported here.

Algorithm 1 SBM

```
let $I_r$, $I_l$ the right and left $N \times N$ images;
let $W$ a $n \times n$ window (with $n \ll N$);
for each pixel $I_l(x, y)$
    for each disparity $\mathbf{d} = (d_x, d_y)$ in some range
        $C(x, y, \mathbf{d}) = \sum_{(\xi,\eta)\in W} [I_l(x + \xi, y + \eta) - I_r(x + \xi - d_x, y + \eta - d_y)]^2$;
    end
    $\mathbf{d}_l(x, y) \leftarrow \arg\min_{\mathbf{d}} C(x, y, \mathbf{d})$
end
```

end

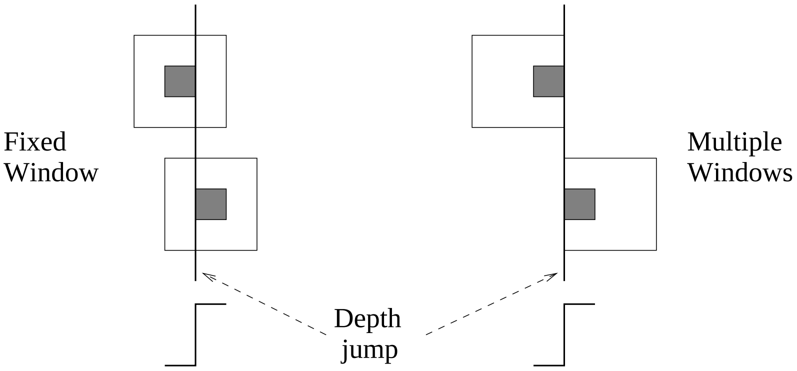

Figure 24: Multiple windows approach. If one use windows of fixed size with different centers, it is likely that one of them will cover a constant depth area.

SBM has an asymptotic complexity of $O(N^2nm)$, with N the image size. However, we can observe that squared differences need to be computed only once for each disparity, and the sum over the window needs not be recomputed from scratch when the window moves by one pixel (see Figure 23). The optimized implementation that follows from this observation [35] has a computational complexity of $O(4N^2)$, that is *independent of the window size.*

4.3.3 The need for multiple windows

As observed by Kanade and Okutomi [79], when the correlation window covers a region with non-constant disparity, area-based matching is likely to fail, and the error in the depth estimates grows with the window size. Reducing the latter, on the other hand, makes the estimated disparities more sensitive to noise.

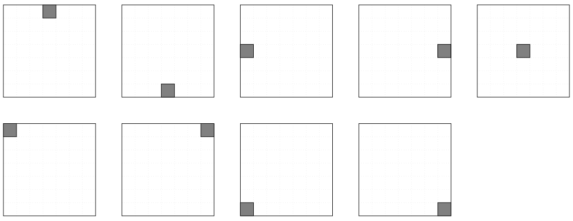

Figure 25: The nine correlation windows. The pixel for which disparity is computed is highlighted.

To overcome such difficulties, Kanade and Okutomi proposed a statistically sound, adaptive technique which selects at each pixel the window size that minimizes the uncertainty in the disparity estimates.

In the present work we take the multiple-window approach, in the simplified version proposed by [74, 47]. For each pixel we perform the correlation with nine different windows (showed in Figure 25), and retain the disparity with the smallest SSD error value. The idea is that a window yielding a smaller SSD error is more likely to cover a constant depth region; in this way, *the disparity profile itself drives the selection of an appropriate window.*

Figure 26 illustrates how the window size is adapted to the disparity profile. The point $x = 43$ is a (left) disparity jump. Point $x = 84$ marks the beginning of an occluded area extending to $x = 91$. Negative/positive window sizes refer to the oriented extent of the window with respect to the pixel for which disparity is computed.

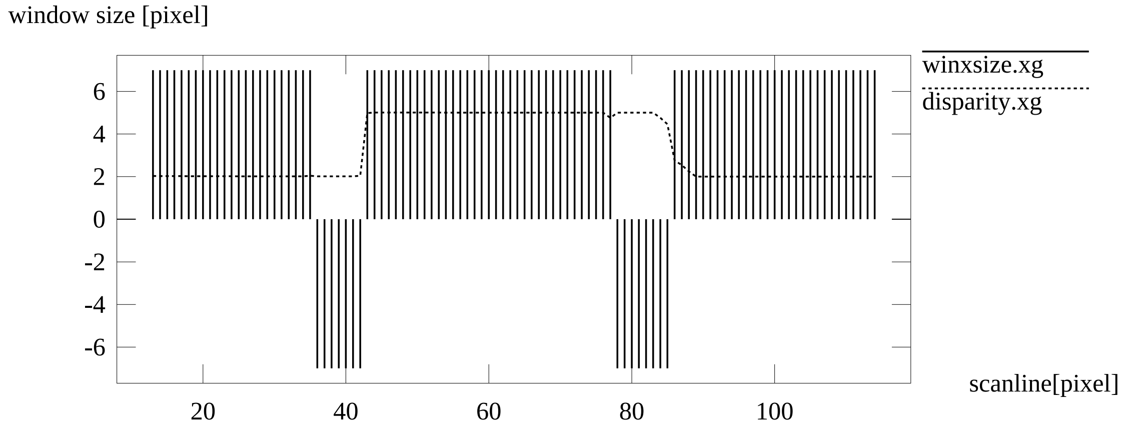

Figure 26: How the window size adapts to a disparity profile. The dashed lines show the disparity profile computed along a raster line of the stereo pair of Figure 31. Solid lines mark the window sizes.

4.3.4 Occlusions and left-right consistency

Occlusions create points that do not belong to any conjugate pairs. Usually, occlusions involve depth discontinuities: indeed, occlusions in one image correspond to disparity jumps in the other [47].

A key observation to address the occlusion problem is that *matching is not a symmetric process*: taking different images (right or left) as reference, one obtains, in general, different sets of conjugate pairs, in which some points are involved in more than one conjugate pairs. Such pairs are not invariant to the choice of the reference image. As each point in one image can have at most one corresponding point in the other (the *uniqueness constraint*), such pairs can be discarded (*left-right consistency*) [41, 35].

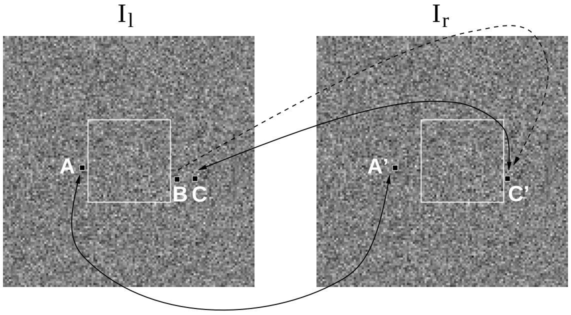

Figure 27: Left-right consistency. Matching left to right, point A is correctly matched to A'. Point B is incorrectly given C' as a match, but C' matches actually $C \neq B$.

Consider for instance point B of Figure 27 and take the left image, I_l, as reference. Although B has no corresponding point in the right image, I_l (its conjugate point is occluded), the SSD minimization returns a match anyhow (C'). If I_r is taken as reference, instead, C' is correctly matched to its conjugate point (C) in the left image. Therefore the conjugate pairs (B, C') and (C, C') violate left-right consistency; in other words, C' does not satisfy the uniqueness constraint. Notice that the (C', C) pair allow us to recognize that point B is occluded (strictly speaking, its conjugate point is occluded); our approach takes advantage of left-right consistency to detect occlusions and suppress the resulting infeasible matches.

For each point (x, y) in the left image, the disparity $d_l(x, y)$ is computed as described in Section 4.3.2. The process is repeated with the right image as reference.

If $d_l(x, y) = -d_r(x + d_l(x, y), y)$ the point is assigned the computed disparity; otherwise it is marked as occluded and a disparity is assigned heuristically. Following [85], we assume that occluded areas, occurring between two planes at different depth, take the disparity of the deeper plane.

It should be said that, in presence of large amount of noise or distortion, the left-right consistency could fail for true conjugate pairs, and points could be wrongly marked as occluded. A simple non-linear filtering of the occlusions map (a binary image showing only occluded points) would discard those "outliers".

4.3.5 Uncertainty estimates

Area-based algorithms are likely to fail not only in occluded regions, but also in poorly textured regions, which make disparity estimates more uncertain; it is therefore essential to assign *confidence estimates* to disparities. Several uncertainty estimation schemes have been proposed for SSD, mostly based on the shape of the SSD error function [2, 148].

Our approach takes advantage of the multiple windows. Disparity estimation is sensitive to window shape in two cases: first, near a disparity jump (as discussed in Section 4.3.3) and, second, where the texture is poor, or the signal-to-noise ratio (SNR) is low. Consequently, we define uncertainty as the estimated variance of the disparity measures obtained with the various windows (see algorithm summary in next section); occluded points are assigned infinite variance. Experimental results show that such an uncertainty measure is consistent, i.e., it grows as the SNR decreases (Section 4.4).

4.3.6 Summary of the SMW algorithm

We summarize our algorithm, called SMW (for Symmetric Multi-Window) in pseudocode. Let $C(x, y, d; I_l, I_r, w)$ be the SSD error computed from I_l to I_r according to (79) at point (x, y), with disparity d and window w. Let s_l be the sub-pixel correction defined by (81). The y coordinate is omitted for the sake of simplicity, since we assume horizontal epipolar lines.

Algorithm 2 SMW

```
let $I_r$, $I_l$ the right and left $N \times N$ images;
for all $(x, y)$ in the left image $I_l$ do
    for all windows $w = 1 \dots K$ do
        $d_{l,w}(x) \leftarrow \arg\min_d C(x, y, d; I_l, I_r, w)$
        $d_{r,w}(x) \leftarrow \arg\min_d C(x, y, d; I_r, I_l, w)$
    end
    $\sigma_d^2(x) = \frac{1}{K-1} \sum_{w=1}^{K} (d_{l,w}(x) - \bar{d}_{l,w}(x))^2$.
    $d_l(x) \leftarrow \arg\min_w C(x, y, d_{l,w}; I_l, I_r, w)$
```

```
        $d_r(x) \leftarrow \arg\min_w C(x, y, d_{r,w}; I_r, I_l, w)$
        $d(x) \leftarrow d_l(x) + \text{subpixel}_l(x)$
    end
    for all $(x, y)$ in $I_l$ do
        if $(d_l(x) \neq -d_r(x + d_l(x)))$ then $\sigma_d^2(x) \leftarrow +\infty$
    end
end
```

It is worth noting that the only one SSD value per pixel needs to be computed. Indeed, each off-centered windows for a pixel is the on-centered window for another pixel.

4.4 Experimental results

This section reports the main results of experimental evaluation of SMW. The evaluation was aimed at assessing

- the accuracy of disparity computation,
- robustness against occlusion,
- the consistency of uncertainty estimation,
- the performance of SMW when compared to similar algorithms.

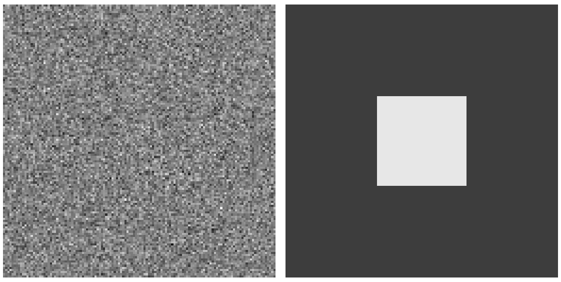

Figure 28: Square RDS. The right image of the stereogram is computed by warping the left one, which is a random texture (left), according to a given disparity pattern (right): the square has disparity 10 pixel, the background 3 pixel.

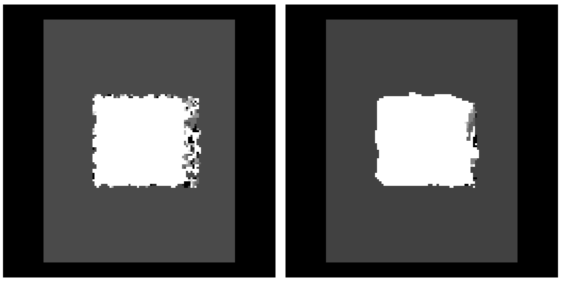

Figure 29: Computed disparity map by SBM for the square RDS with 3×3 window (left) and 7×7 window (right); MAE is 0.240 and 0.144, respectively.

We used synthetic data sets commonly found in the stereo literature and controlled amounts of noise. We also reproduced patterns used for testing algorithms used in our comparative evaluation. The next section reports the results as well as further tests with real stereo pairs of size 128 × 128.

4.4.1 Random-dot stereograms

We first performed experiments on noise-free random-dot stereograms (RDS), shown in Figure 28. In the disparity maps, displayed as images, the gray level encodes the disparity, that is the depth (the brighter the closer); images have been equalized to improve readability; sub-pixel accuracy values have been rounded to integers. The estimated Mean Absolute Error (MAE), that is the mean of absolute differences between estimated and ground true disparities, has been computed as a performance index.

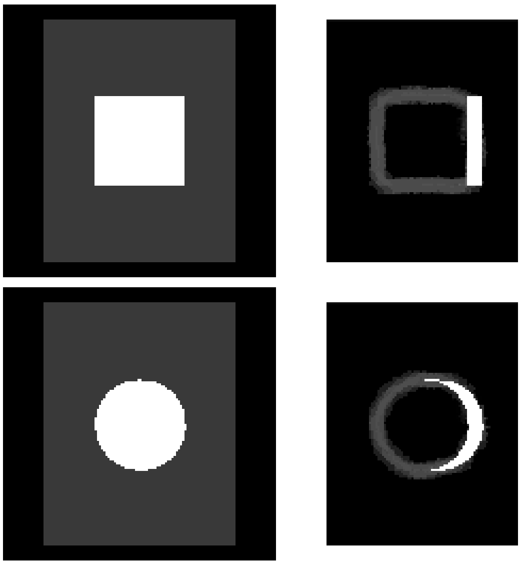

Figure 30: Computed disparity map (left) and uncertainty (right) by SMW for the square RDS (top) and for the circle RDS (bottom). MAE is 0.019 and 0.026. respectively.

The results of SBM applied to the random-dot stereogram of Figure 28 shows how most of the problems outlined in Sections 4.3.3 and 4.3.4 affect disparity computation. Figure 29 shows the disparity maps computed by SBM with fixed windows 3×3 and 7×7. Both pictures show the effect of disparity jumps (near the left and

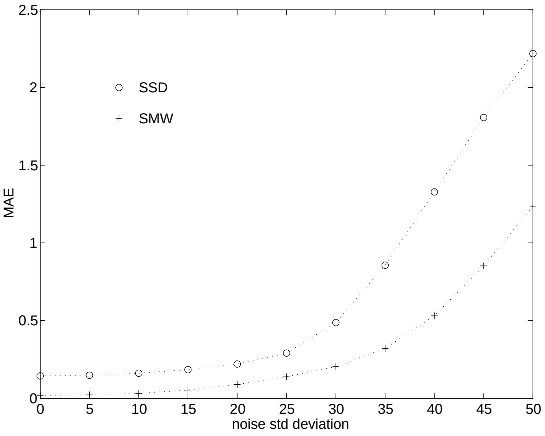

Figure 31: MAE of SMW and SBM vs noise standard deviation for the square RDS. Window is 7×7.

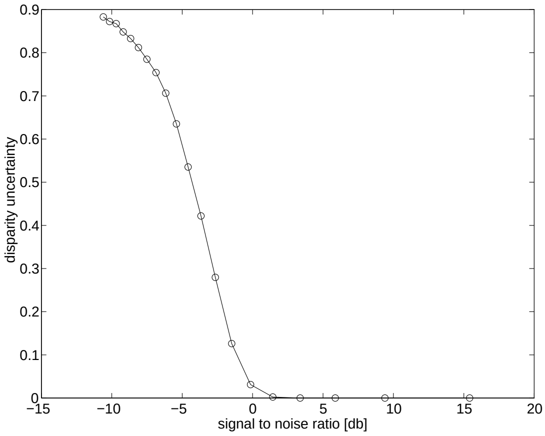

Figure 32: Mean uncertainty vs SNR for a constant disparity region of the square RDS.

horizontal borders of the square patch) and of occlusions (near the right border of the square patch). The SMW algorithm with a 7×7 window was applied to the square RDS of Figure 28 and to a circular RDS (not shown here). Figure 30 show the disparity maps computed by SMW and the estimated uncertainty maps (the darker the lower) in both cases.

The MAE is negligible, and may be ascribed to sub-pixel estimation only. The occluded points, shown in white in the uncertainty maps, are identified with 100% accuracy in both cases. The circle RDS shows that the algorithm is not biased toward square disparity patterns, as the shape of the SSD windows might suggest. The reader could compare the present results to those reported in [26].

Experiments with various noisy RDSs show a graceful degradation when noise increases. Gaussian noise with zero mean and increasing variance was added independently to both images of the square RDS. Figure 31 plots the MAE against the standard deviation of the noise for SMW and SBM. Each point depicts the average result of 20 independent trials. Images were 8-bit deep, monochrome.

In order to assess the uncertainty estimator incorporated in SMW, we plotted the average uncertainty computed over a square patch of uniform disparity against the SNR, defined as

$$\mathrm{SNR} = 10 \log_{10} \frac{\text{Image variance}}{\text{Noise variance}}. \tag{82}$$

The results (Figure 32) show that the computed uncertainty consistently increases as the SNR decreases.

4.4.2 Gray-level ramp

We performed a systematic, quantitative comparison between SMW, our implementation of the Adaptive Window (AW) algorithm [79] (perhaps the closest method to SMW in the literature), and SBM with different window sizes. The evaluation was based on the main test pattern used by [79]: an input stereo pair of an intensity ramp in the horizontal direction, warped according to a given disparity pattern. The left disparity jump creates a "disocclusion" area that is filled with random dots (Figure 33). Gaussian noise with zero mean and unit variance (gray level) was added to both images independently.

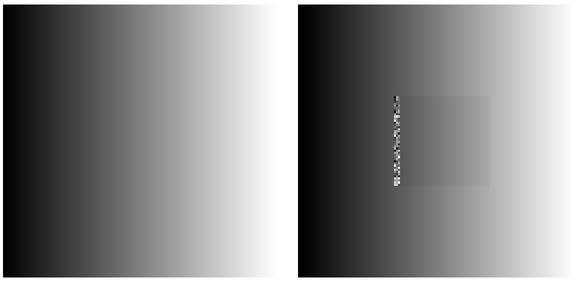

Figure 33: Gray-level ramp stereo pair. The central square has disparity 5 pixel, the background 2 pixel.

Figure 34 illustrates a comparison of the three algorithms using the gray-level ramp stereo pair.

Figure 35 compares qualitatively the isometric plots of the absolute errors (absolute differences of true and reconstructed depths) for AW and SMW. Further comparisons are illustrated in Table 1, which summarizes the results of our comparison of the MAE for SBM, AW, and SMW, using input pairs with different noise levels and different window sizes.

Results with SBM (Figure 34) confirm that too small windows (e.g., 3×3) increase sensitivity to noise, whereas larger windows (e.g., 7×7) act as low-pass filters and are likely to blur depth discontinuities.

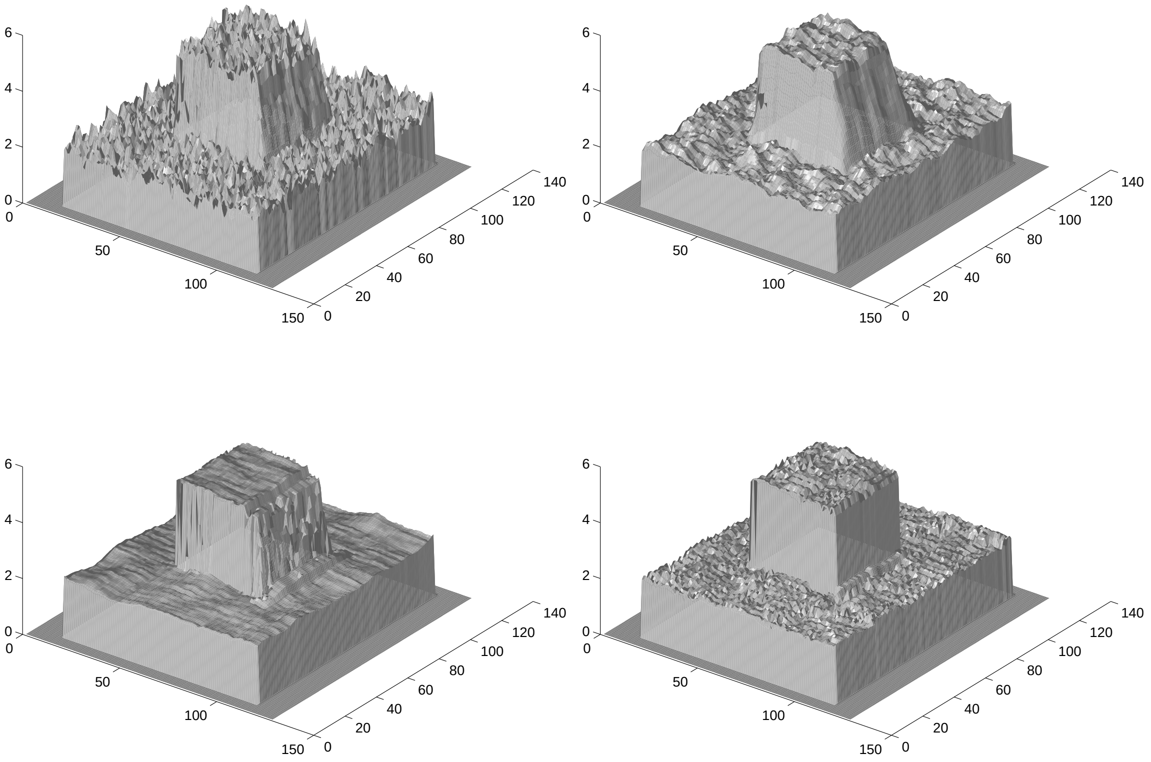

Figure 34: Isometric plots of the disparity maps computed with: SBM 3×3 window (top left) and 7×7 window (top right), AW (bottom left) and SMW 7×7 algorithms (bottom right), with $\sigma^2 = 1.0$. The orientation is chosen to show occluded points.

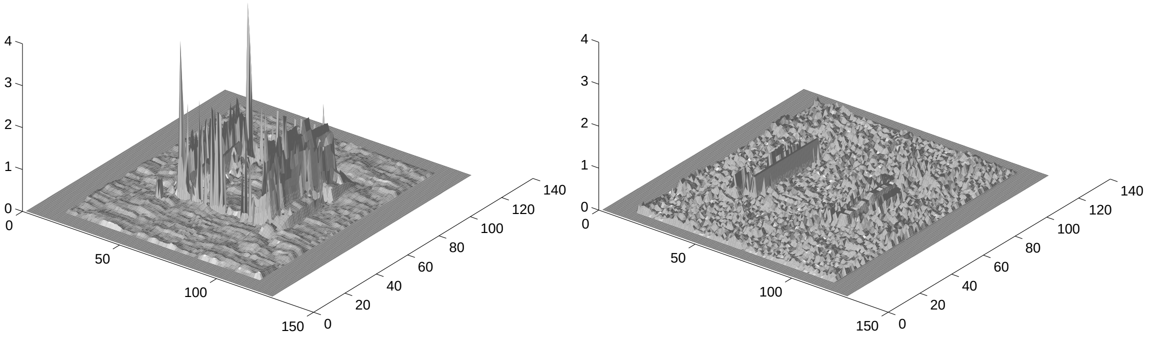

Figure 35: Isometric plots of estimated errors, as differences between computed and true disparities for the AW (left) and SMW algorithm (right).

Algorithm	MAE		
	$\sigma^2 = 1.0$	$\sigma^2 = 3.0$	$\sigma^2 = 10.0$
SBM 7x7	0.182	0.468	1.235
SBM 15x15	0.284	0.392	0.988
AW	0.101	0.244	1.045
SMW 7x7	0.082	0.318	0.979
SMW 15x15	0.059	0.235	0.819

Table 1: Comparison of estimated errors: mean absolute (MAE) for different noise variances. Notice that 15×15 is the maximum window size allowed for AW.

More interestingly, Figure 34 shows that AW is the most accurate (since it reduces simultaneously both random and systematic errors along the disparity edges), but performs poorly within occluded areas, leading to large local errors (Figure 35), as it does not exploit the uniqueness constraint. Sub-pixel corrections are smooth since this algorithm is essentially a complex, iterative sub-pixel adjustment. SMW yields a depth map that is globally more reliable, as it enforces left-right consistency: occluded points are detected with 100% accuracy.

The slight amount of noise across the disparity surface (Figure 35) is due to the simple sub-pixel accuracy method, the main source of errors for SMW. Further experiments with larger disparities (not reported here) show that the improvement in accuracy achieved by SMW with respect to AW increases with disparity, owing to the increasingly large areas of occlusion[1].

Another advantage of SMW with respect to AW is *efficiency*. Running on a SUN SparcStation 4 (110MHz) under SunOS 5.5, our implementation of the SMW takes 8 seconds, on average, to compute the depth maps in Figure 34 (128×128 input images), whereas AW takes 32 minutes on average.

4.4.3 Real data

We report the results of the application of the SMW algorithm on standard image pairs from the JISCT (JPL-INRIA-SRI-CMU-TELEOS) stereo test set, and from the CMU-CIL (Carnegie-Mellon University—Calibrated Imaging Laboratory) in Figure 37. In the disparity maps, the gray level encodes disparity, that is depth

[1] Notice that our implementation of AW failed to converge to a solution with RDSs, probably because this algorithm relies on intensity derivatives, which are ill-defined for random dot patterns.

(the brighter the closer). Images have been equalized to improve readability. Subpixel accuracy values have been rounded to integer values for display. We also report the estimated variance maps (the darker the lower). Small values cannot be appreciated in spite of histogram equalization, due to the large difference between high-uncertainty occlusion points and the rest of the image. Although a quantitative comparison with other methods was not possible with real images, the quality of SMW results seems perfectly comparable to that of the results reported, for example, in [161, 47, 26].

Running on a Sun SparcStation 4 (110MHz) under SunOS 5.5, our current implementation takes 50 seconds, on average, to compute depth maps from 256×256 pairs, with a disparity range of 10 pixels.

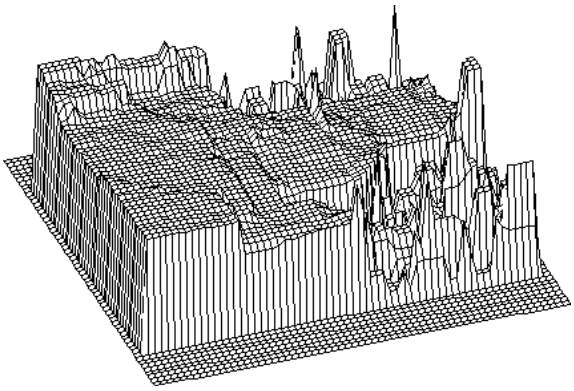

Figure 36: Height field for the “Castle” stereo pair.

Reconstruction. If the camera parameters are known, the three-dimensional structure can be computed from the disparity map (Chapter 3). As an example, Figure 36 shows the *height field* computed from the “Castle” stereo pair, given the focal length in pixels (2497) and the baseline (21mm). The values of the height field are true distances (in mm) from an arbitrary reference plane placed at a 2300mm from the focal plane. Since the disparity range is $[21, 28]$, the corresponding depth range is $[1694\text{mm}, 2259\text{mm}]$, hence the reference plane is behind every scene object.

