Computer Vision using Shape Spaces

Burzin Bhavnagri
Department of Computer Science and
Department of Pure Mathematics
University of Adelaide

List of Figures

Abstract

This thesis investigates a computational model of vision based on assumptions pertaining to the physical structure of a camera and the scattering of light from visible surfaces.

A sufficient condition to detect occlusions, intensity discontinuities, discontinuities in derivatives of intensity, surface discontinuities and discontinuities in derivatives of surfaces is given. This leads to an algorithm with linear time and space complexity to generate a collection of feature points with attributes in cyclically ordered groups.

For each pair of feature point groups in two images, with m and n points respectively, we give an algorithm to compute a hypothesis of m corresponding point pairs in time $O(mn \log m)$. We develop two approaches to rejecting false hypotheses of correspondence: an error minimising approach and an approach based on a formal language.

In the error minimising approach we use a function that is zero precisely when there exists camera parameters and a scene such that each image is an image of the same scene. Convergence and stability conditions for such a function are formulated. It is proved that in the epipolar geometry model, such functions do not exist for general three dimensional scenes. It is also proved that in the new model proposed in this thesis, such a function does exist for general three dimensional scenes. In the affine camera model an algorithm to compute such a function is given in general. An algorithm to compute such a function for planar scenes in a fronto-parallel plane to a camera is given also.

Two images of a scene do not themselves determine a metric reconstruction of that scene. We present a non-iterative algorithm that can use the rotation between two cameras (and principal points) to produce an exact reconstruction. This algorithm requires data that are in correspondence in an exact sense.

Global descriptions of shape usually suffer from sensitivity to occlusions. We point out two methods of comparing global shapes with occlusions: one based on a grammar; the other based on Le's inequality on euclidean shapes.

The properties of all of the above algorithms are proved rigorously.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university, and to the best of the author's knowledge and belief, the thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis. The author consents to the thesis being made available for photocopying and loan if applicable if accepted for the award of the degree.

Parts of the work presented in this thesis have previously been accepted as papers in international journals and conferences:

[1] B. Bhavnagri, *A method for representing shape based on an equivalence relation on polygons*, Pattern Recognition **27** (1994), no. 2, 247–260.

[2] B. Bhavnagri, *Models for recognition and correspondence matching of objects*, Proceedings of the 7th Australian Joint Conference on Artificial Intelligence (C. Zhang, J. Debenham, and D. Lukose, eds.), World Scientific, 1994, pp. 536–543.

[3] B. Bhavnagri, *Construction of a markov process on $\cup_k \Sigma_2^k$ to model a process arising in vision*, Proceedings of Current Issues in Statistical Shape Analysis, 1995, pp. 76–81.

[4] B. Bhavnagri, *Connected components of the space of simple closed non-degenerate polygons*, Current Issues in Statistical Shape Analysis, 1995, pp. 187–188.

[5] H. Le, B. Bhavnagri, *On simplifying shapes by subjecting them to collinearity constraints*, Mathematical Proceedings of the Cambridge Philosophical Society, To appear 1997.

The author's technical reports also overlap with this thesis:

[6] B. Bhavnagri, *Object recognition, geometric invariance and shape spaces*, Tech. Report TR94-08, Dept. Computer Science, University of Adelaide, May 1994.

[7] B. Bhavnagri, *Connected components of the space of simple non-degenerate polygons*, Tech. Report TR95-07, Dept. Computer Science, University of Adelaide, 1995.

[8] B. Bhavnagri, *Proofs and corrections to construction of markov process on shape spaces*, Tech. Report TR95-06, Dept. Computer Science, University of Adelaide, 1995.

[9] B. Bhavnagri, *Proofs for construction of markov process on shape spaces*, Tech. Report TR95-04, Dept. Computer Science, University of Adelaide, 1995.

Acknowledgements

I would like to thank my supervisors Alan Carey and Michael Brooks for lending me a patient ear, and providing me valuable feedback on drafts of this thesis, as well as my papers and reports. I am also especially indebted to Huiling Le for hosting my visit to the University of Nottingham, and our numerous very stimulating discussions about shape spaces. I also thank Michael Murray, Garry Newsam, Wojtek Chojnacki and Du Huynh for their encouragement, suggestions and constructive criticism on many occassions. My visit to the university of Leeds was hosted by John Kent, Kanti Mardia and Ian Dryden, with whom I enjoyed some discussions. Sylvan Elhay provided me some useful advice on numerical analysis. I am also grateful to Steve Maybank, Gunnar Sparr and many others who sent me their papers.

Francis Vaughan of the SA Center for Parallel Computing gave me access to and help using a wonderful super-computer. Many of the experiments were carried out on equipment from the cooperative research Center for Sensor Signal and Information Processing. I would also like to thank Bob Fang for constructing some experimental apparatus for me. I am grateful to Mark Nitzberg and Peter Kovesi for allowing me to use their programs.

I am grateful to the University of Adelaide and CSSIP for funding part of this research, and my conference travel.

Chapter 1

Introduction

Vision is arguably our most important sense. Most of us take it for granted all the time, whether we are looking at scenery, watching TV, reading, or doing almost anything else. Most of our perceptions of the world are formed through the sense of sight, mostly informing us, sometimes misleading us. It is equally important to other sighted animals, to find food, to sense predators and prey, and to survive in their environment. We therefore have an intuitive, even innate sense of what vision is. We are naturally curious as to how this often mysterious sense works.

Our understanding of vision is founded on optics, the anatomical structure of the eye, the physiology and anatomy of the centers in the brain which are responsible for vision, and the psychology of vision. Eyes detect light reflected from objects; after diffracting through an aperture; and refracting through a lens as well as internal fluids and structures. Accurate models of physical processes underlying vision such as reflection, refraction and diffraction were discovered by such pioneers as Newton and Young. The foundations for analysing the structure and function of the eye through optics were laid by Helmholtz in his treatise on physiological optics. A fundamental psychological observation about the nature of vision was made by Wertheimer, who noticed the apparent motion not of individual dots, but of fields in images presented sequentially as a movie [182]. This started the Gestalt school of psychology [183], [98]. Physiologists such as Adrian recorded the minute voltage changes accompanying the transmission of nerve signals. Their experiments led to the view that nerve fibres from the peripheral nervous system gave rise to a simple mapping in the brain of physical events at the body surface [1]. This ultimately led to a famous series of discoveries of "receptive fields" in the retina by Hartline [79], [80], Barlow [7] and Kuffler [104]. This was followed by the even more remarkable discovery of receptive fields in the visual cortex of the brain by Hubel and Wiesel [91], [92], [93], [94], which ultimately won a Nobel prize.

These discoveries led to the view that vision is a process that somehow constructs a representation of the scene from signals emanating from the retina (of both eyes). Marr argued that this process is computational [120]; primarily an information processing task, but also an information representing task.

Computer vision is the subject concerned with developing computational mechanisms

of vision for machines. Video cameras serve the role of eyes, and computers the role of a brain. Machine vision systems are applied in robotic, medical, quality control, automotive and other technologies. Computer vision also aims to further our knowledge of biological vision by studying how visual systems might work. Since biological vision analyses specific visual systems, a more general body of knowledge provides it with valuable models and concepts. Likewise, branches of knowledge such as physiology, psychophysics and computational biology provide computer vision with valuable experimental knowledge of working, functional visual systems.

The general approach of computer vision is that of artificial intelligence, so it helps to compare it with say the chess playing computer [138]. The chess playing system consists of a large set of possible states, rules governing the possible transitions between states, and a goal state. The system attempts to reach its goal by minimizing a cost functional at each step. The total number of possible games is immense, and the computational complexity of finding an exact solution prohibitive. So the chess playing system introduces heuristics into the system. The rules of the game, and therefore the algorithm employed are not laws of nature, fixed and pre-determined. New heuristics can be invented at will, and are always legitimate if they produce results.

The rules or constraints in computer vision are largely obtained from physical laws. Vision does not exist in isolation from the physical world. Eyes detect light reflected from objects, after diffracting through an aperture, focusing through a lens and so on. The behaviour of light in the eyes or a camera must obey the laws of optics. Moving objects must obey kinematic principles. Deforming objects must conform to elastic models, and so forth. Other ad hoc constraints are invented because it is not yet known how to build a system without using them. This type of constraint will be regarded as a potential source of unreliability. Such extra constraints can contradict the more important physical assumptions in some subtle and unanticipated way. Another important limitation arises in connection with using search algorithms to solve vision problems. The search algorithms are almost always exponential in complexity, and only find local optima. We will strive to replace the search with global closed form solutions wherever possible. It usually transpires that the global closed form solution has a low order polynomial complexity. In cases where floating point errors are deemed to cause too much unreliability, it becomes possible to employ exact arithmetic in place of floating point arithmetic.

This thesis is based on assumptions pertaining to the physical structure of a camera and the scattering of light from visible surfaces; there are *no other assumptions.* The mathematical formulation is based on topology and differential geometry, and is founded on the notion of a shape space, first propounded by Kendall. Rigorous proofs are provided of almost every statement made, taking great care to avoid invoking ad hoc assumptions. The author has assumed the reader has a knowledge of algorithms and data structures [38], linear algebra [170] and calculus. The concepts of topology and differential geometry being used will be introduced as needed. Non-mathematicians often find these two branches of mathematics difficult to grasp. The author has endeavoured to make the mathematics well-motivated, self-contained and as easy for the reader to follow as possible. Abstract definitions are supplanted with numerous examples. Some long proofs are omitted however, with references to where

they may be found in the mathematical literature. The author hopes the reader who is not interested in reading abstract texts and papers on topology and differential geometry will treat the proofs (that have not been reproduced) as reasonable assumptions, which could be proved given enough time and effort, and have been verified by other authors. Going to such lengths to be precise may seem an extreme case of being pedantic. In the author's opinion it is justified by the need for reliability in engineering, and the need for a sound base upon which to build further research.

One very important question we can ask about vision is to give a definition of vision. Computer vision approaches vision in a different way. It starts with the functions served by the visual system, provides an algorithm for each function, and states theories that provide some justification for the algorithm. For example, a moving animal or robot needs to detect obstacles and gauge their distance, to avoid collisions. For this it needs three dimensional perception or representation of the scene. An animal also needs to recognise objects such as food, prey and predators. Robots need a similar object recognition capability. The theory of stereoscopic vision is that two eyes can produce three dimensional vision. Stereo is one of the major functions served by vision, and will be the subject of chapters 2 to 9. Recognition is another major function served by vision; it will be the subject of chapter 10. Chapter 11 studies the problem of representation in a more abstract sense, based on applying the theory in chapters 2 to 9 to interpret physiological experiments.

As Wertheimer observed, visual perception is somehow associated with vector fields. The Gestalt school, most notably Kanizsa, presented compelling arguments that visual perception is holistic in nature [98]. This means that what is perceived is a property of whole shapes, rather than a property of parts of shapes. This is a view that will be mathematically formulated in this thesis by employing global differential geometry. This formulation ultimately culminates in an explanation of a puzzling aspect of the receptive field experiments in chapter 11: why are the receptive fields organised into orientation dominant columns? Why do we think the problem is global? Suppose we assume that the surface of an object is visible in its entirety, is smooth, and the intensity of light reflected from it is smooth. Then we can deduce from the camera model a strong, global property of vector fields in such an image. An algorithm that detects the opposite property must therefore detect occlusions, intensity discontinuities, discontinuities in derivatives of intensity, surface discontinuities and discontinuities in derivatives of the surface. Although we know that it is impossible to see all of an object from every side, to reject a hypothesis of no occlusions and no discontinuities we must deduce the strongest possible consequences of such a hypothesis. In chapter 3 we will deduce from this theory that images have a hidden graph structure which is not consciously perceived by human beings. We present an algorithm with linear time and space complexity to generate a collection of feature points with attributes in cyclically ordered groups. The conditions under which the attributes are invariant to perspective, focus and other camera parameters are discussed in chapter 3.

One of the main approaches to reconstructing the third dimension given two images is via stereo correspondence; the pairs of image points that are images of the same scene point are first identified. This is a formidable problem known as the stereo correspondence problem. It is the subject of intensive and ongoing research in computer

vision, and is the subject of many books like [189], [54], [96], [124], [121], [70], [71], [5] and [87]. The algorithms proposed in the computer vision literature can diverge from the correct solution or provide output so unreliable that reconstructions are in gross error [189]. We will see that these problems are fundamental deficiencies of their formulation of the camera model, and provide a new formulation of the camera model in chapter 2.

After the data structure representing features has been computed from an image, two such images can be compared. For each pair of groups in the two images, a hypothetical pairing of corresponding points can be computed in time complexity $O(mn \log n)$ by a cyclic longest common subsequence algorithm. Such algorithms are presented in chapter 8, after we study the deeper problem of rejecting invalid hypotheses in chapters 4 to 7. The time complexity of this algorithm is thought to be sub-optimal.

Owing to the well-known instability of scene reconstruction algorithms based on stereo [52], many researchers' attention shifted to a probabilistic formulation. However, their formulation is generally based on a distribution which has no justification. There is in fact a possible justification for probability distributions in vision. When light passes through an aperture, it gives rise to a diffraction pattern. More generally, any optical system has a point spread function which describes the distribution of light produced by a point source. If we exchange the sensor with a light source, the optical paths are the same, and the point spread function is a distribution of probabilities that points in each position produced a response from a given sensor. However, there is a severe complication: light is reflected from surfaces, and most collections of points in the scene drawn randomly from such a distribution will form bizarre self-intersecting shapes, not surfaces. It follows that the actual distributions of shapes will be very different from the joint distributions of points (see chapter 10).

One of the most influential and important ideas in computer vision arises from a question posed by the psychologist Gibson: "How does one obtain constant perceptions in everyday life on the basis of continually changing sensations?" [62]. He considered that the problem of perception was mainly constituted by the recovery of invariants to motion of the observer and changes in the stimulus. An approach to computer vision based on computing geometric invariants was developed [132] based on projective and affine geometry. There is however a major non-existence result to contend with: there are no general single view invariants [29]. It is therefore necessary to attack the stereo correspondence problem in order to calculate invariants of three dimensional shape. This is tantamount to the knowledge of a function of hypothesised corresponding points that is zero precisely when there exists camera parameters and a scene such that each image is the image of the same scene. Such a function is furnished by the theory of epipolar geometry due to Longuet-Higgins [113]. This was actually a rediscovery of a theory developed by the mathematicians Chasles, Hesse and Sturm almost a century earlier (see [121] for a survey).

There are however two crucial problems which have been overlooked in the (mathematical formulation of the) invariance and epipolar geometry paradigm: bias to viewpoint and stability (even under error free conditions). The reason for calculating invariants

of group actions is that there are many hidden and unknown variables which cannot be directly measured. When inconsistent hypotheses are compared, if the comparison is in any way dependent on these hidden parameters, it will be biased because the parameters are implicitly incorrect. Furthermore, there must be some relation between the value of the comparison function and the Euclidean metric on the scene, otherwise the resulting scene reconstruction will be highly unstable. Unfortunately, invariants to group actions do not themselves provide such a relation, we need recourse to the very abstract notion of a quotient space. These notions are formulated precisely in chapters 4, 5 and 8. These chapters are concerned with the question: does a function satisfying these convergence and stability conditions exist in the epipolar geometry model? A new formulation of the perspective camera model in which the unchanging shape of the retina is incorporated into a group theoretic model is presented. The same existence question is answered in this model also. Such existence problems are properties of quotient spaces, not of invariants to group actions. Thus they cannot be treated adequately in a more naive way than the theory of shape spaces. There is a limiting case and a special case in which an algorithm to calculate such a function is given explicitly.

Chapters 6 and 7 adopt a second approach to the correspondence hypothesis rejection problem. This is based on the author's discovery of a language whose strings are geometric invariants [11]. A grammar which describes changes to strings associated with occlusions (for example) was also obtained. One advantage of this approach over an error minimizing approach is its speed: strings can be compared much faster than algorithms requiring floating point arithmetic. Another advantage is greater stability and freedom from bias. It does not eliminate the error minimizing approach because many distinct shapes have the same strings.

Chapter 9 studies the scene reconstruction problem. Two images of a scene do not themselves determine a metric reconstruction. This thesis presents a non-iterative algorithm that can use the rotation between two cameras to produce an exact reconstruction. The so called principal points of the two cameras must also be known to produce a metric reconstruction. This algorithm must be given data that are in correspondence in an exact sense to work, and it is not known how to produce satisfactory input data. Part of the algorithm is due to Sparr. It has been known for some time that the human vestibular apparatus (which provides our sense of balance) is linked to vision. The sense of balance, together with feedback from ocular muscles can provide enough information to an animal's visual system to build a three dimensional representation of the scene. The sense of balance together with feedback from ocular muscles provides information on the rotation between an eye at one point in time and the same eye at another point in time. The rotation between two different eyes at the same point in time only requires ocular muscle feedback.

A moving eye or scene can also furnish three dimensional vision. This is a case where computer vision has attained relatively greater success. The books [157], [36], [121], [70], [71] and [134] are good surveys. It is not the subject of this thesis, except that it can be viewed as a special case of stereo. In this thesis, we assume the camera parameters are completely general unless otherwise stated. There is no restriction on focal length, angle between cameras, distances between corresponding points or the

like.

Another means of scene reconstruction is via machine interpretation of line drawings. This subject assumes a polyhedral scene with trihedral vertices, and proceeds to label junctions in an image. The collections of labellings that correspond to an image of such a polyhedron turns out to be quite small, so that a combinatorial analysis becomes feasible. A good survey of the theory of line drawings can be found in [171]. A line drawing analysis of shadows can be found in [156].

There are also algorithms to recover three dimensional shape from the intensity variations in a single image, called shape from shading; See [89] for a survey. Shape from shading is thought to be a significant component of the human ability to recognise faces. However an assumption of smooth shading underlies most such algorithms; chapter 3 disputes this assumption. We do however make the assumption that scene surfaces are approximately Lambertian in order to compute corresponding points (in chapter 8).

The scene reconstruction algorithms considered in this thesis produce a finite set of points. Only part of the surface of an object will be visible, so we need an algorithm capable of answering the query: is this three dimensional reconstruction a part of a scaled, rotated and translated model? This is an example of a type of problem involving the comparison of shapes in two different Euclidean shape spaces. Chapter 10 discusses an inequality due to Le which plays a central role in such problems. It is conjectured that there is a quadratic time algorithm to solve this problem.

In chapter 11 we present some evidence that the mathematical models presented in chapters 2 to 10 could be models of biological vision. An excellent review of the types of experiments that have been performed in psychology, anatomy and physiology to study awareness in human and animal vision can be found in the book by Crick [39]. We summarize some of the general attributes of awareness, and the types of experiments that have been performed to study awareness outlined in [39]. It seems reasonable that we should define vision as a process that begins with sensory measurements of light, and produces a representation of the scene. We put forward a tentative hypothesis as to what type of representations we are conscious of or aware of in chapter 11. This hypothesis led to the specific representations adopted in this thesis, and thereby to the specific mechanisms of vision discussed in chapters 2 to 10. It therefore seems possible to test theories about awareness by whether they are consistent with experiments on the detailed mechanisms of vision.

The model proposed in this thesis is by no means a complete model of all the physical phenomena on which vision could be based. There are many interesting and useful schools of thought in computer vision which will not be covered in this thesis. Anything which would disrupt the logical unity and cohesion of the thesis has been excluded. The author hopes that researchers who have not been cited here do not feel snubbed.

The author implemented as many as possible of the algorithms proposed here. They were tested on real images, which were not selectively reproduced just to give an unscientific demonstration that the programs work. Nonetheless, the behaviour of the programs was found to be in accord with the theory.

Chapter 2

Camera models

This chapter introduces the principles underlying the formation of an image in a camera. Using the properties of multi-element lenses and apertures from Born and Wolf [23], it introduces the perspective camera model. It then introduces the projective version of the perspective camera model, found in Faugeras [54], and various special cases of this model. The projective perspective camera model is based on a four dimensional representation of three dimensional space, where points in three dimensional space are considered to be synonymous with lines through the origin in four dimensions. The author introduces another four dimensional representation of three dimensional space, and a novel version of the perspective camera model.

2.1 Properties of lenses and apertures

Three main components of a camera or the eye are a lens, an aperture or iris, and a collection of light sensors on a surface, called the retina, image plane or film plane. The term retina is used to describe this surface in the eye, whereas the terms image plane or film plane are usually used to describe this surface in a camera. The retina of a camera is usually planar, and will be referred to as the retinal plane. The camera forms an image of the scene in front of it.

2.1.1 Thin lens and its limitations

The simplest lens system is a lens with two surfaces rotationally symmetric about the same axis, called the optical axis, whose thickness may be neglected. Such a lens is called a thin lens. Rays that pass through the center of a thin lens pass through undeviated (see figure 2.1).

If the retinal surface is a plane perpendicular to the optical axis, then there is a plane in front of the lens whose image is perfectly focussed, provided the angle to the optical axis is small (paraxial assumption). Points in front of the lens that are not in the plane whose image is perfectly focussed produce an image that is a spot instead of a point (see figure 2.2). We can bring the image into focus by moving the retinal surface

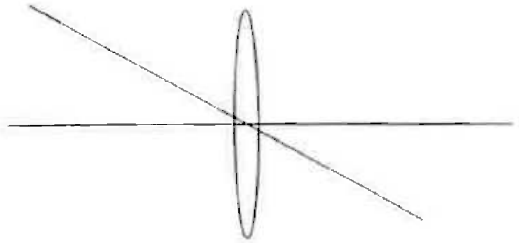


Figure 2.1: Rays passing through center of thin lens pass undeviated

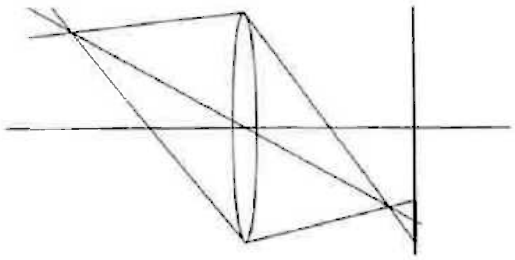


Figure 2.2: Image of imperfectly focussed point is a spot

relative to the lens. However, rays from points in front of the lens at sharp angles to the optical axis will not converge to a single point at all. This problem is known as off-axis aberration. Also, rays of different wavelengths will not form an image at the same point, known as chromatic aberration. These aberrations are minimised by more complex lens systems, which consist of multiple surfaces of revolution about a common axis.

2.1.2 Multi-element and zoom lenses

A lens with multiple surfaces of revolution with a common axis is called a multi-element lens. Zoom lenses are more complex multi-element lens systems whose focal lengths can be altered. Most commercially available cameras use multi-element or zoom lenses. The most general optical system we can consider in this thesis consists of multiple surfaces of revolution with a common axis, called the optical axis.

Suppose the image of a point in front of the camera (object point) is a single point. Then from symmetry of the optical system, the image point lies in the plane that contains this object point and the optical axis. Thus we can reduce our considerations of how light rays pass through a lens system from three dimensions to a plane.

A multi-element lens system has two focal planes: object focal plane and image focal plane. Rays that are parallel in the object space (left hand side of figure 2.3) will be transformed into rays which intersect in a point called the image side focal point (see figure 2.3). Similarly, rays from a point on the object focal plane will transform into rays parallel on the image side. In the special case of a thin lens, the object and image focal planes are equidistant from the center of the lens.

We write the object point coordinates (Y, Z) relative to the object side focal point F;

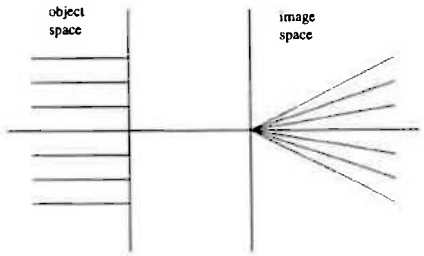


Figure 2.3: Parallel rays in front of lens converge at image side focal point

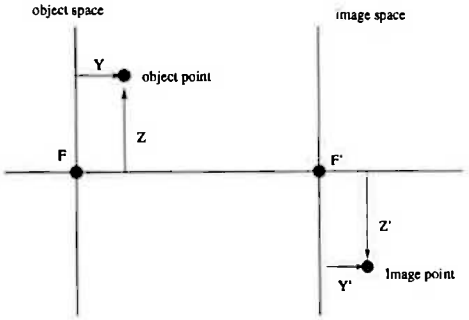


Figure 2.4: Object and image point coordinates in lens equations

we denote the image point coordinates (Y', Z') relative to the image focal point F'. These object and image side coordinate systems are illustrated in figure 2.4.

The optical system has two focal lengths f, f'. The object side focal length is denoted f and the image side focal length is denoted by f'. $Z = f$ and $Z' = f'$ are planes of unit (lateral) magnification. The object ray enters the plane $Z = f$, and the image ray emerges from the plane $Z' = f'$ at the same level. Figure 2.5 illustrates the behaviour of a ray entering and leaving a multi-element lens. The refractive index of the object side is denoted by n, and the refractive index of the image side is denoted by n'.

The lens equations are

$$\frac{Y}{Y'} = \frac{f}{Z} = \frac{Z'}{f'}$$

$$\frac{\tan(\gamma')}{\tan(\gamma)} = -\frac{Z}{f'}$$

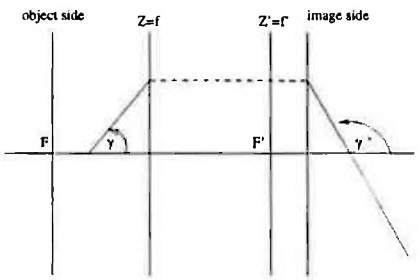


Figure 2.5: Ray entering multi-element lens from left, exiting from right

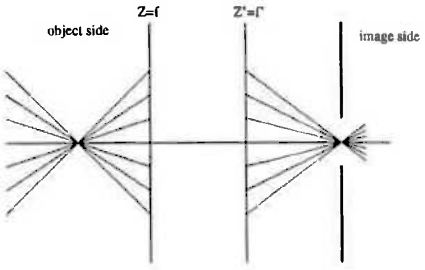


Figure 2.6: Lens and aperture only allow pencil of rays in front of camera to reach retina

$$\frac{f'}{f} = -\frac{n'}{n}$$

From the first lens equation we deduce Newton's equation $ZZ' = ff'$. So only the surface $Z = \frac{ff'}{Z'}$ is brought into perfect focus on the retina (Y', Z').

2.1.3 Apertures

An aperture is also known as a pupil or an iris. It is often used within a lens system to limit the angle of rays. Thus they limit off-axis aberration. However, small apertures also create diffraction effects, and limit the amount of light entering the sensors.

Lemma 2.1
Suppose there is an aperture on the image side, centred on the optical axis. Consider a pencil of rays from the retinal plane to the center of the aperture. We will show that the corresponding object rays also form a pencil with central point on the optical axis. See figure 2.6.

Proof

Suppose t' is the distance between the aperture and the image side focal plane. Consider a ray passing through the aperture at angle γ', intersecting the object side focal plane at height h. Then the ray enters the optical system at some angle γ, intersects the optical axis at some distance t from the object side focal plane, and intersects the object side focal plane at height h. We will show that t is a constant. Figure 2.7 illustrates the notation.

From trigonometry,

$$\tan(\pi - \gamma') = h/t'$$
$$\tan(\gamma) = h/t$$

Eliminating h from these equations we obtain

$$\tan(\pi - \gamma') = \tan(\gamma)t/t'$$

From the second lens equation

$$\tan(\pi - \gamma') = -\frac{Z't}{ft'}\tan(\gamma')$$

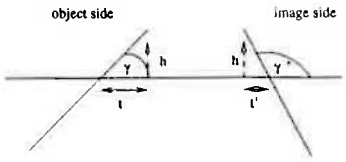


Figure 2.7: Notation used to derive perspective model

which implies

$$t = \frac{f}{Z'}t'$$

Thus t is a constant for any fixed focal length or retinal position. ∎

The lemma also shows that as the camera zooms or refocusses, the center of this pencil of rays in front of the camera can move along the optical axis. The centre of this pencil of rays is called the optical centre of the camera. From figure 2.6, the reader can see that the pencil of rays on the image side only has the effect of a constant scale factor.

Thus the optical system of such a camera can be considered equivalent to the following, simplified model, called the perspective camera model. The only rays that can reach the retina pass through the optical centre of the camera, in a straight line; the image only differs from this by a constant scale factor. The optical centre can move along the optical axis as the camera is refocussed or zoomed. Thus at one point in time the optical centre is at a fixed position, but it may be at different positions at different times.

In reality, the perspective model is an approximation to a real lens system, which can fail under more extreme conditions. If the angle from which an object or scene is viewed is too large, the paraxial assumption is violated, in which case the perspective camera model does not hold. If significant aberrations are present, there may be radial distortion which can however be corrected [187], or tangential distortion which can be ignored in all but the most exacting applications [187].

2.2 Projective version of perspective camera model

Figure 2.8 illustrates the perspective camera model, without the scale factor. The previous section used geometric optics to explain the conditions under which such a model holds.

The perpendicular distance between the optical centre and the retinal plane is called the focal length or the principal distance. The line passing through the optical centre which is perpendicular to the retinal plane is called the optical axis. The point where the optical axis intersects the retinal plane is called the principal point. The plane parallel to the retinal plane which passes through the optical centre is called the focal plane.

The points in an image correspond to light sensors in the retina of the camera which took that image. The camera sends data to a computer where the image is represented

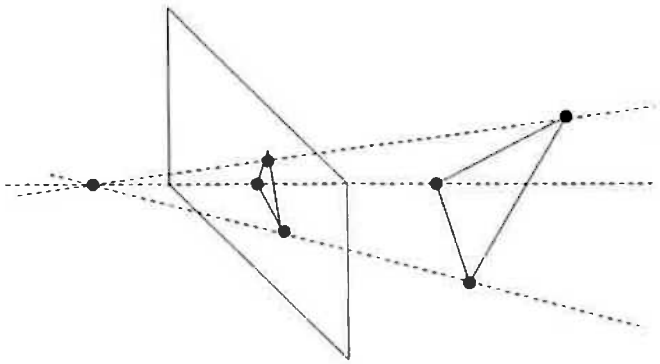
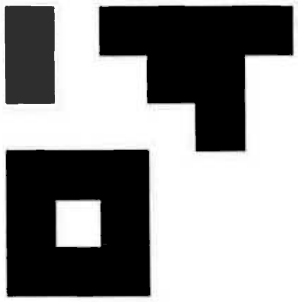


Figure 2.8: Perspective camera model



1	0	1	1	1	1
1	0	0	1	1	0
0	0	0	0	1	0
1	1	1	0	0	0
1	0	1	0	0	0
1	1	1	0	0	0

(a) (b)

Figure 2.9: (a) A binary image (b) the bitmap of that image

as an array whose elements contain the measurements of light sensors. Each element of this array is called a pixel, which is an abbreviation for picture element. Pixels can have attributes such as coordinates, brightness and colour. If there are only two brightness attributes (black and white) at each point and no colour attributes, the image is called a binary image. A binary image can be represented by a bitmap (see figure 2.9). If there is a range of brightness attributes but no colour attributes, the image is called a greyscale image. A greyscale image can be represented by a greymap, which is an array of integers whose values are measurements of light intensity. There is a large number of file formats in use for images and computer graphics (see [135]). The author used the public domain package pbmplus to convert files from one format to another.

In this thesis the coordinates of a scene point W will usually be represented by a column vector. In homogeneous coordinates it has four components; any non-zero scalar multiple denotes the same point. Two non-zero four tuples are regarded as equivalent if one is a non-zero scalar multiple of the other. This is an example of an equivalence relation: A relation denoted $\sim$ on a set is called an equivalence relation if for all x, y, z in that set

(i) $x \sim x$ (reflexive)
(ii) $x \sim y \Rightarrow y \sim x$ (symmetric)
(iii) $x \sim y$ and $y \sim z \Rightarrow x \sim z$ (transitive)

The set of all elements equivalent to a given element is called an equivalence class.

Example 2.1
In a homogeneous scene coordinate system, the point with euclidean coordinates $(1,2,3)$ can be represented by $[1 \quad 2 \quad 3 \quad 1]^\top$ or $[2 \quad 4 \quad 6 \quad 2]^\top$. All the representations of this point lie on a line through the origin, excluding the origin itself, each of which is an equivalence class of points in $\mathbf{R}^4 \setminus \{0\}$. The set of such equivalence classes is three dimensional projective space. □

Example 2.2
In a homogeneous image coordinate system, the point with euclidean coordinates $(1,2)$ can be represented by $[1 \quad 2 \quad 1]^\top$ or $[3 \quad 6 \quad 3]^\top$. All the representations of this point lie on a line through the origin, excluding the origin itself, each of which is an equivalence class of points in $\mathbf{R}^3 \setminus \{0\}$. The set of such equivalence classes is two dimensional projective space. □

Definition 2.1
$\mathbf{P}^n$ denotes n dimensional projective space.

The book by Faugeras [54] gives some introduction to projective geometry, motivated by computer vision. See also [155] for more details about projective geometry.

Whereas in euclidean coordinates the transformation from scene coordinates to image coordinates is non-linear, in homogeneous coordinates it is linear; we introduce the perspective camera matrix from [54] to represent this linear transformation:

$$\mathbf{P} = \begin{bmatrix} \alpha_u R_{11} + u_0 R_{31} & \alpha_u R_{12} + u_0 R_{32} & \alpha_u R_{13} + u_0 R_{33} & \alpha_u t_x + u_0 t_z \\ \alpha_v R_{21} + v_0 R_{31} & \alpha_v R_{22} + v_0 R_{32} & \alpha_v R_{23} + v_0 R_{33} & \alpha_v t_y + v_0 t_z \\ R_{31} & R_{32} & R_{33} & t_z \end{bmatrix}$$

where $\mathbf{R}$ is a rotation matrix (which is a 3×3 orthogonal matrix with determinant one), $\mathbf{t} = (t_x, t_y, t_z)$ is a translation, u_0, v_0 are the retinal coordinates of the principal point, α_u is the size of the focal length in horizontal pixels, and α_v is the size of the focal length in vertical pixels. u_0, v_0 allow for the fact that the origin of the image coordinates may be different from the principal point; the origin is usually the top left hand corner of the image, whereas the principal point is usually near the centre of the image. The rotation and translation describe the orientation and position of the camera in space. α_v/α_u will be called the aspect ratio of the camera.

Example 2.3
Here is an example of a camera matrix:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The optical centre is at the origin, so $(t_x, t_y, t_z) = \mathbf{0}$. $\mathbf{R}$ is the 3×3 identity matrix, so its optical axis is the z-axis. $u_0 = 0$ and $v_0 = 0$ so the origin of image coordinates is the same as the principal point. $\alpha_u = 1 = \alpha_v$, so the physical distance between horizontal and vertical pixels is the same, and the aspect ratio is 1. This can be regarded as a canonical example of a camera. □

In the projective version of the perspective camera model, any non-zero scalar multiple of $\mathbf{P}$ must be considered equivalent to $\mathbf{P}$ because any non-zero scalar multiple of the

column vectors it acts on represents the same point. Thus $\mathbf{P}$ is only defined up to a non-zero scale factor. The reader may think that this means that points on opposite sides of the focal plane are considered equivalent, but example 2.4 shows that this is not the case.

Example 2.4
Let $\mathbf{P}$ be as in example 2.3, and consider the points $(1,1,1)$ and $(-1,-1,-1)$ which are on opposite sides of the focal plane of $\mathbf{P}$. The first point is represented in $\mathbf{P}^3$ by $[1 \quad 1 \quad 1 \quad 1]$, whereas the second point is represented in $\mathbf{P}^3$ by $[1 \quad 1 \quad 1 \quad -1]$. These are two different points of $\mathbf{P}^3$. □

The camera matrix $\mathbf{P}$ is a linear map from $\mathbf{P}^3$ to $\mathbf{P}^2$.

This projective version of the perspective camera model does not prescribe the position of the retinal plane. In fact the retina need not even be a plane; we could choose a spherical retina [124] since any point (x,y,z) is equivalent to $(x,y,z)/\sqrt{x^2+y^2+z^2}$ which lies on the unit sphere. The projective model also does not specify whether the retina is in front of or behind the optical centre, or whether the image is inverted or not. We can specify a retinal plane by choosing a particular coordinate system for the projective version of the retinal plane. Suppose $\mathbf{y} = \mathbf{P}\mathbf{x}$ is mapped to $[y_1/y_3 \quad y_2/y_3 \quad 1]^\top$, then $y_3 = 1$ is the retinal plane; $y_3 = 0$ is called a point at infinity on the retinal plane. The equation of the retinal plane would be $R_{31}x_1 + R_{32}x_2 + R_{33}x_3 + t_z - 1 = 0$. The equation of the focal plane would be $R_{31}x_1 + R_{32}x_2 + R_{33}x_3 + t_z = 0$. The focal plane *is determined* by $\mathbf{P}$, because its homogeneous representation is the third row of $\mathbf{P}$.

Given $\mathbf{y}$ in homogeneous retinal coordinates, produced by a camera matrix, the image coordinates will be presumed to be $[y_1/y_3 \quad y_2/y_3 \quad 1]^\top$. A choice of retinal plane $y_3 = 1$ is implicit in this.

The optical centre $\mathbf{C}$ of a camera with camera matrix $\mathbf{P}$ is given by the solution to $\mathbf{P}\mathbf{C} = \mathbf{0}$. Provided $\mathbf{C}$ is not a point at infinity,

$$\mathbf{C} = -\begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}^{-1} \begin{bmatrix} P_{41} \\ P_{42} \\ P_{43} \end{bmatrix}$$

A perspective camera maps the points on its focal plane to points at infinity on the retinal plane.

Example 2.5
Let $\mathbf{P}$ be the camera matrix introduced in example 2.3. Points on the focal plane have the form $[x \quad y \quad 0 \quad 1]^\top$ and $\mathbf{P}\mathbf{x} = [x \quad y \quad 0]^\top$, which is a point at infinity on the retinal plane. It is called a point at infinity because as z (depth) approaches zero, x/z and y/z approach positive and negative infinity. □

The process of estimating $\mathbf{P}$ from a fixed reference object is known as camera calibration. In reality, the parameters of a camera rarely remain fixed, due to problems such as thermal variations, vibration and changes in focal length. We will develop and use methods that do not assume that highly variable camera parameters are fixed. The reader can consult many books such as [54] or [71] for methods of camera calibration.

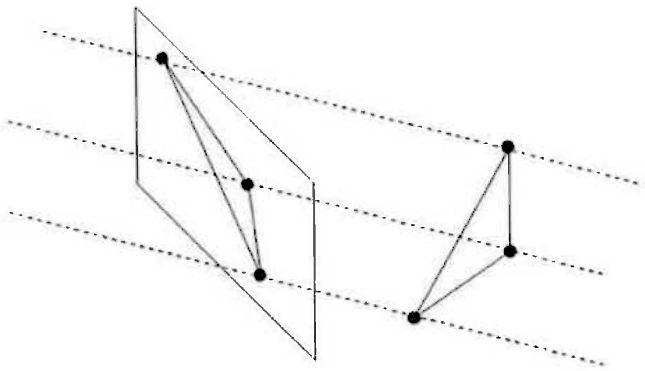


Figure 2.10: Affine camera model

2.3 Special cases of the perspective camera model

There are various special cases of the perspective camera model mentioned in the literature. The most important one is the affine camera model, due to Koenderink and Van Doorn [102]. In the affine camera model, the points in the scene are projected onto the retina along parallel lines (see figure 2.10) in a fixed direction.

Example 2.6
When the retina is the x-y plane (or $z = 0$ plane) and points in the scene are projected onto the retina along the z-axis

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

since it maps $(x, y, z, 1)$ to $(x, y, 1)$. □

The optical centre of an affine camera is a point at infinity. In this example the optical centre is $[0 \quad 0 \quad 1 \quad 0]^\top$. The image of points on the focal plane comprises points at infinity on the retinal plane. In the example, the points on the focal plane have the form $[x \quad y \quad z \quad 0]^\top$ whose image is $[x \quad y \quad 0]^\top$ which is a point at infinity.

Affine camera matrices take the form

$$\begin{bmatrix} G'_{11} & G'_{12} & t'_x \\ G'_{21} & G'_{22} & t'_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} G_{11} & G_{12} & G_{13} & t_x \\ G_{21} & G_{22} & G_{23} & t_y \\ G_{31} & G_{32} & G_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\mathbf{G}'$ is a non-singular 2×2 matrix, $\mathbf{t}'$ is a translation of the retinal plane, $\mathbf{G}$ is a non-singular 3×3 matrix and $\mathbf{t}$ is a translation of the camera in space.

A matrix of the form

$$\begin{bmatrix} \mathbf{G} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$$

where $\det(G) \neq 0$ is called an affine transformation.

The affine camera model is a good approximation to the perspective camera model in certain cases. Namely when an image is formed of a small and distant object (in relation to the size of the camera) as in aerial or satellite photographs. However, if the object is relatively large, such as a mountain, it is *not* a reasonable approximation.

In the special case where the projection is perpendicular to the retinal plane, we have an orthographic camera model. When a scale factor is allowed as well, we have a weak perspective camera model. And when the projection is parallel but in a fixed and known direction, we have a para-perspective camera model [2].

2.4 A new version of the perspective camera model

The perspective camera model of Faugeras is based on a four dimensional representation of space, which is the usual homogeneous representation of the projective space $\mathbf{P}^3$. For reasons that will be clarified in this section, our camera model differs from this model in the following respects.

- The representation of points in three dimensional space is still four dimensional, but scalar multiples do not represent the same point.
- We represent a point in three dimensional space by simply adjoining 1 as its fourth coordinate; we do not use homogeneous scene coordinates.
- Points in the focal plane of a camera are excluded from consideration; points at infinity on the image plane are ignored.
- The representation of image points is not three dimensional, but four dimensional; scalar multiples do not represent the same point.
- The formation of an image is represented by a 4×4 camera matrix g and a non-linear mapping ι to be introduced below; we do not use homogeneous image coordinates.
- The first two components of the four components representing image points are the actually observed image coordinates.

It will turn out that this model and the projective model have different properties (see lemma 4.5 and theorem 5.18 for instance).

If (x, y, z) is a point in $\mathbf{R}^3$, we write it as $(x, y, z, 1)$. Scalar multiples of $(x, y, z, 1)$ do not represent the same point.

Example 2.7
The perspective camera represented by $\mathbf{P}$ from example 2.3 is also represented by the map

$$\iota : (x, y, z, t) \mapsto (x/z, y/z, 1, t/z) \quad z \neq 0$$

Any point with $z = 0$ is on the focal plane. Throughout this thesis, no point on the focal plane has an image on the retina. □

If a function is evaluated at a point in its domain it has only one possible value. But if the domain is itself a set of equivalence classes, then the function is said to be well-defined if it is constant on each equivalence class. If the range is also a set of equivalence classes then the function is said to be well-defined if its values on each equivalence class in the domain are equivalent in the range.

Example 2.8
$x_1 - 1, x_2 - 2, x_3$ is not well-defined on $\mathbf{P}^2$ since evaluating it at $(1,1,1)$ gives $(0,-1,1)$ whereas evaluating it at $(2,2,2)$ gives $(1,0,2)$. $x_1^2 + x_2^2 + x_3^2$ is well-defined as a function mapping $\mathbf{P}^2$ to $\mathbf{P}^0$ since, for any non-zero λ, we have that $\lambda^2(x_1^2 + x_2^2 + x_3^2)$ and $x_1^2 + x_2^2 + x_3^2$ are equivalent. □

Example 2.9
ι does not represent the forming of an image in homogeneous coordinates of $\mathbf{P}^3$, even though it is well-defined in that coordinate system. Evaluating the images of two points $(1,1,1,1)$ and $(-1,-1,-1,1)$ in Faugeras' camera model

$$P[1 \quad 1 \quad 1 \quad 1]^\top = [1 \quad 1 \quad 1]^\top$$

$$P[-1 \quad -1 \quad -1 \quad 1]^\top = [-1 \quad -1 \quad -1]^\top$$

we see that the two different points $(1,1,1,1)$ and $(-1,-1,-1,1)$ have the same image point $(1,1)$ in Faugeras' camera model. Evaluating the images of the same two points in our camera model

$$\iota(1,1,1,1) = (1,1,1,1)$$

$$\iota(-1,-1,-1,1) = (1,1,1,-1)$$

we see that in homogeneous coordinates the image points are different points of $\mathbf{P}^3$, which is why ι does not represent the forming of an image in homogeneous coordinates. Observe however that the actual image coordinates are the first two components obtained after evaluating ι. We could define two 4-tuples to be equivalent when their first three coordinates are equal, and the fourth coordinates are non-zero. □

We now show that the two camera models are equivalent, provided points on the focal plane are ignored. If $\mathbf{P}$ is any perspective camera matrix then let

$$g = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let $[u \quad v \quad w]^\top = \mathbf{P}[x \quad y \quad z \quad t]^\top$. Then we have that

$$\iota g[x \quad y \quad z \quad t]^\top = [u/w \quad v/w \quad 1 \quad t/w]^\top$$

The first three coordinates are equal to the image coordinates in the previous version of the perspective camera model. Thus this model also embodies the perspective camera model.

Example 2.10

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} x+t \\ y \\ z \\ t \end{bmatrix}$$

The affine transformations are not well-defined in the non-homogeneous coordinate system where the fourth coordinate is arbitrary. However, if t is fixed, they are well-defined. This is why we introduced the convention that the fourth coordinate of scene point (x, y, z) is 1. □

Any camera is represented by ιg where

$$g = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\alpha_u > 0, \alpha_v > 0$ and $\mathbf{R}$ is a rotation. Observe that g is an affine transformation with $det(g) = det(R)\alpha_u\alpha_v = \alpha_u\alpha_v > 0$. g transforms the scene coordinates into what we will call the camera coordinate system. Now we can see that ι is really a canonical camera, and that every camera is related to the canonical camera ι by an affine transformation from the scene coordinate system to the camera coordinate system. It is also important that $det(g) > 0$, because this means that even when the camera matrix is unknown, an orientation is preserved (clockwise versus anticlockwise). The only thing that can reverse the sign of $det(g)$ is a reflection between an object and its image, which is clearly impossible without a deliberate inversion of image coordinates.

The use of a fourth coordinate is a device that leads to new constraints on pairs of images in chapter 9. These constraints will turn out to hold in situations where there are no projective invariants and where no epipolar constraint is available. They will enable stereo reconstruction in cases where reconstruction would previously have been regarded as impossible, and require less prior knowledge than algorithms based on projective invariants or epipolar geometry.

2.5 Simulating cameras

In several places in this thesis we will perform simulations of cameras. These simulations will involve the random generation of points in three dimensions (simulated scene), the random generation of one or more camera matrices (cameras), and the formation of their image/s. Simulations are useful to find errors in the implementation of an algorithm, to test the accuracy of an approximation, to test numerical stability of an implemented algorithm, or solve a system of equations by symbolic algebra. The simulations will be of two types: those involving floating point algorithms and those involving symbolic algorithms. Most of the simulations using symbolic algebra have been performed using Maple. Small simulations using floating point data have also been done using Maple. Larger simulations were implemented in C, on Sun SS10

and DEC Alpha computers. Some of the Maple and C source code is tabulated in appendix B.

Most systems contain a random number generator. The author's implementation uses a system function called drand48, which computes uniformly distributed random numbers in the interval $(0, 1)$ using the linear congruential algorithm [186]. The random number sequence is initialised by calling srand48, using the system time as the seed. This ensures that if a program is run twice, the same numbers will not be reproduced. The simulations in this thesis do not produce uniformly distributed tuples of points. The points will be distributed according to a Gaussian distribution with mean zero. This is done by transforming two numbers produced by a linear congruential generator into two numbers with the desired distribution using algorithm 2.1. Figure 2.11 depicts a histogram of ten million random numbers produced by the program testrand.c (see appendix B) using algorithm 2.1. The linear congruential generators have the property that any two numbers produced by them are statistically independent (see the book by Chung [34] for the definition of independence in probability theory). This is a necessary condition for algorithm 2.1 to be used. If a different mean for the points distribution is desired, it is just added to the output of algorithm 2.1. All this is done because of the following result due to Kendall [99]

> *Suppose the distribution of each point of a finite tuple of points in the plane is Gaussian, and they are distributed independently with the same mean and variance. When the influence of rotations, translations and scaling is eliminated, the distribution of the transformed tuple is uniform.*

In other words under the above mentioned conditions every shape is equally likely. Suppose an $m \times n$ matrix is randomly generated by a generator whose random numbers are statistically independent and Gaussian distributed with mean 0 and variance 1. Then the probability that the rank of the matrix is less than $\min(m, n)$ is zero (see [46]). Thus if we randomly generate a $3 \times k$ matrix in this manner, it represents, with probability 1, k points in a scene such that no four of these points are coplanar.

In this thesis certain operations of linear algebra are regarded as primitives: LU factorisation, determinant, inverse, QR decomposition, Singular Value Decomposition (SVD) and computing an orthonormal basis of a collection of vectors. A thorough knowledge of linear algebra is presumed throughout this thesis, see [170] for example. Numerical issues were taken into consideration as far as practicable in developing the C implementation, based largely on the book by Stoer and Bulirsch [169]. The C implementation (see appendix B) uses the public domain package NAPACK written by Hager [69] to implement these operations (see table 2.1).

To perform simulations involving symbolic algebra we will need to generate camera matrices whose entries are rational numbers (the ratio of two integers). The main difficulty in producing camera matrices whose entries are rational numbers is to produce rational rotation matrices. The generator we will describe differs from other generators of random rotations (see [46]) in that the rotation matrices have rational entries. We now give a lemma which shows how to construct rational rotation matrices:

Algorithm 2.1

Inputs	d_1, d_2	uniformly distributed random numbers in $(0, 1)$
	sd	standard deviation of distribution of output random numbers
Outputs	x_1, x_2	random numbers independently Gaussian distributed with mean 0 and standard deviation sd.

do
$v_1 := 2.0d_1 - 1.0$
$v_2 := 2.0d_2 - 1.0$
$s := v_1^2 + v_2^2$
while $(s > 1)$
$s := -2\log(s)/s$
$s := sd\sqrt{s}$
$x_1 := v_1 s$
$x_2 := v_2 s$

Table 2.1

NAPACK routines used	
fact	LU factorisation
det	determinant (from LU factors)
vert	matrix inverse (from LU factors)
qr	QR decomposition
basis	orthonormal basis for collection of vectors
sing	singular value decomposition (by Householder reduction)

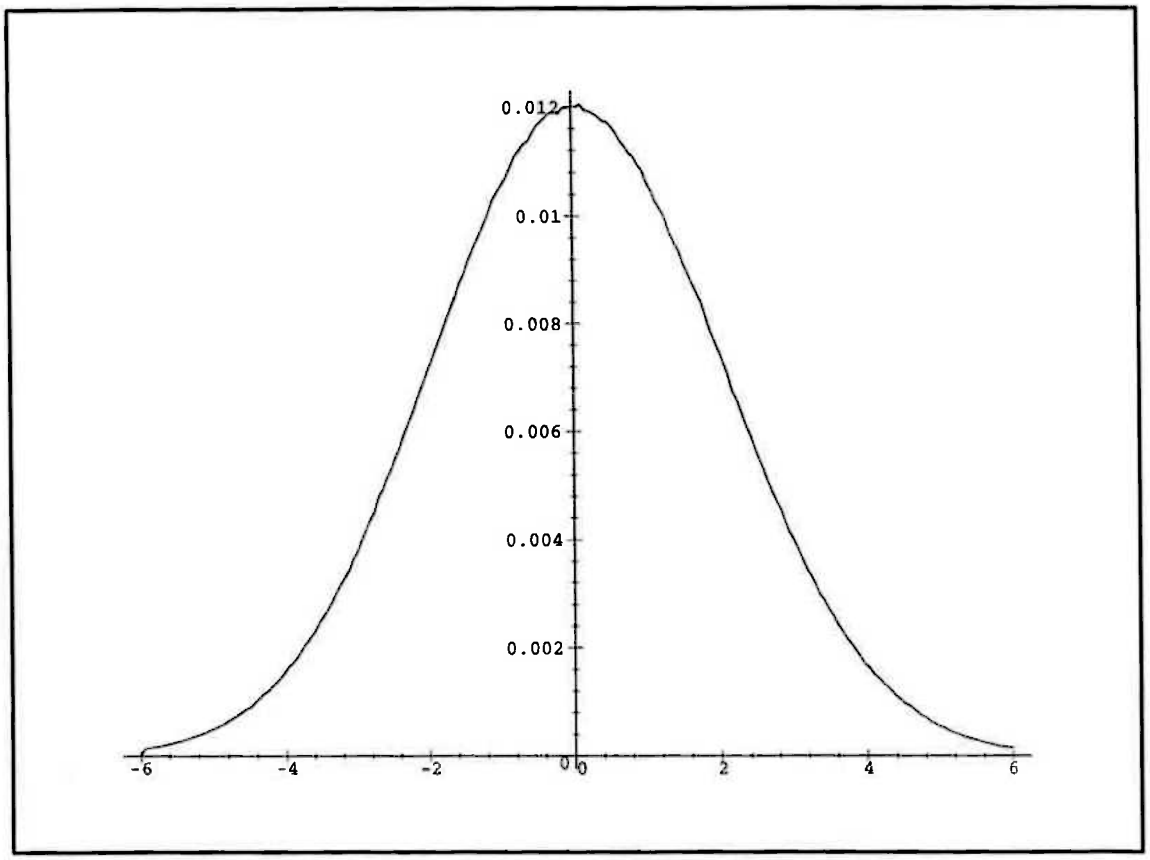


Figure 2.11: Histogram of ten million random numbers (program testrand.c)

Lemma 2.2
If x is rational then $(\frac{2x+1}{2x^2+2x+1}, \frac{2x^2+2x}{2x^2+2x+1})$ is a point on the unit circle with rational coordinates. Any point on the first quadrant of the unit circle can be approximated to any desired accuracy by points of this form.

Proof

Let $p = 2x^2 + 2x$.
$\Rightarrow 2p = 4x^2 + 4x$
$\Rightarrow 2p + 1 = 4x^2 + 4x + 1 = (2x + 1)^2$
$\Rightarrow ((p + 1) + p)((p + 1) - p) = 2p + 1 = (2x + 1)^2$
$\Rightarrow (p + 1)^2 - p^2 = (2x + 1)^2$
$\Rightarrow p^2 + (2x + 1)^2 = (p + 1)^2$
$\Rightarrow (2x + 1/p + 1, \frac{p}{p+1})$ is on the unit circle.

Suppose y is an integer and z is a different and non-zero integer.

$$\frac{y}{z} = \frac{y}{y+z-y} = \frac{y/(z-y)}{y/(z-y)+1}$$

Thus if $p = y/(z - y)$ then $y/z = p/(p + 1)$. It follows that every rational number except 1 can be expressed in the form $p/(p + 1)$ for some rational p.

$$p = 2x^2 + 2x \geq \frac{1}{2}$$

If $p \geq 0$ there is an x such that $p = 2x^2 + 2x$, but x may be irrational. For example $p = 1/3 \neq 2x^2 + 2x$ for any rational x. Since we can approximate x by a rational number to any desired accuracy, we can approximate p to any desired accuracy. ∎

Example 2.11
Taking $x = 1$ and using lemma 2.2 we see that $(3/5, 4/5)$ is a point on the unit circle with rational coordinates. Thus

$$\begin{bmatrix} 3/5 & -4/5 \\ 4/5 & 3/5 \end{bmatrix}$$

is a 2×2 rotation matrix. □

Example 2.12
Any rotation of $\mathbf{R}^3$ can be expressed as a product of rotations about each of the coordinate axes. Consequently

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 5/13 & -12/13 \\ 0 & 12/13 & 5/13 \end{bmatrix} \begin{bmatrix} 3/5 & -0 & 4/5 \\ 0 & 1 & 0 \\ 4/5 & 0 & 3/5 \end{bmatrix}$$

is a rational rotation matrix. □

Example 2.13
Consider a rotation by 15 degrees, which is a rotation by $\pi/12$ radians. Solving

$$(2x^2 + 2x)(1 - \sin(\pi/12)) - \sin(\pi/12) = 0$$

for x and then computing a rational approximation to x we obtain $x = -17174/14913$ and $x = 15780/104081$. Evaluating

$$(\frac{2x+1}{2x^2+2x+1}, \frac{2x^2+2x}{2x^2+2x+1})$$

we obtain

$$(-289834155/300058397, 77660828/300058397)$$

and

$$(14117650921/14615667721, 3782813160/14615667721)$$

respectively. The latter is a rational approximation to $(\cos(\pi/15), \sin(\pi/15))$. In order to check this, we evaluate both in floating point, obtaining approximation $0.9659258263, 0.2588190449$ and true value $0.9659258263, 0.2588190450$, which is correct to a high precision. □

Chapter 3

Feature points

Pattern recognition [45], and optical character recognition in particular, begins with the notion of a feature. A feature is a distinctive measurement, that will identify a class to which an object belongs, and determine that it is unlikely to belong to other classes. Computer vision developed the notion of a feature point from this. Two categories of feature points are predominant in computer vision: edges and corners. Edges are thought of as image points where there is a contrast in brightness. Corners are considered to be image points where there is a sharp change in the shape formed by edges. A large number of algorithms have been proposed for detecting edges and corners; a different definition of edge or corner is implicit in each. Three of the key requirements for a definition of feature to be meaningful are:

- The feature is computable from images, but is a property of surfaces in the three-dimensional scene whose image was taken;
- The property is not dependent on viewpoint;
- The feature provides information that can be utilized in computing point correspondences

This chapter introduces a theory of feature detection with which we can study these types of properties for two types of algorithms.

- (I) Algorithms computing zero-crossings of one or more differential operators. Examples are the Gradient and Laplacian edge detectors, and the Harris corner detector.
- (II) Algorithms computing intersections of curves where a scalar valued differential operator is constant, but an associated vector valued differential operator is non-zero.

This theory will elucidate the relationship between image features and three dimensional surface features in a more rigorous manner than can be found elsewhere in the literature. The theory produces several new conclusions.

- There is a systematic error in certain type (I) algorithms such as the Gradient and Laplacian edge detectors
- Algorithms of type (I) work because of local properties of smooth surfaces
- Algorithms of type (II) work because of global properties of surfaces, and do not require an apriori assumption of smoothness
- Algorithms of type (II) have attributes that can be used to establish point to point correspondences along edges

3.1 Type I feature detectors

3.1.1 Basic principle of edge detectors

Edge detectors attempt to detect image points where there is a discontinuity in brightness. A simple, idealized example is a step discontinuity.

$$f(x) = \begin{cases} 1 & x \in [0,1] \\ 0 & x \in [-1,0) \end{cases}$$

The basic idea of edge detection is that the derivative of a function will tend to be large at a step discontinuity.

Since an image only provides brightness information on a discrete set of points, the Taylor series is used to calculate approximate values of derivatives, where $f^{(n)}$ denotes the n-th derivative of f.

$$f(x) = f(0) + f^{(1)}(0)x + f^{(2)}(0)x^2/2! + \ldots$$

By substituting different values of x, this provides a system of equations in the unknowns $f^{(1)}(0), f^{(2)}(0), \ldots$ that can be solved to compute these derivatives. The simplest example is

$$f^{(1)}(0) \approx f(1) - f(0)$$

Gradient

The gradient edge detector defines a point (x, y) to be an edge if

$$(\frac{\partial I}{\partial x})^2 + (\frac{\partial I}{\partial y})^2$$

evaluated at (x, y) is a maximum.

If the gradient is a maximum at (x, y) then $\frac{\partial^2 I}{\partial x^2} = 0$ and $\frac{\partial^2 I}{\partial y^2} = 0$. Thus this is a type I algorithm.

Laplacian

The Laplacian edge detector defines a point (x, y) to be an edge if

$$\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

evaluated at (x, y) is close to zero. The Laplacian detects local extrema of the gradient, and is a type (I) algorithm.

The basic principle is not rigorous

Consider a one-dimensional cross section of the image. This is a one-dimensional signal $f(t)$. A function f is said to be continuous at t if $f(s) \to f(t)$ as $s \to t$. A function f is said to be differentiable at t if the limit as s tends to t of $\frac{f(s)-f(t)}{s-t}$ exists. In fact, if f is differentiable, then it must also be continuous [20], because as $s \to t$

$$f(s) - f(t) = (s - t)\frac{f(s) - f(t)}{s - t} \to 0 df/dt = 0$$

The simple idea justifying edge detectors such as the Gradient is that derivatives will tend to be large at a discontinuity. We cannot make this simple concept rigorous, because the derivative is meaningless at a discontinuity. We will adopt an alternative approach that is rigorous.

3.1.2 Imperceptible variations in images

Images always contain noise, which can be confirmed by a simple experiment. Point a camera at an unchanging scene, such as a wall. Capture a sequence of images $I_1, \ldots, I_m$ of this unchanging scene. You will find that $I_1(x, y), \ldots, I_m(x, y)$ are not the same for most values of (x, y). Thus the images you have are not the true image brightness $I(x, y)$, but $I(x, y) + n(x, y)$ where n is a random noise.

The electronics of a camera contain devices such as diodes and BJT's which generate noise (in fact the distribution of each device's noise is known [28]). This is one cause of the observed noise. Another is the random arrival of photons at a light sensor. It is important to note that there are also small image fluctuations that are not due to noise.

One such small image fluctuation is line jitter, caused by lack of synchronization between the camera clock and the frame grabber clock, if the camera was not a digital camera. Line jitter is only perceptible (to a human) if it is severe; the video will seem to be vibrating vertically.

Lighting fluctuations may also manifest themselves in images. Since indoor lighting is driven by alternating current, each successive frame may have different brightness. This does not affect most cameras, because they integrate charge over a period of time,

averaging out rapid lighting fluctuations. However, if the electronic shutter speed of the camera can be varied, lighting fluctuations can become apparent at high shutter speeds. An experiment can be performed to determine whether lighting fluctuations are present. Compute the histogram of each image in a sequence of images of an unchanging scene, such as a wall. If the histogram does not shift, lighting fluctuations have no effect on the captured images.

3.1.3 Noise suppression filters

Suppose $n(t) = a\sin(\omega t)$ where a is small but ω is large. This is low amplitude, high frequency noise.

$$\frac{d}{dt}(f+n) = \frac{df}{dt} + a\omega\cos(\omega t)$$

Now the amplitude of the noise term in the derivative has been amplified by ω. Hence differentiation amplifies high frequency noise.

A number of filters have been derived in the literature, to estimate maxima of df/dt given an f perturbed by additive noise [54]. Using these also produces a type I algorithm.

Canny filter

The Canny filter [30] assumes n is stationary white noise, and filters $f(t) + n(t)$ to find local maxima in df/dt with

- maximum signal to noise ratio in filtered signal;
- minimum error in location of maxima of df/dt;
- only one response at a maximum of df/dt.

White noise means that the distribution of $n(t)$ is Gaussian

$$\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-x^2}{2\sigma^2}}$$

with mean zero and standard deviation σ. Stationary means that this distribution does not change with time, so σ is a constant.

The Canny filter is a function h such that

- $h * (f + n)$ satisfies the preceeding conditions, where $*$ denotes convolution
- h is zero outside some interval $[-W, W]$ (finite impulse response filter)

More precisely we

- maximise signal to noise ratio;
- maximise inverse of standard deviation of x_0 where $x_0 \approx 0$ and $h * (f + n)$ is a local maximum;
- subject to the constraint $x_{max} = kW$ where k is a constant and x_{max} is the average distance between extrema of $h * (f + n)$.

The Canny filter is

$$h(x) = e^{-ax}(a_1 \sin(\omega x) + a_2 \cos(\omega x)) + e^{ax}(a_3 \sin(\omega x) + a_4 \cos(\omega x)) - 1$$

where $a = 2.05220$, $\omega = 1.56939$, $a_1 = 0.1486768717$, $a_2 = -0.2087553476$, $a_3 = -1.244653939$, $a_4 = -0.7912446531$. See [54] for the proof.

Deriche filter

The Deriche filter h is an infinite impulse response filter with

- maximum signal to noise ratio; and
- maximum inverse of standard deviation of x_0.

The Deriche filter is

$$h(x) = a_1 \omega e^{-a|x|}$$

where a_1, ω are the same constants as defined in the Canny filter. See [54] for the proof.

Spacek filter

The Spacek filter is a (finite impulse response) filter h with

- $h = 0$ outside $[-W, W]$; and
- signal to noise ratio maximised; and
- $\frac{1}{\text{standarddeviation}(x_0)}$ maximised; and
- $x_{\max}(h)/W$ maximized

Uniqueness of response is no longer a constraint, it is incorporated into the objective function being maximised.

The Spacek filter is

$$h(x) = (a_1 \sin(ax) + a_2 \cos(ax))e^{ax} + (a_3 \sin(ax) + a_4 \cos(ax))e^{-ax} - l/2$$

If $W = 1$ the constants are $a_1 = -13.3816$, $a_2 = 2.7953$, $a_3 = 0.0542$, $a_4 = -3.7953$, $a = 1$, $l = -2$. See [54] for the proof.

Gaussian filter

A Gaussian filter is commonly used before computing the Laplacian edge detector, as advocated by Marr and Hildreth [84].

3.1.4 Canny edge detector

One of the most widely used edge detectors for greyscale images is the Canny edge detector [30], in which local maxima of a filtered gradient are regarded as edges, and an algorithm called hysteresis thresholding is used to extend edges to connected chains. The filter is chosen to be optimal with respect to the criteria of maximal response to a step edge, minimal displacement from the true edge and minimal number of responses to each edge, see [54]. Deriche produced a filter from a larger class which improves on the Canny filter with respect to these criteria. A discrete implementation of this filter is described in [54]. The program canny.c in the hvision package (see appendix A) calculates the gradient of an image, performs non-maxima suppression and hysteresis thesholding, and produces a bitmap comprising edges[1]. The output of canny.c is converted to pbm format (see appendix A) using hs2rast (from hvision), rasttopnm and pgmtopbm (from pbmplus). We generate closed contours around this using edge2.c and edge3.c. The program edge5.c in appendix B calculates binary string descriptors of each connected component. We can see that there are thousands of connected components, because the edges are fragmented. We could not find values of the parameters to canny.c which reduce this fragmentation. (Other parameter values produce as many as seventy thousand connected components.)

3.1.5 Phase congruency, local energy and Gabor functions

Phase congruency is a property that we can prove a step edge possesses [142]. There is evidence that there are phase congruency edge detectors in the brain [174], [175]. It has also been shown that it can detect various other types of edges. Phase congruency is neither a type I nor type II algorithm.

Consider the Heaviside function

$$f(x) = \begin{cases} 1 & x \in [0,1] \\ 0 & x \in [-1,0) \end{cases}$$

Calculate its Fourier Series

$$f(x) = \sum_{n=-\infty}^{+\infty} \frac{< f, e^{i\pi nx} >}{< e^{i\pi nx}, e^{i\pi nx} >} e^{i\pi nx}$$

$$< f, e^{i\pi nx} > \; = \; \int_0^1 e^{-i\pi nx} dx$$

[1] canny.c in hvision does not include a Canny filter.

$$
\begin{aligned}
&= \int_0^1 \cos(-i\pi nx) + i\sin(-i\pi nx)dx \\
&= \frac{i}{n\pi}(\cos(-n\pi) - 1) \\
&= \begin{cases} 0 & \text{n even} \\ -2i/\pi n & \text{n odd} \end{cases}
\end{aligned}
$$

$$< e^{i\pi nx}, e^{i\pi nx} >= 2$$

$$f(x) = \sum_{n=1,3,5,\ldots} \frac{2\sin(n\pi x)}{n\pi}$$

Notice that all the terms are functions of x which intersect at $x = 0$, and that point is a step-edge.

We define the phase congruency function as

$$\mathrm{PC}(x) = \max_{\theta \in [0,2\pi)} \frac{\int a_w cos(wx + \psi_w - \theta)dw}{\int a_w dw}$$

where f is reconstructed from its fourier transform by

$$f(x) = \int_{-\infty}^{+\infty} a_w cos(wx + \psi_w)dw$$

PC has a maximum at a step edge (and various other edges), and PC has values between 0 and 1.

Local energy is the length of the vector

$$\mathbf{E} = (f(x), h(x))$$

where $h(x)$ is the Hilbert tranform of f

$$h(x) = \int_{-\infty}^{+\infty} \frac{f(y)}{x - y}dy$$

Local energy is easier to compute than PC because

$$\hat{h}(w) = isgn(w)\hat{f}(w)$$

that is, the Hilbert Transform corresponds to phase shifting in frequency domain.

Local maxima in E coincide with local maxima in PC because E is proportional to PC.

Responses of certain receptive fields in area of V1 of monkey visual cortex are accurately fitted by Gabor functions.

$$G(x, w_0) = e^{iw_0 x} e^{-\frac{(x-x_0)^2}{2\sigma_0^2}}$$

$G = G_r + iG_i$ is a complex valued function.

The local energy given by the length of the vector

$$(G_r * f, G_i * f)$$

gives a biologically inspired edge detector. Gabor functions achieve the theoretical lower bound of uncertainty in both space and spatial frequency.

3.1.6 Experiments on connectivity of edges

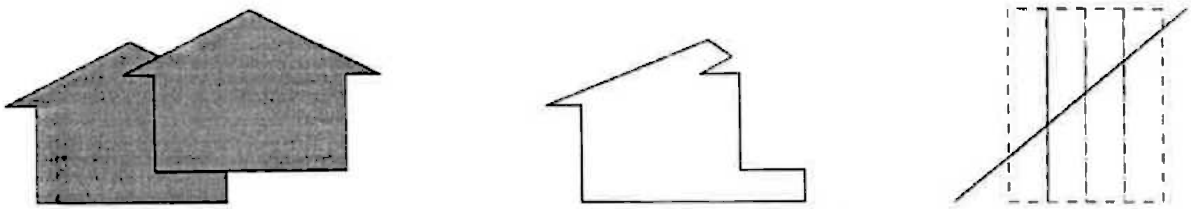
Mumford and Shah [131] developed a variational formulation of the problem of finding an optimal decomposition of the plane (to segment an image into a finite set of regions), and solved it. They prove that the optimum class of functions to approximate boundaries is a set of non-linear elastic splines [139]. However, they report that it takes several hours to calculate such splines. So they developed an approximation based on cubic splines, that is implemented by curves.c in hvision [140]. Their implementation requires many arbitrary parameters, and often crashes if good parameters are not guessed. When we provided a picture of a straight line to curves.c, it output a list of about 200 components.

We also tried the Marr-Hildreth edge detector [54] implementation in Khoros, and a phase congruency edge detector [103] implementation provided by Kovesi. The phase congruency algorithm took several hours to execute but appeared to produce the best result. All these algorithms except the phase congruency algorithm failed to compute the boundary of a black line on a white sheet of paper[2].

A camera creates optical distortions due to its aperture (diffraction), defocus and quantisation. This has been formulated as a convolution of a point spread function with an image; White and Schowengerdt [184] state a point spread function to account for circular aperture, defocus and quantisation. Berenstein [10] showed that the inversion of a simultaneous set of convolution equations can be a well-posed problem. Yaroslavsky and Caulfield extend this to account for white noise [190] and give parameter estimates. The electronics of a camera contain devices such as diodes and BJT's which generate noise (in fact the distribution of each device's noise is known [28]).

