ORIENTATION AND RECOGNITION OF BOTH NOISY AND PARTIALLY OCCLUDED 3-D OBJECTS FROM SINGLE 2-D IMAGES

Diane Patricia Illing

A thesis submitted in partial fulfilment of the requirements
of the Council for National Academic Awards for the
degree of Doctor of Philosophy

September 1990

Department of Mathematics and Computing,
The Polytechnic of Wales,
Pontypridd,
Mid Glamorgan,
CF37 1DL
in collaboration with
The Royal Aerospace Establishment, Aberporth.

ABSTRACT

Orientation and Recognition of Both Noisy and Partially Occluded 3-D Objects from Single 2-D Images

Diane Patricia Illing

This work is concerned with the problem of 3-D object recognition and orientation determination from single 2-D image frames in which objects may be noisy, partially occluded or both.

Global descriptors of shape such as moments and Fourier descriptors rely on the whole shape being present. If part of a shape is missing then all of the descriptors will be affected. Consequently, such approaches are not suitable when objects are partially occluded, as results presented here show.

Local methods of describing shape, where distortion of part of the object affects only the descriptors associated with that particular region, and nowhere else, are more likely to provide a successful solution to the problem.

One such method is to locate points of maximum curvature on object boundaries. These are commonly believed to be the most perceptually significant points on digital curves. However, results presented in this thesis will show that estimators of point curvature become highly unreliable in the presence of noise. Rather than attempting to locate such high curvature points directly, an approach is presented which searches for boundary segments which exhibit significant linearity; curvature discontinuities are then assigned to the junctions between boundary segments. The resulting object descriptions are more stable in the presence of noise.

Object orientation and recognition is achieved through a directed search and comparison to a database of similar 2-D model descriptions stored at various object orientations. Each comparison of sensed and model data is realised through a 2-D pose-clustering procedure, solving for the coordinate transformation which maps model features onto image features. Object features are used both to control the amount of computation and to direct the search of the database.

In conditions of noise and occlusion objects can be recognised and their orientation determined to within less than 7 degrees of arc, on average.

ACKNOWLEDGEMENTS

I would like to thank the following for their helpful comments and suggestions during the drafting of this thesis; Dr. P.T. Fairney, Dr. B.F. Jones, Dr. D.H. Smith and Dr.M.Al-Akaidi.

Special thanks to Dr. P. T. Fairney for his encouragement, support and, above all, positive and most constructive attitude during the course of this work.

Financial support was provided initially by an SERC-CASE award in collaboration with the Royal Aircraft Establishment, Aberporth, whose contribution is acknowledged. My thanks also to the Department of Mathematics and Computing at the Polytechnic of Wales for providing the financial means for me to complete this work.

CONTENTS

1. **Introduction** 1

   1.1 Problem Outline 1

   1.2 Approaches to Model-based Object Recognition and Orientation Determination 2

2. **Hardware and Software Tools** 15

   2.1 Equipment Used 15

   2.2 Definition of Orientation 16

   2.3 Objects Used and Image Generation 17

3. **Representing Shape 1 - Global Shape Descriptors** 22

   3.1 Moment-based Methods 22

   3.2 Fourier Descriptors 28

   3.3 Summary of Global Descriptors of Shape 31

4. **Representing Shape 2 - On the Performance of Point Curvature Estimators** 33

   4.1 Introducing Curvature 33

   4.2 The Literature 34

   4.3 Experimental Analysis of Curvature Estimators 38

   4.4 Results 42

   4.5 Discussion 42

   4.6 Application to the Method of Rosenfeld & Johnston and to the Method of Rosenfeld & Weszka 46

   4.7 Conclusions on the Performance of Curvature Estimators 52

5. **Representing Shape 3 - Locating Sides** 54

   5.1 The Approach 54

   5.2 The Algorithm 55

   5.3 The Performance of the Algorithm 64

6. **The Database of Model Descriptions** ***67***

6.1 Computing the Model Data *67*

6.2 The Database of Arrays of Model Data *68*

6.3 Retrieving Model Data *72*

7. **Matching Sensed and Stored Model Descriptions** ***74***

7.1 The Pose-clustering Paradigm in 2-D Space *74*

7.2 Extension to Determine 3-D Orientation *77*

7.3 The Registration Process *78*

7.4 Error Analysis *90*

7.5 The Effects of Errors *94*

8. **Reconstructing Partially Occluded Object Boundaries and Refining the Match** ***97***

8.1 Information Preserving Methods of Describing Shape *97*

8.2 Reconstructing the Object Boundary *98*

9. **Experimental Results** ***102***

10. **Discussion** ***114***

10.1 Comparison of the Average Performance of the Methods in Noise *114*

10.2 Comparison of the Average Performance of the Methods Under Partial Occlusion *116*

10.3 The Line segment Matching Algorithm: Results from Real Images *120*

10.4 The Line segment Matching Algorithm: Considerations on Effectiveness and Efficiency *121*

10.5 Practical Considerations and Future Work *126*

11. **Conclusions** ***130***

References ***132***

CHAPTER ONE

Introduction

1.1 Problem Outline

The aim of this research project was to investigate and develop techniques for determining the orientation of known 3-D objects from single 2-D images. Specifically, the research has been concerned with man-made, airborne objects such as missiles, aircraft etc., in particular, with reference to the analysis of trials data in which a series of images of flying craft are gathered and require subsequent analysis to determine the orientation of the craft in each frame. In such situations, where very little control over imaging conditions exists, varying background, lighting, clouds, smoke etc. and the resulting segmentation problems may make observation of the whole object physically impossible. Incomplete or distorted shapes can result, either due to the loss of part of the shape, or due to the addition of background to the object. Indeed, the problems associated with poor edge detection and segmentation can make a substantial difference to the resulting description of a shape, and thus to matching and final estimates of orientation obtained.

The objects are effectively located at infinity and thus very little, or no, perspective exists in the images, i.e. orthogonal projection can normally be assumed.

A particular requirement is to achieve accuracies of ±5 degrees in each of the rotation angles defining orientation (see section 2.2). This obviously has implications for the density of stored model shape descriptions.

Thus the problem that needs to be investigated is the determination of the orientation of a known 3-D object from a single instance of some 2-D orthogonal projection of it in conditions of noisy and corrupted or missing data.

Whilst the result of a trial is a time-sequence of images, each frame providing constraints on the orientation of the object in the next frame, this work only assumes that a single isolated frame is available and does not exploit information from the whole time sequence.

In the work presented here the allied problem of object recognition is also addressed, in conjunction with the process of determining orientation.

1.2 Approaches to Model-based Object Recognition and Orientation Determination

Much work has been published on the problem of object recognition/classification, both for the 2-D and 3-D cases, for complete and partially occluded objects. However, the specific problem of the precision of orientation determination is rarely explicitly covered in the literature. This work remedies this situation, investigating the performance of some of the more traditional approaches, with particular reference to the problem outlined, and also presents a novel approach to solving for 3-D orientation from single 2-D images under conditions of weak perspective, or orthogonal projection.

Whilst there is very little literature concerning the precision of orientation determination, a review of the relevant literature on the problem of object recognition/classification is now presented.

In order to achieve object recognition and orientation determination, a method of describing the shape of the object in the image is required. An object in the image is recognised and its orientation determined by comparing its shape description to stored model shape descriptions.

The earliest methods used binary images and simple measures of shape such as area, length of perimeter, maximum internal circle, minimum external circle and ratios of these.

However, identical shape descriptors can be produced by widely differing shapes, and in order to distinguish between similar objects it is necessary to store a large number of such values. The presence of noise in an image can affect these quantities quite significantly and for partially occluded shapes the change can be quite dramatic.

More sophisticated global descriptors of 2-D shape have been used to recognise 3-D objects and determine their orientation. Typically, in these approaches, for each object a set of shape descriptors is computed for each possible viewing angle. (This gives any system view independence at the expense of increased storage requirements). Then, given an arbitrary view of a known object, or one of a small set of objects, a set of shape descriptors is computed and matched against the library, or database, of precomputed model descriptors.

Two methods of this type which have been popular amongst researchers are moment-based methods[1] and Fourier descriptors[2,3,4] in which the boundary of an object is represented by the coefficients of a Fourier transform of some function around the boundary.

Dudani et al[5] use moment invariants extracted from television images to recognise a test set of images comprising 6 different types of aircraft. Classification is performed using a distance-weighted *k*-nearest neighbour rule, giving a success rate of the order of 95%. No detailed results for orientation determination are given, even though solving this problem is an inherent part of the matching to the database. Average errors of between 5 and 10 degrees are quoted. Similar results are shown by Reeves et al[6] and results for the problem of ship identification are given by Smith and Wright[7]. Where range data is available 3-D moments can be defined[8]. The application of Fourier descriptors to the problem of aircraft recognition is discussed by Reeves et al[6] and Richards and Hemami[9] give an application to the recognition of 3-D objects.

Wallace and Wintz[10] use normalised Fourier descriptors to classify a group of 6 aircraft types with accuracies of 80% plus. They present a detailed analysis of the effects of chain-code representation of contours and use an interpolation technique between database entries such that the required density of 2-D projections is much reduced. However, again no results for the precision of orientation determination are given.

Such global descriptors of shape give good results for clean images and complete shapes. However, moments especially are particularly susceptible to noise and errors introduced by discretisation - a review of their performance in a discrete environment is given by Teh and Chin[11,12]. Also, in keeping with all transform techniques, methods such as these are unable to represent local information with any degree of accuracy[13] and degrade rapidly when part of an object is missing or occluded since distortion of an isolated region of a shape will lead to changes in all of the shape descriptors.

For these reasons there has been a move away from global to local descriptions of shape. Local features describe limited portions of a shape and are unaffected by other regions of the shape. Distortion of any section of the shape will thus affect only the local features associated with that section, leaving the rest of the shape descriptors corresponding to other parts of the shape unchanged. As such, local features are ideally suited to the problem of partial shape analysis.

Recent attempts have been made to use Fourier descriptors to describe local features. Lin and Chellapa[14] present results for 2-D contours representing the boundaries of 4 of the Great Lakes and for 8 different aircraft shapes divided into two classes. Matching is based on the Fourier descriptors of segments of the 2-D contours.The classification accuracies achieved are, respectively, 81.25% and 57.5% with 30% of the boundary missing.

Gorman et al[15] use Fourier descriptors of segments of the boundaries of shapes and a dynamic programming technique to compare these to similar model data. On a 6 aircraft experiment similar to that of Dudani et al[5] they are able to achieve classification accuracies of up to 51.67% when 50% of the boundary is missing. This should be compared to the 12.67% using Fourier descriptors of whole occluded boundaries. No results for orientation determination are given.

Wallace et al[16] use estimates of total angular change about points of maximum curvature as a function of arc length around the boundary to recognise 2-D silhouettes of aircraft. Recognition accuracies for complete shapes are similar to those obtained using Fourier descriptors. No orientation determination results are given and partially occluded shapes are not considered.

Most of the published literature involving partially occluded shapes follows an hypothesise and test type of approach in which groups of consistent object-model matching features are used to solve for some coordinate transform which maps features in a stored model representation onto the sensed object in the image. This involves two major assumptions. Firstly that there exists an iconic model of the object, i.e. the object can be represented by a model having undergone some coordinate transformation. Secondly that the object is rigid and will not deform under transformation. Most published papers describe variants in some form or other of the following process:-

(i) Extract a set of local features from the sensed object and from the stored model.

(ii) Match pairs of object-model features.

(iii) Extract a consistent set of matching pairs based on known object constraints.

(iv) Use the set to determine a coordinate transform that maps the model onto the sensed object. This constitutes an hypothesis on the type and pose (position and orientation) of the object in the image.

(v) Use the transformation derived in step (iv) to evaluate some measure of how well model features map onto object features. The result serves to either accept or reject the hypothesis.

Steps (ii) and (iii) together follow one of two approaches. In the first, initially ALL matching object-model feature pairs satisfying the required constraints are considered and incompatible matches are subsequently removed from consideration, leaving clusters of compatible matches. In the second, the process is begun with the minimum amount of data that is required, say a single matching object-model feature pair, and a cluster is grown around this centre as subsequent matching pairs are considered in turn. This follows the approach first suggested by Fischler and Bolles[17]; the RANSAC paradigm.

Step (iii) either adopts a graph-theoretic approach[18] or uses a variant of the generalised Hough transform[19]. In the former approach the set of matches is transformed into an association graph where each node represents a match and arcs connect structurally compatible matches. The largest completely connected sub-graph, or maximal clique, represents the largest set of compatible matches which are then used to derive an hypothesis of the object in the scene. In the second approach all compatible pairs of object-model features are used to derive candidate coordinate transformations. The parameters of these transformations index a Hough transform space. Clusters of points in the Hough space indicate consistent sets of matching features and thus hypotheses.

Alternatively, relaxation algorithms[20] can be used. These involve assigning sets of compatible model labels to each sensed data item. Next, a constraint checking phase is initiated in which labels are removed when some relation is no longer preserved by an interpretation of the scene. This process is iterative and a solution is achieved when enough sensed features have interpretations in the model. An example of the use of this approach in the matching of 2-D contours is given by Davis[21].

Bolles and Cain[22] use simple local features e.g. notches, corners and holes with graph-theoretic methods to extract sets of structurally compatible matches. Structural difference in both the object and model is measured by the difference in orientation and distance between two features. These orientation differences and distances in the object and model are compared to give an estimate of the structure between two matches. Extensive analysis of the model is used to determine 'focus features' and nearby 'local features' in order to control the combinatoric explosion. The matches are represented by an association graph whose maximal clique represents an hypothesis of the object in the scene.

Ayache[23] uses polygons to represent 2-D objects and uses fragments of these as features. The longest fragments are weighted more heavily and are used to determine initial hypotheses when matched to polygon fragments in the sensed image. These comprise a coordinate transformation that takes the model onto the sensed object and a 'quality score'. Matching other model segments updates this score and coordinate transformation. The score is eventually used to determine how well the transformed model corresponds to the sensed object. Hypotheses with low scores are removed from consideration.

Koch and Kashyap[24] use polygons and polygon fragments similarly to Ayache. They consider planar objects which are constrained to lie on a plane parallel to the plane of the image. The image may contain several different objects and objects may overlap one another. Fragments are matched to one another by computing a similarity measure and a coordinate transform that takes a fragment in the model onto a fragment in the scene. The similarity measure is used to decide which hypotheses to try first; these are the most visible objects. The polygonal representation is used to determine what the scene would look like if the hypothesis was incorrect, i.e. the object was absent from the scene. A positive result from this test allows the removal of matching scene features. A negative result removes the hypothesis from further consideration. The extraction of sets of compatible matches is formulated as an association graph. In order to find the maximal clique the RANSAC paradigm is adopted, starting with a single match and growing a cluster around this match. The result gives an hypothesis comprising a match as a cluster centre and a list of matches compatible to this centre. The procedure is sub-optimal in the sense that it does not guarantee the finding of the maximal clique, but there are rewards in the form of decreased time complexity.

Grimson and Lozano-Perez[25] structure the search for compatible matches as the generation and exploration of an interpretation tree.The tree is constructed in a depth-first fashion, beginning at the root node. At the first level the first sensed feature is assigned to all model features, at the second level the second sensed feature is assigned to all model features, and so on. Local geometric constraints are used to remove entire sub-trees from consideration and a 'quality measure' guides the search. The search terminates when an acceptable solution has been found.

This process is adopted by Van Hove[26] who presents a unique approach to 3-D object recognition using 2-D silhouettes. He assumes orthographic projection and describes his silhouettes by a series of straight line segments fitted to the boundary. The interpretation tree is pruned by simple constraints on the relative positions of pairs of edges. This results in a solution which is equivalent to a search through the whole problem space without actually instantiating the complete tree.

Stockman et al [27] use a Hough transform type of approach to register aerial imagery with maps, to detect objects in aerial imagery and to inspect 2-D machine parts. They define features as directed edge segments, or vectors, connecting point features. All model vectors are matched to all image vectors to produce many estimates of 2-D coordinate transforms. The data defining such transformations determine points in a Hough transform space. Clusters of points in this space indicate consistent sets of matching features. The transform space is quantised into a series of overlapping hierarchical bins. This pose-clustering approach may be thought of as the inverse of template matching[28]. This approach is extended by Stockman[29] to cover the case of 3-D data and 3-D models. The problem of 2-D data and 3-D models is covered in[30] where the authors solve for inverse perspective, combining the data from several different cameras to form a 'cluster-space stereo'. The objects are assumed to have few stable states and are constrained to lie on a flat plane.

Ballard[31] decouples his transform space into separate rotation angle and translation spaces. He first locates clusters in the rotation space and then in the translation space in order to extract groups of consistent matches.

Thompson and Mundy[32] use vertex pairs in the image and model to solve for an affine approximation to perspective and search for clusters in the resulting 6-D Hough transform space. The space is decoupled into separate rotation and translation-scale spaces. The rotation space is further decoupled into a rotation in the plane of the image and into a rotation space about the other two axes. A binning approach is adopted first in this latter space and then in the in-plane rotation space. The results are then used in a classical nearest-neighbour clustering algorithm in the translation-scale space. They show results for real outdoor scenes containing complete objects which are modelled as polyhedra.

Turney et al[33] give results for untilted 2-D parts at a fixed distance from the camera. The parts may be occluded. Matching is performed using segments of the boundary. Model features are assigned weights and pairs of segments that uniquely recognise and determine the pose of a part are found. These are termed salient configurations. Segments are compared in a slope angle versus arc length representation, using only the salient configurations.

Dhome and Kasvand[34] use range data comprising simple polyhedra constrained to lie on a horizontal flat surface. They use local patterns in the model and scene which have the same geometric characteristics to solve for a 6-D coordinate transform. The 6 parameters are not computed directly but for each possible match 3 parameters which characterise the model are determined. These are the model view axis, the model orientation about that axis and the position of the model centre. Clustering is then performed on these three parameters in turn. The method can cope with multiple objects, in which case each object, or group of objects, is examined separately and each model applied to it.

