Symbolic and Numerical Computation for Artificial Intelligence

edited by

Bruce Randall Donald
Department of Computer Science
Cornell University, USA

Deepak Kapur
Department of Computer Science
State University of New York, USA

Joseph L. Mundy
AI Laboratory
GE Corporate R&D, Schenectady, USA

Academic Press
Harcourt Brace Jovanovich, Publishers
London San Diego New York
Boston Sydney Tokyo Toronto

ACADEMIC PRESS LIMITED
24–28 Oval Road
London NW1

US edition published by
ACADEMIC PRESS INC.
San Diego, CA 92101

Copyright © 1992 by
ACADEMIC PRESS LIMITED

This book is printed on acid-free paper

All Rights Reserved
No part of this book may be reproduced in any form, by photostat, microfilm or any other means, without written permission from the publishers

A catalogue record for this book is available from the British Library

ISBN 0-12-220535-9

Printed and Bound in Great Britain by
The University Press, Cambridge

Chapter 7

Applications of Invariant Theory in Computer Vision

David Forsyth

Department of Computer Science

University of Iowa, Iowa, IA 52242

Joseph L. Mundy

GE Corporate Research and Development

Schenectady, NY 12345

Andrew Zisserman

Charles Rothwell

Department of Engineering Science

Oxford University, Oxford, UK

Invariant theory yields a range of indexing functions, which can be used to identify plane objects in image data quickly and efficiently, without reference to their pose or to camera calibration. These indexing functions have found application in a fast model-based vision system that works well with a relatively large model base of 33 objects. This system is discussed in detail.

A way in which one might compute indexing functions from the outline of a curved, three dimensional object is then described. This approach uses extensive symbolic computations to construct a system of equations in object parameters which can easily be solved. This technique has been implemented, and is demonstrated on images of simple real scenes.

1. Introduction

Model-based vision systems search for instances of object models in images, typically using some form of shape information as the only clue to object identity. This process is complicated, because the outline observed in an image depends very strongly on the position of the object relative to the camera, and on the camera parameters. A widespread approach to recognition hypothesizes matches between aggregates of image features, and similar aggregates of model features. Given an hypothesized match, the position and orientation of the model can be computed. This information makes it possible to predict

the appearance of the object outline, and the resulting predictions can be checked against image data, to accept or discard the match.

Versions of this approach have been successfully demonstrated by a number of authors, e.g. Huttenlocher and Ullman (1987), Mundy and Heller (1990) and Thompson and Mundy (1987). Ponce's ground-breaking algorithm for recognizing curved surfaces (in this volume, and (Ponce and Kriegman, 1989) has similarities to this approach. The algorithm hypothesizes a model instance, solves for the position and orientation (pose) of the object most consistent with that hypothesis, and verifies the instance using this information.

This form of hypothesize and test has unattractive features as a recognition algorithm. The primary disadvantage is that each model in the model base must be used, in turn, to generate hypotheses. It is, therefore, a difficult algorithm to use when the model base contains parameterized systems of models, rather than a discrete collection of models. For example, it is hard to use this approach to recover the ratio of major radius to minor radius in an image of a torus without generating large optimization problems (like those of Nguyen *et al.*, 1991). Furthermore, the complexity of this approach is linear in the number of models; although this does not appear to be an insurmountable barrier, in practice it has lead to systems that have small model bases (two models is currently viewed as a respectable number).

One way of overcoming this problem is to compute descriptors from image outline information that immediately identify the object. We call such descriptors *indexing functions*, after Wayner (1991). These descriptors in effect provide an index into a model library. Clearly, an indexing function must not produce different values for different views of the same object, so that classical invariant theory can be used as a rich source of such functions. As a number of reviews of the aspects of invariant theory relevant in vision have appeared (e.g. Forsyth *et al.*, 1991a; Mundy and Zisserman, 1991), we do not review this area in detail here, but in section 2 provide some examples of relevant invariants that have been applied.

We then describe a working model-based vision system that uses indexing functions for recognizing planar objects, in section 3. This system is able to determine object identity without searching a model base. Furthermore, the system is able to back-project objects to verify recognition without knowing any camera parameters. Another system of this type is given in Lamdan *et al.* (1988). Lamdan's system uses an affine model of the map from object to image planes, and so uses affine invariants. The affine model of projection breaks down when the object depth is not small with respect to the viewing distance. Our system models projection as a projective mapping, a model that works even for viewing large objects close to the camera.

There are many objects which are not planar. Recognizing curved surfaces from a single view is an important open problem in computer vision. In section 4, we show one way in which indexing functions can be computed from the image outline of an algebraic surface.

2. Background

We adopt the notation that corresponding entities in two different coordinate frames are distinguished by large and small letters. Vectors are written in bold font, e.g. $\mathbf{x}$ and $\mathbf{X}$. Matrices are written in typewriter font, e.g. $\mathtt{c}$ and $\mathtt{C}$.

Given a group $\mathcal{G}$ and a space $\mathcal{M}$, an action of $\mathcal{G}$ on the space associates with each group element $g \in \mathcal{G}$ a map $g : \mathcal{M} \rightarrow \mathcal{M}$:

$$id(x) = x \tag{2.1}$$

$$(g_1 \times g_2)(x) = g_1(g_2(x)) \tag{2.2}$$

where $g_1, g_2 \in \mathcal{G}$, id is the identity element of the group, and $\times$ is the group composition function. A *scalar invariant* of a group action has the following property:

> An invariant, $I(\mathbf{p})$, of a function $f(\mathbf{x}, \mathbf{p})$ subject to a group, $\mathcal{G}$, of transformations acting on the coordinates $\mathbf{x}$, is transformed according to $I(\mathbf{P}) = I(\mathbf{p})$. Here $g \in \mathcal{G}$ and $I(\mathbf{p})$ is a function only of the parameters, $\mathbf{p}$.

In what follows, we use the term invariant to mean exclusively a scalar invariant. An invariant under one group action may not be invariant under another–for example, the x-coordinate of a point on the plane is invariant under translation along the y-direction, but not under plane rotations. We present a variety of invariants, invariant under projection, below.