Some researchers consider idempotence to be a necessary condition for an edge detection operator, and regard edge detection as a projection (in function space) [143], [150]. This involves filters or idempotent operators, which map the space of functions on the plane to a subspace. We can demonstrate by a simple example that projection can render classification impossible. Consider the orthogonal projection of the plane onto a line, and define classes as vertical strips (see figure 3.1(b)). Then the preimage of any point on the line can lie in any of the classes. The compatibility condition on an operator which maps function space to a subspace is that the preimage of each point on the subspace is contained entirely within a class.

[2]Owing to the unknown file format, we have not counted connected components for the phase congruency algorithm.



(a) (b)

Figure 3.1: (a) Contour deformation due to occlusion, (b) incompatible classes and subspace

3.1.7 Corner detectors

Edge detectors such as Canny [30] and Marr-Hildreth [84] are reported to produce no response at corners and junctions [95]. Corners can be defined directly as features of grey level images, without finding edges first.

Nitzberg et al [140] defines the smaller eigenvalue of the matrix

$$\begin{bmatrix} \frac{\partial I}{\partial x}^2 & \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \frac{\partial I}{\partial y}^2 \end{bmatrix}$$

at each image point to be corner strength. The program curves.c in hvision computes this on a 3×3 neighbourhood of each image point, and outputs the result as an image where darker points have higher corner strength.

Harris [72], [73] defines a corner as a point where $\det(A) - 0.04\text{trace}(A)^2$ is large, where A is the matrix

$$\begin{bmatrix} \frac{\partial I}{\partial x}^2 & \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \frac{\partial I}{\partial y}^2 \end{bmatrix}$$

convolved with a Gaussian (to smooth it). This is a type I algorithm. Harris' corner detector was a modification of the Moravec interest operator [129].

A definition of corners using second order differential operators is given by Wang and Brady [95], and implemented on a hybrid parallel processor.

Phase congruency edge detectors can detect junctions and corners as well as edges [103].

Some other corner detectors are [193] (implemented in the image-matching program from INRIA, see appendix A), [44], [101] and [136].

Stereo and motion algorithms based on corners typically use the value of the corner feature to find an initial set of hypotheses of corresponding points, and use other constraints to resolve ambiguous matches and verify matches.

3.1.8 Topological spaces

We gave the reader a definition of continuity for real valued functions. There is a more general definition of continuity which makes sense on more abstract spaces. One type of abstract space is a surface in the scene. We do not want to explicitly describe every conceivable surface, so we just think of surfaces as sets with certain properties. There is also a more abstract type of space we will work with in this thesis, constructed from equivalence classes. This occurs in vision because there are many unknown camera parameters, so we need to regard certain scenes and images as equivalent. Another type of abstract space is the solution set of a system of polynomial equations in several variables, called an algebraic variety. The more general definition of continuity is based on the notion of an open set. A set S in $\mathbf{R}$ is called open if for every $x \in S$ there is an interval (a, b), where $a \neq b$, containing x and contained in S. Note that a, b are allowed to be plus or minus infinity. The open sets of $\mathbf{R}$ are just unions of open intervals. The open sets of $\mathbf{R}^m$ are unions of disks.

Example 3.1
The set $[1, 2)$ is not open, because any interval $(1 - \epsilon, 1 + \epsilon)$ contains points less than 1 when $\epsilon > 0$. These points are not in the set $[1, 2)$. The set $\{x \in \mathbf{R} \mid x \neq 0\}$ is open, because it consists of two open intervals $(-\infty, 0)$ and $(0, \infty)$. □

A set S is called closed if there is an open set O which consists of all those points excluded from S.

Example 3.2
The set $[1, 2)$ is not closed. It is not open either. All finite subsets of $\mathbf{R}^m$ are closed. The intervals $[a, b]$ are closed. □

In topology, we just define some collection of sets to be the open sets of a space, provided:
(i) unions of open sets are open;
(ii) intersections of finite numbers of open sets are open;
(iii) the whole space and the empty set are open.
The space is then called a topological space. Most of the topological spaces we encounter have a metric d. In this case, the open sets are defined to be unions of disks $\{x \mid d(x_0, x) < \epsilon\}$. However, we will actually encounter examples of topological spaces that have no metric in chapter 4. Such spaces are associated with very unstable algorithms.

A function from one topological space to another $f : A \to B$ is called continuous if for every open subset $U \subset B$, $f^{-1}(U)$ is open in A. For real valued functions, this definition is equivalent to the convergent sequence definition we gave earlier (see theorem 10.3 in [133]).

If a continuous function with a continuous inverse exists between two topological spaces, they are said to be homeomorphic. From the viewpoint of topology, homeomorphic spaces have exactly the same properties, because the open sets of the two spaces will correspond one to one.

3.1.9 Manifolds

The spaces we will be most interested in are locally like $\mathbf{R}^m$. A topological space is called locally Euclidean if every point has an open set around it, and a homeomorphism onto an open subset of $\mathbf{R}^m$. The prototype of a locally euclidean space is a sphere. Every point on a sphere can be projected onto a plane, but not all of a sphere can be visible from that plane. In analogy with the earth, we can think of a locally euclidean space as being mapped by an atlas which consists of charts: each chart is an open subset of the space, together with the homeomorphism relating it to $\mathbf{R}^m$. All the (open subsets of the) charts cover the space. It is apparent that surfaces in a scene will be locally euclidean.

The notation for charts is as follows. If M denotes the locally Euclidean space, and U denotes an open subset belonging to a chart, then the homeomorphism is denoted by $\psi : U \to V$, where V is an open subset of $\mathbf{R}^m$. The chart is denoted by (U, ψ).

We can safely assume that surfaces in scenes have two more technical properties called the Hausdorff property and the second countability property. We will defer the introduction of these two other properties to chapter 5, where they will have to be used.

We can define the notion of smoothness on locally euclidean spaces as follows. Suppose two charts (U_1, ϕ_1) and (U_2, ϕ_2) overlap, which is to say that $U_1 \cap U_2$ is not empty. Then $\phi_1 \circ \phi_2^{-1}$ is a mapping from (an open subset of) $\mathbf{R}^m$ to (another open subset of) $\mathbf{R}^m$. If all such mappings are smooth or infinitely differentiable functions, then we consider the locally euclidean space to be smooth.

Example 3.3
Consider the unit circle. We have no method of differentiation on the circle itself, so we cannot say it is smooth because we can differentiate functions on it. Suppose we define charts for the circle using the functions cos and sin on neighbourhoods on which they can be inverted. Then $\theta \mapsto cos(sin^{-1}(\theta))$ and $\theta \mapsto sin(cos^{-1}(\theta))$ are smooth. Hence the unit circle is a smooth locally euclidean space. It is also possible to give a square four charts so that none of the corners are in the region of overlap of two charts. □

Definition 3.1
A locally euclidean, Hausdorff, second countable topological space with a smooth structure is called a manifold.

The branch of mathematics concerned with studying manifolds is called differential geometry. Many of the ideas of calculus have been generalised in differential geometry.

A function f on a manifold can be locally redefined as a function on every chart (U, ϕ), namely $f \circ \phi^{-1}$. We call f a smooth function if all of these local redefinitions on $\mathbf{R}^m$ are smooth.

Given that a manifold is defined as an abstract space, not as a subset of $\mathbf{R}^m$, it is quite difficult to define a tangent vector. Surprisingly, there is a quite ingenious method by which we can define tangent vectors on manifolds. The trick is to observe that

directional derivatives can almost be considered to be tangent vectors for surfaces in Euclidean space.

Example 3.4
Consider the unit circle in the plane, parameterised by $(\cos(t), \sin(t))$. The tangent vector at any point is given by

$$(\frac{d}{dt}\cos(t), \frac{d}{dt}\sin(t)) = (-\sin(t), \cos(t))$$

This is perpendicular to the vector from the origin to the point, as we would expect the tangent of the circle to be. □

Example 3.5
Let U be an open subset of $\mathbf{R}^2$. Differentiation of a function g, in a direction (α, β) is

$$\alpha\frac{\partial g}{\partial t} + \beta\frac{\partial g}{\partial s}$$

So $\alpha(\partial/\partial t) + \beta(\partial/\partial s)$ is the symbolic representation of a map

$$g \mapsto (\alpha\frac{\partial}{\partial t} + \beta\frac{\partial}{\partial s})(g)$$

Each such map produces a direction vector on U, and is regarded as a tangent vector to U. □

Example 3.6
Suppose $\gamma_1(t,s), \gamma_2(t,s), \gamma_3(t,s)$ is a parametrisation of a surface in $\mathbf{R}^3$ on an open subset U of $\mathbf{R}^2$. A tangent vector can be any linear combination

$$\alpha(\frac{\partial\gamma_1}{\partial t}, \frac{\partial\gamma_2}{\partial t}, \frac{\partial\gamma_3}{\partial t}) + \beta(\frac{\partial\gamma_1}{\partial s}, \frac{\partial\gamma_2}{\partial s}, \frac{\partial\gamma_3}{\partial s})$$

However, (α, β) has no meaning as a direction in $\mathbf{R}^3$, it is actually a direction on U. Moreover, we are still using the basis of $\mathbf{R}^3$, which we cannot do for general manifolds. We can write this tangent vector to the surface symbolically as

$$\alpha(\frac{\partial\gamma_1}{\partial t}\frac{\partial}{\partial x} + \frac{\partial\gamma_2}{\partial t}\frac{\partial}{\partial y} + \frac{\partial\gamma_3}{\partial t}\frac{\partial}{\partial z}) + \beta(\frac{\partial\gamma_1}{\partial s}\frac{\partial}{\partial x} + \frac{\partial\gamma_2}{\partial s}\frac{\partial}{\partial y} + \frac{\partial\gamma_3}{\partial s}\frac{\partial}{\partial z})$$

which represents a map from functions on the surface to their directional derivatives in direction $\alpha(\partial/\partial t) + \beta(\partial/\partial s)$. Using the chain rule this map is

$$f \mapsto \alpha\frac{\partial}{\partial t}(f \circ \gamma) + \beta\frac{\partial}{\partial s}(f \circ \gamma)$$

We call this map the tangent vector to the surface, because if f is a function on the surface, then $f \circ \gamma$ is a function on U, and the map can be calculated without referring to $\mathbf{R}^3$. Moreover, (the output of this mapping) is symbolically the same thing as a directional derivative in $\mathbf{R}^3$. □

Definition 3.2
Let $g : \mathbf{R}^2 \to \mathbf{R}$, $b \in \mathbf{R}^2$. The mapping v_b such that

$$v_b(g) = (\alpha\frac{\partial}{\partial t} + \beta\frac{\partial}{\partial s})(g)|_b$$

is called a tangent vector at $b \in \mathbf{R}^2$. We can evaluate $v_{\phi(a)}(f \circ \phi^{-1})$ on each chart (U, ϕ) at a point $a \in M$. The mapping $v_a : f \mapsto v_{\phi(a)}(f \circ \phi^{-1})$ is called a tangent vector at $a \in M$.

Tangent vectors obey the linearity and Liebniz rules that derivatives in $\mathbf{R}^m$ satisfy.

Lemma 3.1
For each a in a manifold M, and smooth functions f, g on M, and $\alpha, \beta \in \mathbf{R}$
$v_a(\alpha f + \beta g) = \alpha v_a(f) + \beta v_a(g)$ (linearity);
$v_a(fg) = v_a(f)g(a) + v_a(g)f(a)$ (Liebniz).
where fg denotes the function $fg(x) = f(x)g(x)$ at each x.

Proof

$$\begin{aligned} v_a(\alpha f + \beta g) &= \sum \tfrac{\partial}{\partial x_i}(\alpha f + \beta g) \circ \phi^{-1} \mid_{\phi(a)} v_i && \text{by definition of } v \\ &= \sum \tfrac{\partial}{\partial x_i}(\alpha f \circ \phi^{-1} + \beta g \circ \phi^{-1}) \mid_{\phi} (a) v_i \\ &= \alpha \sum \tfrac{\partial}{\partial x_i}(f \circ \phi^{-1}) \mid_{\phi(a)} v_i + \beta \sum \tfrac{\partial}{\partial x_i}(g \circ \phi^{-1}) \mid_{\phi(a)} v_i && \text{by linearity of } \partial/\partial x \\ &= \alpha v_a(f) + \beta v_a(g) && \text{by definition of } v \end{aligned}$$

The Liebniz rule follows from the Liebniz rule of calculus as follows

$$\begin{aligned} v_a(fg) &= \sum \tfrac{\partial}{\partial x_i}((f \circ \phi^{-1})(g \circ \phi^{-1})) \mid_{\phi(a)} v_i && \text{by definition of} \\ &= \sum [(f \circ \phi^{-1}) \tfrac{\partial}{\partial x_i}(g \circ \phi^{-1}) \mid_{\phi(a)} + (g \circ \phi^{-1}) \tfrac{\partial}{\partial x_i}(f \circ \phi^{-1}) \mid_{\phi(a)} v_i] && \text{from calculus} \\ &= f(a) v_a(g) + g(a) v_a(f) \end{aligned}$$

∎

The tangent vectors at a point a in a manifold M form a vector space, known as the tangent space, and denoted $T_a(M)$.

Given a smooth function f from one manifold M to another manifold N, there is an analogous mapping from $T_a(M)$ to $T_{f(a)}(N)$, called the differential. It is defined by the rule $df(v)(g) = v(g \circ f)$ for each tangent vector v to M and smooth function g on N. The result is a tangent vector on N.

3.1.10 Properties of type I algorithms

We will now study the behaviour of edge detectors of type I when viewing three dimensional scenes. An occluding point is a point in the image where a ray through the optical centre of the camera was tangent to the surface. We elucidate the relationship between occluding points and zero-crossings of some edge detector. It is shown that if the surface in the scene is smooth, then zero-crossings can be occluding points, but they can also be zero-crossings (critical points) of some unknown vector field on the three-dimensional surface. The occluding points are edges, but the images of surface vector field critical points are not edges; they will be regarded as false positives.

The function we are interested in is the image formation map ι. For each vector on a scene surface, the differential produces a vector on the image.

Example 3.7
Consider the circle $(z-1)^2 + x^2 = 1/2$. Let $\iota : (x, z) \mapsto x/z$ produce a one-dimensional

image of this circle. The circle is a manifold, and ι is considered a function on this manifold. At the point $x = 1/2, z = 1/2$ the tangent to the circle passes through the origin, so this is an occluding point of the image. Let t be the image coordinate, and (α, β) a tangent vector to the circle at point (x, z). Calculating the differential

$$d\iota(\alpha\frac{\partial}{\partial x} + \beta\frac{\partial}{\partial z}) = (1/z - x/z^2)\frac{\partial}{\partial t}$$

So the tangent vector to the circle at $(1/2, 1/2)$ is

$$v = 1/2\frac{\partial}{\partial x} + 1/2\frac{\partial}{\partial z}$$

Evaluating $d\iota(v)$ we find it is zero. Since $\alpha = x, \beta = z$ at any occluding point, $d\iota(v) = 0$ at any occluding point. □

Lemma 3.2
Consider an image of a smooth surface. If a two component smooth vector field on this image is zero at a point, it is either an occluding point, or a critical point of some smooth vector field on the surface.

Proof

Let $\iota : (x, y, z) \rightarrow (t, s)$ where $t = x/z, s = y/z$, and $z \neq 0$. Then

$$d\iota(v_1\frac{\partial}{\partial x} + v_2\frac{\partial}{\partial y} + v_3\frac{\partial}{\partial z}) \mid_a = (\frac{v_1}{z} - \frac{v_3 x}{z^2})\frac{\partial}{\partial t} + (\frac{v_2}{z} - \frac{v_3 y}{z^2})\frac{\partial}{\partial s}$$

Thus $d\iota(v) \mid_a = 0$ if and only if v is a multiple of a, or $v = 0$. Thus $d\iota(v)$ has rank 2 at all points on a surface disjoint from the focal plane, except occluding points, where it has rank 1.

The differential of ι maps a vector field on a smooth surface to a vector field on its image. Vector fields on images minus the occluding points can be mapped to vector fields on a smooth surface by the inverse differential. ∎

This result can be applied to algorithms that compute local maxima of two component differential operators, such as the gradient. Since local maxima of $(\partial I/\partial x, \partial I/\partial y)$ are points where

$$(\partial^2 I/\partial x^2, \partial^2 I/\partial y^2) = \mathbf{0}$$

Local maxima of any operator correspond to zeroes of a higher order operator, and are therefore critical points of some vector field.

If we think purely in terms of grey values, one might think that points with zero gradient have constant brightness, and should not comprise edges. Nonetheless, experiments on real images show that zeroes of the gradient do contain significant numbers of edge points. Of course, it is also well known that zeroes of the Laplacian contain significant numbers of edge points. This can once again be deduced from the fact that

$$\partial^2 I/\partial x^2 + \partial^2 I/\partial y^2 = 0$$

if

$$(\partial^2 I/\partial x^2, \partial^2 I/\partial y^2) = \mathbf{0}$$

However, we can also deduce that both of these algorithms must contain a systematic error. A zero-crossing of a differential operator could indicate an occluding contour, but it might also be the image of a zero-crossing of the unknown surface vector field. This will appear in the image as a false positive of edge detection. Such a false positive is not due to noise.

3.1.11 Limitations of the type I paradigm

The type I algorithms have some other limitations in addition to the systematic error described above (see lemma 3.2). These limitations motivate the development and study of type II feature detectors.

A priori assumption of smoothness

The elementary physical concept of reflection is that a light ray incident to a surface will be reflected at an angle to the surface normal equal to the angle of incidence, such that incident and exitant rays lie in the same plane through the normal vector. A mirror is an example of such an ideal, or specular reflector. The microscopic structure of most materials in scenes is not smooth, so a scene surface will scatter light in different directions. Most materials are also not perfectly homogeneous on a microscopic scale, and thus scatter light rays that penetrate the surface by refraction and reflection at boundaries between regions of different refractive indices. Scattered rays may reemerge near the point of entry, and so contribute to diffuse reflection. Snow and layers of white paint are examples of materials with this behaviour [90].

A surface is called Lambertian if it appears equally bright from all directions, regardless of how it is irradiated, and reflects all incident light [90]. This does not imply that different points on the same surface have the same intensity, because the value of the intensity depends on other variables such as the angle between the surface normal and a light source. When a smooth Lambertian surface is rendered by computer graphics, its appearance is like a dull smooth plastic [115].

Most materials in scenes are neither specular reflectors nor Lambertian reflectors, but a combination of both. A surface rendered by computer graphics will appear shinier as the specular component is increased [115], [81]. The appearance of texture (to the eye) is due to particularly rapid variations in normal vectors over small regions. For example, a random pattern of normal vectors when rendered by computer graphics can produce an effect of fog. Various repetitive patterns of normal vectors can be used to render real materials, such as bricks by computer graphics [81].

Owing to the microstructure of a scene surface, the light radiated from the surface undergoes extremely rapid intensity changes over small regions. This can be confirmed by examining gray values of neighbouring points in a real image. Thus an assumption that image intensity is piecewise smooth may not be a reliable assumption for a feature detection algorithm.

Difficulties with establishing point correspondences

The best known algorithm for establishing point to point correspondences utilize corner feature points. The correlation between a rectangular neighborhood of corners in two video frames is used to match them. However, the corresponding neighborhood to a rectangle in one image will not be a rectangle in another in general. An image of a rectangle from different viewpoints is not a rectangle.

Matching the shapes formed by sets of corners is also difficult because

- feature points can be permuted in a different view, so an a priori assumption of connectivity on the basis of proximity is frequently wrong;
- unless we can associate small regions in two images (as in motion or motion stereo algorithms), it is impossible to solve the correspondence problem given only a set of points with no further structure.
- a set of points viewed from a different viewpoint, with permutations can be recognized using invariants to perspective and permutation [110], [111]. When we have to also find subsets that correspond in this sense, because of occlusion and slight perturbation of the points, the problem becomes intractable.

Problems with edge connectivity

The connectivity of edges is a problem that no known algorithm solves satisfactorily.

- even when to the human eye, edge points appear connected, they were usually disconnected (this is also reported by other authors, such as Rothwell [152] page 157);
- edge detectors may fail to detect many edges obvious to the human eye (even though they may not be illusory edges);
- those contours which do result may intersect themselves or each other, unlike silhouettes or boundaries;
- parts of the contours can be hidden by another object (occlusion), so that the contour is effectively deformed (see figure 3.1(a)).
- in order to group contours into silhouettes an unmanageably large set of thresholds have to be chosen.

Various ad hoc algorithms such as the split and merge algorithm are used in an attempt to overcome connectivity problems in the output of edge detectors, such as the Canny detector. These do not entirely solve the problem. This is once again confirmed by other authors, for example page 159 of [152] states that split and merge fails to segment curves correctly.

In view of the difficulty we encounter with the topology of "edges", it is natural to pose the following questions.

- When do the curves obtained by joining vectors of a vector field in a head to tail fashion form well-behaved families like a topographic map?
- Can we find a closed form solution to these curves for any operator?
- When do the vector fields or curves produced from an image correspond to vector fields or curves on surfaces?

These questions are deep questions about differential geometry. The answers to these questions lead to the idea that the intersections of certain curves we can generate from an image (associated with a differential operator), must indicate edges.

3.2 Type II feature detectors

A feature detector of type II is an algorithm computing intersections of curves where a certain type of scalar differential operator is constant, but an associated vector valued differential operator is non-zero. In the next subsection type II features will be shown to be either surface edges, occluding points or brightness contrasts. A type II feature detector does not require an apriori assumption of smoothness; the type II feature is a contradiction to strong smoothness and visibility assumptions.

This section describes a canonical type II algorithm (algorithm 3.6), that outputs attributes at each image point. Feature points have a non-zero attribute whereas other points have zero attributes. It is also possible to group the feature points into a pl3 data structure; subsequently used to generate point to point hypotheses of corresponding points, along edges in two images (see chapter 8).

3.2.1 Sufficient condition for occlusions and discontinuities

A vector valued differential operator maps a greyscale image to a vector field. It is implicitly suggested by Hoffman [86] that the primary visual cortex of the brain produces some sort of vector field. It should be possible to line up the vectors in a head to tail fashion to form contours. We want to know when this results in a family of closed oriented contours rather like a topographic map. The answer is provided by theorem 3.6 below. The key part of the proof is the famous theorem of Frobenius, from differential geometry.

Example 3.8
$(\partial/\partial x, \partial/\partial y)$ is called the gradient operator. The light intensity (brightness) is a function I of image coordinates x, y. $(\partial I/\partial x, \partial I/\partial y)$ evaluated at any image point is a vector, so it forms a vector field. The gradient operator maps I to such a vector field. □

Example 3.9
$(-\partial/\partial y, \partial/\partial x)$ is called the Hamiltonian operator. $(-\partial I/\partial y, \partial I/\partial x)$ evaluated at any image point is a vector; in fact a vector perpendicular to the gradient vector, since

$$\frac{\partial I}{\partial x}(-\frac{\partial I}{\partial y}) + \frac{\partial I}{\partial y}\frac{\partial I}{\partial x} = 0$$

The Hamiltonian is consequently a perpendicular operator to the gradient operator. □

Example 3.10
Suppose I is a smooth function (intensity). Let M be the subset of the plane on which $(-\partial I/\partial y, \partial I/\partial x)$ is non-zero. M is an open set because by definition of continuity the inverse of a continuous function maps open sets to open sets, and $\mathbf{R} \setminus \{0\}$ is open. The map

$$(x, y) \mapsto \text{span}(\frac{-\partial I}{\partial y}(x, y), \frac{\partial I}{\partial x}(x, y))$$

is a smooth function that maps M to the set of lines through the origin in the plane $\mathbf{P}^1$ (projective space). It is an example of what is called a one dimensional distribution in differential geometry [179], [26]. The vector field is called a basis for this distribution. □

Example 3.11
Consider $f(x, y) = x + y$ and $g(x, y) = e^{x+y}$.

$$-\frac{\partial f}{\partial y} = -1 \quad -\frac{\partial g}{\partial y} = -e^{x+y}$$

$$\frac{\partial f}{\partial x} = 1 \quad \frac{\partial g}{\partial x} = e^{x+y}$$

Thus at every point in the plane, the Hamiltonian vector of g is e^{x+y} times the Hamiltonian vector of f. These are two different vector fields that are a basis for the same distribution. On the other hand the Hamiltonian vector of $e^x + e^y$ is not collinear with the Hamiltonian of f, so not every pair of vector fields does span a one dimensional distribution. □

Definition 3.3
Suppose Y is a vector field on an open subset M of $\mathbf{R}^2$, and it is the basis for a distribution on M. Let y_1, y_2 denote coordinates on M. The distribution is called integrable if at every point one can find a disk on which a coordinate system (x_1, x_2) can be chosen such that

$$Y^1\frac{\partial}{\partial y_1} + Y^2\frac{\partial}{\partial y_2} = \frac{\partial}{\partial x_1}$$

Lemma 3.3
Any one dimensional distribution is integrable.

Proof

See proposition 11.1.1 in [26] or proposition 1.53 in [179]. ∎

Let (a, b) be an interval on the real line, but a can be $-\infty$ and b can be $+\infty$. A differentiable curve $\gamma : (a, b) \rightarrow \mathbf{R^2}$ is an integral curve of a vector field X if $\gamma'(t) = X(\gamma(t))$ for all $t \in (0, 1)$ where γ' denotes the derivative of γ.

Thus an integral curve is a curve whose tangent at each point coincides with the vector of the vector field at the same point. Since the gradient vector points in the direction of steepest descent, perpendicular vectors are produced by the Hamiltonian operator, and if we join these Hamiltonian vectors in a head to tail fashion, they line up into integral curves.

The next lemma is due to Faugeras (see pg112 of [54]). It shows us how to calculate the integral curves of the Hamiltonian in closed form, circumventing all the errors associated with numerical differentiation, numerical equation solving, and even floating point arithmetic.

Lemma 3.4
The integral curves of the Hamiltonian of image intensity are curves with $I(x, y) =$ constant.

Proof

Let C be a curve on the surface $z = I(x, y)$. Let c be the parallel projection of C onto the x-y plane. Let M be a point on C, and m be its projection onto c. Let $G = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, -1)$ be the gradient of the surface at M. Let T be a unit vector tangent to C at M, and t be a unit vector tangent to c at m. Let g be the projection of G onto the x-y plane.

From example 3.6 any tangent vector T to the surface has the form

$$\alpha\frac{\partial}{\partial x} + \beta\frac{\partial}{\partial y} + (\alpha\frac{\partial I}{\partial x} + \beta\frac{\partial I}{\partial y})\frac{\partial}{\partial z}$$

Thus the normal vector G is perpendicular to any tangent vector T since $G.T = 0$. Consequently, $T.G = T_x g_x + T_y g_y - T_z = 0$.

$$t\sqrt{T_x^2 + T_y^2} = [T_x, T_y]^\top$$

$$(t.g)\sqrt{T_x^2 + T_y^2} = g_x T_x + g_y T_y$$

Since $T_x g_x + T_y g_y = T_z$,

$$(t.g)\sqrt{T_x^2 + T_y^2} = T_z$$

Thus $t.g = 0$ implies $T_z = 0$ if $\sqrt{T_x^2 + T_y^2} \neq 0$. From $T.G = 0$, if $T_x = T_y = 0$ then $T_z = 0$ also. Hence $t.g = 0$ implies $T_z = 0$.

We have shown that an image curve whose tangent is always perpendicular to the gradient has $I(x, y) =$ constant. This implies that the image curves that are integral curves of the Hamiltonian have constant intensity. ∎

Corollary 3.5
Let K be a scalar valued smooth differential operator on the plane. The integral

curves of

$$(-\frac{\partial}{\partial y}(K)(I), \frac{\partial}{\partial x}(K)(I))$$

are curves with $K(I)(x, y) = \text{constant}$.

Proof

Let $I' = K(I)$. Then the integral curves of the Hamiltonian of I' are curves with $I'(x, y) = \text{constant}$.

Consider the smooth vector fields X defined by

$$(-\frac{\partial I'}{\partial y}, \frac{\partial I'}{\partial x})$$

and Y defined by

$$((-\frac{\partial}{\partial y}K)(I), (\frac{\partial}{\partial x}K)(I))$$

An integral curve of X is a curve γ such that

$$X(\gamma(t)) = \gamma'(t)$$

An integral curve of Y is a curve γ such that

$$Y(\gamma(t)) = \gamma'(t)$$

Since X and Y coincide at each point of the plane, the integral curves of X and Y coincide. ∎

An integral curve is called a complete integral curve if its domain is $\mathbf{R} = (-\infty, +\infty)$. An integral curve is called a maximal integral curve if it is not contained in any other integral curve.

Theorem 3.6
Let X be a smooth vector field on the plane, and M the subset of the plane on which X is non-zero. Each point of M lies in precisely one maximal integral curve, and there is a one parameter family of such integral curves in M. An integral curve does not intersect itself.

Proof

By lemma 3.3 the distribution $m \mapsto \text{span}(X(m))$ is integrable. Proposition 11.2.1 in [26] asserts that an integrable distribution is involutive. A one dimensional integral submanifold of M is a smooth curve in M whose tangent equals the line in the distribution at each point of the curve. The Frobenius theorem (theorem 1.60) in [179] asserts that there is a cubic coordinate system centred at each point of M, with coordinates x_1, x_2 such that $x_2 = \text{constant}$ are integral submanifolds of M. By theorem 1.64 in [179], there is a unique maximal integral submanifold passing through each point of M. By proposition 11.3.1 in [26] any integral curve of X is a integral submanifold of M. Consequently each point of M lies in precisely one complete integral curve, and there is a one parameter family of such curves in M. ∎

Suppose M_1 is a smooth two dimensional manifold in the scene, whose tangent planes are everywhere disjoint from the optical centre of a camera. Suppose also that the intensity on M_1 is smooth. Then there is a smooth invertible mapping between vector fields on M_1 and vector fields on the image of M_1. Moreover, the non-vanishing vector fields on M_1 correspond to the non-vanishing vector fields on the image of M_1, because the tangent planes were disjoint from the optical centre. From theorem 3.6, each point in a smooth image of a smooth and unoccluded surface lies in precisely one maximal integral curve of a smooth image vector field, and there is a one parameter family of such integral curves. Thus if an image point lies in more than one maximal integral curve, either the intensity is not smooth, or the surface is not smooth, or the surface is occluded. In other words, we have obtained a sufficient condition for a singularity in intensity, or the surface or an occlusion.

The surfaces of objects in scenes have another property, namely they are bounded, and can be considered to be closed sets. A closed and bounded subset of $\mathbf{R}^m$ is called compact. A more general definition of compactness can be given in the more general setting of topological spaces. A collection of open sets is said to cover a set S if S is contained in the union of all the open sets in the collection. A set S is called compact if every collection of open sets covering S has a finite sub-collection that covers S. This is not an easy property to grasp, but it is a very important one. Theorem 6.3 in [133] asserts that the closed and bounded subsets of $\mathbf{R}^m$ coincide with the compact subsets.

Surfaces can therefore be considered to be compact manifolds, but this includes parts of the surface that are not visible, or in contact with other surfaces like the ground. The illumination of this surface defines an intensity at each point on this manifold, which is a function I_3 on this manifold. The image formation function ι maps this manifold onto a manifold in the image plane, The composition $I_3 \circ \iota^{-1}$ defines the intensity on the image plane.

It can be shown that all maximal integral curves of a compact smooth manifold are complete [179]. Thus a hypothesis that an image is a smooth image of an unoccluded, smooth compact surface implies that all maximal integral curves are closed curves. We intuitively know that this follows from an absurd assumption, namely that *all* of a surface is visible. However, since we want to reject such a hypothesis, we must employ the consequences which are most certain to lead to a contradiction. This is why we will generate only closed curves, by completion of curves if necessary. The phenomenon of completion occurs in human vision, and can be demonstrated vividly by an optical illusion called the Kanisza triangle.

The next section will discuss the problem of generating integral curves of the Hamiltonian from grayscale images. The algorithm we shall develop finds an exact solution, avoiding error prone numerical computations such as filtering, and circumventing the somewhat ad-hoc procedures of non-maximum suppression and hysteresis thresholding. By calculating intersections of these curves, we will detect discontinuities in intensity (and discontinuities in derivatives of intensity of all orders), discontinuities in scene surfaces (and discontinuities in derivatives of scene surfaces of all orders) and occlusions.

3.2.2 A new digital geometry

The book by Haralick and Shapiro [70] contains a survey of algorithms that label connected components in a bitmap. Two pixels belong to the same connected component if they can be joined by a sequence of pixels that are neighbours and have the same value. The connected component depends on the definition of neighbour. When the north, east, west and south neighbours of a pixel are considered neighbours the connected components are called 4-connected. When the northeast, northwest, southeast and southwest neighbours are considered neighbours in addition (to the north, east, west and south neighbours) the connected components are called 8-connected. Rosenfeld [151] showed that if one definition is used for 0 pixels and the other definition is used for 1 pixels then of two bordering regions one is a hole in the other. The algorithms in [70] use a different definition of neighbour for 0 and 1 pixels.

Recall that under the hypotheses of an unoccluded smooth image of a smooth compact surface, all maximal integral curves of the image's Hamiltonian vector field are simple closed curves. Of course, images cannot contain hidden surfaces, and a surface must either be occluded somewhere or suffer singularities. However, these events are detected by finding points belonging to two different maximal integral curves. Since level curves in the conventional digital geometry may not be simple or closed, these are not maximal integral curves. We will introduce a digital geometry in which all level curves will become completed into simple closed curves.

An image is only a discrete sample of intensity of light reflected from a surface. If a pixel has a particular grey value, none of its immediate neighbours necessarily has the same grey value. It is therefore not feasible to calculate level curves of image intensity by trying to chain together pixels that have a particular grey value. A more sensible procedure is to calculate the set of image points whose grey value is less than the particular grey value, and then compute the boundary of this set. This operation is called thresholding. In the authors implementation, a greymap is represented by a file in pgm (portable gray map) format. Thresholding converts this file into a pbm (portable bit map) file. The algorithm to compute (completed) level curves therefore operates on a set of pbm data structures.

We will develop an algorithm that finds boundaries of connected components in a bitmap, without first calculating connected components. Zero and one pixels both have the same definition of neighbour, that is eight neighbours. It differs from other algorithms in that two contours result with opposite orientations, that do not intersect each other, never intersect themselves and are always closed. The idea of separate contours with different orientations is due to Ghosh [61], he calls it negative shape. The resulting discrete plane geometry is a new digital geometry, that differs from the conventional digital geometries in [70].

We define a new grid with four positions for every image grid point: east, north, west and south of the image grid point. This new grid will be called the NWES-grid. Thus in a binary image, for each transition between 0 and 1, there can be two grid points marked (see figure 3.2(a)). In a binary image, this prevents a situation where the contours formed by linking edge points together intersect themselves, or

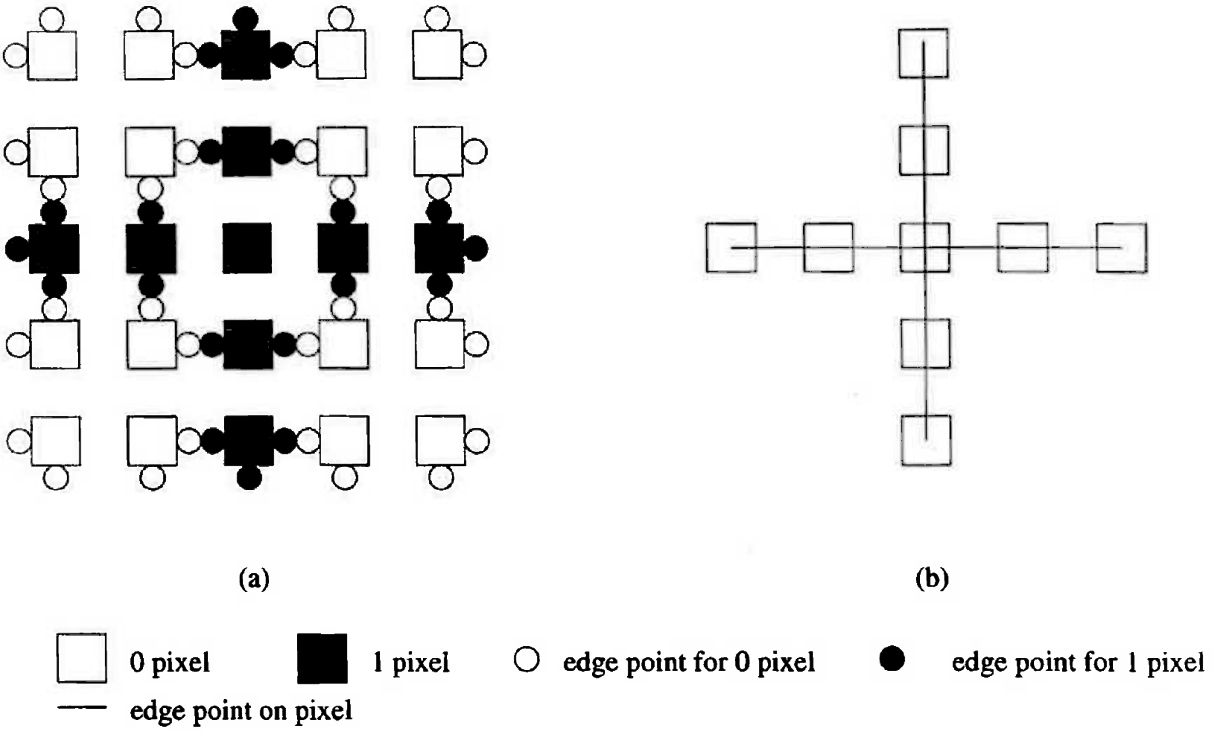


Figure 3.2: (a) Edge points (b) Result of one edge point per transition

fail to close (see figure 3.2(b)). The program of [140] records edges in the interstices between pixels. However we implicitly record two points for each transition between contrasting pixels.

We record image point positions as integers (n, m). And level curve point positions as $(n-0.25, m)$ west, $(n, m-0.25)$ north, $(n+0.25, m)$ east, and $(n, m+0.25)$ south. Neighbouring pairs of points are both involved in a contrast. So at each point only south and east NWES-grid points are set; the neighbours north and west NWES-grid points are also set. The points at the borders of the image are an additional case. The level points of northern, western, eastern and southern border points respectively are set. This ensures that the boundaries of regions adjacent to borders are closed.

The next step is to give rules by which to link these NWES-grid points together, or in mathematical jargon: a discrete topology. The rules have the effect that they give outer boundaries a clockwise orientation, and inner boundaries an anticlockwise orientation. See figure 3.3. Black filled circles in figure 3.3 are points of a NWES-grid that have been set. When a north point is set, (top left diagram), it can be mapped to one of the three states below it (downward arrows depict this). If there is a north point set of the pixel on the west, a link is introduced, as depicted in the diagram in the second row and first column of figure 3.3. If there is an east point set of the north western pixel, a link is introduced, as depicted in the diagram in the third row and first column of figure 3.3. The remaining rules are similar, except for the two diagrams on the bottom row in columns two and three of figure 3.3. If the western NWES-grid point is set (diagram on row 1 column 2 in figure 3.3) and the north NWES-grid point is also set, they can only be linked if the pixel to the north west is opposite in value to the current pixel (diagram on bottom row column 2 of figure 3.3). If the eastern

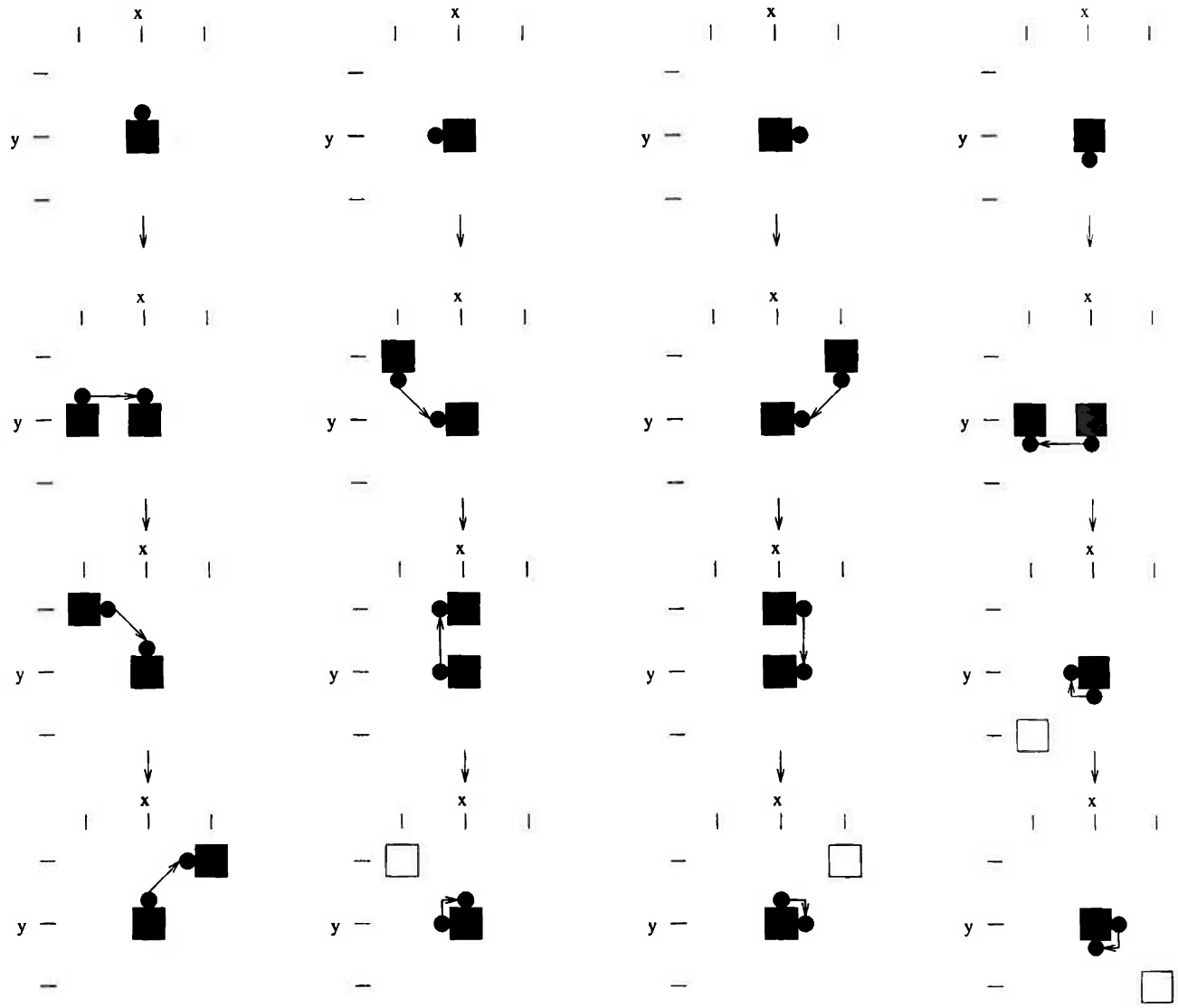


Figure 3.3: Edge point linking rules

NWES-grid point is set (diagram on row 1 column 3 in figure 3.3) and the north NWES-grid point is also set, they can only be linked if the pixel to the north east is opposite in value to the current pixel (diagram on bottom row column 3 of figure 3.3).

The pl1 data structure

The result of the operation described above is a collection of points accompanied by a pointer to the next point in a closed curve. It is represented by a pl1 file, which consists of a list of records defined in C as follows.

```
struct pl1_record {
   float x,y;  /* this vertex coordinate */
   int next;   /* record number of next vertex */
}
```

The curve generating operation must input a pbm file, and output a pll file. However, pll files are expected to have certain ordering properties to calculate intersections efficiently. The records in a pll file must be spatially ordered in the same way we read English: the record representing a point in a lower row must come after it; the record representing a point to the right of another point in the same row must come after it; two records must not represent the same point. If we follow the next pointer of any record, and continue following it, we must return to the record we started at, after at least four steps. In other words, all the records in a pll file describe closed polygons. Pairs of loops in the same pll file are expected to not intersect each other. The name pll is an abbreviation for piecewise linear spatially ordered file.

Note that the next field of a pll record should be set so that record numbers start at one, not zero. Zero or negative record numbers are sometimes used as markers.

The C header file datadict.h provides functions to support basic operations such as opening, creating and closing pll files (see appendix B).

The pbmtopll function

The function pbmtopll() in algorithm 3.1 implements the conversion of a pbm file to a pll file. Its inputs are the pbm array I, the number of rows (num_rows) in I, the number of columns in I (num_cols), the size of pll array (num_records) and two work arrays (Ed and Nx). Its outputs are the pll array (pll) and the number of pll records (num_records), and the return status (return value). The entries of work array Ed(x,y) are records, with four one bit fields N,W,E,S representing the four edge vertices. The pbmtopll function does not need more than 3 rows at a time of Ed. The entries of work array Nx(x,y) are records consisting of x,y coordinates of the next vertex in a loop, and the current record number. The current record number can be set to -1 to represent unused, or 0 to represent not yet allocated. The pbmtopll function does not need more than 4 rows at a time of Nx. The return value is 0 on successful completion, 1 if an error relating to dynamic memory usage occurs, and 2 if the pll array is not large enough. Record numbers are allocated to row y-1 in algorithm 3.3 because the successor of a NWES-grid vertex in row y-1 can be in row y; only row y NWES-grid vertices have been marked and linked yet. Algorithm 3.4 writes pll records in row y-2 because the successor of a NWES-grid vertex in row y-2 can be in row y-1; only row y-1 has record numbers allocated as yet. All functions within the pbmtopll function are inline functions, to avoid the overhead of millions of function calls. If this facility is not available, macros or include files should be used.

Algorithm 3.1

```
function pbmtopll(num_rows,num_cols,I,pll,num_records,Ed,Nx){

  if I or pll or Ed or Nx are NULL return 1
  if(num_rows > 0){
    set North border values Ed(x,0).N=I(x,0)
```

```
    cur_recno=0
    buffer_start=1
    for(y=0; y < num_rows-1; y=y+1){
      set West border value Ed(0,y).W=I(0,y)
      for(x=0; x < num_cols-1; x=x+1){
        Ed(x,y).S=(I(x,y) xor I(x,y+1)) and I(x,y)
        Ed(x,y+1).N=(I(x,y) xor I(x,y+1)) and I(x,y+1)
        Ed(x,y).E=(I(x,y) xor I(x+1,y)) and I(x,y)
        Ed(x+1,y).W=(I(x,y) xor I(x+1,y)) and I(x+1,y)
        set next pointers x,y, and record=-1 (unused) or 0 (not yet allocated)
      }
      x=num_cols-1
      Ed(x,y).S=(I(x,y) xor I(x,y+1)) and I(x,y)
      Ed(x,y+1).N=(I(x,y) xor I(x,y+1)) and I(x,y+1)
      set East border value Ed(x,y).E=I(x,y)
      set next pointers x,y, and record=-1 (unused) or 0 (not yet allocated)
      if(y > 0) allocate record numbers to vertices in row y-1
      if(y > 1) write edge vertex records of row y-2 to pl1 array
    }
    y=num_rows-1
    set SW corner border value Ed(0,y).W=I(0,y)
    for(x=0; x < num_cols-1; x=x+1){
      set South border value Ed(x,y).S=I(x,y)
      Ed(x,y).E=(I(x,y) xor I(x+1,y)) and I(x,y)
      Ed(x+1,y).W=(I(x,y) xor I(x+1,y)) and I(x+1,y)
      set next pointers x,y, and record=-1 (unused) or 0 (not yet allocated)
    }
    x=num_cols-1
    set SE corner border values Ed(x,y).E=Ed(x,y).S=I(x,y)
    set next pointers x,y and record=-1 (unused) or 0 (not yet allocated)
    allocate record numbers to vertices in row y-1
    write edge vertex records of row y-2 to pl1 array
    y=y+1
    allocate record numbers to edge vertices in row y=num_rows-1
    write edge vertex records in row y=num_rows-2 to pl1 array
    y=y+1
    write edge vertex records in row y=num_rows-1 to pl1 array
  }
  num_records=cur_recno  (this argument is altered)
  return 0
}
```

Algorithm 3.2

```
  inline function set next pointers {
```

```
Nx(x,y,N).record=-1
if ((x,y) inside NWES-grid border and Ed(x,y).N==1){
  if ((x-1,y) inside NWES-grid border and Ed(x-1,y).N==1){
    Nx(x-1,y,N).x=x
    Nx(x-1,y,N).y=y-0.25
    Nx(x-1,y,N).record=0
  }
  if ((x-1,y-1) inside NWES-grid border and Ed(x-1,y-1).E==1){
    Nx(x-1,y-1,E).x=x
    Nx(x-1,y-1,E).y=y-0.25
    Nx(x-1,y-1,E).record=0
  }
  if ((x+1,y-1) inside NWES-grid border and Ed(x+1,y-1).W==1){
    Nx(x,y,N).x=x+1-0.25
    Nx(x,y,N).y=y-1
    Nx(x,y,N).record=0
  }
}
Nx(x,y,W).record=-1
if ((x,y) inside NWES-grid border and Ed(x,y).W==1){
  if ((x-1,y-1) inside NWES-grid border and Ed(x-1,y-1).S==1){
    Nx(x,y,W).x=x-1
    Nx(x,y,W).y=y-1+0.25
    Nx(x,y,W).record=0
  } else {
    if ((x,y-1) inside NWES-grid border and Ed(x,y-1).W==1){
      Nx(x,y,W).x=x-0.25
      Nx(x,y,W).y=y-1
      Nx(x,y,W).record=0
    } else {
      Nx(x,y,W).x=x
      Nx(x,y,W).y=y-0.25
      Nx(x,y,W).record=0
    }
  }
}
Nx(x,y,E).record=-1
if ((x,y) inside NWES-grid border and Ed(x,y).E==1){
  if ((x+1,y-1) inside NWES-grid border and Ed(x+1,y-1).S==1){
    Nx(x+1,y-1,S).x=x+0.25
    Nx(x+1,y-1,S).y=y
    Nx(x+1,y-1,S).record=0
  } else {
    if ((x,y-1) inside NWES-grid border and Ed(x,y).E==1){
      Nx(x,y-1,E).x=x+0.25
      Nx(x,y-1,E).y=y
```

```
        Nx(x,y-1,E).record=0
      } else {
        Nx(x,y,N).x=x+0.25
        Nx(x,y,N).y=y
        Nx(x,y,N).record=0
      }
    }
  }
  Nx(x,y,S).record=-1
  if ((x,y) inside NWES-grid border and Ed(x,y).S==1){
    if ((x-1,y) inside NWES-grid border and Ed(x-1,y).S==1){
      Nx(x,y,S).x=x-1
      Nx(x,y,S).y=y+0.25
      Nx(x,y,S).record=0
    } else {
      if (I(x-1,y+1)==1 and (x-1,y+1) inside NWES-grid border){
        skip
      } else {
        Nx(x,y,S).x=x-0.25
        Nx(x,y,S).y=y
        Nx(x,y,S).record=0
      }
    }
    if (I(x+1,y+1)==1 and (x+1,y+1) inside NWES-grid border){
      skip
    } else {
      if ((x,y) inside NWES-grid border and Ed(x,y).E==1){
        Nx(x,y,E).x=x
        Nx(x,y,E).y=y+0.25
        Nx(x,y,E).record=0
      }
    }
  }
}
```

Algorithm 3.3

```
inline function allocate record numbers {

  Allocate N row record numbers first
  for (x=0; x < num_cols; x=x+1){
    if (Nx(x,y,N).record==0){
      cur_recno=cur_recno+1
      Nx(x,y-1,N).record=cur_recno
```

```
    }
  }
  Allocate W and E record numbers alternately in same row
  for (x=0; x < num_cols; x=x+1){
    if (Nx(x,y-1,W).record==0){
      cur_recno=cur_recno+1
      Nx(x,y-1,W).record=cur_recno
    }
    if (Nx(x,y-1,E).record==0){
      cur_recno=cur_recno+1
      Nx(x,y-1,E).record=cur_recno
    }
  }
  Allocate S record numbers
  for (x=0; x < num_cols; x=x+1){
    if (Nx(x,y-1,S).record==0){
      cur_recno=cur_recno+1
      Nx(x,y-1,S).record=cur_recno
    }
  }
}
```

Algorithm 3.4

```
inline function write edge vertex records {

  Write N vertex records first
  for (x=0; x < num_cols; x=x+1){
    if (Nx(x,y-1,N).record > 0){
      this_recno=Nx(x,y-2,N).record
      if (this_recno > num_records)
        return(2)
      pl1[this_recno-buffer_start].x=x
      pl1[this_recno-buffer_start].y=y-2-0.25
      convert Nx(x,y-2,N).x, Nx(x,y-2,N).y to integer (nextx,nexty) and
         direction [N|W|E|S] nextD
      pl1[this_recno-buffer_start].next=Nx(nextx,nexty,nextD)
    }
  }
  Write W and E records on same row
  for (x=0; x < num_cols; x=x+1){
    if (Nx(x,y-2,W).record > 0){
      this_recno=Nx(x,y-2,W).record
      if (this_recno > num_records)
        return(2)
```

```
      pl1[this_recno-buffer_start].x=x-0.25
      pl1[this_recno-buffer_start].y=y-2
      convert Nx(x,y-2,W).x, Nx(x,y-2,W).y to integer (nextx,nexty) and
         direction [N|W|E|S] nextD
      pl1[this_recno-buffer_start].next=Nx(nextx,nexty,nextD)
    }
    if (Nx(x,y-2,E).record > 0){
      this_recno=Nx(x,y-2,E).record
      if (this_recno > num_records)
        return(2)
      pl1[this_recno-buffer_start].x=x+0.25
      pl1[this_recno-buffer_start].y=y-2
      convert Nx(x,y-2,E).x, Nx(x,y-2,E).y to integer (nextx,nexty) and
         direction [N|W|E|S] nextD
      pl1[this_recno-buffer_start].next=Nx(nextx,nexty,nextD)
    }
  }
  Write S records last
  for(x=0; x < num_cols; x=x+1){
    if(Nx(x,y-2,S).record > 0){
      this_recno=Nx(x,y-2,S).record
      if (this_recno > num_records)
        return(2)
      pl1[this_recno-buffer_start].x=x
      pl1[this_recno-buffer_start].y=y-2+0.25
      convert Nx(x,y-2,S).x, Nx(x,y-2,S).y to integer (nextx,nexty) and
         direction [N|W|E|S] nextD
      pl1[this_recno-buffer_start].next=Nx(nextx,nexty,nextD)
    }
  }
}
```

The pbmtopl1 function has linear time and space complexity in the size of an image. However, the space of the output array is almost always considerably smaller than the space taken by an image.

The pbmtopl1 function has been implemented in C by the edge2() function in the pbmtopl1.h header, and the program edge2.c (see appendix B). The pl1 file is a binary format, so it is not necessarily portable across machine architectures. A pl1 file can be converted into text format by the program catpl1.c (see appendix B). Such a text file can be converted back into the binary pl1 format by the program txttopl1.c (see appendix B).

Sorting into component order

The pbmtopl1 function is the fundamental building block of the feature detection algorithm and stereo correspondence hypothesis generation algorithm that will be presented in this thesis. These two types of algorithms operate directly on pl1 data structures. The rendering of the polylines represented in a pl1 file requires a further sorting step. For this purpose we define a second data structure, the pl2 file.

Each pl2 file consists of a list of records defined in C as

```
struct pl2_record {
  int component;  /* number of component (polygon) */
  float x,y;            /* this vertex's coordinate */
}
```

pl2 is an abbreviation for piecewise linear file ordered by component . It is a binary file format representing large sets of closed polygons none of which intersect themselves.

The records in a pl2 file must be ordered by component. The first record must have component number one, and component numbers must either stay constant or increase by one. Each record represents a point; if the successor of that record has the same component number it represents an adjacent point; if the successor's component number is one more or the record is the last in the file, then the first record in that component represents a point adjacent to it. In other words, a sequence of records with the same component numbers describes a closed loop. A record in a pl2 file must never be repeated, as this means the a polygon intersects itself.

The C header file datadict.h provides functions for such basic operations as opening, creating and closing pl2 files (see appendix B).

The algorithm 3.5 converts a pl1 array into a pl2 array. It also performs a number of integrity tests on the pl1 array. Its inputs are the pl1 array (pl1), which it overwrites, and the number of records in the pl1 array (numrecs). It outputs the pl2 array (pl2), which will have the same number of records as the input pl1 array. It simply follows the next record pointers in the pl1 file, marking used records as it goes, until it returns to the starting point. It then skips forward from the starting point until it comes to an unmarked record. It then repeats the exercise, until the whole pl1 array has been marked as used. Since the loop traversal visits every record once, and the skipping forward never backtracks, the complexity of algorithm 3.5 is linear in space and time.

Algorithm 3.5

```
function pl1topl2 (pl1, numrecs, pl2){

  start_record=0
  component_num=0
```

```
pl2_start_record=0
recnum=0
pl2_recnum=0
recs_used=0
while (recs_used+1 < numrecs){
  component_num=component_num+1
  if(pl1[recnum].next < 1 or pl1[recnum].next > numrecs)
    error bad pl1 file: impossible successor pointer
  while(pl1[recnum].next not equal to start_record and
            recs_used < numrecs){
    check record not already used
    if(pl1[recnum].next==0)
      error bad pl1 file: record recnum+1 read twice
    pl2[pl2_recnum].component=component_num
    pl2[pl2_recnum].x=pl1[recnum].x
    pl2[pl2_recnum].y=pl1[recnum].y
    pl2_recnum=pl2_recnum+1
    skip to next record, then mark record as used
      i=recnum
      recnum=pl1[recnum].next-1
      pl1[i].next=0
    if (pl1[recnum].next < 1 or pl1[recnum].next > numrecs)
      error bad pl1 file: impossible successor pointer
    recs_used=recs_used+1  (used to verify pl1 integrity)
  }
  if (recs_used >= numrecs)
    error bad pl1 file: non-closed polygon
  write last record in cycle
    pl2[pl2_recnum].component=component_num
    pl2[pl2_recnum].x=pl1[recnum].x
    pl2[pl2_recnum].y=pl1[recnum].y
  pl2_recnum=pl2_recnum+1
  pl1[recnum].next=0  (mark record as used)
  recs_used=recs_used+1
  if (recs_used+1 < numrecs){
    skip records until unused record found
    recnum=start_record+1
    while(pl1[recnum].next==0 and recnum < numrecs)
       recnum=recnum+1
    start_record=recnum
  }
  pl2_start_record=pl2_recnum
 }
}
```

Algorithm 3.5 has been implemented in C by the header file pl1topl2.h and the program edge3.c (see appendix B). The program pl2tovrml.c converts a pl2 file into VRML

(virtual reality modelling language), so that it can be superimposed on other graphics and text. See appendices B and A for further details. A pl2 file can be output in text format by the program catpl2.c (see appendix B).

3.2.3 Calculating orientation

We explain how to determine whether a list of point coordinates represents a clockwise oriented or anticlockwise oriented closed curve. Let $(x_1, y_1), \ldots, (x_k, y_k)$ be a list of points in the plane that form a simple closed curve when each point in the list is joined to the next point in the list by an edge, and the last point is joined to the first point in the list by an edge. The exterior angle between two consecutive edges is

$$\arccos\left(\frac{(x_{i+1}-x_i)(x_{i+2}-x_{i+1})+(y_{i+1}-y_i)(y_{i+2}-y_{i+1})}{\sqrt{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2}\sqrt{(x_{i+2}-x_{i+1})^2+(y_{i+2}-y_{i+1})^2}}\right)$$

Its range is between 0 and π, whereas we want an angle between $-\pi$ and π. We obtain this by multiplying the exterior angle by the sign of $\Delta_{i(i+1)} = (x_{i+1}-x_i)(y_{i+2}-y_{i+1}) - (x_{i+2}-x_{i+1})(y_{i+1}-y_i)$, which gives a signed exterior angle, where $k+1$ is to be interpreted as 1 and $k+2$ is to be interpreted as 2.

Example 3.12
Suppose an edge vector $(1,0)$ is followed by the edge vector $(0,1)$. So $x_{i+1}-x_i = 1, y_{i+1}-y_i = 0$ and $x_{i+2}-x_{i+1} = 0, y_{i+2}-y_{i+1} = 1$. Then $\Delta_{i(i+1)} = 1$ so the sign is positive. The first vector is along the positive x-axis and the second is along the positive y-axis so the angle between them is $+\pi/2$. □

If the sum of the signed exterior angles is 2π then the orientation is anticlockwise, and if it is -2π then the orientation is clockwise.

Note that if one of the coordinate axes is reflected, but not the other, then when the sum of the signed exterior angles is 2π, the orientation is clockwise in the unreflected plane coordinates. This happens when the origin of image coordinates is taken to be the top left corner of the image (first row), instead of at the bottom left as is usual in mathematics.

Observe that the polygons only have certain fixed angles, and are quite jagged in appearance. Consecutive edge points which lie on a line are slightly perturbed from the line. This would be irrelevant except that collinear points are a degeneracy for algorithms that search for flat point configurations using the coplanarity statistic introduced in chapter 9. The test of collinearity due to Le [106] will be used to prevent this from happening. The problem of approximating a contour with a polygon is called polygonal approximation, and there is much literature on this in computer vision. [42] use regularization to obtain a polygonal approximation, observing that only a finite number of different polygons seem to result. [3] use curvature extrema to obtain dominant points, and then apply the Split and Merge algorithm [145]; convolution with a Gaussian is used to smooth the curvature, because the Gaussian was proved optimal in the sense that it does not create new zero-crossings at a higher scale [191],

[6]. [128] point out that parameterisation by arc-length gives better results. There are a number of problems with the above work:

- split and merge sometimes oscillates and doesn't terminate;
- numerical approximation issues are not mentioned in [3];
- the result of polygonal approximation is not unique, and the question of finding the set of all solutions, and characterizing it is unresolved;
- multiple polygons may represent non-polygonal objects: this is because an image contains a finite number of sample points from a curve or surface. Different samples produce different polygons with different numbers of vertices. The samples represent a fixed curve with probability zero. Hence images can only represent classes of similar curves.

3.2.4 The canonical type II algorithm

As we have explained earlier in the chapter, to compute occlusions, surface and intensity singularities from a greymap, we have to generate several pll files, and find the set of points that occur in more than one pll file. In this section, we will describe an efficient algorithm for this purpose. Recall that pll files are in the following spatial order: rows from top to bottom of image; columns within the same row in left to right order. Moreover, since pll files have no repetitions, the records are in strictly ascending order. In fact we can convert the x, y coordinates of each record into an integer address, such that the addresses of two records in two different pll files produced from the same greymap are equal if and only if the x, y coordinates are equal. Note that this property distinguishes the spatial order from the weaker lexicographical order. In the lexicographical order ($(x_1, y_1) \leq (x_2, y_2)$ if $y_1 < y_2$ or $y_1 = y_2$ and $x_1 \leq x_2$). In the lexicographical order it is possible that (x_1, y_1) is neither less than or equal nor greater than (x_2, y_2).

Since x can only occur in multiples of 0.25, and its smallest value is -0.25, we can map it to a non-negative integer by multiplying by (the floating point constant) 4.0, (converting to an integer) and incrementing. The same reasoning also applies to y. So x is conceptually mapped to $4x + 1$, and y is conceptually mapped to $4y + 1$. Let w denote the number of columns in each row of the greymap. Then the maximum possible value of $4x + 1$ is $4(w - 1) + 1$, so the integer row width is $4(w - 1) + 2$. So the linear address is $4y(4(w - 1) + 2) + 4x + 1 = 4y(4w - 2) + 4x + 1$.

Thus each pll file is considered as a strictly increasing list of integers, and we are seeking those integers that occur twice in two different lists. We start at the beginning of all the lists. Whichever list is last in order but not at end of list is advanced next. The current record of this file is compared to the current records of all files which preced it in order, and the one file which succeeds it in order; if it is equal, the pair of records are marked; if this value has not been output before it is output. This continues until the end of all the lists has been reached.

In order to keep track of the order of files whose current records are in increasing order, we maintain an array listposn which represents a permutation of 0 to numfiles − 1. This permutation is an index into arrays which store data on specific pl1 arrays. This saves a lot of work, because we do not have to sort any other arrays. When the program is initialised, an initial permutation is computed as follows.

```
listposn[0]=0
for(i=1; i < numfiles; i=i+1){
  j=0
  while (j < i and record 0 of file i >= record 0 of file listposn[j])
    j=j+1
  if (i==j){
    listposn[i]=i
  } else {
    for (k=i-1; k >= j; k=k-1)
      listposn[k+1]=listposn[k]
    listposn[j]=i
  }
}
```

At subsequent steps, the permutation is updated, so that the records of files whose end has already been reached are first, followed by the current records of other files in ascending order.

```
j=0
while (j < numeofs and
       current record file listposn[numeofs] >
                                        current record file listposn[j])
  j=j+1
if (j < numeofs)
  i=listposn[numeofs]
  for (k=numeofs-1; k >= j; k=k-1)
    listposn[k+1]=listposn[k]
  listposn[j]=i
}
j=1+numeofs
while (j > 1+numeofs and
       current record file listposn[numeofs] >
                                        current record file listposn[j])
  j=j+1
if (j > 1+numeofs){
  i=listposn[numeofs]
  for (k=1+numeofs; k < j; k=k+1)
    listposn[k-1]=listposn[k]
  listposn[j-1]=i
}
```

The other arrays we need are: recnums to hold current record numbers of each pl1 array; buffers to hold pointers to the start of each pl1 array; infile_reccounts to store the number of records in each pl1 array. The variable numeofs is the number of the first pl1 file whose current record is least, but not at end of file.

Algorithm 3.6 finds occlusions, singularities in scene surfaces and singularities in reflected light intensity. The output of algorithm 3.6 is the array edgelevels. There is one bit for each intensity level, and edgelevels has one field for each NWES-grid point. Each field has one bit for each intensity level. A bit in a field will be set if a (completed) level curve of that intensity has visited the grid point, provided more than one such curve visits the same grid point.

Algorithm 3.6

```
function graddisc (pgm, numfiles, edgelevels){
  for (each multiple of pgm intensity range/numfiles){
    threshold pgm array
    call pbmtopl1
  }
  for(i=0; i < numfiles; i=i+1)
    recnums[i]=0
  calculate initial permutation listposn[] of file numbers
  lastx=-1.0
  lasty=-1.0
  numeofs=0
  while (numeofs < numfiles){
    update listposn[]
    for(i=0; i < numeofs; i=i+1){
      if (current records of files listposn[numeofs] and listposn[i] equal){
        if (current record file listposn[numeofs] not equal to (lastx,lasty)){
          lastx=field x of current record listposn[numeofs]
          lasty=field y of current record listposn[numeofs]
        }
        negate next record pointers in pl1 files to mark intersection
        update edge levels array
      }
    }
    if (numeofs < numfiles-1 and
         current records of files listposn[numeofs] and listposn[numeofs]+1 ar
         equal){
        if (current record file listposn[numeofs] not equal to (lastx,lasty)){
          lastx=field x of current record listposn[numeofs]
          lasty=field y of current record listposn[numeofs]
        }
        negate next record pointers in pl1 files to mark intersection
        update edge levels array
```

```
    }
    i=0
    while(i < numfiles and
          current record file listposn[i] >= infile_reccounts[listposn[i]]-
      i=i+1
    if (i > numeofs)
      numeofs=i
    if (i < numfiles)
      increment recnums[listposn[i]]  (record counter of file listposn[i])
  }
}
```

Algorithm 3.6 has been implemented by the UNIX C shell programs pgmdisc and pgmdisc2; these call the C programs graddisc1.c and graddisc2.c respectively (see appendix B).

The output of algorithm 3.6 with a large parameter (numfiles ≈ 100) will be almost the entire input pgm image. When this parameter is reduced, there is an abrupt change in the output: the output appears intuitively edge-like. The program pgmdisc produced edges on all input images (not from the same camera or scene) with numfiles set to 20. This phenomenon can be understood in terms of the fact that images are not smooth on a small scale, due to the microstructure of the materials in the scene. With a large parameter numfiles, algorithm 3.6 is responding to the surface microstructure (which manifests itself to the eye as texture). With a smaller value of numfiles, the precise value has no significant impact on the output of algorithm 3.6.

3.2.5 Generating edge descriptor files

Algorithm 3.6 not only produces the points in the domain of the image intensity function at which there is an occlusion, surface or intensity singularity, but also produces limit points of the image intensity function (from different directions). Recall that if a function is continuous at a point, it has only one limit at that point, which must equal the value of the function. On the other hand, if it is discontinuous, it will have several limits. The values of these are formed into a symbol at each NWES-grid point. We produce a pbm file by setting an image grid point to 1 if one of its NWES-grid neighbours in non-zero, and 0 otherwise. Then we can call pbmtopl1 to generate a set of simple closed curves. The symbols on the NWES-grid points can then be transferred to this set of simple closed curves, which we will call edge curves.

It is not advisable to assume that there is a discontinuity at a point produced by algorithm 3.6; the point may be an occlusion instead. As long as scene surfaces are approximately Lambertian reflectors, the symbols produced along edge points will be invariant to perspective. From two images, we can then seek alignments between patterns of symbols. In fact, this is how we will generate hypotheses of correspondence in chapter 8.

It is not necessarily the case that every point on an edge curve will have a non-zero

attribute, nor will every NWES-grid point with a non-zero attribute lie on an edge curve. This is a deficiency of recording edge curves on the same grid; a third type of grid is required. At the present time we do not know such a digital topology, so we have used the NWES-grid for convenience.

For each intensity level of an attributed edge curve point, suppose we follow that level curve until another NWES-grid point with a non-zero attribute is reached. This point must either belong to the same edge or not, this being a binary invariant to perspective. If it does belong to the same edge curve, a loop is formed. Ratios of areas of such loops are invariant to plane affine transformations.

We now introduce the pl3 file data structure, which is the last of the polyline file formats in this thesis. The pl3 or edge descriptor file stores the attributes of those edge points which have loops associated with them. It is however, much more complicated than the other file formats. A pl3 file consists of a series of records defined in C as

```
struct pl3_record {
  int file_num;
  int start_level;
  int end_level;
  int edge_ptr;
  float area;
}
```

A pl3 file has several different types of records, a header, edge separators, point terminators, and attribute records. The type of a record is identified by the file_num field: a value greater than or equal to zero identifies an attribute record; a value EDGE_SEPARATOR, PL3_HEADER or PL3_POINT identifies the other types of records.

The header record must be the first record of a pl3 file. It contains values used to allocate memory for programs which process edge descriptors. The format of a header is as follows.

file_num	=	PL3_HEADER
start_level	=	Maximum number of symbols in descriptors
end_level	=	Total number of symbols/points in file
edge_ptr	=	Total number of descriptors in file
area	=	undefined

The total number of points (or symbols) in a pl3 file must be equal to the number of records in the corresponding edge pl1 file.