Linnainmaa et al[35] use the vertices of polyhedral objects to form junction triangles in both the sensed object and the model. A set of constraints are used to control the number of triangles formed. These triangles are then used to solve for the perspective transform that maps the model onto the image. Each model-image triangle pair admits four, or most likely two, solutions to the 6-D space. This space is decoupled into two projections comprising rotations and translations only. The search for clusters is conducted in the translation space and rotation angles are computed from these results. Results are shown for both complete and partially hidden polyhedral objects; in the latter case the cluster size is markedly reduced in relation to false clusters. They suggest that as long as junction triangles can be formed that the method need not be confined to strictly polyhedral objects.

The approach suggested by Horaud[36] is similar to the maximal clique finding of previously mentioned work[25,26]. The main difference is that rather than predicting model appearances in the image, all the model-image matching is performed directly in 3-D space itself. 2-D lines in the image are grouped into angles and junctions and these attributes are back-projected into the 3-D model space using the constraints of perspective. The back-projected image features are put into correspondence with model features. This is implemented as a hypothesise and test approach which is structured as a depth-first tree search. The search is directed by measures of how well the position and orientation of the object is constrained by any match or group of matches.

An iterative solution to the problem which can cope with partially occluded objects, is suggested by Lowe[37,38]. He utilises powerful perceptual constraints such as collinearity and proximity relations to form groupings of structures that are likely to be invariant over large ranges of viewpoint. The size of the search space is controlled by a probabilistic ranking of features and a numerical iterative technique is used to solve for the coordinate transform.

Many of the published works described present attractive solutions assuming a polyhedral blocks world and perspective transform. Whilst this might be a suitable model for the human visual system and be able to represent many everyday objects, e.g. a chair, a table, a car etc. these approaches are clearly not suitable for the given problem as outlined in section 1.1.

None of the papers address the problem of how accurately orientation may be determined, even though solving for the pose of objects in sensed images forms the basis of the described methods.

The work presented in this thesis remedies this situation, analysing the performance of two of the more traditional global methods, namely geometric moment invariants and Fourier descriptors, both for clean, unoccluded data as well as in conditions of noise and occlusion. The nature of their failure as objects become partially hidden is also analysed.

One approach to describing the shape of plane curves is that of locating the critical points, or points of maximum curvature, on object boundaries. Several different models for this method of describing local shape have been presented[47,48,49,50], but despite their popularity with many researchers[16,24,50,51,52], primarily as a result of experiments lending psychological support to the perceptual significance of such points[46], results presented in the thesis will show that their performance becomes unreliable in the presence of noise on boundary curves.

This thesis presents a novel approach to the recognition and orientation determination of 3-D objects from single 2-D silhouettes for the case of orthogonal projection. The only other work, to the authors knowledge, which has been conducted on this specific topic is that of Van Hove[26]. His work is limited to simple polyhedral shapes, or curves that can be approximated by polyhedra, and it is doubtful whether the method of directly mapping 3-D model edges onto 2-D image edges would work for more complex objects. For example, for the aircraft and missile models used here it is a non-trivial matter to determine, for any particular viewing angle, which edges will be completely visible, which edges will be totally hidden and which edges are partially occluded by other edges and in what fashion this occlusion occurs.

Wire-frame models of a Hawk jet trainer, a Concorde and a missile are used to generate both model and test data throughout this study. The objects are described in terms of local features of the boundary. A series of straight line segments of varying length are fitted to the boundary; the end points of these line segments form point features. Each wire frame model in turn is rotated through a discrete number of different viewing angles and the point features extracted. This data is stored as a library or database in a similar fashion to previous work performed on moments and Fourier descriptors. For any fixed viewing angle the only remaining degree of rotational freedom is in the plane of the image. Under these conditions the orientation determination problem is reduced to the 2-D case. Then, given an image of one of these objects in some unknown orientation it is possible to perform a match of the sensed features with the features stored in the database. This is done using a 2-D pose-clustering procedure similar to that adopted by Stockman et al[27]. For some viewing angles the match will be poor, but for viewing angles close to those corresponding to the true orientation the match will be good. The solution to the orientation determination problem is then one of finding the optimum match with the database.

This follows an hypothesise and test type of approach in which simple measures of shape are used to select areas of the database for matching. This restricts the search to less than 20% of the whole database. Each selected database entry can be considered as representing a hypothesised pose of the object in the image. The parameters of the coordinate transform are then derived, for each selected entry, using a 2-D pose-clustering procedure. The problem of combinatoric explosion (which is common to all of these approaches) is alleviated by the extensive use of constraints and features of the object such that for *m* model features and *n* image features the *mn* possible combinations are reduced by about 90%. Each hypothesised pose is then verified by applying the derived transform to the model data and comparing the result to the image. A root mean squared, (r.m.s.), error of fit measure is computed, and after matching with each selected entry the smallest r.m.s. error gives the orientation of the object. The view angle is given by the view angle of the best match and orientation in the plane of the image falls out of the matching process. Extensive use of heuristics means that only one third of the selected database entries reach this stage of hypothesis verification and of these only about 15% are totally compared. This represents a total of as little as 5% of the selected database entries, or 1% of the complete database.

Object recognition is achieved when the data for several different objects is combined into a single multi-object database. In this case the best match both recognises the object and determines its orientation.

Using this method it is possible to recognise an object and to determine its orientation to within less than 7 degrees of arc, on average, even when boundaries are noisy and with up to 40% missing or occluded[62].

This combination of a 2-D pose-clustering procedure with a directed search of a database represents a novel approach to the problem of 3-D object recognition and orientation determination from single 2-D images.

CHAPTER TWO

Hardware and Software Tools

2.1 Equipment Used

All of the images are analysed using a PC compatible fitted with an Imaging Technology Inc. FG-100-AT board which provides a single framestore of 1024 * 1024 pixels each 12 bits deep. A single image occupies 512 * 512 pixels on the image plane. Supplied with the board is a library of 'C' routines which allow the direct capture of images from a video camera or from a video recorder and which also give access to the image plane and its contents. All of the code is written in 'C'; it is however machine dependent to the extent that it uses specialised software to access image data.

The available internal memory of the machine is 640k. Unused portions of the image plane are used as additional RAM, in particular to store the database.

Where real images have been used to test the system they have been captured using a CCD camera; these have very good geometric accuracy and are linear in their response to radiance[59]. Test objects are painted black and are mounted against a white background. This comprises a board with a small hole drilled through the centre; the objects are fixed to the board via this. The horizontal axis passing through this hole is marked and calibration marks at 15 degree intervals are drawn through the complete 360 degree circle. The board is clamped to a bench and is made square using a spirit level. The camera is aligned using a spirit level. A mirror is mounted initially at the centre of the board. When an image of the camera can be seen at the centre of the image plane it is assumed that the optical axis of the camera and the centre line through the mounting hole on the board are coincident. Objects are fixed to the board using a series of right angled wedges providing angles of 30, 60 and 45 degrees. These are also painted white and are fixed to the board in such a fashion as to allow the wedges to be rotated into alignment with the calibration marks on the board. In this fashion the true orientation of test objects can be determined to within less than two degrees.

2.2 Definition of Orientation

Orientation is defined in figure 1. The origin of the system is at some notional point within the object, e.g. the centre of mass. The *z*-axis is parallel to the line of sight with positive pointing away from the eye. The *x*-axis is coincident with the longitudinal axis of the object and the *y*-axis is perpendicular to both of these, forming a left-handed orthogonal set. Any orientation may be described in terms of a sequence of three rotation angles; firstly rotate counter-clockwise about the *y*-axis, then rotate counter-clockwise about the new rotated *x*-axis and finally, rotate counter-clockwise about the line of sight, the *z*-axis. For the present purposes these rotations are termed yaw, roll and pitch respectively.

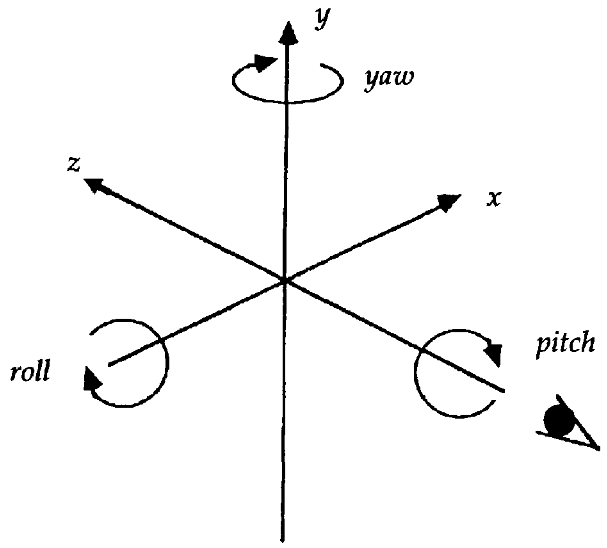


Figure 1. The coordinate system.

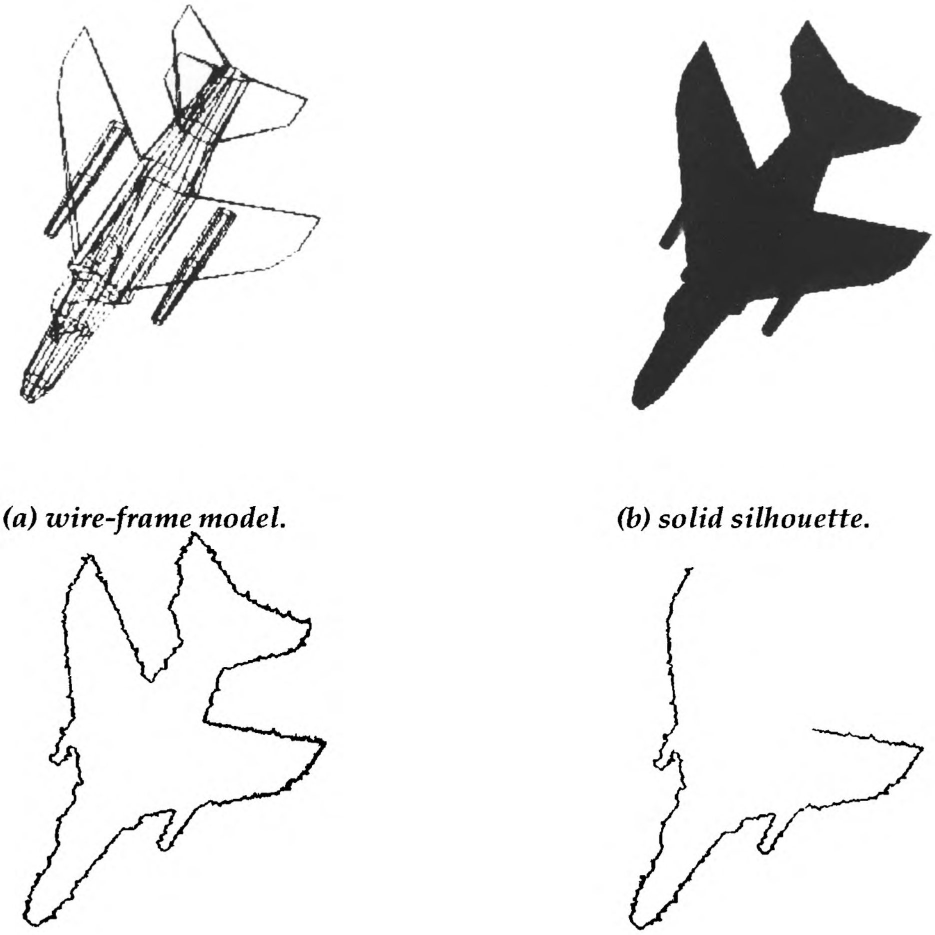
2.3 Objects Used and Image Generation

Models of a missile, a Hawk jet trainer and a Concorde are used in this study, both to generate the model data (the database) and also the test data for the system. The models are corrupted by artificially added noise and are also partially occluded. These processes are described below. Later, (chapter 9), real images of these objects are processed.

Simple wire-frame representations of the objects are used. The data for the missile was supplied by the M.O.D. whilst data for the Hawk jet and the Concorde was abstracted from plastic Airfix-type models. All coordinates are defined with respect to the origin of the coordinate system described in section 2.2.

Different views of the objects are generated by applying the described sequence of rotations to the *x,y,z* coordinates of the wire-frame model. This is achieved using standard 3-D rotation matrices [39]. Then, assuming orthographic projection (i.e. viewpoint at infinity) the resulting *x,y* coordinates are then simply 'dropped' onto the image plane without distortion, the *z* coordinates having been projected away. An example of the wire-frame model of the Hawk is shown in figure 2(a). The wire-frame representation is then converted into a solid silhouette (figure 2(b)) and its boundary is traced[40] producing a closed contour represented by a sequence of 8-connected pixels.

Noisy boundaries such as would result from poor segmentation of noisy images, are simulated by the artificial addition of noise to these boundaries. The noise is added to the original integer coordinate list prior to preprocessing following an algorithm which has been used by the industrial sponsors of this project[41]. It is usual practise to add Gaussian noise to an image. However, this is not quite so straightforward when it is required to perturb pixels on a simple closed contour. For example, there are problems with maintaining the connectivity of the curve. For this reason the algorithm is as described, even though it makes the statistics more difficult to interpret.



(a) wire-frame model.

(b) solid silhouette.

(c) boundary with added noise.

(d) occluded boundary.

Figure 2. Image generation.

Single pixels on the boundary are perturbed inwards or outwards. The amount of noise added is controlled by two input parameters, *c1* and *c2*. For each point on the boundary curve a uniform random number, *nrand,* is generated in the range [0,1]. The following test is then applied in order to determine the amount by which the pixel should be perturbed, if at all.

```
If (0 ≤ nrand < c1)
    perturb pixel outwards by 2
else if (c1 ≤ nrand < c2)
    perturb pixel outwards by 1
else if (1-c1 ≤ nrand < 1)
    perturb pixel inwards by 2
else if (1-c2 ≤ nrand < 1-c1)
    perturb pixel inwards by 1
else
    pixel is not perturbed
end if
```

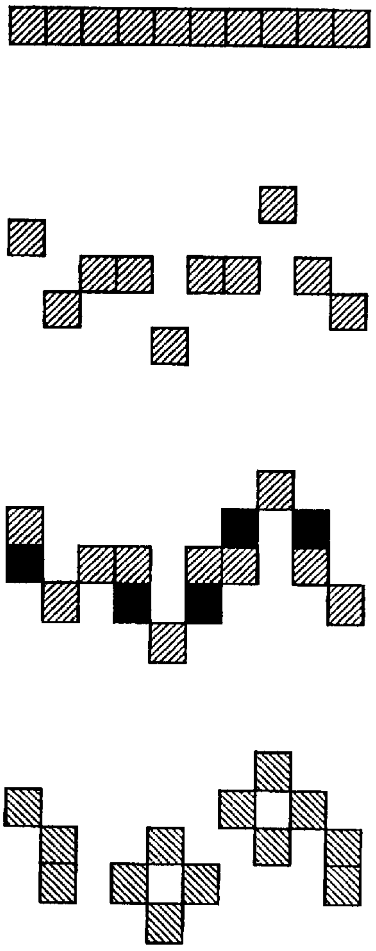
The coefficients *c1* and *c2* must satisfy the following constraints

$$c1 \leq c2 \leq 0.5$$

and

$$(c2 - c1) \geq c1.$$

Obviously, it would be possible to disturb pixels by distances greater than this but perturbations are constrained as described in order to restrict the resultant problems of maintaining the connectivity of the boundary. The fact that connectivity needs to be maintained means that whilst the coefficients *c1,c2* represent the percentage of pixels physically moved by the noise addition algorithm, after reconnection the percentage of boundary pixels which are different from the original unperturbed boundary will be much higher than this figure. This is illustrated in figure 3. Consider the portion of boundary ten pixels long shown in figure 3(a). Noise is to be applied with coefficients *c1* = 0.1 and *c2* = 0.25. This means that 2*(*c2-c1*) = 0.3 of the pixels are to be perturbed by ±1 pixel and 2*c1* = 0.2 of the pixels are to be perturbed by ±2 pixels (see figure 3(b)). The resulting unconnected boundary is then reconnected, producing a boundary section whose length has increased to fifteen pixels (see figure 3(c)). If the number of pixels which are different in the two images shown in figures 3(a) and 3(c) are now found, (this is done by taking the exclusive or of the two images, assuming they are binary valued), then this value can be expressed as a fraction, and hence percentage, of the original boundary length, in pixels. For the example shown, figure 3(d), there are thirteen pixels resulting from the exclusive



(a) original boundary.

(b) boundary with 0.5 of the pixels perturbed.

(c) reconnected boundary.

(d) exclusive or of figures (a) and (c).

Figure 3. Adding noise to the boundary pixels.

or of figures 3(a) and 3(c). This results in a percentage difference of 130%. Wherever %noise or noise levels are mentioned they are really these values of the percentage of pixels which are different in the resulting noisy boundary when compared to the original, undisturbed boundary. Table 1 shows some values of *c1* and *c2* with mean applied noise levels for the objects used in this study. The values are averages because pixel perturbation distances are fixed absolute values whilst the length of projected boundary changes as object orientation changes.

c1	*c2*	% noise
0.0005	0.01	9
0.01	0.02	16
0.02	0.04	25
0.03	0.06	34
0.04	0.08	46
0.045	0.09	49
0.05	0.1	52
0.06	0.12	60
0.08	0.16	84
0.1	0.2	89
0.1	0.25	96
0.0	0.25	106
0.0	0.35	126
0.0	0.4	139
0.1	0.4	147

Table 1. Noise coefficients c1, c2 versus % noise.

As an illustration, figure 2(c) shows the noisy boundary of the Hawk, in this case with 60% of the boundary pixels disturbed.

Partially occluded shapes are generated by simply masking off portions of the 2-D silhouette and then tracing the boundary, adding noise if required. This produces a closed curve. If it is not important that a curve is closed then boundary pixels may simply be masked off to the required degree of occlusion (see figure 2(d)).