2.1. Examples of invariants

Example 1. The cross-ratio: The cross-ratio is a well-known projective invariant for four collinear points. Given $\mathbf{x_i}$, $i \in \{1, .., 4\}$, in homogeneous coordinates, their cross ratio is:

$$\frac{|M_{12}||M_{34}|}{|M_{13}||M_{24}}$$

where $M_{ij} = (\mathbf{x_i}, \mathbf{x_j})$ Four concurrent lines can also be used to form a cross-ratio, because concurrent lines are dual to collinear points on the projective plane.

Example 2. Five coplanar points: Given five coplanar points $\mathbf{x_i}$, $i \in \{1, .., 5\}$, in homogeneous coordinates, two projective invariants are defined:

$$I_1 = \frac{|M_{431}||M_{521}|}{|M_{421}||M_{531}|} \qquad I_2 = \frac{|M_{421}||M_{532}|}{|M_{432}||M_{521}|} \tag{2.3}$$

where $M_{ijk} = (\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k})$ and $|M|$ is the determinant of M. Should any triple of points become collinear the first invariant is undefined. An alternative definition appears in Mohr and Morin (1991). Again, we may use lines instead of points to form the invariants.

Example 3. Projective invariants for pairs of plane conics: A plane conic can be written as $\mathbf{x}^t \mathtt{c}_1 \mathbf{x} = 0$, for $\mathbf{x} = (x, y, 1)$ and a symmetric matrix $\mathtt{c}_1$, which determines the conic. A pair of coplanar conics has two scalar invariants, which we will describe here. Given conics with matrices of coefficients $\mathtt{c}_1$ and $\mathtt{c}_2$, we define:

$$I_{\mathtt{c}_1\mathtt{c}_2} = Trace(\mathtt{c}_1{}^{-1}\mathtt{c}_2)$$

$$I_{\mathbf{C}_2\mathbf{C}_1} = Trace(\mathbf{c}_2^{-1}\mathbf{c}_1)$$

Under the action $\mathbf{x} = \mathbf{T}\mathbf{X}$, $\mathbf{c}_1$ and $\mathbf{c}_2$ go to $\mathbf{C}_1 = \mathbf{T}^t\mathbf{c}_1\mathbf{T}$ and $\mathbf{C}_2 = \mathbf{T}^t\mathbf{c}_2\mathbf{T}$. In particular, using the cyclic properties of the trace, we find:

$$\begin{aligned} I_{\mathbf{C}_1\mathbf{C}_2} &= Trace(\mathbf{T}^{-1}\mathbf{c}_1^{-1}(\mathbf{T}^t)^{-1}\mathbf{T}^t\mathbf{c}_2\mathbf{T}) \\ &= Trace(\mathbf{c}_1^{-1}\mathbf{c}_2) \\ &= I_{\mathbf{c}_1\mathbf{c}_2} \end{aligned}$$

A similar derivation holds for $I_{\mathbf{c}_2\mathbf{c}_1}$. Note that $\mathbf{c}_1^{-1}\mathbf{c}_2$ transforms to $\mathbf{T}^{-1}\mathbf{c}_1^{-1}\mathbf{c}_2\mathbf{T}$, which is a similarity transformation, and so its eigenvalues are preserved. This provides an alternative demonstration of invariance. Since a conic can be represented both by $\mathbf{C}$ and $k\mathbf{C}$, where k is a scalar, to evaluate and interpret these invariants we need to make some assumption to set the relative scale of the conic matrices. All the conics we use will be normalized by the criterion $|\mathbf{C}| = 1$.

Example 4. Projectively invariant measurements: If there is a distinguished conic curve in the plane (say, $\mathbf{c}$), then for two points $\mathbf{x}_1$, $\mathbf{x}_2$ that do not lie on the conic, the function:

$$\frac{(\mathbf{x}_1{}^T\mathbf{c}\mathbf{x}_2)^2}{(\mathbf{x}_1{}^T\mathbf{c}\mathbf{x}_1)(\mathbf{x}_2{}^T\mathbf{c}\mathbf{x}_2)} \tag{2.4}$$

is independent of the frame in which the points and the conic are measured. This can be used to define a projectively invariant metric (see Springer, 1964). Furthermore, this supplies an invariant for a configuration of a conic and two points, or a conic and two lines by duality.

3. Recognizing planar objects using algebraic invariants

This section describes briefly the current implementation of a model-based recognizer for plane objects that uses algebraic invariants as indexing functions. This system has a model base that at the date of writing consisted of 33 objects. A detailed description of this system appears in Rothwell *et al.* (1991). Recognition proceeds in three stages:

1 **Feature extraction:** Conics and lines are extracted from image edge data.
2 **Hypothesis generation:** The invariants for groups of features are computed. We index the invariants measured against invariant values in the library using a hash table and, if a match is found, produce a recognition hypothesis.
3 **Hypothesis combination and verification:** Hypotheses that could have come from the same object are merged. The final system of hypotheses is verified by projecting edge data from an acquisition image to the test scene. Should the projected and scene edge data be sufficiently close the match is confirmed.

Although this system uses an hypothesize and test approach to recognition, the recognition hypotheses are achieved by using image primitives to *index directly into the model base*, without searching the model base and without pose computations. Thus, given that the indexing functions used are sufficiently powerful to distinguish between a wide variety of shapes, the size of the system's model base should not affect its performance.

We describe each stage in greater detail below, assuming that a model library exists. In this system, object models consist of a set of lines and conics, the projective invariants of this set, the model name, and an outline of the modeled object in some view. In fact, all model data are extracted from images of the object, using a process described in section 3.5.

3.1. Segmentation

The segmentation phase aims to extract straight lines, conics, and higher order curves from edge features so that we can form invariants. Chains of edge pixels are taken directly from the output of a local implementation of the Canny edge detector (Canny, 1983). Chains with single pixel breaks are reunited. Points where the chain turns sharply are marked (Pridmore *et al.*, 1987), and the chain is broken into separate segments at these points. We then use an efficient incremental fitting procedure, described in detail in Rothwell *et al.* (1991), to decide whether the points on a chain lie on a conic, a line, or some other curve. We note, however, that fitting conics to small portions of image curves is unstable, and avoid this practice.