An edge separator record marks the beginning of a new descriptor. A descriptor may be null, in which case it is followed by another edge separator or end of file. The format of an edge separator is as follows.

file_num	=	EDGE_SEPARATOR
start_level	=	undefined
end_level	=	undefined
edge_ptr	=	edge pl1 start record number of the edge
area	=	area enclosed by the edge curve

Point terminator records mark the end of the attributes of a symbol. Null symbols have no point terminator record, and are not represented in a pl3 file. Each symbol corresponds to an edge point, where several (completed) level curves intersect. However, each level has its own attributes which are recorded in the attribute records that preced the point terminator record. The format is.

file_num	=	PL3_POINT
start_level	=	undefined
end_level	=	undefined
edge_ptr	=	edge pl1 record number of point
area	=	undefined

Each symbol in a descriptor represents a set of loops starting at the same edge point, following a different level curve to the next point of the same edge, and following the edge back to the start. The attributes of each such loop are recorded in attribute records, whose format is

file_num	=	level pl1 file number >= 0
start_level	=	The record number in level pl1 file file_num where edge_point=level curve point
end_level	=	The record number in level pl1 file file_num where a second level curve point=point in same edge
edge_ptr	=	Edge pl1 record number of same point end_level refers
area	=	Area enclosed by level curve fragment and edge fragment

In this chapter, we studied two types of feature detection algorithms. We pointed out some limitations of the first type of feature detection algorithm, including a previously undetected error in common algorithms of the first type. A rigorous argument detailing the three dimensional meaning of the image feature points computed by type II algorithms was given, using only the perspective camera model. We explained why approximate Lambertian (non-smooth) shading of scene surfaces implies the invariance of the features to viewpoint. Our aim from here on is to produce an algorithm capable of reconstructing the scene from the features of two images. In order to do this, we will first study how to reject pairings of features in two images that are inconsistent with the perspective camera model. The next four chapters are concerned with this problem.

Chapter 4

Induced groups of transformations

This chapter studies the relationship between two images of a scene taken from different viewpoints, with different camera parameters (such as focal lengths). As we saw in chapter 2, a change of viewpoint and camera parameters is equivalent to a transformation of the scene. The set of all possible transformations of the scene forms a structure called a group. In special cases of this problem, a scene transformation group induces an transformation group on the two dimensional image plane. We shall calculate such induced groups in this chapter. In the general case, this approach fails. We will show that in our four dimensional perspective camera model, an image from a general camera and viewpoint is equal to an image from a standard camera, followed by a projective transformation. However, it is not the case that all projective transformations result in this way; an important new equation holds. The projective transformations that do satisfy the equation do not form a group. We show that there does not exist a metric on the quotient of a space of point configurations by the group of projective transformations.

4.1 Induced transformations on the retina

Objects in space obviously move, or are seen from different viewpoints. Such changes in position and attitude are encapsulated by the rigid transformations, which consist of rotations, translations and compositions of these. Thus it is important that we understand how rigid transformations of objects change their images. However, when we do not know the camera matrix of the cameras involved, it is important that we understand how a change in camera parameters affects the images. We will first consider a series of special cases. In each case we choose a type of point configuration, a camera model, and a group of transformations. We then calculate the induced group on the retinal plane.

Suppose

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

This is a perspective camera matrix whose optical axis is the z-axis and whose optical

centre is the origin. Consider a point (x_0, y_0, z_0) in the scene, where $z_0 \neq 0$. If this point undergoes a rotation and translation in the plane $z = z_0$ which is parallel to the retinal plane, what is the effect on the retinal plane? In terms of the projective coordinates of the scene, such a rigid transformation is represented by a matrix of the form

$$\begin{bmatrix} R_{11} & R_{12} & 0 & t_1 \\ R_{21} & R_{22} & 0 & t_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\mathbf{R}$ is orthogonal with determinant one (a plane rotation). The image of the transformed point *in the scene* is

$$\begin{aligned} \mathbf{P}\begin{bmatrix} R_{11} & R_{12} & 0 & t_1 \\ R_{21} & R_{22} & 0 & t_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} &= \mathbf{P}\begin{bmatrix} R_{11}x_0 + R_{12}y_0 + t_1 \\ R_{21}x_0 + R_{22}y_0 + t_2 \\ z_0 \\ 1 \end{bmatrix} \\ &= z_0\begin{bmatrix} R_{11}(x_0/z_0) + R_{12}(y_0/z_0) + t_1/z_0 \\ R_{21}(x_0/z_0) + R_{22}(y_0/z_0) + t_2/z_0 \\ 1 \end{bmatrix} \end{aligned}$$

which is a rigid transformation of the image provided z_0 is constant. This is the induced transformation.

A group is a set G together with a binary operation $\circ$ such that

(0)	$\forall g_1, g_2 \in G \quad g_1 \circ g_2 \in G$	(closed operation)
(i)	$\exists e \in G \quad \forall g \in G \quad e \circ g = e = g \circ e$	(identity)
(ii)	$\forall g \in G \quad \exists g^{-1} \in G \quad gg^{-1} = e$	(inverse)
(iii)	$\forall g_1, g_2, g_3 \in G \quad g_1(g_2g_3) = (g_1g_2)g_3$	(associativity)

All of the groups in this thesis will be formed from matrices and matrix multiplication. The most important thing to verify is property (0). Property (i) will be straightforward, we just have to check that the identity matrix Id is contained in the group. Property (ii) will usually be proved by showing that a matrix is invertible; it is also important to verify that the inverse is contained in the group. Property (iii) will generally hold automatically because matrix multiplication is associative.

Rigid transformations can be decomposed into a rotation $\mathbf{R}$ and a translation $\mathbf{t}$ because:

$$\begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}\begin{bmatrix} \mathrm{Id} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{Rt} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathrm{Id} & \mathbf{Rt} \\ \mathbf{0} & 1 \end{bmatrix}\begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$$

so any sequence of rotations and translations can be rearranged so that all rotations are on one side, and all translations are on the other. It follows from this that products of rigid transformations are rigid transformations, verifying property (0). The identity matrix is obviously a rigid transformation, so (i) holds. The inverse of a rigid transformation with rotation component $\mathbf{R}$ and translation component $\mathbf{t}$ is the rigid transformation with rotation component $\mathbf{R}^\top$ and translation component $-\mathbf{t}$, so (ii) holds. Hence the rigid transformations are a group. They are also called the Euclidean group.

While groups model transformations or changes to scenes or images, the scenes or images are themselves modeled by sets of matrices. The method by which the transformations in a group actually work to change the sets is formalised by the notion of a group action.

Definition 4.1
If G is a group, and M is a set then a group action is a function $\theta : G \times M \to M$ with the properties:
(i) $\theta(e, x) = x$ for all $x \in M$ (identity)
(ii) $\theta(g_1, \theta(g_2, x)) = \theta(g_1 g_2, x)$ (associativity)
If the function θ is continuous, the group action is called continuous.

In most cases the group action is nothing other than matrix multiplication, but in the case of the perspective camera model we come upon something so complicated that we have to refer back to a formal definition to stop ourselves from becoming totally confused!

An invariant to a group action is just a function whose value is unchanged by the group action. The more precise definition is as follows.

Definition 4.2
An invariant of a group G acting on a set M via group action θ is a function f on M such that

$$f(\theta(g, x)) = f(x) \quad \forall x \in M \, \forall g \in G$$

The points in M that can be mapped to each other by the group action θ can be considered to be equivalent. More precisely $x \sim y$ iff $\exists g \in G$ such that $\theta(g, x) = y$. The equivalence classes are called orbits of G under the group action θ. An invariant is then seen to be a function that is constant on the orbits of G.

The Euclidean group acting on a plane parallel to the retinal plane induces the Euclidean group on the retinal plane. Translations along the optical axis make the image larger or smaller. Their induced transformation is multiplication by a positive non-zero scalar. Translations along the optical axis together with the rigid transformations parallel to the retinal plane induce the group of Euclidean similarities which is generated by rotations, translations and scaling by a positive non-zero factor.

The same holds true for any perspective matrix such that $\alpha_u = \alpha_v$. A perspective matrix can be factored into

$$\begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \end{bmatrix}$$

The left hand factor contains the intrinsic parameters, and the right hand factor contains the extrinsic parameters. The extrinsic parameters will only relocate the retinal plane, so without loss of generality we may take

$$\mathbf{P} = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Then assuming $\alpha_u = \alpha_v$,

$$\begin{bmatrix} R_{11} & R_{12} & \alpha_u t_1 + u_0 - R_{11}u_0 - R_{12}v_0 \\ R_{21} & R_{22} & \alpha_u t_2 + v_0 - R_{21}u_0 - R_{22}v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} R_{11} & R_{12} & t_1 & 0 \\ R_{21} & R_{22} & t_2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

Thus for any perspective matrix satisfying $\alpha_u = \alpha_v$ the rotations parallel to the retinal plane and the translations (of space) induce the group of Euclidean similarities on the retinal plane.

Rigid transformations in a general plane induce projective transformations on the retinal plane. This fact is well-known [132]. However, there is a good reason why we do not try to construct shape spaces using projective groups. We will see that the result is so pathological that no metric exists on the resulting topological space. There is no reasonable and completely general way to compare the almost limitless spring of projective invariants being brought forth by the computer vision literature. One is forced to introduce a series of context dependent rules to distinguish between objects. This is not an inherent limitation of the perspective camera model. It is a severe limitation of projective geometry, that can be overcome by other geometric models.

The above induced groups only apply when the object is planar, like a floor or a wall. The general form of the problem is that we have a map ι, a group any element of which is denoted g and a set of points whose typical member is x, and we would like to construct a map

$$\iota(M) \to \iota(M) \quad \iota(x) \mapsto \iota(gx)$$

for each g, which is the induced transformation. In the case above, ι is a perspective map such that $\alpha_u = \alpha_v$, g is a rotation parallel to the retinal plane together with a space translation and x is a point in the scene. But, in general a map ι does not induce a well-defined map $\iota(x) \mapsto \iota(gx)$ since there may exist x, y and g such that $x \neq y$, $\iota(x) = \iota(y)$ and $\iota(gx) \neq \iota(gy)$.

Example 4.1
Let ι be the map represented by

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let

$$g = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$x = [1 \quad 1 \quad 1 \quad 1]^\top$, $y = [1 \quad 1 \quad 2 \quad 1]^\top$. Then $\iota(x) = [1 \quad 1 \quad 1]^\top = \iota(y)$ but $\iota(gx) = [2 \quad 1 \quad 1]^\top \neq [3 \quad 1 \quad 1]^\top = \iota(gy)$. That is, the map induced by the affine group acting on the scene under orthographic projection is not well-defined. □

However, we can show there is always a subgroup which is well-defined. A subgroup is a subset of the group that satisfies all the properties of a group, with the same binary operation.

Lemma 4.1
Let G be a group acting on a set M, and ι a map on M. There is a subgroup G_ι of G whose induced action on $\iota(M)$ is well-defined.

Proof
For any $x, y \in M$ we define $G_{x,y} = G$ if $x = y$ or $\iota(x) \neq \iota(y)$, and
$G_{x,y} = \{g \in G \mid \iota(gx) = \iota(gy)\}$ if $x \neq y$ and $\iota(x) = \iota(y)$.

Let $G_\iota = \cap_{x,y \in M} G_{x,y}$.
Recall that e denotes the identity element of a group.
$\forall x, y \in M\,, e \in G_{x,y} \Rightarrow e \in G_\iota$ and $G_\iota \neq \emptyset$

Suppose $g_1, g_2 \in G_\iota$. We show $g_1 g_2 \in G_\iota$: Let $x, y \in M$.
If $x \neq y$ and $\iota(x) = \iota(y)$ then let $x' = g_2 x$, $y' = g_2 y$. Then $\iota(x') = \iota(y')$ since $g_2 \in G_\iota$, that is $\iota(g_2 x) = \iota(g_2 y)$. Since $g_1 \in G_\iota$, $\iota(g_1 x') = \iota(g_1 y')$, so $\iota(g_1 g_2 x) = \iota(g_1 g_2 y)$.
If $x = y$ or $\iota(x) \neq \iota(y)$ then $g_1 g_2 \in G_{x,y}$.
So for any $x, y \in M$, $g_1 g_2 \in G_{x,y}$, so $g_1 g_2 \in G_\iota$. ∎

In principle this subgroup could be trivial, but in important examples it is not.

Example 4.2
Consider the subgroup of the affine group whose elements satisfy $G_{13} = 0$ and $G_{23} = 0$.
It is a sub-group because

$$\begin{bmatrix} G_{11} & G_{12} & 0 & t_x \\ G_{21} & G_{22} & 0 & t_y \\ G_{31} & G_{32} & G_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} H_{11} & H_{12} & 0 & s_x \\ H_{21} & H_{22} & 0 & s_y \\ H_{31} & H_{32} & H_{33} & s_z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} G_{11}H_{11} + G_{12}H_{21} & G_{11}H_{12} + G_{12}H_{22} & 0 & G_{11}s_x + G_{12}s_y + t_x \\ G_{21}H_{11} + G_{22}H_{21} & G_{21}H_{12} + G_{22}H_{22} & 0 & G_{21}s_x + G_{22}s_y + t_y \\ G_{31}H_{11} + G_{32}H_{21} + G_{33}H_{31} & G_{31}H_{12} + G_{32}H_{22} + G_{33}H_{32} & G_{33}H_{33} & G_{31}s_x + G_{32}s_y + G_{33}s_z + t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which also has zero entries in rows one and two of column three. The induced group of this group under $\iota : (x, y, z) \mapsto (x, y)$ is the affine group of the retinal plane, which acts in a well-defined way. □

Now consider the sub-group of the affine group where $\det(G) > 0$, which will be called the positive affine group. Consider the sub-group of the positive affine group satisfying $G_{11}G_{22} - G_{12}G_{21} > 0$ and $G_{13} = G_{23} = 0$. This sub-group induces the positive affine group on the retinal plane, where ι is an orthogonal projection along the z-axis onto the xy-plane.

Knowing the above induced sub-groups on the image plane turns out to be sufficient for modelling the affine camera. However, an analogous calculation for the perspective camera fails, because the induced sub-groups on the image plane have too low a dimension to be useful.

4.2 $\iota(gX) = g\iota(X)d$

In what follows, $GL(m)$ denotes the set of non-singular $m \times m$ matrices, Id denotes the identity matrix, $diag()$ denotes a diagonal matrix whose diagonal entries are listed inside the brackets, and $M_{m,n}$ denotes the set of $m \times n$ matrices.

Let M be the set of $4 \times n$ matrices which have at least one non-zero four by four minor and no entry of the fourth row zero. A matrix in M that has all entries of its fourth row equal to one represents $n \geq 4$ points in $\mathbf{R}^3$ with at least four points not coplanar. Such a matrix represents feature points in three dimensions, not image points.

Let G be the set of (g, d) such that $g \in GL(4)$ with $g_{41} = g_{42} = g_{43} = 0$ and $g_{44} = 1$, and d is $n \times n$ diagonal and non-singular where $n \geq 4$. G is a group[1] under the product operation $(g_1, d_1) \circ (g_2, d_2) = (g_1 g_2, d_1 d_2)$.

Lemma 4.2
If $A \in M$, $(g, d) \in G$ then $gAd = A$ implies $g = \text{Id}$ and $d = \text{Id}$.

Proof
$A \in M$ is a $4 \times n$ matrix, with no entry of the fourth row zero, and some non-singular 4×4 minor, where $i = (i_1, i_2, i_3, i_4)$.

Expanding out the fourth row of gAd, it equals $(A_{41}d_{11}, \ldots, A_{4n}d_{nn})$. Since $gAd = A$,

$$(A_{41}d_{11}, \ldots, A_{4n}d_{nn}) = (A_{41}, \ldots, A_{4n})$$

This in turn implies $d_{11}, \ldots, d_{nn} = 1$ because all of $A_{41}, \ldots, A_{4n} \neq 0$.

Consider the 4×4 non-singular matrix A_i formed from four columns of A with rank 4. Since $gAd = gA$, $gA_i = A_i$ also. Since A_i is invertible, $g = \text{Id}$. ∎

Let $\tilde{M}$ be the subset of M whose elements (are matrices) with no entry of their third row equal to zero. Let ι denote the map from $\tilde{M}$ to $M_{4,n}$ which maps each column $[x \quad y \quad z \quad t]^\top$ to $[x/z \quad y/z \quad 1 \quad t/z]^\top$. ι represents perspective image formation. The fact that no entry of the third row is zero means that no point in the configuration represented by a matrix in $\tilde{M}$ lies in the focal plane.

Theorem 4.3
If $g \in GL(4)$ with $g_{41} = g_{42} = g_{43} = 0$ and $g_{44} = 1$, $X \in \tilde{M}$, $gX \in \tilde{M}$, then there is a unique non-singular $n \times n$ diagonal matrix d such that

$$g\iota(X)d = \iota(gX)$$

Proof
Let

$$g = \begin{bmatrix} g_{11} & g_{12} & g_{13} & g_{14} \\ g_{21} & g_{22} & g_{23} & g_{24} \\ g_{31} & g_{32} & g_{33} & g_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[1] G is in fact a Lie group

be an affine transformation.

$$X = \begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \\ z_1 & \dots & z_n \\ t_1 & \dots & t_n \end{bmatrix} \in \tilde{M}$$

so $z_1, \dots, z_n \neq 0$, and X is not contained in a plane.

$$g\iota(X) = \begin{bmatrix} (g_{11}x_1 + g_{12}y_1 + g_{13}z_1 + g_{14}t_1)/z_1 & \dots & (g_{11}x_n + g_{12}y_n + g_{13}z_n + g_{14}t_n)/z_n \\ (g_{21}x_1 + g_{22}y_1 + g_{23}z_1 + g_{24}t_1)/z_1 & \dots & (g_{21}x_n + g_{22}y_n + g_{23}z_n + g_{24}t_n)/z_n \\ (g_{31}x_1 + g_{32}y_1 + g_{33}z_1 + g_{34}t_1)/z_1 & \dots & (g_{31}x_n + g_{32}y_n + g_{33}z_n + g_{34}t_n)/z_n \\ t_1/z_1 & \dots & t_n/z_n \end{bmatrix} \tag{4.1}$$

$$\iota(gX) = \begin{bmatrix} \frac{g_{11}x_1+g_{12}y_1+g_{13}z_1+g_{14}t_1}{g_{31}x_1+g_{32}y_1+g_{33}z_1+g_{34}t_1} & \dots & \frac{g_{11}x_n+g_{12}y_n+g_{13}z_n+g_{14}t_n}{g_{31}x_n+g_{32}y_n+g_{33}z_n+g_{34}t_n} \\ \frac{g_{21}x_1+g_{22}y_1+g_{23}z_1+g_{24}t_1}{g_{31}x_1+g_{32}y_1+g_{33}z_1+g_{34}t_1} & \dots & \frac{g_{21}x_n+g_{22}y_n+g_{23}z_n+g_{24}t_n}{g_{31}x_n+g_{32}y_n+g_{33}z_n+g_{34}t_n} \\ 1 & \dots & 1 \\ \frac{t_1}{g_{31}x_1+g_{32}y_1+g_{33}z_1+g_{34}t_1} & \dots & \frac{t_n}{g_{31}x_n+g_{32}y_n+g_{33}z_n+g_{34}t_n} \end{bmatrix} \tag{4.2}$$

Let

$$d = diag(z_1/(g_{31}x_1 + g_{32}y_1 + g_{33}z_1 + g_{34}t_1), \dots, z_n/(g_{31}x_n + g_{32}y_n + g_{33}z_n + g_{34}t_n))$$

Since the third rows of X and gX have no non-zero entries, d is non-singular. Multiplying the right hand side of equation 4.1 by d we see that it equals the right hand side of equation 4.2. Thus $g\iota(X)d = \iota(gX)$ which proves existence.

Suppose $(g', d') \in G$ such that $g'\iota(X)d' = \iota(gX)$. Then $g'\iota(X)d' = \iota(gX) = g\iota(X)d$. Unless $g' = g$ and $d' = d$, $(g'g^{-1}, d'd^{-1})$ fixes $\iota(X)$. $\iota(X) = X\lambda$ for some $\lambda \in diag(M_{n,n})$. Since $X \in \tilde{M}$ its third row has no zero entries, so λ is non-singular, so $X\lambda \in M$, ie $\iota(X) \in M$. By lemma 4.2, $g' = g$ and $d' = d$.
Thus g and d are unique. ∎

Not all diagonal matrices can be realised by a change of viewpoint and camera parameters; those that can be realised depend on the scene being viewed. If we define a group action on $\iota(\tilde{M})$ by $\theta((g, d), \iota(X)) = g\iota(X)d$, then we can calculate its invariants. However, even if we calculate a complete set of invariants, a vector of these invariants will be constant on a much larger set than the set of images $\iota(gX)$ taken of the same scene X with all possible camera parameters g. In other words, many scenes that have distinct images would be deemed equivalent. If we do not define a group action, we cannot calculate any invariants. What we can do is restrict this group action to a subset of $G \times \iota(\tilde{M})$, in such a way that an equivalence relation results. We will see in chapter 5 that we can construct a shape space without using the concept of invariance.

The equivalence classes to be defined in chapter 5 will be contained in orbits of the group G, which are larger than the equivalence classes. The orbits of the group G are

in turn contained in the orbits of the projective group, which are still larger. This course of action has generally been taken in machine vision, but we will see in the next section that it has some very serious repercussions. By contrast, it will turn out that the restricted group action does not behave in a pathological manner, precisely because of the properties enunciated by theorem 4.3 and lemma 4.2.

Theorem 4.3 is useful despite the four dimensional representation, because the first three rows give equations in scene or image coordinates (see equation 9.2).

Remarks

- G is not well-defined if the fourth coordinate is varied, whereas it is well-defined in homogeneous coordinates; ι lacks the intended meaning in homogeneous coordinates (see example 2.8).
- if $(g, d) \in G$ and $X \in M$ then $gXd \in M$; $X \in \tilde{M}$ does not imply gXd is in $\tilde{M}$.
- We can define a different group G from the same set G (with a different product[2]) to act on M by $A \mapsto (gA + [w, \ldots, w])d$ where $w \in \mathbf{R}^3$. With the group we defined as G, $gXd = X \Rightarrow g = \mathrm{Id}, \mathrm{d} = \mathrm{Id}$ and the equation $\iota(gX)d = g\iota(X)$ holds for suitable X. Neither property holds for the alternative group action. The two actions only behave the same prior to applying the map ι.

4.3 Drawbacks of projective invariants

This section is based on some concepts from differential geometry outlined in [59]. We outline the drawbacks of some alternative formulations of the problem of calculating induced groups. One of these alternatives is equivalent to the group whose invariants are called three dimensional projective invariants of n points in the vision literature [31], [40],[126],[125], [153],[154] (by no means an exhaustive list). It has already been shown by Astrom that there are very distinctively shaped curves which can be mapped arbitrarily close to a circle by projective transformations [4], such as silhouettes of a duck and a rabbit. Here we outline some drawbacks of projective invariants of sets of finitely many points. The reader may recall that not all projective transformations can actually result from changes in viewpoint, focal length, and other camera parameters. Thus vectors of projective invariants will have the same value on many more scenes than does the restricted group action. In other words, much information about the scene is lost in a projectively invariant representation. We will show that there is no satisfactory relation between distances in Euclidean space representing scenes, and errors in spaces generated by projective invariants. This is established by proving that no metric exists on a quotient space by the projective group.

$PGL(n)$ denotes the group of $n \times n$ non-singular matrices up to scale. That is, matrices that are scalar multiples of each other are identified with the same element of $PGL(n)$. The projective group $PGL(4)$ acting on 3D configurations of points does not induce the projective group $PGL(3)$ on the image. This is shown by the next two examples.

[2]This was pointed out to me by a referee of [19].

Example 4.3
Let

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad g = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in GL(4)$$

Consider the 3 point configuration $(1,0,0),(0,1,0),(0,0,1)$.
Then

$$\mathbf{P}g \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -1 & 0 \end{bmatrix} \notin GL(3)$$

$$\mathbf{P} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \in GL(3)$$

Thus there cannot exist an invertible mapping which maps one to the other. □

Example 4.4

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \quad g = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in GL(4)$$

Consider the 3 point configuration $(1,0,0),(0,1,0),(0.5,0.5,0.5)$.
Then

$$\mathbf{P}g \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.5 & 0.5 & 0.5 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \notin GL(3)$$

$$\mathbf{P} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.5 & 0.5 & 0.5 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1.5 & 1.5 & 1.5 \end{bmatrix} \in GL(3)$$

Thus there cannot exist an invertible mapping which maps one to the other. □

Suppose we consider the group $GL(4) \times diag(GL(n))$ acting on n points in $\mathbf{R}^4 \setminus \{0\}$. G is a subgroup of this group. The diagonal non-singular matrix varies each homogeneous representation independently. This group action is equivalent to the action of $PGL(4)$ on $(\mathbf{P}^3)^n$. In the vision literature, the invariants to this group are called three dimensional projective invariants of n point configurations. A group H is said to act freely on a space X if for all $x \in X$, and any $h \in H$, $hx = x$ implies $h = e$. In other words, the group action has no fixed points except the identity element of H. The action of $GL(4) \times diag(GL(n))$ is not free, unlike the action of G (which is free by lemma 4.2):

Example 4.5

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 & 1 & 5 \end{bmatrix}$$

A is in general position. Let $g = diag(\alpha, \beta, \gamma, \delta)$, $d = diag(\alpha^{-1}, \beta^{-1}, \gamma^{-1}, \delta^{-1}, 1, 1)$. Then $gAd = A$. The group action has a four dimensional stabiliser at A. A belongs to a 6 parameter family of orbits[3] of $GL(4) \times diag(GL(6))$ parameterised by

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & \alpha & \delta \\ 0 & 0 & 1 & 0 & \beta & \epsilon \\ 0 & 0 & 0 & 1 & \gamma & \lambda \end{bmatrix}$$

because any non-singular 4×4 minor can be mapped to the identity by $g \in GL(4)$. □

A group acting on a space partitions that space into equivalence classes. The set of these equivalence classes can be regarded as a space in its own right, provided it is given a topology. Some extra condition is required to ensure that there is some meaningful relation between the topology of this space of equivalence classes and the original space. Let M be a topological space, and $\sim$ an equivalence relation on M. The set of equivalence classes is denoted by $M/\sim$. There is a natural mapping π from any $x \in M$ to $M/\sim$. Our condition is that open sets in $M/\sim$ can be mapped back to open sets in M. This is a very basic condition, without which there would not be any relation between perturbations of equivalence classes and distances in the image or scene. Any quotient $M/\sim$ can be given a topology that satisfies this condition. With this topology, $M/\sim$ is called a quotient space.

We will prove that no metric exists on the quotient space of the projective group. So although we can construct projective invariants for various structures, we cannot distinguish between perturbed versions of these structures and arbitrary new structures!

Every topological space that has a metric also satisfies a weak condition called the Hausdorff property. We will show that the quotient space of the set of four point configurations by the projective group is not Hausdorff. This will prove that no metric can exist on the quotient space.

Definition 4.3
A topological space is called a Hausdorff space if for every pair of distinct points x, y there are disjoint open sets containing x and y respectively.

The Hausdorff condition just says that every pair of points can be separated from each other by open sets. It is not hard to see that any space with a metric is Hausdorff: for each $x \neq y$ the sets $\{p \mid d(x,p) < d(x,y)/2\}$ and $\{p \mid d(y,p) < d(x,y)/2\}$ are disjoint open sets containing x and y respectively.

All of the groups we have encountered so far are themselves manifolds. Not only this, but the operations of multiplication and inversion in the groups have been smooth.

[3]This was pointed out to me by Michael Murray

Such groups are called Lie groups.

Definition 4.4
A Lie group H is said to act properly on a manifold M if and only if, for all compact subsets K contained in M, $H_K = \{g \in H \mid gK \cap K \neq \emptyset\}$ is relatively compact in H.

Theorem 4.4
If H is a Lie group acting continuously on a manifold M, then M/H is Hausdorff if and only if H acts properly on M.

Proof

Suppose H acts properly on M but M/H is not Hausdorff. Consider $x, y \in M$, and $\{gy \mid g \in H\} \subset M \setminus \{x\}$. If there do not exist open sets separating them, then x is in the closure of $\{gy \mid g \in H\}$. So there must be a sequence $g_i y$ such that x is a limit point of the sequence. The set $K = \{x\} \cup \{g_i y \mid i = 1 \ldots \infty\}$ is compact. Since H acts properly $\{g \in H \mid gK \cap K \neq \emptyset\}$ is also compact. The sequence g_i has no limit, because if it had a limit g, from continuity of the group action and $g_i y \to x$ would imply $gy = x$, contradicting the disjointness of orbits of x and y. Thus the sequence g_i is not closed in H. This contradicts the compactness of H_K, so M/H must be Hausdorff.

Now suppose M/H is Hausdorff. Suppose also that K is a compact subset of M. Since M/H is Hausdorff, if $x, y \in K$ belong to different orbits then there are open sets U, V containing x, y respectively, such that $U \cap gV = \emptyset$ for all $g \in H$. Thus for each subset of K belonging to a single orbit, we obtain an open set covering it that is disjoint from all other such open sets. In other words we obtain an open cover of K with disjoint members. By definition of compactness, this has a finite subcover, which must be the same cover, because the members were disjoint. Hence K consists of a finite number of disjoint subsets K_i each belonging to a single orbit of H. Each K_i must be compact, otherwise the compactness of K would be contradicted. Since $H_K = \{g \in H \mid gK \cap K \neq \emptyset\}$ is a finite union of an intersection of compact sets, it is compact. It follows that H acts properly on M. ∎

Lemma 4.5
$GL(m) \times diag(GL(n))$ does not act properly on $(\mathbf{R}^m \setminus \{0\})^n$

Proof

Let $K = O(m) \times \{p\} \subset M_{m,n}$.
Take any $g \in GL(m)$. It has a singular value decomposition $u_g d_g v_g^\top$ with $d_g \in diag(GL(m))$.
Take $k = [v_g \quad p]$, $v_g \in O(m)$, $d \in diag(d_g^{-1}, 1, \ldots, 1)$.
Then $gkd = u_g d_g v_g^\top [v_g \quad p]\, d = [u_g \quad g_p]$.
So if p is a unit eigenvector of g, $gkd \in O(m) \times \{p\}$.
Hence H_K contains the subset of $GL(m)$ fixing p. This subset is $GL(m)$ if p is the $m \times (n-m)$ matrix with all entries zero, which is non-compact. ∎

Since $GL(m) \times diag(GL(n))$ acts continuously but not properly on M, $M/(GL(m) \times diag(GL(n)))$ is not Hausdorff. Consequently no metric exists on $M/(GL(m) \times$

$diag(GL(n)))$.

The general question of when a group action on algebraic objects is stable is a deep subject called geometric invariant theory. Chapter 3 in Mumford and Fogarty [130] is devoted to the problem of decomposing the quotient of a product of projective spaces by the projective group so that the quotient is stable. However, this decomposition is much more restrictive than the construction to be developed in chapter 5, which requires only that the points in the scene are not all coplanar. In chapter 9, we will see that even a very weak criterion for the stability of a metric reconstruction is compromised by the projective model.

Chapter 5

Construction of shape spaces

In chapter 4 we studied the problem of how changing camera parameters induce changes in images, from the viewpoint of group theory. We discovered that this runs into difficulty with the general perspective camera model. In this chapter, we introduce a construction known as a shape space to overcome those difficulties. The main motivations for this construction are: to avoid treating images of distinct scenes as if they were images of the same scene; to avoid bias towards certain camera configurations in measurements of similarity; to minimise instability in reconstruction. Once again, we will proceed from special or limiting cases of the perspective camera model and scene, to the general perspective camera model. We will establish that the most general case in which a metrisable shape space can be constructed for the perspective camera is the case of non-coplanar tuples of scene points, none of which lie in the focal plane of the camera, where the camera has arbitrary parameters. So for (entirely) coplanar scene points we have to restrict the viewpoint.

If we restrict the viewpoint to be perpendicular to the plane being viewed, and restrict the aspect ratio to unity, the Euclidean shape spaces are an appropriate model. We introduce the Euclidean shape spaces first. These are not only a useful special case, but a prototype of the kinds of properties that are desirable in a shape space. We then study the limiting case where the scene is infinitely far away, for both planar and non-coplanar scenes. This is based on the affine camera model. If no distinction is made between clockwise and anti-clockwise, an affine shape space is constructed. If such a distinction is made, then a positive affine shape space is constructed. Finally, we construct shape spaces for non-coplanar scenes, and arbitrary perspective cameras.

From lemma 4.5, no metric exists on the projective space $\mathbf{P}^2$. Since general changes of perspective viewpoint of coplanar points induce two dimensional projective transformations on the image, this means there is no metric on the spaces of coplanar scenes and arbitrary perspective cameras. This justifies the inclusion of the special cases we shall study first.

The main reason that we study shape spaces rather than individual shapes is that many important properties of algorithms are properties of shape spaces. They are not properties of individual equivalence classes or invariant vectors. The term shape space was originated by Kendall, in his study of what is now called a Euclidean shape

space [99]. However, that paper assumes a considerable knowledge of topology [133], differential geometry [59], [179], [37] and algebraic geometry [74]. Our treatment here is elementary. The term shape was used by Sparr in the study of several problems in computer vision [162], [161], [163], [165], [166]; it will be more specifically denoted affine shape in this thesis. The author identified the affine shape spaces, and the positive affine shape spaces in [14] and [15]. There is an obvious generalisation of the various definitions of shape in [99], [162] and [14], and it is widely used [60], [68], [65]. The author will also construct and study specialized spaces within the positive affine shape spaces in chapter 6.

5.1 Translation shape

The group of translations on $\mathbf{R}^m$ can transform any point to any other point. However it cannot transform any m-tuple of points into all other m-tuples of points. The information that remains unchanged under the influence of translations is captured by the notion of a translation shape. Moreover, if there is any change other than a translation, the translation shapes must be different. This is important to avoid treating images of distinct scenes as if they were images of the same scene. We represent images by matrices, whose columns represent individual points. Formulating the problem in terms of matrices makes sense because a camera has a finite number of sensors, so images consist of a discrete set of points. We would like to quantitatively compare two k-tuples of points to determine how similar they are, while ignoring their location. Such a function must not change in value if one of the k-tuples is translated. If it does, it will be biased against this parameter. It will transpire that such biases can cause optimisation algorithms to diverge from the correct solution to certain problems (see chapter 9). In fact, we will produce a metric on the quotient space by translations. That the metric generates the quotient topology is equivalent to asking that this metric is continuous as a function of the Euclidean metric. This is a basic prerequisite for stability in reconstruction (see chapter 9).

The term shape space was originated by Kendall in his study of what is now called a Euclidean shape space [99]. The construction of translation shape is based on [99], but explicit proofs are given here. The approach we will take may seem unduly complicated for such a simple problem. However, this section should be seen as a step towards the solution of a far more difficult problem. Our aim is to provide machinery to determine whether or not algorithms in chapter 9 have suitable convergence and stability properties.

Definition 5.1
If X, Y are $m \times n$ matrices we define $\|X\| = \sqrt{\sum_{i=1,j=1}^{m,n} X_{ij}}$ and $d(X,Y) = \|X - Y\|$.

Definition 5.2

Let Q be the $k \times k$ Helmert matrix (from D.G. Kendall [99])

$$Q_{i(j+1)} = \begin{cases} 1/\sqrt{k} & \text{if } j = 0 \\ -1/\sqrt{j^2+j} & \text{if } 1 \leq j \leq k-1 \text{ and } 1 \leq i \leq j \\ j/\sqrt{j^2+j} & \text{if } 1 \leq j \leq k-1 \text{ and } i = j+1 \\ 0 & \text{otherwise} \end{cases}$$

$\tilde{Q}$ will denote the $k \times (k-1)$ matrix obtained by deleting the first column of Q.

By multiplying an $m \times k$ matrix by Q on the right, we produce a second $m \times k$ matrix whose first column is the centroid of the k points given by the k columns of the first matrix. It will turn out that dropping the first column of the matrix so produced will lose no information except the location of k points. The locations of the points relative to each other will not be lost.

Example 5.1

$$m = 2, k = 3, Q = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & 0 & \frac{\sqrt{2}}{\sqrt{3}} \end{bmatrix}, \tilde{Q} = \begin{bmatrix} \frac{-1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} \\ 0 & \frac{\sqrt{2}}{\sqrt{3}} \end{bmatrix}, W = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

is a configuration of 3 points in the plane. To normalise against translation, we calculate $W\tilde{Q}$. $W\tilde{Q}$ is the same as WQ without its first column. □

Explicitly multiplying by Q is an algorithm of order mk^2, however there is a closed form expression for this mapping [99]: Denote the k points by $z_1^*, \ldots, z_k^*$, and their centroid by z_c^*.

$$z_0 = z_c^* \sqrt{k} \tag{5.1}$$

$$z_j = \frac{j z_{j+1}^* - (z_1^* + \ldots + z_j^*)}{\sqrt{j^2+j}} \quad 1 \leq j \leq k-1 \tag{5.2}$$

$z_1, \ldots, z_k$ are the columns of the shape transform of the configuration $z_1^*, \ldots, z_k^*$. This can be written as an algorithm of linear complexity. Furthermore, since Q is orthogonal this algorithm does not change the condition number of the matrix it acts on [169], and can be expected to be numerically stable.

Theorem 5.1
Let M be the set of $m \times k$ matrices with *not* all of their columns identical. Let G be the set of $m \times k$ matrices with *all* of their columns identical. Let

$$\pi_1 : M \to M_{m,k-1} \setminus \{0\}, \; X \mapsto X\tilde{Q}$$

π_1 is surjective[1] with the following three properties:

(i) $\forall g \in G \quad \pi_1(X+g) = \pi_1(X)$ (invariance);
(ii) $\pi_1(X) = \pi_1(Y) \Rightarrow \exists g \in G$ such that $X = Y + g$ (complete invariance);
(iii) $\|\pi_1(X) - \pi_1(Y)\| = \inf_{g \in G} \|X - (Y+g)\|$ (least squares).

[1] which means $\pi(M) = M_{m,k-1} \setminus \{0\}$

The quotient space M/G is $M_{m,k-1} \setminus \{0\}$, and $d(\pi_1(X), \pi_1(Y))$ is a metric in the quotient topology.

Proof

We prove π_1 satisfies each property in turn:
(i) $(1,\ldots,1)Q = (\sqrt{k}, 0, \ldots, 0)$ since Q is orthogonal and its first column is $(1,\ldots,1)/\sqrt{k}$.

$$\Rightarrow \begin{bmatrix} t_1 & \ldots & t_1 \\ \vdots & & \vdots \\ t_m & \ldots & t_m \end{bmatrix} Q = \begin{bmatrix} \sqrt{k}t_1 & 0 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ \sqrt{k}t_m & 0 & \ldots & 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} t_1 & \ldots & t_1 \\ \vdots & & \vdots \\ t_m & \ldots & t_m \end{bmatrix} \tilde{Q} = \mathbf{0}$$

$$\Rightarrow X\tilde{Q} = (X + \begin{bmatrix} t_1 & \ldots & t_1 \\ \vdots & & \vdots \\ t_m & \ldots & t_m \end{bmatrix})\tilde{Q}$$

Thus $\pi_1(X) = \pi_1(X + T)$ for any $X \in M, T \in G$.

(ii) Suppose $\pi_1(X) = \pi_1(Y)$.
$\Rightarrow X\tilde{Q} = Y\tilde{Q}$
$\Rightarrow (X - Y)\tilde{Q} = \mathbf{0}$

$$\Rightarrow (X - Y)Q = \begin{bmatrix} t_1 & 0 & \ldots & 0 \\ \vdots & & & \vdots \\ t_m & 0 & \ldots & 0 \end{bmatrix}$$

$(\sqrt{k}, 0, \ldots, 0) = (1,\ldots,1)Q$ by definition of Q

$$\Rightarrow \begin{bmatrix} \sqrt{k}t_1 & 0 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ \sqrt{k}t_m & 0 & \ldots & 0 \end{bmatrix} = \begin{bmatrix} t_1 & \ldots & t_1 \\ \vdots & & \vdots \\ t_m & \ldots & t_m \end{bmatrix} Q$$

Thus $X - Y \in G$.

(iii) First we show that $\|\pi_1(X) - \pi_1(Y)\| \leq \|X - Y - T\|$ for all $T \in G$:

$$\begin{aligned} \|\pi_1(X) - \pi_1(Y)\| &= \|(X - Y)\tilde{Q}\| \\ &= \|(X - Y - T)\tilde{Q}\| && \text{from (i)} \\ &\leq \|(X - Y - T)Q\| && \text{since } \|AQ\|^2 = \|A\tilde{Q}\|^2 + k\|A_c\|^2 \text{ where} \\ & && A_c \text{ denotes the centroid of } A \in M \\ &= \|X - Y - T\| && \text{since } Q \text{ is orthogonal} \end{aligned}$$

Now we show that setting T to the $m \times k$ matrix whose columns are all $X_c - Y_c$ gives $\|\pi_1(X) - \pi_1(Y)\| = \|X - Y - T\|$:
$\|X\tilde{Q}\|^2 = \|X\|^2 - k\|X_c\|^2$ since the left hand side matrix is all but the first column of XQ, which is the centroid of X times $\sqrt{k}$.
We show $\|X\|^2 - k\|X_c\|^2 = \|X - X_c\|^2$:

$$\begin{aligned} (x_{i1} - x_{ci})^2 + \ldots + (x_{ik} - x_{ci})^2 &= x_{i1}^2 + \ldots + x_{ik}^2 + kx_{ci}^2 - 2x_{ci}(x_{i1} + \ldots + x_{ik}) \\ &= x_{i1}^2 + \ldots + x_{ik}^2 - kx_{ci}^2 \end{aligned}$$

Hence $\|X\|^2 - k\|X_c\|^2 = \|X - X_c\|^2$.
Thus $\|\pi_1(X)\| = \|X - X_c\|$
Property (iii) follows from this.

We prove that the range of π_1 is in fact $M_{m,k-1} \setminus \{0\}$:
Suppose $m \in M_{m,k-1} \setminus \{0\}$.
Let $A = m\tilde{Q}^\top$.

$$\tilde{Q}^\top \tilde{Q} = \mathrm{Id} \tag{5.3}$$

$\Rightarrow A\tilde{Q} = m\tilde{Q}^\top\tilde{Q} = m$ from equation 5.3.
And $A \in G$ iff $\pi_1(A) = \mathbf{0}$.
Hence $\pi_1(M) = M_{m,k-1} \setminus \{0\}$, and from (i) and (ii) it follows that there is a one to one correspondence between M/G and $M_{m,k-1} \setminus \{0\}$. Since π_1 is continuous, open sets in $M_{m,k-1} \setminus \{0\}$ are the images of open sets in M under π_1. Consequently, the natural norm on $M_{m,k-1} \setminus \{0\}$ is a quotient metric, so that $M/G = M_{m,k-1} \setminus \{0\}$. ∎

Remarks

- $\tilde{Q}\tilde{Q}^\top \neq \mathrm{Id}$ in contrast to equation 5.3.
- Q is not the only orthogonal matrix one could choose, but we have chosen this one, and will use it throughout.

The reader may wonder why we want to give the shape space a continuous structure. Points in the scene that have been photographed can move continuously. In so doing, their shapes will traverse a curve in shape space.

Example 5.2
Suppose we do not know the absolute positions of the configuration W from example 5.1 and the configurations

$$W' = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}, \quad W'' = \begin{bmatrix} 2 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix}$$

We can use theorem 5.1 to decide whether W and W' are the same except for a translation.

$$\pi_1(W) = \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{6} \\ 1/\sqrt{2} & 1/\sqrt{6} \end{bmatrix}$$

$$\pi_1(W') = \begin{bmatrix} -1/\sqrt{2} & -1/\sqrt{6} \\ 1/\sqrt{2} & -1/\sqrt{6} \end{bmatrix}$$

Since $\pi_1(W) \neq \pi_1(W')$, W and W' fail to satisfy property (i) in theorem 5.1. Consequently it is not possible to translate W to W' or vice versa; they differ by more than a translation.
On the other hand:

$$\pi_1(W'') = \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{6} \\ 1/\sqrt{2} & 1/\sqrt{6} \end{bmatrix}$$

Since $\pi_1(W) = \pi_1(W'')$ property (ii) of theorem 5.1 implies W can be translated to W''; these are the same except for a difference in position.
If the columns of W or W'' or both are perturbed slightly then strictly speaking we will not be able to translate one to the other. A quantitative measure of how much distortion of X and Y would be required before they could be translated into one another is provided by $\|\pi_1(X) - \pi_1(Y)\|$.

$$\|\pi_1(W) - \pi_1(W'')\| = 0$$

so they can be translated into each other with no distortion. □

Property (iii) of theorem 5.1 asserts that this metric on the range of the shape transform π_1 measures the least squares error between configurations after compensating for unknown translations.

Definition 5.3
π_1 will be called the translation shape transform.

We will now explain why we excluded $m \times k$ matrices all of whose columns are identical from M. π_1 maps such a matrix to zero. Whenever scaling is included as well as translation, the scaling generates half-lines through zero in $M_{m,k-1}$. If $\{0\}$ is a shape, then no metric exists on the shape space, because it is not Hausdorff (topological property, see [133]). Consequently we adopt the following definition:

Definition 5.4
A configuration of k points is a tuple of k points, not all identical, in $\mathbf{R}^m$. The set of all configurations of k points in $\mathbf{R}^m$ is called a configuration space.

Example 5.3
$((1,0),(0,1),(1,1))$ is a configuration in $\mathbf{R}^2$ the Euclidean plane. □

Example 5.4
$((1,1),(1,1))$ is not a configuration because all of its points are identical.□

Example 5.5
$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ is not the same configuration as $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ even though they both have the same points.□

This illustrates the difference between a (planar) configuration and a polygon : The points in a configuration are labelled $1, \ldots, k$. The points in a closed polygon can be relabelled by cyclically permuting labels; the order of labels can be reversed if the polygon is not oriented.

Owing to the many to one nature of a transform like π_1, it does not have a well-defined inverse. However, one can ask for an inverse of $s \in M_{m,k-1} \setminus \{0\}$ nearest to a given configuration X. This is furnished by first computing (the rather arbitrary) pseudo-inverse $m \mapsto m\tilde{Q}^\top$, and translating it so its centroid is equal to the centroid of X.

Consider the group consisting of translations, scaling and compositions of these transformations, and denote it by G. The next example demonstrates that the least squares error $\inf_{g\in G} d(x, gy)$ is not a metric (on M/G).

Example 5.6
Consider the point configurations $(0, -1/\sqrt{2}), (0, 1/\sqrt{2})$ and $(0, 0), (0, \sqrt{2})$.

$$Q = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

$$\pi_1(\begin{bmatrix} 0 & -1/\sqrt{2} \\ 0 & 1/\sqrt{2} \end{bmatrix}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$\pi_1(\begin{bmatrix} 0 & 0 \\ 0 & \sqrt{2} \end{bmatrix}) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\pi_1(M)$ is the plane minus the origin. If the first configuration is translated and scaled in all possible ways then π_1 maps this set of configurations to a half-line through $(0, 0)$ and $(0, 1)$. The least squares error between $(1, 1)$ and all the points on this half-line is 1. Similarly the least squares error between $(0, 1)$ and the half-line through $(0, 0)$ and $(1, 1)$ is $1/\sqrt{2}$. Thus $\inf_{g\in G} d(x, gy) \neq \inf_{g\in G} d(gx, y)$, so such a function cannot be a metric.□

On the other hand, the angle between two vectors is a scale invariant metric. An angle is the same thing as distance on a unit sphere. Shape metrics will be chosen to be least squares with respect to spherical distance.

5.2 Euclidean shape spaces

We will begin with what is probably the most important example of a shape space, because it is one of only three that can be visualised. We will construct a shape transform for 3 point configurations (with not all their points identical) to a sphere of radius $1/2$, with G the Euclidean similarity group.

The Euclidean similarity group consists of rotations, scaling (sometimes called dilatation) and translations. It does not contain any reflections, although rotations and translations can be expressed as a product of two reflections [168]. Rotations are represented by matrices R with the properties $RR^{\top} = \mathrm{Id} = \mathrm{R}^{\top}\mathrm{R}$ and $det(R) = 1$, called special orthogonal matrices[2]. The $m \times m$ special orthogonal matrices form a group called the special orthogonal group denoted $SO(m)$. See the book by Curtis for further details [41]. The rotations of the Euclidean plane also have a representation using complex numbers, which will be used heavily in this section.

Example 5.7
$((1, 0), (0, 1), (1, 1))$ is a configuration in $\mathbf{R}^2$ the Euclidean plane. It will either be

[2]An orthogonal matrix R can have $det(R) = -1$

represented by: $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$
or
$(1, i, 1+i) \in \mathbf{C}^3$ where $i = \sqrt{-1}$.□

Example 5.8
$\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 3 & 3 & 3 \\ 4 & 4 & 4 \end{bmatrix}$ is a Euclidean similarity of the configuration of points $(1,0), (0,1), (1,1)$. It rotates them by 45 degrees, then translates them. □

Example 5.9
$(1, i, 1+i) \mapsto e^{i\pi/4}(1, i, 1+i) + (3+4i, 3+4i, 3+4i)$
is the same Euclidean similarity of the same configuration, in complex notation. □

Suppose z is a complex number. $\Re z$ denotes the real part of z, and $\Im z$ denotes its imaginary part. $\bar{z} = \Re z - i\Im z$ denotes the conjugate of z, and $|z| = \sqrt{(\Re z)^2 + (\Im z)^2}$ denotes the absolute value. $z^{-1} = \bar{z}/|z|^2$ is the inverse.

Any pair of distinct points can be transformed by a Euclidean similarity into any other pair of distinct points. To see this, apply the translation shape transform to each, obtaining two 2×1 matrices; treat each as a complex number, which is not zero because the points in each pair were distinct. So their ratio is a scaling and rotation of one to the other.

We first use the transform π_1 from the previous section, to remove the effect of translations. Every complex number is decomposable into the form $re^{i\theta}$, where $r > 0$. Consequently, if the two columns of the translation transform are interpreted as two complex numbers z_1 and z_2, their equivalence class under rotation and scaling is the set of complex multiples of z_1 and z_2. Provided z_1 is not zero their complex quotient $w = z_2/z_1 = z_2\bar{z}_1/|z_1|^2$ is invariant under Euclidean similarities. $z_1 = 0$ can be interpreted as a point at infinity adjoined to the complex plane. It has long been known that the complex plane with infinity adjoined to it can be mapped to a sphere by stereographic projection. See the elegant book by Stillwell for instance [168]. However there is a particular stereographic projection to a sphere of radius $1/2$ such that the metric on the sphere has a similar property to property (iii) of π_1 (cf theorem 5.1). This construction is due to Kendall [99].

But first we must find sufficient conditions for a composition of complete invariant maps to be complete invariant:

Lemma 5.2
Let G_1 be a group acting on M_1 and $\pi_1 : M_1 \to M_2$ a surjective map with the properties:
(i) $\forall g \in G_1 \quad \pi_1(gX) = \pi_1(X)$ (invariance);
(ii) $\pi_1(X) = \pi_1(Y) \Rightarrow \exists g \in G_1 \quad X = gY$ (complete invariance);
Let G_2 be a group acting on M_2, and π_2 a map with the properties:
(i) $\forall g \in G_2 \quad \pi_1(gX) = \pi_1(X)$ (invariance);
(ii) $\pi_1(X) = \pi_1(Y) \Rightarrow \exists g \in G_2 \quad X = gY$ (complete invariance);

If G is a group satisfying

$$\forall g \in G \quad \exists g_1 \in G_1, g_2 \in G_2 \quad g = g_1 g_2 \tag{5.4}$$

$$\forall g_2 \in G_2 \quad \forall x \in M_1 \quad \pi_1(g_2 x) = g_2 \pi_1(x) \tag{5.5}$$

then $\pi = \pi_2 \pi_1$ also has properties (i) and (ii).

Proof

(i)

$\pi(gx)$	$=$	$\pi_2(\pi_1(g_1 g_2 x))$	from equation 5.4
	$=$	$\pi_2(\pi_1(g_2 x))$	from invariance of π_1 to G_1
	$=$	$\pi_2(g_2 \pi_1(x))$	from equation 5.5
	$=$	$\pi_2(\pi_1(x))$	from invariance of π_2 to G_2
	$=$	$\pi(x)$	by definition of π

(ii) $\pi(x) = \pi(y)$
$\Rightarrow \pi_2(\pi_1(x)) = \pi_2(\pi_1(y))$
$\Rightarrow \pi_1(x) = g_2 \pi_1(y) = \pi_1(g_2 y)$
$\Rightarrow x = g_1 g_2 y$

∎

Theorem 5.3
Let W be a 3 point configuration, represented as a 2×3 matrix, some of whose columns are distinct. Let π_1 be as defined in theorem 5.1. Interpret z_1, z_2 as complex numbers. If the first and second columns of W are not equal, let

$$w = z_2 / z_1$$

$$x = \frac{1 - (\Re w)^2 - (\Im w)^2}{2(1 + (\Re w)^2 + (\Im w)^2)}, y = \frac{\Re w}{1 + (\Re w)^2 + (\Im w)^2}, z = \frac{\Im w}{1 + (\Re w)^2 + (\Im w)^2}$$

If the first and second columns of W are equal, let

$$(x, y, z) = (\frac{1}{2}, 0, 0)$$

We define

$$\pi_2(z_1, z_2) = (x, y, z), \mathbf{C}^2 \setminus \{0\} \rightarrow S^2(1/2)$$

π_2 is surjective with the following three properties:

(i) $\forall g \in SO(2) \quad \pi_2(gX) = \pi_2(X)$ (invariance);
(ii) $\pi_2(X) = \pi_2(Y) \Rightarrow \exists g \in SO(2) \quad X = gY$ (complete invariance);
(iii) If $x, y \in M_{2,2}$ such that $\|x\| = 1 = \|y\|$ then $\rho(\pi_2(X), \pi_2(Y)) = \inf_{g \in SO(2)} d(X, gY)$ (spherical least squares)

Let G be the group of euclidean similarities. Then $\pi = \pi_2 \pi_1$ is surjective with the properties:

(i) $\forall g \in G \, \pi(gX) = \pi(X)$ (invariance);
(ii) $\pi(X) = \pi(Y) \Rightarrow \exists g \in G \, X = gY$ (complete invariance);

Proof

See Kendall [100] section 1. Note that our x, y, z is Kendall's X, Y, Z respectively. Substitute $w = re^{i\psi}$ and transform the formulae for X, Y, Z as functions of r, ψ into functions of $\Re w, \Im w$.

Since any Euclidean similarity can be decomposed into $X \mapsto \lambda RX + T$ where $R \in SO(2), \lambda > 0$, T a 2×3 matrix with all columns equal, $g \in G \Rightarrow g = g_1 g_2$. And since $(\lambda RX)Q = \lambda R(XQ)$ it follows that $\pi_1(g_2 x) = g_2 \pi_1(x)$. From lemma 5.2 it follows that $\pi_2 \pi_1$ has properties (i) and (ii).

See [99] for the proof of property (iii) of π_2. ∎

That this shape space is a sphere of radius $1/2$ is quite far reaching. Most importantly it demonstrates that shape spaces are not flat, Euclidean spaces. Yet most of pattern recognition theory assumes that pattern spaces are Euclidean spaces !

Definition 5.5
The Euclidean shape spaces are denoted Σ_m^k where k is the number of points and m is the dimension of the Euclidean space the configurations belong to.

One very important property of most shape spaces emerges in this shape space: it is compact (topological property, see [133]). This implies that every continuous function of shape is bounded. We can see from the above theorem that the shape metric is in fact bounded by $\pi/2$.

Example 5.10
Consider the configuration of 3 points which are the vertices of the equilateral triangle

$$(e^{i0}, e^{i2\pi/3}, e^{i4\pi/3}) = (1, -\frac{1}{2} + \frac{\sqrt{3}}{2}i, -\frac{1}{2} - \frac{\sqrt{3}}{2}i)$$

$$\pi_1(1, -\frac{1}{2} + \frac{\sqrt{3}}{2}i, -\frac{1}{2} - \frac{\sqrt{3}}{2}i) = (\frac{-3\sqrt{2} + \sqrt{6}i}{4}, \frac{-3 - 3\sqrt{3}i}{2\sqrt{6}})$$

$$w = z_2/z_1 = i$$

$$x = 0, y = 0, z = 1/2$$

So the shape transform of this equilateral triangle is $(0, 0, 1/2)$ which is the north pole of the sphere of radius $1/2$. Figure 5.1 depicts this 3 point configuration, and the value of its shape transform in the shape space. □

Example 5.11
Take the vertices of the same equilateral triangle (example 5.10) in reverse order

$$(e^{i4\pi/3}, e^{i2\pi/3}, e^{i0}) = (-\frac{1}{2} - \frac{\sqrt{3}}{2}i, -\frac{1}{2} + \frac{\sqrt{3}}{2}i, 1)$$

$$z_1^* = \frac{-1}{2} - \frac{\sqrt{3}}{2}i$$

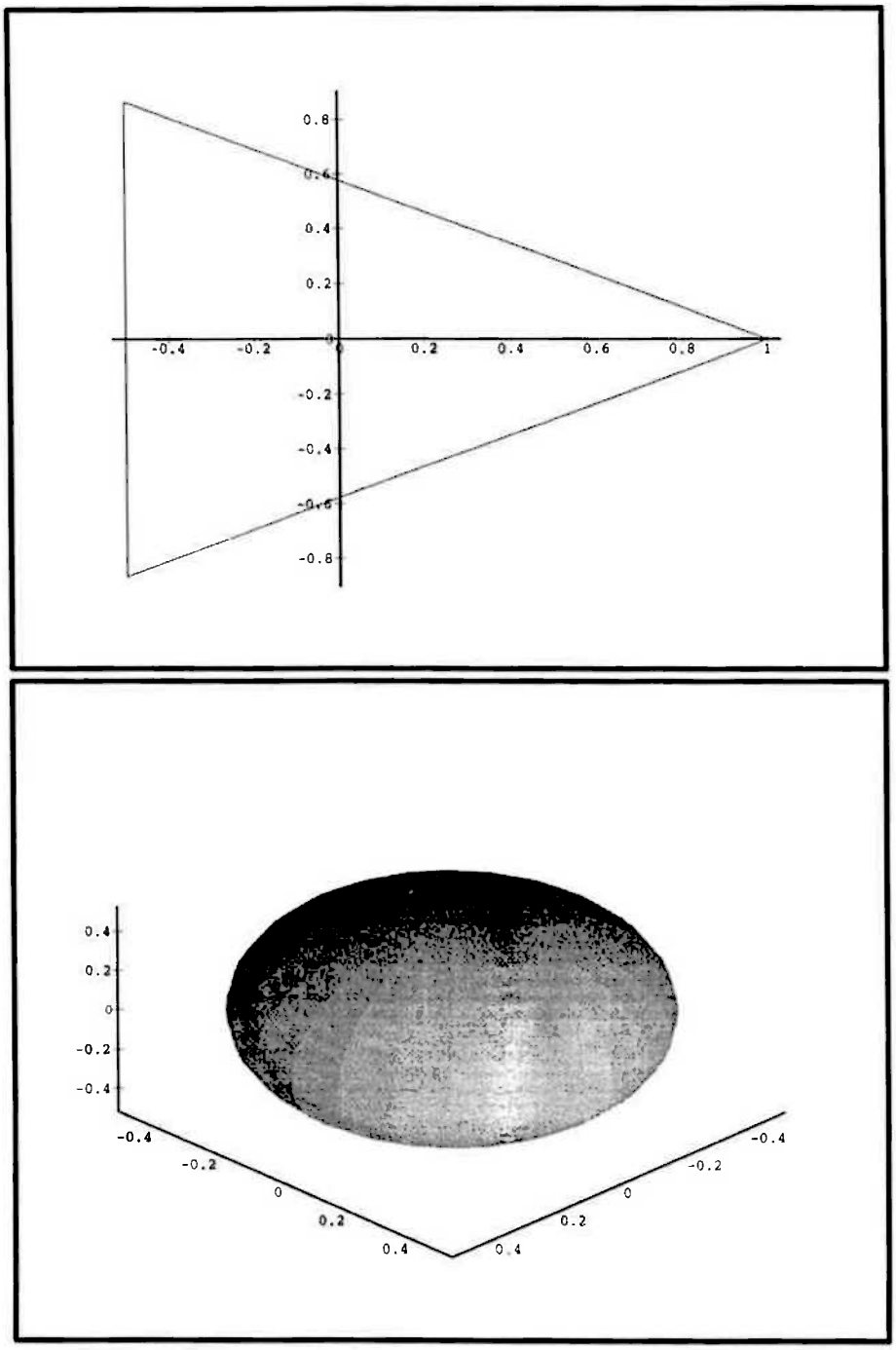


Figure 5.1: A configuration and its shape in a Euclidean shape space

$$z_2^* = \frac{-1}{2} + \frac{\sqrt{3}}{2}i$$

$$z_3^* = 1$$

$$z_1 = \frac{z_2^* - z_1^*}{\sqrt{2}} = \frac{\sqrt{3}}{\sqrt{2}}i$$

$$z_2 = \frac{2z_3^* - z_1^* - z_2^*}{\sqrt{6}} = \frac{\sqrt{3}}{\sqrt{2}}$$

$$w = z_2/z_1 = 1/i = -i$$

$$x = 0, y = 0, z = -1/2$$

Thus the shape transform of this configuration is the south pole of the shape space, which is a sphere of radius $1/2$. □

Examples 5.10 and 5.11 demonstrate that configurations are not the same thing as polygons, and that the shape transform of a configuration and its mirror image are usually distinct.

Example 5.12
Consider the configuration $(-1, 0, 1)$ of three collinear points.

$$z_1^* = -1$$

$$z_2^* = 0$$

$$z_3^* = 1$$

$$z_1 = \frac{z_2^* - z_1^*}{\sqrt{2}} = \frac{1}{\sqrt{2}}$$

$$z_2 = \frac{2z_3^* - z_1^* - z_2^*}{\sqrt{6}} = \frac{\sqrt{3}}{\sqrt{2}}$$

$$w = z_2/z_1 = \sqrt{3}$$

$$x = -1/4, y = \sqrt{3}/4, z = 0$$

The shape transform of this configuration is on the equator of the sphere of radius $1/2$. In fact the equator consists of the shapes of collinear triads of points. □

We identified Σ_2^3 as a sphere. More generally, when $m = 2$, we can identify the Euclidean shape space as a complex projective space.

Definition 5.6
Let $z, w \in \mathbf{C}^n$, such that neither is zero. $z \sim w$ if $z = \alpha w$ for some non-zero complex number α. The set of such equivalence classes is denoted $\mathbf{CP}^n$ and is called complex projective space. We define a metric on $\mathbf{CP}^n$ by

$$\rho(z, w) = arccos\left(\frac{|\sum_1^{k-1} z_i \bar{w}_i|}{\sqrt{\sum_1^{k-1} |z_i|^2 \sum_1^{k-1} |w_i|^2}}\right) \tag{5.6}$$

Lemma 5.4
Suppose π is π_1 regarded as a map from M onto $\mathbf{CP}^{k-2}$. Then π has the properties:
(i) $\forall g \in G \quad \pi(gX) = \pi(X)$ (invariance);
(ii) $\pi(X) = \pi(Y) \Rightarrow \exists g \in G \quad X = gY$ (complete invariance);
(iii) Let $X \in M_{2,k-1}$ and $z \in \mathbf{C}^{k-1}$ be the real and complex representations of the same translation shape. Likewise Y and w.
If $\|X\| = 1 = \|Y\|$ then $\inf_{R \in SO(2)} d(X, RY) = \rho(z, w)$ (spherical least squares).

Proof We prove each property of a shape transform in turn:
(i) Any Euclidean similarity can be expressed in complex notation as $z \mapsto \alpha z + t$, $t \in \mathbf{C}^k, t_1 = \ldots = t_k$, $\alpha \in \mathbf{C} \setminus \{0\}$.
$\pi_1(\alpha z + t) = \alpha \pi_1(z)$
π_1 is invariant up to non-zero complex multiples.

(ii) If $\pi_1(z) = \alpha \pi_1(w)$ then $z = \alpha w + t$, where $t \in \mathbf{C}^k, t_1 = \ldots = t_k$.

(iii) Proved in section 3 of [99]. ∎

The singular values decomposition is one of the important matrix factorisations [170][Appendix A]. A straight-forward definition of singular values can be given: the singular values of an $m \times k$ matrix A with $m \leq k$ are the square-roots of the largest m eigenvalues of $AA^\top$. However, singular values are usually computed by a specialized algorithm using Householder reduction which does not require the computation of eigenvalues [169]. There is a formula for the shape metric via the singular values decomposition. It generalises to Euclidean shape spaces of non-planar configurations, but we state it for the planar case only:

Lemma 5.5
Let X, Y be $2 \times k$ matrices neither of which have all their columns identical.
Let $\lambda_1 \geq \lambda_2$ denote the singular values of $(Y\tilde{Q})(X\tilde{Q})^\top$.
If $det((Y\tilde{Q})(X\tilde{Q})^\top) \geq 0$ then $\rho(\pi(X), \pi(Y)) = arccos(\lambda_1 + \lambda_2)$.
If $det((Y\tilde{Q})(X\tilde{Q})^\top) < 0$ then $\rho(\pi(X), \pi(Y)) = arccos(\lambda_1 - \lambda_2)$.

Proof

See section 2 of [106]. It is an extension of some results of Von Neumann [176]. Note that if $det((Y\tilde{Q})(X\tilde{Q})^\top) = 0$ then either (i) the columns of X or the columns of Y are contained in a line; or (ii) $(Y\tilde{Q})(X\tilde{Q})^\top = 0$ and $\rho(\pi(X), \pi(Y)) = \pi/2$. ∎

If two configurations of k-points (neither of which consist of identical points) are represented by complex k-tuples z and w, then there is a direct formula for $\rho(\pi(z), \pi(w))$ which does not involve computing the shape transform first:

$$\rho(\pi(z), \pi(w)) = arccos\left(\frac{\sum_{i=1}^{i=k} (z_i - z_c)(\bar{w}_i - \bar{w}_c)}{\sqrt{\sum_{i=1}^{i=k} |z_i - z_c|^2} \sqrt{\sum_{i=1}^{i=k} |w_i - w_c|^2}}\right) \tag{5.7}$$

See formulas (20) and (22) in [99]. Its legitimacy stems from the fact that it is a metric on shape space with the spherical least squares property. Its boundedness is apparent from this formula, although the actual bound is $\pi/2$ not π. Also, ρ is not a metric on configuration space, because it is zero for any configurations $z \neq w$ with the same shape. Configurations can be compared with the euclidean metric on matrices.

As was the case with π_1, π does not have a well-defined inverse. The problem of finding a pseudo-inverse to π which is nearest to a given configuration is known as the Procrustes problem. There is a large literature on the Procrustes problem for the Euclidean similarity group, see [149], [64] for a review.

The more general Euclidean shape spaces when $m \geq 3$ have been studied by Le and Kendall [109]. There is also a theory of spherical shape spaces, due to Carne [32]. The above papers assume familiarity with Riemannian geometry; [59] is a good introduction.

5.3 Affine shape spaces

We will study the affine shape spaces in this section. We will start with an example which can be visualised, and then progress to the general case. Sparr has used the term affine shape in a series of papers [137], [162], [161], [164], [165], [166]; Sparr's shape is a point in our shape space. He also defines a shape transform. The actual value of an affine shape transform (shape coordinates) is usually not needed.

The affine group consists of rotations, translations, scaling (dilatation), shearing and reflection. An affine transformation of $\mathbf{R}^m$ is represented by two matrices (g, t) where g is $m \times m$ non-singular and t is the translation vector.

Example 5.13

$$\begin{bmatrix} \sqrt{2} & -\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$

is an affine transformation of the configuration of points $(1,0), (0,1), (-1,0), (1,1)$. □

Note that the affine group is *not* the direct product of $GL(m)$ and $\mathbf{R}^m$ because its group operation takes the form $g'(gx + t) + t' = g'gx + gt + t'$ which does not equal the product group operation which takes the form $g'gx + t + t'$.

The affine group can map any 3 point configuration to any other 3 point configuration if neither is contained in a line. This is because the translation shape transform of both will be 2×2 matrices. These two 2×2 matrices will be non-singular if neither 3 point configuration was contained in a line. Hence there is always some non-singular 2×2 matrix which maps the translation transform of one to the translation transform of the other.

We remove the effect of translations using the translation shape transform. We have

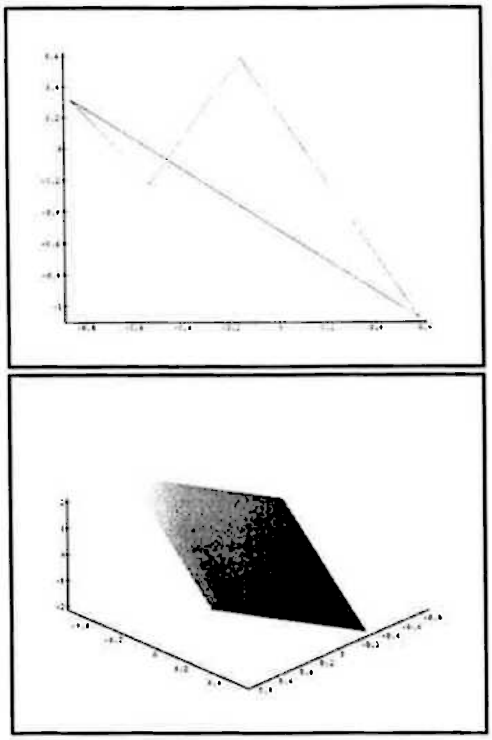


Figure 5.2: Transform from point configuration to affine shape

exhibited a translation shape transform in closed form.

Example 5.14
Let

$$W = \begin{bmatrix} -.1549980895 & -.5339724225 & -.8587859939 & .5740786490 \\ .5865929574 & -.2251144858 & .3144579499 & -1.071341100 \end{bmatrix}$$

$$W\tilde{Q} = \begin{bmatrix} -.2679753207 & -.4199247939 & .9439655115 \\ -.5739638375 & .1091808728 & -1.122934646 \end{bmatrix}$$

The linear space spanned by the rows of this 2×3 matrix is called its rowspace. The rowspace of this 2×3 matrix is the affine shape of W. It is a plane in three dimensional space. The transform from W to its affine shape is depicted in figure 5.2. □

Provided a planar configuration of k points is not contained in a line, its affine shape transform is a plane in $\mathbf{R}^{k-1}$. If $k > 4$ this is a plane in a Euclidean space of more than three dimensions, which is not easy to visualise. Nonetheless it is easy to deal with using linear algebra. Analogously, provided a three dimensional configuration of k points is not contained in a plane, its affine shape transform is a 3 dimensional linear sub-space of $\mathbf{R}^{k-1}$. Any four points not coplanar can be transformed by an affine transformation into any other non-coplanar four points.

Lemma 5.6
If X is an $m \times k$ matrix whose columns are not contained in a hyperplane of $\mathbf{R}^m$, then the rank of $X\tilde{Q}$ is m.

Proof

If the columns of X are not contained in a hyperplane of $\mathbf{R}^m$ then the columns span $\mathbf{R}^m$.
Thus rank$X = m$ and $m \leq k$.
Let X' be the matrix obtained by adding a row below X consisting of entries equal to 1.
If rank$X' < m+1$ there would exist $a_1, \ldots, a_{m+1}$ such that $a_1 X_{1i} + \ldots + a_m X_{mi} + a_{m+1} = 0$ for i from 1 to $m \leq k$, which is the equation of a hyperplane.
So rank$X' = m + 1$.
$X'Q$ is an $(m+1) \times k$ matrix obtained by adjoining to XQ the (last) row $(\sqrt{k}, 0, \ldots, 0)$.
Since Q is orthogonal rank$X'Q$ = rank$X' = m + 1$.
It follows that rank$X\tilde{Q} = m$.since it is the submatrix of XQ' in rows $1 \ldots m$ and columns $2 \ldots k$. ∎

Definition 5.7
The set of m dimensional linear subspaces of $\mathbf{R}^n$ is denoted $G(m, n)$ and is called a Grassmannian. Let M be the set of k point configurations in $\mathbf{R}^m$ which are not contained in a hyperplane of $\mathbf{R}^m$. Let G denote the affine group. By lemma 5.6, $\pi_1(M)$ is the set of $m \times k - 1$ matrices of rank m. We define

$$\pi_2 : \pi_1(M) \to G(m, k-1), X \mapsto \text{rowspace}(X)$$

$$\pi = \pi_2 \pi_1$$

π is called the affine shape transform.