CHAPTER THREE

Representing Shape 1 - Global Shape Descriptors

This chapter introduces the basic theory involved in two global methods of describing the shape of objects. These are moment invariants and Fourier descriptors. Also described are the details of experimental investigations which have been conducted into the suitability of these techniques for the problem described in section 1.1. Presentation and discussion of these results is deferred to chapters 9 and 10 for aesthetic reasons but a statement is made with regard to their performance.

3.1 Moment-based Methods

The first paper on the use of moment invariants for the analysis of 2-D image data was presented by Hu[1]. His approach was based on algebraic invariants, using nonlinear combinations of geometric moments to derive a set of moment invariants which have the properties of being invariant under RST(Rotation,Scaling,Translation) transformations.

Since then several other means of deriving moment invariants have been suggested. Teague[42] introduced the notion of orthogonal moments based on the theory of orthogonal polynomials, allowing the construction of invariants to any order with relative ease. Examples are Zernike moments and Legendre moments which use Zernike and Legendre polynomials, respectively. Complex moment invariants have also been suggested[43,44].

Moments are defined analytically and so are useful as a means of describing shape if they are insensitive to the presence of image noise e.g. sampling, quantisation, discretisation. Theoretic analyses of the performance of the various types are given by Teh and Chin[11,12]. An experimental study of the invariance of geometric moments has been conducted. This method was chosen as being illustrative of the family of moment-based techniques since it is the simplest to implement; a comprehensive survey of the performance of moment-based methods was never the intention of this research. All of the other methods suffer from the same pitfalls and will behave in a similar fashion, even if their theoretical performance might be better than this simple technique. Geometric moments may be defined as follows.

Given a continuous 2-D density function, $f(x,y)$, the geometric moments of order $(p+q)$ are given by [1]

$$M_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x,y)\, dx\, dy \qquad (1)$$

for $p,q = 0,1,2\ldots$

For a binary, digital image the double integral reduces to a double summation over non-zero pixels of the now finite image area

$$M_{pq} = \sum_x \sum_y x^p y^q \qquad (2)$$

for $p,q = 0,1,2\ldots$

The centre of mass, or centroid, $\bar{x}, \bar{y}$, is defined

$$\bar{x} = \frac{M_{10}}{M_{00}} \qquad (3)$$

$$\bar{y} = \frac{M_{01}}{M_{00}} \qquad (4)$$

where M_{00} is the object area.

The origin of the system is then redefined to be the centroid, producing moments, μ_{pq}, which are invariant under translation

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q \qquad (5)$$

for $p,q = 0,1,2...$

The moments up to second order approximate the original image to an ellipse of definite size, eccentricity and orientation (figure 4). The lengths of the semi-major and semi-minor axes are [42]

$$a = \left[\mu_{20} + \mu_{02} + \frac{((\mu_{20}-\mu_{02})^2 + 4\,\mu_{11}{}^2)^{½}}{\mu_{00}/2} \right]^{½} \qquad (6)$$

$$b = \left[\mu_{20} + \mu_{02} - \frac{((\mu_{20}-\mu_{02})^2 + 4\,\mu_{11}{}^2)^{½}}{\mu_{00}/2} \right]^{½} . \qquad (7)$$

The orientation of the ellipse is given by

$$\gamma = \frac{1}{2}\tan^{-1}\left[\frac{2\mu_{11}}{(\mu_{20}-\mu_{02})}\right]. \qquad (8)$$

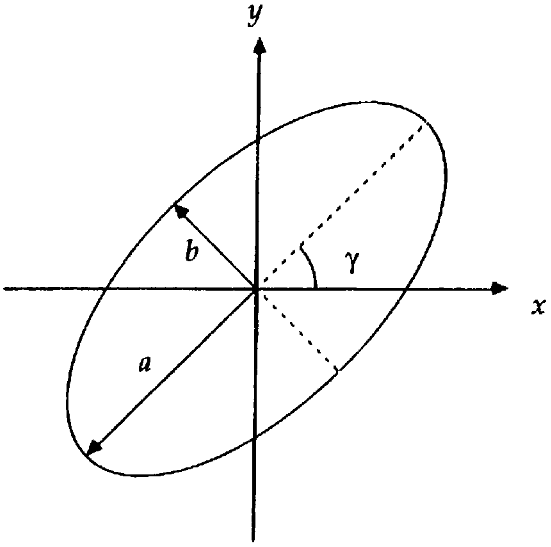


Figure 4. The orientation of the principal axes.

This angle defines a set of orthogonal axes whose orientation defines the orientation of an object in the plane of the image. The semi-major axis of the ellipse is called the principal axis.

The following normalisation is then applied to achieve scale invariance

$$\lambda_{pq} = \frac{\mu_{pq}}{M_{00}{}^{0.5\,(p+q)+1}} \,. \qquad (9)$$

Hu[1] combined the second and third order λ_{pq} to give seven invariant moments, φ_i, for i =1,7; these are invariant under RST transformations.

$$\varphi_1 = \lambda_{20} + \lambda_{02} \qquad (10)$$

$$\varphi_2 = (\lambda_{20} - \lambda_{02})^2 + 4\lambda_{11}{}^2 \qquad (11)$$

$$\varphi_3 = (\lambda_{30} - 3\lambda_{12})^2 + (3\lambda_{21} - \lambda_{03})^2 \qquad (12)$$

$$\varphi_4 = (\lambda_{30} + \lambda_{12})^2 + (\lambda_{21} + \lambda_{03})^2 \qquad (13)$$

$$\varphi_5 = \begin{matrix} (\lambda_{30} - 3\lambda_{12})(\lambda_{30} + \lambda_{12})[(\lambda_{30} + \lambda_{12})^2 - 3(\lambda_{21} + \lambda_{03})^2] + \\ (3\lambda_{21} - \lambda_{03})(\lambda_{21} + \lambda_{03})[3(\lambda_{30} + \lambda_{12})^2 - (\lambda_{21} + \lambda_{03})^2] \end{matrix} \qquad (14)$$

$$\varphi_6 = (\lambda_{20} - \lambda_{02})[(\lambda_{30} + \lambda_{12})^2 - (\lambda_{21} + \lambda_{03})^2] + 4\lambda_{11}(\lambda_{30} + \lambda_{12})(\lambda_{21} + \lambda_{03}) \qquad (15)$$

$$\varphi_7 = \begin{matrix} (3\lambda_{21} - \lambda_{03})(\lambda_{30} + \lambda_{12})[(\lambda_{30} + \lambda_{12})^2 - 3(\lambda_{21} + \lambda_{03})^2] - \\ (\lambda_{30} - 3\lambda_{12})(\lambda_{21} + \lambda_{03})[3(\lambda_{30} + \lambda_{12})^2 - (\lambda_{21} + \lambda_{03})^2] \end{matrix} \,. \qquad (16)$$

A series of experiments were conducted using simple planar geometric shapes cut from black card and mounted on a white background. The amount by which each of the 7 invariants, given by equations (10) - (16), change under translation, scale change and rotation was investigated by capturing a series of images of the shapes via a video camera.

The images were thresholded[45], binarised and the moment invariants computed. Table 2 shows the mean percentage change in each invariant for these transformations.

moment invariant	Mean % change		
	translation	rotation	scale change
ϕ_1	0.4	1.0	0.1
ϕ_2	2.1	7.3	0.3
ϕ_3	2.0	2.2	0.2
ϕ_4	2.8	6.2	1.2
ϕ_5	3.8	10.0	3.1
ϕ_6	1.8	9.6	3.0
ϕ_7	6.8	13.0	1.4

Table 2. Mean % change in moment invariants under RST transformation.

It can be seen that changes of up to 3% for scale change, up to 7% for translation and up to 13% for rotation result. This is for relatively noise-free conditions and does not augur well for their effective use as a means of describing shape in noisy conditions, or indeed when part of a shape is missing or occluded. With these figures in mind it is not likely that geometric moment invariants will provide a very successful solution to the problem posed.

In order to test how well the method might be able to solve the problem outlined, the wire-frame models were rotated through 5 degree increments in yaw and roll and solid silhouettes generated as described in section 2.3. At each orientation the seven moment invariants were computed and written, as a single record, to a file along with the yaw, roll and principal axis of the object.

This file constitutes the database. Three databases were generated, one for each of the three objects. The system was then tested by presenting it with many different views of the object and computing the moment invariants and matching to the database using a simple Euclidian metric, M_e, given by

$$M_e = \sum_{i=1}^{7} (\varphi_{ri} - \varphi_{si})^2 \qquad (17)$$

where φ_{ri} are the reference moment invariants stored in the database and φ_{si} are the sensed moment invariants.

No sophisticated weighting techniques were used and matching to the database was performed using a global search and comparison. The database entry, M_e, with the smallest metric provides the orientation estimate in yaw and roll. Pitch estimates were obtained from the difference in the orientation of the principal axes of sensed and

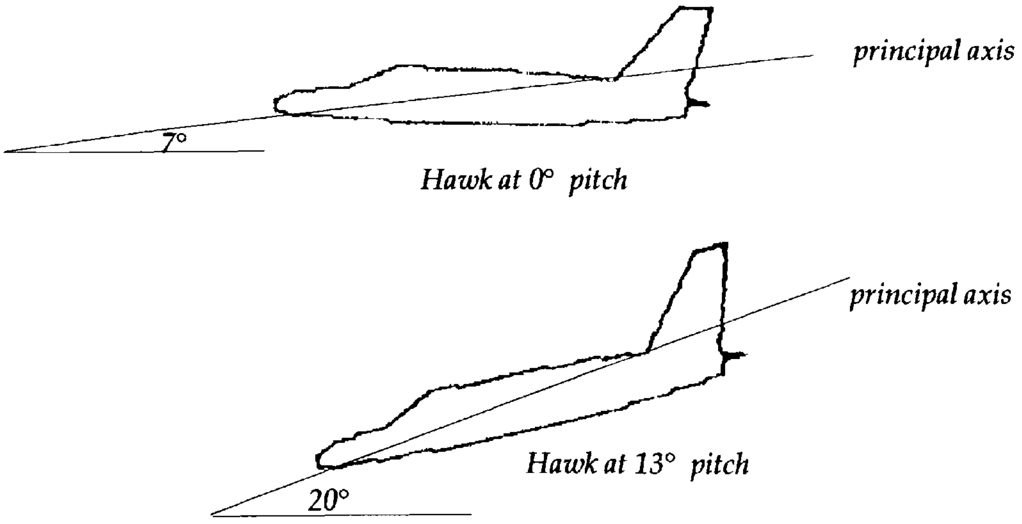


Figure 5. Pitch is determined from the difference in the principal axes of the sensed and model objects.

database objects. Despite all database objects being stored at zero pitch it was necessary to store principal axis and compute pitch estimates in this fashion since only if the current view of the object exhibits symmetry about the longitudinal axis will principal axis represent an accurate estimate of object pitch (see figure 5).

Chapter 9 presents results of an experimental investigation using the models of the missile, Hawk jet and Concorde. The results will show that the method produces reasonable results for relatively noise-free conditions, degrades as levels of noise increase and falls down quite dramatically as objects become occluded.

3.2 Fourier Descriptors

Another popular method of describing the global shape of 2-D objects is that of Fourier descriptors[2,3,4]. The method involves expanding some function of arc length around the boundary in a Fourier series. The coefficients of this Fourier series form the shape descriptors.

Such methods have the advantage that they are backed up by well, and easily, understood mathematical theory. Typically, a 256-point Fourier transform might be used to compute the Fourier series, and only the lowest two or three dozen or so coefficients might be needed to accurately describe the object shape. The actual number of coefficients required will obviously depend upon the nature of the curve in question; rapidly curving boundaries with larger proportions of high frequency components will obviously require more coefficients than gently curving shapes. Thus, since noise on the boundary is predominantly high frequency, the process is inherently noise-removing. For this reason it might be expected that Fourier descriptors be more robust in the presence of noise.

An experimental investigation was conducted into one particular implementation, due initially to Granlund[3].

Consider a plane curve lying in the complex plane. Then, the sequence of boundary coordinates yield a series of complex numbers. The discrete Fourier transform of this series constitutes the Fourier descriptor of the contour. Various operations can be performed on this sequence to produce a description which is invariant to scale changes, translation, changes in orientation in the plane and boundary tracking start point.

Databases were generated for the three model objects using the wire-frame models in a similar fashion to that for moment invariants. At each orientation the boundary curve was resampled to give 256 points equally spaced in arc length. A 256-point fast Fourier transform was used to produce the Fourier descriptor of the curve, and of these, only the lowest 32 frequency coefficients, $C(i), i=1,32$, were used in matching; this was found to be adequate for the objects used in this study. Normalisation of the coefficients followed the procedure given by Wallace and Wintz[10].

To normalise for position the coefficient $C(0)$ is set equal to zero. Scale normalisation is achieved by dividing each coefficient by the magnitude of the fundamental frequency, or the largest coefficient. This will be coefficient $C(1)$, for a closed curve traced in the counter-clockwise direction. In order to normalise for rotation angle, each coefficient is multiplied by e^{jr}, where r is the rotation angle of the contour. A shift in the start point of the curve can be nornalised for by multiplying the i^{th} frequency coefficient by e^{jiT}, where T is the fraction of a period through which the start point is moved. Normalising for these two operations requires that the phases of the two largest coefficients be zero. $C(1)$ is the coefficient with largest magnitude. Then, if the coefficient of second largest magnitude is $C(i)$ then there exist $|i-1|$ possible rotation angle-start point combinations which will satisfy this condition. If $i=2$ then the orientation and start point of the contour are defined uniquely. However, in general, this will not be the case and it is necessary to examine each possible normalisation in order to resolve this ambiguity. This is achieved by computing the sum of the positive real energy, ξ, over the $|i-1|$ possible normalisations, where, for an N-point transform

$$\xi = \sum_{i=1}^{N-1} \mathrm{Re}[C(i)] \; |\, \mathrm{Re}[C(i)] \,| \; . \qquad (18)$$

The normalisation which results in the largest value of ξ is the one finally selected.

The sensed, normalised coefficients were compared to every entry in the database using a simple Euclidian metric in a similar fashion to that of moment invariants. Matching was based on magnitudes only when the metric, M_m, had the form

$$M_m = \sum_{i=1}^{32} (\, |C_{ri}| - |C_{si}| \,)^2 \qquad (19)$$

and also on the use of phase information when the metric, M_p, had the form

$$M_p = \sum_{i=1}^{32} (\, C_{ri} - C_{si})^2 \qquad (20)$$

and the result is a complex number. C_{ri} and C_{si} represent the coefficients of the reference and sensed Fourier descriptors respectively. When only magnitudes of coefficients are used it is necessary only to perform the first two normalisation steps described, i.e. normalise for position and scale.

The database entry with the smallest metric provides the estimate of yaw and roll. The lowest two frequency coefficients define an ellipse, in a similar fashion to second order moments. The orientation of this ellipse gives an estimate of object orientation in the plane of the image. Pitch estimates are computed from the difference in the orientation of these axes in the sensed and selected model database descriptions.

Two sets of experiments were conducted since it was thought that the phase parts would be the most susceptible to image noise. The normalisation process tends to reduce the information content of the phase components relative to the magnitudes anyway and it was desired to investigate how accurately orientation could be determined using them alone. In fact, with the missile especially (since this has an extra axis of symmetry when compared to the Concorde and Hawk), it will be necessary to include phase information in the shape descriptors if distinction between orientations which are reflections, for example, is to be achieved (refer to figure 34, section 10.1 for an example). The experimental results presented in chapter 9 will show how well such orientations can be distinguished as levels of noise increase.

3.3 Summary of Global Descriptors of Shape

The problem with global descriptors of shape such as moments and Fourier descriptors is that they rely on the whole shape being present. Even if part of a shape is missing then all of the descriptors will be affected. Moments are normalised with respect to the centroid of the object; any occlusion will affect the position of the centroid and thus the normalisation. Fourier descriptors suffer from similar problems. Thus, as results presented in Chapter 9 will show, such approaches are not suitable when objects are partially occluded.

Local methods of describing shape, where distortion of part of the object affects only the descriptors associated with that particular region and nowhere else, are more likely to provide a successful solution to the problem.

Recent attempts to use Fourier descriptors of segments of boundary curves to classify partially occluded contours[14,15] have only achieved mediocre success rates and involve considerably sophisticated analysis and matching methods. It is not likely that sufficient resolution will be available in these approaches in order to be able to resolve orientation to within anything like the required accuracy of ±5 degrees.

CHAPTER FOUR

Representing Shape 2 - On the Performance of Point Curvature Estimators

In order to describe a shape in terms of local boundary features a means of determining which parts of the boundary are the most significant is required. Points of maximum curvature are commonly believed to be the most perceptually significant points on digital curves[46]. As such, they have been used as local features in both 2-D and 3-D object recognition by many researchers. However, despite their popularity, it has been the authors experience that they are not robust in the presence of noise. This chapter presents a series of experimental results which will illustrate this argument.

4.1 Introducing Curvature

Typical objects which are considered here are man-made e.g. aircraft,missiles etc. and have well defined geometric outlines. A boundary curve can therefore be considered as a series of straight line segments separated by small regions where the curvature changes rapidly, even discontinuously. A sensible approach to representing the shape of such 2-D contours would therefore be to locate these points of maximum curvature and connect them by straight lines.

The importance of points of high curvature in the perception of the shape of outlines was first mentioned by Attneave[46] some 35 years ago. Since the publication of his psychological experimental results several researchers have presented models for the detection of angles and sides on digital curves e.g. Rosenberg[47], Rosenfeld and Johnston[48], Rosenfeld and Weszka[49]. Davis[50] developed methods leading to a hierarchical description of simple closed contours, again in terms of angles and sides, which represent the contour with varying degrees of refinement. He suggested that such descriptions could be used to produce natural polygonal approximations of boundaries.