The segmentation produces a collection of lines, conics and higher order plane curves. Some duplications can appear in this collection if, for example, a straight line in the image is broken into two aligned segments by an occlusion event. In this case, the two segments, represented by two copies of the same line, are merged, using a procedure detailed in Rothwell *et al.* (1991). The straight lines are then grouped into chains, consisting of all lines that are fitted to a single chain of edge points, with an ordering around the chain. This grouping is an attempt to reduce difficulties with matching combinatorics.

To determine all the invariants for groups of five lines in an image containing n lines we need to compute $O(n^5)$ invariants†, a cost that swamps the benefit of constant time library indexing. By using an ordered cycle we can compute invariants only for consecutive lines in a chain, with a cost of only $O(n)$, yielding potentially fast recognition. This strategy can miss potential matches as a result of poor segmentation, or because occlusion can lead to cycles which do not consist of the right lines in the right order. Our experience has been that this approach works well, when occlusion effects are moderate.

3.2. Generating recognition hypotheses

The segmenter produces a system of geometric primitives, consisting of chained lines, conics and other curves. At present our system uses only the chained lines and the conics. For each possible collection of primitives that will yield invariants, that is for each instance of five lines in sequence in a chain, two conics, or a conic and any two lines, we form a *feature vector*, consisting of the measured values of the invariants for those primitives. These values are then matched via a hash table to an object name in the model library. If the invariant values are within some error bound of indexing a model, the associated collection of primitives is considered to be a recognition hypothesis.

In fact, many collections of primitives may come from the same model instance: for

† Irritatingly, only $O(n)$ of the invariants are functionally independent.

example, an object consisting of a square plate with a circular hole in it admits six collections, each consisting of a conic and two lines. The recognition hypotheses that these collections produce are compatible, in the sense that a single instance could explain all of them simultaneously. We merge all compatible hypotheses into joint hypotheses, using a process described in detail in Rothwell *et al.* (1991). This process substantially reduces the amount of time verifying match hypotheses, as verifying a single joint hypothesis verifies all its many constituents as well.

3.3. Verifying joint hypotheses

The invariants we have used are effective shape descriptors, but they do not capture every detail of a shape. We must therefore confirm hypothesized matches to avoid false matches. Verification is expensive, and can be hard for occluded scenes. In typical present-day systems, verification is based on using the computed pose of the recognized object and the camera parameters (normally the focal length of the lens and the position of the camera center on the sensing array) to predict its outline in the image (e.g. Mundy and Zisserman, 1991). The image is then searched for object features, other than those used to form the hypothesis, which are consistent with the predicted outline.

Our verification scheme proceeds in a similar fashion, but requires no knowledge of camera parameters. This is an advantage, as these parameters are often hard to measure accurately. We exploit the fact that the map from a planar object to its image is a projectivity (see Forsyth *et al.*, 1991a), and so is uniquely determined by four point correspondences or by four line correspondences. In fact, given any structure sufficiently complicated to admit projective invariants, we can determine this projectivity. We then apply this transformation to an outline of the model to project it into the image, where it can be compared with image edge data.

If the projected edge points lie close (within 5 pixels) to image edge points of the same orientation (within 15^o), we count this as support for the recognition hypothesis. If more than a certain proportion of the projected model data is supported [we use 50% as a support threshold; Mundy and Heller (1990), in a system for recognizing polyhedral objects in image data, used 30% successfully], the recognition hypothesis is confirmed.

As computing the Euclidean distance from a point in the image to the nearest edge location is expensive, we use the 3-4 distance transform of Borgefors (1988) to provide an approximation. The distance transform applies chamfer masks to the output of the edge detection system. The orientation associated with image edgels is that of the Canny output, whereas that associated with projected model features depends on the type of the feature, and is detailed in Rothwell *et al.* (1991).

Verification is hard for occluded scenes because an incorrect match may have as much image support as a heavily occluded correct match; that is, for scenes where there is dense edge data it is quite likely that a large number of edges may be close to, and have the same orientation as, the projected model edges. This means that there is a clear tradeoff in setting the support threshold. A high value will ensure that false matches are rejected, but may lead to false negatives. A low value will encourage false positives. More sophisticated verification algorithms are a subject of ongoing research.

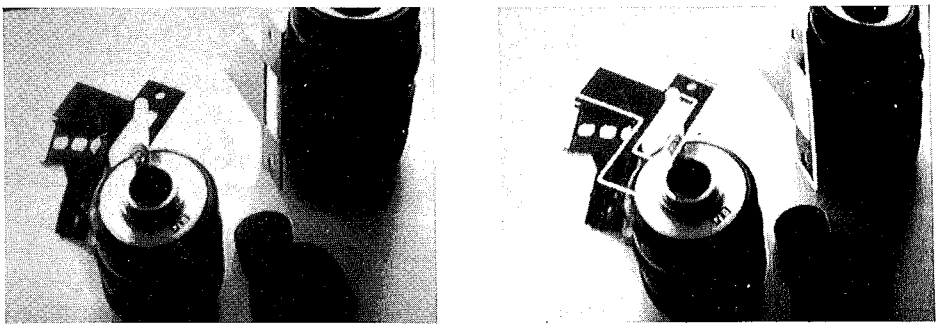

Figure 1. Two different objects (lock striker-plates) are recognized in this image even with substantial occlusion and significant perspective distortion.

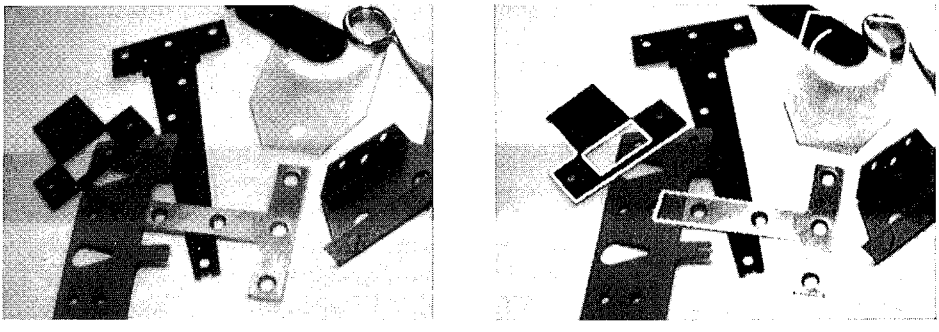

Figure 2. The three objects in the library are recognized despite considerable occlusion and clutter. Edge data from acquisition images is projected onto the test scene using the model to image transformation hypothesised by the match. The close match between the projected data (shown in white), and the scene edges shows that the recognition hypothesis is valid.