Lemma 5.7
π has the following properties:

(i)	$\forall g \in G \; \pi(gX) = \pi(X)$	(invariance)
(ii)	$\pi(X) = \pi(Y) \Rightarrow \exists g \in G \quad X = gY$	(complete invariance)

Proof

$\pi_1(X) = \pi_1(X + T)$
From lemma 5.6 $\pi_1(M)$ is the set of $m \times k - 1$ matrices of rank m.

First we show π_2 has properties (i) and (ii):
(i) Suppose x is an $m \times k - 1$ matrix of rank m, and $g_2 \in GL(m)$.
Then the rows of $g_2 x$ are linear combinations of the rows of x. Hence the rowspace of $g_2 x$ is contained in the rowspace of x.
Since $x = g_2^{-1} g_2 x$, the rows of x are also linear combinations of the rows of $g_2 x$. Hence the rowspace of $g_2 x$ equals the rowspace of x.
Thus $\pi_2(g_2 x) = \pi_2(x)$.
(ii) Suppose $\pi_2(x) = \pi_2(y)$, where x, y have rank m.
Then the rows of x are a basis for the rowspace of x, and the rows of y are another basis for the rowspace of x, because the rowspaces of x and y are equal.
Hence there is a $g_2 \in GL(m)$ mapping one basis into the other, so

$$\exists g_2 \in GL(m) \quad g_2 x = y$$

The affine group has the properties:

$$\forall g \in G \quad \exists \quad g_1 \in \mathbf{R}^m \quad \exists g_2 \in GL(m) \quad g = g_1 g_2$$
$$\forall g_2 \in GL(m) \quad \forall x \in M \quad \pi_1(g_2 x) = (g_2 x)\tilde{Q} = g_2(x\tilde{Q}) = g_2 \pi_1(x)$$

Hence from lemma 5.2 π has properties (i) and (ii). ∎

By definition of configuration, any list of points which has all entries equal is not a configuration. The value of π_1 on such a tuple of points is zero. Since all linear spaces intersect at the origin, there would be no metric on an affine shape space which included this "shape". Like the Euclidean shape spaces the affine shape spaces do not contain this point.

Like the Euclidean shape spaces, the affine shape spaces are also compact. Thus any continuous function of affine shape, such as a metric, will be bounded.

The problem of obtaining a pseudo-inverse for π is part of what is known as the Procrustes problem. See Rohlf and Slice [149] for a survey of the Procrustes problem for the affine group. A solution to the Procrustes problem is also given by Werman and Weinshall [181]. Note that the solution to the Procrustes problem is not always unique.

We have now identified the affine shape spaces as consisting of Grassmannians, but we have not given any coordinates for affine shape. The standard coordinates of the Grassmanian in algebraic geometry are the plucker coordinates [74].

Definition 5.8
Let $I = \{(i_1, \ldots, i_m) \mid 1 \leq i_1 < \ldots < i_m \leq n\}$,
A be a m by n matrix of rank m, where $m \leq n$,
A_i the determinant of the $m \times m$ matrix formed from columns $i_1, \ldots, i_m$, then

$$\psi : G(m,n) \to \mathbf{P}^{\binom{n}{m}-1}, \text{rowspace}(A) \mapsto (A_i)_{i \in I}$$

is called the plucker map.

Lemma 5.8
The plucker map is well-defined

Proof

Determinants have the property that if A, B are square matrices then

$$\det(AB) = \det(A)\det(B)$$

$\Rightarrow \forall i \in I \quad (gA)_i = \det(g) A_i$, where $(gA)_i$ denotes the determinant of the $m \times m$ matrix formed from columns $i_1, \ldots, i_m$ of gA.
$\Rightarrow \psi(gA) = (\det g)\, \psi(A)$ for any non-singular g.
Therefore the plucker map is well-defined. ∎

The Grassmannian is mapped into a projective space by the plucker map. It is not the whole of that projective space, but only that subset cut out by a system of equations known as the plucker relations [74].

Example 5.15
This is an example of Plucker coordinate calculation. Let

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

We evaluate the plucker map of A.

$$I = \{(1,2,3),(1,2,4),(1,3,4),(2,3,4)\}$$

$$A_{(1,2,3)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} = 1 \quad A_{(1,2,4)} = \begin{vmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{vmatrix} = 1$$

$$A_{(1,3,4)} = \begin{vmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{vmatrix} = -1 \quad A_{(2,3,4)} = \begin{vmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{vmatrix} = 1$$

$$\psi(A) = (1,1,-1,1)$$

□

A metric for the Grassmanian was defined by J.H.C. Whitehead [185] to be the largest angle between the two subspaces. Linear subspaces in vector spaces of four or more dimensions can have more than one angle between them. The problem of calculating the angles between two linear subspaces of a vector space is a somewhat obscure problem of Euclidean geometry; a concise solution is outlined by Flanders in [57], see algorithm 5.1.

Algorithm 5.1

angles(X, Y)

Inputs	X, Y	$m \times (k-1)$ matrices of rank m
Outputs	$(\theta_1, \ldots, \theta_m)$	m angles $\theta_1 \geq \ldots \geq \theta_m$

X_b = matrix whose rows are orthonormal basis of $X^\top$
Y_b = matrix whose rows are orthonormal basis of $Y^\top$
Compute $\lambda_1 \leq \ldots \leq \lambda_m$ the singular values of $X_b Y_b^\top$
for i from 1 to m do
  $\theta_i = \arccos(\lambda_i)$
end ∎

Flanders' algorithm is implemented by the function angles in the C implementation (see stereolib.h in appendix B), but

- we do not use Gram-Schmidt orthogonalisation because that is known to be numerically unstable; the Q factor in the QR decomposition is used (by NAPACK's basis routine);

- there is a parameter in the basis calculation called cutoff. If it is too small more than m angles will be returned and if it is too large less than m angles will be returned;
- numerical errors can give singular value estimates less than 0 or greater than 1; singular values are always non-negative.

Definition 5.9
A metric on an affine shape space is defined by

$$d(\pi(X), \pi(Y)) = \text{angles}(\pi_1(X), \pi_1(Y))_1$$

This is the largest angle between $\pi(X)$ and $\pi(Y)$.

The author could only find references to the problem of computing the angles between two linear spaces in the numerical analysis literature [144] pg224. The length of the shortest curve on the Grassmannian joining two points (ie the geodesic length) is calculated in [188] in terms of the angles. The author also obtained a metric in Plucker coordinates:

$$d(\pi(X), \pi(Y)) = \arccos\left(\frac{|< \psi(\pi_1(X)), \psi(\pi_1(Y)) >|}{\|\psi(\pi_1(X))\| \, \|\psi(\pi_1(Y))\|}\right)$$

but the computational complexity of this metric is much higher than the metric in definition 5.9.

We now give a second coordinate system for $G(2, n)$, which will be used in subsequent chapters.

Definition 5.10
Let

$$\Delta_{ij} = (x_{i+1} - x_i)(y_{j+1} - y_j) - (x_{j+1} - x_j)(y_{i+1} - y_i)$$
$$\lambda_{ij} = (y_{j+1} - y_j)(x_j - x_i) - (x_{j+1} - x_j)(y_j - y_i)$$

Let $[(\Delta_{ij}, \lambda_{ij})]$ denote the matrix whose ij-th entry is $(\Delta_{ij}, \lambda_{ij})$. Let L_i be the unique line joining (x_i, y_i) and (x_{i+1}, y_{i+1}).

Lemma 5.9 gives a geometric interpretation to the matrix $[(\Delta_{ij}, \lambda_{ij})]$. We will rely heavily on this in chapters 7 and 11.

Lemma 5.9
$(\Delta_{ij}, \lambda_{ij})$ are the homogeneous coordinates of $L_i \cap L_j$ where $(\Delta_{i(i-1)}, \lambda_{i(i-1)})$ are the homogeneous coordinate of (x_i, y_i) and $(\Delta_{i(i+1)}, \lambda_{i(i+1)})$ are the homogeneous coordinates of (x_{i+1}, y_{i+1}) on L_i.

Proof The parametric equations for $\mathbf{p} \in L_i$ and L_j respectively are $\mathbf{p} = (x_i, y_i) + s(x_{i+1} - x_i, y_{i+1} - y_i)$ and $\mathbf{p} = (x_j, y_j) + t(x_{j+1} - x_j, y_{j+1} - y_j)$. There are two cases:

(i) When L_i and L_j are not parallel. They have a unique point of intersection given by s, t satisfying the matrix equation below:

$$\begin{bmatrix} x_{i+1} - x_i & x_{j+1} - x_j \\ y_{i+1} - y_i & y_{j+1} - y_j \end{bmatrix} \begin{bmatrix} s \\ t \end{bmatrix} = \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} \tag{5.8}$$

Now Δ_{ij} is the determinant of $\begin{bmatrix} x_{i+1}-x_i & x_{j+1}-x_j \\ y_{i+1}-y_i & y_{j+1}-y_j \end{bmatrix}$. Inverting this 2×2 matrix we find that $s = \lambda_{ij}/\Delta_{ij}$ and $t = \lambda_{ji}/\Delta_{ij}$.

(ii) When L_i and L_j are parallel.

L_i and L_j are parallel

$$\Leftrightarrow det \begin{bmatrix} x_{i+1}-x_i & x_{j+1}-x_j \\ y_{i+1}-y_i & y_{j+1}-y_j \end{bmatrix} = 0.$$

$$\Leftrightarrow \Delta_{ij} = 0$$

$\Leftrightarrow (\Delta_{ij}, \lambda_{ij})$ is the point at infinity on L_i. ∎

Sparr has given a different definition of affine shape. We show they are equivalent because the two shape spaces are isomorphic.

Definition 5.11
Let W be the matrix of a configuration of k points in $\mathbf{R}^m$. The Sparr's shape of W is the nullspace of

$$\begin{bmatrix} W_{11} & \dots & W_{1k} \\ \vdots & & \vdots \\ W_{m1} & \dots & W_{mk} \\ 1 & \dots & 1 \end{bmatrix}$$

Lemma 5.10
Every element of $G(m,n)$ can be represented by the rowspace of an $m \times n$ matrix of rank m. Every element of $G(n-m,n)$ can be represented by the nullspace of an $m \times n$ matrix of rank n. The map which maps the rowspace of an $m \times n$ matrix of rank m to its nullspace is an isomorphism of $G(m,n)$ and $G(n-m,n)$.

Proof

The plucker map of $G(m,n)$ and that of $G(n-m,n)$ have ranges of the same dimension, and satisfy identical equations (see the plucker relations in [74]). ∎

Corollary 5.11
The set of Sparr shapes (definition 5.11) of configurations of k points in $\mathbf{R}^m$ not contained in a hyperplane is isomorphic to $G(m,k-1)$.

Proof

Let W be the matrix of a configuration of k points in $\mathbf{R}^m$ not contained in a hyperplane. Let N be a matrix whose columns are a basis for an affine shape by definition 5.11.

$$\phi : N \mapsto \pi_1(N^\top)$$

maps an affine shape into $G(k-1-m,k-1)$, since it is well-defined:

$$\forall g \in GL(k-m-1) \quad \phi(Ng) = \pi_1(g^\top N^\top) = (g^\top N^\top)\tilde{Q} = g^\top(N^\top \tilde{Q}) = g^\top \pi_1(N^\top)$$

The inverse of ϕ is:

$$\phi^{-1}(x) = \begin{bmatrix} 0 & x \\ \sqrt{k} & 0 \end{bmatrix} Q^\top$$

Let $X_{(1,\ldots,m)}$ be obtained by adjoining the rows $m+1$ to k of the $k \times k$ identity matrix to $N^\top$.

$$det(X_{(1,\ldots,m)}Q) = det(X_{(1,\ldots,m)})$$

since Q is orthogonal. The first m rows of $X_{(1,\ldots,m)}Q$ are $N^\top Q$ and the remaining rows are the corresponding rows of Q. $det(X_{(1,\ldots,m)}Q)$ can be expanded into a linear combination of the minors of $N^\top Q$. Thus there is a linear map from $\psi(N^\top Q)$ to $\psi(N^\top)$. Omission of the first row of Q does not affect the linearity. Likewise there is an inverse linear map. Thus ϕ can be regarded as an isomorphism in the sense of algebraic geometry (see the definition of isomorphism of affine varieties in [74]). By lemma 5.10 $G(k-1-m, k-1) \cong G(m, k-1)$. ∎

5.4 Positive affine shape spaces

We will study the positive affine shape spaces in this section, commencing with an example that can be visualised. Unlike affine shapes the positive affine shapes retain orientation, which means the distinction between clockwise and anticlockwise is maintained.

The positive affine group consists of rotations, translations, scaling (dilatation) and shearing. It excludes reflections, in contrast with the affine group. A positive affine transformation is represented by two matrices (g, t) where g is $m \times m$ with $det(g) > 0$ and $t \in \mathbf{R}^m$. t is the translation vector.

Example 5.16

$$\begin{bmatrix} 2 & 1 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$

is a positive affine transformation of the configuration of points $(1,0), (0,1), (-1,0), (1,1)$. □

The positive affine group is not the direct product of $GL_+(m)$ and $\mathbf{R}^m$.

The positive affine group can map any two pairs of distinct points to each other, because it contains the Euclidean similarity group. Unlike the affine group, it cannot map any non-collinear three point configurations to each other. Suppose A, B are matrices representing 3 point configurations, so $\pi_1(A)$ and $\pi_1(B)$ are 2×2 matrices. If neither of A, B consists of three collinear points, then $\pi_1(A)$ and $\pi_1(B)$ are non-singular. The positive affine group preserves the sign of $det(\pi_1(A))$ and $det(\pi_1(B))$. If the signs of $det(\pi_1(A))$ and $det(\pi_1(B))$ are equal, there will exist $g \in GL_+(2)$ such that $\pi_1(A) = g\pi_1(B)$. Thus there are precisely two non-collinear 3 point positive affine shapes. The determinant of a 2×2 matrix is a special case of the plucker map.

Example 5.17
Let

$$W = \begin{bmatrix} -.1549980895 & -.5339724225 & -.8587859939 & .5740786490 \\ .5865929574 & -.2251144858 & .3144579499 & -1.071341100 \end{bmatrix}$$

$$W\tilde{Q} = \begin{bmatrix} -.2679753207 & -.4199247939 & .9439655115 \\ -.5739638375 & .1091808728 & -1.122934646 \end{bmatrix}$$

The direction vector of the cross product of the two rows of $W\tilde{Q}$ is called the orientation of its rowspace. The rowspace of this 2×3 matrix together with its orientation is the positive affine shape of W. It is a plane in three dimensional space, together with a unit vector which points (up or down ?). The cross product of two vectors in $\mathbf{R}^3$ is also a special case of the plucker map. □

We can state a map which is a complete invariant to $GL_+(m)$, based on the plucker map.

$$\tilde{\psi} : A \mapsto \psi(A)/\|\psi(A)\|$$

The transform from each planar point configuration to its positive affine shape is a map from a point configuration to an oriented plane in some Euclidean space.

Lemma 5.12
The positive affine shape of a configuration x in $\mathbf{R}^m$ which does not lie in a hyperplane of $\mathbf{R}^m$ is an m-dimensional linear subspace of $\mathbf{R}^{k-1}$ with an orientation assigned to it, provided $m \leq k-1$.

Proof

Let y be xQ with its first column dropped, so y is a m by $k-1$ matrix which is translation normalised.
The m rows of y are linearly independent precisely when the configuration does not lie in a hyperplane of $\mathbf{R}^m$. So the m rows of y span a m-dimensional linear subspace of $\mathbf{R}^{k-1}$, because $m \leq k-1$ and the configuration does not lie in a hyperplane.
The action of $GL_+(m)$ just changes the basis of rowspace(y), but preserves its orientation, hence the oriented rowspace of y is invariant to affine transformations. ∎

Definition 5.12
An oriented Grassmannian $\tilde{G}(m,n)$ is the set of linear subspaces of $\mathbf{R}^n$ with an orientation assigned to each.

The oriented Grassmannians can each be embedded in a sphere, by

$$\tilde{G}(m,n) \to S^{\binom{n}{m}-1}, A \mapsto \psi(A)/\|\psi(A)\|$$

A metric on the oriented Grassmannian can be stated:

$$d(\tilde{\psi}(X), \tilde{\psi}\pi(Y)) = \arccos(< \tilde{\psi}(X), \tilde{\psi}(Y) >)$$

where $\pi = \tilde{\psi}\pi_1$. Note the absence of absolute value signs.

5.5 General perspectives of non-coplanar points

From lemma 4.5, with $m = 3$, no metric exists on the projective space $\mathbf{P}^2$. Since general changes of perspective viewpoint of coplanar points induce two dimensional projective transformations on the image, this means there is no metric on the spaces of coplanar scenes and arbitrary perspective cameras. This justified the study of the shape spaces we presented above. We can also see from lemma 4.5, with $m = 4$ that three dimensional reconstructions in the sense of projective invariants has no associated metric. In this section we will construct a metrisable shape space for the arbitrary perspective camera, in the case of non-coplanar tuples of scene points, none of which lie in the focal plane of the camera.

In chapter 4 we saw that changes of camera parameters induce the following action of the group G on $\iota(\tilde{M})$:

$$\theta : G \times \iota(\tilde{M}) \to \iota(\tilde{M}) \quad \theta((g,d),\iota(X)) = g\iota(X)d$$

However, many scenes with distinct images would be deemed equivalent by invariants to G. From theorem 4.3 the members of G that are induced depend on the scene. We will restrict θ to those elements of $G \times \iota(\tilde{M})$ that satisfy $g\iota(X)d = \iota(gX)$ and $gX \in \tilde{M}$. We will call this the restricted action of G.

Suppose the fourth coordinates t_i of all scene points equal 1. Then ι becomes invertible.

Lemma 5.13
Let $\iota(X) \sim \iota(Y)$ if there is a $g \in GL(4)$ such that $g_{41} = g_{42} = g_{43} = 0$, $g_{44} = 1$ and $\iota(X) = \iota(gY)$, where $X, Y \in \iota(\tilde{M})$. This is an equivalence relation on $\iota(\tilde{M})$.

Proof

Suppose $\iota(X) = \iota(gY)$, so for each i

$$Y_{1i}/Y_{3i} = (gX)_{1i}/(gX)_{3i}$$
$$Y_{2i}/Y_{3i} = (gX)_{2i}/(gX)_{3i}$$
$$1/Y_{3i} = 1/(gX)_{3i}$$

Thus $Y = gX$. Since the set of $g \in GL(4)$ such that $g_{41} = g_{42} = g_{43} = 0$ and $g_{44} = 1$ are a group, $\sim$ is an equivalence relation. ∎

We can therefore see that the restricted group action partitions $\iota(\tilde{M})$ into equivalence classes. Thus there is a quotient space $\iota(\tilde{M})/\sim$. We will introduce two lemmas from [22] to prove that this quotient space is a manifold.

Definition 5.13
An equivalence relation on a space X is called open if whenever a subset $A \subset X$ is open, then $[A]$ is also open.

Lemma 5.14
The equivalence relation associated with the restricted group action is open.

Proof

Suppose $g_{31} = 0, g_{32} = 1, g_{33} = g_{34} = 0$. Then d is of the form $\text{diag}(v/w)$ for vectors v, w, where division is componentwise, and $v \neq \lambda w$ for all non-zero λ. All $d \neq \text{diag}(\lambda, \ldots, \lambda)$ can be realized like this. If $g_{31} = 1/\lambda, g_{32} = g_{33} = g_{34} = 0$ then $d = \text{diag}(\lambda, \ldots, \lambda)$. Hence d is surjective. Let D be any open subset of the set of non-singular diagonal matrices. Since d in theorem 4.3 is a continuous function of g and X when $gX \in \tilde{M}$, the preimage of D is an open set of (g, X). Thus the restriction is an open subset of $G \times \iota(\tilde{M})$, and the restricted group action is continuous. It follows from this that the equivalence relation is open. ∎

Lemma 5.15
An equivalence relation on X is open if and only if the map $\pi : X \to X/\sim$, $X \mapsto [X]$ is open. When $\sim$ is open and X has a countable basis of open sets, then $X/\sim$ has a countable basis also.

Proof See lemma 2.3 in [22]. ∎

Lemma 5.16
Let $\sim$ be an open equivalence relation on a topological space X. Then $R = \{(x, y) \mid x \sim y\}$ is a closed subset of the space $X \times X$ if and only if the quotient space $X/\sim$ is Hausdorff.

Proof See lemma 2.4 in [22]. ∎

Lemma 5.17
The set

$$\{(\iota(X), \iota(Y)) \mid \exists g\, \iota(X) = \iota(gY)\, X, gY \in \tilde{M}\}$$

is a closed subset of $\iota(\tilde{M}) \times \iota(\tilde{M})$.

Proof

Since $\iota^2 = \iota$, we can rewrite an equation of the form $\iota(gX) = \iota(Y)$ into the form $\iota(gX) = Y$, where $Y_{3i} = 1$. Then $gX_{4i} = X_{4i}$ so $\iota(gX) = Y$ is equivalent to

$$\begin{aligned}(gX)_{1i}/(gX)_{3i} &= Y_{1i}\\ (gX)_{2i}/(gX)_{3i} &= Y_{2i}\\ (gX)_{4i}/(gX)_{3i} &= Y_{4i}\end{aligned}$$

which is equivalent to

$$\begin{aligned}(gX)_{3i}Y_{4i} &= X_{4i}\\ (gX)_{2i}Y_{4i} &= Y_{2i}X_{4i}\\ (gX)_{1i}Y_{4i} &= Y_{1i}X_{4i}\end{aligned}$$

Denoting the columns of X by (x_i, y_i, z_i, t_i) and the columns of Y by $(x'_i, y'_i, 1, t'_i)$

$$\begin{aligned}(g_{11}x_i + g_{12}y_i + g_{13}z_i + g_{14}t_i)t'_i &= t_i x'_i\\ (g_{21}x_i + g_{22}y_i + g_{23}z_i + g_{24}t_i)t'_i &= t_i y'_i\\ (g_{31}x_i + g_{32}y_i + g_{33}z_i + g_{34}t_i)t'_i &= t_i\end{aligned}$$

It suffices to prove that a sequence X_i, Y_i satisfying such equations converges to X, Y also satisfying these equations. There is a unique solution to the equations $(gX)_{1i}Y_{4i} =$

$Y_{1i}X_{4i}$ if and only if the matrix whose rows are $(x_it'_i, y_it'_i, z_it'_i, t_it'_i, t_ix'_i)$ has rank 4. Since this matrix is the limit of a sequence in $\tilde{M}$, where the limit is also in $\tilde{M}$, its rank is at least 4. Since the determinants of all 5×5 minors of the matrices in the sequence are zero, the determinants of all 5×5 minors of the limit matrix are zero. Hence the rank is exactly 4. Similarly, there is a unique solution to the equations $(gX)_{2i}Y_{4i} = Y_{2i}X_{4i}$. There is always a unique solution to the equations $(gX)_{3i}Y_{4i} = X_{4i}$. Thus we have shown the existence of g such that $\iota(gX) = Y$. It remains to prove that such a g is non-singular. Since ι is invertible, $\iota(gX) = \iota(Y)$ implies $gX = Y$. If $det(g) = 0$ then $Y \notin \tilde{M}$, a contradiction. ∎

Theorem 5.18
The quotient space $\iota(\tilde{M})/\sim$ is a manifold, and is metrisable.

Proof

From lemma 5.17 and lemma 5.16 we see that $\iota(\tilde{M})$ is a Hausdorff space. From lemma 5.14 and lemma 5.15 this quotient space also has a countable basis of open sets. Given any $X \in \tilde{M}$, there is a neighbourhood of the restricted orbit of $\iota(X)$ homeomorphic to a neighbourhood of the orbit of X in $G(3, k-1)$, because G acts freely, and $X \mapsto [\iota(X)]$ is continuous and open. Thus $\iota(\tilde{M})/\sim$ is locally Euclidean. Coordinate neighbourhoods on $\iota(\tilde{M})/\sim$ will be compatible because the corresponding neighbourhoods on $G(3, k-1)$ are compatible. Hence $\iota(\tilde{M})/\sim$ is a differentiable manifold. We can use the metric on $G(3, k-1)$ to give us a metric on $\iota(\tilde{M})/\sim$. ∎

It is actually the case that a metric exists on any manifold. This is a deep theorem called the Urysohn metrisation theorem [133].

Chapter 6

Connectedness theorem

In chapter 5 we pointed out that there is a distinction between planar configurations and polygons. Although the notion of shape (Euclidean, affine and positive affine) was based on configurations rather than polygons, we can develop a notion of shape based on polygons. This is based on the affine shape coordinates in definition 5.10.

Most flat shapes in images are boundaries, such as boundaries of letters or occluding contours of objects. These shapes can be viewed as polygons with an extremely large number of vertices. It is of central importance to observe that such shapes cannot intersect themselves: they are simple polygons. In this chapter we develop a necessary and sufficient condition on the coordinates of an affine shape for that shape to be the shape of a simple closed polygon (lemma 6.4).

This chapter presents a proof of theorem 6.6 which will be used in chapter 7 (to construct a language whose strings possess the complete invariance property with respect to certain deformations).

The mathematical difficulty of the material in this thesis achieves its peak in this chapter. It employs linear algebra, projective geometry, topology and differential geometry. The reader may wish to skip this chapter on a first reading.

6.1 Review of some geometry

Definition 6.1
The cross-ratio of four points on a line in $\mathbf{R}^n$ is

$$\{p_1, p_2; p_3, p_4\} = \frac{d(p_1, p_3)d(p_2, p_4)}{d(p_2, p_3)d(p_1, p_4)}$$

The cross-ratio is invariant to projective transformations see [155].

We give two definitions of a polygon.

Definition 6.2
A polygon $p : [0, 1] \to \mathbf{R}^2$ is a piecewise linear continuous function.

Definition 6.3
A polygon $P = (x_0, y_0), \ldots, (x_{k-1}, y_{k-1})$ where $(x_0, y_0), \ldots, (x_{k-1}, y_{k-1}) \in \mathbf{R^2}$ are all distinct is the set of line segments (called edges) $(x_0, y_0)(x_1, y_1), \ldots, (x_{k-2}, y_{k-2})(x_{k-1}, y_{k-1})$, and $(x_0, y_0), \ldots, (x_{k-1}, y_{k-1})$ are its vertices.

Throughout this chapter, indices of vertices are to be interpreted modulo k (ie indices cycle from k-1 back to 0). Some properties of polygons are defined below:

Definition 6.4
A polygon $p : [0, 1] \to \mathbf{R^2}$ is closed if $p(0) = p(1)$, simple if p is invertible. A polygon $P_1, \ldots, P_k$ is non-degenerate if no three consecutive vertices are collinear. A closed polygon $P_1, \ldots, P_k$ is the polygon $P_1, \ldots, P_k, P_1$. An orientation for a polygon is a choice of direction (clockwise or anticlockwise) for its edges.

Definition 6.5
If three vertices coincide they are called triple point vertices.

The notion of connectedness is quite intuitive. A set is connected if it cannot be separated into two parts which do not "touch" one another.

Definition 6.6
A separation of a topological space X is a pair of disjoint open sets U and V whose union is X. A set is connected if there is no separation of X.

Lemma 6.1
The image of a connected set under a continuous map is connected.

Proof See theorem 1.5 in chapter 3 of [133]. ∎

Definition 6.7
A path in a set S is a continuous function $\gamma : [0, 1] \to S$. A set is path connected if any two points in S can be joined by a path in S.

Intuitively, we may feel that there should be a path joining any pair of points in a connected set, that stays within the connected set. However, the next example shows that this is not true in general.

Example 6.1
Let $S = \{(x, sin(1/x) \mid 0 < x \leq 1\}$. Let $\bar{S}$ be the smallest closed set containing S. $\bar{S}$ is connected, but not path connected (see example 4, chapter 3 of [133] for the proof). □

However, every connected subset of a manifold is path connected.

Given a topological space X that is not necessarily connected, it can be decomposed into connected subsets of maximum size.

Definition 6.8
For any $x, y \in X$ we define $x \sim y$ if there is a connected subset of X containing both x and y. The equivalence classes are called connected components of X.

Given a topological space X, and a subset $Y \subset X$, we will need to be able to define

a topology on Y.

Definition 6.9
Let $V \subset Y$ be called open if there is an open subset U of X such that $V = U \cap Y$. This collection of (open subsets of Y) is a topology on Y, and is called the subspace topology.

The product $X \times Y$ of two topological spaces X and Y is the set $\{(x, y) \mid x \in X,\ y \in Y\}$. The product topology consists of sets $\{(x, y) \mid x \in U, y \in V\}$ where $U \subset X$ and $V \subset Y$ are open.

Lemma 6.2
If X and Y are path connected then $X \times Y$ is path connected in the product topology.

Proof

Suppose $(x_0, y_0), (x_1, y_1)$ are two points in $X \times Y$. Then there are paths p_X, p_Y such that $p_X(0) = x_0, p_X(1) = x_1$ and $p_Y(0) = y_0, p_Y(1) = y_1$. Let $p : [0, 1] \to X \times Y$ be defined by $p(t) = (p_X(t), p_Y(t))$. To show p is a path we have to show it is continuous. Let O be an open subset of $X \times Y$ in the product topology. Then there are open sets U, V in X, Y respectively such that $(x, y) \in O \Leftrightarrow x \in U, y \in V$. Since p_X, p_Y are continuous, there is an open subset A of $[0, 1]$ such that $p(A) = O$. Thus p is a path in $X \times Y$. Thus $X \times Y$ is path connected. ∎

See a book on vector or fibre bundles such as Osborn [141] for the definitions of fibre, fibre bundle and structure group.

6.2 Connectedness result

Equation 5.8 has a singularity when L_i and L_j are parallel. The following equation does not:

$$\begin{bmatrix} x_{i+1} - x_i & x_{j+1} - x_j \\ y_{i+1} - y_i & y_{j+1} - y_j \end{bmatrix} \begin{bmatrix} \lambda_{ij} \\ \lambda_{ji} \end{bmatrix} = \Delta_{ij} \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} \tag{6.1}$$

Also

$$\lambda_{ij}/\Delta_{ij} = \{1, \lambda_{ij}/\Delta_{ij}; \infty, 0\} = \{(\Delta_{i(i+1)}, \lambda_{i(i+1)}), (\Delta_{ij}, \lambda_{ij}); (\Delta_{ii}, \lambda_{ii}), (\Delta_{i(i-1)}, \lambda_{i(i-1)})\} \tag{6.2}$$

where $\{,;,\}$ denotes the cross-ratio.

For convenience we introduce the following notation for edge vectors:

Definition 6.10
$v_i = (x_{i+1} - x_i, y_{i+1} - y_i)$ for $i = 0, \ldots, k - 1$.

So these ratios, defined by equation 6.2, are the cross-ratio of four points: the i-th vertex, the intersections of the tangents to two edges v_i, v_j, the point at infinity on the tangent to the i-th edge, and the $(i+1)$-st vertex. See figure 6.1. The cross-ratio is the fundamental projective invariant. But when points at infinity are distinguished from

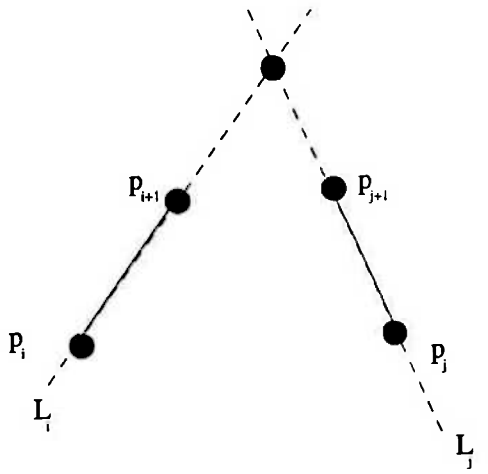


Figure 6.1: λ_{ij}/Δ_{ij} is the cross ratio of p_i, $L_i \cap L_j$, the point at infinity on L_i and p_{i+1}.

real points, we obtain an affine invariant. This suggests how to extend this invariant to full projective invariance: by taking the cross ratio with the fourth point being another tangent intersection.

λ_{ij}/Δ_{ij} is the ratio of two signed distances: the distance from $L_i \cap L_j$ to (x_i, y_i) and the length of the i-th edge. Thus the i-th edge and j-th edge intersect if and only if $\lambda_{ij}/\Delta_{ij} \in [0,1]$ and $\lambda_{ji}/\Delta_{ji} \in [0,1]$.

Equation 6.1 can be expressed in terms of edge vectors:

$$\lambda_{ij} v_i + \lambda_{ji} v_j = \Delta_{ij} \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} \tag{6.3}$$

If $i < j$ $\quad \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} = v_i + \ldots + v_{j-1}$ since $\begin{bmatrix} x_i \\ y_i \end{bmatrix} = v_0 + \ldots + v_{i-1}$

Thus if $i < j$

$$(\Delta_{ij} - \lambda_{ij}) v_i + \Delta_{ij} v_{i+1} + \ldots + \Delta_{ij} v_{j-1} - \lambda_{ji} v_j = \mathbf{0} \tag{6.4}$$

We obtain a system of k^2 such equations. However, not all of them are independent. The relations can be written as a $k^2 \times k$ matrix R acting on a $k \times 2$ matrix of edge vectors from the left, producing zero.

Definition 6.11
For any given matrix $[(\Delta_{ij}, \lambda_{ij})]$,

$$R_{[(\Delta_{ij}, \lambda_{ij})]}$$

denotes the $k^2 \times k$ matrix R whose entries are the coefficients in equation 6.4.

Lemma 6.3
A planar simple non-degenerate k-gon has $rank(R) = k - 2$.

Proof By equation 6.4 R has a submatrix of the form

$$\begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ ? & -\lambda_{21} & 0 & \dots & 0 \\ ? & ? & -\lambda_{31} & \dots & 0 \\ & & & \ddots & \vdots \\ ? & ? & ? & \dots & -\lambda_{k1} \\ ? & ? & -\lambda_{32} & \dots & 0 \\ & & & \ddots & \vdots \\ ? & ? & ? & \dots & -\lambda_{k2} \\ & & \vdots & & \\ 0 & 0 & 0 & \dots & -\lambda_{k(k-1)} \end{bmatrix}$$

The top submatrix is $k \times k$ and lower triangular. This implies that the diagonal entries are pivots. If some of these are zero it is possible to interchange a "compatible" row to bring in a non-zero diagonal entry. $\lambda_{21} = 0, \lambda_{32} = 0, \dots, \lambda_{k(k-1)} = 0$. So the possible row interchanges are:

$\lambda_{41} \leftrightarrow \lambda_{42}$

$\lambda_{51} \leftrightarrow \lambda_{52} \leftrightarrow \lambda_{53}$

$\vdots$

$\lambda_{k1} \leftrightarrow \lambda_{k2} \leftrightarrow \lambda_{k3} \leftrightarrow \dots \leftrightarrow \lambda_{k(k-2)}$

Suppose $\lambda_{ij} = 0$ and $\lambda_{i(j+1)} = 0$. Then lines extending the j-th and (j+1)-th edge intersect (x_i, y_i) or are parallel. Since they are adjacent, they must have a vertex in common, and this must be (x_i, y_i). But they cannot be parallel and adjacent unless there is a degeneracy. The only other possibility is that there is a triple point vertex, which contradicts the assumption that the polygon is simple.

Suppose $\lambda_{31} = 0$. This means L_1 intersects (x_3, y_3), that is the polygon is degenerate.

Hence a simple non-degenerate polygon has rank(R) $\geq$ k $-$ 2.

Now for a closed planar polygon, not less than two of $v_0, \dots, v_{k-1}$ can be linearly independent, so the dimension of the nullspace of R is greater than or equal to 2. Hence rank(R) $\leq$ k $-$ 2.

$\Rightarrow$ rank(R) = k $-$ 2 ∎

The next lemma characterises the set of matrices which represent a simple closed non-degenerate k-gon:

Lemma 6.4
$[(\Delta_{ij}, \lambda_{ij})]$ represents:
(i) a closed k-gon iff rank(R) $\leq$ k $-$ 2 and

$$\forall i \quad \lambda_{i(i-1)} = 0, \lambda_{ii} = 0, \Delta_{ii} = 0, \lambda_{i(i+1)} = \Delta_{i(i+1)}$$

(ii) a non-degenerate closed k-gon where $k \geq 4$ iff (i) and

$$\forall\, j\ \exists i \notin \{j-1, j, j+1\} \quad \begin{vmatrix} \Delta_{ij} & \Delta_{i(j+1)} \\ \lambda_{ij} & \lambda_{i(j+2)} \end{vmatrix} \neq 0$$

hold.

(iii) a non-simple k-gon (assuming (i)) iff $\forall i \notin \{j-1, j, j+1\}$

$$\lambda_{ij} \in [0, \Delta_{ij}] \quad \& \quad \lambda_{ji} \in [\Delta_{ji}, 0] \quad \& \quad \Delta_{ij} \neq 0$$

$$\Leftrightarrow$$

$$\Delta_{ij} - \lambda_{ij} \in [0, \Delta_{ij}] \quad \& \quad -\lambda_{ji} \in [0, \Delta_{ij}] \quad \& \quad \Delta_{ij} \neq 0$$

Proof

(i) If $[(\Delta_{ij}, \lambda_{ij})]$ represents a closed k-gon then rank(R) $\leq$ k $-$ 2 (see lemma 6.3). It follows from definition 5.10, that

$$\forall i \quad \lambda_{i(i-1)} = 0, \lambda_{ii} = 0, \Delta_{ii} = 0, \lambda_{i(i+1)} = \Delta_{i(i+1)}$$

If rank(R) $\leq$ k $-$ 2 then the nullspace of R is a set of vectors which are not all zero or all collinear. If

$$\forall i \quad \lambda_{i(i-1)} = 0, \lambda_{ii} = 0, \Delta_{ii} = 0, \lambda_{i(i+1)} = \Delta_{i(i+1)}$$

then the homogeneous coordinates of vertex intersections are such that consecutive edges intersect at vertices. Hence a closed k-gon results.

(ii) Suppose there is a degeneracy at vertex $j+1$.
$\Rightarrow L_j = L_{j+1}$
$\Rightarrow L_j \cap L_i = L_{j+1} \cap L_i$.

If $\Delta_{ij} = 0$, then also $\Delta_{i(j+1)} = 0$ since $L_j = L_{j+1}$, so

$$\Delta_{ij}\lambda_{i(j+1)} - \Delta_{i(j+1)}\lambda_{ij} = 0.$$

If $\Delta_{ij} \neq 0$ then also $\Delta_{i(j+1)} \neq 0$, and since $L_i \cap L_i = L_{j+1} \cap L_i$

$$\lambda_{ij}/\Delta_{ij} = \lambda_{i(j+1)}/\Delta_{i(j+1)}$$

by geometrical interpretation of cross ratio.

Suppose

$$\exists j\, \forall\, i \notin \{j-1, j, j+1\}\ \Delta_{i(j+1)}\lambda_{ij} = \Delta_{ij}\lambda_{i(j+1)} \tag{6.5}$$

Note that if $k = 3$, the equation 6.5 is vacuously true, hence (ii) does not hold for non-degenerate triangles.

Consider the following cases:

(I) $\Delta_{ij} \neq 0$ and $\Delta_{i(j+1)} \neq 0$.

Rearrange $\Delta_{ij}\lambda_{i(j+1)} = \Delta_{i(j+1)}\lambda_{ij}$ to obtain

$$\lambda_{ij}/\Delta_{ij} = \lambda_{i(j+1)}/\Delta_{i(j+1)}$$

(II) $\Delta_{ij} = 0$ and $\Delta_{i(j+1)} = 0$.

$\Rightarrow$ i-th edge collinear with j-th edge and i-th edge collinear with $(j+1)$-th edge.

$\Rightarrow$ degeneracy at $(j+1)$-th vertex since it is in both the j-th edge and the $(j+1)$-th edge or coincidence of vertices.

(III) $\Delta_{ij} = 0$ and $\lambda_{ij} = 0$ (similarly $\Delta_{i(j+1)} = 0$ and $\lambda_{i(j+1)} = 0$)

$$\lambda_{ij} = \begin{vmatrix} y_{j+1} - y_j & y_j - y_i \\ x_{j+1} - x_j & x_j - x_i \end{vmatrix}$$

$\therefore \lambda_{ij} = 0 \Rightarrow$ Vector joining i-th vertex to j-th vertex and j-th edge vector have zero determinant.

$\Delta_{ij} = 0$ and $\lambda_{ij} = 0 \Rightarrow$ Vertices $i, i+1, j, j+1$ collinear or coincidental.

(IV) $\lambda_{ij} = 0$ and $\lambda_{i(j+1)} = 0$

$\Rightarrow$ Line joining i-th vertex to j-th vertex and line joining i-th vertex to $(j+1)$-th vertex are both parallel to i-th edge.

$\Rightarrow$ Vertices $i, i+1, j+1$ collinear or coincidental of (some of these) vertices.

By equation 6.5 for all $i \notin \{j-1, j, j+1\}$ one of (I)-(IV) hold.

If $k = 4$ then

$$i = j + 2 = j - 2 \pmod 4 \text{ is unique}$$

(I) $\Rightarrow$ L_i contains vertex $j+1$ but $i = j+2$ so there is a degeneracy.

If (II)-(IV) hold there is obviously a degeneracy.

If $k > 4$ then (I)-(IV) hold for some $i, i+1$.

If (II) holds there is a degeneracy at $(j+1)$-th vertex.

If (II) or (IV) hold for i and $(i+1)$-th vertices then vertices $i, i+1, j, j+1$ collinear or some coincident and vertices $i+1, i+2, j+1, j$ collinear or some coincident.

$\Rightarrow$ Vertices $i, i+1, i+2$ collinear or some coincidence.
$\Rightarrow$ Degeneracy at vertex $i+1$ or vertex coincidence degeneracy.

If (I) hold for i and (III) or (IV) hold for $(i+1)$ then vertices $i, i+1, j+1$ collinear by (I) and vertices $i+1, i+2, j, j+1$ collinear by (III) or (IV) (or there is a vertex coincidence degeneracy).

$\Rightarrow$ Degeneracy at vertex $i+1$ or vertex coincidence degeneracy.

Similarly if (III) or (IV) hold for i, (I) holds for $i+1$.
If (I) holds for i and $i+1$ then vertices $i, i+1, j+1$ collinear and vertices $i+1, i+2, j+1$ collinear.

$\Rightarrow$ Degeneracy at vertex $i+1$.

(iii) Follows from definition 5.10. ∎

Definition 6.12
Let P be a non-degenerate closed oriented k-vertex polygon. The sign of the i-th vertex is 0 if $\Delta_{i(i+1)} < 0$, and 1 if $\Delta_{i(i+1)} > 0$. The sign of the i-th vertex may also be denoted by $-$ if $\Delta_{i(i+1)} < 0$ and $+$ if $\Delta_{i(i+1)} > 0$. A binary string descriptor of P is a list of the signs of consecutive vertices.

Binary string descriptors of polygons which are simple (as well as satisfying the requirements of the definition above) can be normalised as follows: write them in binary notation. Rotating them yields the same binary coded binary string descriptor, hence rotating until a number of least magnitude is obtained, yields a normalised binary string descriptor.

$(\Delta_{ij}, \lambda_{ij})$ can also be regarded as coordinates for a polygon in a space of polygons. We define and study such a space.

Definition 6.13
Let E_k be the set of tuples $(v_0, \ldots, v_{k-1})$ of edge vectors of simple closed oriented planar non-degenerate k-vertex polygons. Let E_k^s be the subset of E_k whose binary string descriptor is s. Let L_i be the unique line intersecting (x_i, y_i) and (x_{i+1}, y_{i+1}).

Definition 6.14
Let $\pi : (v_0, \ldots, v_{n-1}) \mapsto [(\Delta_{ij}, \lambda_{ij})]$, $\pi : E_k \to \mathbf{R^{2k^2}}$.
Let $U_s = \pi(E_k^s)$.
Let $\pi(x) \equiv \pi(y)$ if $\mathcal{N}(R_{\pi(x)}) = \mathcal{N}(R_{\pi(y)})$.
$RX_k = Im(\pi)/\equiv$ and RX_k has the quotient topology.
$RX_k^s = \pi(E_k^s)/\equiv$.
$a = (a_1, \ldots, a_{k-2})$ be pivot indices for $R_{\pi(x)}$.
$\phi_a : RX_k^s \to \mathbf{R^{2(k-2)}}, \mathbf{x} \mapsto (\mathbf{\Delta_{a_1}}, \ldots, \mathbf{\Delta_{a_{k-2}}}, \lambda_{a_1}, \ldots, \lambda_{a_{k-2}})$

We later prove that RX_k is a $2(k-2)$ dimensional real manifold with coordinate maps ϕ_a as long as R is not constant (which happens only when $k = 3$).

Lemma 6.5
If $\pi(v_0, \ldots, v_{k-1}) = \pi(w_0, \ldots, w_{k-1})$ where $v_0, \ldots, v_{k-1}$ and $w_0, \ldots, w_{k-1}$ are edge vectors of simple closed non-degenerate polygons then $\exists g \in GL(2) \quad (gw_0, \ldots, gw_{k-1}) = (v_0, \ldots, v_{k-1})$.

Proof

$\pi(v_0, \ldots, v_{k-1}) = \pi(w_0, \ldots, w_{k-1})$ implies that $v_0, \ldots, v_{k-1}$ and $w_0, \ldots, w_{k-1}$ have the same matrix R.

$$R \begin{bmatrix} v_0 \\ \vdots \\ v_{k-1} \end{bmatrix} = \mathbf{0} = R \begin{bmatrix} w_0 \\ \vdots \\ w_{k-1} \end{bmatrix}$$

It follows from lemma 6.3 that if we choose the vectors v_i, v_j and w_i, w_j corresponding to free variables, then the remaining vectors are determined. In the canonical form of R in lemma 6.3, v_0, v_1 correspond to free variables, and since they are consecutive and the polygon is non-degenerate, these two vectors cannot be collinear. Hence $\exists g \in GL(2) \quad gw_i = v_i, gw_j = v_j$. Also for any $k \in \{0, \ldots, k-1\} \quad gw_k = v_k$ since the remaining vectors are determined. Hence $\exists g \in GL(2) \quad (gw_0, \ldots, gw_{k-1}) = (v_0, \ldots, v_{k-1})$. ∎

So the coordinates of RX_k are affine invariants. Furthermore, since the orientation of the polygon cannot be reversed (reflection is not an equivalence) only $g \in GL(2) \quad det(g) > 0$ are in the structure group. We define

$$GL_+(m) = \{g \in GL(m) | \det(g) > 0\}$$

$$GL_-(m) = \{g \in GL(m) | \det(g) < 0\}$$

It follows that $RX_k = E_k/GL_+(2)$.

Theorem 6.6
For each binary string descriptor s E_k^s is path connected.

Proof

We show there exist homeomorphisms ϕ_s such that the following diagram commutes:

$$\begin{array}{ccc} E_k^s & \xrightarrow{\phi_s} & U_s \times GL_f(2) \\ & \searrow^{\pi} \quad \swarrow_{\pi_1} & \\ & U_s & \end{array}$$

From Lemma 6.3 it follows that v_0, v_1 correspond to free variables. The sign of the vertex between v_0, v_1 given by the binary string descriptor s fixes a sign f. Since the polygon is non-degenerate, $v_0, v_1 \in GL_f(2)$. The other vectors are determined by the equivalence class to which the polygon belongs, by Lemma 6.5. $\phi_s : x \mapsto (\pi(x), v_0, v_1)$ is a continuous bijection, and the diagram commutes.

It remains to show that RX_k^s is connected.

Let A_s be the set of $[(\Delta_{ij}, \lambda_{ij})]$ matrices such that

(i) $\forall\, i \; \lambda_{i(i-1)} = 0, \lambda_{ii} = 0, \Delta_{ii} = 0, \lambda_{i(i+1)} = \Delta_{i(i+1)}$.

(ii) $\forall\, j \; \exists i \notin \{j-1, j, j+1\} \begin{vmatrix} \Delta_{ij} & \Delta_{i(j+1)} \\ \lambda_{ij} & \lambda_{i(j+1)} \end{vmatrix} \neq 0.$

(iii) $\forall\, i \in \mathbb{Z}_k \; \Delta_{i(i+1)}$ has fixed sign depending only on s.

Then (ii) asserts that for every column of $[(\Delta_{ij}, \lambda_{ij})]$ there is some entry other than $(j, j-1)$, (j, j) and $(j, j+1)$ which is not parallel to its right hand neighbour. Thus each column of a matrix in A_s lies in

$$A \times \mathbf{R}^{-s_i} \times \{\mathbf{0}\} \times \mathbf{R}^{s_i} \times \mathbf{R}^{s_i} \times \{(\mathbf{0}, \mathbf{0})\}$$

where $A = \{(a_1, \ldots, a_{k-2}) | Re \; a_1 \neq 0$ or $(a_1, a_2)^{\top} \in GL(2)$ or $\ldots$ or $(a_{k-3}, a_{k-2}) \in GL(2)\}$ because

$\Delta_{i(i-1)} = -\Delta_{(i-1)i}$ which has fixed sign $-s_{i-1}$.
$\lambda_{i(i-1)} = 0$.
$(\Delta_{ii}, \lambda_{ii}) = (0,0)$.
$\Delta_{i(i+1)} \in \mathbf{R}^{s_i}$, $\lambda_{i(i+1)} = \Delta_{i(i+1)} \Rightarrow \lambda_{i(i+1)} \in \mathbf{R}^{s_i}$.

When $i = j+2, j+1 = i-1$ so $\lambda_{i(j+1)} = \lambda_{i(i-1)} = 0$ so

$$\begin{vmatrix} \Delta_{ij} & \Delta_{i(j+1)} \\ \lambda_{ij} & \lambda_{i(j+1)} \end{vmatrix} = \begin{vmatrix} \Delta_{ij} & \Delta_{i(i-1)} \\ \lambda_{ij} & 0 \end{vmatrix} \neq 0$$

gives Re $a_1 \neq 0$. When $i \notin \{j-1, j, j+1, j+2\}$ condition (ii) gives $(a_l, a_{l+1}) \in GL(2)$.

A is connected since $A = A_1^+ \cup A_1^- \cup A_2^+ \cup A_2^- \cup \ldots \cup A_{k-2}^+ \cup A_{k-2}^-$ where

$$\begin{aligned} A_1^+ &= \{z \in \mathbf{C}^{k-2} \mid z_1 > 0\} \\ A_1^- &= \{z \in \mathbf{C}^{k-2} \mid z_1 < 0\} \\ A_2^+ &= \mathbf{C} \times GL_+(2) \times \mathbf{C}^{k-5} \\ A_2^- &= \mathbf{C} \times GL_-(2) \times \mathbf{C}^{k-5} \\ &\vdots \\ A_{k-2}^+ &= \mathbf{C}^{k-4} \times GL_+(2) \\ A_{k-2}^- &= \mathbf{C}^{k-4} \times GL_-(2) \end{aligned}$$

and

$$\begin{array}{llll} A_1^+ \cap A_2^+ \neq \emptyset & \text{and} & A_1^+ \cap A_2^- \neq \emptyset & \text{and} \\ A_1^+ \cap A_3^+ \neq \emptyset & \text{and} & A_1^+ \cap A_3^- \neq \emptyset & \text{and} \\ \vdots & & \vdots & \\ A_1^+ \cap A_{k-2}^+ \neq \emptyset & \text{and} & A_1^+ \cap A_{k-2}^- \neq \emptyset & \text{and} \\ A_1^- \cap A_2^+ \neq \emptyset & \text{and} & A_1^- \cap A_2^- \neq \emptyset & \text{and} \\ \vdots & & \vdots & \\ A_1^- \cap A_{k-2}^+ \neq \emptyset & \text{and} & A_1^- \cap A_{k-2}^- \neq \emptyset & \end{array}$$

and $A_1^+, \ldots, A_{k-2}^+, A_1^-, \ldots, A_{k-2}^-$ are open connected sets.

Thus A_s is connected. □

The set of indices of possible pivots of $R_{\pi(x)}$ is

$$P_R = \{(j,i) \mid 0 \leq i \leq j-2,\ 2 \leq j \leq k-1\}$$

because the pivots are of the form $-\lambda_{ji}$, where $i < j$, and if $i = j-1$, $-\lambda_{ji} = 0$.

Let $A_s^a = \{x \in A_s \mid \lambda_{a_1} \neq 0, \ldots, \lambda_{a_{k-2}} \neq 0\}$.

Let $a \in P_R^{k-2}$, so $\bigcup_{a \in P_R^{k-2}} A_s^a = A_s$ since for all j there is an $i \notin \{j-1, j, j+1\}$ such that

$$\begin{vmatrix} \Delta_{ij} & \Delta_{i(j+1)} \\ \lambda_{ij} & \lambda_{i(j+1)} \end{vmatrix} \neq 0,$$

so if $\lambda_{ij} = 0$ and $\lambda_{i(j+1)} = 0$ the determinant is zero, so one of these must be non-zero.

Let $q_{ij} : A_s \to A_s$,

$$q_{ij} : (\Delta_{kl}, \lambda_{kl}) \mapsto \begin{cases} (-\Delta_{kl}, -\lambda_{kl}) & \text{if } (k,l) = (i,j) \text{ or } (j,i) \\ (\Delta_{kl}, \lambda_{kl}) & \text{otherwise.} \end{cases}$$

Then $q_{ij}(A_s^a) = A_s^a$.

Also $\mathcal{N}(R_{\pi(q_{ij}(x))}) = \mathcal{N}(R_{\pi(x)})$, and $A_s/\{q_{ij}\}$ and $A_s^a/\{q_{ij}\}$ are connected.

We can define projections p_a of A_s^a to U_s. This is because if pivots are chosen, then linear combinations of the pivotal rows produce non-pivotal rows, and hence non-pivotal rows can be mapped to zero. Let U_s^a denote the subset of U_s whose matrices R have the pivots a. Then this subset is connected (in the subspace topology), because p_a is continuous.

Let a, b be choices of pivot indices for R-matrices. If $x \in A_s^a \cap A_s^b$ then $\mathcal{N}(R_{p_a(x)}) = \mathcal{N}(R_{p_b(x)})$. Hence a separation of RX_k^s would induce a separation of A_s, contradicting the connectedness of A_s. Thus RX_k^s is connected in the quotient topology. Moreover, since a quotient map is continuous, U_s is also connected.

Hence E_k^s fibres over U_s with fibre space $GL_f(2)$ and structure group $GL_+(2)$. Since $GL_f(2)$ is path connected and U_s is path connected, the product is also path connected. And homeomorphisms preserve connectedness. A connected manifold is path connected. ∎

Chapter 7

A grammar of shapes

Structural pattern recognition describes pictures with strings, and compares them by matching strings. The most popular string descriptor is the chain code, due to Freeman [58], see also [25] and [24]. One approach to recognising coplanar (flat) shapes was to match string descriptors by minimising the number of moves such as insertions and deletions [173], [51], [118]. The author realised that certain strings and moves have geometrical significance. Some moves were not previously recognised, and other moves should be illegal because they cannot be realised geometrically. In chapter 3 we described an algorithm that generates polygons from images that are boundaries of visible surface patches. In this chapter we will define a string descriptor that is an important invariant to viewpoint of such polygons on a non-transparent two or three dimensional object. There is a deep connection between this combinatorial geometric theory and the theory of shape spaces. A highly developed combinatorial analysis of polyhedra has been developed in the theory of line drawings [171]. The main results of this chapter appeared in the author's paper [11]. However, more concise proofs are supplied here where practicable.

This chapter studies a class of deformations of polygons. A continuous sequence of un-occluded images of a suitable scene polygon on a non-transparent surface is a deformation in this class. We will see that there are certain strings which can be computed from the images that are invariant to such deformations. We will use theorem 6.6 to prove that if the strings of two polygons are equal, then there is such a deformation from one polygon to the other. We will develop a grammar on these strings whose rules describe the possible changes to a string when an occluded vertex becomes visible. This leads to an algorithm in chapter 8 to decide whether two given strings are derivable from each other in this grammar.

7.1 Formal languages and grammars

This section introduces the formal definitions of the concepts associated with grammars (see [148] for example). Note however, that we introduce an unusual notion called a strict-production, for reasons that will be made clear in this section. The reader who is already familiar with automata theory may wish to skip this section.

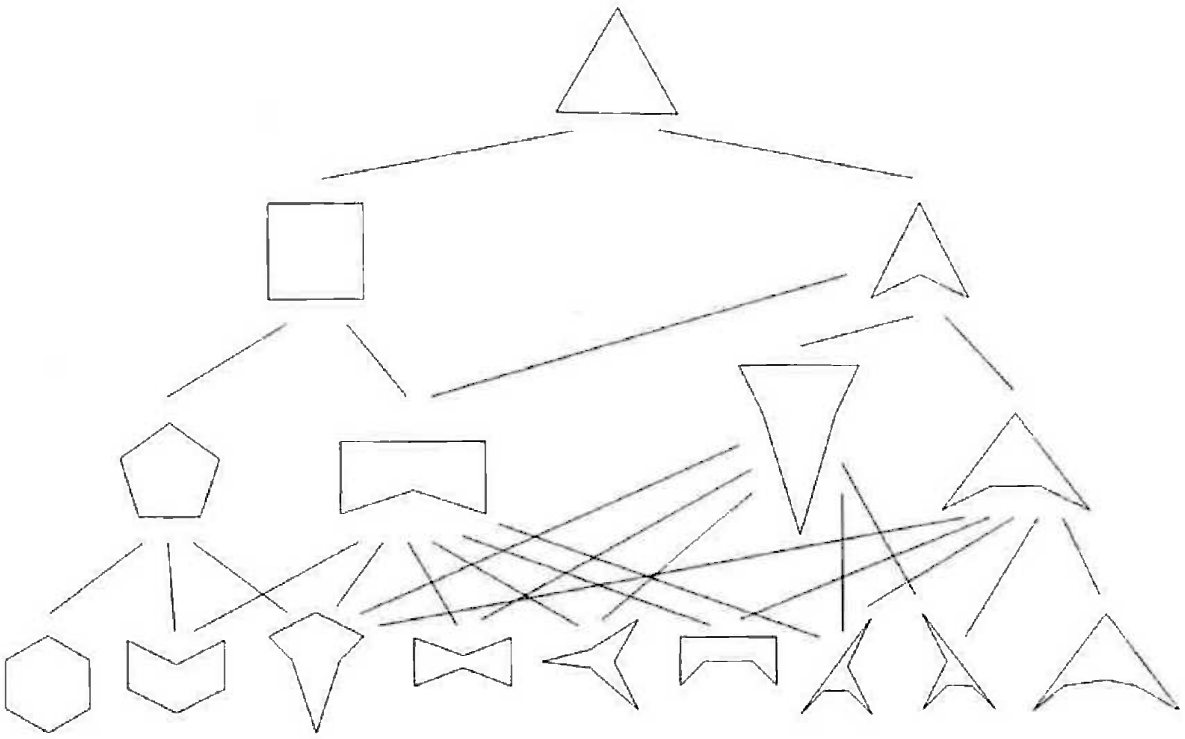


Figure 7.1: The grammar for polygons with up to 6 vertices

The next definition is used in the standard definition of an alphabet, but holds trivially for the grammar to be introduced in this chapter.

Definition 7.1
A relation $\leq$ over a set A is called a partial order if for all $a, b, c \in A$

- (i) $a \leq a$; and
- (ii) $a \leq b$ and $b \leq a$ implies $a = b$; and
- (iii) $a \leq b$ and $b \leq c$ implies $a \leq c$.

A total order is a partial order such that for any $a, b \in A$ either $a \leq b$ or $b \leq a$.

Example 7.1
The only example of a total order we are interested in is $0 \leq 1$. □

Definition 7.2
An alphabet is a nonempty, totally ordered, finite set of symbols that are going to appear in our strings. A string is a finite sequence of symbols from an alphabet. The length is the number of symbols occurring in the string; if the symbol is duplicated n times it contributes n to the length.

Example 7.2
The alphabet $\Sigma = \{0, 1\}$ will be used in this chapter. 101101 is an example of a string over this alphabet. □

Example 7.3
The alphabet $\Sigma = \{-, +\}$ will also be used in this chapter. In fact, we will use $-$ synonymously with 0 and $+$ synonymously with 1 in the previous example. $+-++-+$ is an example of a string over this alphabet, in fact it is the same string as in the previous example. □

Definition 7.3
A language over an alphabet is any subset of the set of all finite length strings over the alphabet. A terminal is a symbol in the language. A non-terminal is a variable, whose symbol does not occur in the language.

Definition 7.4
A production

$$\alpha \to \beta$$

is a rule such that any string of terminals and non-terminals of the form $\gamma_1\alpha\gamma_2$ can be replaced by a string of the form $\gamma_1\beta\gamma_2$.

We will now introduce the idea of a strict production $\alpha \to \beta$, in which it is not allowed to apply the rule if there are non-empty strings to the left or right of α.

Definition 7.5
A strict production

$$\alpha \to \beta$$

is a rule such that any string of terminals and non-terminals of the form α can be replaced by the string β.

Note however that there is no loss of generality in using strict-productions in place of productions, since any production $\alpha \to \beta$ is equivalent to the strict production

$$\gamma_1\alpha\gamma_2 \to \gamma_1\beta\gamma_2$$

where γ_1, γ_2 are non-terminals introduced for this purpose.

Example 7.4
In this chapter, we consider strings that can be obtained from each other by some form of rotation to be equivalent. For example, 1000 would be considered equivalent to 0100, 0010 or 0001. The strict productions

$$x0 \to 0x$$

and

$$x1 \to 1x$$

express this.

However, the production $x1 \to 1x$, which is equivalent to the strict production

$$yx1z \to y1xz$$

permits invalid moves, such as $x = 0$, $y = 1$, $z = 0$ giving $1010 \to 1100$. This does not represent a rotation, explaining why we use a strict production. □

Definition 7.6
A grammar (N, T, P, S) consists of

- a finite set of non-terminals N,
- a finite set of terminals T disjoint from N,
- a finite set of productions and strict productions P, each of the form

$$\alpha \to \beta$$

  where α is a non-empty string of terminals and non-terminals, whearas β is a possibly empty string of terminals and non-terminals,

- $S \in N$ is a symbol designated as the start symbol of the grammar.

We will sometimes abuse notation by using the term productions of a grammar to mean P, consisting of both the productions and the strict productions of that grammar.

The grammar we are interested turns out to have no non-terminals except the start symbol and x. Its terminals are 0 and 1 or $-$ and $+$ respectively.

Definition 7.7
If there is a sequence of productions in a grammar that can transform the string α to the string β then β is said to be derivable from α in this grammar.

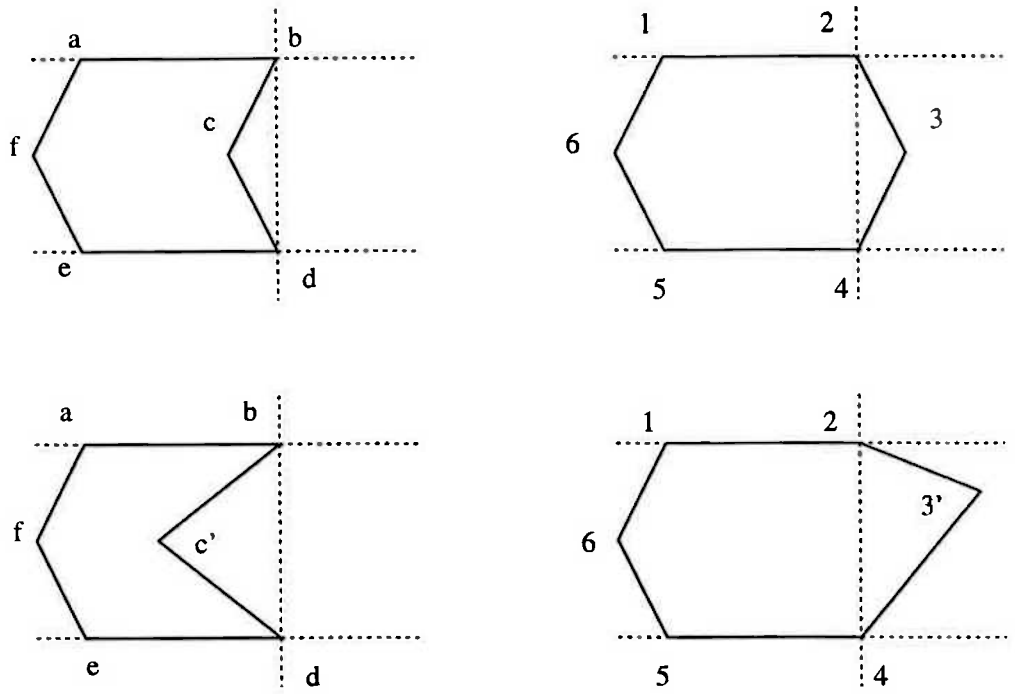
Example 7.5
The string 1100 is not derivable from 1000 in the grammar (N, T, P, S) where $N = \{S, x\}$, $T = \{0, 1\} \times \{0, 1\} \times \{0, 1\} \times \{0, 1\}$, $P = \{x \to x0, x \to x1$. □

7.2 A language of polygon shapes

In this section we will develop a formal language whose strings are computed from polygons in images, provided the polygons have certain properties formally introduced in chapter 6. A more intuitive description of such a polygon is that it is a list of points (vertices) joined by straight edges that do not cross other edges (simple); lead back to the starting point (closed); it is not possible to rub out two consecutive edges and put one edge through three vertices (non-degenerate); the edges are either listed clockwise or anticlockwise, it is not allowed to be reversed (oriented).

We will give a formal definition of deformations of polygons that do not at any intermediate stage make a vertex cross an edge between two other vertices, that is they preserve non-degeneracy at all times. The deformation in figure 7.2(a) and figure 7.2(b) is such a deformation, whereas a deformation of a square to a triangle is not.

The formal definition is based on topology, that is used to prove that strings in a certain language can be regarded as shapes in the same sense as Euclidean, affine and positive affine shapes.



(a)

abcdef equivalent to abc'def

(b)

123456 equivalent to 123'456

Figure 7.2: Examples of step-equivalent polygons

A homeomorphism is a continuous function with a continuous inverse. An isotopy is a continuous one parameter family of homeomorphisms $h_t : [0,1] \rightarrow \mathbf{R}^2$. A PL-isotopy is an isotopy such that h_t is piecewise linear for each t.

Definition 7.8
Two simple closed polygons $h_0, h_1 : [0,1] \rightarrow \mathbf{R}^2$ are equivalent if there is a PL-isotopy $h_t : [0,1] \rightarrow \mathbf{R}^2$ such that for all $t \in [0,1]$:

(i) dh_t/ds is discontinuous at precisely the same points as dh_0/ds. (ie vertices and edges are preserved.)

(ii) Consecutive vertices $h_t(s_1), h_t(s_2), h_t(s_3)$ are never collinear. (ie non-degeneracy is preserved.)

Example 7.6
Rotation, translation, scaling, and positive affine transformations preserve equivalence. This is because they can be expressed by a continuous parameter, and preserve collinearity. □

A much more important example of equivalence is a continuous sequence of images of a simple closed non-degenerate polygon on a surface that is not transparent, such that at no stage is any vertex or pair of consecutive edges occluded. A pair of consecutive edges can become collinear in an image if and only if the plane spanned by the two edges intersects the optical centre of a camera. If the image of a vertex passes through

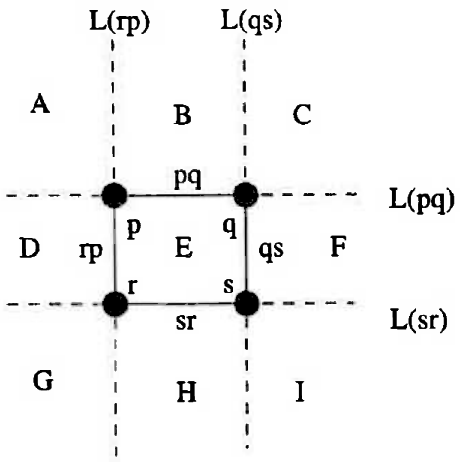


Figure 7.3: Arrangement generated by lines extending a square

a degeneracy like vertex c in figure 7.2(a) to vertex 3 in figure 7.2(b), it must be viewed from the other side of the plane spanned by the two edges incident to it (in the scene). If the surface is not transparent, this is not possible.

Definition 7.9
Non-degenerate simple closed polygons are step-equivalent if there is a PL-isotopy h_t mapping one to the other, such that for all $t \in [0,1]$:
(i) all edges remain edges; and
(ii) h_t preserves non-degeneracy; and
(iii) all vertices except one remain fixed.

Example 7.7
Consider the square with vertices p, q, s, r in figure 7.3. The lines $L(pq), L(qs), L(sr), L(rp)$ extending edges pq, qs, sr and rp respectively are drawn as dashed lines in figure 7.3. These lines dissect the plane into four points p, q, s, r, twelve line segments (three on each line), and nine cells $A, B, C, D, E, F, G, H, I$. □

The example illustrates that a set of lines in the plane dissect it into connected pieces of dimension zero, one and two, which is formalised by the notion of an arrangement[47] below.

Definition 7.10
Let L be a set of lines in the plane $\mathbf{R}^2$. Let P be the set of points where two or more lines in L intersect. Let L' be the set of connected components of $\{(x,y) \in l \mid l \in L\} - P$. Let C be the connected components of $\mathbf{R}^2 - \{(x,y) \in l \mid l \in L\}$. Each member of C is called a cell. The set consisting of the points in P, the segments in L', and the cells in C is called the arrangement generated by L.

A transformation preserving equivalence permits simultaneous movement of vertices. We will prove that the same thing can be accomplished one step at a time. The reader will need to recall the definition of a compact set in chapter 3. The proof also uses the following property of compact sets.

Lemma 7.1

If $f : X \to Y$ is continuous and X is compact then $f(X)$ is compact.

Proof

Let $\mathcal{A}$ be a collection of sets covering Y. Then $\mathcal{B} = \{f^{-1}(A) \mid A \in \mathcal{A}\}$ is a collection of sets covering X. From the definition of compactness, it follows that there is a finite subcollection, say $f^{-1}(A_1), \ldots, f^{-1}(A_n)$ of $\mathcal{B}$ covering X. Then the sets $A_1, \ldots, A_n$ cover $f(X)$. By definition of compactness, $f(X)$ is compact. ∎

Lemma 7.2
Two simple closed nondegenerate n-gons P_1, P_2 are equivalent if and only if there is a finite sequence of simple closed nondegenerate n-gons $S_1, \ldots, S_k$ such that P_1 is step equivalent to S_1, that is step equivalent to S_2 and so on, where S_k is step equivalent to P_2.