The use of such point features as a means of shape description has increasingly been applied in the field of object recognition, both for the two-dimensional and three-dimensional problem e.g. Shirai[51], Wallace et al[16], Chien and Aggarwal[52] and Davis[50]. Due to their local nature, high curvature points have received special attention in the recognition of partially occluded objects where global features are generally rendered almost useless. The paper by Koch and Kashyap[24] is one example.

4.2 The Literature

In a real Euclidian plane, curvature, κ, is defined as the rate of change of slope as a function of arc length. For the curve $y = f(x)$ this may be expressed as

$$\kappa = \frac{\frac{d^2y}{dx^2}}{\left(1 + \left(\frac{dy}{dx}\right)^2\right)^{\frac{3}{2}}} . \qquad (21)$$

For the digital case it is not clear how to define an equivalent measure of slope; successive points on the curve can only differ in slope by multiples of $\pi/4$ (for an 8-connected boundary) so that small changes in slope are impossible. (This is part of the problem of attempting to compute such information directly from chain-coded data. Indeed, Wallace et al[16] convert chain-coded boundaries into lists of real-valued x,y coordinate pairs prior to any processing of the data.) The problem may be overcome by using approximations to the slope at a point, i, on the boundary e.g. $(y_{i+k} - y_{i-k}) / (x_{i+k} - x_{i-k})$ for some smoothing factor, $k > 1$. However, it is not obvious which value of k to choose in any given application. This expression becomes unstable as $(x_{i+k} - x_{i-k})$ approaches zero, therefore direct differences in this discrete analogue to slope cannot be used as estimators of point curvature. Any locally vertical segments of the boundary will produce infinite slopes. This quantity is therefore coordinate system dependent. For these reasons alternative measures of the change in slope are needed.

Possibilities include estimates of the cosine of the angle between the vectors joining the point, *i*, to points at distances of +*k* and -*k* away from *i*, of the angle itself between the same vectors or, in other words, the change in angle at point *i*.

Rosenberg[47], in a paper inspired by Langridge[53], presented an approach for the detection of dominant points on the boundary of a convex blob. He assigned a view, or domain, to every point on the boundary and then obtained measures of significance of each point relative to its domain. Several measures are suggested which, in general, correspond to the degree of 'curvedness' of the domain of a given point relative to the chord joining the end points of the domain. Domain sizes are altered, according to their significance measures, such that if the domain of a point, P_j, lies entirely within the domain of a point, P_i, then P_i is said to be dominant over P_j and the domain of P_j is removed. The final set of dominant points should have domains which constitute perceptually significant segments of the contour.

This approach cannot easily be extended to deal with concave boundaries; the observation that contours with concave parts may be decomposed into entirely convex sub-parts does not always hold. This point is mentioned by Davis[50]. Neither does the process define the dominant points of a contour in any strictly mathematical form.

One of the first models for points of high curvature on a digital curve was proposed by Rosenfeld and Johnston[48]. Let a closed curve, *O*, be represented by the sequence of points

$$O = (x_i, y_i) \qquad \text{for } 1 \leq i \leq n,$$

where there are *n* points on the curve such that $(x_1,y_1) = (x_{n+1},y_{n+1})$.

Define the vectors

$$\vec{a_{ik}} = (x_i - x_{i+k} , y_i - y_{i+k}) \qquad (22)$$

and

$$\vec{b_{ik}} = (x_i - x_{i-k} , y_i - y_{i-k}) \qquad (23)$$

for some smoothing factor, $k > 0$.

Then (see figure 6) compute the cosine, C_{ik}, of the angle between the two vectors from

$$C_{ik} = \frac{(\vec{a_{ik}} \bullet \vec{b_{ik}})}{| \vec{a_{ik}} | | \vec{b_{ik}} |}. \qquad (24)$$

These smoothed k-cosines contain the curvature information such that $-1 \leq C_{ik} \leq +1$ with $C_{ik} = -1$ for straight line segments and $C_{ik} = +1$ for the sharpest angle (zero degrees).

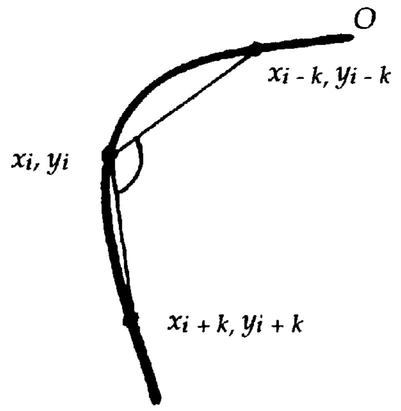


Figure 6. Computing the k-cosine.

In their model Rosenfeld and Johnston[48] propose a method whereby the smoothing factor, k, may be chosen. At each point (x_i, y_i) on the curve, O, compute, $C_{ik,}$ for a range of values of k such that $1 \leq k \leq m$, for some fixed m. A size, h, and smoothed k-cosine, C_{ih}, is assigned to each point such that

$$C_{im} < C_{im-1} < \dots < C_{ih} \geq C_{ih-1}.$$

There is a curvature maximum at point i if $C_{ih} \geq C_{jh}$ for all j such that $[\ i - j\ ] \leq {}^{h}\!/\!_{2}$. with $C_{i0} = -1$ to ensure that there will always exist such a value of h between 1 and m.

In 1975 Rosenfeld and Weszka[49] presented an improved version of the original model. This basically involved smoothing of the C_{ik} prior to the selection of the sizes , h.

Chien and Aggarwal[52] apply the point curvature estimator of Rosenfeld and Johnston[48] to the 3-D object recognition problem. They provide a means of automatically segmenting vertices into convex and concave types depending on the sign of the cross product of the vectors $\vec{a_{ik}}, \vec{b_{ik}}$. They also suggest a method whereby false feature points may be removed based on the premise that real features will have curvatures which tend to remain stable for small changes in the smoothing factor, k, whilst false feature points will have curvatures which change by much larger amounts under the same conditions. In the authors experience, this has been found not to be strictly true in practice.

The curvature estimator of Wallace et al[16] may be described as follows:-

Let $x(t)$ and $y(t)$ be the x and y coordinates representing the boundary, where t is a parameter representing the boundary length (time if the boundary is traced at a uniform velocity). Then change the $x(t), y(t)$ representation to an angle representation, $s(t)$, where

$$s(t) = \tan^{-1}\left(\frac{y'(t)}{x'(t)}\right) \qquad (25)$$

and $x'(t),y'(t)$ are approximations to the derivatives of the x and y coordinates with respect to time respectively; next compute the derivative of $s(t)$

$$s'(t) \approx s(t) - s(t-1). \qquad (26)$$

This contains the curvature information. Whilst the angle function, $s(t)$, is multiple-valued, under practical constraints on continuity, the derivative $s'(t)$ is uniquely defined. Finally locate the curvature maxima - these correspond to positive and negative peaks in $s'(t)$.

4.3 Experimental Analysis of Curvature Estimators

Very little or no information has been published on just how well models such as described in the previous section might be expected to perform under less than ideal imaging conditions. This information is obviously important if such models are to be used as a basis for a representation of shape.

In order to answer this point a series of experiments are performed using regular convex polygons of increasing order from 3 to 25. The aim is to discover which of a number of possible estimators of point curvature might be expected to have the best performance, and to determine, for each case, the smoothing factor, k, which allows the detection of the widest range of angles and side lengths. A series of fixed values for k is used in this investigation and no automatic determination of smoothing factors is attempted.

For each curvature measure and for various values of the smoothing factor, k, increasing amounts of noise are added to the boundaries of these polygons and the points of maximum curvature located. The boundary curve is then approximated by the polygon formed by connecting these points with straight lines. Estimates of the optimal values of k are obtained and the combinations of values of k and levels of applied noise at which the polygons are no longer recognised are observed. By recognised it is meant that the correct number of vertices, as given by the curvature maxima, can be located and that adjacent detected vertices should be approximately equidistant.

The results from these experiments indicate which estimator might be expected to be the most robust to a less than satisfactory choice of smoothing factor and which estimator is the least susceptible to the effects of noise on the boundary curve. Moreover, the study shows how the presence of noise can influence the choice of smoothing factor, or the region of support, of a point on the boundary and hence affect the detected locations of the dominant or high curvature points.

The estimators used are:-

(1) The smoothed *k*-cosine, $C_{ik,}$ as given by equation (24).

(2) The change in angle at a point, *i*, on the curve as given by the $s'(t)$ of Wallace et al[16].

(3) The change in angle at a point *i* on the curve as given by the following. Compute the angle that the vector, $\vec{a_{ik}}$ (equation (22)), makes with some reference axis, a_a, - say the horizontal. Next, compute the angle that the vector $-\vec{b_{ik}}$ (see equation (23)) makes with the same axis, a_b. The estimate of curvature at the point *i* is then given by $a_a - a_b$. (The angles are computed via inverse tangents).

Whilst it is appreciated that this study is limited in that it does not consider boundaries with concave parts it is thought that the results obtained can reasonably be used to indicate the likely range of angles and side lengths that might be detected. Later it is shown how they might be applied to models such as those suggested by Rosenfeld and Johnston[48] and Rosenfeld and Weszka[49]. The results are also applied to real objects with both convex and concave parts, and it is shown how well the detected points of curvature maxima represent the perception of the most significant points on a boundary.

A list of integer x,y coordinate pairs is generated corresponding to the boundary of the polygon in question. However, the discretisation noise is amplified in the curvature functions to such an extent that the following preprocessing step is necessary.The boundary is smoothed by applying a simple 3-point weighted average to the x and y coordinates individually. This produces a real-valued coordinate list. The boundary is then resampled to produce 256 points equally spaced in arc length. This step serves both to smooth out the discretisation noise in the boundary and to provide a scale-invariant representation. It also overcomes the problem of unequal sample spacings arising from the $\sqrt{2}$ factor involved in diagonally adjacent pixels.The curvature function is computed from these coordinates, and is then analysed to locate the curvature maxima.

Noise is then added to the coordinates (see section 4.3).

It has been found necessary to apply a filtering operation to each curvature function in a similar fashion to that applied to the boundary coordinates. This removes some of the noise still remaining after the preprocessing of the boundary and facilitates the location of curvature maxima. Operations such as these will have an adverse effect in that the locations of points of high curvature will tend to be smeared out over a small range of points which are close together leading to a loss of resolution. It is felt though, that the advantages of filtering (or rather the disadvantages of not filtering) far outweigh such drawbacks.

The process applied to the smoothed k-cosines (estimator (1)) is described in some detail. Deviations from this for other estimators are mentioned where necessary. Note that all thresholds described have been experimentally determined.

(i) Locate all local maxima.

(ii) Apply a threshold, *tc*, to remove those points which lie on approximately straight line segments. These are characteristically small amplitude, local maxima due primarily to remaining discretisation noise still not removed by the filtering operations. A value of $tc = -0.975$ is used and only those points with curvatures larger than this value are retained.

(iii) Then, a technique of non-maxima suppression is used, which is basically a test on the domain size of each local maximum surviving step (ii). Only those $C_{ik} > C_{ij}$ for $(i - k/2) \leq j \leq (i + k/2)$ are retained.

(iv) Then a simple threshold test on the amplitudes of these remaining maxima is performed; any maxima with amplitudes less than the threshold, *ta*, are deleted from the list. This step serves to remove any small amplitude maxima arising primarily as a result of discretisation noise. A value of 0.02 is used for *ta*.

(v) The curvature maxima are now given simply by the list of boundary points surviving the above tests.

Where the detection of minima in the curvature function is required those $C_{ik} < C_{ij}$ for $(i-k/2) \leq j \leq (i+k/2)$ are retained.

Estimators (2) and (3) require the location of both curvature maxima and minima. Local maxima and minima which lie on straight line segments are thresholded out by testing whether they have curvatures which lie between +*tc* and -*tc*. The value of *tc* used is 5 degrees. Amplitude tests follow similarly to previous cases using a value of 5 degrees.

4.4 Results

The results are presented in the form of a series of graphs (figures 7,8,9) of the smoothing factor, *k*, versus the maximum percentage of boundary pixels perturbed which still permit recognition of the polygon. Curves for orders 3,6,8,10,12 and 14 are shown, each using values for *k* of 3,5,7,10,15 and 20 sample points. These may be taken as forming a representative sample of the complete set. In order to enable a more meaningful comparison of individual curves amongst different order polygons, k is expressed in terms of a percentage of the polygon side length, rather than using the absolute value of k.

Each graph shows results for a different curvature estimator and comprises a family of curves, each corresponding to a polygon of a given order. The region enclosed by each curve and the *y*-axis represents the conditions under which the polygons were always recognised.

4.5 Discussion

Theoretically it can be expected that for any value of *k* smaller than the length of a side in a polygon that the curvature maxima (or polygon vertices) would still be detectable and the polygon recognisable. Any value of *k* greater than 100% of a side length will mean that no curvature maxima are found and the regular polygon becomes indistinguishable from its circumscribing circle.

In practice it is found that the actual upper limit for *k* is considerably less than this value. Also, the first statement does not strictly hold since small values of *k* are much more susceptible to any remaining discretisation noise which tends to produce apparent curvature maxima along boundary segments which are perceived as straight lines. The end result is spurious polygon vertices.

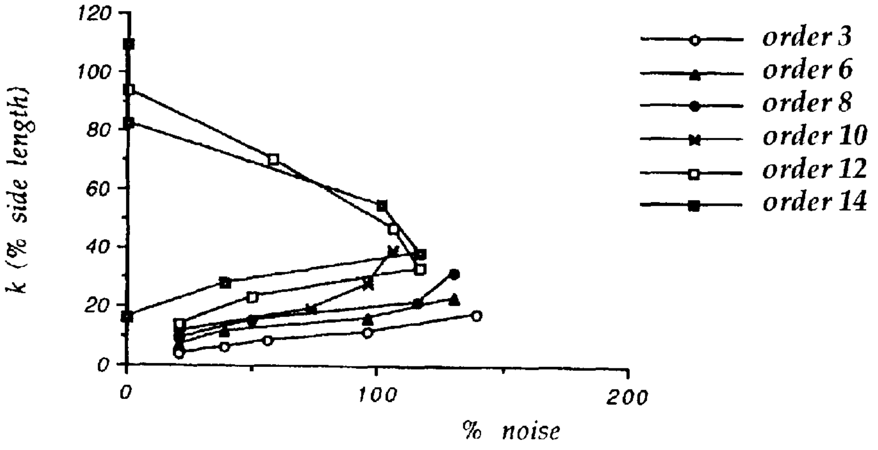


Figure 7. Graph showing the experimental performance of estimator (1). The polygons are always recognised within the area enclosed by each curve and the y-axis.

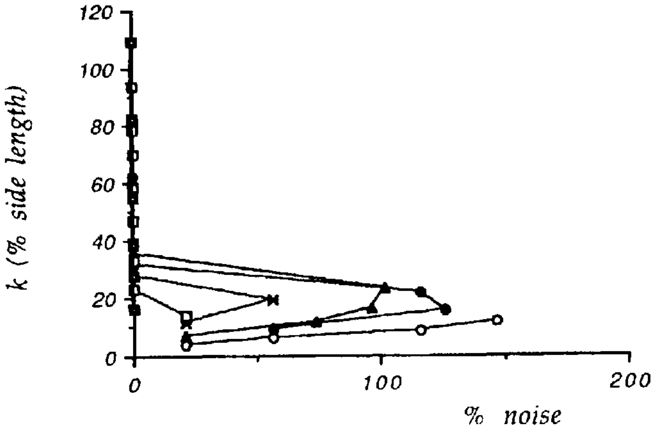


Figure 8. Graph showing the experimental performance of estimator (2). The polygons are always recognised within the area enclosed by each curve and the y-axis.

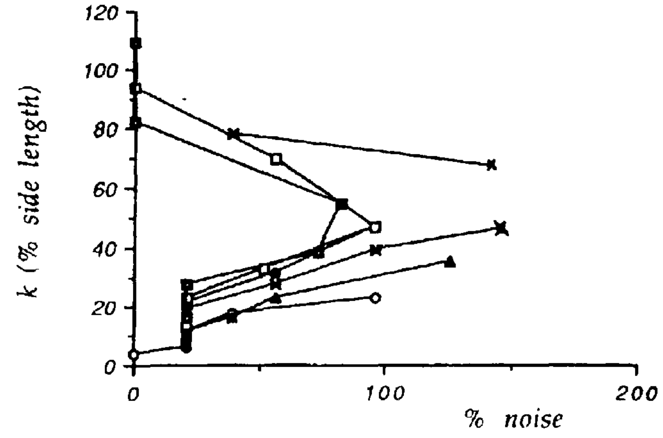


Figure 9. Graph showing the experimental performance of estimator (3). The polygons are always recognised within the area enclosed by each curve and the y-axis.

The graphs illustrate this quite well. They all have the same basic form. For small values of k the maximum amount of noise which can be present before the recognition process breaks down is small. As k increases the permissible noise level increases as well until a point is reached when any further increase in k results in a degradation of the recognition process.

The same absolute values of k were used for each order polygon. These reduce to different ranges when expressed in percentage terms. Thus, for lower order polygons, the values are such that they never reach the upper limit of recognisability. This is why some curves appear to terminate before reaching the y-axis. All curves will eventually turn back upon themselves and intersect the y-axis, but the values of k which would be needed in order to force this to occur would be unrealistically large.

It should be pointed out that the graphs are not intended to be used as a source for the derivation of precise figures. Rather the intention is that they be used to provide a general measure of the relative performance of each point curvature estimator under given conditions (noise level and/or feature size).

The graphs illustrate the range of k values over which a given feature is detectable, and also, how noisy the boundary can be before the detection of that feature becomes improbable. They represent the 'worst-case scenario' in that the smallest noise levels at which the polygons become unrecognisable or features become undetectable have been chosen. A common factor amongst all the curves is that the performance for higher order polygons is not so good as that for lower order polygons. This is a reflection of the decreasing inter-feature distance (side length) and increasing angle size as polygon order increases. For an order 3 polygon 30% of a side length will represent a much larger absolute distance than 30% of a side length for an order 14 polygon. The latter will therefore be much more subject to the effects of boundary noise and thus curvature estimators will have a poorer detection rate and consequently lead to a much poorer recognition rate.