3.4. Examples

A recognition example is shown in figure 1. Two different objects are found in the image by the recognizer even though one of the objects is substantially occluded and the other has undergone significant perspective distortion. Figure 2 shows another set of typical results. In this figure, a bracket is recognized in a scene with occlusion and clutter caused by other objects.

Figure 2 also demonstrates the verification process. The edge data from an image used to model the bracket has been projected onto the test image and is shown in white. The close correspondence between this data and the test edge data (59.3% within 5 pixels and 15°) confirms the recognition hypothesis.

3.5. Building a model library

A novel aspect of this work is the ease with which models may be acquired. We extract all possible invariants from an unoccluded view of the isolated object, and enter these invariants into a feature vector. We now measure another feature vector for the same

object from a different viewpoint. We then compare these two vectors. If, for a given value in the first vector, a corresponding value exists in the second vector, we accept that the lines involved are coplanar, correspond, and will give rise to a useful invariant, and insert this information into the model-base. The edge data forming these features are also stored, for use in the hypothesis verification stage.

If, however, a measurement changes between two different viewpoints we know that the features used to compute the measurement are not coplanar, or that the features are a result of segmentation problems, and so the measurement is not useful, and is rejected.

4. Algebraic techniques for 3D object identification

Not all objects are planar. There is great interest in recognizing three dimensional objects from their outlines. This is a difficult problem because, as one views a surface from different viewpoints, the appearance of the outline changes because the contour generator (the curve, lying on the surface, that projects to the image outline) moves around on the surface. In this section, we consider using the outline as a cue to the shape of the surface.

Indexing functions cannot be recovered from a single outline if the surfaces involved are completely unrestricted because, for example, we can disturb any such function by adding a smooth bump to the side of the surface that is hidden from the viewer. In this section and the next, we show ways of obtaining such descriptors from a single outline, for *algebraic surfaces*, i.e. surfaces given by the vanishing of a single polynomial. Algebraic surfaces have substantial advantages as a model problem.

- They are "rigid", in the sense that local modifications to the surface are not possible (informally, one cannot hide an extra bump on the surface).
- Collections of algebraic surfaces are parameterized in a relatively accessible way. This means that geometrical matters like bi-tangency and singularity are relatively easily settled.
- Algebraic surfaces, particularly quadrics and bi-rational cubic patches, are widely used in industrial applications.
- The contour generator and the outline of an algebraic surface are both algebraic curves. It is particularly easy to determine the form of both curves, given the surface. Algebraic curves have interesting and useful global geometric properties, and, unlike smooth curves, have an important and useful intrinsic geometry.

It is natural to allow complex points when considering algebraic curves: for example, if one admits only real points, the curve $x^2+y^2+1=0$ has no geometry worth mentioning. Although considering complex points leads to elegant geometry, it is hard to envisage imaging them with a CCD camera. In fact, if a curve has a sufficiently rich set of real points, it is possible to determine its coefficients. In turn, if one knows the coefficients of a curve, one can construct its complex points.

From now on, we assume that all the curves and surfaces we deal with are irreducible[†],

[†] Loosely, this means they consist of only a single curve or surface, if one considers the complex points as well. For example, the conic $x^2 - y^2 = 0$ is reducible, because it consists of a pair of lines. The conic $x^2 + y^2 - 1 = 0$ is irreducible. Unfortunately, an irreducible curve can have real points that look

and that the coefficients of the curve can be determined from its real points alone. These are reasonable assumptions. First, the outline of a generic irreducible surface from a generic viewpoint is irreducible. Second, we will primarily deal with surfaces where the real points determine the surface (it is not possible to manufacture algebraic surfaces without this property), and we can reasonably assume that the real points of the outline determine the full outline.

In this section, we are going to concentrate on recovering some parameters of a surface, given its outline, assuming that the surface comes from some restricted class of surfaces. These parameters should be invariant to Euclidean actions at least, and so can be used to identify the surface.

4.1. The outline

When a surface is viewed under perspective, its outline is given by the intersection of the image plane with a cone formed by a system of rays through the camera focal point and tangent to the surface. Without loss of generality, we can assume that the camera focal point is at the origin. Figure 3 shows this cone for a simple shape. If the surface is given by $P(x, y, z) = 0$, the tangency condition is given by $x\frac{\partial P}{\partial x} + y\frac{\partial P}{\partial y} + z\frac{\partial P}{\partial z} = Q(x, y, z) = 0$. The equation of the cone, as a function of the coefficients of the surface, can be computed by eliminating t from $P(tx, ty, tz)$ and $Q(tx, ty, tz)$. This resultant vanishes along the whole length of any line through the origin given by (x, y, z), if there is a point on the line where the line is tangent to the surface. Thus, it is homogeneous and represents the desired cone through the origin. The image outline is the curve obtained by slicing this cone with a plane.

Lemma 4.1. *For a generic surface of degree n viewed from the origin, the cone, and hence its outline, has degree $n(n-1)$.*

Proof. We work in projective space, and consider a generic focal point (f_0, f_1, f_2, f_3). The surface, say $P(x, y, z, t)$, has degree n, and the tangency condition requires that planes tangent to the surface at a point on the contour generator must pass through the focal point. This is:

$$f_0\frac{\partial P}{\partial x_0} + f_1\frac{\partial P}{\partial x_1} + f_2\frac{\partial P}{\partial x_2} + f_3\frac{\partial P}{\partial x_3} = 0$$

and has degree $n-1$. The curve is the complete intersection of these two hypersurfaces, and so has degree $n(n-1)$ in the generic case. □

4.2. Recovering surface parameters from the outline

The cone tangent to the surface is the key to our argument. This cone can be completely reconstructed from the image outline, given the focal point and its position relative to the image plane in camera coordinates. We assume that this information is always available, and work with the cone. Any rigid motion of the camera with respect to the surface coordinate frame can be expressed as a translation of the focal point followed by a rotation

like several distinct connected curves. In this case, each component implies the other, because one can determine the coefficients of the whole curve from each separate component.