Proof

Since composition of PL-isotopies are PL-isotopies, and step-equivalences satisfy the conditions of equivalence, a finite sequence of step-equivalences is an equivalence.

It remains to prove that an equivalence can be accomplished with a finite sequence of step-equivalences. An equivalence is characterised by the paths of the vertices. Hence it is characterised by a continuous function $f : [0,1] \to \mathbf{R}^{2n}$. It suffices to prove that we can construct a step-path γ (ie a path consisting of finitely many line segments parallel to the coordinate axes) satisfying:

(*) For all $s \in [0,1]$, no three consecutive components of $\gamma(s)$ are collinear.

Now for each $t \in [0,1]$ each vertex can be moved within an open cell in the arrangement induced by the lines joining consecutive vertices. Hence there is an open subset D of $\mathbf{R}^{2n}$ containing $f([0,1])$ such that (*) is satisfied. Choose an open cover of the path f consisting of spheres. Since $[0,1]$ is compact and f is continuous, $f([0,1])$ is compact. By definition of compactness, there is a finite subcover of $f([0,1])$ in D. There is obviously a step-path in a finite set of overlapping spheres. ∎

Definition 7.11
Let P_1, P_2 be simple closed nondegenerate n-gons that coincide at all except one vertex. P_1 and P_2 are called combinatorially-equivalent if the vertices at which they differ belongs to the same connected component of the tessellation of the plane induced by the polygon formed by skipping the common vertex.

Example 7.8
The two polygons 123456 and 123′456 in figure 7.2(b) are combinatorially equivalent. They differ at only one vertex, 3 and 3′ respectively, which both belong to the same cell of the arrangement induced by the polygon 12456. This cell is bounded by the dashed lines in figure 7.2(b). Note that although the lines L_5 and L_6 are not drawn in figure 7.2(b), these do not divide the cell of the arrangement induced by 12456 to which vertices 3 and 3′ belong. □

Lemma 7.3
Two polygons are step equivalent if they are combinatorially equivalent.

We gave a formal definition of binary string descriptors (of non-degenerate simple closed polygons) in chapter 6; they are computed as follows: Follow the edges, writing down zeroes and ones for each vertex in turn. If the sign of the determinant between the incoming and outgoing edge vector is positive, write down one. Otherwise, the sign is negative (since the polygon is non-degenerate it cannot be zero) write zero. As it stands the string we obtain is dependent on which vertex was first. If the next vertex was the first vertex the leftmost symbol would become the rightmost symbol, that is the string would be rotated. Each such string can be regarded as a binary number, since it is a string of digits 0 and 1. We normalise the string against starting point by rotating it until a number of least magnitude is obtained, this being the normalised string. For example, in figure 7.1, the triangle has string 000. All rotations of 000 leave it unchanged so the string is already normalised. The four vertex polygons below it (in figure 7.1) have strings 0000 and 0001 respectively, and the polygons with five vertices have strings 00000, 00001, 00101 and 00011 respectively. Two binary string descriptors are the same if they have the same length and can be obtained by rotation from each other. 0001 is the same descriptor as 0010, 0100 and 1000. The corresponding binary numbers are 1, 2, 4 and 8 respectively. Thus 0001 is the normalised descriptor. This idea of normalisation is due to Freeman.

Example 7.9
We have not yet encountered an example that the orientation (it is either clockwise or anticlockwise) matters. Consider the two polygons second and third from the right on the bottom level of figure 7.1. Their normalised binary string descriptors are 001011 and 001101, and their binary coded descriptors are 11 and 13 respectively. These are different descriptors. The polygon with descriptor 001011 cannot be deformed to its mirror image without passing through a degeneracy at some intermediate stage. □

Different images of the same contour will differ in the number of points contained in that contour's image. For example, the image of a triangle consists of a finite number of points. Unless the triangle is extremely small or far away, the image contains many points that are not corners of the triangle. If the triangle is moved further away, its image reduces in size, so there are less image points. Other changes in pose can also change the number of points. A binary string descriptor is not meaningful at a degenerate vertex. The number of intervals to test collinearity upon is exponential in the number of vertices. Consequently, it is not feasible to compute binary string descriptors directly from edge polygons (generated by graddisc in chapter 3). When we study the correspondence problem in chapter 8, an efficient algorithm will become apparent. This will also reveal that even when an interval to test collinearity is given, the problem is not as straightforward as it seems. Some suitable collinearity tests will be introduced in chapter 10.

The following theorem demonstrates that we have a third, and equivalent, definition of equivalence.

Theorem 7.4
Two simple closed nondegenerate n vertex polygons with the same orientation are equivalent if and only if they have the same (normalised) binary string descriptor.

Proof

A binary string descriptor is an invariant of step equivalence, because to change the vertex sign between + and - requires crossing the line joining neighboring vertices, which violates step equivalence.

By lemma 7.2, any invariant of step equivalence is an invariant of equivalence. Hence two simple closed non-degenerate n vertex polygons have the same binary string descriptor if they are equivalent.

Suppose two polygons in E_k have the same binary string descriptor s. Then they both belong to E_k^s. From theorem 6.6 E_k^s is path connected. A path in E_k^s specifies a PL-isotopy satisfying the conditions stipulated in the definition of equivalence. Hence the two polygons are equivalent. ∎

The theorem asserts that if two polygons have distinct strings they cannot be deformed into each other without making a vertex cross an edge between two other vertices. More surprisingly, it asserts that if two polygons have the same string then there is some transformation preserving equivalence mapping one polygon to the other.

We have now shown that certain strings correspond to shapes of polygons, in a similar sense to our shape spaces. Moreover these strings correspond to cells in a positive affine shape space, with degenerate and self-intersecting shapes excised. But we have not yet identified which strings can be the binary string descriptors of polygons. We will do this in the next section.

For computational purposes binary string descriptors can be written as binary numbers, with 0 for - and 1 for +. This will be called a binary coded descriptor. The C program bcode.c converts a polygon string descriptor in string (of + and -) form to numeric form, whereas scode.c converts the numeric form into the string form (see appendix B).

Lemma 7.5
Algorithm for calculating all binary coded descriptors equivalent to the given one.

If x_1 is a binary coded descriptor, then the set of all equivalent binary coded descriptors is $\{x_1, \ldots, x_n\}$ where

$x_{i+1} = 2x_i$ if $x_i < 2^{n-1}$

$x_{i+1} = 2(x_i - 2^{n-1}) + 1$ otherwise.

Proof A left rotation is a multiplication by 2 as long as there is no overflow. An overflow only occurs if the result of multiplication by 2 is greater than or equal to 2^n. And $2x_i \geq 2^n \Leftrightarrow x_i \geq 2^{n-1}$.

If $x_i \geq 2^{n-1}$ then the leading (highest magnitude) bit is 1. And $1x \rightarrow x1$ is the same as $1x \rightarrow 2(1x - 2^{n-1}) + 1$.

All binary string descriptors equivalent to the given one can be obtained by rotation. ∎

Lemma 7.6

Let $f(x) = \begin{cases} 2x, & \text{if } x < 2^{n-1} \\ 2x - 2^n + 1, & \text{if } 2^{n-1} \leq x < 2^n \end{cases}$

Then $f(x) = (2x)\text{mod}(2^n - 1)$.

Proof

$x < 2^{n-1} \Rightarrow 2x \leq 2^n - 2 < 2^n - 1 \Rightarrow 2x = (2x)\text{mod}(2^n - 1)$

if $x \geq 2^{n-1}$ and $x \leq 2^n - 1$ then

$x \leq 2^n - 1 \Rightarrow 2x \leq 2^{n+1} - 2 \Rightarrow 2x - 2^n + 1 \leq 2^{n+1} - 2^n + 1 - 2 = 2^n - 1$

$\Rightarrow 2x - 2^n + 1 = (2x)\text{mod}(2^n - 1)$ ∎

The program normalizeb.c (see appendix B) normalises a binary string descriptor.

7.3 A grammar of polygon shapes

Suppose a polygon with k vertices has binary string descriptor x. Then the possible binary string descriptors of a polygon obtained from it by a continuous deformation that adds one vertex depends only on x:

Lemma 7.7
If x, y are simple closed non-degenerate n-gons then $x \sim y \Rightarrow xP_{n+1} \sim yP_{n+1}$, where xP_{n+1} and yP_{n+1} are extensions of x, y respectively by the same edge (if P_{n+1} belongs to the cell C of the arrangement induced by x, and belongs to the cell D of the arrangement induced by y, then C and D are the same in the sense that the lines bounding C are mapped to the lines bounding D.)

Proof

By lemma 7.2 we can perform the deformation of x to y stepwise. At each step there are at most two new lines in the arrangement. Therefore after m steps the arrangement has at most $n + 2m$ new lines. This arrangement is a refinement of both the arrangement generated by x and the arrangement generated by y, because we can move the entire plane rather than shift P_1, P_n (without loss of generality assume P_nP_{n+1}, $P_{n+1}P_1$ extended edges.)
If P_{n+1} does not lie in a cell of the refined arrangement, choose any cell of the refined arrangement containing the cell (of the arrangement induced by x) to which P_{n+1} belongs, and perturb P_{n+1} into it.
Thus $xP_{n+1} \sim yP_{n+1}$. ∎

Consider a simple closed polygon $P_1, \ldots, P_n$. We label the interior angle at P_1 with α_1, the interior angle P_2 with α_2, etc. When we extend this polygon, we add a new vertex P_{n+1}. We delete the edge joining P_1 and P_n, and add two edges; joining P_n to P_{n+1} and joining P_{n+1} to P_1. Consequently, the interior angle α_1 is changed to α_1', and the interior angle α_n is changed to α_n'. And the new vertex P_{n+1} has a new interior angle α_{n+1}.

$L(ab)$ denotes the (unique) line joining the points a and b. Observe that a simple closed polygon separates the plane, so both angles are on the same side of edge P_1P_n. Also, the sign of α'_1 corresponds to P_{n+1} lying in a particular halfplane bounded by $L(P_1P_2)$. The sign of α'_n corresponds to P_{n+1} lying in a particular halfplane bounded by $L(P_{n-1}P_n)$. And the sign of α'_{n+1} corresponds to P_{n+1} lying in a particular halfplane bounded by $L(P_1P_n)$. Hence the choice of signs corresponds to P_{n+1} lying in a particular cell of the arrangement generated by $L(P_1P_2), L(P_1P_n), L(P_nP_{n+1})$.

Lemma 7.8
The transformation rules for binary string descriptor when a vertex is added (extending a polygon) are:
Rule 1: $xy \rightarrow x - y$
Rule 2: $xy \rightarrow x + y$
Rule 3: $xy \rightarrow \bar{x}xy$
Rule 4: $xy \rightarrow xy\bar{y}\}$
Rule 5: $-- \rightarrow +-+$ iff $\alpha_1 + \alpha_n \in (\pi, 2\pi)$
Rule 6: $++ \rightarrow -+-$ iff $\alpha_1 + \alpha_n \in (2\pi, 3\pi)$ where the bar denotes reversal of sign.

Proof

First we show rules 1 to 6 are the only applicable rules.
Note that $\alpha'_1 + \alpha'_n + \alpha'_{n+1} - \alpha_1 - \alpha_n = \pi$.
Using this, it follows that two possibilities are eliminated: $++ \rightarrow ---$ and $-- \rightarrow +++$, because $\pi \notin (-4\pi, \pi) = (0, \pi) + (0, \pi) + (0, \pi) + (-2\pi, -\pi) + (-2\pi, -\pi)$ and $\pi \notin (\pi, 6\pi) = (\pi, 2\pi) + (\pi, 2\pi) + (\pi, 2\pi) + (-\pi, 0) + (-\pi, 0)$.
Aside from rules 1 to 6 this only leaves $-+ \rightarrow ---, -+ \rightarrow +++, +- \rightarrow ---$, and
$+- \rightarrow +++, ++ \rightarrow --+, ++ \rightarrow +--, -- \rightarrow -++, -- \rightarrow ++-$.
In each of these cases, the cell chosen is such that a new edge would intersect another edge, violating the simpleness of the extended polygon.
Thus rules 1 to 6 are the only applicable rules.

It remains to verify the conditions for application of rules 5 and 6.
Now observe that if P_1P_2 and $P_{n-1}P_n$ are not parallel, $L(P_1P_2), L(P_1P_n), L(P_{n-1}P_n)$ form a triangle.
Since P_1P_n is an edge of that triangle, and both α_1, α_n lie on the same side of P_1P_n, either $\alpha_1, \alpha_n, \pi - \alpha_1 - \alpha_n$, or $\pi - \alpha_1, \pi - \alpha_n, \alpha_1 + \alpha_n - \pi$, or $\alpha_1 - \pi, \alpha_n - \pi, 3\pi - \alpha_1 - \alpha_n$ or $2\pi - \alpha_1, 2\pi - \alpha_n, \alpha_1 + \alpha_n - 3\pi$ are the interior angles of that triangle.
The 5th rule only applies when a $\pi - \alpha_1, \pi - \alpha_n, \alpha_1 + \alpha_n - \pi$ triangle is formed. There are three cases:
(i) segments parallel iff $\alpha_1 + \alpha_n = \pi$
(ii) segments generate $\alpha_1, \alpha_n, \pi - \alpha_1 - \alpha_n$ triangle.
In this case we have to cross a segment to enter R5, so 5th and 6th expressions do not apply. This happens iff $\alpha_1 + \alpha_n \in (0, \pi)$.
(iii) segments generate $\pi - \alpha_1, \pi - \alpha_n, \alpha_1 + \alpha_n - \pi$ triangle. Happens iff $\alpha_1 + \alpha_n \in (\pi, 2\pi)$.
The 6th rule only applies when $\alpha_1 - \pi, \alpha_n - \pi, 3\pi - \alpha_1 - \alpha_n$ triangle formed.
There are three cases:
(i) segments parallel iff $\alpha_1 + \alpha_n = 3\pi$

(ii) segments generate $2\pi - \alpha_1, 2\pi - \alpha_n, \alpha_1 + \alpha_n - 3\pi$ triangle iff $\alpha_1 + \alpha_n \in (3\pi, 4\pi)$
(iii) segments generate $\alpha_1 - \pi, \alpha_n - \pi, 3\pi - \alpha_1 - \alpha_n$ triangle iff $\alpha_1 + \alpha_n \in (2\pi, 3\pi)$ ∎

The program insrule.c effects rules 1-2 of the above lemma on a binary string descriptor, and the program rule5.c effects rule 5 or 6, see appendix B.

Lemma 7.9
(i) The set of all binary string descriptors is the set of strings with three or more minus signs. Every binary string descriptor can be realised by some simple closed polygon.
(ii) The rule $--\ \rightarrow\ +-+$ applies if and only if the lhs has more than three minus signs.
The rule $++\ \rightarrow\ -+-$ applies without restriction.

Proof

(i) Every binary string descriptor obtained from application of the insertion rules on - - - can be realised. This is because there is a cell adjacent to any edge on either side in the arrangement induced by the polygon.
The only way to produce a binary string descriptor with less than three minus signs would be to apply $--\ \rightarrow\ +-+$ at least once. We show this would violate the preconditions for the 5th and 6th rules deduced above.
Without loss of generality, let $\alpha_2 \in (0, \pi)$ and $\alpha_3, \ldots, \alpha_{n-1} \in (\pi, 2\pi)$.
$\alpha_3 + \ldots + \alpha_{n-1} > (n-3)\pi$ since $\alpha_3, \ldots, \alpha_{n-1} \in (\pi, 2\pi)$.
$\sum_{i=1}^{n} \alpha_i = (n-2)\pi$ which is the Euclidean angle sum formula.
$\alpha_3 + \ldots + \alpha_{n-1} = (n-2)\pi - (\alpha_1 + \alpha_2 + \alpha_n) > (n-3)\pi$
$\alpha_1 + \alpha_n + \alpha_2 < (n-2)\pi - (n-3)\pi = \pi$
$\alpha_1 + \alpha_n \leq \pi$ ie 5th and 6th rules do not apply.
Thus the set of all binary string descriptors is the set of strings with three or more minus signs. Every binary string descriptor can be realised by some simple closed polygon.

(ii) In the first case, a string produced by the rule is a binary string descriptor if and only if it has more than two minus signs, by (i). This is possible if and only if the left hand string has more than three minus signs. In the second case, a string produced by the rule always is a binary string descriptor, by (i). ∎

The program validb.c (see appendix B) checks whether a string belongs to the language. The program bcdsign.c (see appendix B) prints out all numbers representing normalised binary string descriptors of length n.

Theorem 7.10
Let T be a finite set of binary string descriptors. Let G be the grammar with start symbol S, variable x, terminals T, strict productions $x+ \rightarrow +x$, $x- \rightarrow -x$, and productions $S \rightarrow ---$, $x \rightarrow x+$, $x \rightarrow x-$, $x++ \rightarrow x-+-$ and $x-- \rightarrow x+-+$ iff x has at least two minus symbols.
The language of G is the set of all strings over the alphabet $\{-, +\}$ that contain at least three minus symbols.
Extensions and equivalences correspond to the productions of G, except $S \rightarrow ---$.
Let n, m be positive integers. A simple closed non-degenerate n-gon with binary string

descriptor x can be extended to a simple closed non-degenerate $(n+m)$-gon with binary string descriptor y if and only if y is derivable from x.
Let G_n be the graph whose nodes are identified with all normalised binary string descriptors of length less than or equal to n, and whose edges correspond to the pairs of nodes mapped to each other by the productions $x \rightarrow x+, x \rightarrow x-, x--\rightarrow x+-+$ iff x contains at least two $-$ symbols, and $x++\rightarrow x-+-$.
Then G_n has the root $---$. Each edge joins nodes corresponding to strings whose lengths differ by exactly one. Hence all n-gon descriptors lie on the same level. Derivations correspond to strictly increasing paths.

Proof Follows from preceding pair of lemmas. ∎

7.4 Derivations and brute force complexity

When part of the boundary of a non-transparent surface patch is fully visible in one view, and partly occluded in another view, the string of the first image polygon will change into the string of the second image polygon according to the production rules of our grammar. Such a sequence of productions is called a derivation. Derivations are equivalent to paths on a graph G_n. The nodes of G_n represent all strings in the language of length less than or equal to n. Figure 7.1 is a graphical depiction of G_6. We want to know whether or not there is a derivation between two given strings. Our motivation is that if the strings of two image polygons have no derivation between them, they cannot be images of the same non-transparent boundary patch with partial occlusion. The obvious brute force approach to this problem is to generate a graph G_n for sufficiently large n, and search for a path joining two given nodes. In this section we will study the size of the graph G_n, and discover that it is exponential in n. We will then prove a lemma about derivations in our grammar that is crucial to determining a low polynomial complexity algorithm in chapter 8.

We will derive a formula for the number of distinct (normalised) binary string descriptors of length n.

Lemma 7.11
The following are impossible binary coded descriptors of length n, as are their rotations.

$2^n-1, 2^{n-1}-1, 2^{n-2}-1$

$2^{n-2}+2^{n-3}-1, 2^{n-2}+2^{n-3}+2^{n-4}-1, \ldots, 2^{n-2}+\ldots+2-1.$

There are $\lfloor n/2 \rfloor + 2$ distinct impossible binary coded descriptors. All other numbers in $[0, 2^n)$ are binary coded descriptors (ie are not impossible, and can be realised by some polygon).

Proof

The first results from no minus signs, the second from one minus sign, the third from two consecutive minus signs.

The first on the second line results from two minus signs separated by one plus, the second from two minus signs separated by two plus signs, and so on.
From lemma 7.9 (i), these are impossible binary coded descriptors because they have less than three $-$ symbols. It remains to determine how many distinct impossible descriptors there are amongst these, and to show that there are no other impossible descriptors.

The first, second and third are distinct from each other. The first and second are distinct from all others because they have different numbers of minus signs. A pair of the third onwards will be distinct if and only if the separations of the minus signs are distinct. However there are two separations: from the left and from the right and wrapping around. For example, (- + - + +) is the same as (- + + - +). There are $n-1$ possible positions for a $-$ symbol if the first is fixed in position. If n is odd, $(n-1)/2$ is an integer, so this is the number of distinct impossible descriptors from the third onwards. If n is even, one of these must be unique, because otherwise there would be three possible separations of two $-$ symbols, an absurdity. Hence there are at most $\lceil (n-1)/2 \rceil + 2$ impossible binary coded descriptors.

By induction on n we show that descriptors with 0, 1 or 2 minus signs are the only impossible ones.
The statement is true for $n = 3$.
Suppose for $n = k$ every descriptor with 3 or more minus signs is realised. An arbitrary descriptor with 3 or more minus signs of length $k+1$ can be produced by adding one plus or one minus sign. ∎

Theorem 7.12
The number of equivalence classes of simple closed nondegenerate n-gons is

$$\frac{1}{n} \sum_{m \in D_n} m\chi_n(m) - (\lfloor \frac{n}{2} \rfloor + 2)$$

where
D_n = the set of divisors of n
$\chi_n(m) = 2^{n/m} - (\chi_n(m_1) + \ldots + \chi_n(m_k))$ where $m_1, \ldots, m_k$ are the multiples of m belonging to $D_n \backslash \{m\}$.

Proof

We show $\chi_n(m)$ is the number of (non-distinct) binary coded descriptors that have:
(i) m symmetries manifesting themselves numerically
(ii) do not have more than m symmetries manifesting themselves numerically
(iii) include numbers in $[0, 2^n)$ that cannot be the descriptor of an n-gon.
A number satisfying (i) consists of m identical blocks, so each block has n/m signs (or bits). There are $2^{n/m}$ ways to choose numbers satisfying (i) but they are not necessarily distinct.
Numbers with more than m-fold symmetry that manifests itself numerically must have some multiple of m symmetries, and this multiple is also a divisor of n. Thus if $m_1, \ldots, m_k$ are the multiples of m belonging to $D_n \backslash \{m\}$ then the number of (non-distinct) binary coded descriptors satisfying (i), (ii) and (iii) is $2^{n/m} - (\chi_n(m_1) + \ldots + \chi_n(m_k))$, as expected.

A number with m-fold numerical symmetry is composed of m identical blocks of length n/m each. Rotation of each block is equivalent to rotation of the whole. And for numbers with exactly m-fold symmetry, there are exactly n/m distinct rotations. Hence the number of distinct numbers that could be descriptors (ie including some which cannot represent n-gons), is

$$\sum_{m \in D_n} \frac{\chi_n(m)}{n/m} = \frac{1}{n} \sum_{m \in D_n} m\chi_n(m)$$

By lemma 7.11 there are precisely $\lfloor n/2 \rfloor + 2$ numbers in $[0, 2^n)$ that do not represent n-gons. ∎

The program ngons.c (see appendix B) calculates the number of equivalence classes of simple closed non-degenerate n-gons. Computer calculations of the number of equivalence classes for n=3 to 200 show that after the first few terms, the ratio between the number of (n+1)-gons and n-gons upto equivalence is between 1.9 and 2, and gets closer to 2 all the time. Thus the number of equivalence classes is proportional approximately to 2^n. This means that if we attempt to generate a reasonably sized graph of binary string descriptors in its entirety, we will have a combinatorial explosion. The most obvious way to decide whether one binary string descriptor is derivable from the other (by the production of the grammar) is to search for a path in a graph. However, since the graph has exponential complexity, this straightforward approach is infeasible. The next lemma leads to a better algorithm in chapter 8; which is based on dynamic programming instead of graph searching.

Lemma 7.13
(i) If a derivation contains the rule 00 → 101 then it is either equivalent to applications of rules 1-4 or 101 remains invariant.
(ii) If a derivation contains the rule 11 → 010 then it is either equivalent to applications of rules 1-4 or 010 remains invariant.

Proof

(i) If the rule 00 → 101 was the last in the derivation then (i) follows. Otherwise consider the possible rule applications on 101.
Before 101 a 0 can be inserted producing 0101 which is equivalent to two insertions of 1 symbols. Before 101 a 1 can be inserted producing 1101 which leaves 101. The only further possibility is to apply 11 → 010 giving 01001 which is derivable from 00 by insertions.
Between the first and second symbols of 101 we can only insert 0 or 1, giving 1001 and 1101 respectively. The former is derivable from 00 by insertions, the latter has been encountered already.
Between the second and third symbols of 010 we can only insert 0 or 1, giving 1001 or 1011 respectively. The former is derivable from 00 by insertions, the latter contains 101. The only further possibility is to apply 11 → 010 giving 10010 which is derivable from 00 by insertions.
After the third symbol of 101 we can insert 0 producing 1010 which contains 101. After the third symbol of 101 we can insert 1 producing 1011, the only further possibility is to apply 11 → 010 giving 10010 which is derivable by insertions.

(ii) Similarly to (i). ∎

This lemma suggests the following algorithm. If the longer string is not derivable by insertions, we can replace any occurrence of 010 or 101 in the longer string with 11 or 00 respectively. If all such strings are not derivable by insertions from the shorter string, the two strings are not derivable from each other. However, this is still exponential in the number of substrings 00 and 11. A faster algorithm using the lemma will be developed in chapter 8.

Chapter 8

Computing corresponding points

In chapter 3 we developed an algorithm that produces a set of cyclic lists of feature points from an image. It was proved that these feature points are invariant to perspective. In this chapter, we develop an algorithm that produces a table of hypotheses of correspondence between the feature points produced from two images. A point in one image and a point in a distinct image are said to be corresponding points if they are images of the same point in the scene. Each entry in the table is a sequence of hypothesised corresponding points.

The problem of how to reject false hypotheses of correspondence is one of the most intensively studied problems in computer vision. In the special case when cameras are fully calibrated, the theory of epipolar geometry presents a successful solution to this problem. However, the general case of completely unknown perspective camera parameters, viewpoints and scenes is far more difficult. The computer vision literature does not give an adequate explanation as to why this problem is so difficult, tending to lay the blame on inaccurate feature points and numerical instability. We expect an algorithm that rejects hypotheses to have three key properties: it must not treat too many images of distinct scenes as if they were images of the same scene; it must not be biased against any hypothesis; a criterion for the stability of reconstruction must be satisfied. We will formulate these properties in mathematical terms, and prove that no algorithm with these properties exists in the epipolar geometry model. We will prove that an algorithm with these properties does exist in the new model introduced in chapters 4 and 5. We will also specify a well-behaved algorithm in the affine camera model.

8.1 Generating a table of hypotheses

8.1.1 Longest common subsequences

A string x is a subsequence of a string y if the symbols of x can be found in the same order in y. A common subsequence of strings x and y is a string that is a subsequence of both x and y. A longest common subsequence is a string whose length is maximum over all possible common subsequences.

Example 8.1
The strings *abdec* and *bdadce* have common subsequences *bdc* and *bde*. □

Let A and B be strings of length m and n respectively. We denote the i-th symbol of A by $A[i]$, and the length of a longest common subsequence of A and B by lcs_len(A, B). Wagner and Fischer provided an algorithm of $O(mn)$ complexity to compute the length of a longest common subsequence, and proved it is correct [178]. They also gave an algorithm of linear complexity (and proof) to subsequently compute a longest common subsequence. We will use the Wagner-Fischer algorithm to compute bounds on the length of a cyclic longest common subsequence, and to reject pairs of binary string descriptors that are not related by a derivation.

The Wagner-Fischer algorithm assigns a cost to the operations of inserting a symbol, deleting a symbol or changing a symbol. Each of these are called edit operations, and a sequence of such operations is called an edit sequence. The Wagner-Fischer algorithm calculates the cost of a minimum cost edit sequence required to change string A into string B. Suppose the insert or delete cost is 1; the change cost is 0 if the symbol is changed to itself, and 2 otherwise. Suppose length$(A) \leq$ length(B), otherwise interchange the strings. There must be length(B) − length(A) insertions and length(A) − lcs_len(A, B) changes (or insert and delete pairs). Then the minimum cost is length(A) + length(B) − 2 lcs_len(A, B). Thus we can calculate lcs_len(A, B) using the Wagner-Fischer algorithm.

Let cost(A, B) denote the cost of an edit sequence of minimum cost. Let cost$(a \rightarrow b)$ denote the cost of an edit operation, where cost$(a \rightarrow$ null$)$ is the cost of an insertion, and cost(null $\rightarrow b$) is the cost of a deletion.

A trace is a collection of pairs $(i_1, j_1), \ldots, (i_p, j_p)$ such that $1 \leq i_1 < \ldots < i_p \leq$ length(A) and $1 \leq j_1 < \ldots < j_p \leq$ length(B). A trace can be visualised as a collection of non-crossing lines joining symbols of A to symbols of B. Each pair (i, j) in a trace represents a change operation $A[i] \rightarrow B[j]$. If i is not in $\{i_1, \ldots, i_p\}$ then $A[i]$ is deleted. If j is not in $\{j_1, \ldots, j_p\}$ then $B[j]$ is inserted. The cost of a trace is defined to be the sum of the costs of these edit operations. A trace represents an edit sequence, except that the order of operations is ignored.

Example 8.2
This is a trace between *abcdef* and *acdf*.

a	b	c	d	e	f
|		|	|	|	
a		c	d	f	

Here b and f are deleted from *abcdef*, and e is changed to f.□

It is apparent that common subsequences correspond to traces with all lines connecting the same symbols. Thus minimum cost traces correspond to longest common subsequences. The next lemma generalises this (the proof is due to Wagner and Fischer [178]).

Lemma 8.1
Suppose cost$(a \rightarrow a) = 0$ and cost$(a \rightarrow c) \leq$ cost$(a \rightarrow b)$ + cost$(b \rightarrow c)$ for all symbols

a, b, c including null. Then minimum cost traces correspond to minimum cost edit sequences.

Proof

To every trace is associated an edit sequence of the same cost. It suffices to prove that to every edit sequence corresponds a trace whose cost is less than or equal to that of the edit sequence. A composition $T_1 \circ T_2$ of traces T_1 and T_2 is the trace $\{(i,j) \mid (i,k) \in T_1$ and $(k,j) \in T_2$ for some $k\}$. Now $\text{cost}(T_1 \circ T_2) \leq \text{cost}(T_1) + \text{cost}(T_2)$. To each edit operation in the sequence is associated a trace of the same cost. Hence the cost of the composition of the traces is less than or equal to the cost of the edit sequence. The lemma follows from this. ∎

The Wagner-Fischer algorithm is based on the following theorem.

Theorem 8.2
If x, y are symbols and A,B are strings then

$$\begin{aligned}\text{cost}(Ax, By) = \min \{ & \text{cost}(A, B) + \text{cost}(x \to y), \\ & \text{cost}(A, By) + \text{cost}(x \to \text{null}), \\ & \text{cost}(Ax, B) + \text{cost}(\text{null} \to y)\}\end{aligned}$$

provided $\text{cost}(a \to c) \leq \text{cost}(a \to b) + \text{cost}(b \to c)$. and $\text{cost}(a \to a) = 0$.

Proof

From lemma 8.1, minimum cost edit sequences correspond to minimum cost traces. Any trace T can be broken into two traces T_1T_2 because lines never cross. If T is a minimum cost trace then so are T_1 and T_2. Suppose T is a minimum cost trace from Ax to By. Since lines in a trace never cross, either:
(i) there is a line joining x to y; or
(ii) there is a line joining x to a symbol in B; or
(iii) there is a line joining y to a symbol in A; or
(iv) there is no line touching x or y.
Since $\text{cost}(x \to y) \leq \text{cost}(x \to \text{null}) + \text{cost}(\text{null} \to y)$, a trace with property (iv) can be replaced by a trace with property (i) without changing its cost. In case (i) $\text{cost}(Ax, By) = \text{cost}(A, B) + \text{cost}(x \to y)$. In case (ii) $\text{cost}(Ax, By) = \text{cost}(Ax, B) + \text{cost}(\text{null} \to y)$. In case (iii) $\text{cost}(Ax, By) = \text{cost}(A, By) + \text{cost}(x \to \text{null})$. The lemma follows from this. ∎

The lemma immediately suggests a recursive algorithm, but observe that each such function call would depend only on the lengths of the strings. Suppose $A[1 \ldots i]$ is the substring of the first i symbols of A, and $B[1 \ldots j]$ is the substring of the first j symbols of B, and $D(i,j) = \text{cost}(A[1 \ldots i], B[1 \ldots j])$. Then $D(i,j)$ depends only on $D(i-1, j-1)$, $D(i-1, j)$ and $D(i, j-1)$, so it can be computed iteratively. This idea of converting a recursive algorithm to an iterative one using a table lookup is called dynamic programming. From

$$lcs\_len(A, B) = (\text{length}(A) + \text{length}(B) - D(\text{length}(A), \text{length}(B)))/2$$

we see that algorithm 8.1 computes the length of a longest common subsequence of (linear) strings A and B. The time complexity of algorithm 8.1 is $O(mn)$. Observe that only two rows at a time of D are ever used. By storing only two rows, and mapping

i to the remainder on division by 2, the space complexity can be reduced to linear. However, the Wagner-Fischer algorithm to compute a subsequence of length len_lcs requires the entire array D. There is an algorithm due to Hirschberg that uses linear space to compute a longest common subsequence [85]. However, the time complexity of Hirschberg's algorithm is still quadratic. Eppstein claims to have obtained an algorithm with time complexity bounded by $O(n \log s + m \log \log m)$, where s is the number of characters in A and B [50].

Note that array indices begin at zero in algorithm 8.1.

A longest common subsequence can be computed from D in linear time by algorithm 8.2.

Algorithm 8.1

```
function lcs_len(A,B)

D(0,0)=0
for (j=1; j <= length(B); j=j+1)
  D(0,j)=D(0,j-1)+1
for (i=1; i <= length(A); i=i+1){
  D(i,0)=D(i-1,0)+1
  for (j=1; j <= length(B); j=j+1){
    if (A[i-1] == B[j-1])
      D(i,j)=D(i-1,j-1)
    else
      D(i,j)=D(i-1,j-1)+2
    if (D(i-1,j)+1 < D(i,j))
      D(i,j)=D(i-1,j)+1
    if (D(i,j-1)+1 < D(i,j))
      D(i,j)=D(i,j-1)+1
  }
}
return integer division of
  length(A)+length(B)-D(length(A),length(B)) by 2
```

Algorithm 8.2

```
function lcs(A,B)

  i=length(A)
  j=length(B)
  while (i <> 0 and j <> 0){
    if D(i,j)==D(i-1,j)+1 then
```

```
      i=i-1
    else if D(i,j)=D(i,j-1)+1 then
      j=j-1
    else {
      print i,j
      i=i-1
      j=j-1
    }
  }
```

8.1.2 Cyclic longest common subsequences

The edge descriptors are cyclic strings, so we are interested in longest common subsequences of cyclic strings. Cyclic subsequences can start in different positions, and wrap around a string, unlike linear subsequences.

Example 8.3
The strings *abaac* and *bcba* have a cyclic common subsequence *bca*. This is longer than any of the longest common (linear) subsequences. However, *abc* is a (linear) common subsequence of *abaac* and *bcbabcba* (the concatenation of *bcba* with itself). □

Example 8.4
The strings *abaab* and *abcb* have cyclic longest common subsequence *abb*. A longer common subsequence of *abaab* and the concatenation of *abcb* with itself is *abab*. Thus it is not true that the longest common subsequence of A with BB is a cyclic longest common subsequence of A and B.□

A brute force algorithm for finding the length of longest common cyclic subsequences is to simply iterate algorithm 8.1 on each different start position of one of the strings. For example to iterate algorithm 8.1 on *bcba*, *cbab*, *babc* and *abcb*. This has complexity $O(m^2n)$.

Maes gave an algorithm of $O(mn \log m)$ to compute longest common cyclic subsequences [117]. Maes made the observation that the Wagner-Fischer algorithm is equivalent to computing minimum weighted paths in a weighted directed graph. The vertices $v(i,j)$ of this graph represent the substrings $A[1 \ldots i]$ and $B[1 \ldots j]$. The edges represent the deletion of $A[i+1]$, the insertion of $B[j+1]$ and changing $A[i+1]$ to $B[j+1]$. The weights of the edges are the costs of these three basic edit operations. The Wagner-Fischer algorithm finds a minimum weighted path from $v(0,0)$ to $v(m,n)$.

Observe that any cyclic common subsequence of A and B is a linear common subsequence of A and BB, where BB is the concatenation of B with itself. Now suppose B is cyclically shifted by l symbols to the right, obtaining B_l. Then the weighted directed graph corresponding to A and B_l is actually contained in the weighted directed graph corresponding to A and BB. Thus the minimum weighted path from $v(0,l)$ to $v(m,n+l)$ in the graph of A, BB is the same as the minimum weighted path from $v(0,0)$ to $v(m,n)$ in the graph of A, B_l.

Maes calls two paths non-crossing if they never cross each other, but some vertices may coincide. The following lemma is due to Maes (see lemma 4.3 in [117]).

Lemma 8.3
Suppose $P(j)$ is a minimum weighted path from $v(0,j)$ to $v(m,n+j)$, and $P(l)$ is a minimum weighted path from $v(0,l)$ to $v(m,n+l)$. If $j < k < l$ and $P(j)$ and $P(l)$ are non-crossing then there is a minimum weighted path from $v(0,k)$ to $v(m,n+k)$ that is non-crossing with both $P(j)$ and $P(l)$.

Proof

Suppose P is a minimum weighted path from $v(0,k)$ to $v(m,n+k)$ that crosses $P(j)$. Then there are at least two vertices in common between P and $P(j)$. The sub-path of $P(j)$ connecting two such vertices is a minimum weighted path. Thus the path formed by deleting the part of P that is to the left of $P(j)$ and replacing it with the sub-path of $P(j)$ is still a minimum weighted path. Similarly, a sub-path of P that crosses $P(l)$ can be replaced with a sub-path of $P(l)$, and the result is still a minimum weighted path. Repeating this as necessary, we obtain a minimum weighted path $P(k)$ from $v(0,k)$ to $v(m,n+k)$ that is non-crossing with both $P(j)$ and $P(l)$. ∎

So to start with, we have to calculate minimum weighted paths for $l = 0$ and $l = m$. Then we calculate a minimum weighted path for $l = m/2$ that is non-crossing with the previous two paths. Repeating this process, it can be shown that minimum weighted paths for all shifts can be computed in time $O(mn \log m)$ [117].

Gregor and Thomason have obtained upper and lower bounds on the costs of each shift l that can be used to eliminate the majority of shifts (see [66] and [67]). The notation $\lceil x \rceil$ denotes the smallest integer greater than x.

Lemma 8.4
If $\text{cost}(A, B_k) < \text{cost}(A, B_q)$ then $\text{cost}(A, B_k) < \text{cost}(A, B_{q \pm i})$ for $0 \leq i \leq i_q$ where

$$i_q = \lceil \frac{\text{cost}(A, B_q) - \text{cost}(A, B_k)}{2} \rceil - 1$$

Proof Substitute $\theta = 1$ and $\vartheta = 1$ into proposition 2.3 in [67]. ∎

Whenever a shift q is found to be suboptimal, the shifts in the range $[q - i_q, q + i_q]$ can be eliminated because they are also suboptimal. This does not reduce the worst case complexity [67]. It is unlikely that this is the optimal algorithm to compute the length of a cyclic longest common subsequence, because there is (claimed to be) a linear longest common subsequence algorithm of lower complexity than $O(mn)$.

Fischetti et al [56] claim that an algorithm of $O(mn)$ can find an optimal cyclic alignment of two strings. However, we found a counter-example to this claim.

Example 8.5
The strings

$$liiffeifkeibigeibekigheeefbfffdnmn$$

and

$$lheiiffeegeebigeeeifkffbfficaffhhhhhjffffkhffn$$

have a common subsequence $liiffeegeeigeeefbffn$. The algorithm of Fischetti et al produces a common subsequence of only 5 symbols when a score of 20 for matching symbols, 0 for non-matching symbols and -1 for inserted or deleted symbols is used. When the score is changed to 2 for matched symbols, 0 for non-matching symbols and -1 for deleted or inserted symbols, their algorithm produces a common subsequence of 16 symbols. There appears to be no choice of score which consistently produces longest common subsequences.□

We denote the length of a cyclic longest common subsequence of strings A and B by $len\_clcs(A, B)$. We now provide some bounds on $len\_clcs(A, B)$ that can be computed in time $O(mn)$ and space $O(n)$.

Lemma 8.5
(i) $len\_clcs(A, B) \leq len\_lcs(A, BB)$
(ii) $\frac{1}{2} len\_lcs(A, BB) \leq len_c lcs(A, B)$
(iii) $len\_clcs(A, B) \leq \frac{2}{3} len\_lcs(AA, BB)$

Proof

(i) From any cyclic common subsequence of A, B we can produce a common subsequence of A, BB. Hence $len\_clcs(A, B) \leq len\_lcs(A, BB)$.

(ii) Any trace from A to BB breaks into two traces A', B and A'', B where $A = A'A''$. Each of these determines a common subsequence of length less than or equal to $len\_clcs(A, B)$. Thus $len\_lcs(A, BB) \leq 2 len\_clcs(A, B)$.

(iii) Suppose a trace T from A to BB represents the cyclic longest common subsequence of A, B. We can build a conjugate pair of traces T_1, T_2. on AA, BB whose combined length is 3 times $len\_clcs(A, B)$. Adjoin to T on the right whatever part of T will fit into BB, resulting in T_1. Adjoin to T on the left whatever part of T will fit into BB, resulting in T_2. The parts that do not fit on the left and right together equal T in length. Thus the combined length of T_1 and T_2 is 3 times the length of T. However, the lengths of T_1 and T_2 are less than or equal to $len\_lcs(AA, BB)$, so the length of T is less than or equal to $2\, len\_lcs(AA, BB)$. Since the length of T equals $len\_clcs(A, B)$, it follows that $3\, len\_clcs(A, B) \leq 2\, len\_lcs(AA, BB)$. ∎

8.1.3 The hypothesis table

In chapter 3 we developed an algorithm that generates a set of cyclic lists of feature points from an image. The features calculated from an image were represented by a pl3 file. A typical image will result in roughly a thousand edge curves. There are therefore approximately a million possible pairings between edge curves in two different images. Each such pairing can produce a hypothesis consisting of a sequence of (hypothesised) corresponding points.

After reading and parsing a pl3 file, a set of cyclic strings can be generated. Since images are only a discrete sample of intensities of light radiated by points on a surface, we cannot expect the cyclic strings of corresponding edges to be exactly the same. It therefore seems reasonable to calculate the cyclic longest common subsequences of

pairs of such strings. Each such common subsequence is a hypothetical alignment between points on edges in two images. However, we are only interested in that part of an alignment which is common to all alignments of the same length between two strings. Otherwise, the positions of points may not be determined. Thus we compute a cyclic longest common subsequence that is invariant to optimal shifts.

The brute force cyclic longest common subsequence algorithm has a $O(nm^2)$ time complexity. The brute force algorithm was implemented by the C program findflat2.c (see appendix B). When this is run on every pairing of edges, the program takes more than an hour to execute (on one processor of an SGI Power Challenge). A faster method is the following. For each left image string, we compute an upper bound on the cyclic longest common subsequence length (see lemma 8.5 (iii)) with each right image string in turn. These are sorted into decreasing order of upper bound. Actual alignments are then computed until the actual length exceeds the next upper bound length. The computation of upper bounds is $O(mn)$. This improved algorithm, still using brute force to compute alignments, was implemented by the C program findflat3.c (see appendix B). On a typical image, findflat3 executes in five minutes (on the same machine). The use of upper bounds therefore successfully eliminates the bulk of the computation. Using Maes and Gregor-Thomason algorithms would further reduce this computation time.

It is possible to interactively select edge connected components using certain tools and a VRML browser that supports anchors. A pl1 file needs to be converted to a pl2 file, which in turn needs to be converted into a VRML and HTML file. See graddisc2, pgmdisc2, edge3, pl2tovrml in appendix B, and appendix A about VRML browsers. Once component numbers in two images are selected, a tool to generate only one entry in the hypothesis table in provided (see dcmp2 in appendix B).

Pairs of strings with the largest number of symbols in common are not necessarily corresponding pairs of edges.

However, those hypotheses that are correct produce dense and accurate alignments.

8.2 Epipolar geometry

If two images of a scene are taken, then each pair of corresponding points in these two images satisfies a constraint known as the epipolar constraint. From the perspective camera model, we can see that there is a line joining a point in the scene, the point on the retina that is its image, and the optical centre of the camera. Two such lines define a plane through the point in the scene and the two optical centres of two cameras. This plane intersects the two retinal planes in two lines, each containing the corresponding image points. We deduce that the corresponding point to each point in one image must lie on a line in the other image. Such lines are known as epipolar lines. All the epipolar lines in an image intersect at a point known as the epipole, which is the image of the optical centre of the other camera.

8.2.1 Essential matrix: calibrated cameras

Suppose the image coordinates in both images are in a world coordinate system. This means that the origin of each image is the principal point, and distances in the image are actually distances on the retina. This is equivalent to knowing the intrinsic parameters $\alpha_u, \alpha_v, u_0, v_0$ of both cameras. When an image is taken and the intrinsic parameters of the camera are known, the camera is referred to as a calibrated camera.

If a set of 8 image correspondences $x_i \leftrightarrow x_i', i = 1 \ldots 8$ is given and the intrinsic parameters of both cameras are known then Longuet-Higgins [113] showed there is a matrix called the *essential matrix* E such that $x_i'^{\top} E x_i = 0$. This matrix represents the transformation between two pencils of lines in the two image planes called the epipolar lines.

There is a well-known relation between the essential matrix and the relative orientation of two cameras. Let $\mathbf{t}$ be the vector from the optical centre of the first camera to the optical centre of the second camera, in the coordinate system of the first camera. Let R be the rotation from the coordinate system of the second camera to a coordinate system parallel to that of the first camera. We can define an antisymmetric matrix T to represent the cross product of t with any vector x

$$T = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

Then

$$E = TR$$

Each image point defines a vector in $\mathbf{R}^3$, obtained just by appending 1 as the third coordinate. Let x, x' be such vectors obtained from two corresponding image points in (calibrated) image coordinates. Since x, x' and $\mathbf{t}$ are coplanar,

$$x(t \wedge Rx') = 0$$

which is the Longuet-Higgins equation. Note that $\wedge$ denotes the cross product in $\mathbf{R}^3$.

Longuet-Higgins also gave a linear method for estimating E from 8 corresponding points (known intrinsic parameters). If x, y and x', y' denote the coordinates of corresponding image points, then we can expand out the Longuet-Higgins equation into

$$x'xE_{11} + x'yE_{21} + x'E_{31} + xy'E_{12} + yy'E_{22} + yE_{32} + xE_{13} + yE_{23} + E_{33} = 0 \quad (8.1)$$

Eight such pairs of corresponding points produces a system of linear equations whose unknowns are the entries of E. If one entry is set to one, this can in general be solved for the remaining entries of E.

However this method of computing E from corresponding points ignores a cubic constraint on the essential matrix, so that the resulting method is incomplete: Essential matrices are characterised by the property that they have one singular value equal to zero (rank 2) and the other two singular values equal. There are also iterative non-linear schemes for estimating the essential matrix such as those due to Buchannan

[123][pg 3] and Horn [88]. The essential matrix is not necessarily unique, even after we allow for an unknown scale factor. Longuet-Higgins [114] catalogues the types of cases leading to non-unique reconstructions (or essential matrices). His system of linear equations had non-unique solutions if and only if all the points in the scene lie on a quadric surface passing through the two viewpoints (called a critical surface) [113]. A special case of this quadric surface is a plane. If the scene points lie in a plane there may be one, two or even infinitely many reconstructions (or essential matrices) [113]. If the scene points do not lie in a plane every non-unique reconstruction lies on a special type of quadric surface first considered by Maybank [122], and there are at most three reconstructions [114]. Note however that the above statements on the number of reconstructions or essential matrices assume dense correspondences. Experiments by Fang and Huang showed that reconstruction is often unstable, in that small changes in the positions of corresponding image points can lead to large changes in the reconstructed scene points [52]. The closer we are to ambiguity, the more unstable is the reconstruction [123].

8.2.2 Fundamental matrix: uncalibrated cameras

In the case when two cameras are uncalibrated and image correspondences are given, there is a matrix called the *fundamental matrix* F such that

$$x_i'^{\mathsf{T}} F x_i = 0 \tag{8.2}$$

These x_i are not restricted in the way the calibrated coordinates were. The fundamental matrix also represents the transformation between two pencils of epipolar lines. If we let

$$F = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}^{-1\mathsf{T}} E \begin{bmatrix} \alpha_u' & 0 & u_0' \\ 0 & \alpha_v' & v_0' \\ 0 & 0 & 1 \end{bmatrix}^{-1}$$

then it follows that $x'^{\mathsf{T}} F x = 0$.

The set of fundamental matrices is the set of 3 by 3 real matrices of rank 2.

Faugeras [53] shows that reconstructions with different camera parameters generically differ by a projective transformation. The same result was also obtained by Hartley, Gupta and Chang [75]. Hartley [76] gives a characterisation of which pairs of camera matrices correspond to a fundamental matrix. Using this he shows any two such pairs are related by a projective transformation of space. A reconstruction up to a projective transformation was then obtained by finding two arbitrary camera matrices compatible with the fundamental matrix, and reconstructing.

The fundamental matrix can be estimated by the same linear algorithm that was used to estimate the essential matrix. Hartley has claimed that by preceding the linear algorithm with a simple normalisation (translation and scaling) of the matched points, results are obtained comparable with the best iterative algorithms [77], [78].

It has been shown by Longuet-Higgins that there are at most three ambiguous essential matrices compatible with given point correspondences [114]. The same result also

applies to fundamental matrices [124]. Luong and Faugeras found two methods for assessing the proximity of a pair of point configurations to the critical surface [116]. One involves estimating the quadratic transformation between the two configurations, and the other involves estimating two fundamental matrices.

An alternative method for projective reconstruction has been obtained by Sparr ([166], [163], [162]). This method is based on the notion of affine shape, rather than on the fundamental matrix; it is called the Reciprocal Chasles Problem. Sparr gives a characterisation of correspondence which he calls the Weak Chasles Problem, which will be used in chapter 9.

The epipolar constraint is a weaker constraint than the equation in theorem 4.3. The epipolar constraint has been generalised to three views and four views. Shashua showed that there is a trilinear relationship between three views [158]. The quadrilinear constraint on four views was obtained independently by Triggs [172], Shashua and Werman [160] and Faugeras and Mourrain [55]. A generalisation is given by Shashua and Werman [159]. That paper also suggests an algorithm for motion segmentation, based on the Hough Transform.

8.2.3 The problem of bias against viewpoint

For every hypothesis, there is an imaginary camera configuration and scene that generates a perturbed version of that hypothesis. We will show that inconsistent hypotheses must have inconsistent camera configurations.

Suppose the first camera is represented by $X \mapsto \iota(g_1 X)$ and the second camera is represented by $Y \mapsto \iota(g_2 Y)$. Suppose $X_1 = \iota(g_1 W_1)$ and $Y = \iota(g_2 W_1)$ is one hypothesis of correspondence, and $X_2 = \iota(g_1 W_2)$ and $Y = \iota(g_2 W_2)$ is the other hypothesis of correspondence. Suppose X_1 and X_2 consist of distinct image points. Then $\iota(g_2 W_1) = \iota(g_2 W_2)$ which is shown below to imply $W_1 = W_2$. This in turn implies $X_2 = \iota(g_1 W_1) = X_1$, a contradiction. Hence inconsistent hypotheses must have inconsistent camera configurations.

Lemma 8.6
Let g_2 be a camera matrix. Let W_1, W_2 be three dimensional point configurations (with the same number of points), with all fourth row entries equal to one, and no non-zero entries in the third row. Then

$$\iota(g_2 W_1) = \iota(g_2 W_2) \Rightarrow W_1 = W_2$$

Proof

Recall that if (x, y, z, t) is a column of a point configuration matrix, then $\iota(x, y, z, t) = (x/z, y/z, 1, t/z)$. Thus

$$\iota(x_1, y_1, z_1, t_1) = \iota(x_2, y_2, z_2, t_2)$$

means that $x_1/z_1 = x_2/z_2$, $y_1/z_1 = y_2/z_2$ and $t_1/z_1 = t_2/z_2$. Since $t_1 = 1$ and $t_2 = 1$, $z_1 = z_2 \neq 0$. Hence $x_1 = x_2$ and $y_1 = y_2$. Applying this to all columns, we see that

if $\iota(W) = \iota(W')$ then $W = W'$ for W and W' satisfying the conditions in the lemma. Thus $\iota(g_2 W_1) = \iota(g_2 W_2)$ implies $g_2 W_1 = g_2 W_2$. Since g_2 is invertible, this implies $W_1 = W_2$. ∎

Thus, if an algorithm to test hypotheses of correspondence is not invariant to camera parameters, it becomes biased towards some hypotheses and biased against others in an unpredictable manner. The effects of such biases can be minimised by drawing hypotheses within a small window in each image. This is frequently done in algorithms that find corresponding points in images taken by moving cameras or of moving scenes. This is not a valid approach in general however. Kanatani has shown that tests of correspondence can be biased in a statistical sense (see [96] and [97]). The reader should not confuse these two different types of bias.

We now formulate the problem of testing hypotheses of correspondence.

Definition 8.1
Let M be a set of scenes, and $\iota(M)$ a set of images of scenes. Let $\sim$ be an equivalence relation on the set of scenes M. A correspondence hypothesis test on M is a pair of non-negative functions $f : \iota(M) \times \iota(M) \to \mathbf{R}$ (defined on images) and $g : M \times M \to \mathbf{R}$ (defined on scenes), with the following properties:

(i)	$\forall W_1, W_2 \in M \; g(W_1, W_2) = 0 \Leftrightarrow W_1 \sim W_2$	(test)
(ii)	$\forall W_1, W_2 \in M \; g(W_1, W_2) = g(W_2, W_1)$	(symmetry)
(iii)	For all $W_1, W_1', W_2, W_2' \in M$ such that $W_1 \sim W_1'$ and $W_2 \sim W_2'$ $g(W_1, W_2) = g(W_1', W_2')$	(unbiased)
(iv)	g is continuous on $M/\sim$	(stability)
(v)	For all scenes $W_1, W_2 \in M$, $f(\iota(W_1), \iota(W_2)) \leq g(W_1, W_2)$ and $f(\iota(W_1), \iota(W_2)) = 0 \Rightarrow g(W_1, W_2) = 0$	(lifting)

In the case of epipolar geometry, the equivalence relation $\sim$ is $W_1 \sim W_2$ if and only if $gW_1 d = W_2$ for some $g \in GL(4)$ and $d \in \mathrm{diag}(GL(n))$. This is of course the equivalence relation of projective geometry.

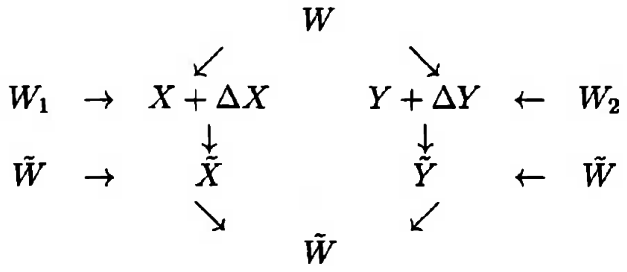
Example 8.6
We state a correspondence hypothesis test for planar scenes parallel to the retina of a camera with aspect ratio 1. Let $\mathbf{n} \in \mathbf{R}^3$ be a fixed vector. Let M be the set of k-points in a plane with normal $\mathbf{n}$, where $k \geq 3$ and not all the points are identical. If $x, y \in M$ we define $x \sim y$ if there is a rotation with axis $\mathbf{n}$ and a translation of $\mathbf{R}^3$ mapping x to y. In chapter 4 we proved that two different images of a scene in M can only differ by rotation, translation and scale (Euclidean similarity). In chapter 5 we stated a metric ρ on the quotient space of such images by Euclidean similarities. We define $g(W_1, W_2) = \rho(\iota(W_1), \iota(W_2))$, and $f(X, Y) = \rho(X, Y)$.□

Example 8.7
Suppose X, Y are perturbed images of the same scene, and we compute $\tilde{X}$ such that $\|X - \tilde{X}\|$ is minimised subject to the constraint that $\tilde{X}$ and Y correspond in an exact sense. Consider M and $\sim$ in example 8.6. Any continuous real valued function on a compact space is bounded. We know from chapter 5 that $M/\sim$ is compact. The least squares error between $X, \tilde{X}$ is not a correspondence hypothesis test, because it is an unbounded function. □

It is a general feature of all the equivalence relations and scene spaces that their correspondence hypothesis tests must be bounded. We can see from this that most of the correspondence search algorithms in the computer vision literature must be biased and/or unstable.

The lifting property and the test property imply that $f(X,Y) = 0$ if and only if X and Y are images of some pair of equivalent scenes. This ensures that the zeros of f represent corresponding image pairs in an ideal sense.

Since all valid hypotheses of correspondence will have suffered some perturbation due to digital geometry, it is necessary to have a relation between errors in reconstruction and perturbations in images. Otherwise, choosing the hypotheses with the smallest image perturbations may result in very inaccurate reconstructions. The situation can be summarised by the following diagram.



The scene is denoted by W. One ideal image of W is X, but it is perturbed to $X + \Delta X$. The other ideal image is Y, and it is perturbed to $Y + \Delta Y$. We want to apply an "inverse" perturbation to $X + \Delta X$ and $Y + \Delta Y$ obtaining images $\tilde{X}$ and $\tilde{Y}$ respectively that are exactly in correspondence. This will enable us to produce a reconstruction $\tilde{W}$. Our aim is to choose $\tilde{X}$ and $\tilde{Y}$ in such a way as to minimise the error in reconstruction $\|W - \tilde{W}\|$. It is difficult to ensure this since W is unknown. Instead, we can seek W_1, W_2 such that $\iota(W_1) = X + \Delta X$ and $\iota(W_2) = Y + \Delta Y$, and $g(W_1, W_2)$ is minimised.

From the unbiased property, g is well-defined on $M/\sim$. When we require that g is continuous on $M/\sim$ we mean that it is continuous with respect to the quotient topology on $M/\sim$. This implies that g is continuous with respect to the Euclidean metric on scenes in M.

Lemma 8.7
(i) For all $\epsilon > 0$ there is a $\delta > 0$ such that $\|W_1 - W_2\| < \delta \Rightarrow |g(W_1, W_2)| < \epsilon$.
(ii) If $\|(X_0 - X, Y_0 - Y)\| < \|(X_0 - X', Y_0 - Y')\|$ then $|g(X_0, Y_0) - g(X, Y)| < |g(X_0, Y_0) - g(X', Y')|$.

Proof

(i) Let $\epsilon > 0$ be given. Since g is continuous in the quotient topology on $M/\sim$ it is also continuous in the Euclidean topology on M. Hence there is a $\delta > 0$ such that $\|W_1 - W_2\| < \delta \Rightarrow |g(W_1, W) - g(W_2, W)| < \epsilon$. Letting $W = W_2$, since $g(W_2, W_2) = 0$ we have $\|W_1 - W_2\| < \delta \Rightarrow |g(W_1, W_2)| < \epsilon$.

(ii) Let $\epsilon = |g(X_0, Y_0) - g(X', Y')|$. From continuity of g there is a $\delta > 0$ such that

$\|(X_0 - X, Y_0 - Y)\| < \delta \Rightarrow |g(X_0, Y_0) - g(X, Y)| < \epsilon$. If $\delta > \|(X_0 - X', Y_0 - Y')\|$ then $|g(X_0, Y_0) - g(X', Y')| < \epsilon$ which is a contradiction. Hence $\|(X_0 - X, Y_0 - Y)\| < \|(X_0 - X', Y_0 - Y')\| \Rightarrow |g(X_0, Y_0) - g(X, Y)| < |g(X_0, Y_0) - g(X', Y')|$. ∎

Suppose we compute scenes W_1, W_2 such that $\iota(W_1) = X + \Delta X$ and $\iota(W_2) = Y + \Delta Y$. This lemma shows that it is necessary to minimise g to minimise error in reconstruction. From the stability property and the test property, convergence to zero of g implies convergence to zero of reconstruction error.

8.2.4 Non-existence results in epipolar model

As we pointed out before, the fundamental matrix determines an orbit of the projective group. As we saw in chapter 4, very few of the projective transformations are realised in this way. This means that the epipolar model treats many images of distinguishable scenes as if they were images of the same scene. Ostensibly, the theory of epipolar geometry was easily generalised to deal with images taken with cameras that were not calibrated. We showed in chapter 4 that there is no means of comparison of two projective equivalence classes, because the associated quotient space is not metrisable. For this reason, we weakened the definition of a correspondence hypothesis test to just ensure lack of bias and convergence properties. Unfortunately, even this weakened definition cannot be realised in the epipolar geometry model.

Theorem 8.8
Let M be the set of n point configurations in $\mathbf{R}^3 \setminus \{0\}$. Let $X \sim Y$ if there is a $g \in GL(4)$ and $d \in \text{diag}(GL(n))$ such that $Y = gXd$, where the columns of X and Y are homogeneous representations of points in M. There is no correspondence hypothesis test on M.

Proof

From lemma 4.5, $M/\sim$ is not Hausdorff. The equivalence relation $\sim$ is open since $X \mapsto gXd$ is continuous. From lemma 5.16 the set $R = \{(x, y) \mid x \sim y\}$ is not closed in $M \times M$. Thus there is a sequence in R that converges outside R. If $g : M \times M \to \mathbf{R}$ is continuous on $M \times M$ and zero on R, then it is zero on a point outside R. Hence there is no correspondence hypothesis test on M. ∎

The reader may have noticed that to construct a shape space for the perspective camera model in chapter 5, we demanded that M contain only non-planar scenes. A careful examination of the proof of lemma 4.5 shows that even with this restriction, there is no correspondence hypothesis test on M with respect to the projective equivalence relation. By contrast, we will see in the next section that a correspondence hypothesis test does exist in the affine camera model. Moreover we will state the test explicitly (as an algorithm).

That epipolar geometry is not very useful to generate corresponding points without some additional restrictions has been accepted empirically in computer vision for some time [52]. We have provided theoretical confirmation of this, which was not previously available. On the other hand, when a sparse but accurate set of corresponding points

are known, epipolar geometry is useful to determine a larger set of corresponding points. There is an extensive literature on the uniqueness of reconstruction in the epipolar model [124].

Using geometric invariant theory, it is possible to give more restrictive conditions on M under which a correspondence hypothesis test may exist on the projective equivalence relation. However, this would raise the difficult question of how to compute membership on different kinds of sets of scenes M from image data. A more general approach is possible using the reformulation of the perspective camera model. This avoids the difficulties of determining membership of different types of sets M from image data.

8.3 String hypothesis rejection algorithm

Once a hypothesis of a set of corresponding points have been made, it is necessary to reject it if it could not result from two images of the same scene. The binary string descriptor is an important invariant for this purpose: although it is not very strict, it is quite stable except for degenerate vertices; binary string descriptors must be equal before we compute stricter correspondence hypothesis tests; the stricter correspondence hypothesis tests are only comparable for tuples of the same number of points. Binary string descriptors are too weak an invariant to be used on their own, but are important as a front-line algorithm.

Lemma 8.9
If two perspective views of a simple closed non-degenerate polygon on a surface that is not transparent, where neither image polygon is degenerate are given with the correspondences, then the binary string descriptors of both images are equal.

Proof

Suppose two symbols of a binary string descriptor are not equal. Then the two views are on opposite sides of the plane spanning the two edges incident to the vertex in the scene. If the surface is not transparent, this is a contradiction. ∎

Thus if the binary string descriptors of an alignment in the hypothesis table do not coincide, the hypothesis can be rejected. Some problems remain however, since this assumes that an image vertex is not degenerate. This could occur due to an occlusion for example. Also an image of a line in the digital topology is often not a line. A simple solution to this problem is to calculate the contiguous intervals of points at which the signs in the two descriptors differ, and compute a collinearity test on these. If any of the collinearity tests are rejected, the hypothesis may be rejected. However, the endpoints of this interval may or may not be part of a collinear segment. This can be seen from the rules of the grammar introduced in chapter 7.

8.3.1 Rejecting non-derivations

When significant occlusions do occur, the binary string descriptor calculated from one image will be a derivation of the binary string descriptor calculated from another image. Also, when one edge curve occludes another, forming a junction, the junction can be bridged in different ways in different images. In such cases, several non-overlapping derivations will exist. We describe an efficient algorithm to reject hypotheses that are not related by derivations.

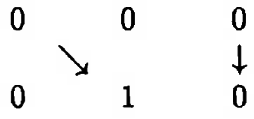
Suppose we define a cost function on strings A, B as follows:
$\text{cost}(\text{null} \rightarrow A[i]) = 0$;
$\text{cost}(A[i] \rightarrow \text{null}) = 1$;
$\text{cost}(A[i] \rightarrow A[i]) = 0$;
$\text{cost}(0 \rightarrow 1) = 0$ if $i \geq 2$, $A[i-1] = 0$, $B[i-1] = 0$ and $B[i-2] = 1$;
$\text{cost}(1 \rightarrow 0) = 0$ if $i \geq 2$, $A[i-1] = 1$, $B[i-1] = 1$ and $B[i-2] = 0$.
Note that the cost of a change is 0 if it conforms with the grammar, and 1 if it does not. It is easy to see that this cost function satisfies $\text{cost}(s \rightarrow s) = 0$ and $\text{cost}(a \rightarrow c) \leq \text{cost}(a \rightarrow b) + \text{cost}(b \rightarrow c)$. From the proof of the Wagner-Fischer algorithm (see [178]) we see that it can calculate a minimum cost edit sequence.

Lemma 8.10
If the minimum cost edit sequence from A to B is not zero, then B is not derivable from A in the grammar.

Proof

From lemma 7.13, the non-insertion rules are either equivalent to a single change or a sequence of insertions. Thus the derivations of the grammar are also derivations in the weaker grammar of zero cost edit sequences. Every step in a derivation has zero cost. Thus if B is derivable from A, there is a zero cost edit sequence from A to B. ∎

Example 8.8
Consider the trace



This zero cost trace is not a legal derivation in the grammar. Hence the condition is necessary, but not sufficient.□

In fact, if we generate all zero cost traces, we can check each one to see whether it is a derivation. If none of the traces are derivations, there is no derivation from A to B. However, the Wagner-Fischer algorithms can only generate one minimum cost trace, not all of them.

8.4 An affine hypothesis rejection algorithm

This section is based on the affine camera model, due to Koenderink and van Doorn [102]. They gave an algorithm for reconstruction, based on a choice of coordinate frame. This differs from the author's algorithm, which does not involve any special choice of coordinates. An exact criterion for correspondence in the affine camera model[1] was obtained by Asmuth, Stiller and Wan [167]. It is also possible to characterise precisely when a reconstruction is not unique or ambiguous. This will be shown to be almost the same as determining whether the three dimensional point configuration is planar; the only exception is when the two affine cameras are parallel. In this section, we will state a correspondence hypothesis test for the affine camera model. A non-uniqueness test is computed simultaneously by the same algorithm. The correspondence hypothesis test and the criterion for ambiguity are formulated in terms of an old but somewhat obscure problem of Euclidean geometry: determining the angles between two linear subspaces [57]. This leads to a simple algorithm, based only on linear algebra, for computing both a correspondence hypothesis test and a non-uniqueness test. This is the main advance of our approach over that of [102], [167]. We will then show that the correspondence hypothesis test does not remain unbiased if it is applied to images from a perspective camera. We will prove that if the affine correspondence hypothesis test is zero on a pair of images of a scene taken by two perspective cameras, then the scene points lie in certain planes. If the optical axes of two perspective cameras are parallel, the affine correspondence hypothesis test will be biased towards fronto-parallel planes. If the optical axes of two perspective cameras are not parallel but are at the same horizontal elevation, the test is biased towards upright planes. Examples of this include walls, doors and road signs.

In the camera's coordinate system the affine camera forms an image as follows:

$$\begin{bmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \\ z_1 & \cdots & z_n \end{bmatrix} \mapsto \begin{bmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \end{bmatrix}, \quad W \mapsto \iota(W) \tag{8.3}$$

A translation can be expressed by a matrix whose columns are equal, so from equation 8.3 it follows that translations in three dimensions induce plane translations. The translation normalised 3 by $n-1$ matrix is mapped to the translation normalised 2 by $n-1$ matrix by omitting its third row. Hence the rowspace of the latter is contained in the rowspace of the former.

> The plane representing the affine shape of the image is contained in the three dimensional subspace representing the affine shape of the object:

Lemma 8.11
Let W be a three dimensional point configuration and ι an affine camera. Then $\text{rowspace}(\pi_1(\iota(W)) \subset \text{rowspace}(\pi_1(W))$.

[1] Actually in a variant of the affine camera model they called a generalised weak perspective camera.

Given two affine views of the three dimensional n-point set, we can compute two such planes in $\mathbf{R}^{n-1}$, which are both contained in a three dimensional linear subspace of $\mathbf{R}^{n-1}$, representing its three dimensional affine shape. If these two planes do not coincide, their span must be the desired three dimensional subspace. On the other hand, two random planes in $\mathbf{R}^{n-1}$ will have a four dimensional span. Thus we have arrived at a criterion for correspondence:

> If the affine shapes of two image point configurations have a four dimensional span they do not correspond, otherwise they correspond.

If the span of the image shapes is three dimensional then this span is the affine reconstruction. If the span is two dimensional, the cameras may be parallel, or the points being viewed may be coplanar.

The problem with this criterion of correspondence is that any image noise can produce two image shapes that have a four dimensional span.