One interesting point to note is that whilst the maximum noise level decreases for increasing polygon order, or decreasing feature size, the optimal values of *k*, (those values which allow recognition of the polygons under the noisiest conditions), are approximately the same for a given curvature estimator.

Estimator (1), (figure 7), has optimal *k* values of about 40%. Estimator (2), (figure 8), has very small optimal *k* values of about only 20%. This may be explained by the fact that the angle at a given point, *s(t)*, is computed from approximations to *x'(t)* and *y'(t)* taken over a range of *i-k* to *i+k* = *2k+1* points coupled with the fact that the curvature measure *s'(t)* is computed from differences in the angles of directly adjacent points on the boundary. This can be compared to, say estimator (3) (figure 9), which has optimal *k* values of about 50%, which uses estimates of *x'(t)* and *y'(t)* over a range of *k+1* points, which obviously more closely approximates the curve, and angles of directly adjacent points are not used to compute curvature. This estimator has a very wide range of *k* values but with a much reduced permissible noise level.

Estimator (2) perhaps shows the poorest results of all. The optimal *k* values of around 20% and less of a side length means that the recognition rate drops off very rapidly as polygon order increases; 20% of the length of a side of a polygon of given order yields an absolute distance which is much smaller than the 40%-50% of a side length suggested by other detectors. This means that this estimator is much more susceptible to the effects of noise than others considered, and that the detection rate for larger angles and smaller distances between features will be comparatively much less.

Estimator (3), by comparison, performs much better than estimator (2). However, for the higher order polygons, although it shows perhaps the widest range of *k* values the maximum allowable amounts of noise are less than for estimator (1).

From a consideration of the above it may be concluded that estimator (2) is really only of limited use in more or less noise-free conditions and for objects which are characterised by relatively 'sharp' angles. Decreasing the values of *k* used, or the distances over which the values of $x'(t)$ and $y'(t)$ are computed, will merely increase the effects of noise. In their paper Wallace et al[16] give an example boundary along with its curvature function. The boundary shows some very rapid angle changes but the maximum in the curvature function is only some 12 degrees per sample. This estimator is also very susceptible to the effects of sub-optimal choice of the smoothing factor. The usefulness of this algorithm for real data must therefore be questioned.

Estimator (3) gives good all-round performance for sides of reasonable length and angles which are reasonably sharp. Its performance is also acceptable for smaller side lengths and less sharp angles under relatively noise-free conditions.

From these experimental results, it would appear that the point curvature estimator which gives the best results for the widest range of angle sizes and side lengths under the noisiest of conditions is estimator (1). Of those estimates of digital curvature investigated it is this *k*-cosine measure which is the most robust in the presence of noise and to sub-optimal choice of smoothing factor.

4.6 Application to the Method of Rosenfeld & Johnston and to the Method of Rosenfeld & Weszka

The specific implementations of the *k*-cosine method due to Rosenfeld and Johnston[48] and Rosenfeld and Weszka[49] are now considered. In particular their performance in the presence of noise is investigated. It has been found that for optimal detection of high curvature points using a smoothed *k*-cosine estimator of point curvature, *k* values of around 40% - 50% of the distance between successive curvature maxima on the boundary are required. This information is used to select the ranges of *k*. The values, *min* and *max* are derived empirically using figure 7 such that

$$k_{min} \leq k \leq k_{max}$$

where

$$k_{max} = 0.8 * \text{smallest expected side length}$$

$$k_{min} = 0.2 * \text{smallest expected side length.}$$

This ensures that the length of the boundary under consideration will never be greater than the distance between two adjacent features of interest. When this occurs incorrect curvatures and domain sizes are assigned to points on the boundary. This is discussed by Davis[50].

This also provides a semi-automatic means of selecting close to optimal *k* ranges for any problem. The only variable is the length of the smallest expected side. This need not be a problem since if only a restricted number of objects are to be presented to any system which might use such a feature detector as curvature, then since *a-priori* information about the objects is available it is sufficient to simply use the smallest side length of all the objects together. For objects which have all their sides longer than this value there is, theoretically, no problem since a wide range of angles and side lengths are still detectable, and in fairly noisy conditions, with small *k* values.

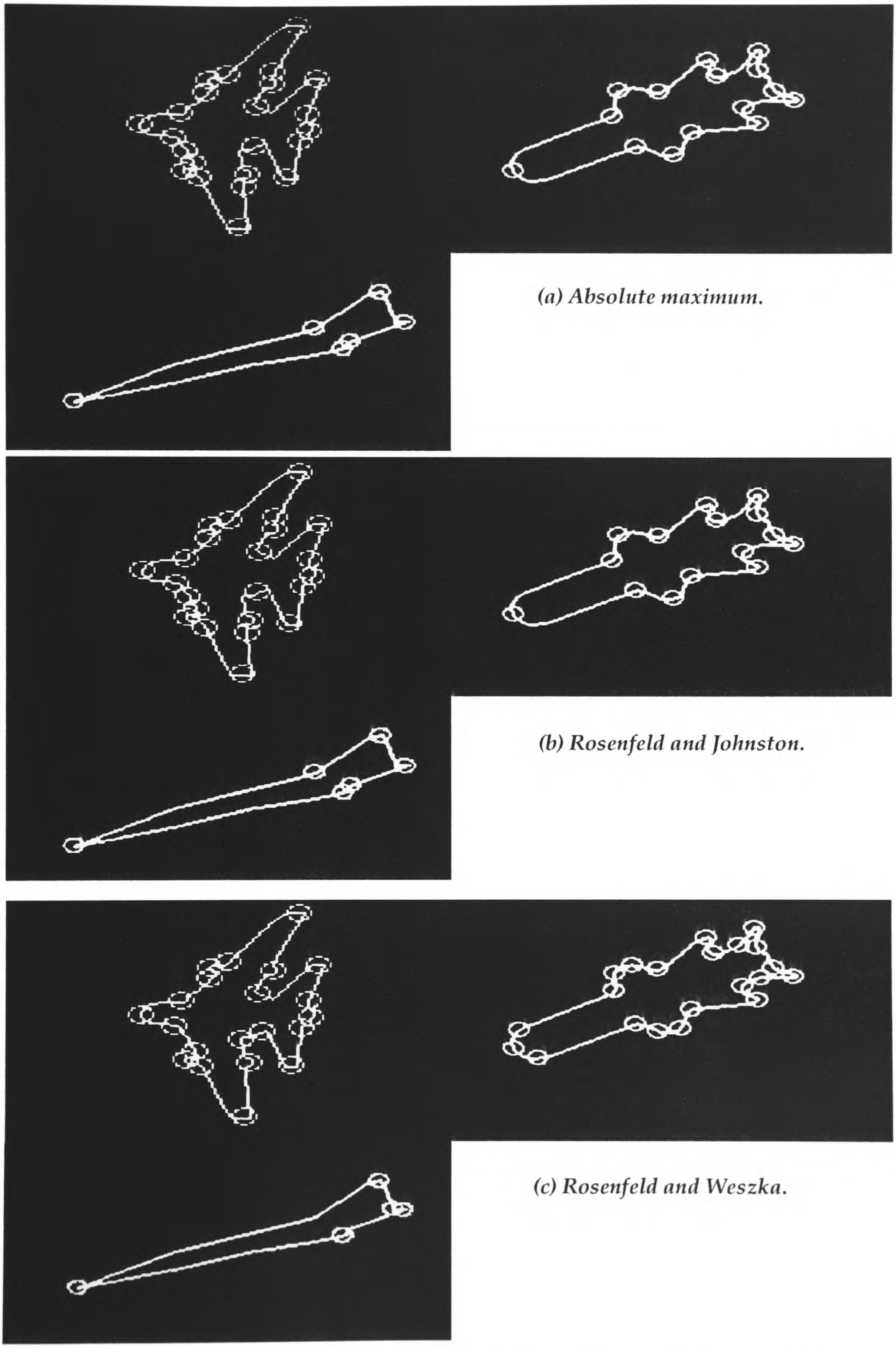
The boundary curves are preprocessed in exactly the same fashion as the polygons and model objects of the earlier analysis and maxima detection follows the procedure already described.

As well as the threshold tests on the k-cosines, a test is also applied to the value of k_{min}. If the value of k_{min} is less than some threshold value, tk, then, k_{min} is set to equal tk. The value of tk has been chosen to be equal to 4. This serves to remove the effects of noise on the boundary curve. The value for tk is chosen from an examination of the performance of this curvature estimator for small k (i.e. $k = 3$) in noisy conditions in the earlier experiments.

These tests constitute an improvement on the original method which assigns a maximum at a point i on the curve, as long as it is a maximum over the required domain, regardless of the values of C_{ik} at that point or of their amplitudes.

Results are presented as a series of outlines with the detected points of maximum curvature marked as small circles. Results are shown for the three objects used throughout this study, namely the missile, the Hawk jet and the Concorde; these boundaries have been generated as described in chapter 2. Results are included for three different methods. Firstly, a different method of selecting the size of each point; simply choose the size with the maximum curvature over the range of k values considered rather than favouring the largest sizes as do Rosenfeld and Johnston[48]. Secondly, the method of Rosenfeld and Johnston[48], with the described modifications and thirdly, the method of Rosenfeld and Weszka[49]. These are indicated by (a), (b) and (c), respectively in figures 10 and 11.

There are basically two problems which occur. Firstly, spurious maxima arise due to the local effects of noise on the boundary. The various threshold tests help to reduce this problem. Also, features may be lost when inappropriate ranges of k values are used. If k

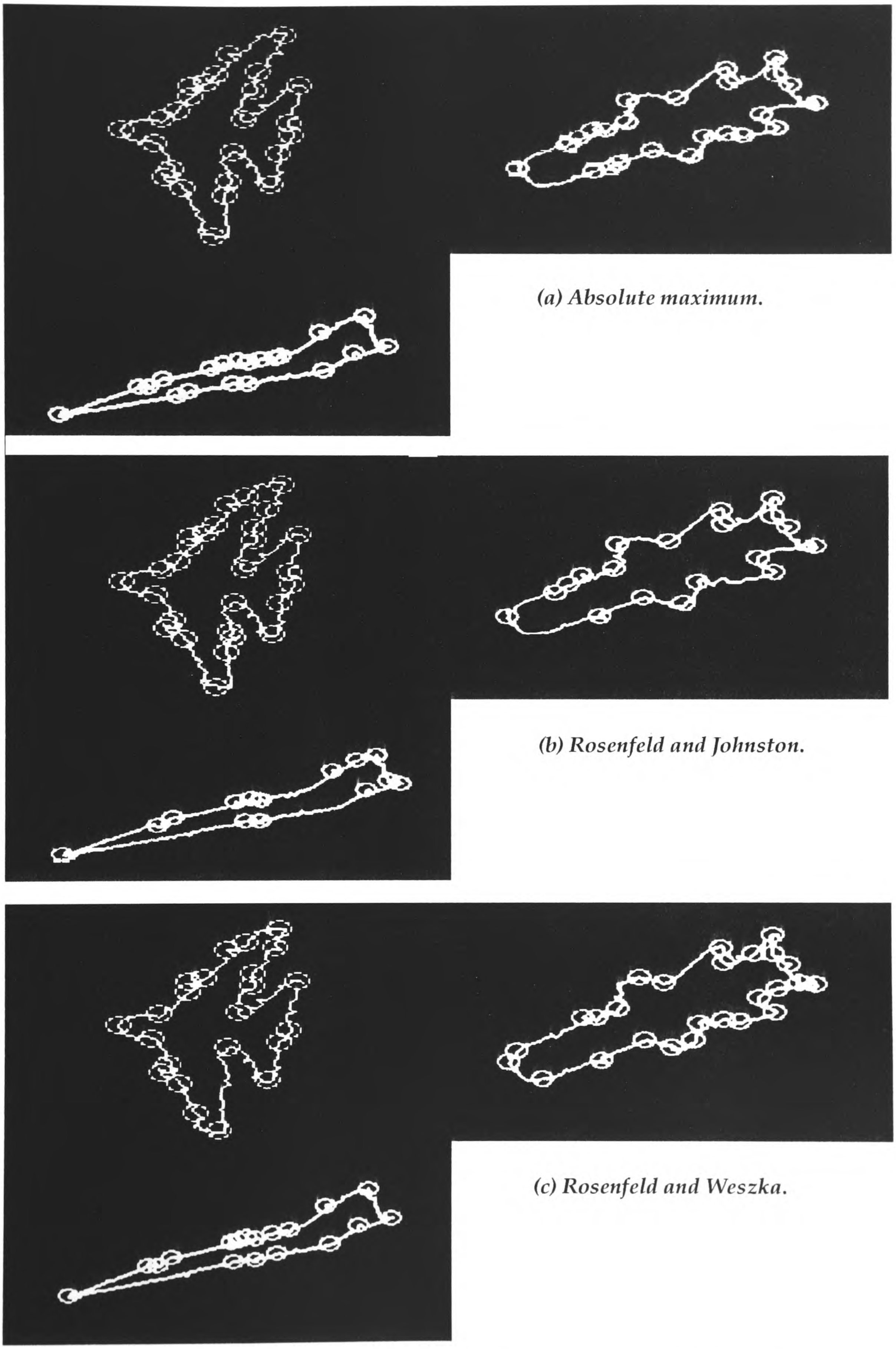


(a) Absolute maximum.

(b) Rosenfeld and Johnston.

(c) Rosenfeld and Weszka.

Figure 10. Performance of Rosenfeld and Johnston type algorithms - no noise; smallest side length = 4% of total boundary length.



(a) Absolute maximum.

(b) Rosenfeld and Johnston.

(c) Rosenfeld and Weszka.

Figure 11. Performance of Rosenfeld and Johnston type algorithms - in ≈52% noise; smallest side length = 4% of total boundary length.

values which are too large are considered, local features will be masked by more global measures of curvature. This has effectively been dealt with by the method of choosing k values but its effects are considered at a later point.

Figure 10 shows the results, in noise-free conditions, for a value of 4% of the total boundary length as the smallest expected side. This is the optimal value for the objects under consideration and corresponds to absolute k values in the range of 5 - 9 sample points. The results are intuitively very pleasing for each object considered. The ability of the Rosenfeld and Weszka[49] method for locating vertices which are 'close' can be seen by the increased detail of the representation of the missile in figure 10(c).

Figure 11 illustrates the performance of these algorithms, for optimal k ranges, in the presence of noise. Noise has been added with coefficients $c1$=0.05 and $c2$=0.10 resulting in about 52% of pixels being disturbed. The effect is clearly visible. Many spurious vertices are detected along straight parts of the boundary whilst in other locations actual features are lost completely in the effects of the noise. The Rosenfeld and Weszka[49] method and the choice of the absolute maximum are most susceptible to this.

In conclusion, the advantages of the Rosenfeld and Weszka[49] technique in near ideal conditions are obvious but in the presence of noise it becomes unreliable. In smoothing, noisy estimates at small values of k are also included in the curvature estimate. Choosing the absolute maximum in the range favours extremely localised, high curvatures of small sizes arising primarily as a consequence of noise on the boundary whereas choosing the sizes as suggested by Rosenfeld and Johnston[48] has the opposite effect, favouring the largest sizes which are much less influenced by noise. For reasonably appropriate ranges of k, and in relatively noise-free conditions the original technique[48], with the present modifications, performs the best.

Recently Teh and Chin[54] have demonstrated an alternative method of automatically determining *k* based only upon the local properties of the curve at the point in question. They claim that precise determination of this value is the primary influence on accurate detection of dominant points and that the actual measure of significance (curvature) used is immaterial. To date, this algorithm has not been implemented and tested in the presence of noise. However, since it is strongly weighted in favour of small *k* it is likely to perform poorly in noisy conditions.

4.7. Conclusions on the Performance of Curvature Estimators.

It has been shown how the effectiveness of these algorithms in representing the boundaries of objects is dependent upon the correct choice of values for *k*. The nature of the 'fall-down' with incorrect *k* values and in conditions of noise has been analysed.

The problem involved with a bad choice of range of *k* values can be overcome by using the presented experimental results coupled with *a-priori* knowledge about the objects under consideration. With these powerful constraints on the choice of *k* values, close to optimal results for relatively noise-free conditions can be achieved.

However, once boundaries become noisy, spurious vertices and lost points of interest affect the representation; any noise which is present on the boundary is, in effect, amplified by the differentiation of boundary coordinates to produce curvature estimates. The spurious vertices can be deleted to some extent by choosing slightly larger values for *k*, but this also causes local features to be lost: they are masked by more global measures of curvature which are derived as a consequence.

The conclusion is that such estimators of point curvature, whilst undoubtedly useful for certain restricted classes of curves (for example, those comprising 'sharp' angles which are not too close together) in relatively noise-free environments, are not robust or efficient enough to be serviceable in the representation of the boundaries of objects in general.

CHAPTER FIVE

Representing Shape 3 - Locating Sides

Computing curvatures from noisy boundaries, even when preprocessed e.g.filtered etc., is not satisfactory. If the boundary were to be approximated by a spline first, then curvature computation would proceed with discretisation-free, mathematically defined boundaries. In many cases this would be sufficient to represent the shape by itself, however, this requires considerable effort. What is required is a robust method whereby the boundary of a 2-D shape can be represented accurately and efficiently and without sophisticated preprocessing.

5.1 The Approach

A rather more promising approach to the problem of representing the shape of 2-D contours might be the detection of the constituent sides of the boundary rather than attempting to locate the junctions between the segments (high curvature points) directly. In such a dual approach, which has also received considerable attention in the literature, the boundary curve is searched for segments which exhibit significant linearity (or curvilinearity) and then curvature discontinuites are assigned to the junctions between boundary segments.