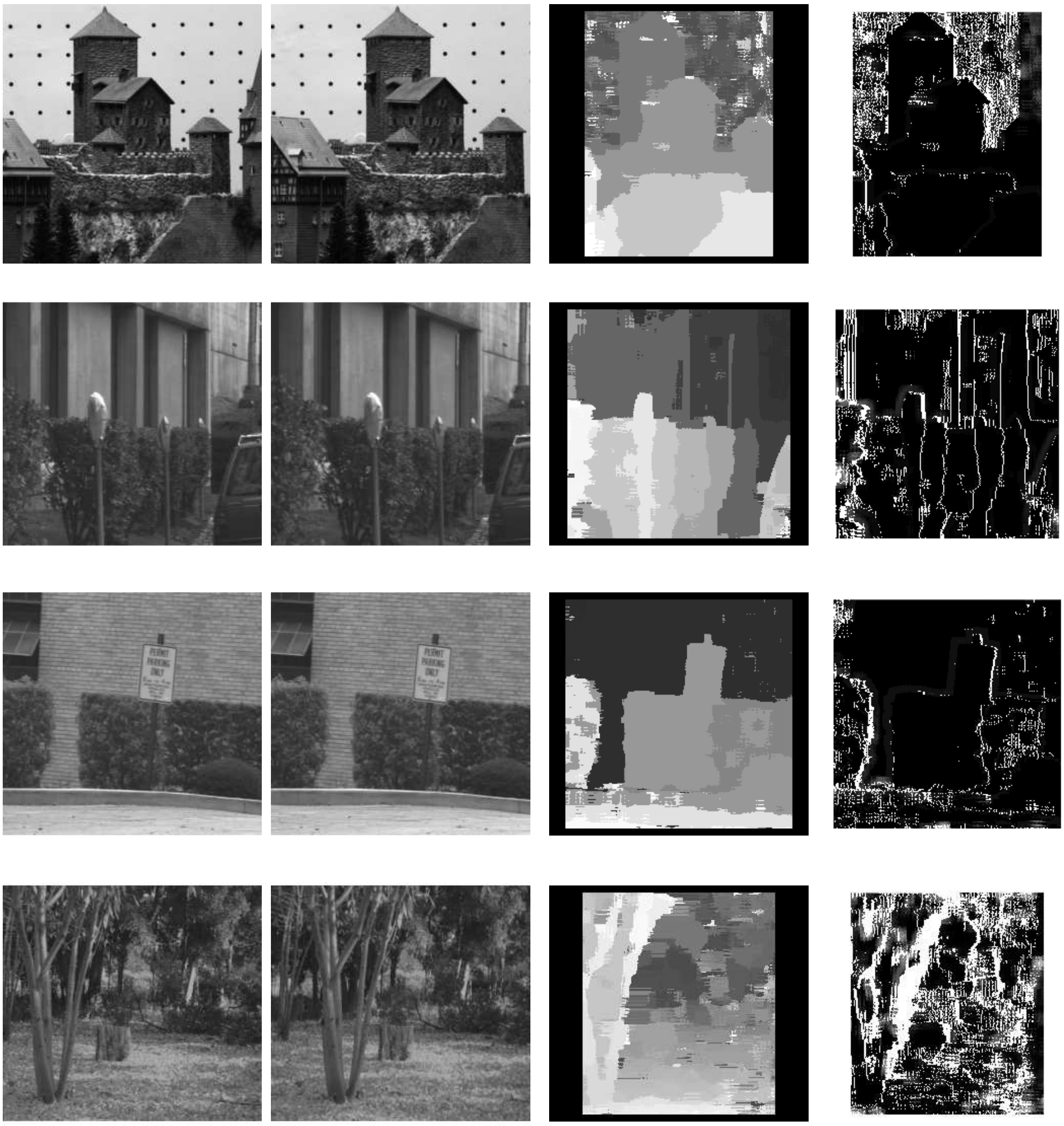

Figure 37: Disparity (left) and uncertainty maps (right) for the “Castle”, “Parking meter”, “Shrub” and “Trees” stereo pairs

4.5 Conclusions

We have introduced a new, efficient algorithm for stereo reconstruction, SMW, based on a multi-window approach, and taking advantage of left-right consistency. Our tests have shown the advantages offered by SMW. The adaptive, multi-window scheme yields robust disparity estimates in the presence of occlusions, and clearly outperforms fixed-window schemes. If necessary, the slight amount of noise caused by sub-pixel interpolation can be kept small by increasing the baseline, which does not worsen performance significantly thanks to the robust treatment of occlusions. This is an advantage over several stereo matching schemes, often limited by the assumption of small baselines.

Left-right consistency proves effective in eliminating false matches and identifying occluded regions (notice that this can be regarded as a segmentation method in itself). In addition, disparity is assigned to occluded points heuristically, thereby achieving reasonable depth maps even in occluded areas. Uncertainty maps are also computed, allowing the use of SMW as a module within more complex data fusion frameworks. As for any area-based correspondence method, SMW's performance is affected adversely by poorly-textured regions, but areas of low texture are associated consistently with high uncertainty values.

The efficiency of SMW is globally superior to that of similar adaptive-window methods, as shown by direct comparisons with [79] reported. The reason is that SMW performs a one-step, single-scale matching, with no need for interpolation and optimization. The main disadvantage is that the window size remains a free parameter; notice, however, that adaptive-window schemes are much slower in achieving comparable accuracies.

Possible developments are to embed the SMW module in a dynamic stereo system. We have experimented with the integration of stereo with shape from shading technique [165], with promising results [27].

Chapter 5

Structure from Motion

In this chapter we will address the *structure from motion* problem: given several views of a scene taken with a moving camera with known intrinsic parameters and given a set of matched points, recover the motion of the camera and the structure of the scene. This is known in photogrammetry as the *relative orientation* problem.

In the previous sections we discussed the fully calibrated case, in which we had a calibrated rig of two cameras and reconstruction was possible as long as correspondences between images could be established. In this chapter we consider a single moving camera; the intrinsic parameters are known but the camera motion is unknown (i.e., the extrinsic parameters are missing). The problem of obtaining the matches themselves which will be studied in detail in Chapter 6.

5.1 Introduction

The structure from motion problem has been studied extensively by the computer vision community in the past decade (see [73] for a review). The approaches to motion estimation can be partitioned into *differential* [136, 130, 129, 153] and *discrete* methods, depending on whether they use as an input image point velocities (the *motion field*) or a set of matched points ([95] discuss the relationship between the two approaches). Among the latter methods, orthographic or para-perspective approximations for the camera have been used [138, 114]. One of the most appealing approaches, using the full perspective camera model, was proposed by Longuet-Higgins [86]. This method is based on the *essential matrix*, that describes the *epipolar geometry* of two perspective images. The so-called Longuet-Higgins equation, which defines the essential matrix, will be derived in Section 5.2.

The essential matrix encodes the rigid displacement of the camera, and indeed a theorem by Maybank and Faugeras [36] allows us to factorize it into a rotation and a translation matrix (Section 5.3). As the intrinsic parameters are known, this is tantamount to knowing the full camera matrices, and structure (i.e., the distance of the points to the camera) follows easily by triangulation (as in Chapter 3). Note that the translational component of displacement can be computed only up to a scale factor, because it is impossible to determine whether a given image motion is caused by a nearby object with slow relative motion or a distant object with fast relative motion (this is known as the *depth-speed ambiguity*).

In Section 5.4 we deal with the problem of computing the essential matrix. A simple linear method, called the *8-point algorithm* [86, 61] is described.

In Section 5.5 a non-linear iterative algorithm that compute motion parameters directly from correspondences in normalized coordinates [71, 72] is outlined.

Our implementation of the algorithm for computing structure and motion is described in Section 5.6. Following [158, 167, 92], we use the results of the 8-point algorithm as the initial guess for the iterative method.

Experimental results with synthetic and real images are reported in Section 5.7.

5.2 Longuet-Higgins equation

Let us assume that we have a camera, with known intrinsic parameters, that is moving in a static environment, following some unknown trajectory. Let us consider two images taken by the camera at two time instants and assume that we are given a number of point matches between the images, *in normalized coordinates.* Let $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{P}}'$ the camera matrices corresponding to two time instants, and $\tilde{\mathbf{p}} = \mathbf{A}^{-1}\tilde{\mathbf{m}}$, $\tilde{\mathbf{p}}' = \mathbf{A}'^{-1}\tilde{\mathbf{m}}'$ the normalized coordinates of two matched image points P and P' respectively.

Working in normalized coordinates and taking the first camera reference frame as the world reference frame, we can write the following projection matrices:

$$\tilde{\mathbf{P}} = [\mathbf{I}|\mathbf{0}] \tag{83}$$

$$\tilde{\mathbf{P}}' = [\mathbf{I}|\mathbf{0}]\tilde{\mathbf{G}} = [\mathbf{R}|\mathbf{t}] \tag{84}$$

Let $\tilde{\mathbf{P}} = [\mathbf{Q}|\tilde{\mathbf{q}}]$. The epipolar line of P is the line containing the epipole E', whose coordinates are

$$\mathbf{e}' = \tilde{\mathbf{P}}' \begin{bmatrix} \mathbf{c} \\ 1 \end{bmatrix} = [\mathbf{R}|\mathbf{t}] \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \mathbf{t}, \tag{85}$$

and the projection through $\tilde{\mathbf{P}}'$ of the point at infinity of the optical ray of P:

$$\tilde{\mathbf{P}}' \begin{bmatrix} \mathbf{Q}^{-1}\tilde{\mathbf{p}} \\ 0 \end{bmatrix} = \mathbf{Q}'\mathbf{Q}^{-1}\tilde{\mathbf{p}} = \mathbf{R}\tilde{\mathbf{p}}. \tag{86}$$

In the projective plane, the line joining two points is represented by the external product $(\mathbf{t} \wedge \mathbf{R}\tilde{\mathbf{p}})$, hence the P', the conjugate point of P satisfies the following equation, which is called the *Longuet-Higgins equation*:

$$\tilde{\mathbf{p}}'^{\top}(\mathbf{t} \wedge (\mathbf{R}\tilde{\mathbf{p}})) = 0. \tag{87}$$

In the collapsed vector space interpretation of the projective plane (see Appendix A), the latter (a triple product) expresses the co-planarity of the three vectors $\tilde{\mathbf{p}}', \mathbf{t}, (\mathbf{R}\tilde{\mathbf{p}})$.xf

By introducing the skew-symmetric matrix $[\mathbf{t}]_\wedge$ for the external product with $\mathbf{t}$, (87) writes

$$\tilde{\mathbf{p}}'^{\top}[\mathbf{t}]_\wedge\mathbf{R}\tilde{\mathbf{p}} = 0. \tag{88}$$

The matrix

$$\mathbf{E} = [\mathbf{t}]_\wedge\mathbf{R} \tag{89}$$

is called the *essential matrix*. Since $\det[\mathbf{t}]_\wedge = 0$, $\mathbf{E}$ has rank 2. Besides, it is only defined up to a scale factor, because (87) is homogeneous with respect to $\mathbf{t}$. This

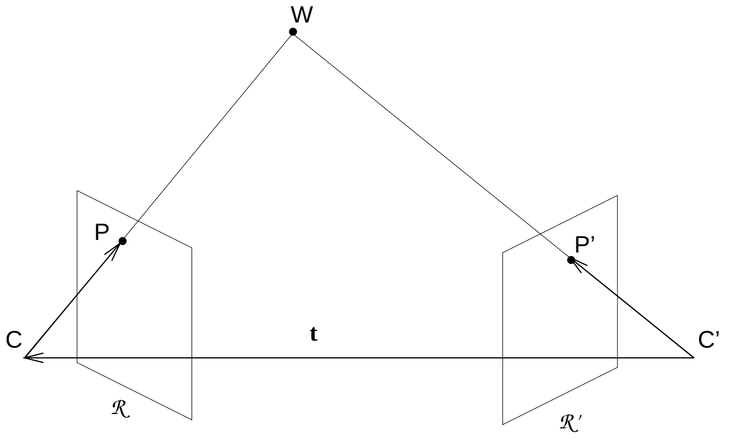

Figure 38: Longuet-Higgins equation as the co-planarity of three ray vectors.

reflects the depth-speed ambiguity, i.e., the fact that we cannot recover the absolute scale of the scene without an extra yardstick, such as knowing the distance between two points. Therefore, an essential matrix has only five degrees of freedom (or, it depends upon five independent parameters), accounting for rotation and translation up to a scale factor.

The essential matrix and the fundamental matrix are linked, since they both encode the rigid displacement between two views. The former links the *normalized* coordinates of conjugate points, whereas the latter links the *pixel* coordinates of conjugate points. It will be shown in Section 7.2 that

$$\mathbf{F} = \mathbf{A}'^{-\top}\mathbf{E}\mathbf{A}^{-1}. \tag{90}$$

5.3 Motion from the factorization of E

Let us assume that the essential matrix is given. The following theorem, by Maybank and Faugeras [36] allows us to factorize the essential matrix into rotation and translation. Unlike the fundamental matrix, the only property of which is to have rank 2, the essential matrix is characterized by this theorem. Following [61], we will give here a more compact proof than in [36], based on Singular Value Decomposition (SVD).

Theorem 5.1

A real matrix $\mathbf{E}$ 3×3 can be factorized as product of a nonzero skew-symmetric matrix and an orthogonal matrix if and only if $\mathbf{E}$ has two identical singular values and a zero singular value.

Proof Let $\mathbf{E} = \mathbf{S}\mathbf{R}$ where $\mathbf{R}$ is orthogonal and $\mathbf{S}$ is skew-symmetric. Let $\mathbf{S} = [\mathbf{t}]_\wedge$ where $||\mathbf{t}|| = 1$. Then

$$\mathbf{E}\mathbf{E}^\top = \mathbf{S}\mathbf{R}\mathbf{R}^\top\mathbf{S}^\top = \mathbf{S}\mathbf{S}^\top = \mathbf{I} - \mathbf{t}\mathbf{t}^\top$$

Let $\mathbf{U}$ the orthogonal matrix such that $\mathbf{U}\mathbf{t} = [0,0,1]^\top$. Then

$$\mathbf{U}\mathbf{E}\mathbf{E}^\top\mathbf{U}^\top = \mathbf{U}(\mathbf{I} - \mathbf{t}\mathbf{t}^\top)\mathbf{U}^\top = \mathbf{I} - \mathbf{U}\,\mathbf{t}\,\mathbf{t}^\top\mathbf{U}^\top = \mathbf{I} - [0,0,1]^\top\,[0,0,1] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

This demonstrate one implication. Let us now give a constructive proof of the converse. Let $\mathbf{E} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ be the SVD of $\mathbf{E}$, with $\mathbf{D} = \mathrm{diag}(1,1,0)$ (with no loss of generality, since $\mathbf{E}$ is defined up to a scale factor). The key observation is that

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathbf{S}'\mathbf{R}'$$

where $\mathbf{S}'$ is skew symmetric and $\mathbf{R}'$ orthogonal.
Hence

$$\mathbf{E} = \mathbf{U}\mathbf{D}\mathbf{V}^\top = \mathbf{U}\mathbf{S}'\mathbf{R}'\mathbf{V}^\top = (\mathbf{U}\mathbf{S}'\mathbf{U}^\top)(\mathbf{U}\mathbf{R}'\mathbf{V}^\top).$$

Taken $\mathbf{S} = \mathbf{U}\mathbf{S}'\mathbf{U}^\top$ and $\mathbf{R} = \mathbf{U}\mathbf{R}'\mathbf{V}^\top$, the sought factorization is $\mathbf{E} = \mathbf{S}\mathbf{R}$. □

This factorization is not unique. We can obtain the same $\mathbf{D}$ matrix by changing both sign of $\mathbf{S}'$ and $\mathbf{R}'$. Moreover, because of the ambiguity in the sign of $\mathbf{E}$, we can change the sign of $\mathbf{D}$, either by taking opposite sign for $\mathbf{S}'$ and $\mathbf{R}'$, or by taking the transpose of $\mathbf{R}$ (because $\mathbf{S}'\mathbf{R}'^\top = -\mathbf{D}$). In total, taking all the combinations of $\pm\mathbf{S}, \pm\mathbf{R}, \pm\mathbf{R}^\top$, we have eight possible factorizations. Since the sought $\mathbf{R}$ must be a *rotation matrix*, there are only four possible factorizations, given by:

$$\mathbf{S} \simeq \mathbf{U}\mathbf{S}'\mathbf{U}^\top \qquad (91)$$

$$\mathbf{R} \simeq \mathbf{U}\mathbf{R}'\mathbf{V}^\top \text{ or } \mathbf{R} \simeq \mathbf{U}\mathbf{R}'^\top\mathbf{V}^\top, \qquad (92)$$

where

$$\mathbf{S}' = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \mathbf{R}' = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{93}$$

with the constraint $\det \mathbf{R} = 1$.

As remarked by Longuet-Higgins, the choice between the four displacements is determined by the requirement that the points location, which can be computed by building the camera matrices (83) and (84), must lie in front of both cameras, i.e., their third coordinate must be positive.

5.4 Computing the essential (fundamental) matrix

In this section we will address the problem of the estimation of $\mathbf{E}$ from points correspondences.

We will consider the computation of the fundamental matrix $\mathbf{F}$ since this is a more general problem; if we can compute $\mathbf{F}$ we are also able to compute $\mathbf{E}$ as long as intrinsic parameters are known, either by using (90) or by switching from pixel to normalized coordinates.

The problem of computing the fundamental matrix has been given a great amount of attention in recent years (see [168] for a review). A variety of method have been proposed and studied, ranging from fairly simple linear methods to robust non-linear ones [142].

5.4.1 The 8-point algorithm

Given a (sufficiently large) set of point matches: $\{(\mathbf{m}_i, \mathbf{m}'_i) \mid i = 1, \dots, n\}$, in pixel coordinates, the fundamental matrix is defined by the following equation:

$$\tilde{\mathbf{m}}_i'^{\top} \mathbf{F} \tilde{\mathbf{m}}_i = 0. \tag{94}$$

which can be used to compute the unknown matrix $\mathbf{F}$, since each point match gives rise to one linear homogeneous equation in the nine unknown entries of the matrix

$\mathbf{F} = [F_{i,j}]$:

$$\mathbf{u}_i^\top \mathbf{f} = 0, \tag{95}$$

where

$$\begin{aligned} \mathbf{u}_i &= [u_i u_i', v_i u_i', u_i', u_i v_i', v_i v_i', v_i', u_i, v_i, 1]^\top \\ \mathbf{f} &= [F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}]^\top. \end{aligned}$$

From n corresponding points we obtain an over-constrained linear system

$$\mathbf{U}_n \mathbf{f} = 0, \tag{96}$$

where

$$\mathbf{U}_n = [\mathbf{u}_1, \ldots, \mathbf{u}_n]^\top.$$

The solution vector $\mathbf{f}$ is defined up to a scale factor; in order to avoid the trivial solution $\mathbf{f} = 0$, it is customary to add the constraint

$$\| \mathbf{f} \| = 1. \tag{97}$$

If we ignore that a proper fundamental matrix should have rank 2, it is possible to find a solution to the system (5.4.1) with as few as eight point matches (excluding degenerate configurations [36]). For this reason this is called the *8-point algorithm*. In practice, more than eight point matches are available, and we can compute the entries of $\mathbf{F}$ by solving a linear least squares problem:

$$\min_{\mathbf{f}} \| \mathbf{U}_n \mathbf{f} \|^2 \quad \text{subject to: } \| \mathbf{f} \| = 1. \tag{98}$$

The solution is the unit eigenvector corresponding to the least eigenvalue of $\mathbf{U}_n^\top \mathbf{U}_n$, which can be computed by SVD of $\mathbf{U}_n$ (this is again the Linear-Eigen method that we used in Section 3.3). Note that the matrix $\mathbf{F}$ found by solving this set of linear equations will not in general have rank 2, as required for a proper fundamental matrix.

Data standardization The 8-point algorithm has been criticized for being sensitive to noise [91], and hence useless for practical purposes. Consequently, many iterative algorithms have been proposed for the computation of the fundamental matrix, all more complicated than the 8-point algorithm (see [168] for a review).

However, Hartley [63] showed that the instability is due mainly to bad conditioning rather than to the linear nature of the algorithm. Indeed by using pixel coordinates we are likely to obtain a bad conditioned system of linear equation, since homogeneous coordinates have very different magnitude: in a 256×256 image, a typical image point will be of the form $[128, 128, 1]$. By preceding the 8-point algorithm with a very simple *standardization* of the coordinates of the matched points, the condition number is made smaller and results become comparable with iterative algorithms. The standardization procedure is the following: the points are translated so that their centroid is at the origin and are then scaled so that the average distance from the origin is equal to $\sqrt{2}$. Let $\mathbf{T}$ and $\mathbf{T}'$ the resulting transformation in the two images and $\tilde{\mathbf{m}}^* = \mathbf{T}\tilde{\mathbf{m}}$, $\tilde{\mathbf{m}}'^* = \mathbf{T}'\tilde{\mathbf{m}}'$ the transformed points. Using $\tilde{\mathbf{m}}^*$ and $\tilde{\mathbf{m}}'^*$ in the 8-point algorithm, we obtain a fundamental matrix $\mathbf{F}^*$ that is related to the actual one by $\mathbf{F}^* = \mathbf{T}'\mathbf{F}\mathbf{T}^{-1}$, as it can be easily seen.

Enforcing constraints After computing $\mathbf{E}$ from $\mathbf{F}$ using (90), we need to enforce the constraints arising from Theorem (5.3), namely that $\mathbf{E}$ has two identical singular values and a zero singular value. This is done by replacing $\mathbf{E}$ with $\hat{\mathbf{E}}$, the closest matrix in Frobenius norm that satisfies the two constraints. Let $\mathbf{E}$ be any 3×3 matrix and $\mathbf{E} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ its SVD with $\mathbf{D} = \mathrm{diag}(r, s, t)$ and $r \geq s \geq t$. It can be shown that $\hat{\mathbf{E}} = \mathbf{U}\hat{\mathbf{D}}\mathbf{V}^\top$ where $\hat{\mathbf{D}} = \mathrm{diag}(\frac{r+s}{2}, \frac{r+s}{2}, 0)$.

In order to compute motion, Theorem (5.3) is used to factorize $\mathbf{E}$. Note that it is not necessary to recompute the SVD of $\mathbf{E}$, which is already available from the constraint enforcement step.

It may be worth noting that, although the linear algorithm we described needs at least eight points for computing $\mathbf{E}$, since the matrix depend on five parameters only, it is possible to compute it with five linear equation plus the polynomial constraints arising from Theorem (5.3). [36] proved that only ten solutions exist in this case.

5.5 Horn's iterative algorithm

The direct method for computing motion from the factorization of $\mathbf{E}$ is linear, fast and easy to implement. Yet, it has been shown to suffer from instability in the presence of noise. For optimal results, an iterative method is needed. In this section we present one due to Horn [71, 72], computing motion parameters directly from correspondences in normalized coordinates. Being a non-linear minimization, the method requires an initial guess close to the solution. This is provided by the results obtained from the factorization method.

Given n corresponding points, the relationship (87) can be re-written using the triple product notation[1]. For each conjugate pair, in normalized coordinates, $(\mathbf{p}_i, \mathbf{p'}_i)$ we have:

$$[\mathbf{t}, \mathbf{R}\mathbf{p}_i, \mathbf{p'}_i] = 0 \tag{99}$$

We can formulate a least-squares solution to the relative orientation problem by minimizing the sum of the square errors of deviations from (99):

$$\chi = \sum_{i=1}^{n} [\mathbf{t}, \mathbf{R}\mathbf{p}_i, \mathbf{p'}_i]^2 \tag{100}$$

subject to $\mathbf{t}^\top \mathbf{t} = 1$.

Given an initial estimate for the rotation and translation, it is possible to make iterative adjustments of the motion parameters that reduce the error (100). Let $\delta\mathbf{t}$ and $\delta\boldsymbol{\omega}$ be the infinitesimal changes in the translation and rotation respectively. Since translation is represented by a unit vector, changes in translation must leave its length unaltered, hence

$$\mathbf{t}^\top \delta\mathbf{t} = 0 \tag{101}$$

The correction to the baseline and rotation will change the triple product for each point to

$$[(\mathbf{t} + \delta\mathbf{t}), (\mathbf{R}\mathbf{p}_i + \delta\boldsymbol{\omega} \wedge \mathbf{R}\mathbf{p}_i), \mathbf{p'}_i] \tag{102}$$

The corrections are obtained by minimizing

$$\sum_{i=1}^{n} (e_i + \mathbf{c}_i^\top \delta\mathbf{t} + \mathbf{d}_i^\top \delta\boldsymbol{\omega})^2 \tag{103}$$

[1] The triple product is defined as $[\mathbf{x}, \mathbf{y}, \mathbf{z}] = \mathbf{x}^\top (\mathbf{y} \wedge \mathbf{z})$.

subject to $\mathbf{t}^\top \delta\mathbf{t} = 0$, where

$$\begin{aligned} e_i &= [\mathbf{t}, \mathbf{R}\mathbf{p}_i, \mathbf{p}'_i] \\ \mathbf{c}_i &= \mathbf{R}\mathbf{p}_i \wedge \mathbf{p}'_i \\ \mathbf{d}_i &= \mathbf{R}\mathbf{p}_i \wedge (\mathbf{p}'_i \wedge \mathbf{t}) \quad . \end{aligned} \tag{104}$$

The constraint can be added onto the minimization problem using the Lagrange multiplier λ to get a system of linear equations for the baseline, the rotation increments, and the Lagrange multiplier:

$$\begin{pmatrix} \mathbf{C} & \mathbf{F} & \mathbf{t} \\ \mathbf{F}^\top & \mathbf{D} & \mathbf{0} \\ \mathbf{t}^\top & \mathbf{0} & 0 \end{pmatrix} \begin{pmatrix} \delta\mathbf{t} \\ \delta\boldsymbol{\omega} \\ \lambda \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{c}} \\ \bar{\mathbf{d}} \\ 0 \end{pmatrix} \tag{105}$$

where

$$\mathbf{C} = \sum_{i=1}^{n} \mathbf{c}_i \mathbf{c}_i^\top \tag{106}$$

$$\mathbf{F} = \sum_{i=1}^{n} \mathbf{c}_i \mathbf{d}_i^\top \tag{107}$$

$$\mathbf{D} = \sum_{i=1}^{n} \mathbf{d}_i \mathbf{d}_i^\top \tag{108}$$

$$\bar{\mathbf{c}} = \sum_{i=1}^{n} e_i \mathbf{c}_i^\top \tag{109}$$

$$\bar{\mathbf{d}} = \sum_{i=1}^{n} e_i \mathbf{d}_i^\top \tag{110}$$

Once we have the corrections to the baseline and rotation, we have to apply them in a way that preserves the constraint that the translation is a unit vector and that rotation is represented correctly. Translation is updated by summing the increment and the result is normalized by dividing by its magnitude. Rotation, represented by an orthonormal matrix, is updated by multiplying it by the matrix

$$\begin{pmatrix} 0 & -\delta\omega_3 & \delta\omega_2 \\ \delta\omega_3 & 0 & -\delta\omega_1 \\ -\delta\omega_2 & \delta\omega_1 & 0 \end{pmatrix} \tag{111}$$

that is not exactly orthonormal for finite increments. Orthogonality is then enforced by SVD as follows. Let $\hat{\mathbf{R}}$ be the nearly orthonormal matrix obtained after updating and $\hat{\mathbf{R}} = \mathbf{UDV}^\top$ its SVD. It can be shown that $\mathbf{R} = \mathbf{UV}^\top$ is the closest (in Frobenius norm) orthonormal matrix.

5.6 Summary of the MOTION&STRUCTURE algorithm

In this section the summary of the MOTION&STRUCTURE algorithm is given. Note that the output structure differs from the true (or absolute) structure by a similarity transformation, composed by a rigid displacement (due to the arbitrary choice of the world reference frame) plus a a uniform change of scale (due to depth-speed ambiguity). This is called a *Euclidean reconstruction*.

1. given: intrinsic parameters $\mathbf{A}$ and point matches (pixels) $(\tilde{\mathbf{m}}, \tilde{\mathbf{m}}')$;
2. estimate $\mathbf{F}$ with the 8-point algorithm, using data standardization;
   compute $\mathbf{E}$ with (90);
3. replace $\mathbf{E}$ with $\hat{\mathbf{E}}$, the closest matrix that satisfies Theorem 5.3;
4. compute the factorization $\hat{\mathbf{E}} = \mathbf{SR}$, according to Theorem 5.3, with $\mathbf{S} = [\mathbf{t}]_\wedge$;
5. start Horn's iterative algorithm from the estimated displacement $(\mathbf{t}, \mathbf{R})$.
6. using the rigid displacement as the extrinsic parameters, instantiate camera matrices for the two views and compute 3-D points position by triangulation (Section 3.3).
7. output: rigid displacement $(\mathbf{t}, \mathbf{R})$ between two camera positions (motion), 3-D points coordinates (structure), in the standard reference frame of the first camera.

5.7 Results

We tested the MOTION&STRUCTURE algorithm with both synthetic and real images. Synthetic images were generated by projecting a set of 3-D points (taken from the model of the calibration jig of Section 3.2), with given camera matrices.