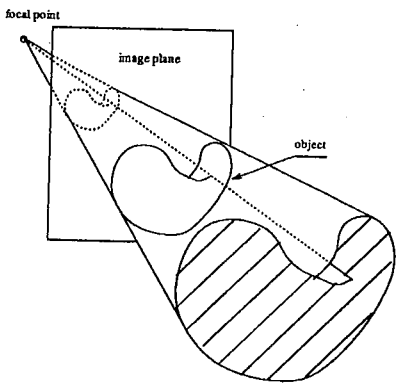

Figure 3. The cone of rays, through the focal point and tangent to the object surface, that forms the image outline, shown for a simple object.

of the camera, keeping the focal point fixed. Although such a rotation changes the outline seen in the image because it changes the position of the image plane with respect to the cone, it has no effect on the geometry of the cone itself, but merely rotates the frame within which the cone is measured. As a result, if we consider only those properties of the cone that are invariant to rotation, camera rotation parameters no longer come into our problem. Much of what follows relies on being able to compute rotation invariants of cones and surfaces. We have found the *symbolic method* of 19th century invariant theory (discussed in, for example, Abhyankar, 1992; Forsyth *et al.*, 1991a; Salmon, 1885; Weyl, 1946) to be the best technique for computing such expressions.

We first compute some of the rotation invariants of a general cone or system of cones of the given degree, as functions of the coefficients of the cone. The numerical values of these rotation invariants can be measured from image data. However, because we are dealing with a parameterized class of surfaces, we can compute symbolic expressions for the cone's rotation invariants as functions of surface parameters and the translation between the surface and the focal point. In particular, the surface $P(x, y, z) = 0$ is translated to some point in the camera frame, so that its equation becomes $P'(x, y, z) = P((x - t_x), (y - t_y), (z - t_z)) = 0$, and the coefficients of the cone over this surface are computed symbolically. The rotation invariants of this cone are functions of the coefficients of the translated surface alone. We can measure these invariants, and compute them as symbolic expressions in the coefficients of the translated surface. The result is a polynomial system in surface parameters and translation.

4.3. Worked example and results

The method as described requires a large amount of symbolic computation to set up, but has been shown to work on simple surfaces. Here we work the example of a pair of spheres of unknown but equal radius, and we will recover the ratio of their 3D separation to the radius (which is clearly a Euclidean invariant) from a single image. We will denote the separation of the spheres by d, and their radius by R.

The cone generated by a sphere is right circular, and so the problem of determining the rotation invariants of the image cone becomes one of determining the rotation invariants of a pair of quadratic cones in space. If we write one cone as $\sum_{i+j+k=2} f_{ijk}x^i y^j z^k = 0$, and the other as $\sum_{i+j+k=2} g_{ijk}x^i y^j z^k = 0$, we obtain from the symbolic method the following invariants, among others:

$$i_1 = \frac{(-3g_{020}g_{101}^2 + 3g_{011}g_{101}g_{110} - 3g_{002}g_{110}^2 - 3g_{011}^2g_{200} + 12g_{002}g_{020}g_{200})}{(2(2g_{002} + 2g_{020} + 2g_{200})^3)}$$

$$i_2 = \frac{(-g_{011}^2 + 4g_{002}g_{020} - g_{101}^2 - g_{110}^2 + 4g_{002}g_{200} + 4g_{020}g_{200})}{(2(2g_{002} + 2g_{020} + 2g_{200})^2)}$$

$$i_3 = \frac{(-3f_{020}f_{101}^2 + 3f_{011}f_{101}f_{110} - 3f_{002}f_{110}^2 - 3f_{011}^2f_{200} + 12f_{002}f_{020}f_{200})}{(2(2f_{002} + 2f_{020} + 2f_{200})^3)}$$

$$i_4 = \frac{(-f_{011}^2 + 4f_{002}f_{020} - f_{101}^2 - f_{110}^2 + 4f_{002}f_{200} + 4f_{020}f_{200})}{(2(2f_{002} + 2f_{020} + 2f_{200})^2)}$$

$$i_5 = \frac{(2(f_{002} + f_{020} + f_{200})\phi_1)}{(-g_{011}^2 + 4g_{002}g_{020} - g_{101}^2 - g_{110}^2 + 4g_{002}g_{200} + 4g_{020}g_{200})} \quad \text{where}$$

$$\begin{aligned}\phi_1 = {} & (-f_{200}g_{011}^2 + 4f_{200}g_{002}g_{020} + f_{110}g_{011}g_{101} - 2f_{101}g_{020}g_{101} - f_{020}g_{101}^2 - \\ & 2f_{110}g_{002}g_{110} + f_{101}g_{011}g_{110} + f_{011}g_{101}g_{110} - f_{002}g_{110}^2 + 4f_{020}g_{002}g_{200} - \\ & 2f_{011}g_{011}g_{200} + 4f_{002}g_{020}g_{200})\end{aligned}$$

$$i_6 = \frac{((-f_{011}^2 + 4f_{002}f_{020} - f_{101}^2 - f_{110}^2 + 4f_{002}f_{200} + 4f_{020}f_{200})\phi_2)}{(4(2g_{002} + 2g_{020} + 2g_{200}))} \quad \text{where}$$

$$\begin{aligned}\phi_2 = {} & (-f_{110}^2g_{002} + 4f_{020}f_{200}g_{002} + f_{101}f_{110}g_{011} - 2f_{011}f_{200}g_{011} - f_{101}^2g_{020} + \\ & 4f_{002}f_{200}g_{020} - 2f_{020}f_{101}g_{101} + f_{011}f_{110}g_{101} + f_{011}f_{101}g_{110} - \\ & 2f_{002}f_{110}g_{110} - f_{011}^2g_{200} + 4f_{002}f_{020}g_{200})\end{aligned}$$

$$i_7 = \frac{((f_{002} + f_{020} + f_{200})\phi_3)}{(2g_{002} + 2g_{020} + 2g_{200})} \quad \text{where}$$

$$\begin{aligned}\phi_3 = {} & (2f_{020}g_{002} + 2f_{200}g_{002} - f_{011}g_{011} + 2f_{002}g_{020} + 2f_{200}g_{020} - f_{101}g_{101} - \\ & f_{110}g_{110} + 2f_{002}g_{200} + 2f_{020}g_{200})\end{aligned}$$

These invariants were obtained from the symbolic method without regard to their geometrical meaning. However, this system will capture such geometrically meaningful rotation invariants as the angle between the cones, and their eigenvalues.