A popular approach, typified by the work of Ramer[55] and Pavlidis[56,57,58], has been to obtain a piece-wise linear approximation to a contour subject to some error of fit constraints. Alternatively, for contours which have little or no significant linear segments, a piece-wise polynomial approximation may be produced. Such polygonal or polynomial approximations tend to locate the breakpoints of segments at, or close to, curvature maxima. A review of these techniques is given by Pavlidis[58].

Using estimates of the general trends in boundary curves to derive point features, rather than attempting to locate the features themselves, means that the detected positions of rapid curvature changes are more stable in the presence of noise which is effectively filtered out. An algorithm of this type for describing the shape of plane contours is presented. It will be shown that the results are superior to those obtained from algorithms for estimating curvature directly.

The approach is based upon a method of constructing representations of contours with varying degrees of refinement. Davis[50] used a similar approach in order to construct a hierarchy of descriptions of a 2-D curve in terms of angles and sides which are able to represent the curve at any level of detail.

By contrast, the method described here incorporates sides and angles of different sizes into a single multi-scale description. The inclusion of an automatic means of selecting appropriate error thresholds and line lengths constitutes an improvement over other methods described.

5.2 The Algorithm

The object boundaries are closed contours, i.e. each boundary is formed by a series of 8-connected pixels. A series of straight line segments of increasing length are fitted to the boundary using a least squares minimisation. The fit is centred about the point, P_k, on the boundary in question (see figure 12). For example, if it is required to fit a line of length n sample points to the curve, C, at point, P_k, then all points on the boundary within a distance of ± $(n-1)/2$ in arc length of point, P_k, are included in the least squares fit.

This is done for a series of points around the boundary which are equidistant in arc length. Then, for each line length, the points around the boundary where the errors of fit are locally minimal are found, producing a sub-set of the original fitted lines. These points

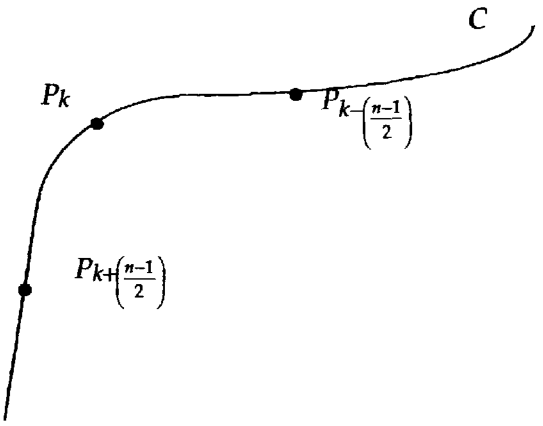

C
P_k
$P_{k-\left(\frac{n-1}{2}\right)}$
$P_{k+\left(\frac{n-1}{2}\right)}$

Figure 12. Points used in the least-squares fit of a line of length, n, to the curve, C, at the point, P_k.

should correspond to regions of the boundary which approximate straight lines. For example, in figure 13 it can be seen that if n sample points are to be used in a least squares fit at point P_k then the error of fit will be large. As the line slides counter-clockwise around the boundary the error of fit will become smaller, until it reaches point P_j where it becomes a minimum. As the line slides further around the boundary towards P_i the error of fit will begin to increase again.

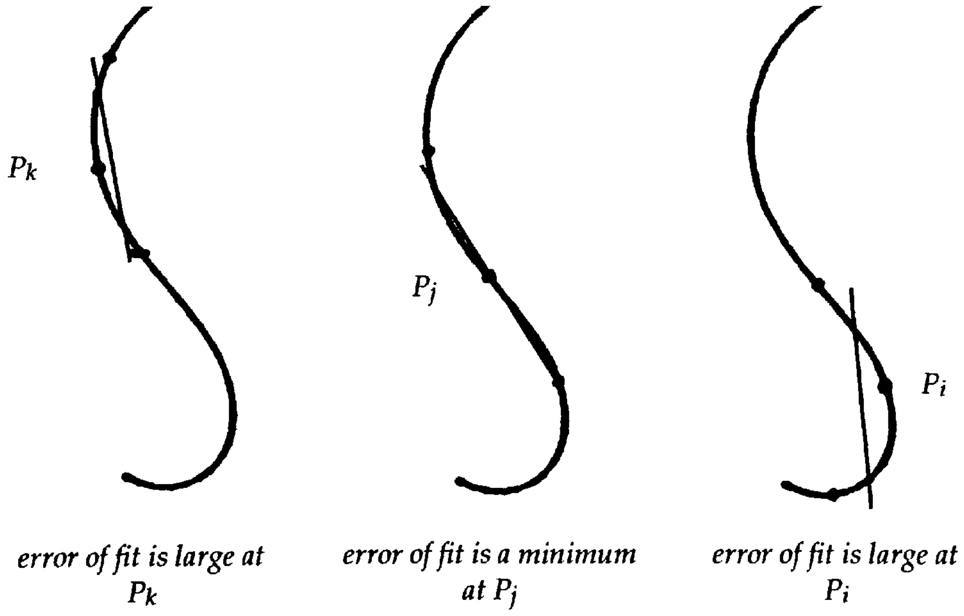

P_k
P_j
P_i
error of fit is large at P_k
error of fit is a minimum at P_j
error of fit is large at P_i

Figure 13. Boundary position affects error of fit.

These lines, corresponding to positions where the error of fit is locally minimal are then assigned to an initially empty plan of the object boundary, in a top-down fashion, such that the longest lines are assigned first. Once a line is assigned to the boundary plan its extent on the plan is marked as full; no other lines are subsequently allowed to be assigned within these full regions of the plan. This process of assigning lines to the boundary plan is continued until either there are no more lines to fit, or the plan is completely full. The assigned lines are then extrapolated across any small gaps which might remain in the plan and their intersection points are found. These points become the object features which correspond to the points of maximum curvature originally mentioned.

Before any line fitting is attempted, boundaries are filtered using a simple 3-point weighted average and resampled in a similar fashion to that described in section 4.3.

Line-fitting proceeds in a bottom-up fashion, beginning with the smallest line length. This has been set to be 5 sample points; this is a reasonable line length to choose since anything smaller than this is likely to be due to noise on the boundary. For noise which influences the boundary over a longer length than 5 sample points it is likely that these short lines will not be included in the boundary representation since that portion of the boundary may already be filled by longer lines; lines are assigned to the representation in a top-down fashion, i.e. longest lines first.

The least squares fit proceeds by minimising the integral square error, E_2, where, for a line n sample points long

$$E_2 = \sum_{i=1}^{n} e_i^2 \qquad (27)$$

and the point-wise error, e_i, is given by the Euclidian distance between the boundary point, i, and the normal form of the approximating line

$$e_i = |\sin\psi\, x_i + \cos\psi\, y_i - \rho\,| \,. \qquad (28)$$

The values ψ and ρ are the orientation of the least squares line and its distance from the origin respectively (figure 14).

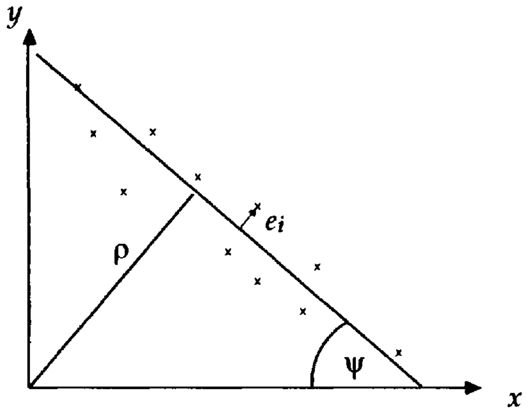


Figure 14. The least-squares line.

It can be shown[57] that E_2 may be computed simply from the following

$$E_2 = \frac{1}{2}(M_{xx} + M_{yy} - [(M_{xx} - M_{yy})^2 + 4M_{xy}{}^2]^{0.5}) \qquad (29)$$

where

$$M_{xx} = \sum_{i=1}^{n}(x_i - M_x)^2 \qquad (30)$$

$$M_{yy} = \sum_{i=1}^{n}(y_i - M_y)^2 \qquad (31)$$

$$M_{xy} = \sum_{i=1}^{n}(x_i - M_x)(y_i - M_y) \qquad (32)$$

and

$$M_x = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad (33)$$

$$M_y = \frac{1}{n}\sum_{i=1}^{n} y_i \,. \qquad (34)$$

The orientation of the line, ψ, and its distance, ρ, from the origin can be computed from equations (30)-(34) as follows

$$\tan(2\psi) = \frac{2M_{xy}}{(M_{xx} - M_{yy})} \qquad (35)$$

$$\rho = \sin\psi\, M_x + \cos\psi\, M_y \,. \qquad (36)$$

This analysis is really the analysis of moments[59] where ψ corresponds to the principal axis, or axis of least inertia, through the *n* boundary points.

It is not necessary to compute the whole summations given by equations (30) - (34) at each point; as the line moves around the boundary it is necessary only to delete the contributions from those points no longer under consideration and add in those from the new points. The same idea can be applied to individual points on the boundary for successive line lengths.

The line fitting process thus begins with a line length of 5 sample points. For each sample point on the boundary the least squares line of length 5, centred at that point, is found. Then, all points around the boundary where the error of fit is a local minimum, subject to certain constraints (these are described later in the chapter), are found. Next, the line length is incremented, a step of 4 sample point in size is used, and the new length line is fitted to each point around the boundary and local minima found. This process continues until the lines become too long, i.e. no local minima satisfying the constraints on error of fit are found.

This represents an automatic method of selecting line lengths which are appropriate to the object under study. This requires no *a-priori* knowledge. If a line of length n sample points can be fitted, then all smaller lines can also be fitted, i.e. the algorithm will only terminate when the longest line possible has been found.

Since a single multi-scale description is required, in contrast to the hierarchical scheme described by Davis[50], it is necessary to compare the quality of fit of lines of different sizes. Therefore, in order that a direct comparison between the error of fit of lines of unequal length may be made, the integral square error given by equation (29) is normalised by dividing by the number of sample points in the fitted line. This results in a line fitting procedure which is based upon minimising the mean squared deviation over the points under consideration.

The search for local minima is carried out in the same pass of the data as the line fitting. When locating local minima, lines are only retained if the value of E_2 is below a given threshold. The error threshold values used have been determined experimentally as follows.

The line fitting part of the algorithm is initially run over the object boundary with two lines of short length, i.e. 11 and 15 sample points. For each length the positions where the error of fit is minimal are found. In this case minima are retained only if the error of fit is less than a pixel. This serves to remove any obviously spurious local minima where the mean squared deviation exceeds this value. The mean and standard deviation of the error over the positions of local minima is computed for each length line. Using a 99.73% confidence level three standard deviations are added to the means and the average of the two resulting errors is computed. This error forms the estimate of the error threshold which should be used.This provides a means of automatically determining appropriate error thresholds, depending upon how noisy the boundary actually is.

The appropriate choice of error threshold is important. Choosing values which are not large enough means that small scale features due to noise on the boundary are likely to become features of the description; this is undesirable. Choice of error thresholds which are too large leads to the inclusion of longer length lines than is necessary and consequently, to the loss of small scale features of the boundary. The setting of thresholds, based upon estimates of how noisy the boundary actually is, minimises this effect.

There now remains a subset of the original fitted lines, each of which satisfies the constraints on error of fit. There may be significant overlap between these lines. This is undesirable and so they are analysed to remove much of the overlap between lines of the same length. This part of the algorithm searches for pairs of lines which overlap each other by more than 10%. Of the overlapping lines, the line with the smallest mean squared deviation is retained and the other is deleted from further study.

Now, consider an initially empty plan, or map, representing the boundary of an object. This can be thought of as keeping track of which parts of the boundary have had lines assigned to them or, in other words, have already been adequately represented by the fitted lines. The least squares lines which still survive are assigned to this map, in a top-down fashion such that the longest lines are assigned first. As each line is assigned its domain (the sample points on the boundary which were used in the least squares fit) is marked as full. Subsequent lines are no longer allowed to be assigned within this region. Some overlap is allowed between lines of different lengths; this can be thought of as being the equivalent of the non-maximum suppression employed in locating curvature maxima. This threshold on overlap is set to be 33% of the current line length. These thresholds have been empirically determined. If thresholds are set too large then lots of overlapping line segments result. If thresholds are set too small then large gaps result.

This process continues until either there are no more lines to fit, or the plan becomes totally full, subject to constraints on allowable overlap. Because the longest lines are assigned first, the effect is to weight the longer lines more heavily than shorter lines. This also helps to reduce the influence of noise.

The result so far is a representation of the boundary in terms of a series of straight line segments. Because of the constraints on allowable overlap there are likely to be small gaps in the representation. In other places the lines will be overlapping by small amounts.

The plan is inspected and the intersection points of the lines found; these constitute the point features which are used in the matching process. Where gaps are large, an algorithm due to Ramer[55] is used to find the piece-wise linear approximation to the boundary between these points. This often simply joins the two end points by a straight line, but sometimes results in the insertion of an extra vertex. Where there is only a small gap, no gap, or lines overlap, the course of action may be to simply intersect the lines, or retain both end points. This depends upon the relationship of the fitted lines to each other and to the object boundary in that region, i.e. whether the lines are approximately parallel or at angles to each other, and whether the boundary curves rapidly in the region of the associated end points of the lines. The coordinates of these points form the point features used in the matching of object and model data.

Once the vertex list has been sorted into sequential order, i.e. order around the boundary, each vertex is examined and assigned an integer code which depends upon whether the boundary is locally convex or concave at that point. This information is written, along with information obtained earlier, to an array. This array constitutes the object description (see chapter 6).

The process is summarised by the pseudo-code algorithm below, comments are enclosed by /**/.

```
PROCEDURE fit_lines_to_boundary
 BEGIN
   /* fit lines */

   until (no. local minima(nlm) = 0)
       l = minimum length;
       i = 1;
       until (i  > no. sample points on boundary(nsp))
           find the least squares line of length l, centred at point i;
           i = i + 1;
       end until
       find points where error of fit of line of length, l,
       is locally minimal and <  threshold;
       l = l + increment;
    end until
    maxlength = l - increment;

    /* remove overlapping lines */

    l = minimum length;
    until (l  > maxlength)
        find all pairs of lines which overlap by  > 10%
        and keep line with smallest error of fit;
        l = l + increment;
    end until

    /* assign lines to plan and initialise plan */

    i = 1;
    until (i  > nsp)
        plan[i] = EMPTY;
        i = i + 1;
    end until

    /* begin with longest line */

    l = maxlength;
    until (l  < minimum length or plan is FULL)
        if(nlm at length l, (nlm[l]), remaining > 0)
          i = 1;
          until (i  > nlm[l])
              if  > 33% of points in line are EMPTY in
              plan then assign line to plan and mark
              corresponding points in plan as FULL;
              i = i + 1;
          end until
        end if
        l = l - increment;
     end until

     /* locate vertices */

     sort assigned lines into sequential order and
  find the intersection points;
END
```

5.3. The Performance of the Algorithm

This section investigates the performance of the line fitting algorithm on noisy boundaries and compares results to those for Rosenfeld and Johnston's method of detecting points of maximum curvature[48]. Figure 15 shows some results for the same objects with the same level of applied noise, i.e.52% or *c1*=0.05 *c2*=0.1, as those used in the curvature analysis and can be compared directly to figure 11. The results are intuitively pleasing.

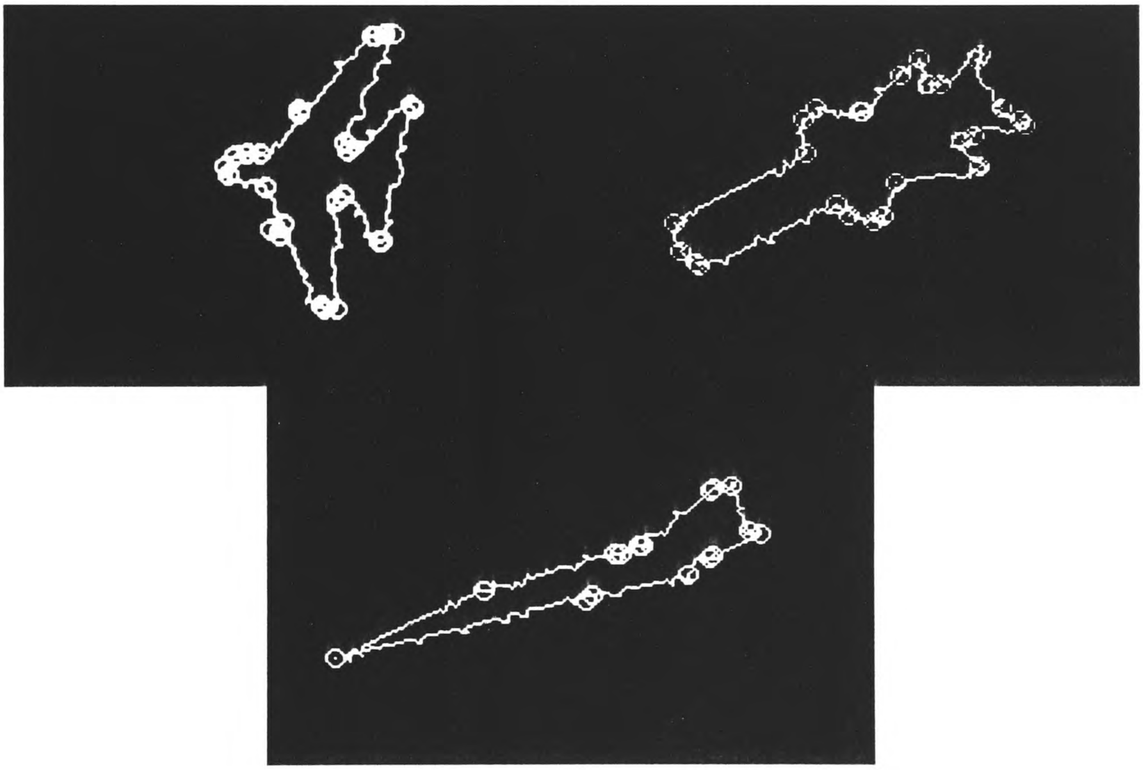


Figure 15. Vertices found using the line-fitting algorithm on noisy images(52%).