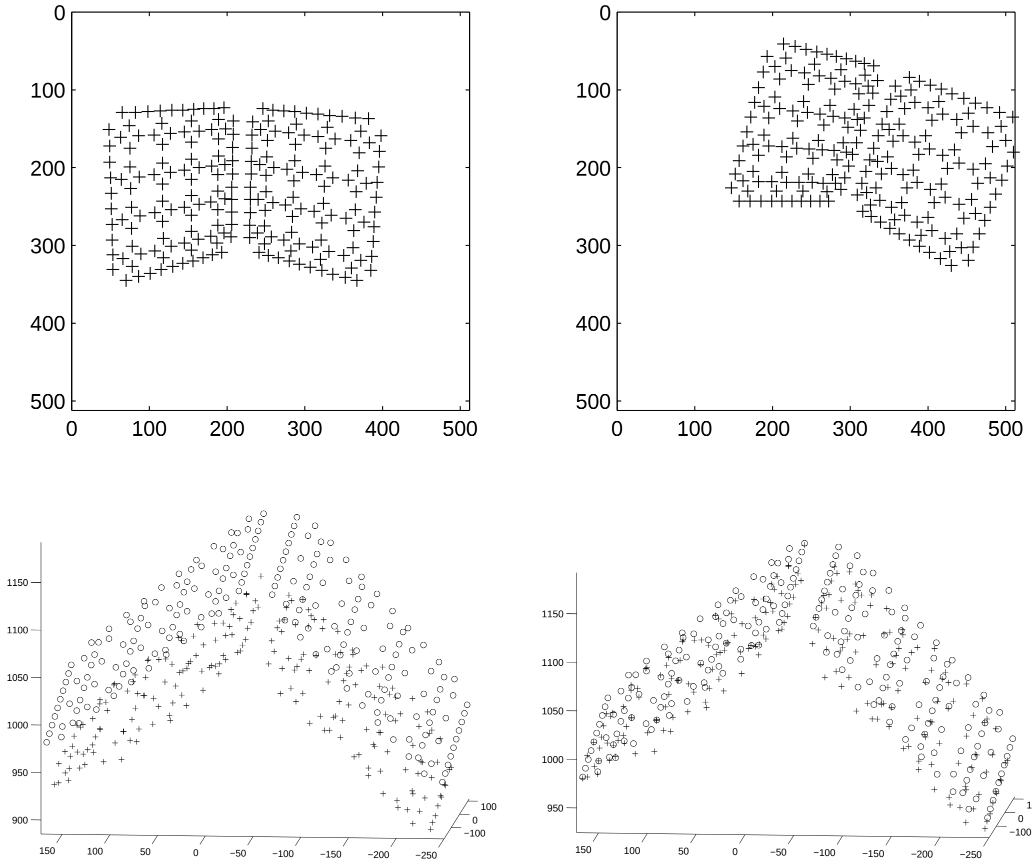

Figure 39: Synthetic frames (top row) and estimated structure (bottom row) using the 8-point algorithm only (left) and Horn's algorithm (right). Crosses are the reconstructed points, whereas the ground truth model is shown with circles.

In order to evaluate the benefit introduced by the iterative refinement, we computed motion and structure first with the 8-point algorithm only, and then running the iterative refinement. Figure 39 shows the reconstructed points for the synthetic pair. As expected, the reconstruction is more accurate after the iterative refinement. The better accuracy in motion estimation can be appreciated in Table 2, where the estimated motion parameters are compared with the ground-truth. Errors are computed as follows. We represent rotation with a vector whose direction gives the the axis of rotation and whose magnitude is the rotation angle. If $\hat{\mathbf{a}}$ is the estimate and $\mathbf{a}$ is the ground truth, errors are computed with

$$\mathrm{err} = \frac{||\mathbf{a} - \hat{\mathbf{a}}||}{||\mathbf{a}||}.$$

	rotation error	translation error
8-point	0.0167	0.0225
8-point + iterative	0.00340	0.00966

Table 2: Relative errors on motion parameters

As to real images, we used the “Stairs” sequence (512 x 768 pixels, 60 frames) for which we know the intrinsic parameters of the camera and the ground truth structure of the imaged object (courtesy of F. Isgrò, Heriot-Watt University). Correspondences between the first and last frame was obtained using our robust tracker, described in Chapter 6.

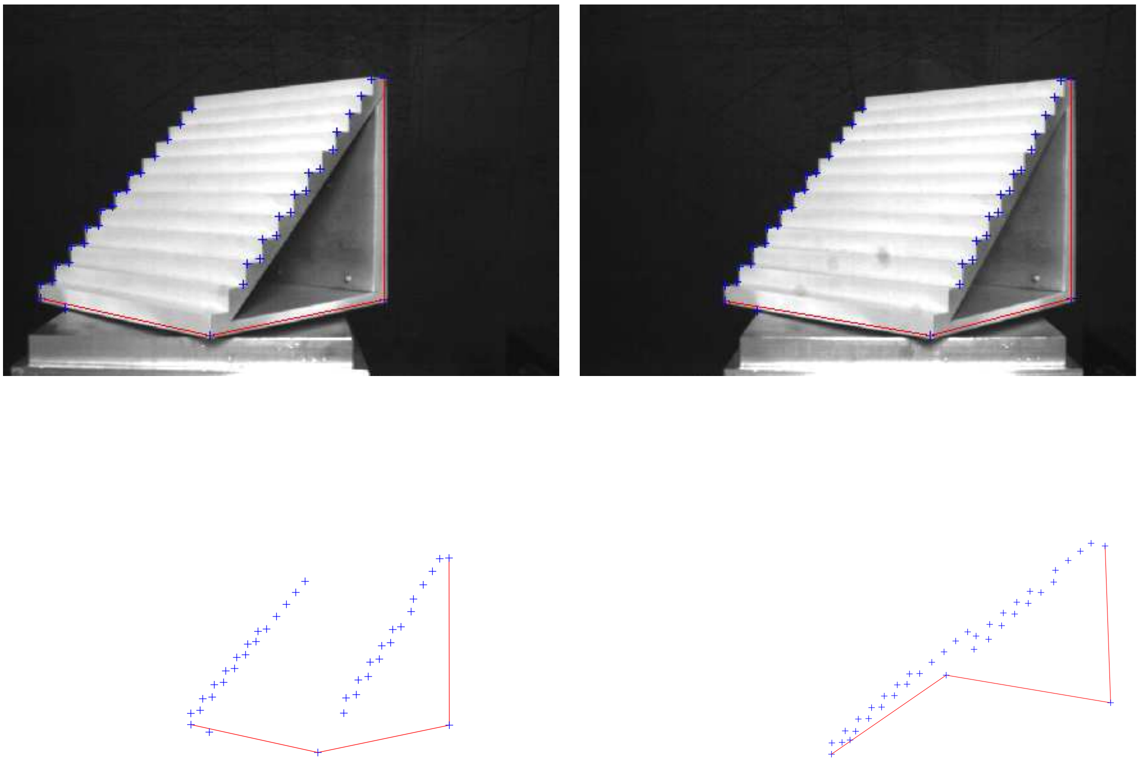

Figure 40: First and last frame of “Stairs” sequence, with tracked features superimposed (top row). Reconstructed object, from different viewpoints (bottom row).

Figure 40 shows the reconstructed structure, up to a scale factor, from two different points of view. The reconstruction appears to be visually correct. Indeed, the average error on right angles is about 4%. Knowing the length of the object, we recovered the unknown scale factor. By comparing the other dimensions with the actual dimensions of the object, we measured an error of 1.8% on the height and of 5% on the depth.

5.8 Conclusions

Structure from motion consist in recovering scene structure from a sequence of pictures of it taken with a moving camera of which we know the intrinsic parameters. We take the so-called discrete approach to the problem. We implemented a structure from motion algorithm composed from the following steps: compute the essential matrix from point matches; factorize the motion out of the matrix; use the motion parameters as the initial estimate of an iterative algorithm; use the estimated motion together with intrinsic parameters to reconstruct 3-D points coordinates. The algorithm proved up to the task both in a synthetic and a real case. In the latter we used the correspondences provided by our robust tracker, which is described in the next chapter.

Chapter 6

Feature Tracking

In this chapter we will address the problem of tracking features over time, by analyzing a small number of snapshots taken at different time instants. In the previous chapters we assumed that correspondences between points in consecutive frames were given, and we studied the problem of estimating the displacement of the camera. Here we address the problem of *computing correspondences.* We extend the well-known Shi-Tomasi-Kanade tracker by introducing an *automatic* scheme for rejecting spurious features. We employ a simple and efficient outlier rejection rule, called X84, and prove that its theoretical assumptions are satisfied in the feature tracking scenario. Experiments with real and synthetic images shows the benefits introduced by the algorithm.

6.1 Introduction

Much work on structure from motion has assumed that correspondences through a sequence of images could be recovered, as we did in Chapter 5. Feature tracking finds matches by selecting image features and tracks these as they move from frame to frame. It can be seen as an instance of the general problem of computing the *optical flow*, that is, the vector's field that describes how the image is changing with time, at relatively sparse image positions [104, 9, 20]. The methods based on the detection of two dimensional features (such as corners) have the advantage that the full optical flow is known at every measurement position, because they do not suffer from the aperture problem effect (a discussion on this subject can be found in [149]).

Works on tracking of two dimensional features include [89, 8, 23, 127, 170].
Robust tracking means detecting automatically unreliable matches, or *outliers*, over an image sequence (see [103] for a survey of robust methods in computer vision). Recent examples of such robust algorithms include [144], which identifies tracking outliers while estimating the fundamental matrix, and [143], which adopts a RANSAC [39] approach to eliminate outliers for estimating the trifocal tensor. Such approaches increase the computational cost of tracking significantly, as they are based on iterative algorithms.

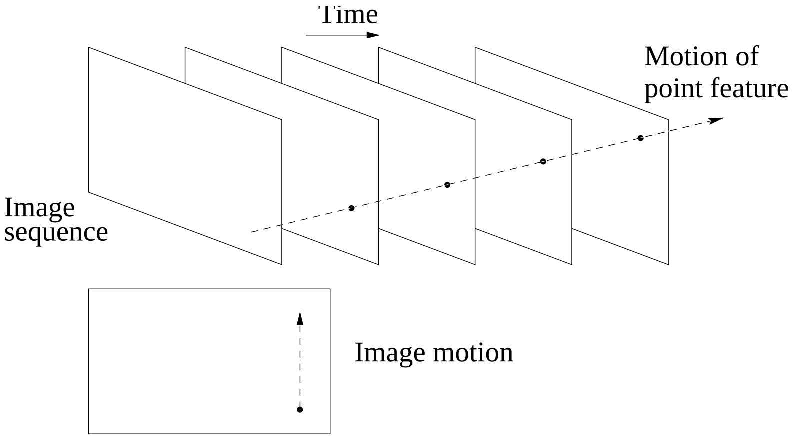

Figure 41: Feature tracking.

This chapter concentrates on the well-known Shi-Tomasi-Kanade tracker, and proposes a robust version based on an efficient outlier rejection scheme. Building on results from [89], Tomasi and Kanade [137] introduced a feature tracker based on SSD matching and assuming translational frame-to-frame displacements. Subsequently, Shi and Tomasi [128] proposed an *affine model*, which proved adequate for region matching over longer time spans. Their system classified a tracked feature as *good* (reliable) or *bad* (unreliable) according to the residual of the match between the associated image region in the first and current frames; if the residual exceeded a user-defined threshold, the feature was rejected. Visual inspection of results demonstrated good discrimination between good and bad features, but the authors did not specify how to reject bad features *automatically*.
This is the problem that our method solves. We extend the Shi-Tomasi-Kanade tracker (Section 6.2) by introducing an *automatic* scheme for rejecting spurious

features. We employ a simple, efficient, model-free outlier rejection rule, called *X84*, and prove that its assumptions are satisfied in the feature tracking scenario (Section 6.3). Our RobustTracking algorithm is summarized in Section 6.4. Experiments with real and synthetic images confirm that our algorithm makes good features to track better, in the sense that outliers are located reliably (Section 6.5). We illustrate quantitatively the benefits introduced by the algorithm with the example of fundamental matrix estimation. Image sequences with results and the source code of the robust tracker are available on line (http://www.dimi.uniud.it/˜fusiello/demo-rtr/).

6.2 The Shi-Tomasi-Kanade tracker

In this section the Shi-Tomasi-Kanade tracker [128, 137] will be briefly described. Consider an image sequence $I(\mathbf{x}, t)$, where $\mathbf{x} = [u, v]^\top$ are the coordinates of an image point. If the time sampling frequency (that is, the frame rate) is sufficiently high, we can assume that small image regions undergo a geometric transformation, but their intensities remain unchanged:

$$I(\mathbf{x}, t) = I(\delta(\mathbf{x}), t + \tau), \tag{112}$$

where $\delta(\cdot)$ is the *motion field*, specifying the *warping* that is applied to image points. The fast-sampling hypothesis allows us to approximate the motion with a translation, that is,

$$\delta(\mathbf{x}) = \mathbf{x} + \mathbf{d}, \tag{113}$$

where $\mathbf{d}$ is a displacement vector. The tracker's task is to compute $\mathbf{d}$ for a number of automatically selected point features for each pair of successive frames in the sequence. As the image motion model is not perfect, and because of image noise, (112) is not satisfied exactly. The problem is then finding the displacement $\mathbf{d}$ which minimizes the SSD residual

$$\epsilon = \sum_{W} \left[I(\mathbf{x} + \mathbf{d}, t + \tau) - I(\mathbf{x}, t) \right]^2, \tag{114}$$

where W is a given feature window centered on the point $\mathbf{x}$. In the following we will solve this problem by means of a Newton-Raphson iterative search.

Thanks to the fast-sampling assumption, we can approximate $I(\mathbf{x}+\mathbf{d}, t+\tau)$ with its first-order Taylor expansion:

$$I(\mathbf{x}+\mathbf{d}, t+\tau) \approx I(\mathbf{x}, t) + \nabla I(\mathbf{x}, t)^\top \mathbf{d} + I_t(\mathbf{x}, t)\tau, \tag{115}$$

where $\nabla I^\top = [I_u, I_v] = [\partial I/\partial u, \partial I/\partial v]$ and $I_t = \partial I/\partial t$. We can then rewrite the residual (114) as

$$\epsilon \approx \sum_W (\nabla I(\mathbf{x}, t)^\top \mathbf{d} + I_t(\mathbf{x}, t)\tau)^2. \tag{116}$$

To minimize the residual (116), we differentiate it with respect to the unknown displacement $\mathbf{d}$ and set the result to zero, obtaining the linear system:

$$\mathbf{C}\mathbf{d} = \mathbf{g}, \tag{117}$$

where

$$\mathbf{C} = \sum_W \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix} \tag{118}$$

$$\mathbf{g} = -\tau \sum_W I_t \, [I_u \; I_v]^\top . \tag{119}$$

If $\mathbf{d}_k = \mathbf{C}^{-1}\mathbf{g}$ is the displacement estimate at iteration k, and assuming a unit time interval between frames, the algorithm for minimizing (116) is the following:

$$\begin{cases} \mathbf{d}_0 = \mathbf{0} \\ \mathbf{d}_{k+1} = \mathbf{d}_k + \mathbf{C}^{-1} \sum_W \Big[(I(\mathbf{x}, t) - I(\mathbf{x}+\mathbf{d}_k, t+1)) \, \nabla I(\mathbf{x}, t) \Big] \end{cases} .$$

6.2.1 Feature extraction

A feature is defined as a region that can be easily tracked from one frame to the other. In this framework, a feature can be tracked reliably if a numerically stable solution to (117) can be found, which requires that $\mathbf{C}$ is well-conditioned and its entries are well above the noise level. In practice, since the larger eigenvalue is bound by the maximum allowable pixel value, the requirement is that the smaller eigenvalue must be sufficiently large. Calling λ_1 and λ_2 the eigenvalues of $\mathbf{C}$, we accept the corresponding feature if

$$\min(\lambda_1, \lambda_2) > \lambda_t \tag{120}$$

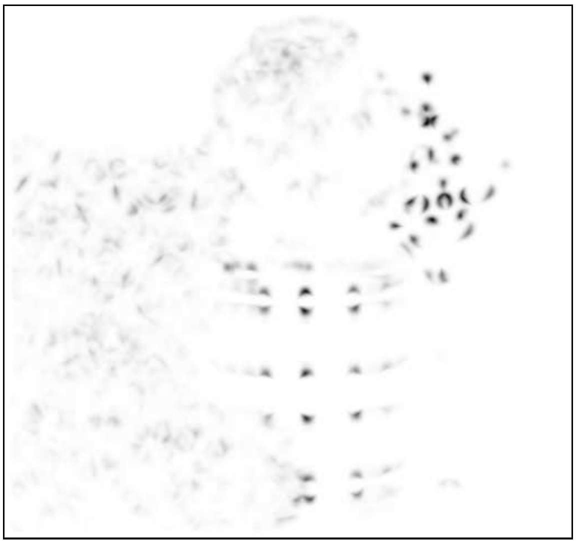

Figure 42: Value of $\min(\lambda_1, \lambda_2)$ for the first frame of ‘Artichoke”. Window size is 15 pixels. Darker points have an higher minimum eigenvalue.

where λ_t is a user-defined threshold [128].

This algebraic characterization of “trackable” features has an interesting interpretation, as they turns out to be *corners*, that is image features characterized by an intensity discontinuity in two directions. Since the motion of an image feature can be measured only in its projection on the brightness gradient (aperture problem), corners are the features whose motion can be measured.

Discontinuity can be detected, for instance, using normalized cross-correlation, which measures how well an image patch matches other portions of the image as it is shifted from its original location. A patch which has a well-defined peak in its auto-correlation function can be classified as a corner. Let us compute the change in intensity, as the sum of squared differences, in the direction $\mathbf{h}$ for a patch W centered in $\mathbf{x} = (u, v)$:

$$E_h(\mathbf{x}) = \sum_{\mathbf{d} \in W} \left(I(\mathbf{x} + \mathbf{d}) - I(\mathbf{x} + \mathbf{d} + \mathbf{h}) \right)^2 \tag{121}$$

Using the Taylor series expansion truncated to the linear term:

$$\begin{aligned} E_h(\mathbf{x}) &\approx \sum_{\mathbf{d}\in W} \left(\nabla I(\mathbf{x}+\mathbf{d})^\top \mathbf{h}\right)^2 \\ &= \sum_{\mathbf{d}\in W} \mathbf{h}^\top (\nabla I(\mathbf{x}+\mathbf{d}))(\nabla I(\mathbf{x}+\mathbf{d}))^\top \mathbf{h} \\ &= \sum_{\mathbf{d}\in W} \mathbf{h}^\top \begin{pmatrix} I_u^2 & I_u\, I_v \\ I_u\, I_v & I_v^2 \end{pmatrix} \mathbf{h} \\ &= \mathbf{h}^\top \left(\sum_{\mathbf{d}\in W} \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix} \right) \mathbf{h}. \end{aligned} \tag{122}$$

The change in intensity around $\mathbf{x}$ is therefore given by

$$E_h(\mathbf{x}) = \mathbf{h}^\top \mathbf{C}\, \mathbf{h} \tag{123}$$

where $\mathbf{C}$ is just the matrix defined in (118). Elementary eigenvector theory tells us that, since $||\mathbf{h}|| = 1$, then

$$\lambda_1 < E_h(\mathbf{x}) < \lambda_2, \tag{124}$$

where λ_1 and λ_2 are the eigenvalues of $\mathbf{C}$. So, if we try every possible orientation $\mathbf{h}$, the maximum change in intensity we will find is λ_2, and the minimum value is λ_1. We can therefore classify the structure around each pixel by looking at the eigenvalues of $\mathbf{C}$:

- no structure: $\lambda_1 \approx \lambda_2 \approx 0$;
- edge: $\lambda_1 \approx 0$, $\lambda_2 \gg 0$;
- corner: λ_1 e λ_2 both large and distinct.

Hence, the features selected according to criterion criterion (120) are to be interpreted as corners. Indeed, this method is very closely related to some classical corner detectors, such as [105, 109, 57].

Figure 42 shows the value of the minimum eigenvalue for the first frame of the "Artichoke" sequence (see Section 6.5).

6.2.2 Affine model

The translational model cannot account for certain transformations of the feature window, for instance rotation, scaling, and shear. An *affine motion field* is a more accurate model [128], that is,

$$\delta(\mathbf{x}) = \mathbf{M}\mathbf{x} + \mathbf{d}, \tag{125}$$

where $\mathbf{d}$ is the displacement, and $\mathbf{M}$ is a 2×2 matrix accounting for affine warping, and can be written as $\mathbf{M} = \mathbf{1} + \mathbf{D}$, with $\mathbf{D} = [d_{ij}]$ a deformation matrix and $\mathbf{1}$ the identity matrix. Similarly to the translational case, one estimates the motion parameters, $\mathbf{D}$ and $\mathbf{d}$, by minimizing the residual

$$\epsilon = \sum_{W} \left[I(\mathbf{M}\mathbf{x} + \mathbf{d}, t + \tau) - I(\mathbf{x}, t) \right]^2. \tag{126}$$

By plugging the first-order Taylor expansion of $I(\mathbf{M}\mathbf{x} + \mathbf{d}, t + \tau)$ into (126), and imposing that the derivatives with respect to $\mathbf{D}$ and $\mathbf{d}$ are zero, we obtain the linear system

$$\mathbf{B}\mathbf{z} = \mathbf{f}, \tag{127}$$

in which $\mathbf{z} = [d_{11}\ d_{12}\ d_{21}\ d_{22}\ d_1\ d_2]^\top$ contains the unknown motion parameters, and

$$\mathbf{f} = -\tau \sum_{W} I_t \left[uI_u\ uI_v\ vI_u\ vI_v\ I_u\ I_v \right]^\top,$$

with

$$\mathbf{B} = \sum_{W} \begin{bmatrix} \mathbf{U} & \mathbf{V} \\ \mathbf{V}^\top & \mathbf{C} \end{bmatrix},$$

$$\mathbf{U} = \begin{bmatrix} u^2I_u^2 & u^2I_uI_v & uvI_u^2 & uvI_uI_v \\ u^2I_uI_v & u^2I_v^2 & uvI_uI_v & uvI_v^2 \\ uvI_u^2 & uvI_uI_v & v^2I_u^2 & v^2I_uI_v \\ uvI_uI_v & uvI_v^2 & v^2I_uI_v & v^2I_v^2 \end{bmatrix},$$

$$\mathbf{V}^\top = \begin{bmatrix} uI_u^2 & uI_uI_v & vI_u^2 & vI_uI_v \\ uI_uI_v & uI_v^2 & vI_uI_v & vI_v^2 \end{bmatrix}.$$

Again, (126) is solved for $\mathbf{z}$ using a Newton-Raphson iterative scheme.

If frame-to-frame affine deformations are negligible, the pure translation model is preferable (the matrix $\mathbf{M}$ is assumed to be the identity). The affine model is used for comparing features between frames separated by significant time intervals to monitor the quality of tracking.

6.3 Robust monitoring

In order to monitor the quality of the features tracked, the tracker checks the residuals between the first and the current frame: high residuals indicate bad features which must be rejected. Following [128], we adopt the affine model, as a pure translational model would not work well with long sequences: too many good features are likely to undergo significant rotation, scaling or shearing, and would be incorrectly discarded. Non-affine warping, which will yield high residuals, is caused by occlusions, perspective distortions and strong intensity changes (e.g. specular reflections). This section introduces our method for selecting a robust rejection threshold *automatically.*

6.3.1 Distribution of the residuals

We begin by establishing which distribution is to be expected for the residuals when comparing good features, i.e., almost identical regions. We assume that the intensity $I(\delta(\mathbf{x}), t)$ of each pixel in the current-frame region is equal to the intensity of the corresponding pixel in the first frame $I(\mathbf{x}, 0)$ plus some Gaussian noise $n \equiv \eta(0, 1)$[1]. Hence

$$I(\delta(\mathbf{x}), t) - I(\mathbf{x}, 0) \equiv \eta(0, 1).$$

Since the square of a Gaussian random variable has a chi-square distribution, we obtain

$$\left[I(\delta(\mathbf{x}), t) - I(\mathbf{x}, 0)\right]^2 \equiv \chi^2(1).$$

[1] $\equiv$ means that the variable to the left has the probability distribution specified to the right.

The sum of n chi-square random variables with one degree of freedom is distributed as a chi-square with n degrees of freedom (as it is easy to see by considering the moment-generating functions). Therefore, the residual computed according to (114) over a $N \times N$ window W is distributed as a chi-square with N^2 degrees of freedom:

$$\epsilon = \sum_{W} \left[I(\delta(\mathbf{x}), t) - I(\mathbf{x}, 0) \right]^2 \equiv \chi^2(N^2). \tag{128}$$

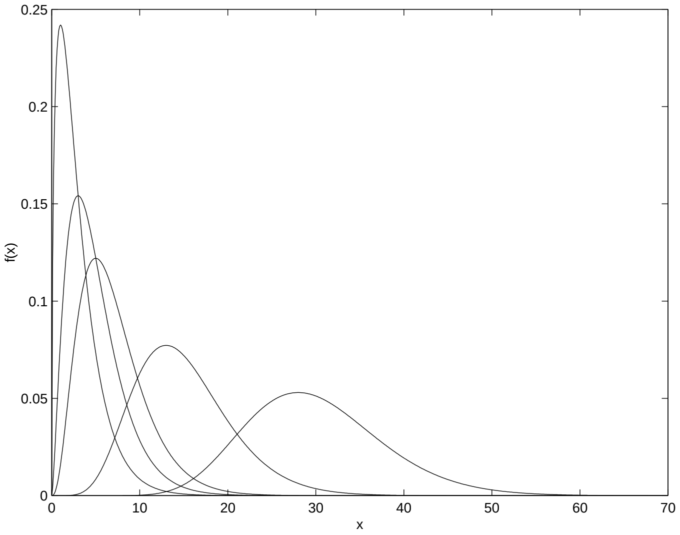

Figure 43: Chi-square density functions with 3,5,7,15 and 30 degrees of freedom (from left to right).

As the number of degrees of freedom increases, the chi-square distribution approaches a Gaussian, which is in fact used to approximate the chi-square with more than 30 degrees of freedom. Therefore, since the window W associated to each feature is at least 7×7, we can safely assume a Gaussian distribution of the residual for the good features:

$$\epsilon \equiv \eta(N^2, 2N^2).$$

6.3.2 The X84 rejection rule

When the two regions over which we compute the residual are bad features (that is, they are not warped by an affine transformation), the residual is not a sample from the Gaussian distribution of good features: it is an *outlier*. Hence, the detection of bad features reduces to a problem of outlier detection. This is equivalent to the problem of estimating the mean and variance of the underlying Gaussian distribution from the corrupted data ϵ_i, the residuals (given by (114)) between the i-th feature in the last frame and the same feature in the first frame. To do this, we employ a simple but effective model-free rejection rule, X84 [55], which use robust estimates for location and scale to set a rejection threshold. The median is a robust location estimator, and the Median Absolute Deviation (MAD), defined as

$$\text{MAD} = \underset{i}{\text{med}}\{|\epsilon_i - \underset{j}{\text{med}}\ \epsilon_j|\}. \tag{129}$$

is a robust estimator of the scale (i.e., the spread of the distribution). It can be seen that, for symmetric (and moderately skewed) distributions, the MAD coincides with the *interquartile range*:

$$\text{MAD} = \frac{\xi_{3/4} - \xi_{1/4}}{2}, \tag{130}$$

where ξ_q is the qth quantile of the distribution (for example, the median is $\xi_{1/2}$). For normal distributions we infer the standard deviation from

$$\text{MAD} = \Phi^{-1}(3/4)\sigma \approx 0.6745\sigma. \tag{131}$$

The X84 rule prescribes to reject values that are more than k Median Absolute Deviations away from the median. A value of k=5.2, under the hypothesis of Gaussian distribution, is adequate in practice, as it corresponds to about 3.5 standard deviations, and the range $[\mu - 3.5\sigma, \mu + 3.5\sigma]$ contains more than the 99.9% of a Gaussian distribution . The rejection rule X84 has a breakdown point of 50%: any majority of the data can overrule any minority.