Example 8.9
Consider the 6 point scene configuration W

$$\begin{bmatrix} 75.42481664 & -8.193420460 & -69.67478249 & 39.63628963 & -27.85321998 & 39.81722423 \\ -30.96458980 & 8.923052791 & 74.99525990 & -101.4717114 & 51.25546696 & -56.60868763 \\ 30.57416999 & -39.11001557 & 76.12354353 & 82.89820510 & 71.06062940 & 21.02331013 \end{bmatrix}$$

And affine cameras

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -200 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 0 & -1/\sqrt{2} & 10 \\ 0 & 1 & 0 & 0 \\ 1/\sqrt{2} & 0 & 1/\sqrt{2} & -200 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The left image is X

$$\begin{bmatrix} 75.42481664 & -8.193420460 & -69.67478249 & 39.63628963 & -27.85321998 & 39.81722423 \\ -30.96458980 & 8.923052791 & 74.99525990 & -101.4717114 & 51.25546696 & -56.60868763 \end{bmatrix}$$

The right image is

$$\begin{bmatrix} 41.71419637 & 31.86133405 & -93.0949850 & -20.59079378 & -59.94265363 & 23.28930412 \\ -30.96458980 & 8.923052791 & 74.99525990 & -101.4717114 & 51.25546696 & -56.60868763 \end{bmatrix}$$

$$\begin{bmatrix} \pi_1(X) \\ \pi_1(Y) \end{bmatrix} = \begin{bmatrix} -59.12702247 & -84.33632424 & 35.03137862 & -33.22926329 & 34.64280146 \\ 28.20482256 & 70.23179311 & -103.1635030 & 56.69323538 & -52.17621821 \\ -6.967025759 & -106.0488215 & -12.19736931 & -44.64541513 & 39.52720581 \\ 28.20482256 & 70.23179311 & -103.1635030 & 56.69323538 & -52.17621821 \end{bmatrix}$$

The rank of this is 3. Adding noise to left and right images we obtain noisy left image X'

$$\begin{bmatrix} 75.78534027 & -9.178351692 & -71.19381016 & 40.24226736 & -26.29505214 & 39.6669825 \\ -30.62115938 & 9.122345336 & 74.63198907 & -101.3001907 & 51.20177150 & -57.125678 \end{bmatrix}$$

and noisy right image Y'

$$\begin{bmatrix} 40.42436869 & 32.63456327 & -93.05024126 & -20.87649285 & -59.03850845 & 21.93287174 \\ -31.16120277 & 8.260902585 & 76.33433950 & -100.3120010 & 51.39455104 & -56.6714150(\end{bmatrix}$$

and repeating the calculation

$$\begin{bmatrix} \pi_1(X') \\ \pi_1(Y') \end{bmatrix} = \begin{bmatrix} -60.07840274 & -85.32169182 & 36.17492718 & -31.49180970 & 34.50186900 \\ 28.10290169 & 69.71361798 & -103.0667650 & 56.56672881 & -52.70243885 \\ -5.508224235 & -105.8013879 & -12.30857925 & -43.66732892 & 38.26219432 \\ 27.87563802 & 71.67573562 & -102.2978197 & 56.45091484 & -52.55829998 \end{bmatrix}$$

The rank of this is 4.□

To overcome this problem, we define a correspondence hypothesis test. In this case, M will have to be the set of non-coplanar scenes of k points where $k \geq 4$. We define $W_1 \sim W_2$ if rowspace$(\pi_1(W_1))$ = rowspace$(\pi_1(W_2))$. In chapter 5 we saw that $M/\sim$ is the Grassman manifold $G(3, k-1)$, and the largest angle between rowspace$(\pi_1(W_1))$ and rowspace$(\pi_1(W_2))$ is a quotient metric. If we define g to equal this metric, then most of the properties of a correspondence hypothesis test will hold. This leaves us to find an f satisfying the remaining properties. Observe that the smallest angle between rowspace$(\pi_1(\iota(W_1)))$ and rowspace$(\pi_1(\iota(W_2)))$ is zero precisely when these two planes in $\mathbf{R}^{k-1}$ intersect non-trivially. Observe also that rowspace$(\pi_1(\iota(W_1)))$ and rowspace$(\pi_1(\iota(W_2)))$ are subspaces of rowspace$(\pi_1(W_1))$ and rowspace$(\pi_1(W_2))$ respectively. This means that the smallest angle between these two image affine shapes is less than or equal to the largest angle between the two scene affine shapes. These two observations imply that if we define f to equal the smallest angle between rowspace$(\pi_1(\iota(W_1)))$ and rowspace$(\pi_1(\iota(W_2)))$, and g as before, this is a correspondence hypothesis test.

The algorithm 5.1 can therefore be used to compute the correspondence hypothesis test. As a useful by-product, this algorithm also computes a test for the uniqueness of the affine reconstruction. If the largest angle between two image shapes is zero, those shapes are identical, hence no three dimensional reconstruction is possible from these two image configurations. Thus the largest angle is a measure of ambiguity. For the sake of brevity we will call the value of f affine disparity. Affine disparities will be quoted in radian units.

The least squares distances between points and linear subspaces do not furnish a correspondence hypothesis test because the Grassman manifolds are compact. This is why we must use the ostensibly more complicated notion of angles.

The reader should note that the definition of disparity we have just arrived at is bounded because it is an angle. However, the upper bound may in fact be smaller than π. Experiments in psychophysics have led to conjectures that for the definition of disparity in the calibrated case (see [54]) the disparity gradient is upper bounded. Contrast this with our definition of disparity, which is automatically bounded.

We discuss an experiment to study how the affine correspondence hypothesis test behaves on images taken by perspective cameras. We will examine the accuracy of reconstruction and the accuracy of the affine disparity. The accuracy of reconstruction is measured by comparing the reconstructed three dimensional shape with the true three dimensional shape, using an affine shape metric. The accuracy of affine disparity is measured by its deviation from zero, relative to its maximum value. We generate a random three dimensional point configuration. The experiment involves simulating two perspective cameras, and computing the images they form of the (fixed) three dimensional point configuration. The camera parameters of the second camera are varied, and graphs are displayed to show how the reconstruction accuracy and disparity vary as the camera parameters vary. In figure 8.1 the angle of the second perspective camera is varied; an affine reconstruction is calculated; this is compared to the correct affine shape using the plucker affine metric. The maximum value of the metric was less than 0.1 radians which is a small angle. In figure 8.2 the angle of the second perspective camera is varied; the affine disparity between the two images is calculated. The maximum disparity is more than 1.4 radians which is a very large angle, whereas the images are in perspective correspondence. In figure 8.3 the distance between the two perspective cameras is varied; an affine reconstruction is calculated; this is compared to the correct affine shape using the plucker affine metric. The maximum value of the metric was less than 0.09 radians which is a small angle. In figure 8.4 the distance between the two perspective cameras is varied; the affine disparity between the two images is calculated. The maximum disparity is more than 1.4 radians which is a very large angle, whereas the images are in perspective correspondence. In figure 8.5 the distance between the object and the second perspective camera is varied; an affine reconstruction is calculated; this is compared to the correct affine shape using the plucker affine metric. When the second camera is extremely close to the object the reconstruction error rises dramatically; as it recedes slightly the reconstruction error falls sharply and becomes very small. In figure 8.6 the distance between the object and the second perspective camera is varied; the affine disparity between the two images is calculated. The disparity is about 0.4 radians unless the second camera is extremely close to the object, which is significant, whereas the images are in perspective correspondence. We can see that the affine correspondence hypothesis test does not remain unbiased when given images from perspective cameras. On the other hand the affine reconstruction is an accurate approximation in simulations.

Consider two images of four points. The affine shape transforms of these two image configurations are planes in three dimensional space. The smallest angle between two planes in three dimensions is automatically zero, since they always intersect. Thus no information on correspondence can be extracted from a pair of images of four points; every pair of four point configurations could correspond. On the other hand the largest angle between two planes in three dimensions is not necessarily zero. Thus we can

decide whether four points are coplanar or not, but we cannot decide whether they correspond or not.

Real cameras are modelled by the perspective camera model. In view of the fact that the affine disparity was severely biased by a perspective camera, it seems very unlikely that any useful information on coplanarity could be obtained from applying the affine camera model to real images. However, we will prove that if the affine disparity is zero for two perspective cameras, the scene points must lie in certain planes.

Lemma 8.12
Suppose the affine correspondence hypothesis test is satisfied by two perspective cameras. Let x_i, y_i, z_i be the coordinates of $n \geq 4$ points in the scene, none of which are in the focal planes of either camera. Let R be the 3×3 matrix representing the rotation that makes the optical axis of the right camera parallel to the optical axis of the left camera. Let T be the baseline vector. Then for some $\lambda \neq 0$ and all $i = 1 \ldots n$

$$\lambda R_{31} x_i + \lambda R_{32} y_i + (\lambda R_{33} - 1) z_i + T_3 = 0$$

Proof

Let

$$g_1 = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$g_2 = \begin{bmatrix} \alpha'_u & 0 & u'_0 & 0 \\ 0 & \alpha'_v & v'_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let W be the $4 \times n$ matrix whose columns are $(x_i, y_i, z_i, 1)$. From theorem 4.3 $\iota(g_1 W) = g_1 \iota(W) d_1$, where the entries of the diagonal matrix d_1 in the i-th row and i-th column are $z_i / ((g_1)_{31} x_i + (g_1)_{32} y_i + (g_1)_{33} z_i + (g_1)_{34})$. Likewise $\iota(g_2 W) = g_2 \iota(W) d_2$, where the entries of the diagonal matrix d_2 are $z_i / ((g_2)_{31} x_i + (g_2)_{32} y_i + (g_2)_{33} z_i + (g_2)_{34})$. In chapter 9 we will prove that

$$\iota(g_1 W) = (g_1 g_2^{-1}) \iota(g_2 W) d_2^{-1} d_1$$

If the affine correspondence hypothesis test is satisfied, there must be an affine transformation from $\iota(g_1 W)$ to $\iota(g_2 W)$, so $d_2^{-1} d_1 = \lambda I$ for some $\lambda \neq 0$. Substituting $(g_1)_{31} = 0, (g_1)_{32} = 0, (g_1)_{33} = 1, (g_1)_{34} = 0$, and $(g_2)_{31} = R_{31}, (g_2)_{32} = R_{32}, (g_2)_{33} = R_{33}, (g_2)_{34} = T_3$ into the expressions for the entries of d_1, d_2, we have

$$z_i = \lambda (R_{31} x_i + R_{32} y_i + R_{33} z_i + T_3)$$

It remains to show that there is an affine transformation from the affine shape of one image to the affine shape of the other image. Expanding the equation $\iota(g_1 W_1) = \lambda g \iota(g_2 W_2)$ into

$$[x_i \quad y_i \quad 1 \quad 1/z_i]^\top = \lambda g [x'_i \quad y'_i \quad 1 \quad 1/z'_i]^\top$$

we obtain

$$g_{34}x_i = \lambda(g_{11}g_{34} - g_{14}g_{31})x'_i + \lambda(g_{12}g_{34} - g_{14}g_{32})y'_i + \lambda(g_{13}g_{34} - g_{14}g_{33}) + g_{14}$$

$$g_{34}y_i = \lambda(g_{21}g_{34} - g_{24}g_{31})x'_i + \lambda(g_{22}g_{34} - g_{24}g_{32})y'_i + \lambda(g_{23}g_{34} - g_{24}g_{33}) + g_{24}$$

Thus there is an affine transformation from one image to the other. The lemma follows from this. ∎

If two perspective cameras are parallel, we can expect an algorithm that minimises affine disparity to be biased towards scene points in the plane $(\lambda - 1)z_i + T_3 = 0$. This is a plane parallel to both cameras retinas. If two perspective cameras have optical axes in the $y = 0$ plane, then $R_{32} = 0$, so the normal to the plane is $(\lambda R_{31}, 0, \lambda R_{33} - 1)$. All normal directions in the $y = 0$ plane, except the ones at right angles to the optical axis of the first camera can be achieved. However, for each direction there is a fixed plane. Thus for a pair of cameras that can only rotate on a horizontal level, and cannot tilt up or down, an algorithm that minimises affine disparity can be expected to be biased towards upright planes. Examples of upright planes will include surfaces of doors, walls and road signs. The ground or floor is not an upright plane.

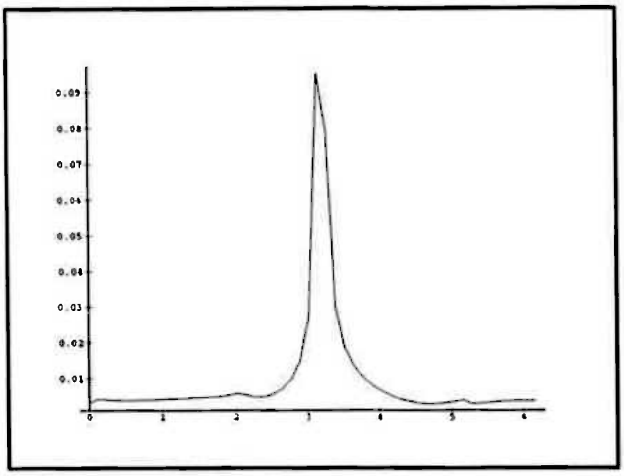
We performed a number of experiments to study this bias towards certain planes in the scene.

The program stereo8.c (see appendix B) was used to generate a random configuration of 4 coplanar points in the scene, simulate two perspective cameras with varying parameters, and generate graphs to show how the affine coplanarity statistic varies. The program stereo9.c (see appendix B) was used to generate a random configuration of 4 points which were *not* coplanar in the scene, simulate two perspective cameras with varying parameters, and generate graphs to show how the affine coplanarity statistic varies. The graphs were displayed by Maple.

Figures 8.7 and 8.8 are graphs of the affine coplanarity statistic as a function of the angle of the second perspective camera; in figure 8.7 the scene points were coplanar, whereas in figure 8.8 the scene points were not coplanar. The coplanarity statistic is under 0.01 radians when the points are coplanar. If the angle between the cameras is greater than about 0.02 radians, the coplanarity statistic on the unflat configuration is greater than about 0.04 radians, and can discriminate between the flat configuration and the other configuration. The further the cameras are from being parallel, the greater the coplanarity statistic. When the cameras are parallel everything seems flat.

Figures 8.9 and 8.10 are graphs of the affine coplanarity statistic as a function of the distance between the two perspective cameras; in figure 8.9 the scene points were coplanar, whereas in figure 8.10 the scene points were not coplanar. The coplanarity statistic is less than 0.002 radians when the points are coplanar. Unless the distance between the cameras is about 200, the coplanarity statistic of the unflat configuration is greater than 0.002 radians, and can discriminate between the flat and unflat configurations. The point configuration is about 2 units wide.

Figures 8.11 and 8.12 are graphs of the affine coplanarity statistic as a function of the distance between the object and the second perspective camera; in figure 8.11 the scene points were coplanar, whereas in figure 8.12 the scene points were not coplanar. The



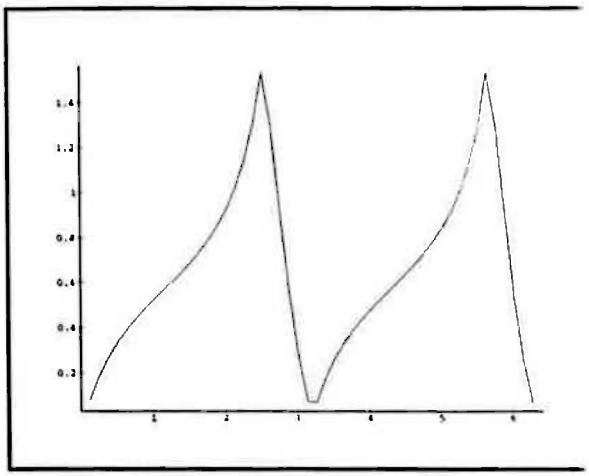
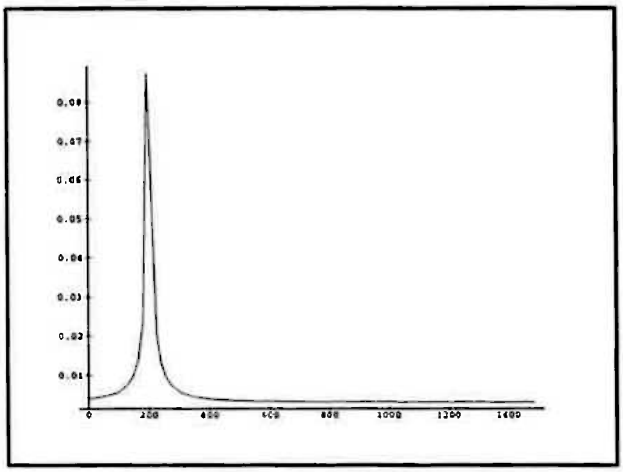


Figure 8.1: angle of 2nd perspective camera vs affine reconstruction accuracy

Figure 8.2: angle of 2nd perspective camera vs affine correspondence function



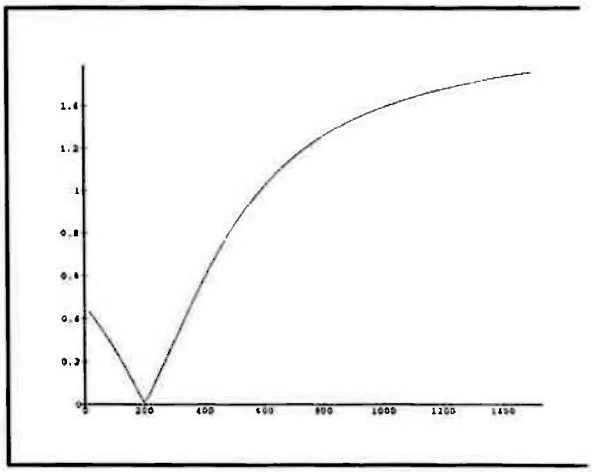
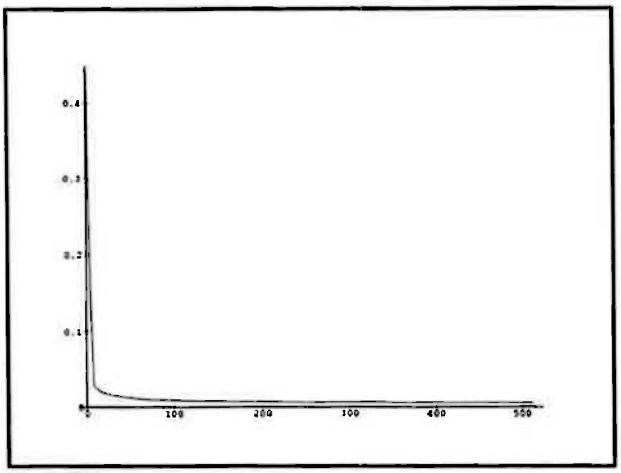


Figure 8.3: perspective camera separation vs affine reconstruction accuracy

Figure 8.4: perspective camera separation vs affine correspondence function

coplanarity statistic of the flat points is always smaller than the coplanarity statistic of the unflat points, regardless of how far away the second camera recedes from the object.

The same programs can be used to generate such graphs for many different scene configurations, with four or more points, obtaining similar results. Surprisingly, we find that if four or more points in the scene are coplanar, the affine coplanarity statistic is very small, except when the perspective cameras are almost parallel, or the distance between the cameras is a certain value. If four or more points in the scene are not coplanar, the affine coplanarity statistic is significantly different from zero.



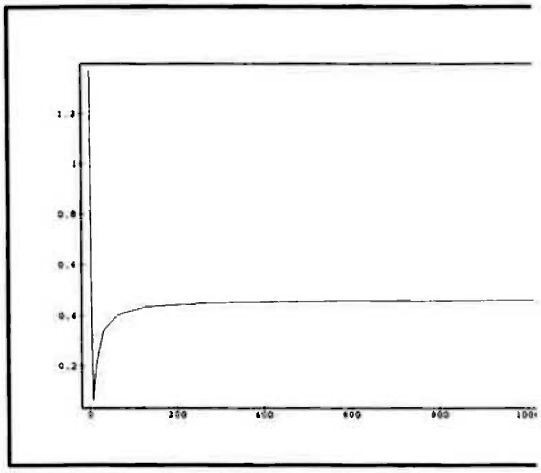


Figure 8.5: object to 2nd perspective camera distance vs affine reconstruction accuracy

Figure 8.6: object to 2nd perspective camera distance vs affine correspondence function

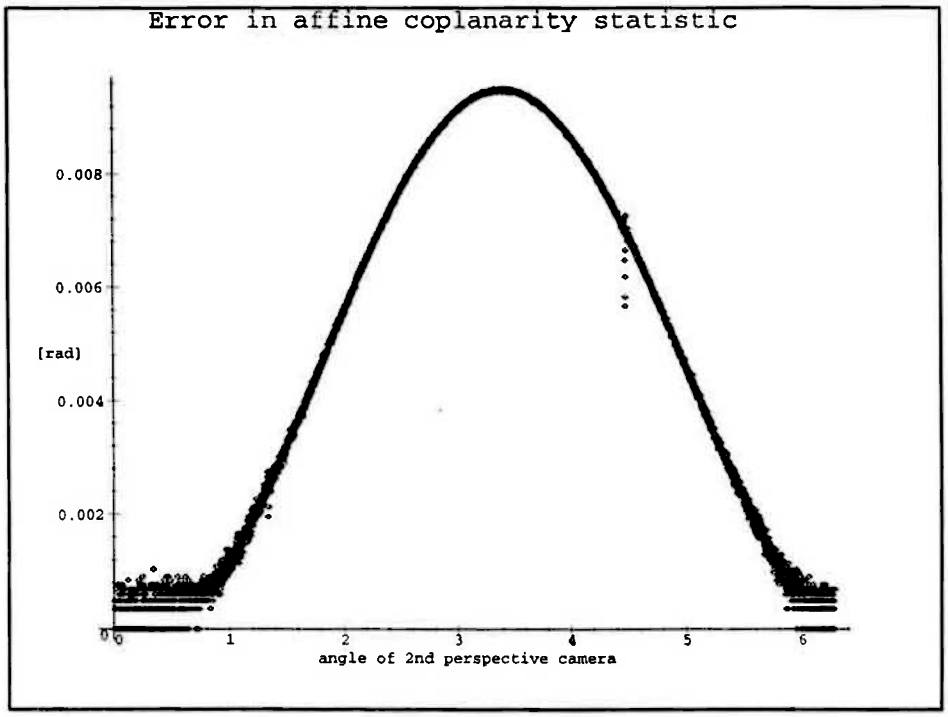


Figure 8.7

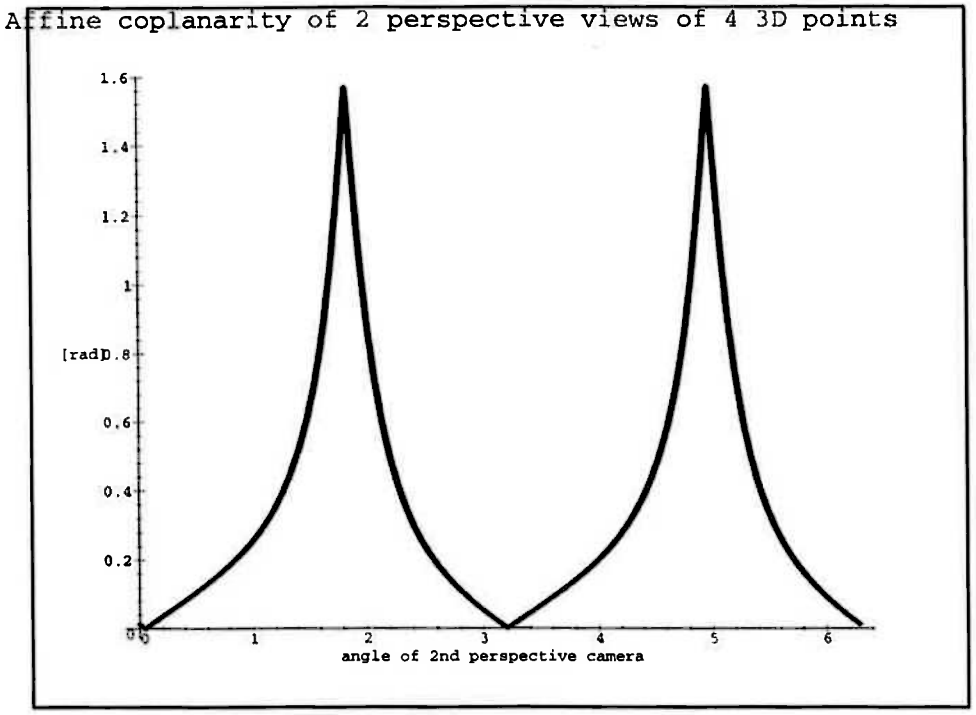


Figure 8.8

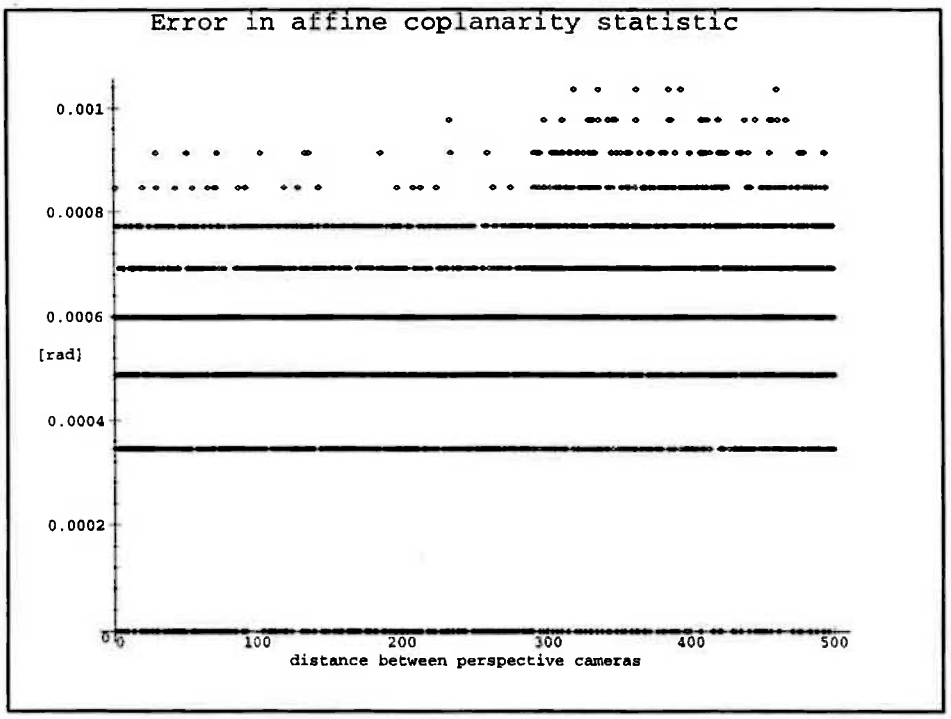


Figure 8.9

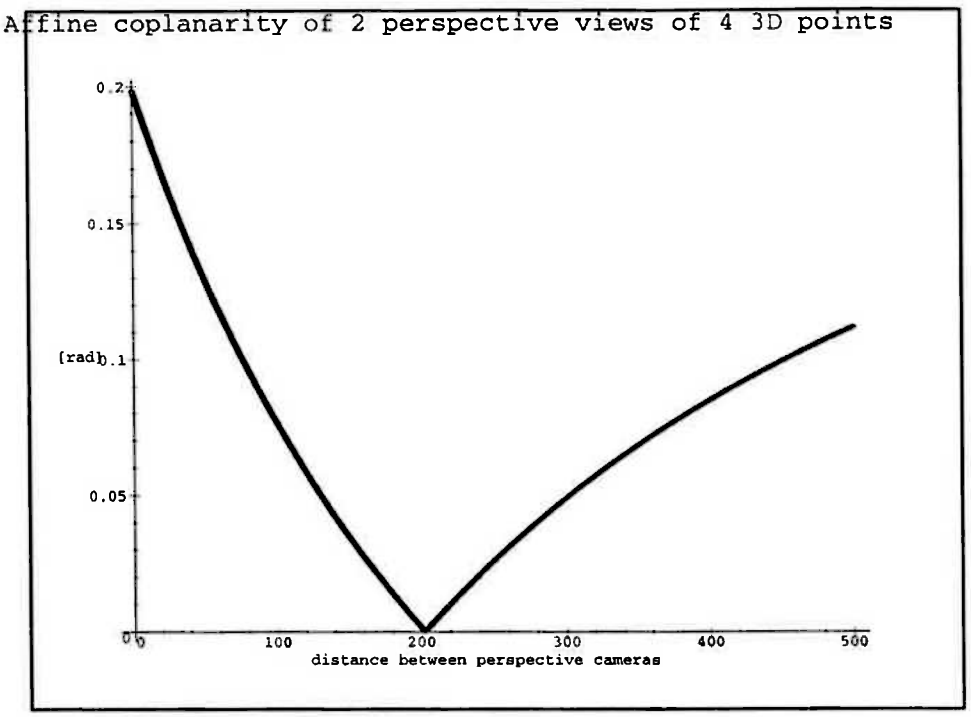


Figure 8.10

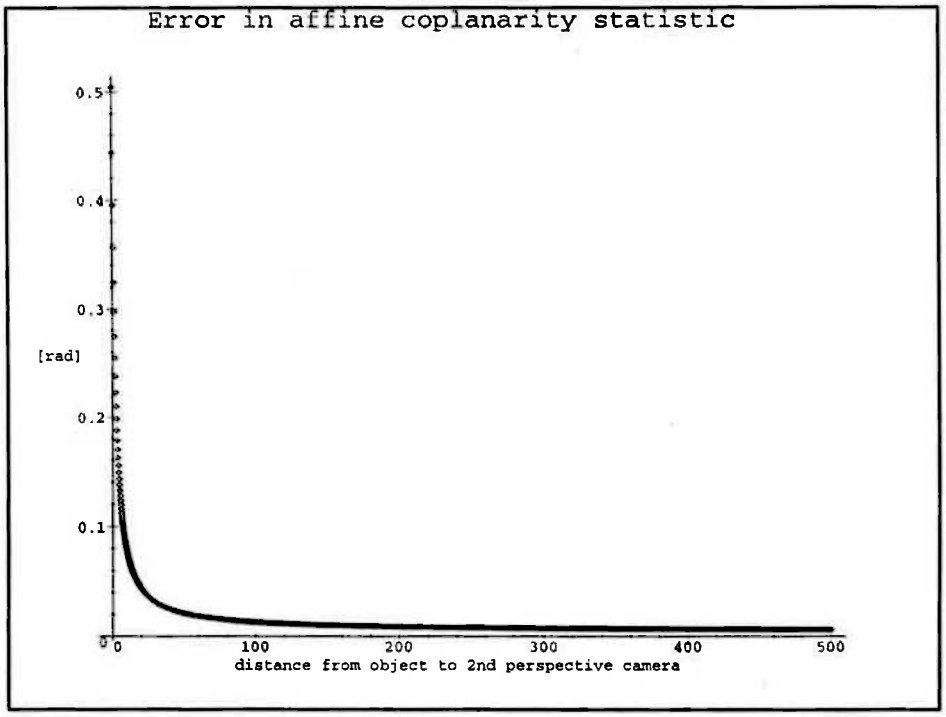


Figure 8.11

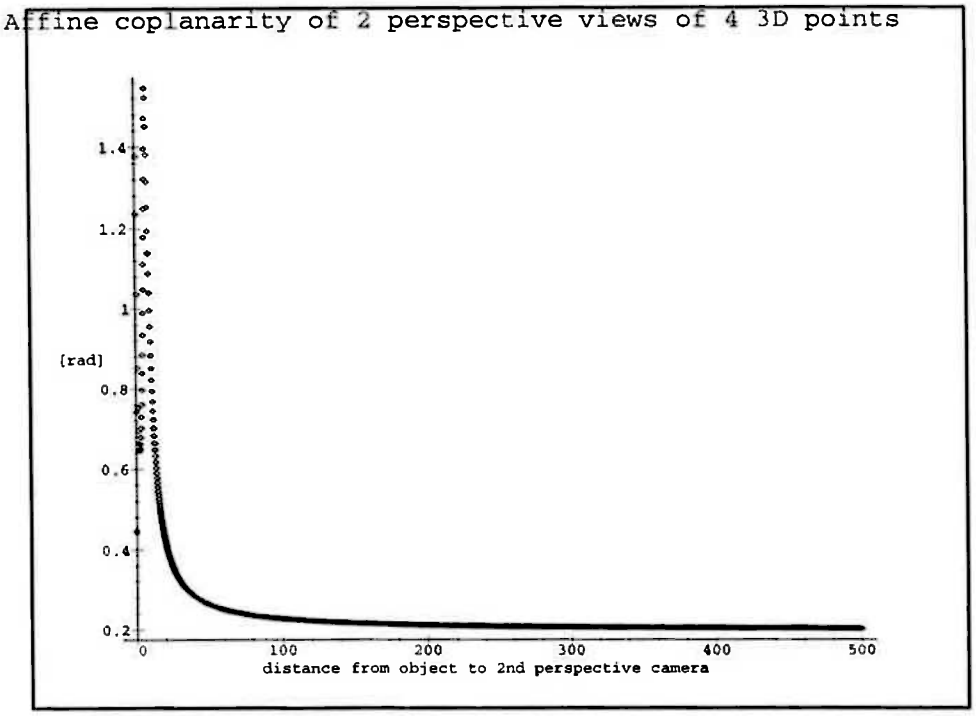


Figure 8.12

8.5 Existence of perspective hypothesis rejection algorithm

In this section, we will prove that an algorithm that is unbiased and stable does exist in the new formulation of the perspective camera model. This result is based on chapters 2, 4 and 5. The derivation of the algorithm is an unsolved optimisation problem.

The reader will recall from chapter 4 that the set $\tilde{M}$ represents non-coplanar scenes, no point of which lies in the focal plane of a camera. The first two rows of each matrix in $\iota(\tilde{M})$ represents an image of such a scene. In chapter 5 we defined an equivalence relation on $\iota(\tilde{M})$. Equivalent matrices in $\iota(\tilde{M})$ represent images of the same scene from different views or cameras, including the possibility that focal lengths differ. The quotient space $\iota(\tilde{M})/\sim$ is metrisable (theorem 5.18). If we let g be a metric on $\iota(\tilde{M})/\sim$, then most of the properties of a correspondence hypothesis test will hold; it remains only to construct a function f on the image plane. For any $W \in \tilde{M}$, let $\tilde{\iota}(W)$ denote the first two rows of $\iota(W)$. The value of this function is the actual image coordinates. Consider

$$f(X,Y) = \inf\{g(W_1,W_2) \mid W_1,W_2 \in \tilde{M},\, \tilde{\iota}(W_1) = X, \tilde{\iota}(W_2) = Y\}$$

If the minimum does not exist, then it is possible that $f(X,Y) = 0$ for some images X,Y that are not images of the same scene. To show that f,g is a correspondence hypothesis test, we have to prove that the minimum always exists.

Lemma 8.13

If $f(X,Y) = 0$ then there are $W_1, W_2 \in \tilde{M}$ such that $g(W_1, W_2) = 0$, $\tilde{\iota}(W_1) = X$ and $\iota(\tilde{W}_2) = Y$.

Proof

Suppose there is a sequence W_i, W_i' in $\tilde{M} \times \tilde{M}$ converging to W, W' such that $\iota(\tilde{W}_i) = X, \iota(\tilde{W}_i') = Y$. Since $\iota(\tilde{W}_i)$ is constant and the function $\tilde{\iota}$ is continuous, $\iota(\tilde{W}) = X$. Likewise $\iota(\tilde{W}') = Y$. From theorem 5.18 $\iota\tilde{M}/\sim$ is metrisable, hence Hausdorff. From lemma 5.16 the set $\{(x,y) \mid x \sim y\}$ is closed. It follows that if $g(W_i, W_i') = 0$ then $g(W, W') = 0$. The lemma follows from this. ∎

Thus f, g are a correspondence hypothesis test.

In this chapter we studied the stereo correspondence problem in its most general form. Once corresponding points have been found, a reconstruction of the scene is possible. We give some new theoretical algorithms for reconstruction in chapter 9.

Chapter 9

Reconstruction from two views

In chapter 8 we studied the correspondence problem, and specified some algorithms. In this chapter we develop some theoretical algorithms for scene reconstruction. Each image of a scene is two dimensional, but using two it is possible to extract three dimensional information about the objects depicted. It is also possible to extract information about the relative orientation of the two cameras, and information about their intrinsic parameters. Relative orientation comprises a rotation and a translation; the rotation renders one camera parallel to the other; the translation moves one camera's optical centre to the other camera's optical centre. The relative orientation problem has a long history in the photogrammetry literature and has been revisited in computer vision [27].

There are many algorithms for reconstruction from corresponding points, some of which are tabulated in table 9.1; many older algorithms are tabulated in table 16.1 in [71]. They differ in many ways, such as the minimum number of corresponding points required, what is assumed about the scene points or the image points, and even the definition of reconstruction. With calibrated cameras there is a unique reconstruction except in certain rare cases. On the other hand, unless some calibration information is provided, a unique reconstruction is impossible; there is a whole set of Euclidean reconstructions compatible with the same input data.

Suppose $x_1, \ldots, x_n$ are points in the scene, whose images give rise to corresponding points in two images, from which a reconstruction is performed. A projective reconstruction comprises points $y_1, \ldots, y_n \in \mathbf{R}^4$ such that $y_i = h \begin{bmatrix} x_i \\ 1 \end{bmatrix} \lambda_i$, where h is a 4×4 non-singular matrix and $\lambda_i \neq 0$.

An affine reconstruction comprises points $y_i \in \mathbf{R}^3$ such that $y_i = h(x_i + t)$, where h is a 3×3 non-singular matrix and $t \in \mathbf{R}^3$ is a translation vector.

In this chapter we will show that the set of depths compatible with eight corresponding points in two images is a four parameter set (when no calibration information is available). The set of x, y, z coordinates or structures compatible with these corresponding points has a further four parameters (it is an eight parameter set). Thus we will produce an eight parameter set of reconstructions, which is much smaller than the projective reconstruction.

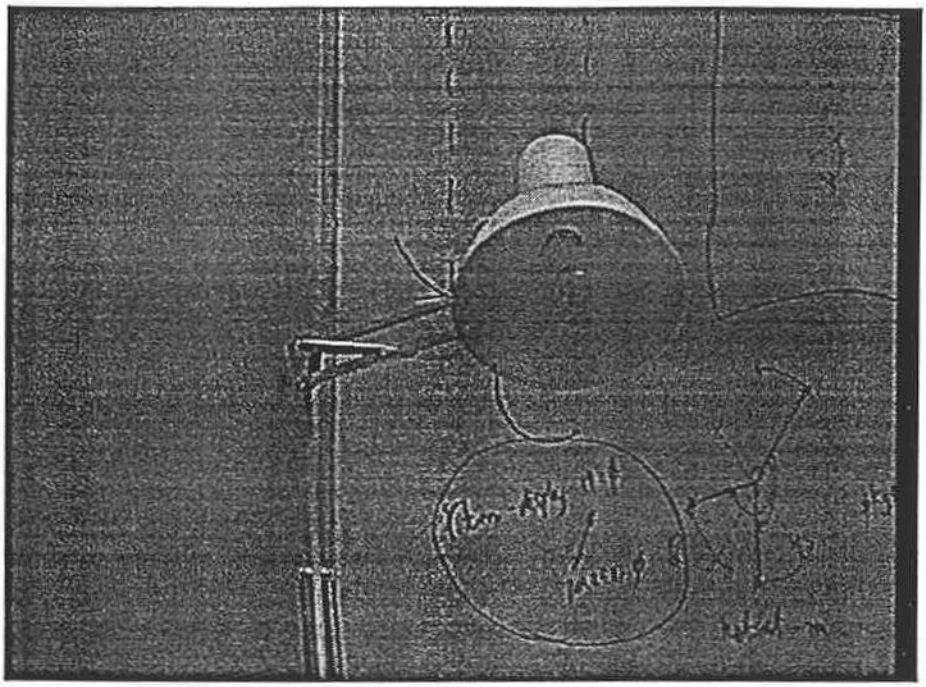


Figure 9.1: left image of a scene

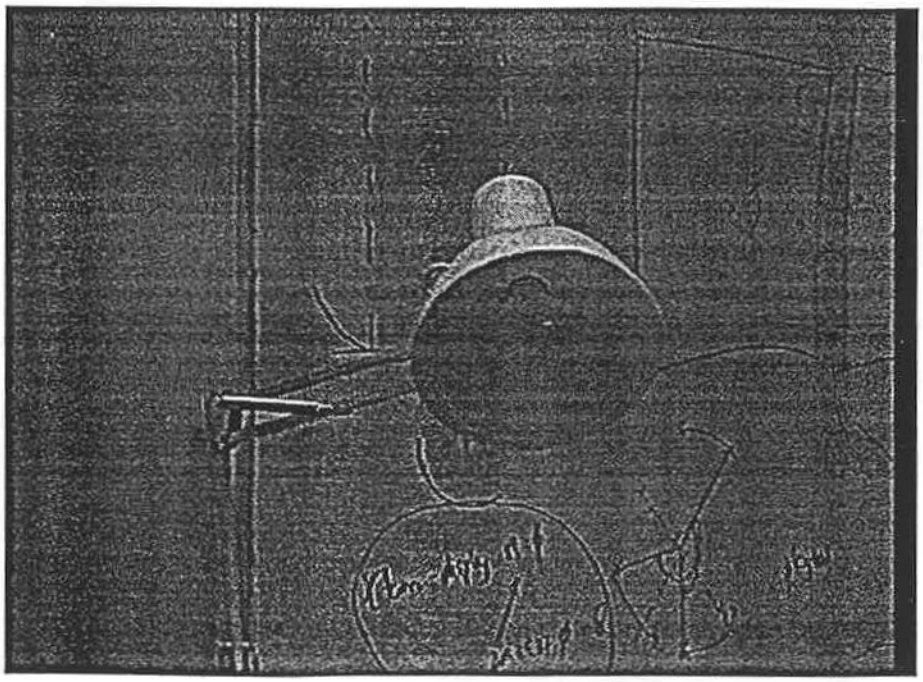


Figure 9.2: right image of a scene

Table 9.1

Reconstruction algorithms from corresponding points			
Name	Other inputs	Outputs	Camera model
Longuet-Higgins	Intrinsic parameters	Relative orientation	Perspective
Faugeras/Hartley	None	Projective reconstruction	Perspective
Sparr's RCP	None	Projective reconstruction	Perspective
Bhavnagri this chapter	None	Eight parameter reconstruction	Perspective
Koenderink and Van Doorn	None	Affine reconstruction	Affine
Hartley/Pan	Intrinsic parameters except focal lengths	Scaled reconstruction and other parameters	Perspective
The author, this chapter	Angles b/w cameras & a principal point	Scaled reconstruction & other camera parameters	Perspective

If the rotation between the cameras and the principal point of both cameras are provided in addition to eight corresponding points, then an algorithm is given to compute all other intrinsic parameters, relative orientation, a reconstruction up to scale, an essential matrix and a fundamental matrix. If the rotation between the cameras and the principal points of both cameras are provided in addition to five corresponding points, where precisely four points are coplanar then we can also compute all other intrinsic parameters, relative orientation, a reconstruction up to scale, an essential matrix and a fundamental matrix. If the rotation between the cameras and the principal point of only one camera are provided in addition to corresponding points, then we can calibrate the camera whose principal point is given, and compute the baseline direction and an essential matrix. It is not necessary to calibrate both cameras to compute an essential matrix. Projective reconstruction can be performed in the case where two views of four scene points that are coplanar and two scene points that are not coplanar are given [8], [43], [127]. According to Zisserman and Maybank [194] there are no projective invariants of five points with four points coplanar and the fifth not coplanar with the other four points; furthermore the epipolar geometry cannot be determined. Thus the algorithm to be presented in this chapter can perform a reconstruction in a case where no projective invariants exist, and when epipolar geometry is unavailable. Also the minimal case of such a reconstruction algorithm requires one less corresponding point than the minimal case of a reconstruction algorithm based on projective invariants.

The constraint for reconstruction is closely related with impossible objects [161], [165], [82].

9.1 Reconstruction without calibration

In this section, we use equation 4.3 to derive a system of quadratic equations whose knowns are image coordinates of corresponding point pairs, and whose unknowns are parameters describing depths in one camera coordinate system, the entries of d, and

Table 9.2

Criteria for correspondence		
Camera model	Assumptions	Constraint
Perspective	Known intrinsic parameters ≥ 8 point pairs	Equation 8.1 (Longuet-Higgins)
Perspective	≥ 8 point pairs	Equation 8.2 (fundamental matrix)
Perspective	≥ 5 point pairs	Corollary 9.4 Sparr's WCP
Perspective	≥ 4 point pairs	Equations 9.2 to 9.5

the transformation between the two camera coordinate systems.

Let X be a $4 \times k$ matrix whose fourth row is $(1, \ldots, 1)$, representing a configuration of $k \geq 4$ points in the scene. Let ιg_1 represent the first camera, and ιg_2 represent the second camera. Suppose no point in the configuration X lies in the focal plane of either camera, and they are not all coplanar. Let

$$\iota g_1 X = \begin{bmatrix} x_1 & \ldots & x_k \\ y_1 & \ldots & y_k \\ 1 & \ldots & 1 \\ 1/z_1 & \ldots & 1/z_k \end{bmatrix}, \quad \iota g_2 X = \begin{bmatrix} x'_1 & \ldots & x'_k \\ y'_1 & \ldots & y'_k \\ 1 & \ldots & 1 \\ z'^{-1}_1 & \ldots & z'^{-1}_k \end{bmatrix} \tag{9.1}$$

The parameters $z_1, \ldots, z_k$ are depths in the first camera's coordinate system, not the scene coordinate system. Likewise, $z'^{-1}_1, \ldots, z'^{-1}_k$ are depths in the second camera's coordinate system, not the scene coordinate system.

We now derive the quadratic equations:

$$\begin{aligned} \iota g_1 X &= g_1 \iota X d_1 && \text{from theorem 4.3} \\ &= g_1 (g_2^{-1} g_2) \iota X (d_2 d_2^{-1}) d_1 && \text{using invertibility of } g_2, d_2 \text{ in their groups} \\ &= (g_1 g_2^{-1})(g_2 \iota X d_2)(d_2^{-1} d_1) && \text{using associativity in groups} \\ &= (g_1 g_2^{-1})(\iota g_2 X)(d_2^{-1} d_1) && \text{from theorem 4.3} \end{aligned}$$

Let $g = g_1 g_2^{-1}$ and $d = d_2^{-1} d_1$ so that the equation is $\iota g_1 X = g(\iota g_2 X) d$.
Expanding this using equation 9.1 we obtain

$$\begin{bmatrix} x_1 & \ldots & x_k \\ y_1 & \ldots & y_k \\ 1 & \ldots & 1 \\ 1/z_1 & \ldots & 1/z_k \end{bmatrix} = g \begin{bmatrix} x'_1 & \ldots & x'_k \\ y'_1 & \ldots & y'_k \\ 1 & \ldots & 1 \\ z'^{-1}_1 & \ldots & z'^{-1}_k \end{bmatrix} d = \begin{bmatrix} (g_{11}x'_1 + g_{12}y'_1 + g_{13} + g_{14}z'^{-1}_1)d_1 & \ldots \\ (g_{21}x'_1 + g_{22}y'_1 + g_{23} + g_{24}z'^{-1}_1)d_1 & \ldots \\ (g_{31}x'_1 + g_{32}y'_1 + g_{33} + g_{34}z'^{-1}_1)d_1 & \ldots \\ z'^{-1}_1 d_1 & \ldots \end{bmatrix}$$

This produces four equations for each corresponding point:

$$\begin{aligned} x_1 &= (g_{11}x'_1 + g_{12}y'_1 + g_{13} + g_{14}z'^{-1}_1)d_1 & \ldots \\ y_1 &= (g_{21}x'_1 + g_{22}y'_1 + g_{23} + g_{24}z'^{-1}_1)d_1 & \ldots \\ 1 &= (g_{31}x'_1 + g_{32}y'_1 + g_{33} + g_{34}z'^{-1}_1)d_1 & \ldots \\ 1/z_1 &= z'^{-1}_1 d_1 & \ldots \end{aligned} \tag{9.2}$$

The variables $x_1, \ldots, x_k$, $y_1, \ldots, y_k$ are coordinates of points in the left image; $x'_1, \ldots, x'_k$, $y'_1, \ldots, y'_k$ are coordinates of points in the right image. The matrix g describes the transformation from the coordinate system of the right camera to the

coordinate system of the left camera, and is unknown. The matrix d is called the relative depth or kinetic depth matrix. The matrix d is initially unknown, but becomes known before applying equation 9.2. Once d has been calculated by using algorithms due to Sparr [163], the equations 9.2 are a quadratic system. We remark that equations 9.2 can then be solved by symbolic algebra, provided the coefficients are calculated and are rational. When the coefficients of a system of polynomial equations are rational, there is an algorithm for solving the system based on the theory of Grobner bases [48], [9], [105]. Unlike other algorithms the entire solution set of the system can be obtained. This algorithm is implemented in Maple's Grobner basis package, and in MAS (which is able to deal with coefficients that are algebraic numbers like $\sqrt{2}$ although no facility for arithmetic with algebraic numbers is provided). We will not need to use this method because we can provide an algorithm to solve equations 9.2. In fact we can solve equations 9.2 in closed form, but this is not as sound as the numerical algorithm.

9.1.1 Computing relative depths d

Let $\mathcal{N}(X)$ denote the nullspace of a matrix X.

Definition 9.1
If V is a vector space in $\mathbf{R}^k$, and q is a $k \times k$ diagonal matrix, let qV denote the vector space spanned by $(qv_1, \ldots, qv_m)$ where $v_1, \ldots, v_m$ are column vectors which are an arbitrary basis of V. The orthogonal complement of V denoted $V^\perp$ is $\{x \in V \mid x.y = 0\}$.

Lemma 9.1
(i) qV is well-defined (does not depend on the choice of basis of V).
(ii) If A, B are vector spaces and $A \subset B$ then $B^\perp \subset A^\perp$.

Proof

(i)

Let $w \in V$, and $v_1, \ldots, v_m$ be a basis of V.
$\Rightarrow w = \lambda_1 v_1 + \ldots + \lambda_m v_m$ for some $\lambda_1, \ldots, \lambda_m \in \mathbf{R}$
$\Rightarrow qw = \lambda_1 q v_1 + \ldots + \lambda_m q v_m$
$\Rightarrow qw \in qV$
Thus in particular for any other basis $w_1, \ldots, w_m$ of V $qw_1, \ldots, qw_m \in qV$, so their span is a subspace of qV. Similarly the span of $qv_1, \ldots, qv_m$ is a subspace of the span of $qw_1, \ldots, qw_m$, so they are equal.

(ii)

Suppose $y \in B^\perp$. Then $y.x = 0$ for all $x \in B$. Since $A \subset B$ for all $x \in A$ $y.x = 0$, so $y \in A^\perp$. Thus $B^\perp \subset A^\perp$. ∎

Lemma 9.2

We have that

$$\mathcal{N}(\begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \\ z_1 & \dots & z_n \\ 1 & \dots & 1 \end{bmatrix}) \subset \text{diag}(z_1, \dots, z_n)\mathcal{N}(\begin{bmatrix} x_1/z_1 & \dots & x_n/z_n \\ y_1/z_1 & \dots & y_n/z_n \\ 1 & \dots & 1 \end{bmatrix})$$

Proof

$$\begin{aligned} \text{rowspace}(\begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \\ z_1 & \dots & z_n \\ 1 & \dots & 1 \end{bmatrix}) &\supset \text{rowspace}(\begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \\ z_1 & \dots & z_n \end{bmatrix}) \\ &= \text{rowspace}(\begin{bmatrix} x_1/z_1 & \dots & x_n/z_n \\ y_1/z_1 & \dots & y_n/z_n \\ 1 & \dots & 1 \end{bmatrix})\text{diag}(z_1, \dots, z_n) \end{aligned}$$

Taking orthogonal complements:

$$\mathcal{N}(\begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \\ z_1 & \dots & z_n \\ 1 & \dots & 1 \end{bmatrix}) \subset \text{diag}(z_1, \dots, z_n)\mathcal{N}(\begin{bmatrix} x_1/z_1 & \dots & x_n/z_n \\ y_1/z_1 & \dots & y_n/z_n \\ 1 & \dots & 1 \end{bmatrix})$$

∎

Given two images of four coplanar points where neither of the images contain four collinear points, it is easy to compute $d_1^{-1}, d_2^{-1}, d_3^{-1}, d_4^{-1}$ up to scale. Let X be the matrix whose columns comprise the coordinates of left image points and 1 in the third entry. Let Y be the matrix whose columns comprise the coordinates of right image points and 1 in the third entry. Since the four left image points are not collinear $\dim\mathcal{N}(X) = 1$ and likewise $\dim\mathcal{N}(Y) = 1$. Since the four points are coplanar $\dim\mathcal{N}(W) = 1$ where W is the matrix whose columns comprise the coordinates of the scene points and 1 in the fourth entry. From lemma 9.2, $\text{diag}(z)\mathcal{N}(X) \supset \mathcal{N}(W)$ and $\text{diag}(z')\mathcal{N}(Y) \supset \mathcal{N}(W)$. Since the dimensions of these three vector spaces are all equal to 1, these three vector spaces are equal. Thus $d\mathcal{N}(X) = \mathcal{N}(Y)$. d is obtained up to scale by dividing the null vector of Y by the null vector of X.

Example 9.1
Consider the four coplanar scene points

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and their images

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

and

$$\begin{bmatrix} 5/3 & 0 & 5/7 & 10/11 \\ -4/3 & -1/7 & -1/7 & 2/11 \end{bmatrix}$$

The first cameras optical centre is at the origin, its optical axis is along the z-axis, its focal length and aspect ratio are 1, and its principal point is at the origin. The other camera is the first camera rotated by

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3/5 & -4/5 \\ 0 & 4/5 & 3/5 \end{bmatrix}$$

Then $\mathcal{N}(X) = \text{span}(1, 1, -3, 1)$ and $\mathcal{N}(Y) = \text{span}(1, 7/3, -7, 11/3)$ so $d = (1, 7/3, 7/3, 11/3)$ up to scale. The depths of the four points are $(3/5, 7/5, 7/5, 11/5)$ in the coordinate system of the second camera, and $(1, 1, 1, 1)$ in the coordinate system of the first camera. Observe that $3d/5 = (3/5, 7/5, 7/5, 11/5)$, so that d really is the ratio of depths in the two camera coordinate systems up to scale. □

The next theorem is due to Sparr, and is the basis for several algorithms to compute d from pairs of views of non-coplanar scene points. See [163], but interchange left and right images [1].

Theorem 9.3
Let $W = W_1, \ldots, W_n$ be $n \geq 5$ points in $\mathbf{R}^3$. If $(x_1, y_1), \ldots, (x_n, y_n)$ is an image of W with a perspective camera and $(x'_1, y'_1), \ldots, (x'_n, y'_n)$ is an image of W with a perspective camera then

$$\dim(\mathcal{N}(\begin{bmatrix} x_1 & \ldots & x_n \\ y_1 & \ldots & y_n \\ 1 & \ldots & 1 \end{bmatrix}) \cap d^{-1}\mathcal{N}(\begin{bmatrix} x'_1 & \ldots & x'_n \\ y'_1 & \ldots & y'_n \\ 1 & \ldots & 1 \end{bmatrix})) \geq n - 4 \tag{9.3}$$

Proof

Let

$$X' = \begin{bmatrix} x_1 & \ldots & x_n \\ y_1 & \ldots & y_n \\ 1 & \ldots & 1 \end{bmatrix}$$

$$Y' = \begin{bmatrix} x'_1 & \ldots & x'_n \\ y'_1 & \ldots & y'_n \\ 1 & \ldots & 1 \end{bmatrix}$$

$$W' = \begin{bmatrix} W_1 & \ldots & W_n \\ 1 & \ldots & 1 \end{bmatrix}$$

From lemma 9.2, $\mathcal{N}(W') \subset \text{diag}(z_1, \ldots, z_n)\mathcal{N}(X')$ and $\mathcal{N}(W') \subset \text{diag}(z'_1, \ldots, z'_n)\mathcal{N}(Y')$. Since no point in W lies in the focal plane of either camera, $\text{diag}(z'_1, \ldots, z'_n)$ is non-singular. Thus

$$\text{diag}(z_1, \ldots, z_n)^{-1}W' \subset \mathcal{N}(X') \cap \text{diag}(z'_1/z_1, \ldots, z'_n/z_n)\mathcal{N}(Y')$$

Since $\text{diag}(z'_1, \ldots, z'_n)^{-1}$ is non-singular it does not change the rank of any minor of W'. $\dim(\mathcal{N}(W')) \geq 4$ implies $\dim(\text{diag}(z'_1, \ldots, z'_n)^{-1}\mathcal{N}(W')) \geq 4$. The theorem follows from this. ∎

[1] Sparr's $q_i = \frac{d(O,W_i)/d(O,(x_i,y_i,1))}{d(O',W_i)/d(O',(x'_i,y'_i,1))} = \frac{f'}{f}\frac{1}{d_i}$ where O, O' are the optical centres of the two cameras.

Corollary 9.4
Let $W = W_1, \ldots, W_5$ be 5 points in $\mathbf{R}^3$. Suppose $(x_1, y_1), \ldots, (x_5, y_5)$ is an image of W with a perspective camera, and not all of these image points are collinear. Suppose $(x'_1, y'_1), \ldots, (x'_5, y'_5)$ is an image of W with a perspective camera, and not all of these image points are collinear. Let v_1, v_2 be a basis of

$$\mathcal{N}(\begin{bmatrix} x_1 & \ldots & x_5 \\ y_1 & \ldots & y_5 \\ 1 & \ldots & 1 \end{bmatrix})$$

Let w_1, w_2 be a basis of

$$\mathcal{N}(\begin{bmatrix} x'_1 & \ldots & x'_5 \\ y'_1 & \ldots & y'_5 \\ 1 & \ldots & 1 \end{bmatrix})$$

Then all 4×4 minors of $[\, v_1 \quad v_2 \quad d^{-1}w_1 \quad d^{-1}w_2 \,]^\top$ vanish.

Proof

$$\begin{bmatrix} x_1 & \ldots & x_5 \\ y_1 & \ldots & y_5 \\ 1 & \ldots & 1 \end{bmatrix}$$

has a null space of dimension two because not all of its points are collinear.
Likewise

$$\begin{bmatrix} x'_1 & \ldots & x'_5 \\ y'_1 & \ldots & y'_5 \\ 1 & \ldots & 1 \end{bmatrix}$$

has a null space of dimension two because not all its points are collinear.
From theorem 9.3

$$\dim(\mathcal{N}(\begin{bmatrix} x_1 & \ldots & x_5 \\ y_1 & \ldots & y_5 \\ 1 & \ldots & 1 \end{bmatrix}) \cap d^{-1}\mathcal{N}(\begin{bmatrix} x'_1 & \ldots & x'_5 \\ y'_1 & \ldots & y'_5 \\ 1 & \ldots & 1 \end{bmatrix})) \geq 1$$

so there is a vector η in both.
$\Rightarrow \eta = \alpha_1 v_1 + \alpha_2 v_2 = \beta_1 d^{-1} w_1 + \beta_2 d^{-1} w_2$ for some $\alpha_1, \alpha_2, \beta_1, \beta_2$
$\Rightarrow \alpha_1 v_1 + \alpha_2 v_2 - \beta_1 d^{-1} w_1 - \beta_2 d^{-1} w_2 = 0$
$\Rightarrow \text{rank}([\, v_1 \quad v_2 \quad d^{-1}w_1 \quad d^{-1}w_2 \,]^\top) \leq 3$
$\Rightarrow$ All 4×4 minors of this matrix vanish. ∎

From two views of four coplanar scene points and two non-coplanar scene points, the relative depths can be computed [83].

Example 9.2
The Maple program exper45 (see appendix B) generates a random configuration of 6 scene points with four coplanar

$$\begin{bmatrix} 92 & -69 & 3 & -72 & 5 & 2 \\ 63 & -48 & 15 & -35 & 8 & 1 \\ 10 & 10 & 10 & 10 & 5 & 1 \end{bmatrix}$$

random left camera parameters

$$u_0 = 4 \quad v_0 = 1 \quad \alpha_u = 92 \quad \alpha_v = 42$$

$$R_1 = \begin{bmatrix} -\frac{1067325}{2385221} & \frac{63986244}{87924181} & \frac{1325299040}{2549801249} \\ -\frac{127260}{2385221} & -\frac{52920635}{87924181} & \frac{2031669276}{2549801249} \\ \frac{208}{233} & \frac{81900}{249077} & \frac{76755}{249077} \end{bmatrix}, \quad t = [8,1,5]$$

and random right camera parameters

$$u_0' = 1 \quad v_0' = 9 \quad \alpha_u' = 61 \quad \alpha_v' = 86$$

$$R_2 = \begin{bmatrix} \frac{2898840}{4075249} & \frac{52388315}{118182221} & \frac{64461996}{118182221} \\ \frac{2819025}{4075249} & -\frac{68428296}{118182221} & -\frac{51002560}{118182221} \\ \frac{32}{257} & \frac{5100}{7453} & -\frac{5355}{7453} \end{bmatrix}, \quad t = [6,6,6]$$

The program computes the left and right images, and computes the relative depth of the four coplanar points up to scale using the method of example 9.1. It then obtains five constraints on d_5, d_6 by applying corollary 9.4 to the first, second, third, fifth and sixth pairs of corresponding points. It solves these, obtaining

$$d_1^{-1} = -\frac{932529123484200}{471544559524763}, \quad d_2^{-1} = -\frac{7335735025}{4740999583}$$

$$d_3^{-1} = -\frac{3750200}{2373841}, \quad d_4^{-1} = -\frac{189717843995400}{100409630168357}$$

$$d_5^{-1} = -\frac{458500652064000}{300498776569289}, \quad d_6^{-1} = -\frac{249593750956800}{219598359684977}$$

up to scale. □

It might appear from corollary 9.4 that if we compute the first four elements of d from two views of four coplanar scene points, then we can compute the fifth element of d from any other corresponding point. Unfortunately, if the first four pairs of corresponding points depict coplanar scene points, the 4×4 minors of the matrix mentioned in corollary 9.4 vanish regardless of the value of the fifth relative depth. Nevertheless, we will be able to use the rotation between two cameras and the principal points of both cameras to compute the fifth relative depth.

We now give an overview of Sparr's algorithm to compute d^{-1} up to scale from eight corresponding points. Let NXi be the matrix whose rows are a basis of

$$\mathcal{N}\left(\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_i \\ y_1 & y_2 & y_3 & y_4 & y_i \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}\right)$$

for $i = 5,6,7,8$. Let NYi be the matrix whose rows are a basis of

$$\mathcal{N}\left(\begin{bmatrix} x_1' & x_2' & x_3' & x_4' & x_i' \\ y_1' & y_2' & y_3' & y_4' & y_i' \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}\right)$$

for $i = 5, 6, 7, 8$. Let

$$A_j = \begin{bmatrix} NXi_{11} & NXi_{12} & NXi_{13} & NXi_{14} \\ NXi_{21} & NXi_{22} & NXi_{23} & NXi_{24} \\ NYi_{11}d_1^{-1} & NYi_{12}d_2^{-1} & NYi_{13}d_3^{-1} & NYi_{14}d_4^{-1} \\ NYi_{21}d_1^{-1} & NYi_{22}d_2^{-1} & NYi_{23}d_3^{-1} & NYi_{24}d_4^{-1} \end{bmatrix}$$

where $j = 1, 2, 3, 4$ and $i = 5, 6, 7, 8$ respectively. From the corollary above $det(A_j) = 0$. The algorithm solves these four equations to express $d_2^{-1}, d_3^{-1}, d_4^{-1}$ as a multiple of d_1^{-1}, but there are two such solutions.
From the corollary a further four equations are obtained

$$det(A_i) = det(\begin{bmatrix} NXi_{12} & NXi_{13} & NXi_{14} & NXi_{15} \\ NXi_{22} & NXi_{23} & NXi_{24} & NXi_{25} \\ NYi_{12}d_2^{-1} & NYi_{13}d_3^{-1} & NYi_{14}d_4^{-1} & NYi_{15}d_i^{-1} \\ NYi_{22}d_2^{-1} & NYi_{23}d_3^{-1} & NYi_{24}d_4^{-1} & NYi_{25}d_i^{-1} \end{bmatrix}) = 0$$

for $i = 5, 6, 7, 8$
Since $d_2^{-1}, d_3^{-1}, d_4^{-1}$ are now known, these can be solved for $d_5^{-1}, d_6^{-1}, d_7^{-1}, d_8^{-1}$ respectively as a multiple of d_1^{-1}. Having obtained two values of d^{-1} up to scale, we eliminate one that does not satisfy

$$\dim(\mathcal{N}(\begin{bmatrix} x_1 & \dots & x_8 \\ y_1 & \dots & y_8 \\ 1 & \dots & 1 \end{bmatrix}) \cap d^{-1}\mathcal{N}(\begin{bmatrix} x'_1 & \dots & x'_8 \\ y'_1 & \dots & y'_8 \\ 1 & \dots & 1 \end{bmatrix})) \geq 4$$

Expanding $det(A_j)$ we see that it is a linear combination of the variables $t_1 = d_1^{-1}d_2^{-1}$, $t_2 = d_1^{-1}d_3^{-1}$, $t_3 = d_1^{-1}d_4^{-1}$, $t_4 = d_2^{-1}d_3^{-1}$, $t_5 = d_2^{-1}d_4^{-1}$, $t_6 = d_3^{-1}d_4^{-1}$:

$$det(A_j) = L_{j,1}d_1^{-1}d_2^{-1} + L_{j,2}d_1^{-1}d_3^{-1} + L_{j,3}d_1^{-1}d_4^{-1} + L_{j,4}d_2^{-1}d_3^{-1} + L_{j,5}d_2^{-1}d_4^{-1} + L_{j,6}d_3^{-1}d_4^{-1} = 0$$

where L can be calculated from the corresponding points.
If rank$L = 4$ then we can solve $Lt = 0$ for t as a function of two of its entries v_1, v_2. Now since $t_1t_6 = d_1^{-1}d_2^{-1}d_3^{-1}d_4^{-1} = t_3t_4$, the equation $t_1t_6 - t_3t_4 = 0$ is quadratic in v_1, v_2. Using this we can solve for v_1 as a multiple of v_2, to which there are two solutions in general.

For each of these two solutions we proceed to calculate d^{-1}. Also,

$$t_4/t_2 = d_2^{-1}/d_1^{-1}, t_4/t_1 = d_3^{-1}/d_1^{-1}, t_5/t_1 = d_4^{-1}/d_1^{-1}$$

from which we obtain

$$d_2^{-1} = t_4/t_2d_1^{-1}, d_3^{-1} = t_4/t_1d_1^{-1}, d_4^{-1} = t_5/t_1d_1^{-1}$$

Thus $d_2^{-1}, d_3^{-1}, d_4^{-1}$ are determined as a multiple of d_1^{-1}.

To decide between the two possible values of d^{-1} we set

$$A = \begin{bmatrix} NX_{11} & \dots & NX_{18} \\ \vdots & & \vdots \\ NX_{51} & \dots & NX_{58} \\ NY_{11}d_1^{-1} & \dots & NY_{18}d_8^{-1} \\ \vdots & & \vdots \\ NY_{51}d_1^{-1} & \dots & NY_{58}d_8^{-1} \end{bmatrix}$$

By theorem 9.3 the rank of A must be 6 if the tuple is d^{-1}.

Algorithm 9.1
(Weak Chasles Problem) To compute d^{-1} up to scale

Inputs	$(x_1, y_1), \ldots, (x_8, y_8)$	left image points
	$(x'_1, y'_1), \ldots, (x'_8, y'_8)$	corresponding right image points
Outputs	d^{-1}/d_1^{-1}	

for $i = 5$ to 8 begin
  evaluate NXi and NYi
  for $j = 1$ to 6 begin
    evaluate $L_{i-4,j}$
  end
end
if the two smallest singular values are not very small, stop.
solve $Lt = [0 \quad 0 \quad 0 \quad 0]^\top$ for t where t is parametrised by v_1, v_2.
solve $t_1 t_6 - t_3 t_4 = 0$ for v_2 (quadratic in v_1, v_2), which obtains two solutions.
for each solution do
  $d_2^{-1} := t_4/t_2 * d_1^{-1}$
  $d_3^{-1} := t_4/t_1 * d_1^{-1}$
  $d_4^{-1} := t_5/t_1 * d_1^{-1}$
  solve $det(A_5) = 0$ for d_5^{-1}
  solve $det(A_6) = 0$ for d_6^{-1}
  solve $det(A_7) = 0$ for d_7^{-1}
  solve $det(A_8) = 0$ for d_8^{-1}
  let NX be a matrix whose rows are a basis for $\mathcal{N}(X)$
  NY a matrix whose rows are a basis for $\mathcal{N}(Y)$
  evaluate the singular values of A
  if two singular values are almost zero, output d^{-1}.
end ∎

The above algorithm was implemented on Maple (exper19). In the next example we will randomly generate eight three dimensional points (configuration X) and two cameras. We simulate two images of the eight points using these two cameras.

Example 9.3

$$X = \begin{bmatrix} -0.1842879905 & 0.8648852811 & 1.865362120 & 0.8582373480 \\ -1.368734879 & -0.002583625889 & -0.9671463792 & -0.1891627713 \\ -1.129758204 & 0.06325214636 & -1.990126081 & -1.175816881 \end{bmatrix}$$

$$\begin{bmatrix} -1.186959320 & 0.8407346002 & 0.7267777490 & 0.1697442866 \\ 0.6767953442 & -1.758357118 & 2.700701005 & -0.01202904051 \\ -1.550475719 & 1.909004669 & 1.124755197 & -0.2883555725 \end{bmatrix}$$

$$g_1 = \begin{bmatrix} 0.3046639789 & -0.704413586 & -6.277816607 & 78.0 \\ -5.986800917 & -0.9517785183 & -0.5023270136 & 21.0 \\ -0.2682614140 & 0.8628408167 & -0.4284174820 & 9.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

$$g_2 = \begin{bmatrix} -9.991403123 & -2.252051904 & -5.576748686 & 104.0 \\ -1.722059542 & -0.983495442 & -5.483360981 & 31.0 \\ -0.5111313932 & -0.5616795819 & -0.6505849263 & 5.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

The left image is

$$\begin{bmatrix} 10.29643683 & 8.910778093 & 10.77088340 & 9.415317672 \\ 2.870241642 & 1.807233041 & 1.379790586 & 1.825692414 \end{bmatrix}$$

$$\begin{bmatrix} 8.223550030 & 10.48390465 & 6.500996926 & 8.808311098 \\ 2.672637147 & 2.590494970 & 1.268459513 & 2.221097600 \end{bmatrix}$$

The right image is

$$\begin{bmatrix} 17.46352214 & 21.02851210 & 16.76241905 & 18.85087631 \\ 5.889426110 & 6.455253720 & 6.738264152 & 6.655357596 \end{bmatrix}$$

$$\begin{bmatrix} 19.72358849 & 20.60125943 & 35.45773827 & 20.34992999 \\ 6.556289082 & 4.822539275 & 8.792543668 & 6.324048955 \end{bmatrix}$$

Applying Sparr's algorithm (Weak Chasles Problem) we obtain

$$d^{-1} = [1, 1.527827212q_1, 1.143428325q_1, 1.324727896q_1,$$
$$1.338687650q_1, 1.178640130, 3.536221131q_1, 1.402409364q_1]$$

Observe furthermore that the actual value of d^{-1}/d_{11}^{-1} is

$$\begin{bmatrix} 1.0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.527827491 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.143428697 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.324728454 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.338688729 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.178610321 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3.536222405 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.402409838 \end{bmatrix}$$

Thus Sparr's algorithm computes the correct value of d up to scale. □

Sparr has also obtained an algorithm to compute d^{-1} up to scale from seven corresponding points [166]. Another such algorithm is outlined in [83].

We give some examples where rank$L < 4$.