The apex of the cones lies at the focal point, and the image outlines form a section of the cone. Thus, if the position of the camera focal point is with respect to the image coordinate system, we can reconstruct the cones from an image outline, and so determine its coefficients numerically. By substituting these numerical values into the symbolic expression above, we can determine numerical values of these invariants from image data. These numerical values capture the geometry of the cones and the focal point, up to rotation, because they are rotationally invariant. As a result, we can ignore camera rotations. We will use these measurements to determine surface parameters.

However, we have more than just numerical measurements, because we can determine symbolic expressions for the cones over a pair of spheres, both of arbitrary radius R, and separated by an arbitrary distance d, and translated to some arbitrary point in space

(t_x, t_y, t_z). These expressions can be computed using elimination, in the manner described above. We now substitute the symbolic expression for each coefficient in the expressions for the invariants given above, and obtain symbolic expressions for the invariants in terms of the geometry of the surface, and the translation from the surface to the focal point. For example, in the geometry we are working with, the following substitution would apply:

$$g_{101} \rightarrow (-8dt_z - 8t_x t_z)$$

This process yields a system of equations of the form $\frac{p_j}{q_j} = m_j$, where p_j and q_j are polynomials in R, d, and (t_x, t_y, t_z). Here m_j is the measured value of i_j, obtained by substituting the numerical values of the cone coefficients we determined above.

These expressions are extremely complicated, but admit a simplifying change of variables. We write:

$$\begin{aligned}
v_1 &= R^2 \\
v_2 &= R^2 - t_x^2 - t_y^2 - t_z^2 - d^2 - 2dt_x \\
v_3 &= 3R^4 - R^2d^2 - 2R^2dt_x - 3R^2t_x^2 - 3R^2t_y^2 + d^2t_y^2 - 3R^2t_z^2 + d^2t_z^2 \\
v_4 &= d^2 + 2dt_x
\end{aligned}$$

By making this change of variables, and then cross multiplying and subtracting to get equations that vanish (i.e. $p_j - m_j q_j = 0$), we obtain a system of seven equations. It turns out that we require only the following three, obtained from i_5, i_6, and i_7:

$$\begin{aligned}
e_1 &= 128v_2(4m_5v_1^2 + 8m_5v_1v_4 + 10m_5v_1v_2 + 4m_5v_4v_2 + 4m_5v_2^2 - v_3) \\
e_2 &= 128(v_4 + v_2)(4m_6v_1^2 + v_1v_4 + 2m_6v_1v_4 + 10m_6v_1v_2 + 4m_6v_4v_2 + 4m_6v_2^2 - v_3) \\
e_3 &= -16(m_7(4v_1^2 + 8v_1v_4 + 16v_1v_2 + 16v_4v_2 + 16v_2^2) - 2v_2^2 - v_1v_2 - 2v_4v_2 - v_3)
\end{aligned}$$

each of which vanishes. Because these equations vanish, the expressions $16e_1 + 128v_2e_3$ and $16e_2 + 128(v_4 + v_2)e_3$ must vanish; expanding these expressions, factoring the result, dropping irrelevant factors (those not containing any m_i), and recalling that the values of the m_i are known, we obtain two homogeneous linear equations in four unknowns, $v_1 \ldots v_4$. Adjoin e_3, substitute u_0 for $\frac{v_2}{v_1}$, u_1 for $\frac{v_3}{v_1}$ and u_2 for $\frac{v_4}{v_1}$, and we obtain:

$$\begin{aligned}
4m_5 - 4m_7 + (1 + 2m_5 - 8m_7)u_1 &= 0 \\
4m_6 - 4m_7 + (1 + 2m_6 - 8m_7)u_2 + (1 + 2m_6 - 8m_7)u_1 &= 0 \\
-64m_7 - 128m_7u_2 + 16y - 256m_7u_1 + 32u_2u_1 - 256m_7u_2u_1 + & \\
32u_1^2 - 256m_7u_1^2 + 16u_0 &= 0
\end{aligned}$$

Here the m_i are known, numerical measurements, so this system is triangular, and can be solved uniquely for u_0, u_1 and u_2.

We now consider the expressions for the v_i; write these as $v_i - f_i(t_x, t_y, t_z, d, R) = 0$, and substitute as above, writing as well α for $(\frac{d}{R})^2$, β for $\frac{dt_x}{R^2}$ and γ for $\frac{t_x^2 + t_y^2 + t_z^2}{R^2}$. Note that α is the parameter we wish to recover. Discarding irrelevant factors, we obtain:

$$\begin{aligned}
-1 + \alpha + 2\beta + \gamma + u_0 &= 0 \\
-3 + \alpha + 2\beta + \beta^2 + 3\gamma - \alpha\gamma + u_1 &= 0 \\
-\alpha - 2\beta + u_2 &= 0
\end{aligned}$$

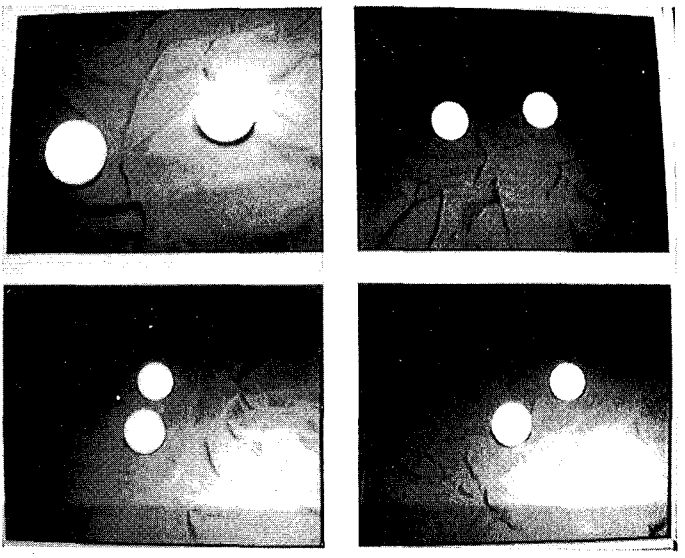

Figure 4. Four views of a pair of ping-pong balls, placed approximately 5cm apart.