6.3.3 Photometric normalization

Our robust implementation of the Shi-Tomasi-Kanade tracker incorporates also a *normalized* SSD matcher for residual computation. This limits the effects of intensity changes between frames, by subtracting the average grey level (μ_J, μ_I) and dividing by the standard deviation (σ_J, σ_I) in each of the two regions considered:

$$\epsilon = \sum_{W} \left[\frac{J(\mathbf{Mx}+\mathbf{d})-\mu_J}{\sigma_J} - \frac{I(\mathbf{x})-\mu_I}{\sigma_I} \right]^2, \tag{132}$$

where $J(\cdot)=I(\cdot, t+1)$, $I(\cdot)=I(\cdot, t)$.

It can be easily seen that this normalization is sufficient to compensate for intensity changes modeled by $J(\mathbf{Mx}+\mathbf{d}) = \alpha I(\mathbf{x}) + \beta$. A more elaborate normalization is described in [25], whereas [54] reports a modification of the Shi-Tomasi-Kanade tracker based on explicit photometric models.

6.4 Summary of the RobustTracking algorithm

The RobustTracking algorithm can be summarized as follows:

1. given an image sequence;
2. filter the sequence with a Gaussian kernel in space and time (for the selection of the scale of the kernel, see [18]);
3. select features to be tracked according to (120);
4. register features in each pair of consecutive frames in the sequence, using translational warping (113);
5. in the last frame of the sequence, compute the residuals between this and the first frame, for each feature, using affine warping (125);
6. reject outlier features according to the X84 rule (120).

The decision of which frame is deemed to be the *last* one is left open; the only, obvious, constraint is that a certain fraction of the features present in the first frame should be still visible in the last. On the other hand, monitoring cannot be done at every frame, because the affine warping would not be appreciable.

6.5 Experimental results

We evaluated our tracker in a series of experiments, of which we report the most significant ones.

"Platform" (Figure 44, 256×256 pixels). A 20-frame synthetic sequence, courtesy of the Computer Vision Group, Heriot-Watt University, simulating a camera rotating in space while observing a subsea platform sitting on the seabed (real seabed acquired by a sidescan sonar, rendered as an intensity image, and texture-mapped onto a plane).

"Hotel" (Figure 45, 480 × 512 pixels). The well-known Hotel sequence from the CMU VASC Image Database (59 frames). A static scene observed by a moving camera rotating and translating.

"Stairs" (Figure 48, 512 × 768 pixels). A 60-frame sequence of a white staircase sitting on a metal base and translating in space, acquired by a static camera. The base is the platform of a translation stage operated by a step-by-step motor under computer control (courtesy of F. Isgrò, Heriot-Watt University).

"Artichoke" (Figure 49, 480 × 512 pixels). A 99-frame sequence, the most complex one shown here (see later on). The camera is translating in front of the static scene. This sequence was used by [138].

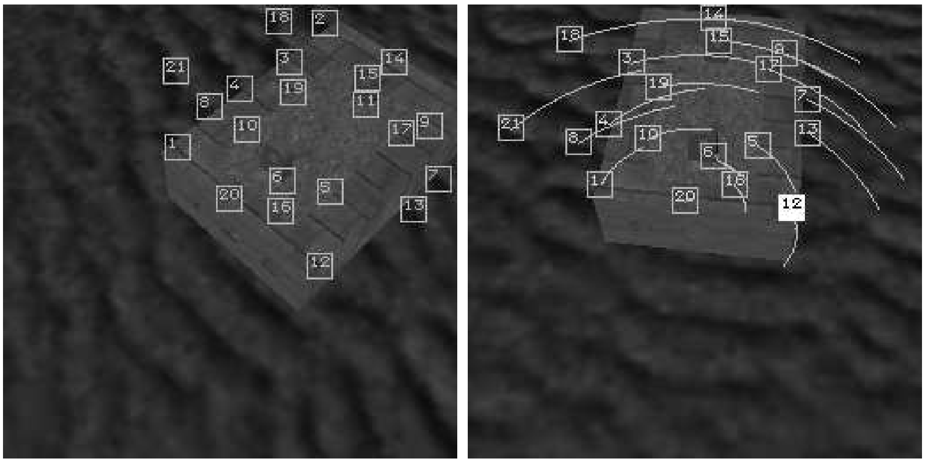

Figure 44: First (left) and last frame of the "Platform" sequence. In the last frame, filled windows indicate features rejected by the robust tracker.

"Platform" is the only synthetic sequence shown here. No features become occluded, but notice the strong effects of the coarse spatial resolution on straight lines. We

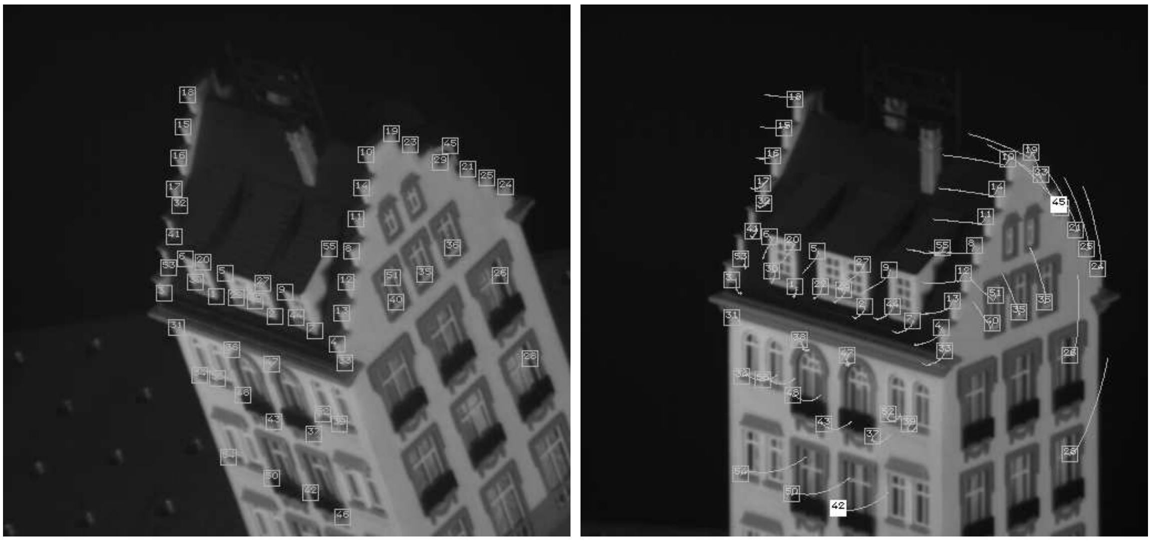

Figure 45: First (left) and last frame of the "Hotel" sequence. In the last frame, filled windows indicate features rejected by the robust tracker.

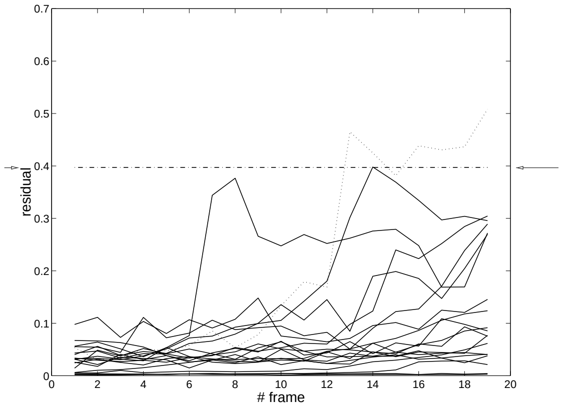

Figure 46: Residuals magnitude against frame number for "Platform". The arrows indicate the threshold set automatically by X84 (0.397189).

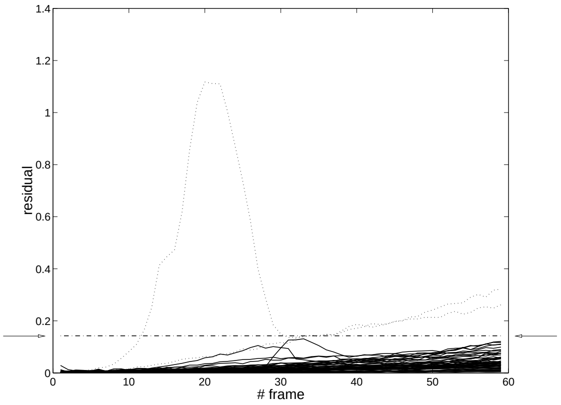

Figure 47: Residuals magnitude against frame number for "Hotel". The arrows indicate the threshold set automatically by X84 (0.142806).

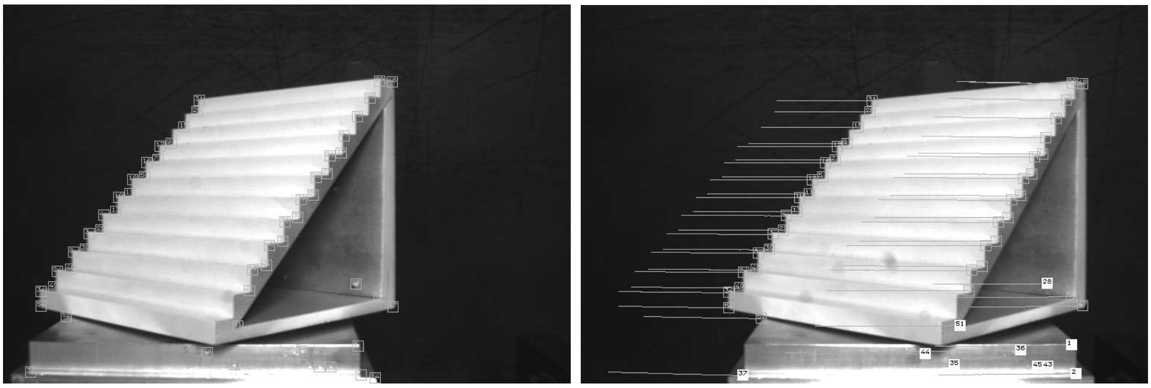

Figure 48: First (left) and last frame of the “Stairs” sequence. In the last frame, filled windows indicate features rejected by the robust tracker.

plotted the residuals of all features against the frame number (Figure 46). All features stay under the threshold computed automatically by X84, apart from one that is corrupted by the interference of the background. In “Stairs”, some of the features picked up in the first frame are specular reflections from the metal platform, the intensity of which changes constantly during motion. The residuals for such features become therefore very high (Figure 50). All these features are rejected correctly. Only one good feature is dropped erroneously (the bottom left corner of the internal triangle), because of the strong intensity change of the inside of the block. In the “Hotel” sequence (Figure 47), all good features but one are preserved. The one incorrect rejection (bottom center, corner of right balcony) is due to the warping caused by the camera motion, too large to be accommodated by the affine model. The only spurious feature present (on the right-hand side of the stepped-house front) is rejected correctly. All features involved in occlusions in the “Artichoke” sequence (Figure 51) are identified and rejected correctly. Four good features out of 54 are also rejected (on the signpost on the right) owing to a marked contrast change in time between the pedestrian figure and the signpost in the background.

In our tests on a SPARCServer 10 running Solaris 2.5, the initial feature extraction phase took 38s for “Platform” and 186s for “Artichoke”, with a 15×15 window. The tracking phase took on average 1.6s per frame, independently from frame dimensions. As expected, extraction is very computationally demanding, since the eigenvalues of the $\mathbf{C}$ matrix are to be computed *for each pixel*. However, this process can implemented on a parallel architecture, thereby achieving real-time performances (30Hz), as reported in [12].

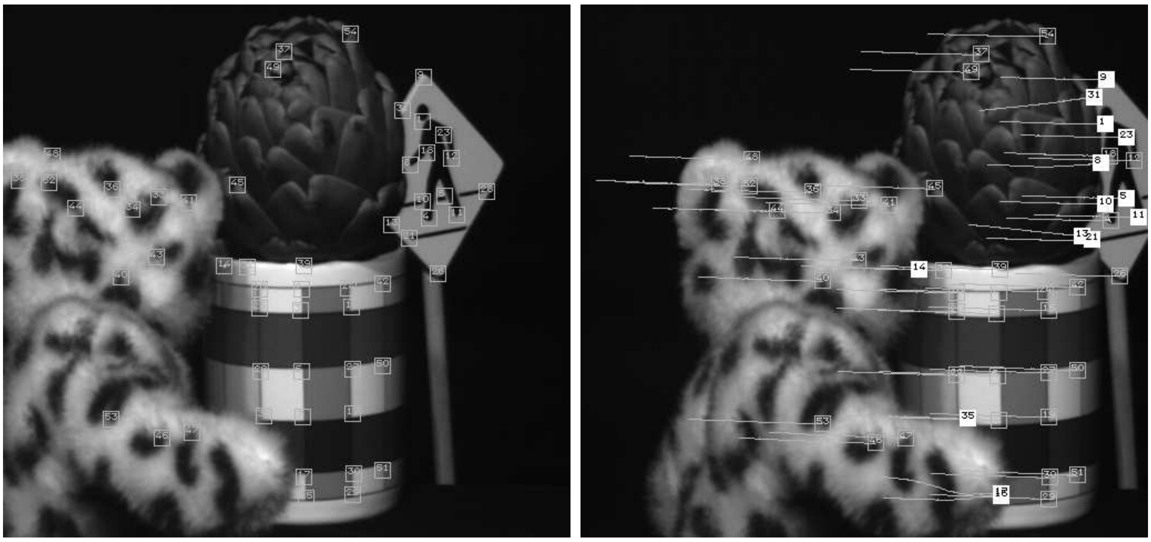

Figure 49: First (left) and last frame of the “Artichoke” sequence. In the last frame, filled windows indicate features rejected by the robust tracker.

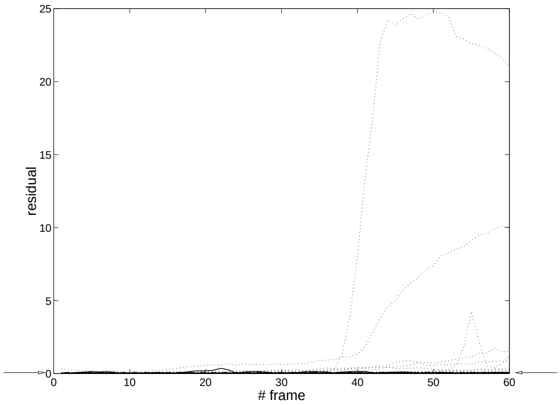

Figure 50: Residuals magnitude against frame number for “Stairs”. The arrows indicate the threshold set automatically by X84 (0.081363) .

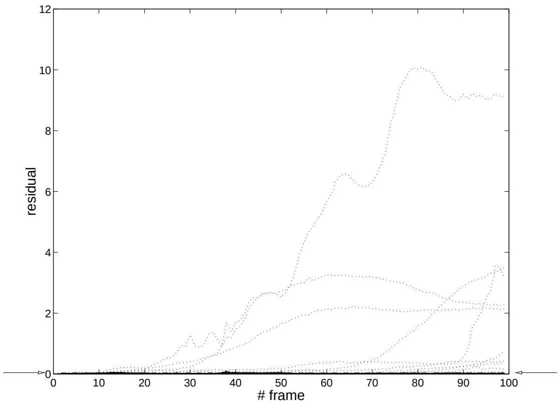

Figure 51: Residuals magnitude against frame number for “Artichoke”. The arrows indicate the threshold set automatically by X84 (0.034511).

Quantifying improvement: an example

To illustrate quantitatively the benefits of our robust tracker, we used the feature tracked by robust and non-robust versions of the tracker to compute the fundamental matrix (see Chapter 5) between the first and last frame of each sequence, then computed the RMS distance of the tracked points from the corresponding epipolar lines, using the 8-point algorithm (Section 5.4.1): if the epipolar geometry is estimated exactly, all points should lie on epipolar lines. The results are shown in Table 3. The robust tracker brings always a decrease in the RMS distance. Notice the limited decrease and high residual for "Platform"; this is due to the significant spatial quantization and smaller resolution, which worsens the accuracy of feature localization.

	Artichoke	Hotel	Stairs	Platform
All	1.40	0.59	0.66	1.49
X84	0.19	0.59	0.15	1.49

Table 3: RMS distance of points from epipolar lines. The first row gives the distance using all the features tracked (non-robust tracker), the second using only the features kept by X84 (robust tracker).

6.6 Conclusions

We have presented a robust extension of the Shi-Tomasi-Kanade tracker, based on the X84 outlier rejection rule. The computational cost is much less than that of schemes based on robust regression and random sampling like RANSAC or Least Median of Squares [103, 143], yet experiments indicate excellent reliability in the presence of non-affine feature warping (most right features preserved, all wrong features rejected). Our experiments have also pointed out the pronounced sensitivity of the Shi-Tomasi-Kanade tracker to illumination changes.

Chapter 7

Autocalibration

This chapter provides a review on techniques for computing a three-dimensional model of a scene from a single moving camera, with unconstrained motion and unknown parameters. In the classical approach, called *autocalibration* or *self-calibration*, camera motion and parameters are recovered first, using rigidity; then structure is easily computed. Recently, new methods based on the idea of *stratification* have been proposed. They upgrade a *projective reconstruction*, achievable from correspondences only, to a *Euclidean* one, by exploiting all the available constraints.

7.1 Introduction

In Chapter 5 we assumed that the intrinsic parameters of the camera (focal length, image center and aspect ratio) were known, and showed how to compute camera motion and scene structure.

However, there are situations wherein the intrinsic parameters are unknown (e.g., if the image sequence comes from a pre-recorded video tape) or off-line calibration is impracticable (e.g, if the camera is mounted on an unmanned vehicle which cannot be distracted from operation if calibration is lost). In these cases the only information one can exploit is contained in the video sequence itself.

Yet, some assumptions are necessary to make the problem tractable. We will focus on the classical case of a *single camera with constant but unknown intrinsic parameters and unknown motion.* Other approaches restrict the motion [3, 59, 154] or assume a rigidly moving stereo rig [169].

In the next section (7.2), we will derive again the fundamental matrix and then (Section 7.3) introduce the homography of a plane, which will be used later in this chapter. In Section 7.4 the reconstruction problem will be formulated and some highlights on projective reconstruction technique will be given. Section 7.5 will introduce autocalibration and stratification methods for upgrading to Euclidean reconstruction. In Section 7.6 the "classical" autocalibration approach, based on Kruppa equations, will be outlined. Stratification methods will be described in some details in Section 7.7. Applicability of the methods will be discussed in Section 7.8. Finally (Section 7.9), conclusions will be drawn.

7.2 Uncalibrated epipolar geometry

In Section 3.4 we saw how epipolar geometry is used in the calibrated case to constraint the search for conjugate points. In Section 5.2 we derived the Longuett-Higgins equation, which gives the epipolar geometry when intrinsic parameters are known. Here we will derive again the epipolar geometry in the uncalibrated case.

Let us consider the case of two cameras. If we take the first camera reference frame as the world reference frame, we can write the two following general camera matrices (see Chapter 2):

$$\tilde{\mathbf{P}} = \mathbf{A}[\mathbf{I}|\mathbf{0}] = [\mathbf{A}|\mathbf{0}] \tag{133}$$

$$\tilde{\mathbf{P}}' = \mathbf{A}'[\mathbf{R}|\mathbf{t}] \tag{134}$$

Let

$$\tilde{\mathbf{m}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{w}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{135}$$

the projection equations are

$$\kappa \tilde{\mathbf{m}} = \tilde{\mathbf{P}} \tilde{\mathbf{w}}, \tag{136}$$

and

$$\kappa' \tilde{\mathbf{m}}' = \tilde{\mathbf{P}}' \tilde{\mathbf{w}}. \tag{137}$$

where κ is the projective depth, that is the distance of points from the focal plane of the camera, if $\tilde{\mathbf{P}}$ is suitably normalized (see Section 2.2.3).
From (136) and (134) we obtain:

$$\kappa'\tilde{\mathbf{m}}' = \mathbf{A}'[\mathbf{R}|\mathbf{t}]\tilde{\mathbf{w}} = \mathbf{A}'[\mathbf{R}|\mathbf{t}]\left(\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}\right) = \mathbf{A}'\mathbf{R}\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \mathbf{A}'\mathbf{t}, \tag{138}$$

and from (137) and (133) we obtain:

$$\kappa\mathbf{A}^{-1}\tilde{\mathbf{m}} = [\mathbf{I}|\mathbf{0}]\ \tilde{\mathbf{w}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{139}$$

Substituting the latter in (138) yields

$$\kappa'\tilde{\mathbf{m}}' = \kappa\mathbf{A}'\mathbf{R}\mathbf{A}^{-1}\tilde{\mathbf{m}} + \mathbf{A}'\mathbf{t} = \kappa\mathbf{H}_\infty\tilde{\mathbf{m}} + \mathbf{e}' \tag{140}$$

where $\mathbf{H}_\infty = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}$ (the reason for this notation will be manifest in the following), and $\mathbf{e}' = \mathbf{A}'\mathbf{t}$ is the epipole in the second camera. Similarly, the epipole in the first camera is $\mathbf{e} = -\mathbf{A}\mathbf{R}\mathbf{t}$.
Equation[1] (140) links the left and right projections of the same point $\mathbf{w}$. If we know the conjugate pair $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{m}}'$, we can solve for the depth κ and κ'. Vice versa, if κ e κ' are known we can locate $\tilde{\mathbf{m}}'$ given $\tilde{\mathbf{m}}$.
Equation (140) says that $\tilde{\mathbf{m}}'$ lies on the line going trough $\mathbf{e}'$ and the point $\mathbf{H}_\infty\tilde{\mathbf{m}}$. In projective coordinates the collinearity of these three points can be expressed with the relation:

$$\tilde{\mathbf{m}}'^\top(\mathbf{e}' \wedge \mathbf{H}_\infty\tilde{\mathbf{m}}) = 0, \tag{141}$$

or

$$\tilde{\mathbf{m}}'^\top\mathbf{F}\tilde{\mathbf{m}} = 0. \tag{142}$$

where

$$\mathbf{F} = [\mathbf{e}']_\wedge\mathbf{H}_\infty \tag{143}$$

[1] Compare to (53)

is the *fundamental matrix*. From (55) we can see that $\tilde{\mathbf{m}}'$ belongs to the line $\mathbf{F}\tilde{\mathbf{m}}$ in the second image, which is called the *epipolar line* of $\tilde{\mathbf{m}}$. It is easy to see that $\mathbf{e}'^{\top}\mathbf{F} = \mathbf{0}$, meaning that all the epipolar lines contain the point $\mathbf{e}'$, which is called the *epipole*.

Since $\mathbf{F}\mathbf{e} = \mathbf{F}^{\top}\mathbf{e}' = \mathbf{0}$, the rank of $\mathbf{F}$ is in general two and, being defined up to a scale factor, depends upon seven parameters. The only geometrical information that can be computed from pairs of images is the fundamental matrix. Its computation from point correspondences has been addressed in Section 5.4.1.

The essential matrix (Section 5.2) is linked to the fundamental matrix: it can be obtained from the latter as long as the intrinsic parameters are known. Indeed, (142) is equivalent to

$$\begin{aligned}
\tilde{\mathbf{m}}'^{\top}[\mathbf{A}'\mathbf{t}]_{\wedge}\mathbf{A}'\mathbf{R}\mathbf{A}^{-1}\tilde{\mathbf{m}} = 0 &\iff \\
\tilde{\mathbf{m}}'^{\top}\det(\mathbf{A}')\mathbf{A}'^{-\top}[\mathbf{t}]_{\wedge}\mathbf{A}'^{-1}\mathbf{A}'\mathbf{R}\mathbf{A}^{-1}\tilde{\mathbf{m}} = 0 &\iff \\
\tilde{\mathbf{m}}'^{\top}\mathbf{A}'^{-\top}[\mathbf{t}]_{\wedge}\mathbf{R}\mathbf{A}^{-1}\tilde{\mathbf{m}} = 0 &\iff \\
(\mathbf{A}'^{-1}\tilde{\mathbf{m}}')^{\top}[\mathbf{t}]_{\wedge}\mathbf{R}(\mathbf{A}^{-1}\tilde{\mathbf{m}}) = 0,&
\end{aligned} \tag{144}$$

thanks to

$$[\mathbf{A}\mathbf{u}]_{\wedge} = \det(\mathbf{A})\mathbf{A}^{-\top}[\mathbf{u}]_{\wedge}\mathbf{A}^{-1}.$$

From (144) it is easy to see that

$$\mathbf{F} = \mathbf{A}'^{-\top}\mathbf{E}\mathbf{A}^{-1}. \tag{145}$$

7.3 Homography of a plane

Equation (140) can be specialized to the case of 3-D points lying on a plane. Let us take a plane Π with Cartesian equation $\mathbf{n}^{\top}\mathbf{w} = d$, that is

$$\mathbf{n}^{\top}\begin{bmatrix} x \\ y \\ z \end{bmatrix} - d = 0 \tag{146}$$

Substituting (139) in the latter yields:

$$\kappa\mathbf{n}^{\top}\mathbf{A}^{-1}\tilde{\mathbf{m}} - d = 0 \tag{147}$$

from which an expression for κ is obtained:

$$\kappa = \frac{d}{\mathbf{n}^\top \mathbf{A}^{-1}\tilde{\mathbf{m}}}. \tag{148}$$

Let us divide (140) by κ

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{m}}' = \mathbf{H}_\infty \tilde{\mathbf{m}} + \frac{\mathbf{e}'}{\kappa} \tag{149}$$

and substitute (148) for κ in the right-hand side, thereby obtaining

$$\begin{aligned}\frac{\kappa'}{\kappa}\tilde{\mathbf{m}}' &= \mathbf{H}_\infty \tilde{\mathbf{m}} + \frac{(\mathbf{n}^\top \mathbf{A}^{-1}\tilde{\mathbf{m}})\ \mathbf{e}'}{d} \\ &= \mathbf{H}_\infty \tilde{\mathbf{m}} + \frac{(\mathbf{e}'\ \mathbf{n}^\top \mathbf{A}^{-1})\tilde{\mathbf{m}}}{d} \\ &= \left(\mathbf{H}_\infty + \frac{\mathbf{e}'\ \mathbf{n}^\top \mathbf{A}^{-1}}{d}\right)\tilde{\mathbf{m}}.\end{aligned} \tag{150}$$

Therefore, given two views of a scene, there is a linear projective transformation (an *homography*, or *collineation*) relating the projection $\mathbf{m}$ of the point of a plane Π in the first view to its projection in the second view, $\mathbf{m}'$. This application is given by a 3×3 invertible matrix $\mathbf{H}_\Pi$ such that:

$$\tilde{\mathbf{m}}' \simeq \mathbf{H}_\Pi \tilde{\mathbf{m}}. \tag{151}$$

$\mathbf{H}_\Pi$ is the homography matrix for the plane Π. Hence, from (150) :

$$\mathbf{H}_\Pi = \mathbf{H}_\infty + \mathbf{e}'\frac{\mathbf{n}^\top}{d}\mathbf{A}^{-1}. \tag{152}$$

Since, by definition,

$$\mathbf{H}_\infty = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}, \tag{153}$$

by substituting in (152)we obtain:

$$\mathbf{H}_\Pi = \mathbf{A}'(\mathbf{R} + \mathbf{t}\frac{\mathbf{n}^\top}{d})\mathbf{A}^{-1}. \tag{154}$$

If one let $d \to \infty$ in (150), it becomes clear that $\mathbf{H}_\infty$ *is just the homography matrix for the infinity plane*, that maps vanishing points to vanishing points (that is the reason for the notation). Notice that it and depends only on the rotational component of the rigid displacement.

The same result could be obtained by observing that if a point goes to infinity, its relative depths κ and κ' grow to infinity as well, but their ratio tends to a constant value. Therefore in (149) the term with $\mathbf{e}'$ vanishes. Moreover, if we take a point at infinity as $\mathbf{w} = [x\ y\ z\ 0]^\top$ in (138), the term $\mathbf{A}'\mathbf{t}$ (the epipole) disappears.

7.4 Projective reconstruction

Consider a set of three-dimensional points viewed by N cameras with matrices $\{\tilde{\mathbf{P}}^i\}_{i=1\dots N}$. Let $\tilde{\mathbf{m}}_j^i \simeq \tilde{\mathbf{P}}^i\tilde{\mathbf{w}}_j$ be the (homogeneous) coordinates of the projection of the j-th point onto the i-th camera. The *reconstruction problem* can be cast in the following way: given the set of pixel coordinates $\{\tilde{\mathbf{m}}_j^i\}$, find the set of camera matrices $\{\tilde{\mathbf{P}}^i\}$ and the scene structure $\{\tilde{\mathbf{w}}_j\}$ such that

$$\tilde{\mathbf{m}}_j^i \simeq \tilde{\mathbf{P}}^i\tilde{\mathbf{w}}_j. \tag{155}$$

Without further restrictions we will, in general, obtain a projective reconstruction [32] defined up to an arbitrary projective transformation. Indeed, if $\{\tilde{\mathbf{P}}^i\}$ and $\{\tilde{\mathbf{w}}_j\}$ satisfy (155), also $\{\tilde{\mathbf{P}}^i\tilde{\mathbf{T}}\}$ and $\{\tilde{\mathbf{T}}^{-1}\tilde{\mathbf{w}}_j\}$ satisfy (155) for any 4×4 nonsingular matrix $\tilde{\mathbf{T}}$.