Example 9.4
Consider the configuration (a cube)

$$W = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The Maple program exper32 (see appendix B) generates random cameras

$$\iota \begin{bmatrix} 249.0 & 4.0 & -473.0 & 53.0 \\ 238.0 & 92.0 & -431.0 & 36.0 \\ 25.0 & 28.0 & -61.0 & 5.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}, \quad \iota \begin{bmatrix} 690.0 & 390.0 & -965.0 & 75.0 \\ -348.0 & -134.0 & -16.0 & 32.0 \\ 22.0 & 5.0 & -88.0 & 4.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

and computes the rank of L which is 3. □

Example 9.5
The Maple program exper31 (see appendix B) generates a random configuration of 8 coplanar points

$$W = \begin{bmatrix} -5670 & 699 & 2477 & -907 & -5662 & 4947 & 4625 & -2753 \\ 5640 & -228 & -1276 & 164 & 3752 & -4356 & -4396 & 1900 \\ -4860 & -2013 & -4517 & 3349 & 5656 & 1083 & 2821 & 2327 \end{bmatrix}$$

random cameras

$$\iota \begin{bmatrix} 8.0 & -378.0 & 264.0 & 126.0 \\ 473.0 & 140.0 & 152.0 & 67.0 \\ 19.0 & 0 & 60.0 & 9.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}, \quad \iota \begin{bmatrix} -92.0 & 971.0 & -534.0 & 100.0 \\ -15.0 & 420.0 & 114.0 & 36.0 \\ -68.0 & 67.0 & -38.0 & 4.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

and computes the rank of L which was 2. □

We can explicitly calculate g as a function of z'_1, z'_2, z'_3, z'_4 provided the first four points in the scene are not coplanar. Let

$$A = \begin{bmatrix} x'_1 d_1 & y'_1 d_1 & d_1 & z'^{-1}_1 d_1 \\ x'_2 d_2 & y'_2 d_2 & d_2 & z'^{-1}_2 d_2 \\ x'_3 d_3 & y'_3 d_3 & d_3 & z'^{-1}_3 d_3 \\ x'_4 d_4 & y'_4 d_4 & d_4 & z'^{-1}_4 d_4 \end{bmatrix} \tag{9.4}$$

Factoring this into

$$\begin{bmatrix} z'^{-1}_1 d_1 & 0 & 0 & 0 \\ 0 & z'^{-1}_2 d_2 & 0 & 0 \\ 0 & 0 & z'^{-1}_3 d_3 & 0 \\ 0 & 0 & 0 & z'^{-1}_4 d_4 \end{bmatrix} \begin{bmatrix} x'_1 z'_1 & y'_1 z'_1 & z'_1 & 1 \\ x'_2 z'_2 & y'_2 z'_2 & z'_2 & 1 \\ x'_3 z'_3 & y'_3 z'_3 & z'_3 & 1 \\ x'_4 z'_4 & y'_4 z'_4 & z'_4 & 1 \end{bmatrix}$$

we see that A is non-singular because the four points are not coplanar, none of d_1, d_2, d_3, d_4 are zero, and since none of the points are in the focal plane of the second camera $z'_1, \ldots, z'_4$ are non-zero.
The first three rows of equations 9.2 can be expressed by

$$A[g_{11} \quad g_{12} \quad g_{13} \quad g_{14}]^\top = [x_1 \quad x_2 \quad x_3 \quad x_4]^\top$$

$$A[g_{21} \quad g_{22} \quad g_{23} \quad g_{24}]^\top = [y_1 \quad y_2 \quad y_3 \quad y_4]^\top$$

$$A[g_{31} \quad g_{32} \quad g_{33} \quad g_{34}]^\top = [1 \quad 1 \quad 1 \quad 1]^\top$$

Since A is invertible, g is a function of z'_1, z'_2, z'_3, z'_4.

The scale ambiguity in d can be absorbed into a scale ambiguity in g for the purposes of non-metric reconstruction, so that there is an at most four dimensional set of depths up to scale compatible with eight corresponding points, provided four of the associated scene points are not coplanar.

If four points in a scene are coplanar, then $det(A) = 0$, which is a linear constraint on $z'^{-1}_1, \ldots, z'^{-1}_4$, so one of these can be obtained as a linear combination of the others.

If we take the coordinate system of the 8 points in the scene to be equal to the coordinate system of first camera except for intrinsic parameters, that is set

$$g_1 = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

then $z_1, \ldots, z_8$ will be physical depths, and will be determined by $z'_1, \ldots, z'_4$. The x and y coordinates of the points will be determined by the eight parameters z'_1, z'_2, z'_3, z'_4, $\alpha_u, \alpha_v, u_0, v_0$.

Thus Euclidean structure is determined by only 8 parameters, which is considerably less than in projective reconstruction. Thus it appears that equation 4.3 contains much more shape information than does the fundamental matrix.

9.2 Self-calibration and metric reconstruction

Like the *fundamental matrix*, matrix g can be factorised into matrices representing the intrinsic parameters of the left camera, the relative orientation of the two cameras, and the intrinsic parameters of the right camera. However, the factorisation of g is more explicit than the factorisation of the *fundamental matrix*. Suppose the intrinsic parameters of the left camera are $\alpha_u, \alpha_v, u_0, v_0$, the intrinsic parameters of the right camera are $\alpha'_u, \alpha'_v, u'_0, v'_0$, R is the 3×3 matrix representing the rotation from the right camera to the left camera, and T is the translation from the right camera's optical centre to that of the left camera (in the coordinate system of the left camera). Since

$g = g_1 g_2^{-1}$

$$g = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha'_u & 0 & u'_0 & 0 \\ 0 & \alpha'_v & v'_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$

and

$$\begin{bmatrix} \alpha'_u & 0 & u'_0 & 0 \\ 0 & \alpha'_v & v'_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1/\alpha'_u & 0 & -u'_0/\alpha'_u & 0 \\ 0 & 1/\alpha'_v & -v'_0/\alpha'_v & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Expanding out the factors of g we obtain 12 equations between entries of g, intrinsic parameters and relative orientation.

$$\begin{aligned}
g_{11} &= \alpha'^{-1}_u(\alpha_u R_{11} + u_0 R_{31}) \\
g_{21} &= \alpha'^{-1}_u(\alpha_v R_{21} + v_0 R_{31}) \\
g_{31} &= \alpha'^{-1}_u R_{31} \\
g_{12} &= \alpha'^{-1}_v(\alpha_u R_{12} + u_0 R_{32}) \\
g_{22} &= \alpha'^{-1}_v(\alpha_v R_{22} + v_0 R_{32}) \\
g_{32} &= \alpha'^{-1}_v R_{32} \\
g_{13} &= -\tfrac{u'_0}{\alpha'_u}(\alpha_u R_{11} + u_0 R_{31}) - \tfrac{v'_0}{\alpha'_v}(\alpha_u R_{12} + u_0 R_{32}) + \alpha_u R_{13} + u_0 R_{33} \\
g_{23} &= -\tfrac{u'_0}{\alpha'_u}(\alpha_v R_{21} + v_0 R_{31}) - \tfrac{v'_0}{\alpha'_v}(\alpha_v R_{22} + v_0 R_{32}) + \alpha_v R_{23} + v_0 R_{33} \\
g_{33} &= -\tfrac{u'_0}{\alpha'_u} R_{31} - \tfrac{v'_0}{\alpha'_v} R_{32} + R_{33} \\
g_{14} &= \alpha_u T_1 + u_0 T_3 \\
g_{24} &= \alpha_v T_2 + v_0 T_3 \\
g_{34} &= T_3
\end{aligned} \tag{9.5}$$

T is also known as the baseline vector.

9.2.1 Solution using rotation between cameras

We will show how to solve this equation system under quite general conditions. Suppose that the rotation between cameras is known from feedback from motors, or other physical apparatus, and that the principal points of both cameras are also known. The output will include a metric reconstruction (upto scale), the remaining camera parameters, the direction of the baseline vector, the transformation between camera matrices, and relative depths.

More precisely, the inputs are the entries of R, u_0, v_0, u'_0, v'_0, d as a multiple of its last entry, and image coordinates of corresponding points. The quantities to be determined include entries of g, α_u, α_v, α'_u, α'_v, (T_1, T_2, T_3) upto scale, entries of d, $z'_1, \ldots, z'_k$ upto scale, and $z_1, \ldots, z_k$ upto scale. Note that Sparr's algorithm must be used to compute the entries of d upto scale, but the algorithm to be introduced can compute the actual entries of d.

The easiest way to determine the principal point of a camera is to take an image of a static scene , zoom in or out, and then take a second image of the same scene . The second image expands or contracts about the principal point, leaving the principal point fixed. If we choose two pairs of corresponding points, the lines joining each pair should intersect at the principal point. The images in this thesis were acquired by a Sony Handycam video camera. Its focal length varies between 6.1mm and 61mm.[2] It is an autofocus camera with a manual zoom function. We have determined the principal point of images taken by this camera as follows. We take an image of a static scene, zoom in or out, and then take a second image of the same scene. The second image expands or contracts about the principal point, leaving the principal point fixed. The principal point is computed to be $400, 275$, and this was verified by repeating the exercise with different zoom settings and different scenes. The images are 720×575 pixels wide, so the principal point is near the center.

In theory there is an alternate method to find the principal points of a so called stereo head or common elevation platform. A stereo head or common elevation platform is a pair of cameras mounted in such a way that the optical axes of both cameras are in the x-z plane. For such a special camera configuration, the fundamental matrix has a special form. In theory v_0 and v_0' can be computed from six or more corresponding points. When the author tried this using exact arithmetic, no consistent answers were obtained. The fundamental matrix has the right form, but is a very unstable function of corresponding points. The use of a larger number of corresponding points does not change this pathological behaviour. This is also not influenced by the method of choosing corresponding points; whether manually or automatically (by INRIA's program image-matching).

We observe the following relations between entries of g.

$$
\begin{aligned}
R_{31}g_{11} &= R_{11}\alpha_u g_{31} + R_{31}u_0 g_{31} \\
R_{31}g_{21} &= R_{21}\alpha_v g_{31} + R_{31}v_0 g_{31} \\
R_{32}g_{12} &= R_{12}\alpha_u g_{32} + R_{32}u_0 g_{32} \\
R_{32}g_{22} &= R_{22}\alpha_v g_{32} + R_{32}v_0 g_{32} \\
R_{31}R_{32}g_{13} &= -u_0'R_{11}R_{32}\alpha_u g_{31} - u_0u_0'R_{31}R_{32}g_{31} - v_0'R_{12}R_{31}\alpha_u g_{32} - v_0'u_0R_{31}R_{32}g_{32} \\
&\quad + R_{13}R_{31}R_{32}\alpha_u + u_0R_{31}R_{32}R_{33} \\
R_{31}R_{32}g_{23} &= -u_0'R_{21}R_{32}\alpha_v g_{31} - v_0u_0'R_{31}R_{32}g_{31} - v_0'R_{22}R_{31}\alpha_v g_{32} - v_0'v_0R_{31}R_{32}g_{32} \\
&\quad + R_{23}R_{31}R_{32}\alpha_v + v_0R_{31}R_{32}R_{33} \\
g_{33} &= -g_{31}u_0' - g_{32}v_0' + R_{33} \\
g_{13} &= -u_0'g_{11} - v_0'g_{12} + \alpha_u R_{13} + u_0R_{33} \\
g_{23} &= -u_0'g_{21} - v_0'g_{22} + \alpha_v R_{23} + v_0R_{33}
\end{aligned}
$$

Suppose that four corresponding points whose associated scene points are not coplanar are given. Let

$$
X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$

[2]This is not the same quantity as the focal length in the perspective camera model.

Recall that $g = XA^{-1\mathsf{T}}$ where A was defined in the uncalibrated case by equation 9.4. The inverse of A equals $A_c/\det(A)$ where A_c is the cofactor matrix defined by $(A_c^{\mathsf{T}})_{i,j} = (-1)^{i+j}\det(M_{ij})$ and M_{ij} is the matrix obtained from A by deleting its i-th row and j-th column. See [170] for the proof. From the definition of A, all entries in the first three rows of A_c are linear combinations of $z_1'^{-1}, \ldots, z_4'^{-1}$, and all entries in the last row of A_c are constants. The determinant has a cofactor expansion about the last column of A given by

$$\det(A) = A_{41}A_{c4,1} + \ldots + A_{44}A_{c4,4}$$

which is a linear combination of $z_1'^{-1}, \ldots, z_4'^{-1}$. It follows from this that the entries in the first three columns of $g\det(A)$ are linear combinations of $z_1'^{-1}, \ldots, z_4'^{-1}$; the entries in the last column of $g\det(A)$ are constants.

Without loss of generality, we can suppose $u_0 = v_0 = u_0' = v_0' = 0$ because we can always transform image coordinates so that the origins are at the respective principal points. Then

$$\begin{aligned} g_{33} &= R_{33} \\ g_{13} &= \alpha_u R_{13} \\ g_{23} &= \alpha_v R_{23} \\ R_{31}g_{11} &= \alpha_u g_{31} R_{11} \\ R_{31}g_{21} &= \alpha_v g_{31} R_{21} \\ R_{32}g_{12} &= \alpha_u g_{32} R_{12} \\ R_{32}g_{22} &= \alpha_v g_{32} R_{22} \end{aligned}$$

Since the scale of d is unknown, we do not know A but only some multiple of A. So let $A = \lambda A'$ where A' is the matrix we know instead of A, and λ is an unknown scale factor. Since $g = XA^{-1\mathsf{T}}$, $\lambda g = XA'^{\mathsf{T}-1}$. The entries of $XA'^{\mathsf{T}-1}\det(XA'^{\mathsf{T}-1})$ are linear combinations of $z_1'^{-1}, \ldots, z_4'^{-1}$ that can be computed from image data. $\det(XA'^{\mathsf{T}-1})$ is also a linear combination of $z_1'^{-1}, \ldots, z_4'^{-1}$ that can be computed from image data. Let $h = \lambda\det(XA'^{-1\mathsf{T}})g = XA'^{-1\mathsf{T}}\det(XA'^{-1\mathsf{T}})$. From this

$$h_{33} = \lambda\det(XA'^{-1\mathsf{T}})R_{33} \tag{9.6}$$

and

$$h_{13} = \lambda\det(XA'^{-1\mathsf{T}})\alpha_u R_{13}$$

Multiplying both sides by R_{33} and applying equation 9.6

$$h_{13}R_{33} = \lambda\det(XA'^{-1\mathsf{T}})R_{33}\alpha_u R_{13} = h_{33}\alpha_u R_{13}$$

we eliminate $\lambda\det(XA'^{-1\mathsf{T}})$. By definition of h

$$h_{23} = \lambda\det(XA'^{-1\mathsf{T}})\alpha_v R_{23}$$

so as before

$$h_{23}R_{33} = \lambda\det(XA'^{-1\mathsf{T}})R_{33}\alpha_v R_{23} = h_{33}\alpha_v R_{23}$$

we eliminate $\lambda \det(XA'^{-1\top})$. Multiplying both sides of

$$R_{31}h_{11} = \alpha_u h_{31} R_{11}$$

by $h_{33}R_{13}$ we get

$$R_{31}h_{11}h_{33}R_{13} = h_{33}\alpha_u R_{13}h_{31}R_{11} = h_{13}R_{33}h_{31}R_{11}$$

Similarly three more equations are obtained, yielding the system

$$\begin{array}{rcl} R_{13}R_{31}h_{11}h_{33} & = & R_{11}R_{33}h_{13}h_{31} \\ R_{13}R_{32}h_{12}h_{33} & = & R_{12}R_{33}h_{13}h_{32} \\ R_{23}R_{31}h_{21}h_{33} & = & R_{21}R_{33}h_{23}h_{31} \\ R_{23}R_{32}h_{22}h_{33} & = & R_{22}R_{33}h_{23}h_{32} \end{array} \tag{9.7}$$

which is a system of homogeneous quadratic equations in four variables $z_1'^{-1}, \ldots, z_4'^{-1}$. Setting $T_3 = 1$ we obtain

$$h_{34} = \lambda \det(XA'^{-1\top})$$

so that

$$h_{33} = h_{34}R_{33}$$

is a linear equation in $z_1'^{-1}, \ldots, z_4'^{-1}$ because h_{34} is a known constant. One variable, say $z_1'^{-1}$ is thus eliminated, so that we now have a system of four quadratic equations in three variables $z_2'^{-1}, z_3'^{-1}, z_4'^{-1}$. We now discuss how to solve such a system by eliminating variables. We can write this system as

$$\begin{bmatrix} a_1 & b_1(z_3'^{-1}, z_4'^{-1}) & c_1(z_3'^{-1}, z_4'^{-1}) \\ a_2 & b_2(z_3'^{-1}, z_4'^{-1}) & c_2(z_3'^{-1}, z_4'^{-1}) \\ a_3 & b_3(z_3'^{-1}, z_4'^{-1}) & c_3(z_3'^{-1}, z_4'^{-1}) \\ a_4 & b_4(z_3'^{-1}, z_4'^{-1}) & c_4(z_3'^{-1}, z_4'^{-1}) \end{bmatrix} \begin{bmatrix} z_2'^{-2} \\ z_2'^{-1} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where a_i, b_i, c_i are polynomials of degree $0, 1, 2$ respectively. So the rank of the matrix

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ \vdots & \vdots & \vdots \\ a_4 & b_4 & c_4 \end{bmatrix}$$

is less than 3, so all its 3×3 minors vanish. This yields four cubic equations in $z_3'^{-1}, z_4'^{-1}$. However, in all known examples, the cubic terms vanish. These four quadratic equations can be written as the product of a 4×3 matrix and $[\, z_3'^{-2} \quad z_3'^{-1} \quad 1\,]^\top$, so all 3×3 minors of this 4×3 matrix vanish. This yields four cubic equations in $z_4'^{-1}$. Once again, in all known examples, the cubic terms vanish. Solving any of these quadratics for $z_4'^{-1}$, we can then solve the quadratics in $z_3'^{-1}, z_4'^{-1}$ for $z_3'^{-1}$. We can solve for $z_2'^{-1}$ using the original system, and solve for $z_1'^{-1}$ using the linear relation. There are usually two solutions, one of which always seems to be $z_1' = z_2' = z_3' = z_4'$. Substituting these values into the larger equation system, it is straightforward to solve for all remaining variables.

The fact that a camera configuration with coplanar optical axes is a degenerate case for self-calibration was first pointed out by Brooks et al in [27]. The equation system

9.7 degenerates to one equation when the optical axes of both cameras lie in a plane parallel to the yz-plane. We will now explain how to perform metric reconstruction in this case.

We assume the aspect ratios $k = \alpha_v/\alpha_u$ and $k' = \alpha'_v/\alpha'_u$ are known. The rotation R has a special form , with $R_{11} = 1$ and $R_{12} = R_{13} = R_{21} = R_{31} = 0$. We assume that both principal points are known, so without loss of generality let $u_0 = v_0 = u'_0 = v'_0 = 0$.

Under these conditions we obtain

$$\begin{aligned}
h_{11} &= \lambda \det(XA'^{-1^\top})\alpha'^{-1}_u \alpha_u && (9.8)\\
h_{21} &= 0 && (9.9)\\
h_{22} &= \lambda \det(XA'^{-1^\top})\alpha'^{-1}_v \alpha_v && (9.10)\\
h_{31} &= 0 \\
h_{32} &= \lambda \det(XA'^{-1^\top})\alpha'^{-1}_v R_{32} \\
h_{23} &= \lambda \det(XA'^{-1^\top})\alpha_v R_{23} \\
h_{33} &= \lambda \det(XA'^{-1^\top})R_{33} && (9.11)\\
h_{34} &= \lambda \det(XA'^{-1^\top})T_3 && (9.12)\\
h_{24} &= \lambda \det(XA'^{-1^\top})\alpha_v T_2
\end{aligned}$$

and h_{12}, h_{13}, h_{14} are trivial linear combinations of $z'^{-1}_1, \ldots, z'^{-1}_4$ so these are not useful. From equations 9.8 and 9.10 we obtain

$$k' h_{22} = R_{22} k h_{11}$$

which is a linear equation in $z'^{-1}_1, \ldots, z'^{-1}_4$. Setting $T_3 = 1$, from equations 9.11 and 9.12 we get

$$h_{33} = h_{34} R_{33}$$

which is also linear in $z'^{-1}_1, \ldots, z'^{-1}_4$. Either $h_{21} = 0$ or $h_{31} = 0$ provide one more linear equation, because they are usually multiples of each other. Using these three linear equations in $z'^{-1}_1, \ldots, z'^{-1}_4$, we can eliminate three variables. The equation

$$R_{23} R_{32} h_{22} h_{33} - R_{22} R_{33} h_{23} h_{32} = 0$$

is then a univariate quadratic which can be solved for the last remaining variable. Thus two solutions for $z'^{-1}_1, \ldots, z'^{-1}_4$ can be found, one of which always seems to be $z'_1 = z'_2 = z'_3 = z'_4$ which is eliminated by substituting into the larger equation system and solving. We can always solve for α_u/α'_u and α_v/α'_v, which represent the ratio of focal lengths of the two cameras. Using this, a metric reconstruction up to scale can still be performed.

Example 9.6
The Maple program exper38 (see appendix B) generates a random configuration of 5 scene points with four coplanar

$$\begin{bmatrix} 76 & 37 & 82 & 29 & 56 \\ 42 & 47 & 21 & 41 & 85 \\ 70 & 70 & 70 & 70 & 35 \end{bmatrix}$$

random left camera parameters

$$u_0 = 10 \quad v_0 = 6 \quad \alpha_u = 62 \quad \alpha_v = 8$$

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = [9, 4, 5]$$

random right camera parameters

$$u_0' = 10 \quad v_0' = 3 \quad \alpha_u' = 82 \quad \alpha_v' = 75$$

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\frac{31}{481} & \frac{480}{481} \\ 0 & -\frac{480}{481} & -\frac{31}{481} \end{bmatrix}, \quad t = [9, 0, 6]$$

The optical axes of these two cameras lie in a plane parallel to the yz plane. The program computes the left and right images and relative depth of the four coplanar points. It then uses $R_1 R_2^{\mathsf{T}}$, the coordinates of the principal points, and the aspect ratios to solve equation system 9.5. The fifth relative depth is computed in this process. It computes

$$z_1'^{-1} = -\frac{1201}{4861\, A_{2,4}}, \quad z_2'^{-1} = -\frac{1201}{5461\, A_{2,4}}$$

$$z_3'^{-1} = -\frac{1201}{2341\, A_{2,4}}, \quad z_4'^{-1} = -\frac{1201}{4741\, A_{2,4}}$$

$$z_5'^{-1} = -\frac{4804}{38999\, A_{2,4}}$$

baseline direction

$$A_{14} = 0, \quad A_{24} = A_{2,4}, \quad A_{34} = \frac{2591\, A_{2,4}}{4804}$$

left camera focal lengths

$$\alpha_u = 62, \quad \alpha_v = 8$$

and inverse right camera focal lengths

$$1/\alpha_u' = \frac{1}{82}, \quad 1/\alpha_v' = \frac{1}{75}$$

□

We have not as yet given an explicit method for recovering the relative depth d of two views of a five point configuration where four of the points are coplanar and the fifth is not. We will do this for a stereo head camera configuration. For a stereo head, $T_2 = 0$. From the equation system 9.5 $g_{24} = v_0 g_{34}$. Consider the first, second, third and fifth pairs of corresponding points. Since $g = XA^{-1\mathsf{T}}$ we have that

$$g_{24} = -y_1 \det(M_{14}) + y_2 \det(M_{24}) - y_3 \det(M_{34}) + y_5 \det(M_{44})$$

$$g_{34} = -\det(M_{14}) + \det(M_{24}) - \det(M_{34}) + \det(M_{44})$$

where

$$M_{14} = \begin{vmatrix} x'_2 d_2 & y'_2 d_2 & d_2 \\ x'_3 d_3 & y'_3 d_3 & d_3 \\ x'_5 d_5 & y'_5 d_5 & d_5 \end{vmatrix}$$

$$M_{24} = \begin{vmatrix} x'_1 d_1 & y'_1 d_1 & d_1 \\ x'_3 d_3 & y'_3 d_3 & d_3 \\ x'_5 d_5 & y'_5 d_5 & d_5 \end{vmatrix}$$

$$M_{34} = \begin{vmatrix} x'_1 d_1 & y'_1 d_1 & d_1 \\ x'_2 d_2 & y'_2 d_2 & d_2 \\ x'_5 d_5 & y'_5 d_5 & d_5 \end{vmatrix}$$

$$M_{44} = \begin{vmatrix} x'_1 d_1 & y'_1 d_1 & d_1 \\ x'_2 d_2 & y'_2 d_2 & d_2 \\ x'_3 d_3 & y'_3 d_3 & d_3 \end{vmatrix}$$

Recall that d_1, d_2, d_3 can be computed up to scale from corresponding points. Thus M_{14}, M_{24}, M_{34} are scalar multiples of d_5, and M_{44} is a scalar. Thus if v_0 is known, then we obtain a linear equation in d_5, which can be solved for d_5.

The next example is also performed by exper38 in Maple, but the cameras optical axes are skew, and aspect ratios are not used.

Example 9.7
The Maple program exper38 (see appendix B) generates a random configuration of 5 scene points with four coplanar

$$\begin{bmatrix} 25 & 18 & 98 & 80 & 73 \\ 5 & 59 & 15 & 37 & 49 \\ 13 & 13 & 13 & 13 & 86 \end{bmatrix}$$

random left camera parameters

$$u_0 = 8 \quad v_0 = 7 \quad \alpha_u = 98 \quad \alpha_v = 40$$

$$R_1 = \begin{bmatrix} -\frac{200655}{241217} & \frac{1313587564}{2950807561} & -\frac{978074580}{2950807561} \\ -\frac{132300}{241217} & -\frac{2215339455}{2950807561} & \frac{1086385736}{2950807561} \\ -\frac{188}{2213} & \frac{13212360}{27071629} & \frac{23516325}{27071629} \end{bmatrix}, \quad t = [6, 0, 8]$$

random right camera parameters

$$u'_0 = 6 \quad v'_0 = 5 \quad \alpha'_u = 68 \quad \alpha'_v = 30$$

$$R_2 = \begin{bmatrix} -\frac{788640}{2520709} & \frac{13729486444}{16740028469} & -\frac{8018717715}{16740028469} \\ \frac{2381184}{2520709} & \frac{5375914515}{16740028469} & -\frac{1123885460}{16740028469} \\ \frac{505}{5113} & -\frac{16078080}{33955433} & -\frac{29719008}{33955433} \end{bmatrix}, \quad t = [1, 7, 1]$$

The program computes the left and right images and relative depth of the four coplanar points. It then uses $R_1 R_2^\top$ and the coordinates of the principal points to solve equation system 9.5. The fifth relative depth is computed in this process. It computes

$$z_1'^{-1} = -\frac{4016027257801369575}{4657055098209724262\,A_{3,4}}, \quad z_2'^{-1} = -\frac{4016027257801369575}{16557859119609687997\,A_{3,4}}$$

$$z_3'^{-1} = -\frac{1338675752600456525}{1178478472184243919\,A_{3,4}}, \quad z_4'^{-1} = -\frac{1338675752600456525}{3020645378283924469\,A_{3,4}}$$

$$z_5'^{-1} = -\frac{803205451560273915}{8180982189173609374\,A_{3,4}}$$

baseline direction

$$A_{14} = \frac{472370742138948386153\,A_{3,4}}{437746971100349283675}, \quad A_{24} = \frac{302747100234663720739\,A_{3,4}}{437746971100349283675}, \quad A_{34} = A_{3,4}$$

left camera focal lengths

$$\alpha_u = 98, \quad \alpha_v = 40$$

inverse right camera focal lengths

$$1/\alpha_u' = \frac{1}{68}, \quad 1/\alpha_v' = 1/30$$

Essential matrix

$$\begin{bmatrix} \frac{3665035476714978581\,A_{3,4}}{5836626281337990449} & \frac{116220152485831336237\,A_{3,4}}{145915657033449761225} & -\frac{58704812215754337728\,A_{3,4}}{8754939422006985735} \\ \frac{4508961893042346609\,A_{3,4}}{5836626281337990449} & -\frac{93413254263147585624\,A_{3,4}}{145915657033449761225} & \frac{94204417407686383051\,A_{3,4}}{8754939422006985735} \\ -\frac{530500072651352842256\,A_{3,4}}{437746971100349283675} & -\frac{36484649242926153089\,A_{3,4}}{8754939422006985735} & -\frac{3006601751404708021\,A_{3,4}}{145915657033449761225} \end{bmatrix}$$

and Fundamental matrix

$$\begin{bmatrix} \frac{3665035476714978581\,A_{3,4}}{3889527753883636835212 36} & \frac{116220152485831336237\,A_{3,4}}{428992031678342298001500} & -\frac{213003227519970671 8419\,A_{3,4}}{243095484617727302200850} \\ \frac{4508961893042346609\,A_{3,4}}{1587562348523933402 1280} & -\frac{3892218927631149401\,A_{3,4}}{729578285167248806 1250} & \frac{165882598558456660872 53\,A_{3,4}}{595335880696475025798000} \\ -\frac{2399507225631491360620 57\,A_{3,4}}{116685832616509105056408 00} & -\frac{1982581232106852975367 9\,A_{3,4}}{160872011879378361750562 5} & \frac{269829793992883850461166 3\,A_{3,4}}{875143744623818287923060 00} \end{bmatrix}$$

□

In the example above, the equation system 9.5 actually has two solutions; one is rational and the other is irrational. We choose the rational solution, which is the correct solution.

The reader may wonder whether we can self-calibrate using the epipolar constraint. Suppose that we are given five pairs of corresponding points which are images of four coplanar scene points and one non-coplanar scene point; the rotation between the cameras; and the principal points of both cameras. Without loss of generality we can assume $u_0 = v_0 = u_0' = v_0' = 0$ as before. Then

$$F = \begin{bmatrix} \alpha_u^{-1} & 0 & 0 \\ 0 & \alpha_v^{-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -T_3 & T_2 \\ T_3 & 0 & -T_1 \\ -T_2 & T_1 & 0 \end{bmatrix} R \begin{bmatrix} \alpha_u'^{-1} & 0 & 0 \\ 0 & \alpha_v'^{-1} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and the epipolar constraint is

$$[x_i \quad y_i \quad 1]F[x_i' \quad y_i' \quad 1]^\top = 0$$

for $i = 1 \ldots 5$.

It proves to be impractical to solve the epipolar constraint for the unknowns, as the next two examples will demonstrate.

Example 9.8
The Maple program exper40 (see appendix B) generates a random configuration of 5 scene points with four coplanar

$$\begin{bmatrix} 87 & 24 & 82 & 52 & 43 \\ 5 & 53 & 66 & 83 & 42 \\ 41 & 41 & 41 & 41 & 20 \end{bmatrix}$$

random left camera parameters

$$u_0 = 0 \quad v_0 = 0 \quad \alpha_u = 91 \quad \alpha_v = 86$$

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = [7, 0, 9]$$

random right camera parameters

$$u_0' = 0 \quad v_0' = 0 \quad \alpha_u' = 69 \quad \alpha_v' = 39$$

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = [6, 4, 3]$$

The two cameras are parallel to the xy plane.The program solves Longuet-Higgins equation for the unknowns. There are 5 solutions.

$$\left\{t3 = 0, au2 = \frac{364\, au\, (-42354\, av2 + 34529\, av)}{-8736975\, av2 + 2662044\, av}, av = av, au = au, av2 = av2, t2 = 0, t1 = 0\right\}$$

$$\left\{t3 = 0, av = av, au = au, au2 = au2, t2 = 0, t1 = 0, av2 = \frac{602\, av}{377}\right\}$$

$$\{t3 = 0, av = av, au = au, av2 = av2, au2 = au2, t2 = 0, t1 = 0\}$$

$$\left\{t3 = 0, av = av, au = au, t2 = 0, t1 = 0, av2 = \frac{602\, av}{377}, au2 = \frac{650\, au}{609}\right\}$$

$$\left\{av = \frac{39\, av2}{86}, t1 = \frac{23\, au2\, t3}{2}, av2 = av2, au2 = au2, t3 = t3, t2 = -26\, av2\, t3, au = \frac{69\, au2}{91}\right\}$$

□

When the cameras are not parallel, Maple is no longer able to solve Longuet-Higgins equation (in a reasonable time). However, exper40 is not able to exploit the coplanarity constraint on the four pairs of corresponding points that were images of four coplanar points. By way of contrast, the Maple program exper38 succeeds in solving the equation system 9.5 uniquely[3].

Example 9.9
The Maple program exper38 (see appendix B) generates a random configuration of 5 scene points with four coplanar

$$\begin{bmatrix} 8 & 76 & 59 & 5 & 51 \\ 96 & 87 & 63 & 60 & 89 \\ 19 & 19 & 19 & 19 & 29 \end{bmatrix}$$

random left camera parameters

$$u_0 = 6 \quad v_0 = 5 \quad \alpha_u = 87 \quad \alpha_v = 56$$

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = [1, 7, 0]$$

random right camera parameters

$$u_0' = 3 \quad v_0' = 5 \quad \alpha_u' = 55 \quad \alpha_v' = 64$$

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad t = [1, 5, 7]$$

The program computes the left and right images, and computes the relative depth of the four coplanar points. It then uses $R_1 R_2^\top$, the coordinates of the principal points to solve equation system 9.5. The fifth relative depth is computed in this process. It computes

$$z_1'^{-1} = -\frac{7}{26\, A_{3,4}}, \quad z_2'^{-1} = -\frac{7}{26\, A_{3,4}}$$

$$z_3'^{-1} = -\frac{7}{26\, A_{3,4}}, \quad z_4'^{-1} = -\frac{7}{26\, A_{3,4}}$$

$$z_5'^{-1} = -\frac{7}{36\, A_{3,4}}$$

baseline direction

$$A_{14} = 0, \quad A_{24} = -\frac{16\, A_{3,4}}{b1}, \quad A_{34} = A_{3,4}$$

left camera focal lengths

$$\alpha_u = a1, \quad \alpha_v = b1$$

[3]However exper38 does use the coplanarity constraint on the images of four coplanar scene points.

inverse right camera focal lengths

$$1/\alpha'_u = \frac{87}{55\,a1}, \quad 1/\alpha'_v = \frac{7}{8\,b1}$$

Essential matrix

$$\begin{bmatrix} 0 & -A_{3,4} & -\frac{16\,A_{3,4}}{b1} \\ A_{3,4} & 0 & 0 \\ \frac{16\,A_{3,4}}{b1} & 0 & 0 \end{bmatrix}$$

and Fundamental matrix

$$\begin{bmatrix} 0 & -\frac{7\,A_{3,4}}{8\,a1\,b1} & -\frac{93\,A_{3,4}}{8\,a1\,b1} \\ \frac{87\,A_{3,4}}{55\,a1\,b1} & 0 & -\frac{261\,A_{3,4}}{55\,a1\,b1} \\ \frac{87\,A_{3,4}}{5\,a1\,b1} & \frac{21\,A_{3,4}}{4\,a1\,b1} & \frac{351\,A_{3,4}}{20\,a1\,b1} \end{bmatrix}$$

□

Example 9.10

The Maple program exper42 (see appendix B) generates a random configuration of 8 scene points

$$\begin{bmatrix} 6 & 4 & 4 & 10 & 9 & 10 & 1 & 4 \\ 4 & 7 & 8 & 10 & 9 & 4 & 9 & 10 \\ 1 & 3 & 5 & 7 & 3 & 4 & 8 & 5 \end{bmatrix}$$

random left camera parameters

$$u_0 = 3 \quad v_0 = 6 \quad \alpha_u = 1 \quad \alpha_v = 98$$

$$R_1 = \begin{bmatrix} \frac{257697}{789281} & -\frac{15179369664}{58660153201} & -\frac{53327195344}{58660153201} \\ \frac{1630848}{3946405} & \frac{265167328019}{293300766005} & -\frac{6389589756}{58660153201} \\ \frac{8624}{10145} & -\frac{19726356}{57998965} & \frac{4664439}{11599793} \end{bmatrix}, \quad t = [2, 0, 8]$$

random right camera parameters

$$u'_0 = 7 \quad v'_0 = 10 \quad \alpha'_u = 57 \quad \alpha'_v = 60$$

$$R_2 = \begin{bmatrix} \frac{6027840}{16086253} & -\frac{65239216653}{163195036685} & \frac{136516827104}{163195036685} \\ \frac{14317500}{16086253} & \frac{66545176096}{163195036685} & -\frac{33264029253}{163195036685} \\ -\frac{371}{1429} & \frac{2380224}{2899441} & \frac{1474668}{2899441} \end{bmatrix}, \quad t = [1, 4, 4]$$

The program computes the left and right images, and computes the relative depth using Sparr's eight point algorithm. It then uses $R_1 R_2^\top$ and the coordinates of the principal points to solve equation system 9.5. It computes

$$z_1'^{-1} = -\frac{24195627451895610198 9689}{29841922354497229206 2950\,A_{2,4}}, z_2'^{-1} = -\frac{34565182074136585998527}{69977599248891591432500\,A_{2,4}}$$

$$z_3'^{-1} = -\frac{24195627451895610198 9689}{57782588219722301505 0500\,A_{2,4}}, z_4'^{-1} = -\frac{24195627451895610198 9689}{63054110766834479110 3250\,A_{2,4}}$$

$$z_5'^{-1} = -\frac{3456518207413658599852 7}{72328053518503503932575\, A_{2,4}}, z_6'^{-1} = -\frac{2419562745189561019896 89}{32174511197251430866235 0\, A_{2,4}}$$

baseline direction

$$A_{14} = -\frac{2797661058149422449453 0\, A_{2,4}}{3456518207413658599852 7}, \quad A_{24} = A_{2,4}, \quad A_{34} = -\frac{4271534254713820668433 6\, A_{2,4}}{3456518207413658599852 7}$$

left camera focal lengths

$$\alpha_u = 1, \quad \alpha_v = 98$$

and inverse right camera focal lengths

$$1/\alpha_u' = \frac{1}{57}, \quad 1/\alpha_v' = \frac{1}{60}$$

□

9.2.2 An image based test for coincident optical centers

Suppose the optical centres of the two cameras do not coincide.
If $T_3 \neq 0$ then $g_{34} \neq 0$.
Otherwise, since $\alpha_u, \alpha_v \neq 0$ and $T_3 = 0$ one of T_1, T_2 must be non-zero, so one of g_{14}, g_{24} will also be non-zero.
Hence if the optical centers of the two cameras do not coincide, one of g_{14}, g_{24}, g_{34} is non-zero.

If the optical centers of two cameras coincide then $g_{14} = g_{24} = g_{34} = 0$, and equations 9.2 hold. Thus if there exist $g_{11}, g_{12}, g_{13}, g_{21}, g_{22}, g_{23}, g_{31}, g_{32}, g_{33}$ such that

$$\begin{aligned} x_1 &= (g_{11}x_1' + g_{12}y_1' + g_{13})d_1 \quad \ldots \\ y_1 &= (g_{21}x_1' + g_{22}y_1' + g_{23})d_1 \quad \ldots \\ 1 &= (g_{31}x_1' + g_{32}y_1' + g_{33})d_1 \quad \ldots \end{aligned}$$

then the optical centers could be coincident. This gives a linear test for coincident optical centers.

If the optical centers of two cameras coincide then $g_{14} = g_{24} = g_{34} = 0$, and equations 9.2 hold. Thus if there exist $g_{11}, g_{12}, g_{13}, g_{21}, g_{22}, g_{23}, g_{31}, g_{32}, g_{33}$ such that

$$\begin{aligned} x_1 &= (g_{11}x_1' + g_{12}y_1' + g_{13})d_1 \quad \ldots \\ y_1 &= (g_{21}x_1' + g_{22}y_1' + g_{23})d_1 \quad \ldots \\ 1 &= (g_{31}x_1' + g_{32}y_1' + g_{33})d_1 \quad \ldots \end{aligned}$$

then the optical centers could be coincident. This would be a linear test for coincident optical centers, except that the scale of d is unknown. Instead we express the equations

in the form

$$\begin{bmatrix} -x'_1 & -y'_1 & -1 & 0 & 0 & 0 & x_1x'_1 & x_1y'_1 & x_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -x'_k & -y'_k & -1 & 0 & 0 & 0 & x_kx'_k & x_ky'_k & x_k \\ 0 & 0 & 0 & -x'_1 & -y'_1 & -1 & y_1x'_1 & y_1y'_1 & y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -x'_k & -y'_k & -1 & y_kx'_k & y_ky'_k & y_k \\ x'_1d_1 - x'_2d_2 & y'_1d_1 - y'_2d_2 & d_1 - d_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_1d_1 - x'_kd_k & y'_1d_1 - y'_kd_k & d_1 - d_k & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} g_{11} \\ \vdots \\ g_{33} \end{bmatrix} = \mathbf{0}$$

If this system has a solution, the $3k \times 9$ matrix will have a null vector. Using corresponding points from two images, we can use the least singular value of this matrix as a measure of the likelihood of coincident optical centers.

Example 9.11
The coordinates of six pairs of corresponding points were manually extracted from two images using the image viewer xv. The coordinates are roughly within 2 pixels of the correct values. Using Maple we set

$$X = \begin{bmatrix} 456 & 537 & 534 & 452 & 453 & 627 \\ 188 & 192 & 263 & 259 & 323 & 306 \end{bmatrix}$$

and

$$Y = \begin{bmatrix} 172 & 253 & 251 & 169 & 170 & 341 \\ 184 & 189 & 261 & 256 & 320 & 302 \end{bmatrix}$$

and use the Maple program exper51 which reveals the smallest singular value of the $3k \times 9$ matrix to be 0.001395, which is relatively small (compared to other image pairs). Thus the optical centers are almost coincident. In fact, after the first view, the camera was rotated on its tripod, and then the second view was obtained. Since this effectively forms a common elevation platform, we compute F from these six corresponding points, and compute principal point coordinates $v_0 = -743.66, v'_0 = -745.83$ which are absurd. This demonstrates that invalid results might be avoided using this test for coincident optical centers. □

In this chapter we have seen that metric reconstruction can (in theory) be performed from stereo given information that is measurable, and does not change. The most important data required is the rotation between the cameras. Aside from this, corresponding points and the principal points of both cameras are required. The focal lengths are not required to be fixed, equal or known in advance.

Chapter 10

Comparisons across euclidean shape spaces

In chapter 9 we discussed some algorithms for three dimensional scene reconstruction. However, at best the visible part of a solid object will be reconstructed with some scale ambiguity. In this chapter we consider problems that involve testing hypotheses that one shape is part of another shape. Many objects like human beings, articulated vehicles and robot arms are composed of rigidly moving parts.

As we have seen in chapter 8, the correspondence problem requires the ability to find collinear parts of planar shapes. It is interesting to note from chapter 8 that least squares line fitting errors are biased to viewpoint. Moreover, sequences of line fits do not always converge. We will present a collinearity statistic due to Kendall [99] that is not biased when the scene points are all parallel to the retina. Intuitively, we may think that the problem of subjecting a specified subset of image vertices to a collinearity constraint, with minimum perturbation to all points, without bias (to any fronto-parallel viewpoint) is the same. Surprisingly, this is not the case. An optimal solution to this problem was obtained in [108]. As a useful byproduct, the same tests can determine non-uniqueness, which turns out to be a form of spherical symmetry. The value of these tests lie in an interval, with one endpoint representing collinear or degenerate shapes, and the other endpoint representing spherical or spherical-part shapes.

These problems involve comparisons between euclidean shape spaces of different dimensions. In this chapter we will review some useful results of this type in the theory of Euclidean shape spaces.

We will present a new algorithm to the problem of testing two planar shapes to determine whether one is part of the other (based on a conjecture).

We will explain some flaws in the widespread belief in Gaussian point distributions in computer vision, and construct a more sophisticated stochastic process model (which first appeared in the authors publications [16], [18] and [17]).

10.1 Nearest collinear shapes

There are many algorithms in use in computer vision for finding lines. The most prominent are the Hough transform, Kalman filter and many methods based on line fitting. The relation of collinearity is important because it is a degeneracy for the correspondence hypothesis tests in chapter 9. It is also a useful feature because there is a high a priori probability of finding straight edges in man-made environments (some stereo algorithms are based on finding straight-edges [195]);

We will introduce a statistic of collinearity based on shape spaces [99] rather than line fitting or the Hough Transform. As is acknowledged by Zisserman [194] fitting a line to a set of points, and then projecting the points onto the line can lead to large errors along the line itself. More surprising than this is the fact that line fitting is biased against some viewpoints (see chapter 9). The collinearity statistic is similar to the Hough Transform in the sense that shape spaces are also parameter spaces, but unlike the Hough Transform it is a closed form expression. The Hough Transform can also be used to find circles, but the same shape statistic can detect both collinearity and sphericity. It is also possible to find a best fitting collinear shape to the given shape. In the case that the original shape suffers sphericity, the best fit will not be unique, and there will be a whole spectrum of best fitting collinear shapes.

Suppose we are given three points in the plane, and we wish to decide whether they are collinear or not. Since the size, position and attitude of the points are irrelevant to whether or not they are collinear, we can compare the euclidean shape of this point configuration to the nearest euclidean shape all of whose points are collinear. As we saw in chapter 5, the euclidean shape space for three point configurations is a sphere of radius $1/2$, and the set of euclidean shapes of three collinear points is the equator of the sphere. We can see immediately that the desired distance is the distance between a point on the sphere and the equator, because the former is the shape of our three points, and the latter is the set of all collinear shapes. Moreover, the regular polygon (equilateral triangle) shape is on the north and south poles, so if we draw a circle through the north pole, south pole, and the shape we started with:

- This circle intersects the equator in two points, the nearer of which is the nearest collinear shape;
- The distance between the nearest pole and the equator is $\pi/2$, so the sum of the distances to the nearest collinear set and the set farthest from it is constant.

Such a circle is an example of a geodesic on the sphere: a path of least distance, analogous to straight lines in the plane or in three dimensional space. We will show that something very similar happens with shapes containing more points, and state the general formulae.

The collinearity set and the sphericity set are defined as follows:

Definition 10.1

Let W be a $2 \times k-1$ matrix such that $\|W\| = 1$, λ_1^2, λ_2^2 be the eigenvalues of $WW^\top$.

$$Sph = \{[W] \in \Sigma_2^k \mid |\lambda_2| = \lambda_1\},$$

$$Coll = \{[W] \in \Sigma_2^k \mid \lambda_2 = 0\}$$

Coll consists of shapes all of whose points are collinear.

We now state the formula for the distance between a shape and the collinearity set. The translation and size normalised matrix W is obtained by multiplying by $\tilde{Q}$ and dividing by the norm of the original configuration. The set of all such matrices is denoted S_2^k (k is the number of points). The formula for $\rho([W], Coll)$ was first obtained by Kendall (see theorem 8 in [99]) and generalised by Le (see theorem 5 in [106]).

$$\rho([W], Coll) = \arccos(|\lambda_1|) \tag{10.1}$$

$Sph = \{z \in S_2^k \mid \sum_{j=1}^{k-1} z_j^2 = 0\} = \{(1/\sqrt{2}, \pm i/\sqrt{2}, 0, \ldots, 0)S \mid S \in SO(k-1)\}$ where z_j is regarded as a complex number. See [99] for the proof.

The image of a circle consists of a discrete set of points that in some sense are near a regular polygon. Regular polygons embody circularity. We will show that the regular polygons are contained in the sphericity set.

Lemma 10.1
Regular polygon shapes are contained in *Sph*

Proof

Let $w = e^{2\pi i/k}$ be the primitive k-th root of unity,
$z_1^* = 1, z_2^* = w, z_3^* = w^2, \ldots, z_k^* = w^{k-1}$
So $z_c^* = 0$.
$\sum_{j=1}^k z_j^{*2} = 1 + w^2 + w^4 + w^6 + \ldots + w^{2(k-1)} = 0$ since this is a geometric series with sum $\frac{1-w^{2k}}{1-w^2}$.
$\sum_{j=1}^k z_j^{*2} = \sum_{j=1}^k (\bar{z}_j^*) z_j^* =< \bar{z}^*, z^* >$
Since Q is orthogonal, $< \bar{z}, z >=< \bar{z}^*, z^* >$.
Hence $\sum_{j=0}^{k-1} z_j^2 = 0$, but $z_0 = z_c^* = 0$, so $\sum_{j=1}^{k-1} z_j^2 = 0$ which is equivalent to the sphericity condition. ∎

We will next show that a shape is in the sphericity set precisely when it does not have a unique nearest collinear shape. Such non-uniqueness is a signal of symmetry.

Lemma 10.2

$$Sph = \{\sigma \mid \exists \sigma_1, \sigma_2 \in Coll,\ \sigma_1 \neq \sigma_2, d(\sigma, Coll) = d(\sigma, \sigma_1) = d(\sigma, \sigma_2)\}$$

Proof If $\sigma \in Sph$ then $d(\sigma, Coll) = \pi/4$, from formula 10.1 and definition of *Sph*. By Theorem 1 of [99] there is a set $\{\sigma' \mid d(\sigma, \sigma') = \pi/4\}$ identifiable with S^1, because the conjugate of σ is maximally remote from it. This is a subset of *Coll* because we would otherwise contradict theorem 7 [99]. Thus the nearest shapes in *Coll* are not unique. Conversely, if $\sigma \notin Sph$ there is a unique nearest shape in *Coll*, by theorem 8 [99]. ∎

A more general result has been obtained by Le, that is useful for testing whether points in higher dimensional spaces lie in a subspace of lower dimension, such as a line, plane or hyperplane.

The author studied the problem where three points are constrained to be collinear, but other points are not [16], [18], [17]. He developed a closed form expression for such a degeneracy statistic, and for a best fitting degenerate configuration. Regular polygons which approximate circles were shown to have a non-unique best fitting degenerate configuration. After discussions with Le, Le developed a solution for the more general case, where several points are subjected to a collinearity constraint, and several points are not. This result is different to the least squares projection of the points subject to the collinearity constraint onto the best fitting line. The least squares projection method is actually sub-optimal. Moreover the iteration of the three point collinearity constraint does not lead to the optimum either. The author conjectures that with multiple collinearity constraints on disjoint sets of points, the iteration of individual collinearity constraints is sub-optimal.

10.2 Collinear subshapes

A polygon with collinear parts, or coincident vertices is called degenerate. We are interested to know when a shape is approximately degenerate, because we do not want a test of degeneracy which is biased to some viewpoints. The most obvious method of proceeding is to merely fit lines and project points onto lines. As we have already seen, this leads to large errors along the lines, and is biased. Even worse, intersecting these lines to obtain vertices can lead to large distances between the vertex and its associated point. As we have seen on previous occasions, the best way of formulating this problem is to think of a subset of shapes in shape space, and ask for the distance to that subset. In this case we are interested in the distance between a given shape and a set of degenerate shapes. The most general formula known to date is the distance to the set of shapes with certain points being constrained to be collinear. The solution is not known for two or more collinearity constraints.

We may also interested be interested in obtaining a corrected or simplified shape with less points that is not degenerate, but requires minimum perturbation to restore the original shape. This problem has also been solved for shapes subjected to one collinearity constraint. We will again see that symmetry is associated with non-unique solutions to this problem.

10.2.1 Symmetry and non-unique nearest degenerate shapes

We now adopt the convention that arithmetic on indices is to be interpreted modulo k.

Definition 10.2
z^* is degenerate if for some $1 \leq i \leq k$ there is $\lambda \in \mathbf{R}$ such that $z^*_{i+1} - z^*_i = \lambda(z^*_i - z^*_{i-1})$.

Definition 10.3
$D_i = \{\pi_1(z^*)/\|\pi_1(z^*)\| \mid \exists\lambda \in \mathbf{R}\ z^*_{i+1} - z^*_i = \lambda(z^*_i - z^*_{i-1})\}$, π is the euclidean shape transform and, $Deg = \pi(D_0) \cup \ldots \cup \pi(D_{k-1})$.

The set D_i represents those closed polygons whose $i-1$-th, i-th and $i+1$-th vertices are collinear. The set Deg represents euclidean shapes of degenerate polygons.

Surprisingly, there are some degenerate polygon shapes in the sphericity set. Also, collinear polygons are degenerate (which is more obvious).

Lemma 10.3
(i) $Sph \cap Deg \neq \emptyset$ except for $k = 3$. (ii) $Coll \subset Deg$

Proof

(i) Recall that $Sph = \{z \in S^k_2 \mid \sum_{j=1}^{k-1} z_j^2 = 0\}$ where z_j is regarded as a complex number. Thus Sph consists of $(\mathbf{x}, \mathbf{y})$ such that $\mathbf{x}, \mathbf{y} \in \mathbf{R}^{k-1}$,
$\|\mathbf{x}\| = \|\mathbf{y}\|$, $< \mathbf{x}, \mathbf{y} >= 0$ and $\|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 = 1$.
Thus $\mathbf{x}, \mathbf{y} \in S^{k-2}(1/\sqrt{2})$ and $\mathbf{x} \perp \mathbf{y}$. If the pre-shape $(\mathbf{x}, \mathbf{y})$ is degenerate then $\mathbf{x}$ and $\mathbf{y}$ lie on the same hyperplane. If $k = 3$ this is a line, whose intersection with the sphere $S^{k-2}(1/\sqrt{2})$ is a pair of non-orthogonal points. For $k > 3$ the intersection contains a circle, which contains orthogonal $\mathbf{x}$ and $\mathbf{y}$.

(ii) $Coll = \pi(D_0) \cap \ldots \cap \pi(D_{k-1}) \subset Deg$ ∎

Definition 10.4
A smooth function $f : M \to N$ between manifolds M, N is called a submersion if its differential is of maximal rank at all points of M. A manifold M contained in $\mathbf{R}^{n+k}$ is called a submanifold if for all $x \in M$ there is an open set U containing x and a submersion $f : U \to \mathbf{R}^k$ such that $U \cap M = f^{-1}(0)$. The constant k is called the codimension of M.

Example 10.1
Consider the unit sphere in $\mathbf{R}^m$ $\{(x_1, \ldots, x_m) \mid x_1^2 + \ldots + x_m^2 = 1\}$. The function $f : (x_1, \ldots, x_m) \mapsto x_1^2 + \ldots + x_m^2 - 1$ is a submersion, because $df\ |_x = (2x_1, \ldots, 2x_m)$ has maximal rank. Thus the unit sphere is a submanifold of codimension 1. □

The next lemma asserts that the degeneracy set consists of a finite number of submanifolds of the euclidean shape space, each of which can be identified with a lower dimensional euclidean shape space.

Lemma 10.4
D_i is a submanifold of S^k_2 of codimension $m-1$.

Proof

Let $f : \mathbf{R}^{mk+1} \to \mathbf{R}^m, (z^*, \lambda) \mapsto \lambda z^*_{i-1} - (1+\lambda)z^*_i + z^*_{i+1}$.

Then $D_i = \{z \mid \exists\lambda \in \mathbf{R}\ (z^*, \lambda) \in \ker f\}$.

Let $\lambda_j(z^*) = \frac{z^*_{ji} - z^*_{j(i+1)}}{z^*_{j(i-1)} - z^*_{ji}}$,

$g : \mathbf{R}^{mk} \backslash \{z^*_{i-1} = z^*_i = z^*_{i+1}\} \to \mathbf{R}^m, z^* \mapsto f(z^*, \lambda_j(z^*))$.

$dg_j = 0$ and $dg_1, \ldots, \widehat{dg_j}, \ldots, dg_m$ are surjective.
Hence g is a submersion on $D_i Q^\top \backslash \{z^*_{i-1} = z^*_i\}$, onto $\mathbf{R}^{m-1}$.

The $\lambda = 0$ section of D_{i-1} is $\{z \mid z^*_{i-1} = z^*_i\}$. Hence there is a corresponding submersion on this.

It follows that D_i is a submanifold of codimension $m - 1$. ∎

Definition 10.5
Let $\Delta_{i(i-1)} : \mathbf{R}^{mk} \to \mathbf{R}, X \mapsto \det \begin{bmatrix} X_{(i+1)1} - X_{i1} & X_{i1} - X_{(i-1)1} \\ X_{(i+1)2} - X_{i2} & X_{i2} - X_{(i-1)2} \end{bmatrix}$.

This is a shape function.

Lemma 10.5
$S^k_2 - \bar{D}_i$ is disconnected. $\cup_i D_i = \cup_i \Delta^{-1}_{i(i-1)}(0)$.

Proof

If $z^*_{i+1} - z^*_i = \lambda(z^*_i - z^*_{i-1})$ then $z^* \in \Delta^{-1}_{i(i-1)}(0)$ ie $D_i \subset \Delta^{-1}_{i(i-1)}(0)$.
If $\Delta_{i(i-1)} = 0$ then either $\exists \lambda\, z^*_{i+1} - z^*_i = \lambda(z^*_i - z^*_{i-1})$ or $\exists \lambda\, \lambda(z^*_{i+1} - z^*_i) = z^*_i - z^*_{i-1}$.
Thus only z such that $z^*_i = z^*_{i-1}$ and $z^*_i \neq z^*_{i+1}$ are in $\Delta^{-1}_{i(i-1)}(0) \backslash D_i$.
These lie in the $\lambda = 0$ section of D_{i-1}, which is a $2k - 2$-plane perpendicular to two fixed vectors. These are the same two fixed vectors perpendicular to a plane in $\bar{D}_i$, which can be viewed as corresponding to $\lambda = \infty$.
Since $\Delta_{i(i-1)}$ is continuous, surjective and $\Delta^{-1}_{i(i-1)}(0) = \bar{D}_i$, $\Sigma^k_2 - \bar{D}_i$ is disconnected.

∎

The above lemma enables us to classify to which connected component a given non-degenerate shape belongs, by using the signs of determinants. Recall that euclidean shapes presume a specific labelling of points. If points are relabelled, their euclidean shapes will generally change. Regular polygons are an exception: their shape remains unchanged under all cyclic relabellings. A cyclic relabelling also transforms D_i into D_j for some j.

Lemma 10.6
If σ is isotropic under a relabelling g (that is $\sigma = g\sigma$), then $d(\sigma, D_i) = d(\sigma, gD_i)$.

Proof Let σ_d be the nearest shape in D_i to σ, ie $d(\sigma, \sigma_d) = d(\sigma, D_i)$.
Suppose $\exists \sigma_0 \in gD_i \quad d(\sigma, \sigma_0) < d(g\sigma, g\sigma_d)$
$d(\sigma, \sigma_0) = d(g^{-1}\sigma, g^{-1}\sigma_0) = d(\sigma, g^{-1}\sigma_0)$ where $g^{-1}\sigma_0 \in D_i$
$d(g\sigma, g\sigma_d) = d(\sigma, \sigma_d)$
$\Rightarrow d(\sigma, g^{-1}\sigma_0) < d(\sigma, \sigma_d)$ contradicting assumption σ_d is nearest to D_i.
Thus $d(\sigma, g\sigma_d) = d(\sigma, gD_i)$, and $d(\sigma, D_i) = d(\sigma, \sigma_d) = d(\sigma, g\sigma_d)$, so $d(\sigma, D_i) = d(\sigma, gD_i)$. ∎

Definition 10.6

$$Reg = \{\sigma \in \Sigma_2^k \mid \exists Y_1 \neq Y_2 \in Deg\ d(\sigma, Deg) = d(\sigma, Y_1) = d(\sigma, Y_2)\}$$

Reg is analogous to *Sph*, and consists of those shapes whose nearest degenerate shapes are non-unique.

Lemma 10.7
(i) Regular polygons are contained in *Reg*. (ii) If $k > 3$ then $Sph \not\subset Reg$.

Proof

(i) Let σ be a regular polygon, σ_d be any nearest degenerate shape. From the definition of *Deg*, $\sigma_d \in D_i$ for some i. From lemma 10.6 $d(\sigma, D_i) = d(\sigma, D_j)$ for all j. If $\sigma_d \in D_j$ for all j then $\sigma_d \in Coll$. If $\sigma_d \in Coll$ then $d(\sigma, Coll) = d(\sigma, Deg)$ and from lemmas 10.1 and 10.2 the nearest shapes in *Coll* to σ are non-unique, hence $\sigma \in Reg$. Otherwise, $\sigma_d \notin D_j$ for some $j \neq i$, and from lemma 10.6 $d(\sigma, D_i) = d(\sigma, D_j)$ so there is a $\sigma_2 \in \bar{D}_j$ such that $d(\sigma, \sigma_2) = d(\sigma, D_j)$. From lemma 10.5 *Deg* is the pre-image of zero of a continuous function and therefore a closed set. Thus $\sigma_2 \in Deg$ and $\sigma_2 \neq \sigma_d$. Hence $\sigma \in Reg$ in both cases.

(ii) Suppose $Sph \subset Reg$. $Sph \cap Deg \neq \emptyset \Rightarrow Reg \cap Deg \neq \emptyset$ contradicting definition of *Reg*. ∎

10.2.2 One collinearity constraint

Definition 10.7
Let I be a subset of the labels of k points. Let $\mathcal{A}_I$ denote the set of euclidean shapes of k points whose points labelled by I are constrained to be collinear. Let z_I denote the matrix representing the points of z with labels in I.

In the following theorem, the distance of the euclidean shape of z from the subset of shapes subjected to (only one) collinearity constraint $\mathcal{A}_I$ is denoted by $d(\pi(z), \mathcal{A}_I)$; the distance from the shape of the constrained subset z_I to the collinearity set is denoted by $d(\pi(z_I), Coll)$. The theorem gives a relation between these two distances, which depends on the relative size of the constrained subset.

Theorem 10.8

$$1 - \cos^2(d(\pi(z), \mathcal{A}_I)) = \|\pi_1(z_I)\|^2(1 - \cos^2(d(\pi(z_I), Coll)))$$

Proof

See theorem 1 in [108]. ∎

The next theorem tells us precisely when the nearest constrained shape to the shape of z is not unique.

Theorem 10.9

The nearest shapes $\pi(w) \in \mathcal{A}_I$ to $\pi(z)$ are not unique if and only if

$$\cos^2(d(\pi(z), \mathcal{A}_I)) = 1 - \frac{1}{2}\|\pi_1(z_I)\|^2$$

Proof See corollary to theorem 1 in [108]. ∎

An explicit formula for a nearest shape satisfying the collinearity constraint to the given shape is given in [108].

10.3 Computing subshape relation

The main limitation of the discrete model of shape that is embodied by shape spaces is that it is dependent on the number of points. In [21] Blue et al compare a large number of pattern classification methods for optical character recognition and fingerprint recognition. In their work, the Karhunen-Loeve transform is always used regardless of the classification algorithm to reduce the dimensionality of the feature space. There is some similarity between the Karhunen-Loeve transform (also called the Hotelling transform, see [63]) and euclidean shape.

Let A and B be $2 \times k$ matrices representing planar point configurations. Let $\lambda_1 \geq \lambda_2$ be the singular values of $A\tilde{Q}/\|A\tilde{Q}\|$ and $\mu_1 \geq \mu_2$ be the singular values of $B\tilde{Q}/\|B\tilde{Q}\|$. In [107] Le showed the following estimate on the distance between the euclidean shapes of A and B holds:

$$\rho(\pi(A), \pi(B)) \geq \arccos(\lambda_1\mu_1 + \lambda_2\mu_2) \tag{10.2}$$

This estimate is useful because the right hand side is invariant to independent permutations of the columns of A and B, as proved in the next lemma.

Lemma 10.10
If X is a $m \times k$ matrix then $X\tilde{Q}(X\tilde{Q})^\top$ is invariant to permutations of the columns of X. Thus the singular values of $X\tilde{Q}/\|X\tilde{Q}\|$ are invariant to permutations of the columns of X.

Proof

The entry in the i-th row and j-th column of $X\tilde{Q}(X\tilde{Q})^\top$ is the inner product of the i-th row and the j-th row of $X\tilde{Q}$. If x, y are row vectors with k entries, then

$$< x\tilde{Q}, y\tilde{Q} >=< x, y > -k(x_1 + \ldots + x_k)(y_1 + \ldots + y_k)$$

since Q is an isometry. Hence $< x\tilde{Q}, y\tilde{Q} >$ is invariant to permutations of $(x_1, y_1), \ldots, (x_k, y_k)$. It follows that all entries of $X\tilde{Q}(X\tilde{Q})^\top$ are invariant to permutations of columns of X. The singular values of $X\tilde{Q}/\|X\tilde{Q}\|$ are eigenvalues of $X\tilde{Q}(X\tilde{Q})^\top/\|X\tilde{Q}\|^2$, Thus the singular values of $X\tilde{Q}/\|X\tilde{Q}\|$ are invariant to permutations of the columns of X. ∎

However the estimate may be close to zero when the metric is large, as the next example demonstrates.

Example 10.2
Consider the triangle with vertices $(-54.50338767, -14.33529190)$, $(7.217371119, 13.85931563)$ and $(23.14356810, 5.117606396)$. The singular values of its shape are .9882631603 and .1527610088. Also consider the triangle with vertices $(76.893356, -24.750309)$, $(25.253164, 39.428738)$ and $(-14.681376, 47.245914)$. The singular values of its shape are .9850093839 and .1725007625. The estimate is

$$\arccos(0.98826316030.9850093839 + 0.15276100880.1725007625) = .02000647271$$

which is near zero. However, the shape metric between these two triangles is 0.3605413406, which is much larger. If the vertices of the first triangle are taken in the order 321, 312, 231, 213 and 132 respectively then the shape metric has values .5635249663, 1.222279454, .9336416710, 1.235808053, .4234353713 respectively. Thus even if we permute one of the triangles in 6 possible ways, the metric does not decrease. □

We will now use equation 10.2 to find k points of a configuration of $k+n$ points whose shape is most similar to another configuration of k points. Let $A_1, \ldots, A_k$ be a configuration of k points, and $B_1, \ldots, B_{k+n}$ be a configuration of $k+n$ points. Starting with $B_1, \ldots, B_k$, we replace B_1 with each of the remaining points and find the minimum of estimate 10.2. Once this is found, the first point is kept, and the procedure is repeated on each of the k places. After $(n-k)k$ iterations, k points are selected from $B_1, \ldots, B_{k+n}$ that minimise the estimate 10.2 of the distance between the two configurations A and the best fitting subset of B. Once a minimum is found, we need to compute the shape metric itself, and ensure that the minimum is not a local minimum.

10.4 A probabilistic model of dimension reduction

The application of the Kalman filter to line finding is usually based on the assumption of Gaussian point distributions. This is also a common assumption in error analyses of a large number of models and algorithms in computer vision. There are two serious flaws in the assumption that point distributions are Gaussian:

- Triangle shapes have probability zero in a distribution of four or more points, and yet it is obvious that triangles have a non-zero probability of occurring in reality.
- Self-intersecting shapes are impossible as boundaries, yet they have enormous probability in the Gaussian model.

In short, the Gaussian distribution assigns a non-zero probability to events of zero probability and assigns a zero probability to events of non-zero probability. Moreover any of the usual probability distributions will suffer exactly the same problem, because sets of measure zero have non-zero probability.

It is therefore interesting to know that there is a Markov process on unions of euclidean shape spaces whose sample paths travel from shapes in *Reg* to shapes in *Deg*. In what

follows we shall give an existence proof only, the actual probability distributions are not known (at present).

The results in this section were published by the author in [16], and the proofs were in [17].

From theorem 8 in [99] each shape in $\Sigma_2^k \backslash (Sph \cup Coll)$ has a unique nearest shape in $Coll$, and a unique nearest shape in Sph, and lies on the shortest geodesic arc between these two shapes. We want to know whether a similar property holds with Deg instead of $Coll$, and Reg in place of Sph.

Lemma 10.11
Let $\sigma_1 \in \Sigma_2^k \backslash M$, an closed subset M. There is a $\sigma_2 \in M$ nearest to σ_1. Let γ_1 be a geodesic such that $\gamma_1(0) = \sigma_1$, $\gamma_1(1) = \sigma_2$. Then $\forall t \in [0,1]$ σ_2 is a nearest shape in M to $\gamma_1(t)$. Moreover, σ_2 is the unique nearest shape to $\gamma_1(t)$ if it is the unique nearest shape to σ_1.

Proof

A closed subset of a compact set is compact. The function $M \to \mathbf{R}, \sigma_2 \mapsto d(\sigma_1, \sigma_2)$ is continuous in the relative topology of M, so it achieves a minimum. Thus there is a $\sigma_2 \in M$ nearest to σ_1.

Suppose for some $t \in [0,1]$ σ_2 is not a nearest shape to $\gamma_1(t)$. Then $\exists \sigma_3 \in M \backslash \{\sigma_2\}$ such that

$$d(\gamma_1(t), \sigma_3) < d(\gamma_t(t), \sigma_2) \quad (*)$$

So let γ_2 be a geodesic joining $\gamma_1(t)$ to σ_3, and γ_3 be a geodesic joining $\gamma_1(0) = \sigma_1$ to σ_3. (These exist by geodesic completeness of complex projective space.)
Then

$$\begin{aligned} d(\sigma_1, \sigma_3) &\leq d(\sigma_1, \gamma_1(t)) + d(\sigma_3, \gamma_1(t)) \text{ by triangle inequality} \\ &< d(\sigma_1, \gamma_1(t)) \quad + \quad d(\sigma_2, \gamma_1(t)) \quad \text{by} \quad (*) \\ &= d(\sigma_1, \sigma_2) \end{aligned}$$

$\Rightarrow d(\sigma_1, \sigma_3) < d(\sigma_1, \sigma_2)$ contradicting $d(\sigma_1, \sigma_2) = \min_{\sigma \in M} d(\sigma_1, \sigma)$.
Moreover, if σ_2 is the unique nearest shape in M to σ_1 then it is the unique nearest shape in M to $\gamma_1(t)$ for all $t \in [0,1]$. ∎

Suppose $\sigma \in \Sigma_2^k \backslash (Deg \cup Reg)$. Then $\exists \sigma_d \in Deg$ unique nearest shape to σ, and a unique geodesic γ joining them. The next lemma is a weak analogue of theorem 8 in [99].

Lemma 10.12
Let $\gamma(0) = \sigma_d, \gamma(t_0) = \sigma$. $\exists t \in (t_0, \pi/4]\, \gamma(t) \in Reg$.

Proof

Let $\sigma_d \in D_{i_1}, \sigma_1 = \gamma(\pi/4)$.
If $d(\sigma_1, Deg) = d(\sigma_1, D_{i_1})$ then $\sigma_1 = \gamma(\pi/4) \in Reg$.
Otherwise, $\exists i_2\, d(\sigma_1, D_{i_2}) < d(\sigma_1, D_{i_1})$. ie $\exists i_2\, d(\gamma(\pi/4), D_{i_2}) < d(\gamma(\pi/4), D_{i_1})$. But $d(\gamma(t_0), D_{i_2}) > d(\gamma(t_0), D_{i_1}) = d(\gamma(t_0), \sigma_d)$.

$\Rightarrow \exists t_1 \in (t_0, \pi/4)\, d(\gamma(t_1), D_{i_2}) = d(\gamma(t_1), D_{i_1})$.
If $d(\gamma(t_1), Deg) = d(\gamma(t_1), D_{i_1})$ then $\gamma(t_1) \in Reg$.
We can repeat the argument, obtaining a sequence $(t_j)_1^\infty$ monotonically decreasing and bounded below by t_0, such that $d(\gamma(t_j), D_{i_j}) = d(\gamma(t_1), D_{i_1})$.
$(t_j)_1^\infty$ converges to some $t \geq t_0$. In fact $t > t_0$ since $i_j \neq i_1$, $j > 1$. Since $range(t_j)$ is finite, there is a convergent subsequence $t_{l_j} \to t$ such that $i_j = l$, $d(\gamma(t_{i_j}), D_l) = d(\gamma(t_{i_j}), D_{i_1}), \quad l \neq i_1$.
By continuity of d, γ, $d(\gamma(t), D_l) = d(\gamma(t), D_{i_1})$. Hence $\gamma(t) \in Reg$. ∎

Thus we can define the projection of $\Sigma_2^k \backslash Reg$ onto Deg. Let $s = \inf\{t \in (t_0, \pi/4] \mid \gamma(t) \in Reg\}$. Then $\gamma((0, s)) \subset \Sigma_2^k \backslash (Deg \cup Reg)$. Every $\sigma \in \Sigma_2^k \backslash (Deg \cup Reg)$ has a unique geodesic path to Deg contained in $\Sigma_2^k \backslash (Deg \cup Reg)$.