However, a rather more quantitative comparison between the two methods would be desirable. The two techniques were applied to the missile, the Hawk and Concorde many times. In each case the located vertices were compared to the known model vertices and the percentage of correct locations computed. By correct it is meant that the model and computed vertices are within a distance of ±5 pixels of each other. This was done for several different increasing levels of noise, and at each noise level the mean value of the percentage correct over the three test objects was computed. Figure 16 shows a plot of mean percentage of correctly located pixels against level of applied noise. It can be seen that the line fitting algorithm consistently outperforms the curvature detector and is able to cope with far higher levels of noise.

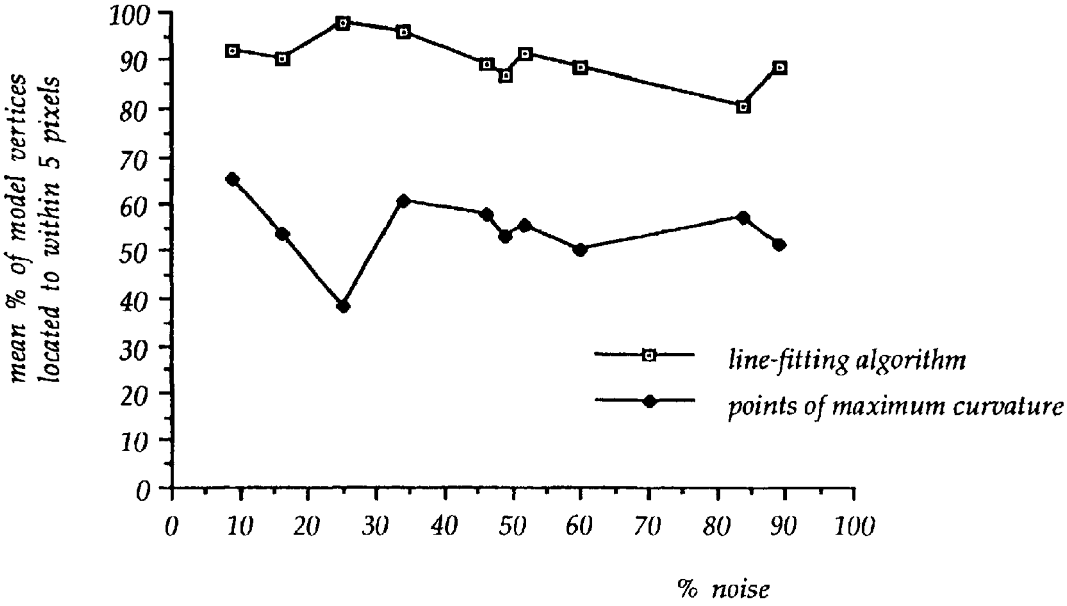


Figure 16. Graph showing the mean percentage of model vertices located to within 5 pixels averaged over the three test objects.

The line fitting algorithm does not have the drawbacks of algorithms for computing curvature. Their chief disadvantage is the fact that in one way or another, some assumptions have to be made regarding the domain size of points on the curve. This is reflected in the choice of smoothing factor, *k*. Choosing *k* too large will lead to the loss of some smaller scale features in the boundary, whilst weighting *k* in favour of smaller values means that curvature estimates are influenced more heavily by any noise which may be present, leading to the production of spurious vertices. The problem does not arise in the line fitting algorithm described. No assumptions about domain sizes are made and lines of all possible lengths are fitted to the boundary at all possible locations. No lines which are too long will be fitted since once error of fit constraints are exceeded the line fitting terminates. Automatic selection of error thresholds means that the most suitable set of fitted lines is obtained. Because longer lines are assigned to the representation of the boundary first, the influence of noise (noise is usually localised and small scale) is reduced. This is in contrast to methods for finding high curvature points directly. These involve differentiation and consequently result in noise amplification. Thus, it has been shown that using estimates of the general trends in boundary curves to derive point features, rather than attempting to locate the features themselves, means that the detected positions of rapid curvature changes are more stable in the presence of noise which is effectively filtered out.

CHAPTER SIX

The Database of Model Descriptions

This chapter describes the construction and method of storage of the database of model descriptions. It also describes how records are retrieved from the database during matching. A spare quadrant of the image plane is used to store the database. The approach is analogous to that used by Fairney[60]. Since records are stored in order of computation only, a method of selective retrieval of database entries is used involving a simple measure of object shape. This acts as a pointer to regions of the database where possible matching records lie.

6.1 Computing the Model Data.

Wire frame models of a missile, Concorde and a Hawk jet trainer are used to generate the database. Using a standard transformation matrix[39], each wire frame model is rotated in turn through 5 degree increments in yaw & roll; pitch angle is maintained at zero degrees throughout. The density of database entries is a reflection of the required accuracies in orientation stated in section 1.1. Yaw and roll angles are used in the range [0,90]. Using the outline of the rotated model, a description is generated at each orientation, i.e. each yaw and roll angle, in terms of straight line segments, as described in section 5.2. The result is a series of integer *x,y* coordinate pairs; each pair corresponds to the end-point of a segment. These values are stored as a series of arrays; each array represents the model description at a particular orientation. Information which is of use when matching is also produced and this information is included in each array of model data. The contents of each array is shown in figure 17.

The first 8 fields constitute a header record which contains information about the object which is of use when matching. The first field, *header*[0], is a count of the number of vertices which make up the model description. The 2nd and 3rd fields hold the yaw and

no.of vertices	yaw	roll	proj. area	pr. axis	x centroid	y centroid	object type	x	y	vertex type	....	x	y	vertex type

Header record **vertex coordinates and types**

Figure 17. The array of model data.

roll of the model. Field 4 holds the projected area of the silhouette in that orientation and the 5th field holds the orientation of the principal axis of the silhouette. Fields 6 and 7 contain the x and y coordinates of the centroid of the silhouette and field 8 is an integer code which identifies the model as either a missile, a Hawk jet or a Concorde.

The remainder of the record is filled with the integer x,y coordinates of each vertex and integer codes representing the type of each vertex, i.e. whether convex or concave. The whole record has a length which is given by *header*[0]x3+8.

Sensed object descriptions, section 5.2, also produce similar arrays of data, enabling a comparison of like with like during matching.

6.2 The Database of Arrays of Model Data.

A spare quadrant of the image plane is used to store the database of model arrays. Because the image plane will only accept integer values in the range 0-4095, a maximum of 4096 locations on the image plane only can be addressed using integer values. To maximise use of the image plane the quadrant is divided into 4096 fixed size segments, each 64 pixels wide. The length of model arrays will rarely be exact multiples of 64 pixels and as a result some space is wasted. However, the method does mean that the entire quadrant can be addressed in blocks of 64 pixels. Each model array occupies the image plane starting at locations which are multiples of 64 pixels, i.e. $512^2/2^{12}$. In practice, even with the data for three different models simultaneously loaded onto the image plane quadrant less than half of it is used.

As the data for each orientation of the model is computed, it is written to the image plane in the next available contiguous location, (*line, col*), given by

$$line = INT\left(\frac{p \times 64}{512}\right) \qquad (37)$$

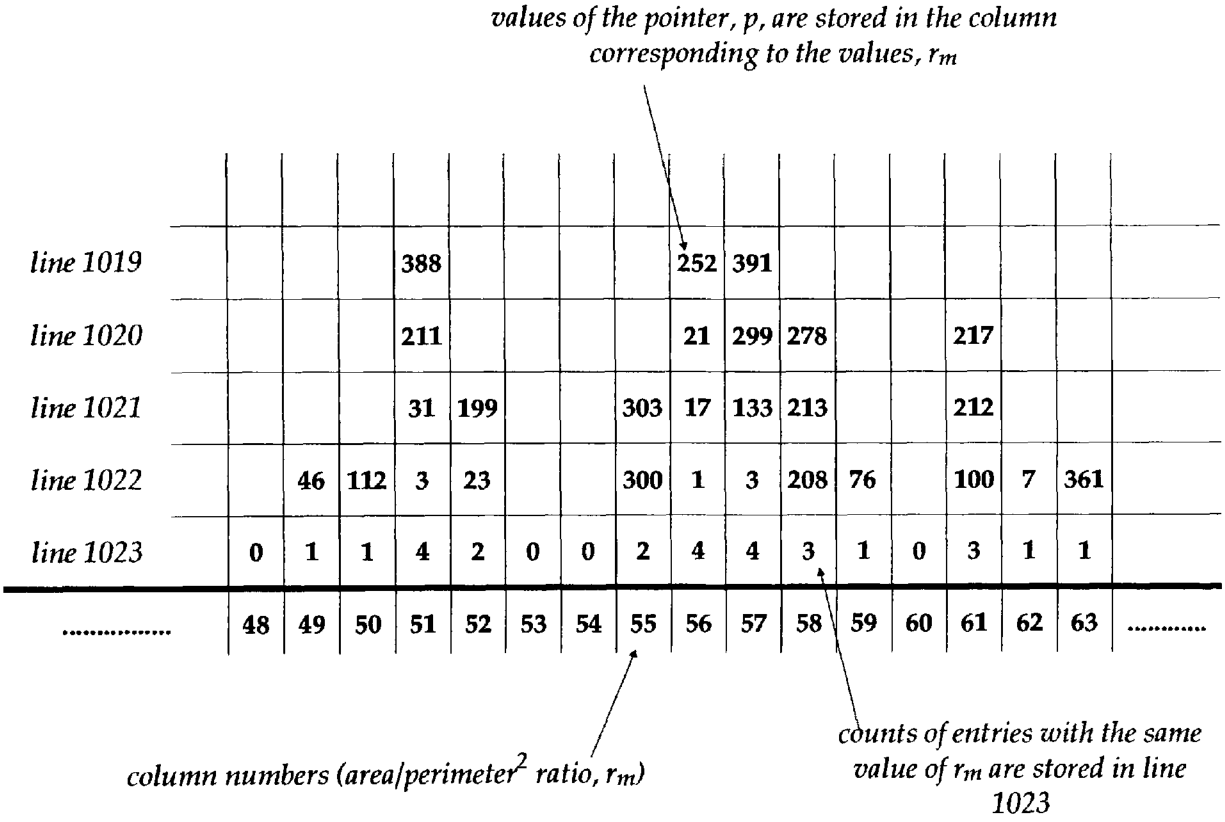
$$col = MOD\left(\frac{p \times 64}{512}\right) \qquad (38)$$

where, *p*, is a count of the number of 64 pixel blocks currently used. Thus, records are stored in order of computation only. The current value of *p* is incremented accordingly and written to the image plane at a fixed location as each new array is added to the database. The value, *p*, therefore acts as a pointer to the next available contiguous block of 64 pixels.

Since records are stored in order of computation only, an efficient search of the database requires selective retrieval of records. Thus, when searching the database for possible matching model arrays it is necessary to avoid a global search. With this aim, a method of pointers to individual model arrays based upon the area/perimeter^2 ratio, r_m, was devised. The ratio , r_m, for this model orientation was used as a subscript to a pixel in line 1023 of the image plane (in practice, any convenient location can be used). In the column directly above this location the pointer, *p*, to each model orientation whose area/perimeter^2 ratio, r_m, is the same is stored (see figure 18). In practice the value, r_m, is in the range 0-1, however because the image plane can only accept positive integer values, r_m is scaled by a constant to give values in the range 0-500.

Line 1023 contains a count of entries per ratio; as a new model is generated and its data written to the image plane so this count is incremented in the corresponding location. Storing this value tells at a single pixel read how many arrays need to be compared for a particular value of r_m.

When the database is complete, line 1023 contains integer values corresponding to the number of model records in the database whose ratio, r_m, corresponds to that column number. The pixels directly above a non-zero entry will contain integer values which point to the location in the database of a model array. In effect, the last 8-10 rows of the image plane quadrant resemble a histogram, in which the x-axis represents the area/perimeter^2 ratio, r_m, and the y-axis is the number of entries per value of r_m, see figure 18.



values of the pointer, p, are stored in the column corresponding to the values, r_m

line 1019				388					252	391							
line 1020				211					21	299	278			217			
line 1021				31	199			303	17	133	213			212			
line 1022		46	112	3	23			300	1	3	208	76		100	7	361	
line 1023	0	1	1	4	2	0	0	2	4	4	3	1	0	3	1	1	
....	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	....

column numbers (area/perimeter^2 ratio, r_m)

counts of entries with the same value of r_m are stored in line 1023

Figure 18. Layout of the image plane showing pointers to database records.

Adding new models to the existing database is straight forward with the value of p, which gives the next available storage location, being read initially from the image plane. Four databases were generated; one for each of the three objects, and one multi-object database with the data for all three objects written to the image plane, one after the other, as described. This latter database is used when object recognition is required as well as orientation determination.

The pseudo-code algorithm below illustrates the building of the database; comments are enclosed by /* */.

```
MAIN()
BEGIN
 INCR = 64;
 LEN = 512;

 if (all new data)
   clear image plane; this sets all counts to zero;
   else
      read value of p from image plane;
 end if

while (there are still orientations to build)
     generate view of model;     /* see section 2.3 */
     fit_lines_to_boundary  /* see section 5.2 */;
     find the  area/perimeter$^2$ ratio, pratio;
     write description to array;
     length = header[0] * 3 + 8;

     /* no. of 64 pixel blocks this record requires */

     newlength = length DIV INCR + 1;

     /* find position of next contiguous location */

     col = (p*INCR) DIV LEN;
     line = MOD [(p*INCR) DIV LEN];
     if (record overflows the image plane quadrant)

       /* increment pointer, p and find new (line,col)*/

       p = p + newlength - 1;
       col = (p*INCR) DIV LEN;
       line = MOD [(p*INCR) DIV LEN];
     end if
     write array to image plane at (col,line);

     /* m is a count of the no. of records so far */
     /* with the same value of pratio */

     read count, m, from image plane at (pratio,1023);

     /* update the count, m, and write the value of p */
     /* to the appropriate position in the column, pratio. */
     /* p points to the position (col,line) at which the */
     /* current model record has been written. Update the */
    /* pointer, p */

     mptr = m + 1;
     write mptr to image plane at (pratio,1023);
     lpos = 1023 - mptr;
     write p to image plane at (pratio,lpos);
     p = p + newlength;
     write p to image plane;
 end while
END
```

6.3 Retrieving Model Data

When searching the database for a best match it is desirable to limit the number of model arrays that are read and compared, to a small but suitable subset of the database. To achieve this the area/perimeter^2 ratio of the image object, r_i, is computed; this is used to direct the search of the database. Those model arrays are scanned whose ratio, r_m, is within the range

$$r_i - t \le r_m \le r_i + t \qquad (39)$$

where t is a predetermined threshold.

The values $r_i \pm t$ represent the upper and lower bounds of the area/perimeter^2 ratio of model arrays to compare. By scanning line 1023 of the image plane between these two points, the values directly above non-zero entries will produce pointers to model arrays whose ratio, r_m, is within these bounds. In this way only a small proportion of the database is searched.

A value for t is derived in the following way. Firstly a large number (>30) of images are matched to model data using a global search and comparison of the database. For each match the distance d, is computed where

$$d = | r_m - r_{i.} |$$

In this instance, r_m, is the area/perimeter^2 ratio of the matching model array. From the resulting distribution the mean, $\bar{d}$, and the standard deviation, σ, are computed. Using a 99.73% confidence level the threshold, t, is set to be

$$t = 3\,\sigma.$$

In practice, this limits searches to less than 20% of the database.

The value, *t*, is written to the image plane in a fixed location as part of the database. For the case of the multi-object database the largest threshold of the three objects is used and applied to all objects.

Having located a model array in this way a match is computed with the image array; this match is described in chapter 7.

CHAPTER SEVEN

Matching Sensed and Stored Model Descriptions

This chapter describes the steps which are necessary in order to obtain an estimate of object orientation, given an arbitrary 2-D projection.

The first step is to generate the description of the 2-D contour as described in section 5.2. The area/perimeter^2 ratio of the description is computed; this guides the search of the database (see section 6.3). Model database records whose area/perimeter^2 ratio, r_m, lies within the range given by equation (39) are selected for comparison. Then, the object description is compared to each selected model record in turn. This comparison is realised through a pose-clustering procedure in 2-D (see section 7.1). Thus, the object in the image is registered with each selected model projection. The best match and orientation estimate is provided by the model record which provides the best registration with the model data. These processes are described in sections 7.2 and 7.3.

7.1. The Pose-clustering Paradigm in 2-D Space

Consider figure 19 which shows a single object edge vector, $\vec{v_i}$, and a single model edge vector, $\vec{v_m}$. The RST transformation which will bring $\vec{v_m}$ into registration with $\vec{v_i}$ may be computed as follows.

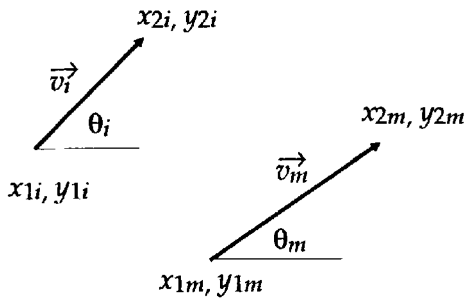


Figure 19. An RST transformation.

The rotation angle, θ, is given by

$$\theta = \theta_i - \theta_m \qquad (40)$$

where θ_m is the orientation of the model vector, $\vec{v_m}$, and θ_i is the orientation of the object vector, $\vec{v_i}$.

The scale change, s, is given by

$$s = \frac{L_i}{L_m} \qquad (41)$$

where L_m = length of model vector, $\vec{v_m}$

and L_i = length of object vector, $\vec{v_i}$.