In the next section we will see how a projective reconstruction is obtained starting from the fundamental matrix, *in the case of two cameras.*

7.4.1 Reconstruction from two views

As seen in the previous section, the infinity plane homography gives rise to the following factorization of $\mathbf{F}$:

$$\mathbf{F} = [\mathbf{e}']_\wedge \mathbf{H}_\infty. \tag{156}$$

Note the similarity with the factorization $\mathbf{E} = [\mathbf{t}]_\wedge\mathbf{R}$, since $\mathbf{e}'$ depends only on the translation and $\mathbf{H}_\infty$ depends only on the rotation. Unfortunately the factorization is not unique, making it impossible to recover $\mathbf{H}_\infty$ from $\mathbf{F}$ directly. Indeed, if a matrix $\mathbf{M}$ satisfies $\mathbf{F} = [\mathbf{e}']_\wedge\mathbf{M}$, then also $\mathbf{M} + \mathbf{e}'\mathbf{v}^\top$ for any vector $\mathbf{v}$ yields a factorization, since

$$[\mathbf{e}']_\wedge(\mathbf{M} + \mathbf{e}'\mathbf{v}^\top) = [\mathbf{e}']_\wedge\mathbf{M} + [\mathbf{e}']_\wedge\mathbf{e}'\mathbf{v}^\top = [\mathbf{e}']_\wedge\mathbf{M}.$$

If a matrix $\mathbf{M}$ satisfies

$$\mathbf{F} = [\mathbf{e}']_\wedge\mathbf{M} \tag{157}$$

then $\mathbf{M}$ is said to be *compatible* with $\mathbf{F}$.

In particular, from (152) we obtain that every plane homography $\mathbf{H}_\Pi$ is compatible, that is:

$$\mathbf{F} = [\mathbf{e}']_\wedge \mathbf{H}_\Pi. \tag{158}$$

A special compatible matrix is the *epipolar projection* matrix $\mathbf{S}$[94], defined as follow:

$$\mathbf{S} = -\frac{1}{||\mathbf{e}'||}[\mathbf{e}']_\wedge \mathbf{F} \tag{159}$$

Although $\mathbf{S}$ is singular (it is not an homography), since it is compatible with the fundamental matrix, it can be interpreted as the correspondence induced by the plane $\Pi_{e'}$ that contains the optical center of the second camera and whose image on the second camera is the line represented by $\mathbf{e}'$.

This factorization allows us to compute a projective reconstruction from two views. Let $\mathbf{F}$ be the fundamental matrix for the two cameras. If $\mathbf{M}$ is compatible with $\mathbf{F}$, the following pair of PPMs:

$$\tilde{\mathbf{P}} = [\mathbf{I} \mid \mathbf{0}] \qquad \tilde{\mathbf{P}}' = [\mathbf{M} \mid \mathbf{e}'] \tag{160}$$

yield the given fundamental matrix, as can be easily verified. There are an infinite number of perspective projection matrices which all satisfy the epipolar geometry. A canonical representation [94] is obtained by using the epipolar projection matrix $\mathbf{S}$. Once the two PPMs have been instantiated, structure follows by triangulation (see Section 3.3).

7.4.2 Reconstruction from multiple views

In the case of more than two cameras, the projective reconstruction cannot be computed by simply applying the method just described to each pair of views. We would obtains, in general, a set of projective reconstructions linked to each other by an unknown projective transformation (i.e., each defines its own projective frame). Therefore, there would not be a unique transformation yielding a Euclidean reconstruction.

To obtain a coherent projective reconstruction, some authors [62, 10] use the reconstruction obtained from the first two views to compute the positions of the other cameras in the arbitrary projective frame of the initial reconstruction (solving the exterior orientation problem, Section 3.2). The 3-D location of additional points may be computed as long as the camera matrices are known for two cameras in which these points are visible. Then, a global minimization of the reprojection error is performed, incrementally or batch-wise (this is the so-called *bundle adjustment* [66]).

A very elegant method is described in [133], based on the recovery of the projective depths. Taken individually, the projective depths are arbitrary (because they depend on arbitrary scale factors), but in a sequence of images they are linked together, and this is the missing constraint that gives a coherent projective reconstruction. Let $\mathbf{F}' = \mathbf{F}^\top$ the fundamental matrix of the second camera; from (140) the following relationship can be obtained

$$\kappa' \mathbf{F}' \tilde{\mathbf{m}}' = \kappa (\mathbf{e} \wedge \tilde{\mathbf{m}}) \tag{161}$$

This equation relates the projective depths of a single 3-D point $\mathbf{w}$ in two images. From the latter one can obtain

$$\kappa = \frac{(\mathbf{e} \wedge \tilde{\mathbf{m}}) \mathbf{F}' \tilde{\mathbf{m}}'}{||\mathbf{e} \wedge \tilde{\mathbf{m}}||^2} \kappa'. \tag{162}$$

By estimating a sufficient number of fundamental matrices and epipoles, we recursively chain together equation like (162) to give estimates for the complete set of depths for point $\mathbf{w}$, starting from $\kappa_1 = 1$. A similar method has been presented in [151].

Another approach [4, 151] to the problem moves from the following remark. The matrix $\mathbf{M}$ in (160) can be interpreted as a plane homography, hence we can say that the reconstruction is *referred* to that plane. It is this reference plane that should not change from one reconstruction to another.

7.5 Euclidean reconstruction

We have seen that a projective reconstruction can be computed starting from points correspondences only, without any knowledge of the camera matrices. Despite it conveys some useful informations [122], we would like to obtain an *Euclidean reconstruction*, a very special one that differs from the true reconstruction by a similarity transformation. This is composed by a rigid displacement (due to the arbitrary choice of the world reference frame) plus a a uniform change of scale (due to the well-known depth-speed ambiguity, Chapter 5).

Maybank and Faugeras [100] proved that, if intrinsic parameters are constant, Euclidean reconstruction is achievable. The procedure is known as *autocalibration*.

In this approach the internal unchanging parameters of the camera are computed from at least three views. Once the intrinsic parameters are known, the problem of computing the extrinsic parameters (motion) from point correspondences is the well-known relative orientation problem (Chapter 5).

Recently, new approaches based on the idea of *stratification* [94, 34] have been introduced. Starting from a projective reconstruction, which can be computed from the set of correspondences $\{\tilde{\mathbf{m}}_j^i\}$ only, the problem is computing the *proper* $\tilde{\mathbf{T}}$ that upgrades it to an Euclidean reconstruction, by exploiting all the available constraints. To this purpose the problem is stratified into different representations: depending on the amount of information and the constraints available, it can be analyzed at a projective, affine[2], or Euclidean level.

7.6 Autocalibration

In the case of two different cameras, the fact that for any fundamental matrix $\mathbf{F}$ one can find intrinsic parameters matrix $\mathbf{A}$ and $\mathbf{A}'$ such that $\mathbf{E} = \mathbf{A}'^{\top}\mathbf{F}\mathbf{A}$ is called the *rigidity constraint*.

The seven parameters of the fundamental matrix are available to describe the geometric relationship between the two views; the five parameters of the essential matrix are needed to describe the rigid displacement, thus at most two independent constraint are available for the computation of the intrinsic parameters from the fundamental matrix. Indeed, Hartley [61] proposed an algorithm to factor the fundamental matrix that yields the five motion parameters and the two different focal lengths. He also noticed that no more information could be extracted from the fundamental matrix without making additional assumptions.

In the case of a moving camera with constant intrinsic parameters, it is possible to obtain an Euclidean reconstruction by cumulating constraints over different displacements. There are five unknown (the intrinsic parameters), each displacement

[2] An affine reconstruction differs from the true one by an affine transformation.

yields two independent constraints, hence three views are sufficient (between three views there are three independent displacements: 1-2, 1-3 and 2-3).

7.6.1 Kruppa equations

With a minimum of three displacements, we can obtain the internal parameters of the camera using a system of polynomial equations due to Kruppa [82], which are derived from a geometric interpretation of the rigidity constraint [36, 100].
The unknown in the Kruppa equations is the matrix $\mathbf{K} = \mathbf{A}\mathbf{A}^\top$, called the *Kruppa coefficients matrix*, that represents the dual of the image of the *absolute conic* (see [33] for details). From $\mathbf{K}$ one can easily obtain the intrinsic parameters by means of Cholesky factorization ($\mathbf{K}$ is symmetric and definite positive), or in closed form:

$$\text{if}\quad \mathbf{K} = \begin{bmatrix} k_1 & k_2 & k_3 \\ k_2 & k_4 & k_5 \\ k_3 & k_5 & 1 \end{bmatrix} \quad\text{then}\quad \mathbf{A} = \begin{bmatrix} \sqrt{k_1 - {k_3}^2 - \frac{(k_2 - k_3 k_5)^2}{k_4 - {k_5}^2}} & \frac{k_2 - k_3 k_5}{\sqrt{k_4 - {k_5}^2}} & k_3 \\ 0 & \sqrt{k_4 - {k_5}^2} & k_5 \\ 0 & 0 & 1 \end{bmatrix}. \tag{163}$$

Kruppa equations were rediscovered and derived by Maybank and Faugeras [100]. Recently Hartley [64] provided a simpler form, based on the Singular Value Decomposition of the fundamental matrix. Let $\mathbf{F}$ be written as $\mathbf{F} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ (with SVD), and

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \\ \mathbf{u}_3^\top \end{bmatrix} \quad \mathbf{V} = \begin{bmatrix} \mathbf{v}_1^\top \\ \mathbf{v}_2^\top \\ \mathbf{v}_3^\top \end{bmatrix} \quad \mathbf{D} = \operatorname{diag}(r, s, 0).$$

Then the Kruppa equations write (the derivation can be found in [64])

$$\frac{\mathbf{v}_2^\top \mathbf{K} \mathbf{v}_2}{r^2 \mathbf{u}_1^\top \mathbf{K} \mathbf{u}_1} = \frac{-\mathbf{v}_2^\top \mathbf{K} \mathbf{v}_1}{rs \mathbf{u}_1^\top \mathbf{K} \mathbf{u}_2} = \frac{\mathbf{v}_1^\top \mathbf{K} \mathbf{v}_1}{s^2 \mathbf{u}_2^\top \mathbf{K} \mathbf{u}_2}. \tag{164}$$

From (164) one obtains two independent quadratic equations in the five parameters of $\mathbf{K}$ for each fundamental matrix (i.e., for each displacement). Moreover, assuming that $\gamma = 0$, which is a good approximation for usual cameras, one has the additional constraint $k_3 k_5 = k_2$ [92]. There are basically two classes of methods for solving the resulting system of equations (assuming that more than three views are available) [164, 92]:

- Partition the equations set in groups of five and solve each group with a global convergent technique for systems of polynomial equations, like homotopy continuation methods [106, 131]. Each system will give a set of solutions and the solution common to all of them is chosen. This method – presented in [92] – has the great advantage of global convergence, but is computationally expensive. Moreover, the number of systems to be solved rapidly increases with the number of displacements.
- The over-constrained system of equation is solved with a non-linear least-squares technique (Levenberg-Marquardt [48], or Iterated Extended Kalman Filter [101]). The problem with non-linear least-squares is that a starting point close to the solution is needed. This can be obtained by applying globally convergent methods to subsets of equations (like in the previous case), or by making the additional assumption that (u_0, v_0) is in the center of the image, thereby obtaining (from just one fundamental matrix) two quadratic equations in two variables k_1, k_4, which can be solved analytically [64]. This technique is used in [164].

7.7 Stratification

Let us assume that a projective reconstruction is available, that is a sequence $\{\tilde{\mathbf{P}}^i_{\text{proj}}\}$ of camera matrices such that:

$$\tilde{\mathbf{P}}^0_{\text{proj}} = [\mathbf{I} \mid \mathbf{0}]; \qquad \tilde{\mathbf{P}}^i_{\text{proj}} = [\mathbf{Q}^i \mid \mathbf{q}^i]. \tag{165}$$

We are looking for an Euclidean reconstruction, that is a 4×4 nonsingular matrix $\tilde{\mathbf{T}}$ that upgrades the projective reconstruction to Euclidean. If $\{\tilde{\mathbf{w}}_j\}$ is the sought Euclidean structure, $\tilde{\mathbf{T}}$ must be such that: $\mathbf{m}^i_j = \tilde{\mathbf{P}}^i_{\text{proj}}\tilde{\mathbf{T}}\tilde{\mathbf{T}}^{-1}\mathbf{w}_j$, hence

$$\tilde{\mathbf{P}}^i_{\text{eucl}} \simeq \tilde{\mathbf{P}}^i_{\text{proj}}\tilde{\mathbf{T}} \ , \tag{166}$$

where the symbol $\simeq$ means "equal up to a scale factor."

7.7.1 Using additional information

Projective reconstruction differs from Euclidean by an unknown projective transformation in the 3-D projective space, which can be seen as a suitable change of basis. Thanks to the fundamental theorem of projective geometry (see Appendix A), a collineation in space is determined by five points, hence the knowledge of the true (Euclidean) position of five points allows to compute the unknown 4×4 matrix $\tilde{\mathbf{T}}$ that transform the Euclidean frame into the projective frame. An application of this is reported in [113].

Moreover, if intrinsic parameters $\mathbf{A}$ are known, then $\tilde{\mathbf{T}}$ can be computed by solving a linear system of equations derived from (194).

7.7.2 Euclidean reconstruction from constant intrinsic parameters

The challenging problem is to recover $\tilde{\mathbf{T}}$ without additional information, using only the *hypothesis of constant intrinsic parameters.* The works by Hartley [58], Pollefeys and Van Gool [118], Heyden and Åström[68], Triggs [145] and Bougnoux [17] will be reviewed, but first we will make some remarks that are common to most of the methods.

We can choose the first Euclidean-calibrated camera to be $\tilde{\mathbf{P}}^0_{\text{eucl}} = \mathbf{A}[\mathbf{I} \mid \mathbf{0}]$, thereby fixing arbitrarily the rigid transformation:

$$\tilde{\mathbf{P}}^0_{\text{eucl}} = \mathbf{A}[\mathbf{I} \mid \mathbf{0}] \qquad \tilde{\mathbf{P}}^i_{\text{eucl}} = \mathbf{A}[\mathbf{R}^i \mid \mathbf{t}^i]. \tag{167}$$

With this choice, it is easy to see that $\tilde{\mathbf{P}}^0_{\text{eucl}} = \tilde{\mathbf{P}}^0_{\text{proj}}\tilde{\mathbf{T}}$ implies

$$\tilde{\mathbf{T}} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{r}^\top & s \end{bmatrix} \tag{168}$$

where $\mathbf{r}^\top = [r_1 \; r_2 \; r_3]$. Under this parameterization $\tilde{\mathbf{T}}$ is clearly non singular, and being defined up to a scale factor, it depends on eight parameters ($s = 1$).

Substituting (165) in (166) one obtains

$$\tilde{\mathbf{P}}^i_{\text{eucl}} \simeq \tilde{\mathbf{P}}^i_{\text{proj}}\tilde{\mathbf{T}} = [\mathbf{Q}^i\mathbf{A} + \mathbf{q}^i\mathbf{r}^\top \mid \mathbf{q}^i], \tag{169}$$

and from (167)

$$\tilde{\mathbf{Q}}^i_{\text{eucl}} = \mathbf{A}[\mathbf{R}^i \mid \mathbf{t}^i] = [\mathbf{A}\mathbf{R}^i \mid \mathbf{A}\mathbf{t}^i], \tag{170}$$

hence

$$\mathbf{Q}^i\mathbf{A} + \mathbf{q}^i\mathbf{r}^\top \simeq \mathbf{A}\mathbf{R}^i. \tag{171}$$

This is the basic equation, relating the unknowns $\mathbf{A}$ (five parameters) and $\mathbf{r}$ (three parameters) to the available data $\mathbf{Q}^i$ and $\mathbf{q}^i$. $\mathbf{R}$ is unknown, but must be a rotation matrix.

Affine reconstruction. Equation (171) can be rewritten as

$$\mathbf{Q}^i + \mathbf{q}^i\mathbf{r}^\top\mathbf{A}^{-1} \simeq \mathbf{A}\mathbf{R}^i\mathbf{A}^{-1} = \mathbf{H}^i_\infty, \tag{172}$$

relating the unknown vector $\mathbf{a}^\top = \mathbf{r}^\top\mathbf{A}^{-1}$ to the homography of the infinity plane (compare (172) with (152)). It can be seen that $\mathbf{T}$ factorizes as follows

$$\tilde{\mathbf{T}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{a}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix}. \tag{173}$$

The right-hand matrix is an *affine transformation*, not moving the infinity plane, whereas the left-hand one is a transformation moving the infinity plane.
Substituting the latter into (166) we obtain:

$$\tilde{\mathbf{P}}^i_{\text{eucl}} \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} = \tilde{\mathbf{P}}^i_{\text{affi}} \simeq \tilde{\mathbf{P}}^i_{\text{proj}} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{a}^\top & 1 \end{bmatrix} = [\mathbf{H}^i_\infty | \mathbf{q}^i] \tag{174}$$

Therefore, the knowledge of the homography of the infinity plane (given by $\mathbf{a}$) allows to compute the Euclidean structure up to an affine transformation, that is an *affine reconstruction*.

From affine to Euclidean. Another useful observation is, if $\mathbf{H}_\infty$ is known and the intrinsic parameters are constant, the intrinsic parameters matrix $\mathbf{A}$ can easily be computed. In other words, updating from affine to Euclidean reconstruction is straightforward.
Let us consider the case of two cameras. If $\mathbf{A}' = \mathbf{A}$, then $\mathbf{H}_\infty$ is exactly known (with the right scale), since

$$\det(\mathbf{H}_\infty) = \det(\mathbf{A}\mathbf{R}\mathbf{A}^{-1}) = 1. \tag{175}$$

From (153) we obtain $\mathbf{R} = \mathbf{A}'^{-1}\mathbf{H}_\infty\mathbf{A}$, and, since $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$, it is easy to obtain:

$$\mathbf{H}_\infty\mathbf{K}\mathbf{H}_\infty^\top = \mathbf{K} \tag{176}$$

where $\mathbf{K} = \mathbf{A}\mathbf{A}^\top$ is the Kruppa coefficients matrix. As (176) is an equality between 3×3 symmetric matrices, we obtain a linear system of six equations in the five unknown k_1, k_2, k_3, k_4, k_5 . In fact, only four equations are independent [94, 155], hence at least three views (with constant intrinsic parameters) are required to obtain an over-constrained linear system, which can be easily solved with a linear least-squares technique.

Note that two views would be sufficient under the usual assumption that the image reference frame is orthogonal ($\gamma = 0$), which gives the additional constraint $k_3 k_5 = k_2$ [94, 155].

If points at infinity (in practice, sufficiently far from the camera) are in the scene, $\mathbf{H}_\infty$ can be computed from point correspondences, like any ordinary plane homography [155] Moreover, with additional knowledge, it can be estimated from vanishing points or parallelism [37, 34].

In the rest of the section, some of the most promising stratification techniques will be reviewed.

Hartley

Hartley [58] pioneered this kind of approach. Starting from (171), we can write

$$(\mathbf{Q}^i + \mathbf{q}^i \mathbf{a}^\top)\mathbf{A} \simeq \mathbf{A}\mathbf{R}^i. \tag{177}$$

By taking the QR decomposition of the left-hand side we obtain an upper triangular matrix $\mathbf{B}^i$ such that $(\mathbf{Q}^i + \mathbf{q}^i \mathbf{a}^\top)\mathbf{A} = \mathbf{B}^i \mathbf{R}^i$, so (177) rewrites

$$\mathbf{B}^i \mathbf{R}^i = \lambda^i \mathbf{A} \mathbf{R}^i \qquad \text{or} \qquad \frac{1}{\lambda^i} \mathbf{A}^{-1} \mathbf{B}^i = \mathbf{I}. \tag{178}$$

The scale factor $1/\lambda^i$ can be chosen so that the sum of the squares of the diagonal entries of $(1/\lambda^i)\mathbf{A}^{-1}\mathbf{B}^i$ equals three. Each camera excluding the first, gives six constraints in eight unknowns, so three cameras are sufficient. In practice there are more than three cameras, and the non-linear least squares problem can be solved with Levenberg-Marquardt minimization algorithm [48]. As noticed in the case of Kruppa equations, a good initial guess for the unknowns $\mathbf{A}$ and $\mathbf{a}$ is needed in order for the algorithm to converge to the solution.

Given that from $\mathbf{H}^i_\infty$ the computation of $\mathbf{A}$ is straightforward, a guess for $\mathbf{a}$ (that determines $\mathbf{H}^i_\infty$) is sufficient. The *cheirality constraint* [62] is exploited by Hartley to estimate the infinity plane homography, thereby obtaining an approximate affine (or *quasi-affine*) reconstruction.

Pollefeys and Van Gool

In this approach [118], a projective reconstruction is first updated to affine reconstruction by the use of the *modulus constraint* [94, 119]: since the left-hand part of (172) is conjugated to a (scaled) rotation matrix, all eigenvalues must have equal moduli. Note that this holds if and only if intrinsic parameters are constant. To make the constraint explicit we write the characteristic polynomial:

$$\det(\mathbf{Q}^i + \mathbf{q}^i\mathbf{a}^\top - \lambda\mathbf{I}) = l_3\lambda^3 + l_2\lambda^2 + l_1\lambda + l_0. \tag{179}$$

The equality of the roots of the characteristic polynomial is not easy to impose, but a simple necessary condition holds:

$$l_3 l_1^3 = l_2^3 l_0. \tag{180}$$

This yields a fourth order polynomial equation in the unknown $\mathbf{a}$ for each camera except the first, so a finite number of solutions can be found for four cameras. Some solutions will be discarded using the modulus constraint, that is more stringent than (180).

As discussed previously, autocalibration is achievable with only three views. It is sufficient to note that, given three cameras, for every plane homography, the following holds [94]:

$$\mathbf{H}^{1,3} = \mathbf{H}^{2,3}\mathbf{H}^{1,2}. \tag{181}$$

In particular it holds for the infinity plane homography, so

$$\mathbf{H}^{i,j}_\infty = \mathbf{H}^j_\infty {\mathbf{H}^i_\infty}^{-1} \simeq (\mathbf{Q}^j + \mathbf{q}^j\mathbf{a}^\top)(\mathbf{Q}^i + \mathbf{q}^i\mathbf{a}^\top)^{-1}. \tag{182}$$

In this way we obtain a constraint on the plane at infinity for each pair of views. Let us write the characteristic polynomial:

$$\det((\mathbf{Q}^j + \mathbf{q}^j\mathbf{a}^\top)(\mathbf{Q}^i + \mathbf{q}^i\mathbf{a}^\top)^{-1} - \lambda\mathbf{I}) = 0 \quad \Longleftrightarrow \tag{183}$$

$$\det((\mathbf{Q}^j + \mathbf{q}^j\mathbf{a}^\top) - \lambda(\mathbf{Q}^i + \mathbf{q}^i\mathbf{a}^\top)) = 0 \tag{184}$$

Writing the constraint (180) for the three views, a system of three polynomial of degree four in three unknowns is obtained. Here, like in the solution of Kruppa equations, homotopy continuation methods could be applied to compute all the $4^3 = 64$ solutions.

In practice more than three views are available, and we must solve a non-linear least-squares problem: Levenberg-Marquardt minimization is used by the author. The initial guess leading to convergence is obtained by starting form a *quasi-Euclidean* [10] reconstruction, i.e., a reconstruction such that (171) is approximately satisfied. This can be achieved by approximate knowledge of camera parameters and motion or by using Hartley's method for computing a quasi-affine reconstruction.

Heyden and Åström

The method proposed by Heyden and Åström [68] is again based on (171), which can be rewritten as

$$\tilde{\mathbf{P}}^{i}_{\text{proj}} \begin{bmatrix} \mathbf{A} \\ \mathbf{r}^\top \end{bmatrix} \simeq \mathbf{A}\mathbf{R}^{i}. \tag{185}$$

Since $\mathbf{R}^i\mathbf{R}^{i\top} = \mathbf{I}$ it follows that:

$$\tilde{\mathbf{P}}^{i}_{\text{proj}} \begin{bmatrix} \mathbf{A} \\ \mathbf{r}^\top \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \mathbf{r}^\top \end{bmatrix}^\top \tilde{\mathbf{P}}^{i\top}_{\text{proj}} = \tilde{\mathbf{P}}^{i}_{\text{proj}} \begin{bmatrix} \mathbf{A}\mathbf{A}^\top & \mathbf{A}\mathbf{r} \\ \mathbf{r}^\top\mathbf{A}^\top & \mathbf{r}^\top\mathbf{r} \end{bmatrix} \tilde{\mathbf{P}}^{i\top}_{\text{proj}} \simeq \mathbf{A}\mathbf{R}^i\mathbf{R}^{i\top}\mathbf{A}^\top = \mathbf{A}\mathbf{A}^\top. \tag{186}$$

Note that (186) contains five equations, because the matrices of both members are symmetric, and the homogeneity reduces the number of equations with 1. Hence, each camera matrix, apart from the first one, gives five equations the eight unknowns $\alpha_u, \alpha_v, \gamma, u_0, v_0, r_1, r_2, r_3$. A unique solution is obtained when three cameras are available. If the unknown scale factor is introduced explicitly, (186) rewrites:

$$0 = f_i(\mathbf{A}, \mathbf{r}, \lambda_i) = \lambda_i^2 \mathbf{A}\mathbf{A}^\top - \tilde{\mathbf{P}}^{i}_{\text{proj}} \begin{bmatrix} \mathbf{A}\mathbf{A}^\top & \mathbf{A}\mathbf{r} \\ \mathbf{r}^\top\mathbf{A}^\top & \mathbf{r}^\top\mathbf{r} \end{bmatrix} \tilde{\mathbf{P}}^{i\top}_{\text{proj}}. \tag{187}$$

Therefore, 3 cameras yield 18 equations in 11 unknowns.

Triggs

Triggs [145] proposed a method based on the *absolute quadric* and, independently from Heyden and Åström, he derived an equation closely related to (186). The absolute quadric $\mathbf{\Omega}$ consists of planes tangent to the absolute conic [33], and in an Euclidean frame, is represented by the matrix

$$\mathbf{\Omega}_{\text{euc}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}. \tag{188}$$

If $\tilde{\mathbf{T}}$ is a projective transformation acting as in (166), then it can be verified [145] that it transforms $\mathbf{\Omega}_{\text{euc}}$ into $\mathbf{\Omega} = \tilde{\mathbf{T}}\mathbf{\Omega}_{\text{euc}}\tilde{\mathbf{T}}^{\top}$. Since the projection of the absolute quadric yields the dual image of the absolute conic [145], one obtain

$$\tilde{\mathbf{P}}^{i}_{\text{proj}}\mathbf{\Omega}{\tilde{\mathbf{P}}^{i}}^{\top}_{\text{proj}} \simeq \mathbf{K} \tag{189}$$

from which, assuming (168), (186) follows immediately. Triggs, however, does not assume any particular form for $\tilde{\mathbf{T}}$, hence the unknown are $\mathbf{K}$ and $\mathbf{\Omega}$. Note that both these matrix are symmetric and defined up to a scale factor.

Let $\mathbf{k}$ be the matrix composed by the the six elements of the lower triangle of $\mathbf{K}$, and $\boldsymbol{\omega}$ be the matrix composed by the six elements of the lower triangle of $\mathbf{\Omega}$, then (186) is equivalent to

$$\boldsymbol{\omega} \wedge \mathbf{k} = \mathbf{0} \tag{190}$$

in which the unknown scale factor is eliminated. For each camera this amounts to 15 bilinear equations in $9 + 5$ unknowns, since both $\mathbf{k}$ and $\boldsymbol{\omega}$ are defined up to a scale factor. Since only five of them are linearly independent, at least three images are required for a unique solution.