Given that u_0, u_1 and u_2 are uniquely defined by the triangular system above, we can solve this system up to a two-fold ambiguity in α.

Figures 4 to 6 each show four views of a pair of ping-pong balls, lying on a sheet of black cloth. In each figure the parameter α is different. Figure 7 shows a graph of the values of α obtained for these figures, by fitting conics to the image outlines and using the measurements and system of equations described above. The value is constant as the viewpoint changes, and changes as the geometry changes, which is the property required for recognition. These measurements require a calibrated camera, but are impressively robust to errors in camera calibration. The results shown were obtained from a camera that had been calibrated by reading the focal length from the lens, computing the aspect ratio using a tape measure, and assuming the camera center was in the center of the pixel array.

It is, however, essential for a good result to ensure that the coefficients of the conic fitted to the image outline genuinely represent a right circular cone (i.e. the conic matrix has two equal eigenvalues), and the fitting program was designed to accomplish this.

4.4. Discussion

In the example given above, a number of apparently arbitrary changes of variable led to a simple system with a low Bézout number. In fact, this is an illustration of a general principle, that the system above admits a functional decomposition, as the following argument illustrates.

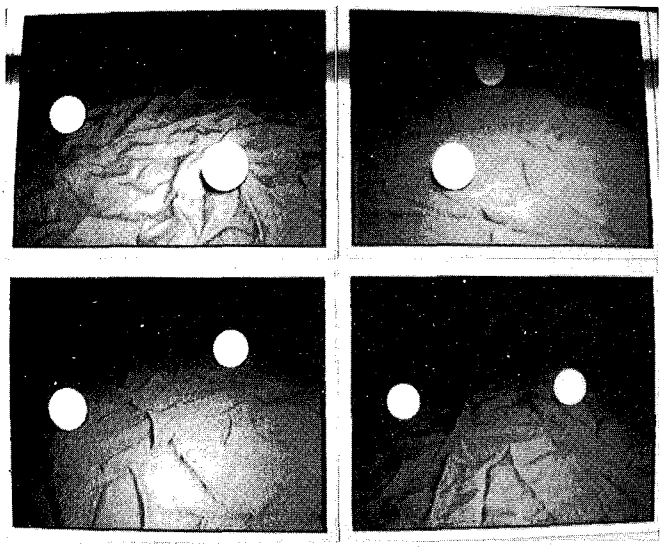

Figure 5. Four views of a pair of ping-pong balls, placed approximately 10cm apart.

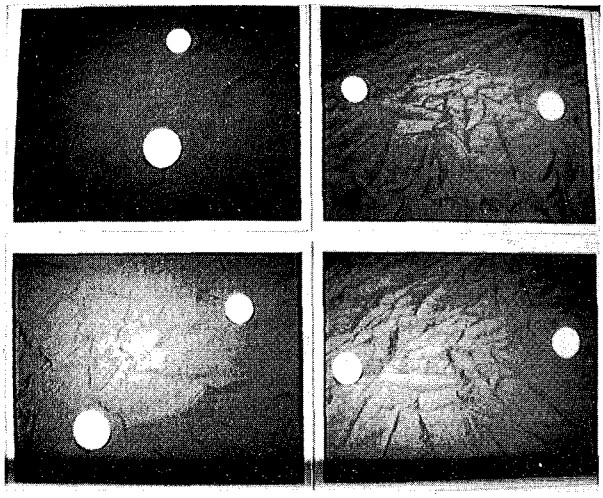

Figure 6. Four views of a pair of ping-pong balls, placed approximately 15cm apart.

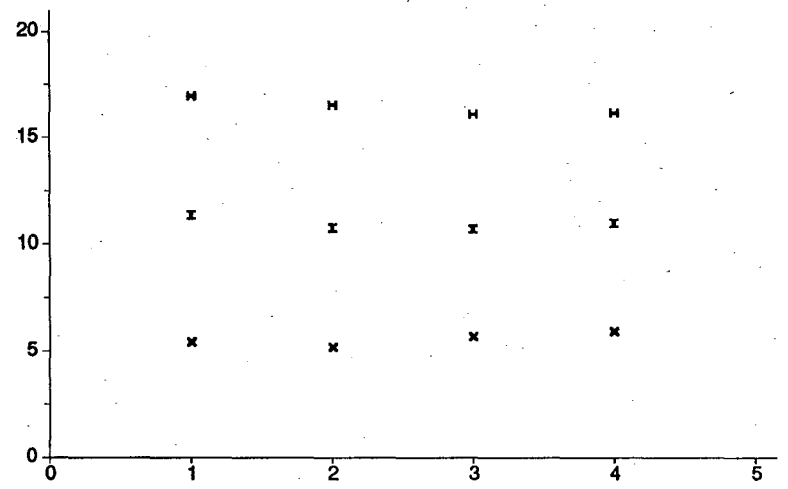

Figure 7. This graph shows the parameter α measured and plotted against the view number, for the views of ping-pong balls shown above. In each case, the parameter is constant as the viewpoint changes, and changes when the geometry changes.

Recall that the surface $P(x, y, z) = 0$ is translated to some point in the camera frame, so that its equation becomes $P'(x, y, z) = P((x - t_x), (y - t_y), (z - t_z)) = 0$, and the resulting surface is viewed from the origin.

The outline cone is a cone of tangents to P', passing through the focal point. The coefficients of this cone are functions of the coefficients of P', and nothing else. This means that the rotation invariants of this cone are also functions of the coefficients P', and nothing else. Since they are invariant to rotation they must be *rotation invariants of* P'.

In turn, for the groups we are dealing with, there is a finite number of "primitive" invariants, and all others are functions of these primitives. As a result, a functional decomposition of the system derived from the rotation invariants of the cone is guaranteed to exist, although it may be trivial. Hence, as in the example above, the cone system consists of polynomials in a set of new variables which are themselves non-trivial polynomial functions of the original variables, that is, the coefficients of P and the translation of that surface (t_x, t_y, t_z) above. We can therefore write the system derived from the cone as $Q_i(v_j(t_x, t_y, t_z, \lambda)) = 0$, where the v_j are known functions. This is what yielded the substitutions $v_1 ... v_4$ above.