We construct a process on $\Sigma_2^k \backslash Deg$ by restricting the sample paths to take values in such geodesics. Its transition density can be given by the Brownian motion with absorbing barriers at 0 and s, and its initial density can be given by a shape density. Given a Markov transition density, there exists a Markov process with this transition density and a given initial density [35]. By identifying Deg with Σ_2^{k-1}, we can construct a Markov process on $\cup_k \Sigma_2^k$.

Chapter 11

Representational consistency

This chapter presents some evidence that the mathematical models presented in previous chapters could be models of biological vision. We then study some examples that suggest it may be possible to model the phenomenon of awareness. This chapter provides some directions for future research, rather than conclusive results.

Our own visual system would probably be of very little use to us if we could not be aware of what we see. It may surprise the reader to learn that the results of this thesis were arrived at only after making a tentative mathematical hypothesis of what visual awareness is. This hypothesis narrowed down the range of mathematical structures that were possible as representations in the problems of feature detection and stereo. This in turn led to the discovery of the algorithms and correctness proofs in previous chapters. This chapter presents this hypothesis.

This novel approach to the computational study of vision came about when the author noticed a puzzling property of the receptive field experiments, called representational consistency. At roughly the same time, the mathematician Roger Penrose gave a seminar at the author's university, arguing that consciousness needs to be understood before intelligence can be fully understood (see his books [146] and [147]).

The subject of consciousness has always been of interest to medical science in relation to anaesthetics, but has only recently become the subject of study by physiologists. The survey by Crick [39] studies this problem, and will be much used in this chapter.

11.1 Biological evidence for our vision models

11.1.1 Perspective model

Studies of the lens system of the eye, and studies of the retina, provide some evidence that a perspective model holds for the eye. In fact, these studies suggest that the perspective model may hold more accurately in the eye than in a camera.

Light which enters the eye encounters the cornea, the lens, fluids with refractive indices different from air, as well as a membrane in the retina itself before it reaches the

retinal sensors. The lens is known to be a multi-layered structure, with layers of different refractive indices. However, the paraxial model has been verified for the eye[33]. Consequently, the eye could be considered as a multi-element zoom lens.

The retina in an eye is ellipsoidal in shape [33], whereas we assumed the retina to be a plane perpendicular to the optical axis (see chapter 2). The retinal light sensors only detect light that hits them from a narrow range of directions, because they are waveguides[33]. The preferred directions are known to be aimed at the center of the aperture of the eye (Stiles-Crawford effect of the first type). From lemma 2.1, there is an optical centre in the object space of the lens system of the eye. This effect suggests that the eye is designed to provide an accurate perspective model. The light sensors in a camera do not employ waveguides with preferred directions aimed at a point; thus there is no accurate pencil of rays on the image side of the lens system of a camera.

For objects that are reasonably far away, we can infer that an affine model is applicable. Psychophysicists often employ affine models of vision [177], [102].

The retina in a primate eye has two types of light sensors, rods and cones. The cones are colour sensors, whereas the rods are more akin to greyscale sensors. There are three types of cones: each type has different wavelength sensitivities. The retina contains a very high density area of cones called the fovea, in which most visual acuity is concentrated. The remainder of the retina is called the periphery and is populated with rods, but much less densely than the fovea, and extends to a wide angle. Thus the arrangement of sensors in the retina is not as periodic as was our digital geometry (in chapter 3).

11.1.2 Type II feature detection

Most experiments on vision discover a physical representation in the brain of some visual stimulus or cognitive experience. For example, in [93] electrodes are inserted into the primary visual cortex of anaesthetised monkeys, and it is found that the electrical response corresponds to tangents to contrast boundaries in the visual stimulus, within a certain range of positions (receptive fields) and within a certain range of orientations (orientation dominance). Anatomists attempt to map out the regions of the brain which have specific functions (i.e. represent some specific type of sensation). They also develop a map of the well-used connections between regions. The book by Zeki [192] provides an introduction to the anatomy of vision. A recent map of the visual areas of the brain and their inter-connections may be found in figure 52 of [39].

We will adopt a formalism that is based on the affine model, but is independent of a particular grid. As to the ellipsoidal shape of the retina, it is likely the model can be altered to accommodate this without losing any utility; we will ignore this issue.

The retina in an eye does not merely act as a relay to the brain. The process of vision begins in the retina itself. Several types of neurons have been identified in the retina, and the way they respond to stimuli has been studied. The ganglion cells are the type that transmits information to the brain via the optic nerve. The part of the visual field that influences a particular nerve cell is called its receptive field. The receptive fields

of retinal ganglion cells have been measured using spot stimuli by H.B. Barlow [7] and S.W. Kuffler [104]. In total darkness, a ganglion cell usually fires at a low, irregular rate called its background rate. For one type of ganglion cell, the on-centre type, its firing will increase dramatically if a small spot of light is shone onto the very center of its receptive field. This small center has a circular region surrounding it where exactly the opposite happens. If the spot falls entirely in this annular region, the background rate of firing ceases. If the spot is then turned off, there is a burst of rapid firing. In addition to on-center cells, there are an equal number of off-center cells. Loosely speaking the cells fire when a spot in the middle of their receptive field is turned off. This is analogous to our thresholding operation with two types of NWES-grid points in chapter 3. Retinal ganglion cell receptive fields have also been measured for sinusoidal grating stimuli by C. Enroth-Cugell and J.G. Robson [49], with a quantitative model. Their non-linearity becomes evident from this model, which is why there is only an analogy with chapter 3. It is important to note that if the optic nerve is severed, there is a total loss of vision. Thus visual awareness does not emerge from the retina itself. We will have to explain this using our model of visual awareness later in the chapter.

The retina is connected to two main regions of the brain, called the lateral geniculate nucleus, and the superior colliculus. The lateral geniculate nucleus is in turn connected to the visual cortex. The superior colliculus is mainly concerned with control of eye movements (see page 127 in [39]). The lateral geniculate nucleus also receives input back from the first visual area (V1) of the cortex. This could be related to the generation of loops outlined in chapter 3. However, we are not aware of the individual loops, so this must be explained. It is well-known in physiology that short term memory is associated with reverberatory neural circuits.

The phenomenon of orientation dominant receptive fields was discovered by physiologists D.H. Hubel and T.N. Wiesel in their experiments on primate primary visual cortex [93]. In the experiments [93] an anaesthetised primate is presented with a simple shape such as a line or square on a projector, and the responses of certain cells in the primates primary visual cortex (area V1) to these stimuli is measured. It was found that the simple cells respond optimally to line edges in a particular orientation and position in the field of view. As the orientation or position is varied, the response eventually falls to zero.

A planar point configuration represents points on the edge of a shape being viewed by the eye. If the configuration is viewed from a different fronto-parallel viewpoint, its retinal image will be transformed by a positive affine transformation. We can express this in terms of positive affine shape spaces (see chapter 5) as follows. The simple cell encodes the orientation and position of each tangential line L_i, which determines intersections $L_i \cap L_j$. Lemma 5.9 asserts that the matrix $[(\Delta_{ij}, \lambda_{ij})]$ is a representation of the intersections $L_i \cap L_j$, which possesses invariance to affine transformations with positive determinant. Moreover, the response of such cells varies with length of line and width of bar, is optimal for a straight edge, has an optimal orientation, and declines to zero well before ninety degrees to the optimal. These support the idea that an intrinsic intersection measurement is being made, but confined to a particular vertex sign.

Recall from chapter 3 that we proved that when integral curves of vector fields intersect, there is either an occlusion, a discontinuity in intensity, a discontinuity in a derivative of intensity, a discontinuity in the surface being viewed, or a discontinuity in a derivative of a surface being viewed. It was long ago postulated by Hoffman [86] that the primary visual cortex generates vector fields. If the vectors are joined in a head to tail fashion, integral curves will be obtained. An intersection measurement model of orientation dominant cells in area $V1$ of primate visual cortex therefore implies that these kinds of features are being detected. The experiment itself confirms that this is indeed the case. It is therefore not necessary to make any further assumptions. Note however that the specific vector fields that are being generated are unlikely to be Hamiltonian fields, and we will see that there will be at least two vector fields being generated at the same time.

The cells in area $V1$ are also reported to be involved in binocular interaction. We have argued that there is a natural data structure associated with intersections of integral curves (in chapter 3), and that a correspondence hypothesis table can be generated from two such data structures (in chapter 8), provided the surfaces in the scene are approximately Lambertian.

Pairs of anatomically adjacent simple cells which are selective for visual stimuli of the same orientation, spatial frequency, direction and spatial location exhibit responses to a drifting sine-wave grating in either quadrature phase or anti-phase (see [112]). We infer from this that at least two vector fields are implicit. Away from occlusions and discontinuities these span a parallelization.

The later experiments by von der Heydt and Peterhans [174], [175] report orientation specific cells responding to illusory lines in area V2 of the visual cortex of alert rhesus monkeys. In other words, they discovered cells encoding the intersection of vectors or line elements, not edges or corners. This further reinforces the idea that shape coordinates are being constructed because these involve intersections of vectors or line elements, not edges. The reader may recall from chapter 3 that the connection with occlusions and discontinuities relies on the interaction between the geometry of complete surfaces including their hidden parts and the geometry of the retina. Completion was part of our algorithm, not an anomaly of some kind to be explained away. In addition, the experiments report some dependence on configuration of lines, which is to be expected if shape coordinates are being constructed. However, this explanation is too simple in itself, because we can see illusory contours of somewhat irregular shapes. This will be explained in terms of a mapping resembling a reflection called an involution later in this chapter.

11.1.3 Stereo vision

There is some evidence that rotation is used in binocular vision. It has been known for some time that the human vestibular apparatus (which provides our sense of balance) is linked to vision. The cause of motion sickness is considered to be a conflict between the vestibular and visual senses. Also, stimulus-response studies of three dimensional recognition of shapes in rotated views show that the response time for recognition is

proportional to the angle of rotation between the object in the two views.

Our results in chapter 9 show that the sense of balance, together with feedback from ocular muscles can provide enough information to an animal's visual system to build a three dimensional representation of the scene. The sense of balance, together with feedback from ocular muscles provides information on the rotation between an eye at one point in time and the same eye at another point in time. The rotation between two different eyes at the same point in time only requires ocular muscle feedback.

We studied two different kinds of tests of hypotheses of correspondence. The first kind of test was obtained from the metrics of shape spaces in chapters 4 and 5. We proved that unbiased and stable correspondence hypothesis tests do not exist for projective invariants of general non-planar scenes, and therefore that no satisfactory correspondence hypothesis test exists in the epipolar geometry model. We also proved that there is a shape space if the retina is assumed to be planar, and that a correspondence hypothesis test exists. Moreover, in the affine camera model we gave a correspondence hypothesis test, and it was bounded (as any unbiased function must be). The affine shape coordinates mentioned above can distinguish between self-intersecting and non-self intersecting shapes, which is of essential utility on an irregular grid.

A (not necessarily planar) closed polygon on an opaque surface gives rise to an invariant string, formed from the signs of Δ_{ij}. Occlusions cause changes in the string according to the rules of a fixed grammar introduced in chapters 6 and 7. This furnishes the second type of correspondence hypothesis test in chapter 8. The Hubel-Wiesel experiment reported that the striate cortex has a hierarchical and layered architecture. This could be explained by a network of neurons organised in a graph structure according to the grammar. The neurons in V1 with orientation specific receptive fields are arranged in columns; the preferred orientation progressively changes down a column, interposed with unpredictable jumps and reversals. Since there are approximately 2^{100} string descriptors with 100 signs, the exhaustive representation of them would require far more than the 200 million neurons of the primary visual cortex. If only useful string descriptors were represented, this would explain the jumps.

11.1.4 Shape distributions

There is also a probabilistic interpretation of receptive fields. We will now discuss this. Suppose a retinal light receptor responds to a point source. Then there is an uncertainty in the position of that point source. Suppose we treat the light receptor as if it were a light source; a pattern of light would be observed on a plane in front of the eye. This would be the point spread function of the optic system in reverse. In geometric optics, the optical paths are the same regardless of the direction of light[23]; rays are reversible in geometric optics[1]. Thus the probability distribution of position of a point source given a point sensor's response must be that point spread function.

For the diffraction limited or aberration free eye model, the point spread function is approximately Gaussian. This is because the line spread function is the Fourier

[1] but not in wave optics

Transform of $T(R)e^{i\psi(R)}$, T is approximately Gaussian, $\psi = 0$ for the diffraction limited eye, and the Fourier Transform of a Gaussian is Gaussian. The symmetry of the point spread function for the diffraction limited case implies the point spread function is Gaussian. See [33] for a review, but note that these facts may not imply the point spread function of the eye in reverse is Gaussian. In any case, the receptive field of a retinal light receptor is associated with a probability distribution. The distribution induced on shape space by Gaussian point distributions has been explicitly found by K.V. Mardia and I.L. Dryden [119].

We can hypothesise that the response of the simple cell is a probability distribution on positive affine shape space. However, since boundaries cannot be self-intersecting curves, there is a set of shapes which must be excluded.

11.2 Awareness

11.2.1 Physiological and psychological background

The phenomenon of consciousness has long been the subject of study by psychologists and various disciplines of medical science. It has also been the subject of many long-standing philosophical debates. The book by Crick provides an excellent review from a neuroscientist's viewpoint [39]. Some of the general features of consciousness are:

- it is closely associated with attention;
- it has a close relation with short term memory, which is based on reverberatory neural circuits;
- neural activity can take place without evoking awareness;
- it is a phenomenon which emerges from large scale neural activity;
- learned behaviour can become automatic, and have little to do with awareness;
- awareness is lost under the influence of anaesthetic.

This last fact is probably the most important, since most physiological experiments are performed on an anaesthetised subject. The subject cannot be aware of the representations that were observed. On the other hand, most experiments are not performed on humans, so we cannot be sure that the subjects are conscious, and they cannot communicate what they are aware of. This thesis will only consider a very specific aspect of consciousness: visual awareness. The many other forms of consciousness such as awareness of pain, thinking, emotion and self-awareness will not be considered.

There are several broad categories of experiments on visual awareness outlined in [39]:

- study which kinds of visual awareness are lost when particular regions of the brain associated with vision are damaged;

- compare an alert brain response with a slow-wave sleep response;
- study how direct electrical stimulation of particular neurons influences visual awareness;
- artificial retinal stabilization;
- experiments that discover binding;
- experiments that discover reverberatory circuits in the visual cortex and lateral geniculate nucleus;
- experiments to study the mechanisms of attention.

It is known that if a person or monkey has extensive damage to all parts of V1, he is almost totally blind for that half of the visual field (see page 132 of [39]). Thus a functioning V1 is necessary for visual awareness.

There is reported to be a difference between the experience of seeing (conscious perception), and recognition, which is demonstrated by two pathological conditions: blindsight and aphasia. In blindsight the patient reports he cannot see [180], but when asked to guess what is within his visual field is able to recognise accurately. In aphasia the patient reports he can see, but cannot recognise. Both conditions can be selective: affecting certain parts of visual field and not others, or certain types of sensations (objects, colours etc) and not others. The experiment that first reported blindsight [180] is now one of many experiments that demonstrate that perception is possible without awareness.

There are different types of sleep, rapid eye movement (REM) sleep, and slow-wave sleep. In REM sleep the brain waves closely resemble those of an alert brain. This suggests that the brain is at least partly conscious, as we seem to be in our dreams. In slow-wave sleep the brain waves are very different from those of an alert brain; very few if any dreams occur. This suggests that the brain is not conscious in slow-wave sleep.

The eye undergoes tiny involuntary movements called micro-saccades. Experiments that attempted to stabilize an image with respect to the retina of a human revealed that the image fades completely very soon after stabilization. This could be caused by the retina itself, but Crick argues that there is evidence against this (see page 223 of [39]).

Awareness is thought to be related to what is called the binding problem. There cannot be a neuron which encodes every possible percept, because the combinatorial possibilities are too numerous. The binding problem is to determine how several neurons become temporarily active as a unit. Some recent experiments suggest that special binding, where needed for operations such as object tagging takes the form of coordinated firing, often with rhythms in the 40Hz range [39]. Our theory will have to explain why our visual awareness seems connected even though the retina only produces a discrete array of signals, which is a form of binding.

Any mathematical model of visual awareness will have to be consistent with all these experiments. This is rather challenging; the author's approach is to put forward a mathematical condition such that

- Given a model of a brain function such as type II feature detection; and
- given a representation produced from physiological experiments, together with data as to whether or not this representation is perceived consciously; then
- if this representation is perceived consciously, the condition is true; otherwise the condition is false.

If this were true of all known models of mechanisms of vision, then this would substantiate such a model of awareness.

As a somewhat tentative hypothesis, we will put forward such a condition, called representational consistency. We will then evaluate whether or not this property holds for some of the models presented in previous chapters. However, the hypothesis we will propose was used to determine particular types of representations, which led to the specific algorithms in previous chapters. These appear to be consistent with the basic architecture of biological vision (in animals whose visual apparatus resembles the human's). This makes such a theory much more amenable to experimental testing than qualitative theories. It also makes it more useful in an engineering sense.

The basic idea of representational consistency can be introduced by considering what the orientation dominant receptive fields could represent. The orientation dominant receptive fields could represent:

- one or more vector fields; or
- one or more fields of line elements (distributions); or
- the intersections of integral curves; or
- the intersections of tangents to edges; or
- a probability distribution of one of the above.

There are obviously many possible representations, and it is not acceptable for the representation to depend on the interpretation of the person conducting the experiment. If the representation is not there when no one (except the subject of the experiment) is present, it cannot really exist! To consider some of the great difficulties inherent here, consider the following simple example. The fingers of my hand can represent one, two, three, four and five; but they can also represent two, one, three, four and five respectively. Both these representations exist simultaneously, so we can pair them up to give the pair one,two repeated twice, and three, four and five, which no longer has the structure of five elements. These representations are called inconsistent. In this chapter, we will introduce a definition of representations and of their consistency.

11.2.2 The hypothesis

Definition 11.1
Let S be a set that satisfies a collection of relations, called the structure of S. An interpretation is a set that satisfies the same collection of relations, called the structure of the interpretation. A representation of S is a one to one mapping from S to an interpretation, which preserves the structure of S. The set of representations of S will be denoted by $Rep(S)$.

The reader should note that the definition of a representation does not require it to be bijective, so it is not necessarily an isomorphism. Also, the definition of a structure allows an empty structure, with no relations.

Definition 11.2
Let R_1, R_2 be representations of S. Let $\cup(R_1, R_2) = \{(s, \{R_1(s)\} \cup \{R_2(s)\}) \mid s \in S\}$. Let $\cup(R_1, \ldots, R_n) = \{(s, \{R_1(s), \ldots, R_n(s)\}) \mid s \in S\}$ where $R_1, \ldots, R_n$ are representations of S.

The reader should note that $\cup(R_1, R_2, R_3)$ is neither defined as $\cup(R_1, \cup(R_2, R_3))$ nor as $\cup(\cup(R_1, R_2), R_3)$.

Definition 11.3
The representations of S are pairwise consistent if for all pairs of representations R_1, R_2 the map $\cup(R_1, R_2)$ is one to one. The representations of S are consistent if for all tuples of representations $R_1, \ldots, R_n$ the map $\cup(R_1, \ldots, R_n)$ is one to one.

Example 11.1
Let $S = \{s_1, s_2, s_3\}$ have empty structure, and consider the table below, which gives the value of each representation, and their product.

	ϕ	ϕ^{-1}	$\cup(\phi, \phi^{-1})$
s_1	s_2	s_3	$\{s_2, s_3\}$
s_2	s_3	s_1	$\{s_1, s_3\}$
s_3	s_1	s_2	$\{s_1, s_2\}$

In this case $\cup(\phi, \phi^{-1})$ is one to one. □

Example 11.2
Let $S = \{s_1, s_2, s_3\}$ have empty structure, and consider the three representations with interpretations $S, \{s_1, s_2, a\}, \{s_2, s_3, b\}$ below, and their products.

	id	ϕ_1	ϕ_2	$\cup(id, \phi_1, \phi_2)$	$\cup(id, \phi_1)$
s_1	s_1	s_2	s_2	$\{s_1, s_2\}$	$\{s_1, s_2\}$
s_2	s_2	s_1	s_3	$\{s_1, s_2, s_3\}$	$\{s_2, s_1\}$
s_3	s_3	a	b	$\{s_3, a, b\}$	$\{s_3, a\}$

$\cup(id, \phi_1, \phi_2)$ is a representation. However $\cup(id, \phi_1)$ is not a representation because it is not one to one. □

Example 11.3
$Z_3 = \{s_0, s_1, s_2\}$ together with binary operation

$$s_x + s_y = s_{(x+y) \bmod 3}$$

is a group called the cyclic group with 3 elements. Let $\phi : Z_3 \to Z_3$ be defined by $\phi(s_0) = s_0$, $\phi(s_1) = s_2$ and $\phi(s_2) = s_1$. We can verify that ϕ is structure preserving by considering the possible sums

$$\phi(s_0) + \phi(s_0) = s_0 + s_0 = s_0 = \phi(s_0) = \phi(s_0 + s_0)$$

$$\phi(s_0) + \phi(s_1) = s_0 + s_2 = s_2 = \phi(s_1) = \phi(s_0 + s_1)$$

$$\phi(s_0) + \phi(s_2) = s_0 + s_1 = s_1 = \phi(s_2) = \phi(s_0 + s_2)$$

$$\phi(s_1) + \phi(s_1) = s_2 + s_2 = s_1 = \phi(s_2) = \phi(s_1 + s_1)$$

$$\phi(s_1) + \phi(s_2) = s_2 + s_1 = s_0 = \phi(s_0) = \phi(s_1 + s_2)$$

likewise those with arguments reversed. Thus ϕ is a representation of Z_3. Consider the representations below, and their products.

	id	ϕ	$\cup$id, ϕ
s_0	s_0	s_0	$\{s_0\}$
s_1	s_1	s_2	$\{s_1, s_2\}$
s_2	s_2	s_1	$\{s_1, s_2\}$

Since $\cup(\mathrm{id}, \phi)$ is not one to one, the representations of Z_3 are inconsistent. □

The cyclic group Z_n is defined as $\{s_0, \ldots, s_{n-1}\}$ together with the binary operation $s_x + s_y = s_{(x+y) \bmod n}$.

Lemma 11.1
The representations of Z_n are inconsistent if $n > 2$.

Proof

Let $\phi : Z_n \to Z_n$, $\phi : s_m \mapsto s_{m(n-1) \bmod n}$. Suppose $n-1$ and n have a common factor $p > 1$. Then $n - 1 = pr$ and $n = ps$ for some integers r, s. Since $n = n - 1 + 1$, $ps = pr + 1$, so $p(r - s) = -1$. This would imply p and $r - s$ equal $+1$ or -1, contradicting $p > 1$. Thus $n - 1$ and n are relatively prime, so that the multiples of $\phi(s_1) = s_{n-1}$ generate Z_n. Thus ϕ is a structure preserving mapping, hence a representation of Z_n. Since $\phi(s_{n-1}) = s_1$, $\cup(\mathrm{id}, \phi)(s_1) = \{s_1, s_{n-1}\} = \cup(\mathrm{id}, \phi)(s_{n-1})$, so the representations of Z_n are inconsistent if $n > 2$. ∎

Example 11.4
Let S be a collection of image points on an NWES-grid (see chapter 3), and its structure is the topology of the NWES-grid. Let ψ be the map which links points of S with their successors. Let ψ^l denote the composition of ψ with itself l times, for any positive integer l. Choose a basepoint x in each connected component of S. Then for any $y \in S$ there is an $x \in S$ and integer m such that $y = \psi^m(x)$. Let

$$\phi_n(y) = \psi^{m(n-1) \bmod n}(x)$$

For each y, there is an $n \geq 4$ such that $\phi_n(\psi(x)) = \psi^{n-1}(x)$ and $\phi_n(\psi^{n-1}(x)) = \psi(x)$. This is because repeated composition of successors eventually returns to the starting point. Moreover, this is a structure preserving map, so the representations of S are

not consistent. In our model, this explains why visual awareness does not emerge from the retinal image itself, but requires more elaborate processing. □

Lemma 11.2
The operation $\cup(\ ,\ )$ is commutative, but is not associative, has no identity and no inverses. Hence we can write $\cup(\ ,\ldots,\ )$ with any number of arguments because the order of composition is irrelevant. If the representations of S are consistent then $\cup(\ ,\ )$ is a closed operation.

Proof

The operation $\cup(\ ,\ )$ is commutative because

$$\cup(R_1, R_2) = \{(s, \{R_1(s), R_2(s)\}) \mid s \in S\} = \cup(R_2, R_1)$$

It is not associative because

$$\cup(\cup(R_1, R_2), R_3) = \{(s, \{\{R_1(s), R_2(s)\}, R_3(s)\}) \mid s \in S\}$$

whearas

$$\cup(R_1, \cup(R_2, R_3)) = \{(s, \{R_1(s), \{R_2(s), R_3(s)\}\}) \mid s \in S\}$$

and in general

$$\{\{R_1(s), R_2(s)\}, R_3(s)\} \neq \{R_1(s), \{R_2(s), R_3(s)\}\}$$

If it had an identity then for some I, $\cup(I, R_1) = I$ for all R_1, so

$$\{I(s), R_1(s)\} = I(s)$$

which is impossible. Without an identity element, inverses would be meaningless. ∎

Until this point we have only found examples of sets S whose representations are not consistent. We will give a lemma that is useful to establish that the representations of a set are consistent. It makes use of group theoretic properties. An element g of a group is called an involution if g^2 is the identity element of the group, but g is not itself the identity element. A reflection is perhaps the simplest example of an involution.

Example 11.5
A rotation by π is an involution. The matrix

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

is an involution. The matrix

$$\begin{bmatrix} 5/13 & -12/13 \\ 12/13 & -5/13 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 5/13 & -12/13 \\ 12/13 & -5/13 \end{bmatrix}^{\top}$$

is also an involution. A diagonal matrix whose diagonal entries are ± 1, where at least one of these are negative is an involution. □

Symmetric matrices are never involutions unless all eigenvalues are ± 1 and at least one eigenvalue is -1. This is because the spectral decomposition $A = U\Lambda U^{\top}$ gives $A^2 =$

$U\Lambda^2 U^\top$, so that $A^2 = I$ implies $\Lambda^2 = I$. This in turn implies that all eigenvalues are ± 1, and unless one of these is negative, $\Lambda = I$ and $A = I$. Conversely, if all eigenvalues of a symmetric matrix A are ± 1 and one or more eigenvalues is negative, then A is an involution. Hermitian matrices have real eigenvalues, so likewise, Hermitian matrices are involutions if and only if all eigenvalues are ± 1 and at least one eigenvalue is negative.

Example 11.6
The matrix

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

is not an involution. It is skew-symmetric. □

Suppose A is a skew-symmetric matrix. Then the i-th column is the negative of the i-th row, so the diagonal entries of A^2 are minus the squared length of the rows of A. Hence the diagonal entries of A^2 are negative, so $A^2 \neq I$. Consequently a skew-symmetric matrix is never an involution.

Suppose A is a skew-Hermitian matrix. Then the i-th column is the conjugate of the i-th row, so the diagonal entries of A^2 are minus the squared length of the rows of A. Thus a skew-Hermitian matrix is never an involution.

Lemma 11.3
Let the automorphisms of S be denoted $Aut(S)$, and suppose they are a group under composition, which acts freely on S. If the representations of S are pairwise inconsistent then $Aut(S)$ has elements that are involutions.

Proof

Let $\phi_1, \ldots \phi_n$ be representations of S, and id the identity automorphism.

If the representations of S are inconsistent then for some representations $\phi_1, \ldots, \phi_n$ the map $\cup(\phi_1, \ldots, \phi_n)$ is not one to one. So there exist $s, t \in S$ such that $s \neq t$ and $\{\phi_1(s), \ldots, \phi_n(s)\} = \{\phi_1(t), \ldots, \phi_n(t)\}$. Since $\phi_1, \ldots, \phi_n$ are representations, they are one to one, so $\phi_i(s) \neq \phi_i(t)$ for $i = 1 \ldots n$. $\phi_1(s) = \phi_{i_1}(t)$ for $i_1 \neq 1$. Since $Aut(S)$ is a group, we have $\phi_{i_1}^{-1} \circ \phi_1(s) = t$. Likewise $\phi_{i_2}^{-1} \circ \phi_2(s) = t$ for some $i_2 \neq 2$. Hence $\phi_1^{-1}\phi_{i_1}\phi_{i_2}^{-1}\phi_2(s) = s$. Since $Aut(S)$ acts freely on S, $\phi_1^{-1}\phi_{i_1}\phi_{i_2}^{-1}\phi_2 = \mathrm{id}$.

Suppose $n = 2$. Then $i_1 = 2$ and $i_2 = 1$. Since $\cup(\phi_1, \phi_2)$ is not one to one and ϕ_1 and ϕ_2 are one to one, $\phi_1 \neq \phi_2$. From $\phi_1^{-1}\phi_2\phi_1^{-1}\phi_2 = \mathrm{id}$, it follows that $(\phi_1^{-1}\phi_2)^2 = \mathrm{id}$ and since $\phi_1 \neq \phi_2$ the group $Aut(S)$ has an involution.

∎

The above lemma was the author's original motivation for seeking free group actions, which led to the new formulation of the perspective camera model in chapter 2. We will now see that there is also a free automorphism group in the case of our model of features.

A set of vector fields on a manifold are called independent if the vectors are linearly independent at each point of the manifold. An ordered pair of n independent vector

fields on a manifold of dimension n is called a parallelization of that manifold. A parallelization of an open subset of the plane is analogous to a grid on that subset.

Lemma 11.4
The group of automorphisms of a parallelization of a connected Hausdorff manifold act freely.

Proof See proposition 13.6.1 in [26]. ∎

Because a parallelization is analogous to a grid on a manifold, an involution preserving the parallelization is analogous to a reflection, and is a kind of symmetry. The representations of a connected region with a parallelization are pairwise consistent if and only if there is no such symmetry with respect to that grid. It is possible that the extraction of symmetry by the visual system could be part of the construction of a representation that is consistent.

The hypothesis on awareness is that we cannot be aware of a region unless its representations are consistent.

The hypothesis on awareness suggested that manifolds with parallelizations would be an appropriate model of features. It suggested that what should be computed are points where integral curves intersect, in accordance with the interpretation of orientation dominance. It also suggested that the integral curves themselves would not be visible to humans, hence overcoming the initial reservation that this must be very different to how we see, because the world seems rather smooth and connected. This led to the model in chapter 3.

The hypothesis also suggested that the group representing how images change in response to varying camera parameters must be one that acts freely. This led to the new formulation of the perspective camera model in chapter 2. This in turn led to a method of scene reconstruction that closely resembles our own vision, in that feedback from ocular muscles could provide the missing information needed to produce a sensation of depth from two eyes. Feedback from the vestibular apparatus and the ocular muscles could provide the missing information needed to produce a sensation of depth from a moving eye. The correspondence problem was studied in detail using shape spaces, leading to convergence and stability conditions, as well as non-existence and existence proofs.

The author conjectures that if $Aut(S)$ is a group which acts freely on S and some product of representations $\cup(\phi_1, \ldots, \phi_n)$ is not one to one then there is an $x \in Aut(S)$ such that $x^m = id$ for some $1 < m \leq n$ but $x \neq id$. In other words, if $Aut(S)$ is a group which acts freely on S and the representations of S are inconsistent then $Aut(S)$ contains an element of finite order.

Chapter 12

Conclusions

The perspective camera model was justified from geometric optics; assuming a lens system consisting of multiple surfaces of revolution with a common axis, and a small aperture, under the paraxial assumption. Experimental evidence, together with our lemma 2.1, demonstrates that the model may hold more accurately for the eye, because the retinal waveguides point at the center of the iris.

In chapter 2 we arrived at a new formulation of the perspective camera model. The new formulation differs significantly from previous projective and affine formulations of the model, in that there is associated a group theoretical model which incorporates the constant and unchanging shape of the retina. The model of epipolar geometry originally discovered by photogrammetrists and rediscovered in computer vision suffers from certain pathologies. An intuitive explanation for these pathologies is that the retina is arbitrary, so that calculations can result in a state where an everywhere discontinuous retina is implicit.

In chapter 3 we showed that in the perspective camera model, there is a sufficient condition for occlusions, discontinuities in intensity, discontinuities in derivatives of intensity, discontinuities in scene surfaces or discontinuities in their derivatives. Taking to its logical extreme the assumption that a smooth image of a smooth scene surface is visible in its entirety within a bounded window, we deduced that the integral curves of any 2-vector valued smooth differential operator will be a one parameter family of simple closed curves disjoint from each other. Taking the contrapositive of this statement, gives the sufficient condition. The theory is based on minimal assumptions which are obviously justified. We explained how a new digital geometry leads to a successful implementation of an algorithm based on this theory. The algorithm needs no tunable parameters, does not modify the data to impose conditions that were not true (intensity is not smooth due to surface microstructure), responds to narrow regions like lines and wires, does not produce no response at corners and junctions, and does not produce highly fragmented edges. The algorithm produces no counter-intuitive responses. There are however some gaps where no response is produced. Although it is possible to bridge these gaps, this is not thought to be useful because no attributes would be so produced. There are two reasons why there are gaps: the condition is sufficient but not necessary; if the edge is tangential to integral curves,

no response can be expected. It is seen that images have a natural data structure associated with them, which can be generated in linear time and space. This data structure has previously been overlooked because it is not consciously perceived by human beings.

The feature detection algorithm leads to a mechanism of generating hypotheses of corresponding groups of points in chapter 8. The algorithm produces a hypothesis for each pair of groups in two images. Each hypothesis is an equal number of pairs of points, one from each image. The hypothesis is generated by calculating the maximal shift invariant substring of the longest common subsequences of two attribute sequences. Many sub-optimal hypotheses can be eliminated by utilising an upper bound on the length of a cyclic longest common subsequence of two strings. The fastest known algorithm has an expected time of $O(mn)$ and a worst case time complexity of $O(mn \log n)$, but this is known to be sub-optimal.

In order to test hypotheses of correspondence, we must avoid treating images of distinct scenes as if they were images of the same scene; avoid bias towards some camera configurations or scenes; and minimise instability in reconstruction. The problem of bias is due to the fact that inconsistent hypotheses must imply inconsistent camera parameters; if a hypothesis test is influenced by the unknown parameters, it is biased. If a hypothesis test is not continuous with respect to the Euclidean metric on scenes, it will be unstable. We formulated these properties into a formal definition of a correspondence hypothesis test. The properties of bias and stability are not related to invariance, but to the topological concept of a quotient space. A theoretical test for hypotheses of correspondence between two images is provided by the theory of epipolar geometry. A theoretical notion of projective reconstruction has also been developed in this framework. In chapter 4 we proved that no metric exists on the quotient space by the projective group of non-planar scenes. In chapters 4 and 5 we proved that no correspondence hypothesis test exists for non-planar scenes in the theory of epipolar geometry. Thus we have seen that the stereo correspondence problem is intractable within the framework of epipolar geometry.

A viewpoint invariant string can be assigned to an image of a closed polygon on an opaque surface. Aside from the issue of degeneracy, the string is a stable invariant. It is also unbiased. It therefore becomes the first hypothesis test: where it fails, we apply a collinearity test. If that fails, we reject the hypothesis. Occlusions cause changes in the string according to the rules of a grammar introduced in chapters 6 and 7. An algorithm to compute that two such strings are not derivable from each other in time $O(mn)$ was given in chapter 8. This can be extended to an algorithm to determine whether they are derivable also.

In chapters 4 and 5 we established three cases of the reformulated perspective camera model in which correspondence hypothesis tests exist. One case was planar scenes in a fronto-parallel plane, and an algorithm was obtained from the theory of Euclidean shape spaces in chapters 4 and 5. The second case was the affine camera model, in which an algorithm was obtained for general scenes in chapters 4 and 5. The third case is the case of all non-planar scenes. In this case we proved that a correspondence hypothesis test exists, but the algorithm is currently unknown.

In chapter 9 we undertook a theoretical study of non-metric and metric reconstruction from corresponding points using our perspective camera model. We gave non-iterative algorithms for both problems, based on kinetic depth algorithms. In theory, the rotation between two cameras and the principal points of the cameras determine a reconstruction up to scale. A non-metric reconstruction is a function of only eight parameters. This is less than the affine and projective reconstructions. However, no adequate correction algorithm is known to apply these results to real images, because no correspondence hypothesis test algorithm is currently known.

In chapter 10 we undertook a theoretical study of what we term the Euclidean subshape problem. We conjectured an algorithm of quadratic complexity to find a subset of a larger set of points whose Euclidean shape resembles that of a smaller set. The algorithm is based on Le's inequality, which in turn is based on a rather difficult theorem of Von Neumann.

In chapter 11 we reviewed some of the most influential experiments on biological vision. These discoveries led to the view that vision is a process that somehow constructs a representation of the scene from signals emanating from the retina (of both eyes). Our own visual systems would probably be of very little use to us if we could not be aware of what we see. We summarised some broad categories of experiments that studied the phenomenon of awareness in humans and animals. In chapter 11 we put forward the somewhat tentative hypothesis that visual awareness is a consistency condition on representations. This hypothesis seems to explain some of the phenomena of visual awareness. It also narrows down the range of mathematical structures that are possible as representations in the problems of feature detection and stereo.

Appendix A

Public domain implementations of algorithms

We list some of the readily available packages that have been used to implement algorithms described in this thesis. You may be able to find an ftp site near you that contains some of the packages listed below by using xarchie if it is installed, or by telnet to an archie server such as archie.au. Alternatively it may be possible to find it by searching thw world wide web using mosaic or netscape.

NAPACK

NAPACK is a public domain linear algebra package, written in fortran. It can be obtained by ftp at research.att.com, login as netlib, or from xnetlib if you have this installed.

pbmplus

pbmplus is a public domain package of routines mainly for converting files from one graphics format to another. Most of the graphics file formats in common use are inter-convertible using pbmplus. pbmplus can be obtained from the world wide web at http://www.acme.com/software/pbmplus/. The package libpnmrw contains C code to read and write portable bitmap, greymap and colormap files. libpnmrw can be found on the web in the same place as pbmplus.

VRWeb

VRWeb is a public domain browser for the Virtual Reality Modelling Language. It is also a netscape plug-in, and can be used to view three dimensional web pages (.wrl files) on the World Wide Web. The Virtual Reality Modelling Language (VRML) is a visualisation and graphics standard for the World Wide Web.

Example uses of VRWeb include rendering reconstructions, displaying numbered corresponding points, inspecting features at any desired resolution. Using anchors in VRML, it is also possible to interactively pick an object in a VRML file, and display a cross-reference to the object. This is very helpful when debugging programs with large data files.

See the VRWeb homepage at http://hyperg.iicm.tu-graz.ac.at/vrweb/ to find out more, or download VRWeb. A useful tutorial to VRML can be found at http://vrml.wired.com/3.html on the world wide web.

HVision

HVision is a package of routines for computer vision that was developed by Harvard (and MIT). It can be obtained by anonymous ftp to math.harvard.edu, login as anonymous, and is contained in the vision directory. Mark Nitzberg is one of the authors of HVision, his email is nitzberg@abel.math.harvard.edu (you need to obtain permission to use HVision).

MAS

MAS is a public domain symbolic algebra package, obtainable by anonymous ftp to alice.fmi.uni-passau.de, login as anonymous; version 7 is contained in the pub/ComputerAlgebra directory. The MAS package includes C source code, it is actually written in MODULA-2 and machine translated to C.

Image-Matching

The program Image-Matching is a sun4 executable for stereo matching. It includes algorithms to compute a corner feature, a correlation feature, and iterative estimator of the fundamental matrix. It can be obtained by anonymous ftp from krakatoa.inria.fr in directory /pub/tmp/zhang.

Khoros

Khoros is a sophisticated image processing library. It is obtainable by anonymous ftp to ftp.khoros.unm.edu, login as anonymous.

Appendix B

Source code and documentation

The files tabulated below can be obtained by anonymous ftp from ftp.cs.adelaide.edu.au in (specified subdirectories of) pub/stereo.

B.1 C programs

The following header files needed to compile the programs are in **polygon/include**

datadict.h	Data dictionary
pbmtopl1.h	Convert pbm to pl1 file
pl1topl2.h	Convert pl1 to pl2 file
polylib.h	Polygon string descriptor library
stereolib.h	Stereo library

The files libpnmrw.c and libpnmrw.h from the libpnmrw package (see appendix A) are required to compile many of the programs below. The fortran files basis.f, det.f, fact.f, pack.f, qr.f, sing.f and vert.f from NAPACK (see appendix A) are also required to compile many of the programs below. The fortran files should be compiled first.

The following C programs are in **polygon/src**

threshold	Convert pgm to pbm file
edge2.c	Generate family of integral curves of Hamiltonian (pl1 file)
edge2inl1.c	Include file for edge2.c
edge2inl2.c	Include file for edge2.c
edge2inl3.c	Include file for edge2.c
edge3.c	Convert pl1 file to (component order) pl2 file
edge5.c	Binary string descriptors & orientation from pl2 file
graddisc1.c	Detect occlusions, discontinuities (output VRML, pl2 files)
graddisc2.c	Detect features (output pl3, edge pl1 files)
findflat2.c	Brute force correspondence hypothesis table
findflat3.c	Correspondence hypothesis table using hypothesis elimination
dcmp2.c	Correspondence hypothesis for specified descriptors (VRML)
header.wrl	VRML header to append computer generated VRML to
pl2tovrml.c	Transform pl2 file to VRML (+HTML) for display, component picking
pl2tops.c	Convert pl2 file to postscript file for printing

catpl1.c	Convert binary pl1 file to text
catpl2.c	Convert binary pl2 file to text
txttopl1.c	Convert text pl1 to pl1 file
stereo8.c	Graph affine coplanarity statistic on random 2D scenes
stereo9.c	Graph affine coplanarity statistic on random 3D scenes
testrand.c	Histogram of gaussian random number generator
bcode.c	Convert binary string descriptor to number, length form
scode.c	Convert number, length binary string descriptor to string form
normalizeb.c	Normalize binary string descriptor (number,length form)
insrule.c	Apply insertion production to a binary string descriptor
rule5.c	Apply $xx \rightarrow \bar{x}x\bar{x}$ production to binary string descriptor
validb.c	Validate binary string descriptor
bcdsign.c	Print all binary coded string descriptors of length n
ngons.c	Print number of equivalence classes of n-gons

Manual pages for some of these programs can be found in **polygon/man/man1**. All other source files begin with comments advising the user of how to compile and use the programs. Manual pages for the pl1, pl2 and pl3 files can be found in **polygon/man/man5**. The files have been successfully compiled on SUN 10, DEC Alpha, DEC MIPS, SGI Indy and SGI Power Challenge machines. If compilation errors occur, please check all C library include files. Different systems put their header files in different directories. The names of header files to include particular functions can also differ. Some systems use IEEE arithmetic and generate NaN (not a number) values. If this is the case, the nan.h header should be included, otherwise nan.h should not be included. Memory allocation calls can produce run time errors due to system dependent behaviour of malloc, calloc and free.

The following shell scripts can be found in **polygon/bin**

pgmdisc	Detect occlusions, discontinuities (as VRML) from pgm file
pgmdisc2	Generate features (as pl3, edge pl1) from pgm file

B.2 Maple programs

Maple is a symbolic algebra package which was used to perform simulations in this thesis. The files described below can be found in the **polygon/maple** subdirectory.

Two libraries of Maple functions have been written for two different releases of Maple. The library shape2 has been written for release 2, and shape3 has been written for release 3. The appropriate library should be read into Maple using the Maple command

```
read 'shape3';
```

The library functions can then be used interactively. The file called guide in the help directory is a guide to interactive use.

A number of maple scripts that perform specific experiments are described below. This is a small subset of the Maple scripts written to date.

exper19	Compute d by Sparr's algorithm on two random views of 8 3D points
exper31	Compute d by Sparr's algorithm on two random views of 8 2D points
exper32	Test Sparr's algorithm on a cube
exper38	Compute reconstruction from 2 views of 5 points, 4 coplanar, rotation, principal p
exper40	Compute E from two views of 5 points, 4 coplanar, rotation, principal points
exper42	Self-calibrate from two views of 8 3D points, using symbolic solver
exper45	Compute d from 2 views of six points with four coplanar
exper46	Compute closed form solution to stereo equations
exper51	Test for coincident optical centers

Bibliography

1. E.D. Adrian, *The basis of sensation*, Christophers, 1928.

2. J.Y. Aloimonos, *Perspective approximations*, Image and Vision Computing **8** (1990), 179–192.

3. N. Ansari and E.J. Delp, *On detecting dominant points*, Pattern Recognition **24** (1991), no. 5, 441–451.

4. K. Astrom, *Fundamental limitations on projective invariants of planar curves*, IEEE Transactions on Pattern Analysis and Machine Intelligence **17** (1995), no. 1, 77–81.

5. N. Ayache, , Artificial intelligence, MIT Press.

6. J. Babaud, A.P. Witkin, M. Baudin, and R.O. Duda, *Uniqueness of the gaussian kernel for scale space filtering*, IEEE Transactions on Pattern Analysis and Machine Intelligence **8** (1986), no. 1, 26–33.

7. H.B. Barlow, *Summation and inhibition in the frog's retina*, Journal of Physiology (London) **119** (1953), 69–88.

8. P. Beardsley, D. Sinclair, and A. Zisserman, *Ego-motion from six points*, Insight meeting, Catholic University Leuven.

9. T. Becker and V. Weispfenning, *Grobner bases: a computational approach to commutative algebra*, Springer-Verlag, 1993.

10. C.A. Berenstein and E.V. Patrick, *Exact deconvolution for multiple convolution operators: an overview, plus performance characterization for imaging sensors*, Proceedings of the IEEE **78** (1990), no. 4, 723–734.

11. B. Bhavnagri, *A method for representing shape based on an equivalence relation on polygons*, Pattern Recognition **27** (1994), no. 2, 247–260.

12. ———, *Models for recognition and correspondence matching of objects*, Proceedings of the 7th australian joint conference on artificial intelligence (C. Zhang, J. Debenham, and D. Lukose, eds.), World Scientific, pp. 536–543.

13. ———, *Object recognition, geometric invariance and shape spaces*, Tech. Report TR94-08, University of Adelaide, Department of Computer Science, South Australia 5005, May 1994.

14. _____, *Connected components of the space of simple closed non-degenerate polygons*, Current Issues in Statistical Shape Analysis, pp. 187–188.

15. _____, *Connected components of the space of simple non-degenerate polygons*, Tech. Report TR95-07, University of Adelaide Computer Science Department, 1995.

16. _____, *Construction of a markov process on* $\cup_k \Sigma_2^k$ *to model a process arising in vision*, Proceedings of Current Issues in Statistical Shape Analysis.

17. _____, *Proofs and corrections to construction of markov process on shape spaces*, Tech. Report TR95-06, University of Adelaide Computer science department, 1995.

18. _____, *Proofs for construction of markov process on shape spaces*, Tech. Report TR95-04, University of Adelaide Computer Science Department, 1995.

19. _____, *The stereo correspondence problem*, August 1995.

20. K.G. Binmore, *Mathematical analysis: a straightforward approach*, Cambridge University Press, 1982.

21. J.L. Blue, G.T. Candela, P.J. Grother, R. Chellappa, and C.L. Wilson, *Evaluation of pattern classifiers for fingerprint and ocr applications*, Pattern Recognition **27** (1994), no. 4, 485–501.

22. W.M. Boothby, *An introduction to differentiable manifolds and riemannian geometry*, Academic Press, 1975.

23. M. Born and E. Wolf, *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*, Pergamon Press, 1969.

24. E. Bribiesca, *Arithmetic operations among shapes using shape numbers*, Pattern Recognition **13** (1981), 123–138.

25. E. Bribiesca and A. Guzman, *How to describe pure form and how to measure differences in shape using shape numbers*, Pattern Recognition **12** (1980), 101–112.

26. F. Brickell and R.S. Clark, *Differentiable manifolds: an introduction*, Van Nostrand Reinhold, 1970.

27. M.J. Brooks, L. de Agapito, D.Q. Huynh, and L. Baumela, *Direct methods of self-calibration of a moving stereo head*, European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 1065, Springer-Verlag, pp. 415–426.

28. D.V. Bugg, *Electronics: circuits, amplifiers and gates*, Adam-Hilger, 1991.

29. J.B. Burns, R.S. Weiss, and E.M. Riseman, *The non-existence of general case view invariants*, Geometric invariance in computer vision, MIT Press, 1992, pp. 120–131.

30. J.F. Canny, *A computational approach to edge detection*, IEEE Pattern Analysis Machine Intelligence **8** (1986), no. 6, 679–698.

31. S. Carlsson, *The double algebra: an effective tool for computing invariants in computer vision*, Applications of invariance in computer vision, Lecture Notes in Computer Science, vol. 825, Springer-Verlag, pp. 145–164.

32. T.K. Carne, *The geometry of shape spaces*, Proceedings of the London Mathematical Society **61** (1990), no. 3, 407–432.

33. W.N. Charman, Optics of the human eye, Vision and Visual dysfunction, vol. 1, ch. 1, pp. 1–20, Vision and Visual dysfunction, Macmillan press, 1991, pp. 1–20.

34. K.L. Chung, *A course in probability theory*, Academic Press, 1974.

35. _____, *Lectures from markov processes to brownian motion*, Springer-Verlag, 1982.

36. R. Cipolla, *Active visual inference of surface shape*, Lecture notes in computer science, vol. 1016, Springer-Verlag.

37. L. Conlon, *Differentiable manifolds, a first course*, Birkhauser, Boston, 1993.

38. T.H. Cormen, C.E. Leiserson, and R.L. Rivest, *Introduction to algorithms*, MIT Press, 1990.

39. F.H.C. Crick, *The astonishing hypothesis: the scientific search for the soul*, Charles Sribner's Sons Macmillan Publishing, 1994.

40. G. Csurka and O.D. Faugeras, *Computing three dimensional projective invariants from a pair of images using the grassman-cayley algebra*, Proceedings Europe-China Workshop on geometrical modelling and invariants for computer vision.

41. M.L. Curtis, *Matrix groups*, Springer, New York, 1984.

42. K. Deguchi and S. Aoki, *Regularized polygonal approximation for analysis and interpretation of planar contour figures*, Proceedings of the International Conference on Pattern Recognition, vol. 1, IEEE, pp. 865–869.

43. S. Demey, A. Zisserman, and P. Beardsley, *Affine and projective structure from motion*, Proceedings British Machine Vision Conference.

44. R. Deriche and G. Giraudon, *Accurate corner detection: an analytical study*, 3rd International Conference on Computer Vision, pp. 66–70.

45. R.O. Duda and P.E. Hart, *Pattern classification and scene analysis*, Wiley, New York, 1973.

46. M.L. Eaton, , Regional conference series in probability and statistics, Institute of Mathematical Statistics.

47. H. Edelsbrunner, *Algorithms in combinatorial geometry*, Springer-Verlag, 1987.

48. D. Eisenbud, Commutative algebra with a view toward algebraic geometry, ch. 15, pp. 317–381, Springer-Verlag, 1995, pp. 317–381.

49. C. Enroth-Cugell and J.G. Robson, *The contrast sensitivity of retinal ganglion cells of the cat*, Journal of Physiology (London) **187** (1966), 517–552.

50. Eppstein,
Design and analysis of algorithms lecture notes, http://www.ics.uci.edu/ eppstein/161, February 1996.

51. T.I. Fan, *Optimal matching of deformed patterns with positional influence*, Information Science **41** (1987), 259–280.

52. J.Q. Fang and T.S. Huang, *Some experiments on estimating the 3-d motion parameters of a rigid body from two consecutive image frames*, IEEE Transactions on Pattern Analysis and Machine Intelligence **6** (1984), 545–554.

53. O. Faugeras, *What can be seen in three dimensions with an uncalibrated stereo rig ?*, European Conference on Computer Vision, pp. 563–578.

54. _____, *Three dimensional computer vision: a geometric viewpoint*, MIT Press, 1993.

55. O.D. Faugeras and B. Mourrain, *On the geometry and algebra of the point and line correspondences between n images*, Proceedings 5th International Conference on Computer Vision, pp. 951–956.

56. V.A. Fischetti, G.M. Landau, P.H. Sellers, and J.P. Schmidt, *Identifying periodic occurrences of a template with applications to protein structure*, Information processing letters **45** (1993), 11–18.

57. D.A. Flanders, *Angles between flat subspaces of a real n-dimensional euclidean space*, Studies and essays presented to R. Courant on his 60th birthday, Interscience, 1948, pp. 129–138.

58. H. Freeman, *Computer processing of line drawings*, Computer Surveys **6** (1974), 57–97.

59. S. Gallot, D. Hulin, and J. Lafontaine, *Riemannian geometry*, Springer-Verlag, 1990.

60. F. Ghorbel, *A complete invariant description for grey-level images by the harmonic analysis approach*, Pattern recognition letters **15** (1994), 1043–1051.

61. P.K. Ghosh, *An algebra of polygons through the notion of negative shapes*, CVGIP Image Understanding **64** (1991), no. 1, 119–144.

62. J.J. Gibson, *The senses considered as perceptual systems*, Houghton Mifflin, 1966.

63. R.C. Gonzalez and P. Wintz, *Digital image processing*, Addison-Wesley, 1987.

64. C. Goodall, *Procrustes methods in the statistical analysis of shape*, Journal of the Royal Statistical Society Series B **53** (1991), 285–339.

65. ———, *Procrustes methods in the statistical analysis of shape revisited*, Current Issues in Statistical Shape Analysis (K.V. Mardia and C. Gill, eds.), Leeds University Press, pp. 18–33.

66. J. Gregor and M.G. Thomason, *Dynamic programming alignment of sequences representing cyclic patterns*, IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (1993), no. 2, 129–135.

67. J. Gregor and M.J. Thomason, *Efficient dynamic programming alignment of cyclic strings by shift elimination*, Pattern Recognition **29** (1996), no. 7, 1179–1185.

68. U. Grenander and D.M. Keenan, *On the shape of plane images*, SIAM Journal of Applied Mathematics **53** (1993), no. 4, 1072–1094.

69. W.W. Hager, *Applied numerical linear algebra*, Prentice-Hall, 1988.

70. R.M. Haralick and L.G. Shapiro, *Computer and robot vision*, vol. I, Addison Wesley, 1992.

71. ———, *Computer and robot vision*, vol. II, Addison Wesley, 1992.

72. C. Harris, *Determination of ego-motion from matched points*, Proceedings 3rd Alvey Vision Conference, pp. 189–192.

73. C. Harris and M. Stephens, *A combined corner and edge detector*, Proceedings 4th Alvey Vision Conference.

74. J. Harris, *Algebraic geometry: a first course*, Springer-Verlag, 1992.

75. R. Hartley, R. Gupta, and T. Chang, *Stereo from uncalibrated cameras*, IEEE Conference on Computer Vision and Pattern Recognition, pp. 761–764.

76. R.I. Hartley, *Projective reconstruction and invariants from multiple images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **16** (1994), no. 10, 1036–1041.

77. ———, *In defence of the 8-point algorithm*, Proceedings 5th International Conference on Computer Vision, pp. 1064–1070.

78. ———, *In defence of the 8-point algorithm*, Full version of ICCV95 paper, 1995.

79. H.K. Hartline, *The response of single optic nerve fibers of the vertebrate eye to illumination of the retina*, Am. J. Physiol. **121** (1938), 400–415.

80. ———, *The receptive fields of optic nerve fibers*, Am. J. Physiol. **130** (1940), 690–699.

81. D. Hearn and M.P. Baker, *Computer graphics*, Prentice-Hall, 1986.

82. A. Heyden, *On the consistency of line-drawings, obtained by projections of piecewise planar objects*, Dept of Mathematics, Lund Institute of Technology, Sweden, 1993.

83. ———, *Geometry and algebra of multiple projective transformations*, Ph.D. thesis, Lund University, 1995.

84. E.C. Hildreth and D.C. Marr, *Theory of edge detection*, Proceedings Royal Society of London B (1980), 187–217.

85. D.S. Hirschberg, *A linear space algorithm for computing maximal common subsequences*, Communications of the ACM **18** (1975), no. 6, 341–343.

86. W.C. Hoffman, *The lie algebra of visual perception*, Journal of Mathematical Psychology **3** (1966), no. 1, 65–98.

87. B.K.P. Horn, , MIT electrical engineering and computer science series, McGraw-Hill.

88. ———, *Relative orientation*, International Journal of Computer Vision 4 (1990), 59–78.

89. B.K.P. Horn and M.J. Brooks, eds., *Shape from shading*, MIT Press, 1989.

90. B.K.P. Horn and R.W. Sjoberg, *Calculating the reflectance map*, Shape from shading (B.K.P. Horn and M.J. Brooks, eds.), MIT Press, 1989, pp. 215–244.

91. D.H. Hubel and T.N. Wiesel, *Integrative action in the cats lateral geniculate nucleus*, J. Physiol. (London) **155** (1961), 385–398.

92. ———, *Receptive fields, binocular interaction and functional architecture in the cat's visual cortex*, J. Physiol. (London) **166** (1962), 106–154.

93. ———, *Receptive fields and functional architecture of monkey striate cortex*, Journal of Physiology (London) **195** (1968), 215–243.

94. ———, *Cells sensitive to binocular depth in area 18 of the macaque monkey cortex*, Nature **225** (1970), 41–42.

95. H.Wang and M. Brady, *Real-time corner detection algorithm for motion estimation*, Image and Vision Computing **13** (1995), no. 9, 695–703.

96. K. Kanatani, , Oxford engineering science series, Oxford University Press.

97. ———, *Statistical optimization for geometric computation: theory and practise*, Gunma University, Japan, 1995.

98. G. Kanizsa, *Organization in vision, essays on gestalt perception*, Praeger, 1979.

99. D.G. Kendall, *Shape manifolds, procrustean metrics and complex projective spaces*, Bull. London Math. Soc. **16** (1984), 81–121.

100. _____, *Exact distributions for shapes of random triangles in convex sets*, Advances in Applied Probability **17** (1985), 308–329.

101. L. Kitchen and A. Rosenfeld, *A gray level corner detection*, Pattern Recognition Letters **1** (1982), 95–102.

102. J.J. Koenderink and A.J. van Doorn, *Affine structure from motion*, Optical Society of America A **8** (1991), no. 2, 377–385.

103. P. Kovesi, *Image features from phase congruency*, Tech. Report 95/4, University of Western Australia, Department of Computer Science, June 1995.

104. S.W. Kuffler, *Discharge patterns and functional organization of mammalian retina*, Journal of Neurophysiology **16** (1953), 37–68.

105. D. Lazard, *Systems of algebraic equations (algorithms and complexity)*, Computational algebraic geometry and commutative algebra (D. Eisenbud and L. Robbiano, eds.), Symposia mathematica, vol. XXXIV, Cambridge University Press, pp. 84–105.

106. H. Le, *On geodesics in euclidean shape spaces*, Journal of the London Mathematical Society **44** (1991), 360–372.

107. _____, *A stochastic calculus approach to the shape distribution induced by a complex normal model*, Mathematical Proceedings of the Cambridge Philosophical Society **109** (1991), 221–228.

108. H. Le and B. Bhavnagri, *On simplifying shapes by subjecting them to collinearity constraints*, Mathematical Proceedings of the Cambridge Philosophical Society **121** (1997), no. 2, To Appear.

109. H. Le and D.G. Kendall, *The riemannian structure of euclidean shape spaces: a novel environment for statistics*, Annals of statistics **21** (1993), no. 3, 1225–1271.

110. R. Lenz and P. Meer, *Experimental investigation of projection and permutation invariants*, Pattern Recognition Letters **15** (1994), 751–760.

111. _____, *Point configuration invariants under simultaneous projective and permutation transformations*, Pattern Recognition **27** (1994), no. 11, 1523–1532.

112. Z. Liu, J.P. Gaska, L.D. Jacobson, and D.A. Pollen, *Interneuronal interaction between members of quadrature phase and anti-phase pairs in the cat's visual cortex*, Vision Research **32** (1992), no. 7, 1193–1198.

113. H.C. Longuet-Higgins, *A computer algorithm for reconstructing a scene from two projections*, Nature **293** (1981), 133–135.

114. _____, *Multiple interpretations of a pair of images of a surface*, Proceedings Royal Society of London A **418** (1988), 1–15.

115. G. Lorig, *Advanced image synthesis - shading*, Advances in computer graphics I, Eurographic seminars, vol. XII, Springer-Verlag, pp. 441–456.

116. Q.T. Luong and O.D. Faugeras, *A stability analysis of the fundamental matrix*, European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 800, Springer-Verlag, pp. 577–588.

117. M. Maes, *On a cyclic string-to-string correction problem*, Information processing letters **35** (1990), 73–78.

118. _____, *Polygonal shape recognition using string matching techniques*, Pattern Recognition **24** (1991), 433–440.

119. K.V. Mardia and I.L. Dryden, *Shape distributions for landmark data*, Advances in Applied Probability **21** (1989), 742–755.

120. D. Marr, *Vision*, W.H. Freeman, 1982.

121. S.J. Maybank, , Springer series in information sciences, Springer-Verlag.

122. _____, *The angular velocity associated with the optical flow field arising from motion through a rigid environment*, Proc. Roy. Soc. Lond. A **401** (1985), 317–326.

123. _____, *Properties of essential matrices*, International journal of imaging systems and technology **2** (1990), 380–384.

124. _____, *Theory of reconstruction from image motion*, Springer-Verlag, 1992.

125. _____, *Application of the twisted cubic to model based vision*, Proceedings Europe-China Workshop on geometrical modelling and invariants for computer vision, pp. 158–165.

126. _____, *Relation between 3d invariants and 2d invariants*, Proceedings IEEE Workshop on representation of visual scenes, pp. 53–57.

127. R. Mohr, *Projective geometry and computer vision*, Handbook of pattern recognition and computer vision (Chen, Pau, and Wang, eds.), 1992.

128. F. Mokhtarian and A. Mackworth, *A theory of multiscale, curvature based representation for planar curves*, IEEE Transactions on Pattern Analysis and Machine Intelligence **14** (1992), no. 8, 789–805.

129. H.P. Moravec, *Towards automatic visual obstacle avoidance*, Proceedings International Joint Conference on Artificial Intelligence, p. 584.

130. D. Mumford and J. Fogarty, *Geometric invariant theory*, Ergebnisse der Mathematik und ihrer Grenzgebiete, vol. 34, Springer-Verlag, 2nd ed., 1982.

131. D. Mumford and J. Shah, *Optimal approximations by piecewise smooth functions and associated variational problems*, Communications on pure and applied mathematics **XLII** (1989), no. 5, 577–684.

132. J.L. Mundy and A. Zisserman, eds., *Geometric invariance in computer vision*, MIT Press, 1992.

133. J.R. Munkres, *Topology: A first course*, Prentice-Hall, Englewood Cliffs New Jersey, 1975.

134. D.W. Murray and B.F. Buxton, , Artificial Intelligence, MIT Press.

135. J.D. Murray and W. VanRyper, *Encyclopedia of graphics file formats*, O'Reilly, 1994.

136. H.H. Nagel, *Constraints for the estimation of displacement vector fields from image sequences*, Proceedings Internation Joint Conference on Artificial Intelligence, pp. 945–951.

137. L. Nielsen and G. Sparr, *Projective area-invariants as an extension of the cross-ratio*, CVGIP Image Understanding **54** (1991), no. 1, 145–159.

138. N.J. Nilsson, *Principles of artificial intelligence*, Morgan Kaufmann, 1980.

139. M. Nitzberg and D. Mumford, *The 2.1-d sketch*, 3rd International Conference on Computer Vision, pp. 138–144.

140. M. Nitzberg, D. Mumford, and T. Shiota, *Filtering, segmentation and depth*, Lecture Notes in Computer Science, vol. 662, Springer-Verlag, New York, 1993.

141. H. Osborn, *Vector bundles: Foundations and stiefel whitney classes*, vol. 1, Academic Press, 1982.

142. R. Owens, *Lecture notes in computer vision*, http://www.cs.uwa.edu.au/ robyn/Visioncourse/vision.html, February 1997.

143. R.A. Owens, S. Venkatesh, and J. Ross, *Edge detection is a projection*, Pattern Recognition Letters **9** (1989), 223–244.

144. B.N. Parlett, *The symmetric eigenvalue problem*, Prentice-Hall, 1980.

145. T. Pavlidis and S.L. Horowitz, *Segmentation of plane curves*, IEEE Transactions on computers **C-23** (1974), no. 8, 860–870.

146. R. Penrose, *The emperors new mind: concerning computers, minds and the laws of physics*, Oxford University Press, 1989.

147. ———, *Shadows of the mind: a search for the missing science of consciousness*, Oxford University Press, 1994.

148. V.J. Rayward-Smith, *A first course in formal language theory*, Mc Graw-Hill, 1995.

149. F.J. Rohlf and D. Slice, *Extensions of the procrustes method for the optimal superimposition of landmarks*, Systematic Zoology **39** (1990), no. 1, 40–59.

150. C. Ronse, *On idempotence and related requirements in edge detection*, IEEE PAMI **15** (1993), no. 5, 484–491.

151. A. Rosenfeld, *Connectivity in digital pictures*, Journal of the association for computing machinery **17** (1970), 146–160.

152. C. Rothwell, *Object recognition through invariant indexing*, Oxford University Press.

153. C. Rothwell, G. Curka, and O. Faugeras, *A comparison of projective reconstruction methods for pairs of views*, Proceedings 5th International Conference on Computer Vision, pp. 932–937.

154. C. Rothwell, O. Faugeras, and G. Curka, *Different paths towards projective reconstruction*, Proceedings Europe-China Workshop on geometrical modelling and invariants for computer vision, pp. 245–252.

155. J.G. Semple and G.T. Kneebone, *Algebraic projective geometry*, 1963.

156. S.A. Shafer, *Shadows and silhouettes in computer vision*, Kluwer Academic Publishers, 1985.

157. L.S. Shapiro, *Affine analysis of image sequences*, Cambridge University Press, 1995.

158. A. Shashua, *Algebraic functions for recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence **17** (1995), no. 8, 779–789.

159. A. Shashua and M. Werman, *The study of 3d-from-2d using elimination*, International Conference on Computer Vision, pp. 473–479.

160. ———, *Trilinearity of three perspective views and its associated tensor*, International Conference on Computer Vision, pp. 920–925.

161. G. Sparr, *Depth-computations from polyhedral images, or: why is my computer so distressed about the penrose triangle ?*, Tech. Report LUTFD2/TFMA-91/7004, Lund Institute of Technology, Dept of Mathematics, December 1991.

162. ———, *Projective invariants for affine shapes of point configurations*, Workshop on Invariants in Vision, Reykjavik (A. Zisserman and J. Mundy, eds.).

163. ———, *An algebraic/analytic method for reconstruction from image correspondences*, Tech. Report LUTFD2/TFMA-91/7003, Lund Institute of Technology, Dept of Mathematics, December 1992.

164. ———, *Depth computations from polyhedral images*, Image and Vision Computing **10** (1992), 683–688.

165. ———, *On the reconstruction of impossible objects*, Proceedings Swedish Society for automated image analysis.

166. ———, *A common framework for kinetic depth, reconstruction and motion for deformable objects*, European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 801, Springer-Verlag, pp. 471–482.

167. P.F. Stiller, C.A. Asmuth, and C.S. Wan, *A general theory of single view recognition - the affine case - with applications to indexing image databases for content based retrieval*, Personal Communication, 1995.

168. J. Stillwell, *Geometry of surfaces*, Springer-Verlag, 1992.

169. J. Stoer and R. Bulirsch, *Introduction to numerical analysis*, Springer-Verlag, 1993.

170. G. Strang, *Linear algebra and its applications*, Harcourt Brace Jovanovich, 1988.

171. K. Sugihara, *Machine interpretation of line drawings*, MIT Press, 1986.

172. B. Triggs, *Matching constraints and the joint image*, Proceedings 5th International Conference on Computer Vision, pp. 338–343.

173. W.H. Tsai and S.S. Yu, *Attributed string matching with merging for shape recognition*, IEEE PAMI **7** (1985), 453–462.

174. R. von der Heydt and E. Peterhans, *Mechanisms of contour perception in monkey visual cortex. i. lines of pattern discontinuity*, Journal of neuroscience **9** (1989), no. 5, 1731–1748.

175. ———, *Mechanisms of contour perception in monkey visual cortex. ii. contours bridging gaps*, Journal of Neuroscience **9** (1989), no. 5, 1749–1763.

176. J. von Neumann, Some matrix-inequalities and metrization of matrix-space, John von Neumann collected works, vol. 4, ch. 20, pp. 205–219, John von Neumann collected works, Pergammon Press, 1962, pp. 205–219.

177. J. Wagemans, P.V. Bossche, N. Segers, and G. D'Ydewalle, *An affine group model and the perception of orthographically projected planar random polygons*, Journal of mathematical psychology **38** (1994), 59–72.

178. R.A Wagner and M.J. Fischer, *The string-to-string correction problem*, Journal of the ACM **21** (1974), no. 1, 168–173.

179. F. W. Warner, *Foundations of differential manifolds and lie groups*, Springer-Verlag, New York, 1983.

180. L. Weiskrantz, E.K. Warrington, M.D. Sanders, and J. Marshall, *Visual capacity in the hemianopic field following a restricted occip ital ablation*, Brain **97** (1974), 709–728.

181. M. Werman and D. Weinshall, *Similarity and affine invariant distances between 2d point sets*, IEEE transactions on pattern analysis and machine intelligence **17** (1995), no. 8, 810–814.

182. M. Wertheimer, *Experimentelle studien uber das sehen von bewegung*, Zeitschrift f. Psychol. **61** (1912), 161–265.

183. _____, *Laws of organization in perceptual forms*, Harcourtm Brace and Co, 1938.

184. R.G. White and R.A. Schowengerdt, *Effect of point-spread functions on precision edge measurements*, Optical Society of America A **11** (1994), no. 10, 2593–2603.

185. J.H.C. Whitehead, *Manifolds with transverse fields in euclidean space*, Annals of Mathematics **73** (1961), 154–212.

186. G. Winkler, *Image analysis, random fields and dynamic monte carlo methods*, Applications of Mathematics, vol. 27, Springer-Verlag, 1995.

187. P.R. Wolf, *Elements of photogrammetry*, McGraw-Hill, 1974.

188. Y.C. Wong, *Differential geometry of grassman manifolds*, Proceedings of the National Academy of Sciences of the USA **57** (1967), 589–594.

189. G. Xu and Z. Zhang, *Epipolar geometry in stereo, motion and object recognition*, Kluwer Academic Publishers, 1996.

190. L.P. Yaroslavsky and H.J. Caulfield, *Deconvolution of multiple images of the same object*, Applied Optics **33** (1994), no. 11, 2157–2162.

191. A.L. Yuille and T.A. Poggio, *Scaling theorems for zero crossings*, IEEE Transactions on Pattern Analysis and Machine Intelligence **8** (1986), no. 1, 15–25.

192. S. Zeki, *A vision of the brain*, Blackwell Scientific Publications, 1993.

193. Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong, *A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry*, Artificial Intelligence (1995), To Appear.

194. A. Zisserman and S. Maybank, A case against epipolar geometry, Lecture Notes in Computer Science, vol. 825, 69–88, Lecture Notes in Computer Science, Springer Verlag, 1994, pp. 69–88.

195. Z.Zhang and O.D. Faugeras, *3d dynamic scene analysis*, Springer-Verlag, 1992.

Index