Triggs uses two methods for solving the non-linear least-squares problem: sequential quadratic programming [48] on $N \geq 3$ cameras, and a quasi-linear method with SVD factorization on $N \geq 4$ cameras. He recommend to use data standardization (see Section 5.4.1) and to enforce $\det(\mathbf{\Omega}) = 3$. The sought transformation $\tilde{\mathbf{T}}$ is computed by taking the eigen-decomposition of $\mathbf{\Omega}$.

Bougnoux

This methods [17] is different from the previous ones, because it does not require constant intrinsic parameters and because it achieves an approximate Euclidean reconstruction without obtaining meaningful camera parameters as a by-product. Let us write (166) in the following form:

$$\tilde{\mathbf{P}}^i_{\text{eucl}} = \left[\begin{array}{c|c} \mathbf{q}_1^{i\top} & \\ \mathbf{q}_2^{i\top} & \mathbf{q}^i \\ \mathbf{q}_3^{i\top} & \end{array}\right] \simeq \tilde{\mathbf{P}}^i_{\text{proj}}\tilde{\mathbf{T}} \tag{191}$$

where $\mathbf{q}_1^{i\top}, \mathbf{q}_2^{i\top}, \mathbf{q}_3^{i\top}$ are the rows of $\tilde{\mathbf{P}}^i_{\text{eucl}}$. The customary assumptions $\gamma = 0$ and $\alpha_u = \alpha_v$, are used to constraint the Euclidean camera matrices:

$$\gamma = 0 \iff (\mathbf{q}_1^i \wedge \mathbf{q}_3^i)^\top(\mathbf{q}_2^i \wedge \mathbf{q}_3^i) = 0 \tag{192}$$

$$\alpha_u = \alpha_v \iff ||\mathbf{q}_1^i \wedge \mathbf{q}_3^i|| = ||\mathbf{q}_2^i \wedge \mathbf{q}_3^i||. \tag{193}$$

Thus each camera, excluding the first, gives two constraints of degree four. Since we have six unknown, at least four cameras are required to compute $\tilde{\mathbf{T}}$. If the principal point (u_0, v_0) is forced to the image center, the unknowns reduce to four and only three cameras are needed.

The non-linear minimization required to solve the resulting system is rather unstable and needs to be started in a close initialization: we need to estimate the focal length and $\mathbf{r}$. Assuming known principal point, no skew, and unit aspect ratio, the focal length can be computed from the Kruppa equations in closed form [17]. Then, assuming known intrinsic parameters $\mathbf{A}$, an estimation of $\mathbf{r}$ can be computed by solving a *linear* least-squares problem. From (186) the following is obtained:

$$\mathbf{Q}^i\mathbf{A}\mathbf{A}^\top\mathbf{Q}^{i^\top} + \mathbf{Q}^i\mathbf{A}\mathbf{r}\mathbf{q}^{i^\top} + (\mathbf{Q}^i\mathbf{A}\mathbf{r}\mathbf{q}^{i^\top})^\top + ||\mathbf{r}||^2\mathbf{q}^i\mathbf{q}^{i^\top} = \lambda^2\mathbf{A}\mathbf{A}^\top. \tag{194}$$

Since $[\mathbf{A}\mathbf{A}^\top]_{3,3} = \mathbf{K}_{3,3} = 1$, then λ is fixed. After some algebraic manipulation [17] one ends up with four linear equations in $\mathbf{A}\mathbf{r}$. This method works also with varying intrinsic parameters, although, in practice, only the focal length is allowed to vary, since principal point is forced to the image center and no skew and unit aspect ratio are assumed. The estimation of the camera parameters is inaccurate, nevertheless Bougnoux proves that the reconstruction is correct up to an anisotropic homotethy, which he claims to be enough for the reconstructed model to be usable.

7.8 Discussion

The applicability of autocalibration techniques in the real world depends on two issues: sensitivity to noise and solutions bracketing. The challenge is to devise a method that exhibits graceful degradation as noise increases and needs only an approximate initialization.

As for the Kruppa equations, in [92] the authors compare three solving methods: the homotopy continuation method, Levenberg-Marquardt and the Iterated Extended Kalman Filter. From the simulations reported, it appears that all the methods give comparable results. However, the homotopy continuation method is suitable for the case of few displacements, as it would be difficult to use all the constraints provided by a long sequence, and its computational cost would be too high. Iterative approaches (Levenberg-Marquardt and Iterated Extended Kalman Filter) are well suited to the case where more displacements are available. The main limitation of all these methods is the sensitivity to the noise in the localization of points.

The autocalibration methods based on stratification that we described have appeared only recently, and only preliminary and partial results are available. Trigg's non-linear algorithm is reported to be accurate, fast and stable and requires only approximate initialization. Both Hartley's and Pollefey's algorithms require a quasi-affine reconstruction to start with; the number of unknown in the latter is only three, whereas in the former is eight. Unfortunately, in Pollefey's work the Euclidean reconstruction is evaluated only visually. Also in Heyden and Åström the reconstruction is assessed only visually, and initialization is taken very close to the ground-truth.

Bougnoux's algorithm is quite different form the others, since it does not even try to obtain an accurate Euclidean reconstruction. Assessment of reconstruction quality is deliberately visual.

7.9 Conclusions

This chapter presented a review of recent techniques for Euclidean reconstruction from a single moving camera, with unconstrained motion and unknown *constant parameters*. Such unified, comparative discussion has not yet been presented in the literature.

Even though formulations may be different, to all the methods reviewed, much of the underlying mathematics is common. However, since problems are inherently non-linear, proper formulation is very important to avoid difficulties created by the numerical computation of the solutions.

Despite this problem is far from being completely solved, the more general one in which intrinsic parameters are varying is gaining the attention of researchers. In fact, Bougnoux's method already copes with varying parameters. Heyden and Åström [69] proposed a method that works with varying and unknown focal length and principal point. Later, they proved [70] that it is sufficient to know any of the five intrinsic parameters to make Euclidean reconstruction, even if all other parameters are unknown and varying. A similar method that can work with different types of of constraints has been recently presented in [117].

Chapter 8

3-D Motion

This chapter address the *3-D motion problem*, where the points correspondences and the rigid displacement between two sets of 3-D points are to be recovered. One application is to register sets of 3-D measurements obtained with different reconstruction algorithm or depth measuring devices. The existence of missing points in the two sets makes the problem difficult. We present RICP, a robust algorithm for registering and finding correspondences in sets of 3-D points with significant percentages of missing data. RICP exploits LMedS robust estimation to withstand the effect of outliers. Our extensive experimental comparison of RICP with an existing method (ICP) shows RICP's superior robustness and reliability.

8.1 Introduction

This chapter presents a solution to recovering the rigid transformation (rotation and translation) that brings two 3-D point sets into alignment, when the correspondences between points are not known and there exist missing data. Given a set of 3-D points on a rigid body in one Cartesian system, and another set of points from the same body in a rotated and translated coordinate system, and given the correspondences between 3-D points, to estimate the rotation and translation is called the *3-D motion problem* (also known as *absolute orientation problem*). To recover the correspondences of the points in the two sets is called the *correspondence problem.* The two problems are intimately connected; [156] gives a nice illustration of their mathematical symmetry. Least-squares (LS) solutions are well-known for the *ideal* *motion problem*, in which both sets contain the same number of points affected by moderate sensor noise [80], but fail for the *general motion problem*, whereby several points, called *outliers*, have no correspondence in the other set and may lie far from matched points.

3-D motion estimation is an important problem in many aspects of Computer Vision. First, it can be used to register several range views [24, 31, 132], acquired by active ranging systems like laser scanners [146], to recover an accurate, complete surface model of a 3-D object (*reverse engineering*). Second, 3-D based motion is useful in those cases where 3-D data can be reconstructed from 2-D images [49, 75, 83, 156], as we described in this thesis. An intriguing scenario is structure reconstruction from unregistered video sequences acquired by an uncalibrated camera. Consider several, uncalibrated video sequences of the same scene. Usually each sequence spans a continuous range of viewpoints, but the camera jumps discontinuously between sequences, and there is no information about such movements. Approximate, point-based Euclidean reconstructions can be computed from each sequence; such 3-D data could be registered to integrate independent sequences.

A popular method for registering 3-D data sets, without a-priori knowledge of correspondences, is the iterative closest point algorithm (ICP) introduced by Besl and McKay [13], and that has been applied in various vision systems using 3-D sensors. The ICP algorithm is an iterative procedure with each iteration consisting of two steps. In the first one, closest neighboring points are put into correspondences, while keeping the current object pose fixed. The second step updates the current registration by least-squares minimization of the displacement of matched point pairs. It can be shown that the iteration converges to a minimum of residual error. Since convergence is only local, the initial position is a critical parameter. [14, 19] report quantitative studies of ICP performance. The most relevant findings for our purposes are that (i) the initial registration guess affects only the speed of convergence (not registration accuracy), as long as it is chosen within the convergence basin of the target minimum; (ii) accurate registration is possible with no outliers, and requires very accurate measurements and high numbers of points; (iii) acceptable accuracy (for reverse engineering) can be achieved with 2-300 points.

Here we introduce RICP, an algorithm for registering robustly a limited number of sparse 3-D points (say about 100) corrupted by significant percentages of outliers.

We replaced the LS minimization of ICP with the robust Least Median of Squares (LMedS) regression [126] to withstand the effect of outliers.

As shown by our experiments, RICP achieves a larger basin of attraction and more accurate registrations than ICP. We noticed that RICP still works with dense data, but the advantages over ICP are smaller unless many outliers are present.

[99] also reports a robust registration method based on ICP and LMedS. Their method iterates a 3-step sequence of processes: random sampling, estimation of the motion parameters with ICP, and evaluation. The sequence *as a whole* makes up the LMedS algorithm. On the contrary, in our approach, LMedS (with random sampling) is used *inside* the ICP, where it replaces the LS rotation estimation. This enables us to use a dynamic translation estimate based on outlier-free data in the ICP iteration.

In the following, Section 8.2 summarizes ICP and its main features, Section 8.3 presents RICP, Section 8.4 reports our experimental evaluation of RICP, and Section 8.5 discusses RICP's contributions and limitations.

8.2 A brief summary of ICP

This section summarizes ICP and some features of our ICP implementation. Let $\mathcal{P} = \{\mathbf{p}_i\}_1^{N_p}$ and $\mathcal{M} = \{\mathbf{m}_i\}_1^{N_m}$ the two sets of 3-D points to align, which we call respectively *data* and *model*. In general, $N_p \neq N_m$. The problem is to compute the rotation $\mathbf{R}$ and translation $\mathbf{t}$ producing the best alignment of $\mathcal{P}$ and $\mathcal{M}$:

$$\mathcal{M} = \mathbf{R}\mathcal{P} + \mathbf{t}, \tag{195}$$

meaning that $\mathbf{R}$ and $\mathbf{t}$ are applied to each point in the set $\mathcal{P}$. In general, this equation will not be satisfied exactly by all points, hence the equality should be interpreted in the least square sense.

Let us define the *closest point* in the model to a data point $\mathbf{p}$ as

$$\mathrm{cp}(\mathbf{p}) = \arg\min_{\mathbf{m}\in\mathcal{M}} ||\mathbf{m} - \mathbf{p}||.$$

We can then summarize ICP as follows:

1. Compute the subset of CPs: $\mathcal{Y} = \{\mathbf{m} \in \mathcal{M} \mid \mathbf{p} \in \mathcal{P} : \mathbf{m} = \mathrm{cp}(\mathbf{p})\}$;

2. Compute a LS estimate of the motion bringing $\mathcal{P}$ onto $\mathcal{Y}$:

$$(\mathbf{R}, \mathbf{t}) = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{N_p} ||\mathbf{y}_i - \mathbf{R}\,\mathbf{p}_i - \mathbf{t}||^2. \tag{196}$$

   where $\mathbf{y}_i \in \mathcal{Y}$ and $\mathbf{p}_i \in \mathcal{P}$.

3. Apply the motion to the data points:

$$\mathcal{P} \leftarrow \mathbf{R}\mathcal{P} + \mathbf{t}.$$

4. If the stopping criterion (see below) is satisfied, exit; else go to **1**.

The algorithm stops as soon as one of the following conditions is satisfied:

- the mean square error (MSE) $d = 1/N_p \sum_{i=1}^{N_p} ||\mathbf{y}_i - \mathbf{p}_i||^2$ is sufficiently small;
- the MSE difference between two successive iterations is sufficiently small;
- the maximum allowed number of iterations has been reached.

It has been proven [13] that ICP converges monotonically to a local minimum of the MSE, an index commonly used along with its derivative with respect to the step index [13, 14, 132, 166].

For step **1**, we have implemented CP algorithms based on exhaustive search (acceptable with small point sets) and k-D trees [13, 166].

In step **2**, motion parameters are computed using a technique involving the SVD, which has been shown to yield the best global accuracy and stability [87]. Since (195) is satisfied by the centroids of the point sets as well, we can eliminate translation by defining the *centralized* sets:

$$\mathbf{p}_{c,i} = \mathbf{p}_i - \bar{\mathbf{p}} \quad \text{and} \quad \mathbf{y}_{c,i} = \mathbf{y}_i - \bar{\mathbf{y}}$$

where

$$\bar{\mathbf{p}} = 1/N_p \sum_{i=1}^{N_p} \mathbf{p}_i \quad \bar{\mathbf{y}} = 1/N_p \sum_{i=1}^{N_p} \text{cp}(\mathbf{p}_i).$$

Note that we estimate centroids $\bar{\mathbf{p}}$ (data) and $\bar{\mathbf{y}}$ (model) *at each iteration*, using only the N_p points that are CP for at least one data point, hence a model point increases its weight in the computation if it is the CP of several data points.

Problem (196) is then equivalent to the following problem:

$$\min_{\mathbf{R}} \sum_{i=1}^{N_p} \|\mathbf{y}_{c,i} - \mathbf{R}\mathbf{p}_{c,i}\|^2, \tag{197}$$

that is minimized when trace($\mathbf{RK}$) is maximized [80], where

$$\mathbf{K} = \sum_{i=1}^{N_p} \mathbf{y}_{c,i}\mathbf{p}_{c,i}^{\top}.$$

If the SVD of $\mathbf{K}$ is given by $\mathbf{K} = \mathbf{VDU}^{\top}$, then the optimal rotation matrix that maximizes the trace is $\mathbf{R} = \mathbf{VU}^{\top}$. The optimal translation is then computed as $\mathbf{t} = \bar{\mathbf{y}} - \mathbf{R}\bar{\mathbf{p}}$.

Extensive experimentation with our ICP implementation confirmed ICP's good performance with full overlap (all points in both views) and initial motion guesses very close to the solution, and its sensitivity to outliers (e.g., partial overlap) [14, 19]. Outliers skew the distribution of the residuals $r_i = \|\mathbf{y}_i - (\mathbf{R}\mathbf{p}_i + \mathbf{t})\|$ (Figure 52), and consequently LS motion estimates. In addition, outliers skew the centroid estimate, and consequently rotation estimates obtained after shifting data points to the centroid [80].

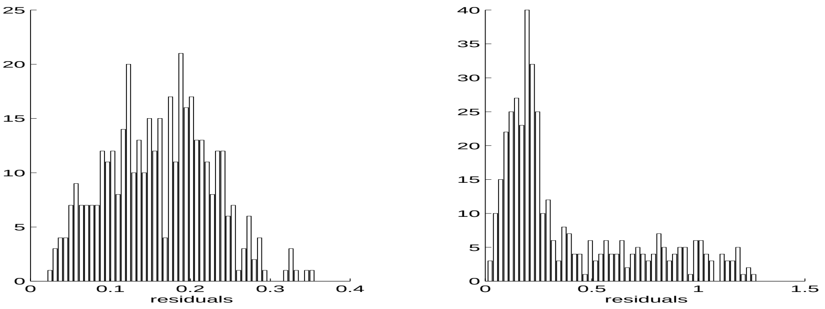

Figure 52: Residual distributions for synthetic point sets corrupted by Gaussian noise should be Gaussian, but are skewed by outliers. Two realizations of residuals are shown, with full (left) and partial (right) overlap, for one of the last iterations.

8.3 RICP: a Robust ICP algorithm

This section outlines RICP, our robust algorithm for correspondenceless point matching. Problem and notation are the same as in Section 8.2. RICP replaces step **2** of ICP with a robust estimation of motion, based on LMedS. The principle behind LMedS is the following: given a regression problem, where the number of parameters is d, compute a candidate model based on a randomly chosen d-tuple from the data; estimate the fit of this model to *all* the data, defined as the median of the residuals, and repeat optimizing the fit. The data points that do not belong to the optimal model, which represent the majority of the data, are *outliers*. The *breakdown point*, i.e., the smallest fraction of outliers that can yield arbitrary estimate values, is 50%. In principle all the d-tuples should be evaluated; in practice a Monte Carlo technique is applied, in which only a random sample of them of size m is considered. Assuming that the whole set of points may contain up to a fraction ϵ of outliers, the probability that at least one of the m d-tuple consist of d inliers is given by

$$P = 1 - (1 - (1 - \epsilon)^d)^m. \tag{198}$$

Hence, given d, ϵ, and the required P (close to 1), one can determine m:

$$m = \frac{\log(1 - P)}{\log(1 - (1 - \epsilon)^d)}. \tag{199}$$

In our implementation we assume $\epsilon = 0.5$, and require $P = 0.95$, thus $m = 1533$. When Gaussian noise is present in addition to outliers, the relative statistical efficiency (i.e., the ratio between the lowest achievable variance for the estimated parameters and the actual variance) of the LMedS is low; to increase the efficiency, it is advisable to run a weighted LS fit after LMedS, with weights depending on the residual of the LMedS procedure [126].

Estimating rotation. As in the previous case, we first eliminate translation by shifting data and model in the centroid (see next subsection), then, releasing temporarily the orthogonality constraint on $\mathbf{R}$, we cast the problem of computing the rotation $\mathbf{R}$ as a linear regression problem:

$$\left[\mathbf{y}_{c,1} \dots \mathbf{y}_{c,N_p}\right] = \mathbf{R}\left[\mathbf{p}_{c,1} \dots \mathbf{p}_{c,N_p}\right]$$

which can be re-written as follows:

$$\begin{bmatrix} \mathbf{X} & 0 & 0 \\ 0 & \mathbf{X} & 0 \\ 0 & 0 & \mathbf{X} \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} = \mathbf{b} \tag{200}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{p}_{c,1}^{\top} \\ \dots \\ \mathbf{p}_{c,N_p}^{\top} \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} \mathbf{r}_1^{\top} \\ \mathbf{r}_2^{\top} \\ \mathbf{r}_3^{\top} \end{bmatrix},$$

and $\mathbf{b}$ is obtained by juxtaposing the rows of the matrix $[\mathbf{y}_{c,1} \dots \mathbf{y}_{c,N_p}]$. The nine entries of $\mathbf{R}$ are then computed by solving the linear regression with the Monte Carlo LMedS method, outlined before.

The residuals s_j, $j = 1, \dots, 3N_p$ of (200) are used to generate the weights for the final, weighted LS regression as follows. First, a robust standard deviation estimate [126] is computed as

$$\hat{\sigma} = 1.4826 \left(1 + \frac{5}{2N_p - d + 1}\right) \sqrt{\underset{j}{\mathrm{med}}\, s_j^2}, \tag{201}$$

where d is the number of parameters (9 in our case). Second, a weight is assigned to each residual, such that

$$w_j = \begin{cases} 1 & \text{if } |s_j|/\hat{\sigma} \leq 2.5, \\ 0 & \text{otherwise.} \end{cases}$$

Notice that the w_j are associated to the individual coordinates of 3-D data point $\mathbf{p}_{c,i}$. A weight w_i^* is assigned to each point $\mathbf{p}_{c,i}$, which is zero if at least one of its coordinates has a zero weight, and one otherwise. We therefore deem a point $\mathbf{p}_{c,i}$ an outlier if at least one of its coordinates is an outlier. Finally, we estimate $\mathbf{R}$ by solving (197) with each point weighted by w_i^*. We use SVD to solve the weighted LS problem (similarly to Section 8.2), which yields a rotation matrix by construction.

Estimating centroids. As outliers skew centroid estimates, we adopt a weighted version of the dynamic average (Section 8.2) taking the average on the outlier-free data: $\bar{\mathbf{p}} = \sum_{i=1}^{N_p} w_i^* \mathbf{p}_i$ and $\bar{\mathbf{m}} = \sum_{i=1}^{N_p} w_i^* cp(\mathbf{p}_i)$.

8.4 Experimental results

Synthetic data. A first set of experiments was devoted to compare the accuracy and robustness of RICP and ICP with controlled noise and outliers. We generated model sets of 50 random points each within a unitary cube (performance depends

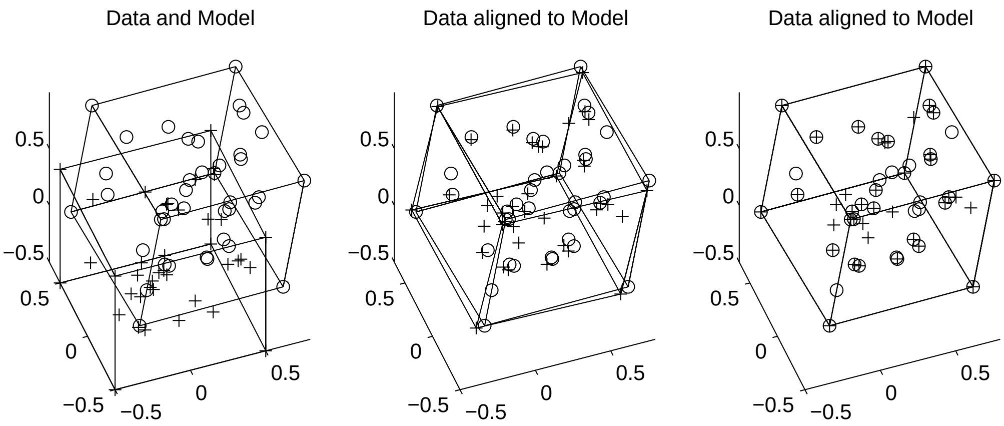

Figure 53: Cloud-of-points tests: example of registration with missing data (outliers). From left to right: starting position, ICP alignment, RICP alignment.

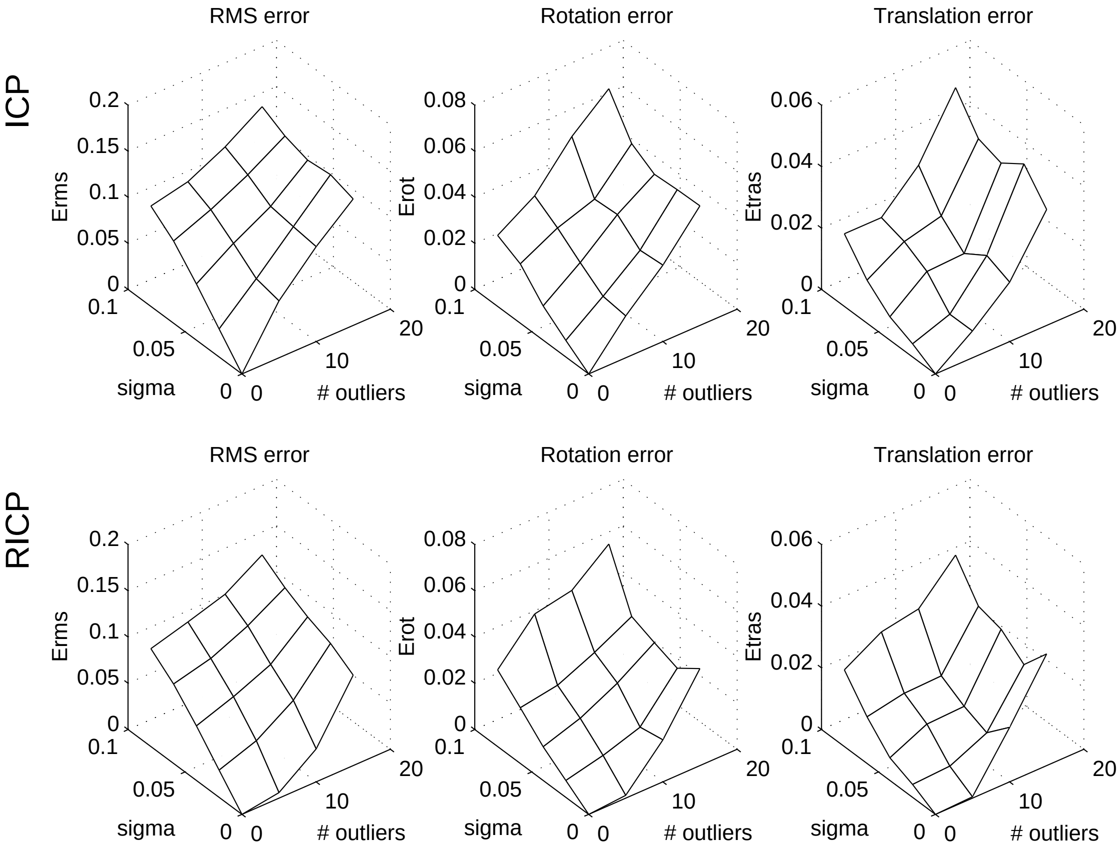

Figure 54: RMS error, rotation error and translation error vs standard deviation of Gaussian noise and number of outliers. Cloud-of-points tests. Top row: ICP results. Bottom row: RICP results.

on shape [19], but a reasonable indication of performance is achieved with non-elongated sets of random points). The data sets were obtained by translating and rotating the models ($\mathbf{t} = (0.2, 0.1, 0.4)^\top$, rotation by 0.17 rad around axis $(1, 1, 1)^\top$; notice the small rotation to guarantee ICP convergence to the correct alignment) and adding Gaussian noise of varying standard deviation. Following [156] outliers were simulated by dropping points at random from both sets, but avoiding to drop corresponding pairs from the two sets. For each noise and outlier level, we averaged and recorded the RMS errors, the absolute rotation and translation errors over 50 different realizations of noise and outliers.

Figure 53 shows a typical example of final alignment for ICP and RICP with outliers; the cubes attached to the data emphasize the different quality of the results. Figure 54 summarizes the results, suggesting the better accuracy of RICP. The figure plots the RMS, rotation and translation errors against the intensities of Gaussian noise and outliers (up to 20 points, that 40% of the data). The rotation and translation errors are the Frobenius norms of the difference between the true and estimated $\mathbf{R}$ and $\mathbf{t}$, respectively. These measures were chosen because (a) they are simple, scalar indices, (b) errors in the direction of the rotation axis (used previously) were artificially high with small rotations, which make axis estimates poorly conditioned, and (c) the RMS error (but not both Frobenius norms of $\mathbf{R}$ and $\mathbf{t}$) may be small for completely wrong alignments with certain shapes. Notice that, with no outliers, the RMS follows the standard deviation of the Gaussian noise, as one expects; in this case RICP benefits from final the weighted LS estimation, its performances being the same as ICP. With outliers, the increase of all error indices with the number of outliers is much sharper for ICP than for RICP. The performance degradation of both algorithms seems comparable with 40% outliers (recall that the initial displacement is small to ensure ICP convergence).

We verified the better accuracy of RICP also with different shapes. Figure 55 visualizes an example of final registration with outliers using as model points the corners of a standard calibration jig formed by regular grids of squares arranged on two perpendicular planes. Notice that, unlike the cloud of points above, which spans 3-D volumes, these data are surfaces. Figure 56 shows the results of the same type of tests leading to Figure 54.

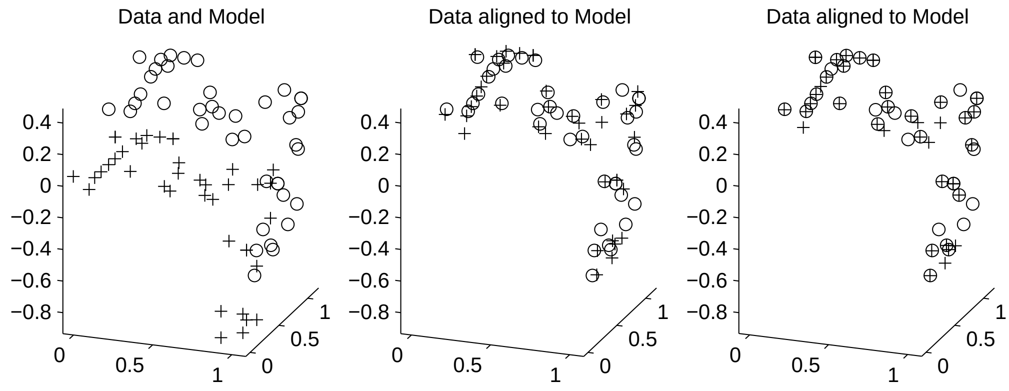

Figure 55: Calibration jig tests: example of registration with missing data (outliers). From left to right: starting position, ICP alignment, RICP alignment.