It appears that this functional decomposition might be achieved automatically by a symbolic algebra program, at least in principle, because the v_j can be determined. To date, we have constructed the functional decomposition by hand.

Since the Euclidean invariants of the translated surface must also be invariant to rotation, they are functions of its primitive rotation invariants, and so it should be possible to recover some or all of the surface's Euclidean invariants after only the first

stage of solution process. In fact, we could simply solve for the numerical values of the "primitive" rotation invariants (the v_j), and plug these values into the expressions for the Euclidean invariants as functions of the rotation invariants. The argument has merit, however, because it suggests an explanation for the relatively low Bézout number observed.

5. Conclusions

Indexing functions offer a highly successful approach to building a model-based vision system based around plane objects. We have demonstrated this approach in a model-based vision system that is successful at recognizing instances of objects drawn from a large model base.

Constructing indexing functions that identify a three-dimensional curved surface from its outline remains a challenging problem. In this paper, we have shown an approach to this problem which has been successfully implemented. This approach requires demanding symbolic computations, but works on real images of simple scenes. In a companion paper (Forsyth *et al.*, 1991b), we describe a construction that yields indexing functions for rotationally symmetric curved surfaces, which require no knowledge of camera parameters. Recent unpublished work shows that indexing functions can be computed from the outline of a generic algebraic surface of any degree, by studying the embeddings of the outline into space.

Acknowledgments

Our thanks go to Michael Brady and the Robotics Research Group, Oxford University, and Jonathan Simon and the Department of Computer Science, University of Iowa. We thank Jean Ponce for allowing us to use his distributed system for solving systems of polynomial equations. Our thanks to Andrew Blake and Margaret Fleck for a number of stimulating discussions. We thank General Electric for providing telephone conferencing facilities during the development of this work. Forsyth was supported by Magdalen College of Oxford University, General Electric and the University of Iowa. Mundy and Rothwell received support from General Electric. Zisserman was supported by the Science and Engineering Research Council. We thank anonymous referees, whose extensive comments have substantially improved both the content and the accuracy of this paper.

References

S.S. Abhyankar (1992), "Invariant theory and the enumerative combinatorics of young tableaux", J.L. Mundy and A.P. Zisserman, eds., *Geometric Invariance in Vision*, MIT Press, Cambridge, MA.

G. Borgefors (1988), "Hierarchical chamfer matching: a parametric edge matching algorithm", *IEEE Trans. Patt. Anal. Mach. Intell.*, **10**(6), 849–865.

J.F. Canny (1983), *Finding Edges and Lines in Images*, Technical Report 720, AI Laboratory, MIT, Cambridge, MA.

J. Dieudonné and J.B. Carrell (1971), *Invariant Theory Old and New*, Academic Press, London.

D.A. Forsyth, J.L. Mundy, A.P. Zisserman, A. Heller, C. Coehlo and C.A. Rothwell (1991a), "Invariant descriptors for 3D recognition and pose", *IEEE Trans. Patt. Anal. Mach. Intell.*, **13**, 10.

D.A. Forsyth, J.L. Mundy, A.P. Zisserman and C.A. Rothwell (1991b), "Recognizing curved objects from

their outlines", J.L. Mundy and A.P. Zisserman, eds., *Geometric Invariance in Vision*, MIT Press, Cambridge, MA.

J.H. Grace and A. Young (1903), *The Algebra of Invariants*, Cambridge University Press, Cambridge.

D.P. Huttenlocher and S. Ullman (1987), "Object recognition using alignment", *Proc. Int. Conf. Comput. Vision 1*, London, 102–111.

Y. Lamdan, J.T. Schwartz and H.J. Wolfson (1988), "Object recognition by affine invariant matching", *Proc. IEEE Conf. Comput. Vision Patt. Recog.*, Ann Arbor, Michigan, 335–344.

E.P. Lane (1941), *A Treatise on Projective Differential Geometry*, University of Chicago press, Chicago, IL.

R. Mohr and L. Morin (1991), "Relative positioning from geometric invariants", *Proc. IEEE Comput. Vision Patt. Recog.*, Maui, Hawaii, 139–144.

J.L. Mundy and A.J. Heller (1990), "The evolution and testing of a model-based object recognition system", *Proc. 2nd Int. Conf. Comput. Vision*, Japan, 268–282.

J.L. Mundy and A.P. Zisserman (1991), "Introduction", J.L. Mundy and A.P. Zisserman, eds., *Geometric Invariance in Vision*, MIT Press, Cambridge, MA.

V.-D. Nguyen, J.L. Mundy and D. Kapur (1991), "Modeling generic polyhedral objects with constraints", *Proc. IEEE Conf. Comput. Vision Patt. Recog.*, Maui, Hawaii, 479–485.

J. Ponce and D.J. Kriegman (1989), "On recognizing and positioning curved 3 dimensional objects from image contours", *Proc. DARPA Image Understanding workshop*, Palo Alto, CA, 461–470.

A.P. Pridmore, J. Porrill and J.E.W. Mayhew (1987), "Segmentation and description of binocularly viewed contours", *Image Vision Comput.*, 5(2), 132–138.

C.A. Rothwell, A.P. Zisserman, D.A. Forsyth and J.L. Mundy (1991), "Efficient model library access by projectively invariant indexing functions", to appear.

G. Salmon (1885), *Lessons Introductory to the Modern Higher Algebra*, reprinted by Chelsea Publishing Company, NY.

C.E. Springer (1964), *Geometry and Analysis of Projective Spaces*, W.H. Freeman and Co., San Francisco, CA.

D.W. Thompson and J.L. Mundy (1987), "3D model matching from an unconstrained viewpoint", *Proc. IEEE Conf. Robotics and Automation*, Atlanta, 208–220.

P.C. Wayner (1991), "Efficiently using invariant theory for model-base matching", *Proc. Conf. Comput. Vision Patt. Recog.*, Maui, Hawai, 473–479.

I. Weiss (1988), "Projective invariants of shapes", *Proc. DARPA Image Understanding workshop*, Cambridge, MA, 1125–1134.

H. Weyl (1946), *The Classical Groups and their Invariants*, 2nd ed., Princeton University Press, Princeton, NJ.