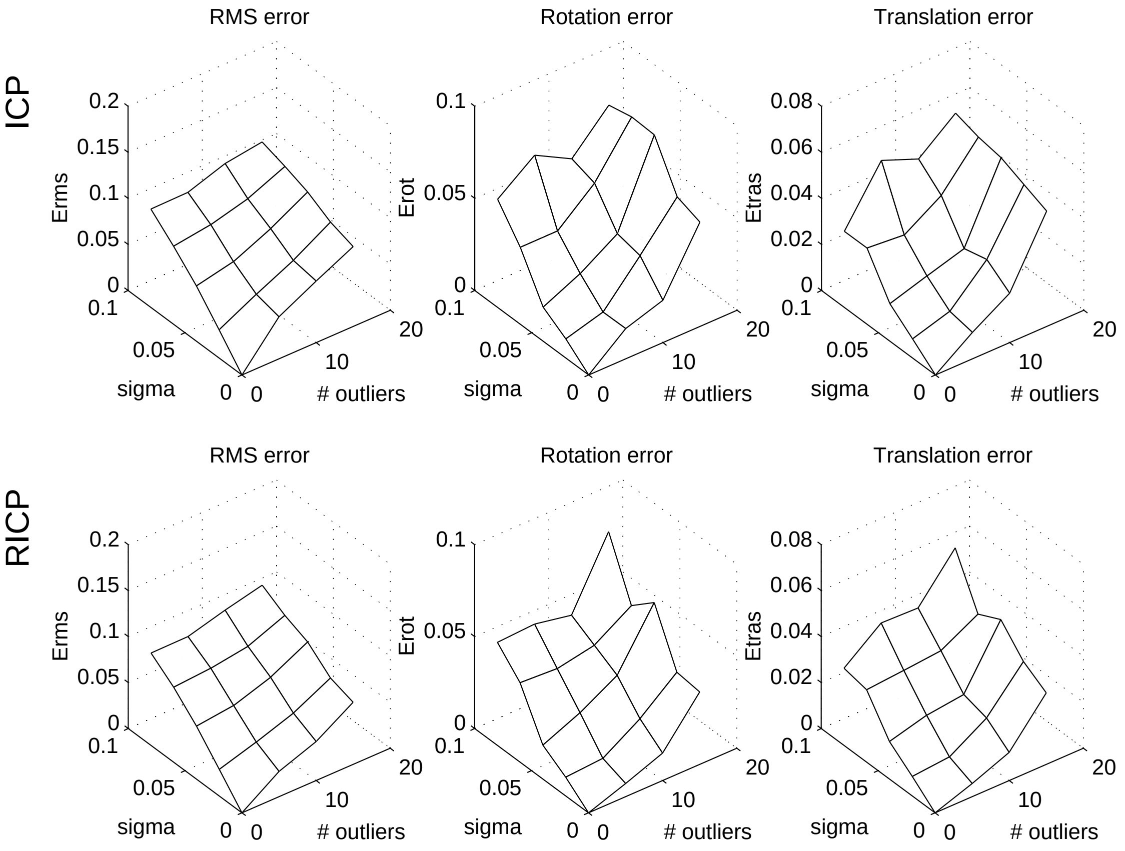

Figure 56: RMS error, rotation error and translation error vs. standard deviation of Gaussian noise and number of outliers. Calibration jig tests. Top row: ICP results. Bottom row: RICP results.

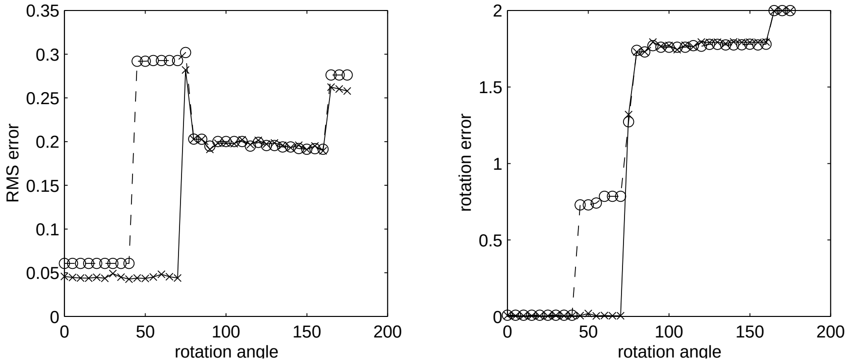

Figure 57: Basins of attraction. Final RMS (left) and rotation error (right) for ICP (dashed line and circles) and RICP (solid line and crosses) with increasing initial rotation angle.

In a second set of controlled experiments we verified the larger basin of convergence (the region in $\mathbf{R}, \mathbf{t}$ space guaranteeing convergence to the correct alignment) of RICP with respect to ICP, by observing the RMS and rotation errors (defined as above) for increasingly different initial rotations (from 0 to 180 degrees). We used sets of 30 points within the unitary cube, corrupted by outliers and Gaussian noise as before. Translation was fixed, as we found that rotation has the largest influence on the basin of convergence (because translation is eliminated by centroids subtraction). Figure 57 shows an example of results (with rotation axis $[1, 1, 1]^\top$, 20% outliers, 0.02 noise standard deviation), showing clearly that ICP stops converging before RICP (here, by about 35 degrees) as the initial rotation difference increases. Figure 58 visualizes a case in which ICP does not converge and RICP does, at a parity of initial displacement and noise/outliers conditions.

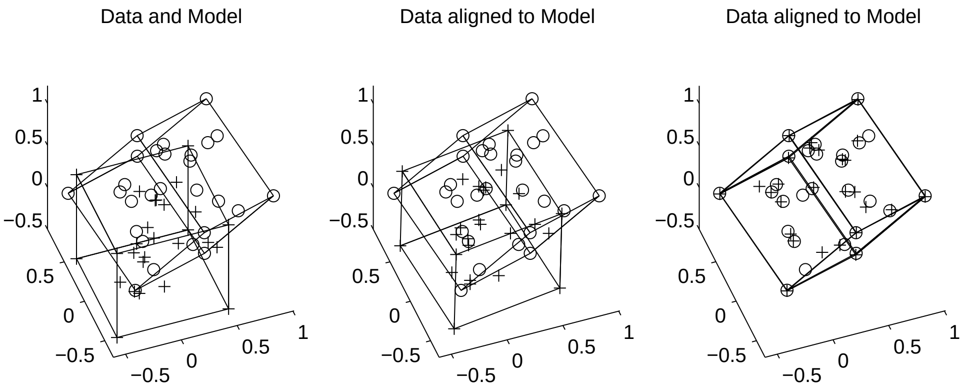

Figure 58: A case in which RICP finds the correct registration and ICP does not. From left to right: starting position, ICP alignment, RICP alignment.

A final set of experiments proved that RICP leads to more accurate registrations than ICP even with dense data with outliers (partial overlap between views). For instance, Figure 59 shows two range views of a mechanical widget, acquired by a laser scanner, and the registration found by RICP. Figure 60 shows the histograms of the absolute residuals for RICP and ICP, clearly smaller for RICP; the MSE is 7.21 for ICP and 5.01 for RICP.

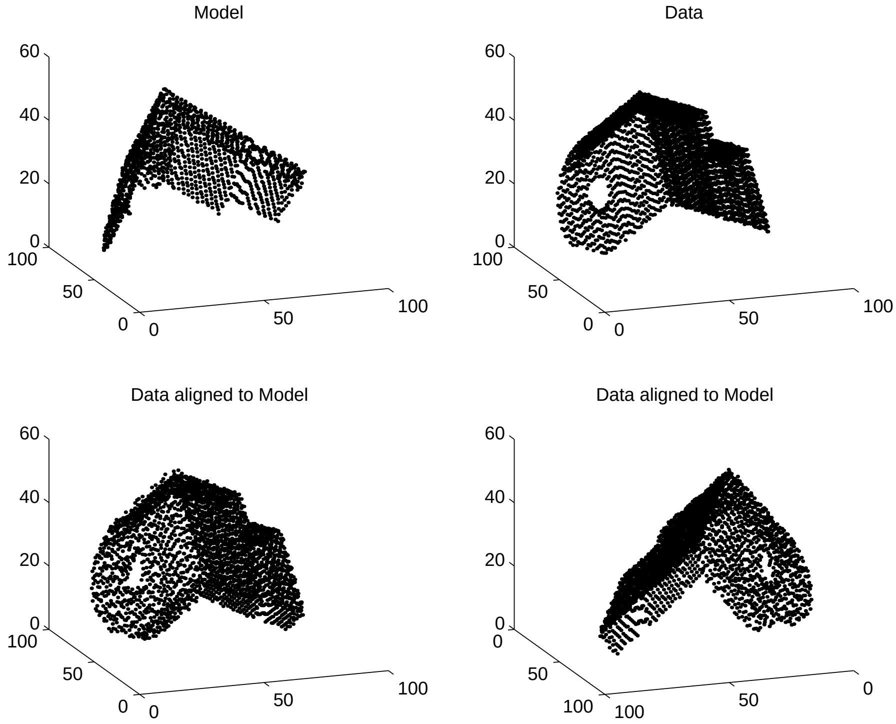

Figure 59: Two range views of a mechanical widget (top row). The registration found by RICP, from two viewpoints (bottom row). All views are subsampled for display.

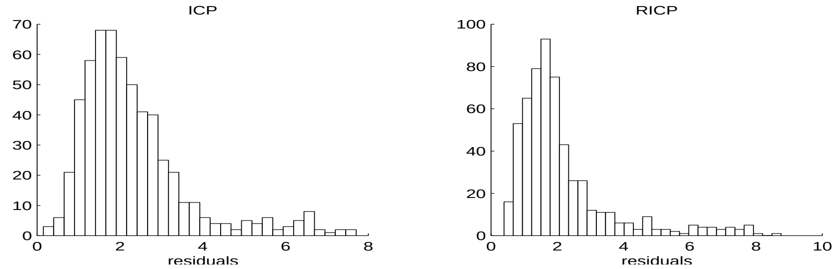

Figure 60: Residual histograms for the widget experiment.

8.5 Conclusions

We have presented RICP, a robust version of the ICP algorithm for correspondence-less registration of sparse sets of 3-D points corrupted by sensor noise and outliers. RICP is based on the robust, high-efficiency estimator, LMedS, and implements a dynamic, weighted scheme for estimating translation using corresponding points only.

Unlike ICP, it works on sparse point sets, and tolerates substantial amounts of wrong measurements and missing data. With Gaussian noise only, the performances of ICP and RICP are very similar, and both RMS errors converge to the standard deviation of the noise. With outliers, RICP achieves more accurate alignments than ICP (indeed the better the higher the outlier percentage) and converges to the correct registration from a wider range of initial displacements.

Inevitably, RICP's robustness comes at the cost of a higher complexity. In our tests on a SPARCServer 10 running Solaris 2.5, RICP took, on average, 88 seconds to register synthetic clouds of 50 points with noise and outliers, ICP only half a second. This points strongly to off-line applications for RICP.

Chapter 9

Conclusions

This thesis makes five main contributions.

The first is a simple and compact rectification algorithm, developed in Chapter 3. The correct behavior of the algorithm has been demonstrated with both synthetic and real images. Tests showed that reconstruction performed directly from the disparities of the rectified images does not introduces appreciable errors compared with reconstructing from the original images.

The second major contribution is a new, efficient algorithm for stereo correspondence, SMW, based on a multi-window approach, and taking advantage of left-right consistency (Chapter 4). Tests showed the advantages offered by SMW. The adaptive, multi-window scheme yields robust disparity estimates in the presence of occlusions, and clearly outperforms single-window schemes. Left-right consistency proves effective in eliminating false matches and identifying occluded regions. In addition, disparity is assigned to occluded points heuristically, thereby achieving reasonable depth maps even in occluded areas. Uncertainty maps are also computed, allowing the use of SMW as a module within more complex data fusion frameworks. As for any area-based matching method, SMW's performance is affected adversely by poorly-textured regions, but areas of low texture are associated consistently with high uncertainty values.

Another contribution of this thesis is a robust extension of the Shi-Tomasi-Kanade tracker, based on the X84 outlier rejection rule (Chapter 6). The computational cost is much less than that of schemes based on robust regression and random sampling like RANSAC or LMedS. Yet experiments indicate excellent reliability in the presence of non-affine feature warping. The algorithm locates and discards unreliable features accurately and consistently (most right features are preserved, all wrong features are rejected), and tracks good features reliably over many frames. The fourth major contribution is an original, unified account of some of the most promising techniques for computing the Euclidean structure from uncalibrated images (Chapter 7). Such a comparative account, which does not yet exist in the literature, sheds light on the relations between different methods, presented in different ways and formalisms in the original research articles.

The last contribution of this thesis is RICP, a robust version of the ICP algorithm for correspondenceless registration of sparse sets of 3-D points corrupted by sensor noise and outliers (Chapter 8). RICP is based on LMedS regression, and implements a dynamic, weighted scheme for estimating translation using corresponding points only. Unlike ICP, it works on sparse point sets, and tolerates substantial amounts of wrong measurements and missing data. Inevitably, RICP's robustness comes at the cost of a higher complexity, and this points strongly to off-line applications for RICP.

These five contributions cover the main elements for building a robust system for structure recovery, coping with various degrees of a-priori knowledge. A complete system should include projective reconstruction and autocalibration, that could not be implemented during this research.

Appendix A

Projective Geometry

> Mine eye hath play'd the painter, and hath stell'd
> Thy beauty's form in table of my heart;
> My body is the frame wherein 'tis held,
> And perspective it is best painter's art.
> For through the painter must you see his skill,
> To find where your true image pictur'd lies,
> ...[1]

The understanding of perspective projections was one of the great achievements of the *Rinascimento* (Reneissance). The Italian architect F. Brunelleschi studied this topic in some detail, but the first explicit formulation of perspective projections is found in the treatise by L. B. Alberti *De Pictura* [1], written in 1435. This treatise describes a method for projecting the horizontal "plane of the floor" onto the vertical "plane of the painting". Piero della Francesca pushed the theory forward: in his *De Prospectiva Pingendi* [29], written in 1478, he dealt with the general problem of depicting 3-D objects and, as a painter, he also put his theory in practice (Figure 62).

In the XVII century G. Desargues, building on the works on perspective and on astronomical research by Keplero, introduced projective geometry as a tool for studying the conics (see [135]). Projective geometry, thanks to the concept of points at infinity, deals with elegance with all the particular cases found in theorems on conics.

[1] W. Shakespeare, Complete Sonnets, Dover Publications Inc, NY, 1991

From the analytic standpoint, the most important aspect of projective geometry is the introduction of homogeneous coordinates, which allows many of the significant aspects of projective geometry to be proven using linear algebra.

In this appendix some concepts of analytic projective geometry will be briefly reviewed and summarized for the reader's convenience. A more detailed knowledge of the subject can be acquired by reading [6, 160, 107, 33].

Figure 61: The well-known "Flagellazione" by Piero della Francesca, painted in 1460, Galleria Nazionale delle Marche, Urbino. This painting have been studied as one of the most important examples of perspective drawing [159].

Figure 62: "La camera degli Sposi" by Andrea Mantegna painted in fresco in 1474, Palazzo Ducale, Mantova.

Points and lines A *point* on the projective plane is represented by an ordered triple of real numbers $[x_1, x_2, x_3] \neq [0, 0, 0]$ with the convention that $[x_1, x_2, x_3]$ and $[\lambda x_1, \lambda x_2, \lambda x_3]$ – where $\lambda \neq 0$ – represent the same point.

A *line* on the projective plane is represented by an ordered triple of real numbers $[x_1, x_2, x_3] \neq [0, 0, 0]$ with the convention that $[x_1, x_2, x_3]$ and $[\lambda x_1, \lambda x_2, \lambda x_3]$ where $\lambda \neq 0$ represents the the same line . We shall see that a suitable coordinate system can be established in the plane, so that this number triplets are the coordinates of points.

Projective basis Four points $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$, no three of which are collinear, define a projective basis for the projective plane. Let us choose the representations (i.e., the

scale factors) of the first three points so that we have: $\mathbf{a}_1+\mathbf{a}_2+\mathbf{a}_3=\mathbf{a}_4$. In terms of this coordinate system, we define the relative *homogeneous coordinates* of any points $\mathbf{x}$ to be $[x_1,x_2,x_3]$ if $x=x_1\mathbf{a}_1+x_2\mathbf{a}_2+x_3\mathbf{a}_3$. The word "homogeneous" refers to the fact that the homogeneous coordinates of a projective point may be multiplied by any nonzero scalar. Note that the role of $\mathbf{a}_4$ is simply to fix the scale factors for $\mathbf{a}_1,\mathbf{a}_2$ and $\mathbf{a}_3$, which can be otherwise chosen arbitrarily. Indeed, if $\mathbf{x}=[x_1,x_2,x_3]$ and we change the representation for the reference points, the linear combination $x_1\lambda_1\mathbf{a}_1+x_2\lambda_2\mathbf{a}_2+x_3\lambda_3\mathbf{a}_3$ gives a representation of a point different from $\mathbf{x}$.
Any point $[x_1,x_2,x_3]$ may be written as $x_1[1,0,0]+x_2[0,1,0]+x_3[0,0,1]$, hence, referred to this coordinate system, it has relative homogeneous coordinates $[x_1,x_2,x_3]$. The coordinate system defined by the four points $[1,0,0],[0,1,0],[0,0,1],[1,1,1]$ is called the *natural coordinate system.*

Collinear points In the projective plane, points and lines are dual elements; the point $\mathbf{x}$ belongs to the line $\mathbf{y}$ if an only if their scalar product is zero, in symbols

$$\mathbf{x}\cdot\mathbf{y}=0. \tag{202}$$

When $\mathbf{x}$ is a variable point on the fixed line $\mathbf{y}$, (202) is called the equation of the line.
It can be easily proved that a necessary and sufficient condition for the distinct points $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ to be collinear is

$$\det(\mathbf{x},\mathbf{y},\mathbf{z})=0, \tag{203}$$

which is equivalent to

$$\mathbf{x}\cdot(\mathbf{y}\wedge\mathbf{z})=0. \tag{204}$$

Hence the line containing the two distinct points $\mathbf{y}$ e $\mathbf{z}$ is represented by $(\mathbf{y}\wedge\mathbf{z})$.
It can also be proved that if $\mathbf{y}$ e $\mathbf{z}$ are distinct points, then $\alpha\mathbf{y}+\beta\mathbf{z}$ with $\alpha,\beta\in\mathbb{R}^+$ is another point on the line determined by $\mathbf{y}$ e $\mathbf{z}$. If we let $\lambda=\beta/\alpha$ and accept the convention $\mathbf{y}+\lambda\mathbf{z}=\mathbf{z}$ when $\lambda=\infty$, the line containing the two distinct points $\mathbf{y}$ and $\mathbf{z}$ has parametric equation:

$$\mathbf{x}=\mathbf{y}+\lambda\mathbf{z}\quad \lambda\in\mathbb{R}\cup\{\infty\}\ . \tag{205}$$

Collineations A non-singular linear transformation of the projective plane into itself is called *collineation* (or *homography*).

The most general collineation is represented by a non-singular 3 × 3 matrix **H**:

$$\begin{bmatrix} \lambda x_1' \\ \lambda x_2' \\ \lambda \end{bmatrix} = \begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,1} & H_{3,2} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}. \tag{206}$$

The collineation maps points into points and lines into lines and preserves collinearity (hence its name).

The projective transformation matrix **H** requires eight independent parameters to define a unique mapping. Each point correspondence in the plane provides two equations:

$$\begin{bmatrix} x_1 & x_2 & 1 & 0 & 0 & 0 & -x_1x_1' & -x_2x_1' \\ 0 & 0 & 0 & x_1 & x_2 & 1 & -x_1x_2' & -x_2x_2' \end{bmatrix} \begin{bmatrix} H_{1,1} \\ H_{1,2} \\ H_{1,3} \\ H_{2,1} \\ H_{2,2} \\ H_{2,3} \\ H_{3,1} \\ H_{3,2} \end{bmatrix} = \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}. \tag{207}$$

It is then necessary to find four point correspondences to define the transformation matrix uniquely. This gives a constructive proof that four points (provided that no three of them are collinear) determine a unique transformation matrix. This is in agreement with the fact that a base for the projective plane is composed by four elements: the collineation is completely specified by its action on a base. This result, generalized in a projective space of any dimension, is known as the *fundamental theorem of the projective geometry.*

Cross ratio On the line determined by $\mathbf{y}$ and $\mathbf{z}$ take four points $\mathbf{a} = \mathbf{y} + \alpha\mathbf{z}$, $\mathbf{b} = \mathbf{y} + \beta\mathbf{z}$, $\mathbf{c} = \mathbf{y} + \gamma\mathbf{z}$ and $\mathbf{d} = \mathbf{y} + \delta\mathbf{z}$. We define the *cross ratio* of these points in terms of the parameters $\alpha, \beta, \gamma, \delta$ as

$$(\alpha, \beta; \gamma, \delta) = \frac{(\alpha - \gamma)(\beta - \delta)}{(\alpha - \delta)(\beta - \gamma)}. \tag{208}$$

The significance of the cross ratio is that it is invariant under collineations. The parameters $\alpha, \beta, \gamma, \delta$ can also be interpreted as the distances from a fixed point on the line.

Models for the projective plane In this paragraph we present two common models of projective spaces.

In the first model, we build up an affine space to form a projective space by inserting the directions of lines as additional points. The projective plane is built up from the affine plane by adding points at infinity (*ideal points*) in such a way that parallel lines always meet at an ideal point. Hence, we add one ideal point for each pencil of parallel lines. The set of all ideal points form the line at infinity.

In the second method we collapse a vector space to form a projective space by using the lines in that vector space as our projective points. Let V be an n-dimensional vector space. The associated $(n-1)$-dimensional projective space is $\bar{V} = \{Q|Q \text{ is a 1-dimensional subspace of V}\}$. A model for the projective plane is constituted by a pencil of lines in 3-D space, all emanating from the origin, and an arbitrary plane π, not passing through the origin. Each line represents a projective point. The lines which intersect the plane correspond to points in the affine plane, whereas lines parallel to π correspond to ideal points. Only the direction of lines is important in this model. This is in agreement with the homogeneous representation of projective points.

Although both these models are useful to understand projective geometry, the "collapsed vector space" approach is less cumbersome since one does not have to discuss two cases, one for ideal points and the other for affine points.

Axioms for the projective plane In defining the projective plane we took the analytic approach, introducing immediately coordinates. Yet, projective geometry is often formalized from a synthetic point of view. The following three statements are usually taken as axioms defining the projective plane:

(1) Two points are contained in one and only one line.

(2) Two lines intersect in exactly one point.

(3) There are four points such that no three are on the same line.

Homogeneous vs Cartesian coordinates Homogeneous coordinates (triplets of real numbers) are used to represent points on the projective planes. Representation is not unique, since $\mathbf{x}$ and $\lambda\mathbf{x}$ with $\lambda \in \mathbb{R}$ represent the same projective point. Cartesian coordinates (pairs of real numbers) are used to represent points in the affine plane. The representation is unique. Since the projective plane can be viewed as an extended affine plane, we can draw a relationship between representations of affine points (ideal points, of course, do not have a Cartesian representation). From the "collapsed vector space" point of view, the Cartesian representation of a projective point are the coordinates, in the plane $\pi : x_3 = 1$, of the intersection of the line representing the projective point with the plane π. The Cartesian coordinates corresponding to a projective point $[x_1, x_2, x_3]$ are $[x_1/x_3, x_2/x_3]$. Vice versa, the homogeneous representation of the point $[x_1, x_2]$ is $\lambda[x_1, x_2, 1]$ with $\lambda \in \mathbb{R}^+$.

List of symbols

$I(\cdot,\cdot)$	image brightness
$\tilde{\mathbf{P}} = [\mathbf{Q}\vert\tilde{\mathbf{q}}]$	perspective projection matrix (camera matrix)
$\tilde{\mathbf{w}}$	homogeneous coordinates of a world point
$\tilde{\mathbf{m}}$	homogeneous coordinates (in pixels) of an image point
$\tilde{\mathbf{p}}$	normalized homogeneous coordinates of an image point (ray vector)
κ	relative depth
$\simeq$	equality up to an arbitrary scale factor
λ	arbitrary scale factor
$\mathcal{F}$	focal plane
$\mathcal{R}$	retinal plane
(u, v)	image reference frame
(x, y, z)	world reference frame
(X, Y, Z)	camera std reference frame
$\mathbf{A}$	intrinsic parameters matrix
f	focal distance
k_u	effective pixel horizontal size
k_v	effective pixel vertical size
α_u	focal distance in horizontal pixels
α_v	focal distance in veritcal pixels
(u_0, v_0)	principal point
γ	skew factor
$\mathbf{G} = [\mathbf{R}\vert\mathbf{t}]$	extrinsic parameters matrix
$\mathbf{R}$	rotation matrix
$\mathbf{t}$	translation vector

$\mathbf{c}$	optical center
$\wedge$	external product
$[\ ]_\wedge$	external product matrix
$\mathrm{diag}(\dots)$	diagonal matrix; the arguments are the diagonal elements
$\mathrm{trace}(\cdot)$	sum of the diagonal elements of a matrix
$\mathbf{e}$	epipole
$\mathbf{E}$	essential matrix
$\mathbf{F}$	fundamental matrix
$[\ ]_i$	projection operator extracting the i-th component
$\mathbf{H}_\Pi$	homography matrix of plane Π
$\mathbf{H}_\infty$	infinity plane homography matrix
$\mathbf{K} = \mathbf{A}\mathbf{A}^\top$	Kruppa's coefficients matrix

Credits

The software for this thesis was written by the author, with the exceptions listed below. It was coded in MATLAB (© Copyright The Math Works Inc.), SCILAB[2], a public domain MATLAB-like package from INRIA (registered at APP under the number 93-27-011-00) or in ANSI C, and compiled using the public domain GNU `gcc` compiler v2.7 (© Copyright Free Software Foundation, Inc.). Some programs in C inclues *Meschach* (© Copyright David E. Stewart), a public domain library for linear algebra (available from Netlib[3]) and the *Numerical Recipes* [120] routines in C (© Copyright Numerical Recipes Software). Images are read, written and visualized using the HIPS (© Copyright SharpImage Software) package. The calibration code is by Luc Robert[4] (INRIA), and the *Calibtool* interface was written in Tcl/Tk by a Erman Petrei (University of Udine). The tracking algorithm was coded by Tiziano Tommasini [140], and the RICP code is due to Stefano Morson, Orazio Stangherlin and Gerard Martin [98], who also helped in performing the experiments.

Computing and laboratory facilities were provided by the Dipartimento di Matematica ed Informatica, University of Udine, by courtesy of Vito Roberto. Visits to the Heriot-Watt University, Edinburgh, was supported by a British Council grant the first time and by a EC Socrates grant the second time.

Stereo pairs in Chapter 3 (© Copyright INRIA-Syntim[5]) were calibrated by Jean-Philippe Tarel [134]. The stereo pairs "Parking meter", "Shrub", and "Trees" in Chapter 4 are part of the JISCT (JPL-INRIA-SRI-CMU-TELEOS) stereo test set[6];

[2] http://www-rocq.inria.fr/scilab/scilab.html

[3] http://www.netlib.org/

[4] http://www.inria.fr/robotvis/personnel/lucr/detecproj.html

[5] http://www-syntim.inria.fr/syntim/analyse/paires-eng.html

[6] ftp://ftp.vislist.com/IMAGERY/JISCT/

the “Castle” stereo pair, with calibration data, is courtesy of Carnegie Mellon University — Calibrated Imaging Laboratory[7] (supported by ARPA, NSF, and NASA). The “Hotel” and “Artichoke” sequences in chapter 6 are taken from the Carnegie Mellon University – VASC[8] database. “Platform” is part of the SOFA[9] synthetic sequences, courtesy of the Computer Vision Group, Heriot-Watt University. “Stair” is courtesy of Francesco Isgrò, Heriot-Watt University. Thanks to Bob Fisher and Anthony Ashbrooks (Department of Artificial Intelligence, University of Edinburgh) for the widget data used in Chapter 8. The calibration jig was designed by Alessandro Verri and manufactured at the University of Genova, Department of Phisics. Parts of this dissertation are adapted from papers written by the candidate during this three years, in conjuction with other authors: Vito Roberto, Tiziano Tommasini, Emanuele Trucco and Alessandro Verri. In particular, Chapter 3 incorporates a revised version of [46, 42]; Chapter 4 comes from the merging of [44, 45, 124]; Chapter 6 is a revised version of [141]; Chapter 7 is based on [43] and Chapter 8 is adapted from [147].

[7] http://www.cs.cmu.edu/afs/cs/project/cil/www/cil-ster.html

[8] http://www.ius.cs.cmu.edu/idb/

[9] http://www.cee.hw.ac.uk/~mtc/sofa

Bibliography

[1] Alberti, L. *De pictura.* Bari, Italy, 1980. edited by C. Grayson.

[2] Anandan, P. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision 2* (1989), 283–310.

[3] Armstong, M., Zisserman, A., and Hartley, R. Self-calibration from image triplets. In *Proceedings of the European Conference on Computer Vision* (Cambridge, UK, 1996), pp. 5–16.

[4] Avidan, S., and Shashua, A. Threading fundamental matrices. In *Proceedings of the European Conference on Computer Vision* (University of Freiburg, Germany, 1998), pp. 124–140.

[5] Ayache, N. *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception.* The MIT Press, 1991.

[6] Ayres, F. *Theory and Problems of Projective Geometry.* Schaum's Outline Series in Mathematics. McGraw-Hill, 1967.

[7] Baker, H., and Binford, T. Depth from edge and intensity based stereo. In *Proceedings of the International Joint Conference on Artificial Intelligence* (1981), pp. 631–636.

[8] Barnard, S. T., and Thompson, W. B. Disparity analysis of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 2*, 4 (1980), 333–340.

[9] Barron, J. L., Fleet, D. J., and Beauchemin, S. Performance of optical flow techniques. *International Journal of Computer Vision 12*, 1 (1994), 43–77.

[10] Beardsley, P., Zisserman, A., and Murray, D. Sequential update of projective and affine structure from motion. *International Journal of Computer Vision 23*, 3 (1997), 235–259.

[11] Belhumeur, P. N. A Bayesian approach to binocular stereopsis. *International Journal of Computer Vision 19*, 3 (1996), 237–260.

[12] Benedetti, A., and Perona, P. Real-time 2-d feature detection on a reconfigurable computer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Santa Barbara, CA, June 1998), IEEE Computer Society Press, pp. 586–593.

[13] Besl, P., and McKay, N. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 2 (February 1992), 239–256.

[14] Bispo, E. M., and Fisher, R. B. Free-form surface matching for surface inspection. In *The Mathematics of Surfaces VI*, G. Mullineux, Ed. Clarendon Press, Oxford, 1996, pp. 119–136.

[15] Blake, R., and Wilson, H. R. Neural models of stereoscopic vision. *Trends in Neuroscience 14* (1991), 445–452.

[16] Bolles, R. C., Baker, H. H., and Hannah, M. J. The JISCT stereo evaluation. In *Proceedings of the Image Understanding Workshop* (Washington, DC, April 1993), ARPA, Morgan Kaufmann, pp. 263–274.

[17] Bougnoux, S. From projective to Euclidean space under any practical situation, a criticism of self-calibration. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, 1998), pp. 790–796.

[18] Brandt, J. W. Improved accuracy in gradient-based optical flow estimation. *International Journal of Computer Vision 25*, 1 (1997), 5–22.

[19] Brujic, D., and Ristic, M. Analysis of free-form surface registration. In *Proceedings of the IEEE International Conference on Image Processing* (Lausanne, Switzerland, 1996), vol. II, pp. 393–396.

[20] Campani, M., and Verri, A. Motion analysis from first order properties of optical flow. *CVGIP: Image Understanding 56* (1992), 90–107.

[21] Campus, O., Flynn, P. J., and Stockman, G. C., Eds. *Special Issue on CAD-Based Computer Vision* (March 1998), vol. 69:3 of *Computer Vision and Image Understanding.*

[22] Caprile, B., and Torre, V. Using vanishing points for camera calibration. *International Journal of Computer Vision 4* (1990), 127–140.

[23] Charnley, D., and Blisset, R. J. Surface reconstruction from outdoor image sequences. *Image and Vision Computing 7*, 1 (1989), 10–16.

[24] Chen, Y., and Medioni, G. Object modeling by registration of multiple range images. *Image and Vision Computing 10*, 3 (1992), 145–155.

[25] Cox, I., Roy, S., and Hingorani, S. Dynamic histogram warping of image pairs for constant image brightness. In *Proceedings of the IEEE International Conference on Image Processing* (Washington, D.C, 1995), vol. 2, pp. 366–369.

[26] Cox, I. J., Hingorani, S., Maggs, B. M., and Rao, S. B. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding 63*, 3 (May 1996), 542–567.

[27] Dacol, L. Integrazione di tecniche per la ricostruzione di forme in visione artificiale. Tesi di laurea in scienze dell'informazione, Università degli Studi di Udine, 1998.

[28] Davies, E. *Machine Vision: Theory, Algorithms, Practicalities*, 2nd ed. Academic Press, 1997.

[29] della Francesca, P. *De Prospectiva Pingendi.* Edizione critica. Sansoni, Firenze, Italy, 1942.

[30] Dhond, U. R., and Aggarwal, J. K. Structure from stereo – a review. *IEEE Transactions on Systems, Man and Cybernetics 19*, 6 (November/December 1989), 1489–1510.

[31] Eggert, D. W., Fitzgibbon, A. W., and Fisher, R. B. Simultaneous registration of multiple range views for use in reverse engineering. In *Proceedings of the International Conference on Pattern Recognition* (Vienna, 1996), pp. 243–247.

[32] Faugeras, O. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proceedings of the European Conference on Computer Vision* (Santa Margherita L., 1992), pp. 563–578.

[33] Faugeras, O. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* The MIT Press, Cambridge,MA, 1993.

[34] Faugeras, O. Stratification of 3-D vision: projective, affine, and metric representations. *Journal of the Optical Society of America A 12*, 3 (1994), 465–484.

[35] Faugeras, O., Hotz, B., Mathieu, H., Viéville, T., Zhang, Z., Fua, P., Théron, E., Moll, L., Berry, G., Vuillemin, J., Bertin, P., and Proy, C. Real-time correlation-based stereo: algorithm, implementation and applications. Tech. Rep. 2013, Unité de recherche INRIA Sophia-Antipolis, August 1993.

[36] Faugeras, O., and Maybank, S. Motion from point matches: multiplicity of solutions. *International Journal of Computer Vision 4*, 3 (June 1990), 225–246.

[37] Faugeras, O., Robert, L., Laveau, S., Csurka, G., Zeller, C., Gauclin, C., and Zoghlami, I. 3-d reconstruction of urban scenes from image sequences. *Computer Vision and Image Understanding 69*, 3 (March 1998), 292–309.

[38] Faugeras, O., and Toscani, G. Camera calibration for 3D computer vision. In *Proceedings of the International Workshop on Machine Vision and Machine Intelligence* (Tokyo, Japan, February 1987).

[39] Fischler, M. A., and Bolles, R. C. Random Sample Consensus: a paradigm model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6 (June 1981), 381–395.

[40] Foley, J. D., van Dam, A., Feiner, S. K., Huges, J. F., and Phillips, R. L. *Introduction to Computer Graphics.* Addison-Wesley, 1990.

[41] Fua, P. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *Proceedings of the International Joint Conference on Artificial Intelligence* (Sydney, Australia, August 1991), pp. 1292–1298.

[42] Fusiello, A. Tutorial on rectification of stereo images. In *CVonline: On-Line Compendium of ComputerVision [Online]*, R. Fisher, Ed. 1998. Available: http://www.dai.ed.ac.uk/CVonline/.

[43] Fusiello, A. Uncalibrated Euclidean reconstruction: A review. Research Report UDMI/10/98/RR, Dipartimento di Matematica e Informatica, Università di Udine, July 1998. Submitted for publication in *Computing.*

[44] Fusiello, A., Roberto, V., and Trucco, E. Efficient stereo with multiple windowing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Puerto Rico, June 1997), IEEE Computer Society Press, pp. 858–863.

[45] Fusiello, A., Roberto, V., and Trucco, E. Experiments with a new area-based stereo algorithm. In *Proceedings of the 9th International Conference on Image Analysis and Processing* (Florence, Italy, September 1997), A. Del Bimbo, Ed., IAPR, Springer, pp. 669–676. Lecture Notes in Computer Science 1310.

[46] Fusiello, A., Trucco, E., and Verri, A. Rectification with unconstrained stereo geometry. In *Proceedings of the British Machine Vision Conference* (September 1997), A. F. Clark, Ed., BMVA Press, pp. 400–409.

[47] Geiger, D., Ladendorf, B., and Yuille, A. Occlusions and binocular stereo. *International Journal of Computer Vision 14*, 3 (April 1995), 211–226.

[48] Gill, P., Murray, W., and Wright, M. *Practical Optimization.* Academic Press, 1981.

[49] Goldgof, D. B., Lee, H., and Huang, T. Matching and motion estimation of three-dimensional point and line sets using eigenstructure without correspondence. *Pattern Recognition 25*, 3 (1992), 271–286.

[50] Golub, G. H., and Loan, C. F. V. *Matrix Computations*, third ed. The John Hopkins University Press, 1996.

[51] Grimson, W. A computer implementation of a theory of human stereo vision. *Philosophical Transactions of the Royal Society of London, B. 292*, 1058 (1981), 217–253.

[52] Grimson, W. Computational experiments with a feature based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence 7*, 1 (January 1985), 17–34.

[53] Grimson, W. E. L. *From Images to Surfaces.* The MIT Press, 1981.

[54] Hager, G., and Belhumeur, P. Real-time tracking of image regions with changes in geometry and illumination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (San Francisco, CA, 1996), pp. 403–410.

[55] Hampel, F., Rousseeuw, P., Ronchetti, E., and Stahel, W. *Robust Statistics: the Approach Based on Influence Functions.* Wiley Series in probability and mathematical statistics. John Wiley & Sons, 1986.

[56] Hannah, M. A system for digital stereo image matching. *Photogrammatic Engineering and Remote Sensing* (1989), 1765–1770.

[57] Harris, C., and Stephens, M. A combined corner and edge detector. *Proceedings of the 4th Alvey Vision Conference* (August 1988), 189–192.

[58] Hartley, R. Euclidean reconstruction from uncalibrated views. In *Applications of Invariance in Computer Vision* (1993), J. Mundy and A. Zisserman, Eds., vol. 825 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 237–256.

[59] Hartley, R. Self-calibration from multiple views with a rotating camera. In *Proceedings of the European Conference on Computer Vision* (Stockholm, 1994), pp. 471–478.

[60] Hartley, R., and Gupta, R. Computing matched-epipolar projections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (1993), pp. 549–555.

[61] Hartley, R. I. Estimation of relative camera position for uncalibrated cameras. In *Proceedings of the European Conference on Computer Vision* (Santa Margherita L., 1992), pp. 579–587.

[62] Hartley, R. I. Cheirality invariants. In *Proceedings of the Image Understanding Workshop* (Washington, DC, April 1993), ARPA, Morgan Kaufmann, pp. 745–753.

[63] Hartley, R. I. In defence of the 8-point algorithm. In *Proceedings of the IEEE International Conference on Computer Vision* (1995).

[64] Hartley, R. I. Kruppa's equations derived from the fundamental matrix. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 2 (February 1997), 133–135.

[65] Hartley, R. I. Minimizing algebraic error. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, 1998).

[66] Hartley, R. I., and Sturm, P. Triangulation. *Computer Vision and Image Understanding 68*, 2 (November 1997), 146–157.

[67] Henkel, R. D. Fast stereovision with subpixel-precision. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, 1998), pp. 1024–1028.

[68] Heyden, A., and Åström, K. Euclidean reconstruction from constant intrinsic parameters. In *Proceedings of the International Conference on Pattern Recognition* (Vienna, 1996), pp. 339–343.

[69] Heyden, A., and Åström, K. Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Puerto Rico, 1997), pp. 438–443.

[70] Heyden, A., and Åström, K. Minimal conditions on intrinsic parameters for Euclidean reconstruction. In *Proceedings of the Asian Conference on Computer Vision* (Hong Kong, 1998).

[71] Horn, B. Relative orientation. *International Journal of Computer Vision 4*, 1 (January 1990), 59–78.

[72] Horn, B. Relative orientation revisited. *Journal of the Optical Society of America A 8*, 10 (October 1991), 1630–1638.

[73] Huang, T. S., and Netravali, A. N. Motion and structure from feature correspondences: A review. *Proceedings of IEEE 82*, 2 (1994), 252–267.

[74] Intille, S. S., and Bobick, A. F. Disparity-space images and large occlusion stereo. In *European Conference on Computer Vision* (Stockholm, Sweden, May 1994), J.-O. Eklundh, Ed., Springer-Verlag, pp. 179–186.

[75] Ito, E., and Aloimonos, Y. Is correspondence necessary for the perception of structure from motion? In *Proceedings of the Image Understanding Workshop* (1988), pp. 921–929.

[76] Jenkin, M. R. M., and Jepson, A. D. Recovering local surface structure through local phase difference measurements. *CVGIP: Image Understanding 59*, 1 (1994), 72–93.

[77] Jenkin, M. R. M., Jepson, A. D., and Tsotsos, J. K. Techniques for disparity measurements. *CVGIP: Image Understanding 53*, 1 (1991), 14–30.

[78] Jones, D. G., and Malik, J. Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing 10*, 10 (1992), 699–708.

[79] Kanade, T., and Okutomi, M. A stereo matching algorithm with an adaptive window: Theory and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*, 9 (September 1994), 920–932.

[80] Kanatani, K. *Geometric Computation for Machine Vision.* Oxford University Press, 1993.

[81] Kass, M., Witkin, A., and Terzopulos, D. Snakes: Active contour models. *International Journal of Computer Vision 1*, 4 (1988), 321–331.

[82] Kruppa, E. Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung. *Sitz.-Ber. Akad. Wiss., Wien, math. naturw. Kl., Abt. IIa. 122* (1913), 1939–1948.

[83] Lee, C., and Joshi, A. Correspondence problem in image sequence analysis. *Pattern Recognition 26* (1993), 47–61.

[84] Lindeberg, T. Scale space for discrete signals. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12* (1990), 234–254.

[85] Little, J. J., and Gillett, W. E. Direct evidence for occlusions in stereo and motion. *Image and Vision Computing 8*, 4 (November 1990), 328–340.

[86] Longuet-Higgins, H. C. A computer algorithm for reconstructing a scene from two projections. *Nature 293*, 10 (September 1981), 133–135.

[87] Lorusso, A., Eggert, D. W., and Fisher, R. B. A comparison of four algorithms for estimating 3-D rigid transformations. *Machine Vision and Applications 9* (1997), 272–290.

[88] Lowe, D. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 5 (May 1991), 441–450.

[89] Lucas, B. D., and Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence* (1981).

[90] Ludwig, K. O., Neumann, H., and Neumann, B. Local stereoscopic depth estimation. *Image and Vision Computing 12*, 10 (1994), 16–35.

[91] Luong, Q.-T., Deriche, R., Faugeras, O., and Papadopoulo, T. On determining the fundamental matrix: Analysis of different methods and experimental results. Tech. Rep. 1894, Unité de recherche INRIA Sophia-Antipolis, April 1993.

[92] Luong, Q.-T., and Faugeras, O. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision 22*, 3 (1997), 261–289.

[93] Luong, Q.-T., and Faugeras, O. D. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision 17* (1996), 43–75.

[94] Luong, Q.-T., and Viéville, T. Canonical representations for the geometries of multiple projective views. *Computer Vision and Image Understanding 64*, 2 (1996), 193–229.

[95] Ma, Y., Košecká, J., and Sastry, S. Motion recovery from image sequences: Discrete viewpoint vs. differential viewpoint. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, 1998), pp. 337–353.

[96] Marr, D., and Poggio, T. Cooperative computation of stereo disparity. *Science 194* (1976), 283–287.

[97] Marr, D., and Poggio, T. A theory of human stereo vision. A.I. Memo 451, Massachusetts Institute of Technology, November 1977.

[98] Martin, G. Registrazione di forme in visione artificiale. Tesi di laurea in scienze dell'informazione, Università degli Studi di Udine, 1998.

[99] Masuda, T., and Yokoya, N. A robust method for registration and segmentation of multiple range images. *Computer Vision and Image Understanding 61*, 3 (May 1995), 295–307.

[100] Maybank, S. J., and Faugeras, O. A theory of self-calibration of a moving camera. *International Journal of Computer Vision 8*, 2 (1992), 123–151.

[101] Maybeck, P. S. *Stochastic Models, Estimation and Control*, vol. 1. Academic Press: New York, NY, 1979.

[102] Medioni, G., and Nevatia, R. Segment-based stereo matching. *Computer Vision, Graphics, and Image Processing 31*, 1 (July 1985), 2–18.

[103] Meer, P., Mintz, D., Kim, D. Y., and Rosenfeld, A. Robust regression methods in computer vision: a review. *International Journal of Computer Vision 6* (1991), 59–70.

[104] Mitchie, A., and Bouthemy, P. Computation and analysis of image motion: A synopsis of current problems and methods. *International Journal of Computer Vision 19*, 1 (1996), 29–55.

[105] Moravec, H. P. Towards automatic visual obstacle avoidance. In *Proceedings of the International Joint Conference on Artificial Intelligence* (August 1977), p. 584.

[106] Morgan, A. P., Sommese, A. J., and Watson, L. T. Finding all isolated solutions to polynomial systems using hompack. *ACM Trans. Math. Software 15* (1989), 93–122.

[107] Mundy, J., and Zisserman, A. *Geometric Invariance in Computer Vision.* MIT Press, 1992, ch. 23.

[108] Nishihara, H. K. PRISM, a practical real-time imaging stereo matcher. A.I. Memo 780, Massachusetts Institute of Technology, 1984.

[109] Noble, J. Finding corners. *Image and Vision Computing 6* (May 1988), 121–128.

[110] Ohta, Y., and Kanade, T. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence 7*, 2 (March 1985), 139–154.

[111] Okutomi, M., and Kanade, T. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*, 4 (April 1993), 353–363.

[112] Papadimitriou, D. V., and Dennis, T. J. Epipolar line estimation and rectification for stereo images pairs. *IEEE Transactions on Image Processing 3*, 4 (April 1996), 672–676.

[113] Plakas, K., Trucco, E., and Fusiello, A. Uncalibrated vision for 3-D underwater applications. In *Proceedings of the OCEANS'98 Conference* (Nice, France, September 1998), IEEE/OS, pp. 272–276.

[114] Poelman, C. J., and Kanade, T. A paraperspective factorization method for shape and motion recovery. Technical Report CMU-CS-93-219, Carnegie Mellon University, Pittsburg, PA, December 1993.

[115] Poggio, T., Torre, V., and Koch, C. Computational vision and regularization theory. *Nature 317* (September 1985), 314–319.

[116] Pollard, S. B., Mayhew, J., and Frisby, J. PMF: A stereo correspondence algorithm using a disparity gradient constraint. *Perception 14* (1985), 449–470.

[117] Pollefeys, M., Koch, R., and Van Gool, L. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, 1998), pp. 90–95.

[118] Pollefeys, M., and Van Gool, L. A stratified approach to metric self-calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Puerto Rico, 1997), pp. 407–412.

[119] Pollefeys, M., Van Gool, L., and Oosterlinck, A. The modulus constraint: a new constraint for self-calibration. In *Proceedings of the International Conference on Pattern Recognition* (Vienna, 1996), pp. 349–353.

[120] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes in C: The Art of Scientific Computing*, second ed. Cambridge University Press, 1992.

[121] Robert, L. Camera calibration without feature extraction. *Computer Vision, Graphics, and Image Processing 63*, 2 (March 1996), 314–325.

[122] Robert, L., and Faugeras, O. Relative 3-D positioning and 3-D convex hull computation from a weakly calibrated stereo pair. *Image and Vision Computing 13*, 3 (1995), 189–197.

[123] Robert, L., Zeller, C., Faugeras, O., and Hébert, M. Applications of non-metric vision to some visually-guided robotics tasks. In *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, Y. Aloimonos, Ed. Lawrence Erlbaum Associates, 1997, ch. 5, pp. 89–134.

[124] Roberto, V., Yeshurun, Y., Fusiello, A., and Trucco, E. On stereo fusion in humans and machines. Research Report UDMI/54/96/RR, Dipartimento di Matematica e Informatica, Università di Udine, December 1996.

[125] Rogers, D. F., and Adams, J. *Mathematical Elements for Computer Graphics.* The MIT Press, 1996.

[126] Rousseeuw, P. J., and Leroy, A. M. *Robust regression & outlier detection.* John Wiley & sons, 1987.

[127] Shapiro, L., Wang, H., and Brady, J. A matching and tracking strategy for independently moving objects. In *Proceedings of the British Machine Vision Conference* (1992), BMVA Press, pp. 306–315.

[128] Shi, J., and Tomasi, C. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (June 1994), pp. 593–600.

[129] Soatto, S., and Brockett, R. Optimal structure from motion: Local ambiguities and global estimates. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Santa Barbara, CA, June 1998), pp. 282–288.

[130] Soatto, S., Frezza, R., and Perona, P. Motion estimation via dynamic vision. *IEEE Transactions on Automatic Control 41*, 3 (March 1996), 393–413.

[131] Sommese, A. J., and Wampler, C. W. Numerical algebraic geometry. In *Mathematics of Numerical Analysis: Real Number Algorithms*, J. Renegar, M. Shub, and S. Smale, Eds., vol. 32 of *Lectures in Applied Mathematics.* Park City, Utah, 1996, pp. 749–763.

[132] Stoddart, A. J., Lemke, S., Hilton, A., and Renn, T. Estimating pose uncertainty for surface registration. *Image and Vision Computing 16* (1998), 111–120.

[133] Sturm, P., and Triggs, B. A factorization based algorithm for multi-image projective structure and motion. In *Proceedings of the European Conference on Computer Vision* (Cambridge, UK, 1996), pp. 709–720.

[134] Tarel, J.-P., and Gagalowicz, A. Calibration de caméra à base d'ellipses. *Traitement du Signal 12*, 2 (1995), 177–187.

[135] Taton, R. *L'oeuvre mathématique de Desargues.* Paris, 1951.

[136] Tian, T., Tomasi, C., and Heeger, D. Comparison of approaches to egomotion computation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (San Francisco, CA, June 1996), pp. 315–320.

[137] Tomasi, C., and Kanade, T. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, April 1991.

[138] Tomasi, C., and Kanade, T. Shape and motion from image streams under orthography – a factorization method. *International Journal of Computer Vision 9*, 2 (Nov. 1992), 137–154.

[139] Tomasi, C., and Manduchi, R. Stereo without search. In *European Conference on Computer Vision* (Cambridge (UK), April 1996), B. Buxton and R. Cipolla, Eds., pp. 452–465.

[140] TOMMASINI, T. Un metodo robusto per l'inseguimento di punti caratteristici in visione artificiale. Tesi di laurea in scienze dell'informazione, Università degli Studi di Udine, 1998.

[141] TOMMASINI, T., FUSIELLO, A., TRUCCO, E., AND ROBERTO, V. Making good features track better. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Santa Barbara, CA, June 1998), IEEE Computer Society Press, pp. 178–183.

[142] TORR, P., AND MURRAY, D. W. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision 24*, 3 (September 1997), 271–300.

[143] TORR, P. H. S., AND ZISSERMAN, A. Robust parameterization and computation of the trifocal tensor. In *British Machine Vision Conference* (September 1996), R. Fisher and E. Trucco, Eds., BMVA, pp. 655–664. Edinburgh.

[144] TORR, P. H. S., ZISSERMAN, A., AND MAYBANK, S. Robust detection of degeneracy. In *Proceedings of the IEEE International Conference on Computer Vision* (1995), E. Grimson, Ed., Springer–Verlag, pp. 1037–1044.

[145] TRIGGS, B. Autocalibration and the absolute quadric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Puerto Rico, 1997), pp. 609–614.

[146] TRUCCO, E., FISHER, R. B., FITZGIBBON, A. W., AND NAIDU, D. K. Calibration, data consistency and model acquisition with a 3-D laser striper. *International Journal of Computer Integrated Manufacturing 11*, 4 (1998), 293–310.

[147] TRUCCO, E., FUSIELLO, A., AND ROBERTO, V. Robust motion and correspondences of noisy 3-D point sets with missing data, 1998. Submitted for publication in *Pattern Recognition Letter.*

[148] TRUCCO, E., ROBERTO, V., TINONIN, S., AND CORBATTO, M. SSD disparity estimation for dynamic stereo. In *Proceedings of the British Machine Vision Conference* (1996), R. B. Fisher and E. Trucco, Eds., BMVA Press, pp. 342–352.

[149] Trucco, E., and Verri, A. *Introductory Techniques for 3-D Computer Vision.* Prentice-Hall, 1998.

[150] Tsai, R. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation 3*, 4 (August 1987), 323–344.

[151] Ueshiba, T., and Tomita, F. A factorization method for projective and Euclidean reconstruction from multiple perspective views via iterative depth estimation. In *Proceedings of the European Conference on Computer Vision* (University of Freiburg, Germany, 1998), pp. 296–310.

[152] Ullman, D. *High-level Vision.* The MIT Press, 1996.

[153] Verri, A., and Trucco, E. Finding the epipole from uncalibrated optical flow. In *Proceedings of the IEEE International Conference on Computer Vision* (Bombay, India, 1998), pp. 987–991.

[154] Viéville, T. Autocalibration of visual sensor parameters on a robotic head. *Image and Vision Computing 12*, 4 (1994), 227–237.

[155] Viéville, T., Zeller, C., and Robert, L. Using collineations to compute motion and structure in an uncalibrated image sequence. *International Journal of Computer Vision 20*, 3 (1996), 213–242.

[156] Wang, X., Cheng, Y., Collins, R., and Hanson, R. Determining correspondences and rigid motion of 3-D point sets with missing data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (San Francisco, CA, June 1996), pp. 252–257.

[157] Weng, J., Ahuja, N., and Huang, T. Matching two perspective views. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 8 (1992), 806–825.

[158] Weng, J., Ahuja, N., and Huang, T. Optimal motion and structure estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*, 9 (September 1993), 864–884.

[159] Wittkower, R., and Carter, B. A. R. The perspective of Piero della Francesca's Flagellation. *Journal of the Warburg and Courtauld Institutes 16* (1953), 292–302.

[160] Yale, P. B. *Geometry and Symmetry.* Dover, 1988.

[161] Yang, Y., and Yuille, A. L. Multilevel enhancement and detection of stereo disparity surfaces. *Artificial Intelligence 78*, 1-2 (October 1995), 121–145.

[162] Yeshurun, Y., and Schwartz, E. Cepstral filtering on a columnar image architecture: a fast algorithm for binocular stereo segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 11* (1989), 759–767.

[163] Zabih, R., and Woodfill, J. Non-parametric local transform for computing visual correspondence. In *Proceedings of the European Conference on Computer Vision* (Stockholm, 1994), pp. 151–158.

[164] Zeller, C., and Faugeras, O. Camera self-calibration from video sequences: the Kruppa equations revisited. Research Report 2793, INRIA, Feb. 1996.

[165] Zhang, R., Tsai, P. S., Cryer, J. E., and Shah, M. Analysis of shape from shading techniques. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Seattle, Washington, June 1994), pp. 377–384.

[166] Zhang, Z. Iterative point matching of free-form curves and surfaces. *International Journal of Computer Vision 13*, 2 (1994), 119–152.

[167] Zhang, Z. A new multistage approach to motion and structure estimation: From essential parameters to Euclidean motion via fundamental matrix. Rapport de Recherche 2910, INRIA, Institut National de Recherche en Informatique et an Automatique, June 1996.

[168] Zhang, Z. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision 27*, 2 (March/April 1998), 161–195.

[169] Zhang, Z., Luong, Q.-T., and Faugeras, O. Motion of an uncalibrated stereo rig: self-calibration and metric reconstruction. *IEEE Transactions on Robotics and Automation 12*, 1 (Feb. 1996), 103–113.

[170] Zheng, Q., and Chellappa, R. Automatic feature point extraction and tracking in image sequences for arbitrary camera motion. *International Journal of Computer Vision 15*, 15 (1995), 31–76.

Index