The translations, Δx and Δy are then given by

$$\Delta x = s\, y_{1m} \sin\theta - s\, x_{1m} \cos\theta + x_{1i} \qquad (42)$$

$$\Delta y = -s\, x_{1m} \sin\theta - s\, y_{1m} \cos\theta + y_{1i}. \qquad (43)$$

The resulting transformation, in homogeneous coordinates, is then

$$[\,x_i \;\; y_i \;\; 1] = [\,x_m \;\; y_m \;\; 1]\begin{bmatrix} s\cos\theta & s\sin\theta & 0 \\ -s\sin\theta & s\cos\theta & 0 \\ \Delta x & \Delta y & 1 \end{bmatrix}. \qquad (44)$$

Now, consider an image (the sensed object) which is comprised of a series of points, each of which represent some feature of the object under study e.g. points of maximum curvature around the boundary (figure 20). Exactly how such points are located is discussed in section 5.2. These points may be connected by straight line segments such that the image now comprises a series of directed line segments or edge vectors. Next, consider a similar image which will be the model. It is required to bring the model into registration with the sensed object, or in other words, find the RST transformation which optimally brings the model data into alignment with the corresponding image data.

A solution to this problem is provided by a pose-clustering procedure which can be described as follows. Consider first in figure 20, the model edge vector $\vec{a_m}$. An RST transformation can be computed which will bring this into registration with image edge vector $\vec{a_i}$. This transformation comprises a scale change, s, a rotation angle, θ, and two

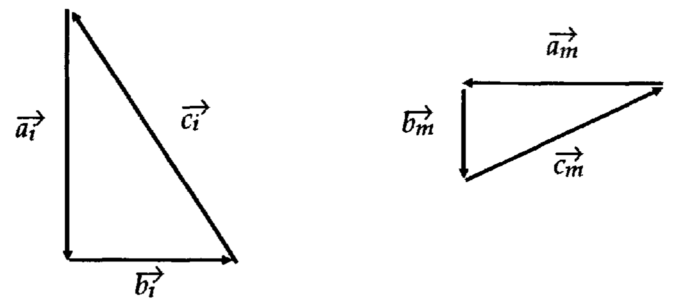


Figure 20. Both image and model data must be brought into registration.

translations, Δx and Δy. If one imagines a 4-D space indexed by s, θ, Δx and Δy which is initially empty, then this computed transformation can be considered to contribute a count of one to the particular cell, or bin, in that space. RST transformations which will bring this vector into alignment with all of the other image edge vectors $\vec{b_i}$, $\vec{c_i}$ can also be computed. These transformations also contribute counts to their respective bins in the 4-D space. Next model edge vector $\vec{b_m}$ is considered and RST transformations are computed which register it with all of the image edge vectors. At the same time the counts associated with the corresponding bins of the 4-D space are incremented. This procedure is followed until all model edge vectors have been considered. The globally optimum transformation is then found by looking for the most significant cluster in this 4-D space. This process of allocating data quads representing the four parameters of the candidate transformations to cells in the 4-D space is termed binning.

Some care needs to be taken with this process. A hierarchical approach is adopted and a series of overlapping bins are used in practice in order to prevent the splitting of clusters between bins. Several cluster centres may be followed up at any given level, and in the final level the best cluster is chosen from several candidates by computing 'the average degree of alignment' between the image and model data. The cluster with the smallest r.m.s. distance between the edge vectors provides the final match which should also be optimum in a global sense.

7.2. Extension to Determine 3-D Orientation.

This approach has been extended such that rather than just determining localisation and orientation in one plane, (i.e the plane of the image) it is possible to determine the orientation of an object in 3-D space and its position within the field of view.

Orientation has been defined in figure 1. For any fixed yaw and roll angles the problem reduces to the 2-D case already mentioned. Consider a database containing descriptions of an object in terms of its edge segments for a series of yaw and roll angles. Then, given an image of this object at some unknown orientation, it is possible to register the description of the object in this image with the elements of the database. For some combinations of yaw and roll the registration will be very poor but for orientations close to the true orientation of the sensed object the registration will be good. The problem then reduces to that of finding the optimum registration of the model edge vectors in the database with those of the object. This follows a hypothesise and test type of approach in which simple measures of shape are used to select areas of the database for matching. Each selected model entry can be considered as representing a hypothesised pose of the object in the image. A pose-clustering procedure is used to derive the parameters of the

RST transformation which will register the object with each of its selected model descriptions. The validity of each hypothesised pose is then tested by applying the derived transformation to the image data and comparing the result to model descriptions. The yaw and roll angles are given by the yaw and roll angles of the best match entry in the database whilst the pitch angle falls out of the registration procedure; this is just the computed rotation angle, θ, between the sensed model and sensed image.

7.3 The Registration Process

The output from the feature detection algorithm is a list of integer x,y coordinates corresponding to end-points of linear segments around the boundary of the object. Each vertex has an associated integer code depending upon the angle between adjacent segments. For angles between zero and π a type 'convex' is assigned whilst for angles between π and 2π a type 'concave' is assigned.

The area/perimeter^2 ratio of the description, r_i, is computed and model database records whose area/perimeter^2 ratio, r_m, lies within the specified range of this value are selected for comparison (see section 6.3). Then, the object description is compared to each selected model record in turn through a pose-clustering procedure.

For each selected model database record, only combinations of image and model edge vectors with the same vertex types are included in the 4-D transform parameter space. This controls the combinatorics of the problem. A more significant reduction is provided by the following. The principal axis of objects is used; this is precomputed during the feature detection stage of the procedure in the case of image data and at the stage of database generation in the case of model data. Combinations of object and model edge vectors are rejected if the computed rotation angle does not fall within a predetermined threshold of the difference in the principal axes of the object and model. A threshold of ±15 degrees is used. This is a robust procedure to adopt since even in the case of corrupt data the principal axis should not be in error by more than this amount. This obviously depends upon exactly which part of the object is occluded and is discussed further in section 10.2. As an example, for the Hawk jet and the Concorde with up to 30% of the boundary occluded the average deviation, computed over all test orientations used (see chapter 9), in the orientation of the principal axis (when compared to unoccluded shapes) is 4.2 degrees with a standard deviation of 4.5. The deviations for the missile are somewhat less than this figure. This use of object features serves to decrease the number of possible combinations of image and model edge vectors from an average of 400 (from 20 object edges and 20 model edges) to between 40 and 50 i.e. by about 90%. Further reductions in the number of hypothesised poses are achieved by using the difference in the coordinates of the centroids of the image and model to constrain the translations Δx, Δy and ratio of the square root of the projected area of image and model to constrain scale factors.

Every possible object-model edge vector combination for the selected database record is taken in turn, and if the conditions described above are satisfied, the parameters of the RST transformation are computed. Initially the resulting 4-D transform parameter space will consist of many different combinations of model and object edge vectors. It is not possible for any one object edge vector to correspond to more then one model edge vector. Conversely it is not possible for any model edge vector to match to more than one object edge vector. The number of unique object-model edge vector pairs in the space is computed and if this is below a predetermined threshold then any further computation involving this particular database entry is terminated; matching then proceeds by retrieving the next appropriate record from the database and forming the new 4-D transform parameter space. The threshold is set to be 30% of the number of model edge vectors; this is a reasonable value since, with less than this number of possible corresponding object-model edges it is unlikely that there is sufficient evidence in the image to support the hypothesis

The algorithm used to identify candidate clusters or transformations lies at the heart of the proposed method and therefore needs to be carried out efficiently. An approach which is hierarchical in nature is described and, rather than clustering in the 4-D transform parameter space directly, producing a 4-D histogram with its associated high storage requirements, a simple 2-D table structure is used.

The parameters of each computed candidate transformation are examined in turn. This information is entered into an initially empty table which comprises five columns, s, θ, Δx, Δy and a count. If a given combination of parameters is not present then that combination is entered and the count initialised to one. If already present then the only step needed is to increment the appropriate count. In practise, a single record in the table will not represent a single transformation but rather, is a bin containing a group of transformations whose parameters lie within a specified range of each other. Bins are identified by their bin number.

This is illustrated by the following simple example. Consider the data below which comprises the 4-D transformation parameter space:-

s	θ	Δx	Δy
1.5	70.0	-2.5	-7.29
0.98	5.0	-0.01	-0.7
2.5	72.0	3.0	-8.0
1.01	6.0	0.02	0.2
0.99	5.5	0.01	0.1
1.05	6.5	-0.5	0.05

As an illustration, this data is to be binned up using one bin for s and θ and three bins for Δx and Δy, producing the following:-

For s :-

bin 0	**0.98**	$\leq$	**s**	$\leq$	**2.50**

For θ :-

bin 0	**5.0**	$\leq$	θ	$\leq$	**72.0**

For Δx :-

bin 0	**-2.5**	$\leq$	Δ**x**	<	**-0.667**
bin 1	**-0.667**	$\leq$	Δ**x**	<	**1.167**
bin 2	**1.167**	$\leq$	Δ**x**	$\leq$	**3.0**

For Δy :-

bin 0	**-8.0**	$\leq$	Δ**y**	<	**-5.267**
bin 1	**-5.267**	$\leq$	Δ**y**	<	**-2.537**
bin 2	**-2.537**	$\leq$	Δ**y**	$\leq$	**0.2**

Looking at the first entry in the cluster space it can be seen that the scale change of 1.5 falls into bin 0, the rotation angle of 70.0 falls into bin 0, the x translation of -2.5 falls into bin 0, and the *y* translation of -7.29 falls into bin 0. Thus after looking at the first entry the table will look like this:-

s bin	θ bin	Δx bin	Δy bin	count
0	0	0	0	1

A consideration of the second entry in the cluster space will give bin 0 for *s*, bin 0 for θ, bin 1 for Δ*x* and bin 2 for Δ*y*. The table will now look like this :-

s bin	θ bin	Δx bin	Δy bin	count
0	0	0	0	1
0	0	1	2	1

This procedure is continued until all entries in the cluster space are examined, producing a table which is given below:-

s bin	θ bin	Δx bin	Δy bin	count
0	0	0	0	1
0	0	1	2	4
0	0	2	0	1

It is easy to see the cluster in this small example. Of course, real world examples will be more complex but the basic procedure has been illustrated.

This procedure is actually carried out in a hierarchical fashion, binning up the data initially on a coarse scale, locating the candidate clusters at that level which are to be further binned in subsequent levels. A maximum number of such iterations has been set to be 5 and the binning is terminated when this is exceeded or when the standard deviation within individual bins falls within acceptable bounds or, if the number of unique object-model edge vector pairs in the modal bin falls below a predetermined threshold. This threshold has been set to be 30% of the total number of model edge vectors; this means that any viewed object in which less than 30% of model features are represented cannot be reliably recognised. In practise the process converges after only two or three iterations, at most, and cases which continue beyond this point are more than likely mismatches with the database. In this instance there will be very few possible object-model vector pairs in the final bin.

A series of pointers which refer back to the original data are used to keep track of which data is going into which bin and a series of overlapping bins are used in practice in order to prevent the splitting of clusters between bins. The overlap is notionally set to 0.25 of a bin width. The result is that rather than being allocated to a single bin, each quad of transform parameters is distributed to a set of overlapping bins.

When defining bin sizes it is important to keep in mind the likely magnitude of absolute errors which might result from errors in the positions of detected features due to the feature detection algorithm. The bin widths should never be smaller than twice this value. Also, the required precision of the final answer should be considered. If an accuracy of ±5 degrees in rotation angle is required then it is obviously insufficient to have a smallest bin width of less than 10 degrees.

It would clearly be advantageous if probable ranges of transform parameters to be encountered could be predicted with reasonable accuracy. For rotation angles the maximum range is obviously 360 degrees. For any given application, order of magnitude estimates of likely ranges in scale factor can be made. For example, if the smallest size image that can reasonably be dealt with is 64*64 pixels, then the largest magnification which can result is 8x, producing an image 512*512 pixels in size which is the maximum that many image processing systems can handle. The translation parameters present the only problem; their magnitudes depend upon the computed scale factor and rotation angle. It is thus difficult to predict, with any degree of certainty, the likely ranges of these two parameters.

The procedure which has been adopted is one of examining the data in the 4-D transform parameter space and setting up initial bin widths depending upon the range of the individual parameters. Rotation angles are binned using widths of 60, 30 and, finally, 15 degrees. Scale factors are initially binned using a width of 1.0. If too many bins result, the width is increased by 0.5 until a manageable number of bins are produced. (This is purely a reflection of the hardware limitations of the equipment). The progression is then from a width of 1.0 to 0.5 and finally to 0.25. Translations are binned using widths of 1000, 100, 50, 25 and 10 units. These translation bin widths represent a coarse-to-fine division of the translation space and final bin widths in scale and rotation angle are selected on the basis of the likely range in errors computed in section 7.5. At each level of the binning the range of the data is examined and the most appropriate bin width selected for each parameter. In each case the central sub-bin is centred over the mean of the data in the selected bin to minimise the splitting of transformations with similar parameters between bins.

In any level of the binning process, once all possible transform parameters have been computed and distributed to the table the next step is to examine the table and determine which bin, or bins, need to be further binned. Each bin is examined in turn and the number of unique object-model edge vector pairs computed. The data in the table is then sorted into descending count order. All records in the table whose counts then exceed the threshold value of 30% of the total number of model edge vectors are considered as candidates for subsequent binning in the next level. Often, records with high counts are close to one another, i.e. the four bins will either be coincident or adjacent to the associated bins in the other record. Only if candidate records in the table are greater than a predetermined threshold distance apart are they considered as separate clusters and both further binned. This threshold distance is set according to the relative magnitudes of the standard deviations of the data in each bin; a threshold value of ± 3 standard deviations is used.

In each level the non-unique object-model edge vector pairs are examined and those which are further from the current estimate of the mean of the data in the bin are removed from consideration. This is done by normalising the data in each bin to the range [0,1] and using a simple Euclidian distance measure to calculate the distance of each non-unique data quad from the mean of the data in the bin. In this fashion new estimates of the mean and standard deviation of the individual transform parameters are computed, and in the final level only the most appropriate data is used in the hypothesis verification stage of the matching process.

The number of unique object-model edge vector pairs in the finally selected bin is computed. If this figure is below 30% of the total number of model edge vectors in the current database record then the assumption is made that there is insufficient evidence in the image to support this hypothesis on object orientation. In this case the hypothesis is rejected and the next selected database record is retrieved and the new 4-D transformation parameter space formed. If this threshold value is exceeded, i.e. there exists enough supporting evidence in the image, then the next step is to evaluate how good this registration is - poor registration is indicative of an incorrect match and thus invalidates the hypothesis.

The data in the finally selected bin give an estimate of the RST transformation which will register the object data with the current database entry. An estimate of the transformation parameters is given by the mean and standard deviation of the data in the bin finally chosen. The 'goodness of fit' of the estimated transformation is evaluated by applying the transformation itself to the *x,y* coordinates corresponding to the model data in the final bin. The r.m.s difference between these transformed coordinates and those of the image is then found. This forms the basis for the final selection of the best match; after registering each selected model database record with the object data, the smallest r.m.s error gives the determined orientation of the sensed object.

Simple r.m.s. errors are, by themselves, inadequate in determining the most appropriate hypothesis. It is quite conceivable that identical r.m.s. values will result from two different hypotheses with different numbers of object-model edge vector pairs in the final bins. In such circumstances it is desirable to choose that hypothesis with the highest percentage of model edge vectors having interpretations in the sensed image. It is also important to consider the percentage of matched model boundary length. For this reason a weighting system has been devised which takes into account both the percentage of model edge vectors matched, l_m, and the percentage of the total boundary length which these edges represent, l_b. This involves dividing the raw r.m.s. values by the product $l_b\, l_m{}^4$. In practise, good results can be achieved using different combinations of these two values, however, this has been found to produce the most satisfactory results of a number of different weightings experimented with.

An overview of the matching process, including the retrieval of records from the database, is given by the pseudo-code algorithm below. /* */ encloses comments.

```
MAIN()
BEGIN
 initialise minimum r.m.s. value, minrms;
  fit_lines_to_boundary;  /* see section 5.2 */
 read value of t from image plane;
 compute minimum(min) and maximum(max) values of pratio
 to scan using equation (39);
 i = min;
 while (i < max)
     read count, m, from image plane at (pratio,1023);

     /* there are m records with area/perimeter$^2$ ratios */
     /* equal to the value, pratio */

     j = 1;
     while (j < m)
         lpos = 1023 - j;
         read pointer, p, from image plane at position (pratio,lpos);

         /* find location of record */

         col = (p*INCR) DIV LEN;
         line = MOD [(p*INCR) DIV LEN];

         /* retrieve record */

         read no. of vertices, nv, from image plane at position (col,line);
         length, len = nv*3 + 8;
        read array of length, len, from image plane at position (col,line);

         /* match retrieved model record to sensed image description */
         /*using a 2-D pose clustering procedure (see section 7.3) */

         pose_cluster;

         for (each selected bin)
            if (enough object-model vector pairs in bin)
              remove non-unique pairs;
              if (enough object-model vector pairs)

              /* back transform data in bin using the estimated*/
              /* transform parameters and compute the weighted */
              /* r.m.s. error of fit with the model data. If this */
              /* value is less than the current minimum, */
              /* minrms, then the current model record */
              /* becomes the new best estimate of object */
              /* orientation */

              back_transform;
              if (r.m.s. < minrms)
                minrms = r.m.s.;
                store orientation, transform parameters
                and the list of model-object edge
                vector correspondences;
              end if
            end if
          end if
         end for

         j = j + 1;
    end while
     i = i + 1;
 end while
END
```

```
PROCEDURE pose_cluster
BEGIN

 /* compute the parameters of the transformation for each */
 /* possible object-model edge vector pair satisfying the */
 /* constraints */

 form_cluster_space;
 if (enough object-model edge vector pairs)

    /* perform hierarchical binning of the 4-D parameter space */
    /* to identify clusters of transform parameters or the */
    /* hypothesised poses of the object in the sensed image */

    bin_up_data;
  end if
END
```

```
PROCEDURE form_cluster_space
BEGIN

  /* initialise size of cluster space */

  count = 0;

  /* the number of vertices in the model and object */
  /* descriptions are nm and nt, respectively */

  i = 1; j = 1;
  while (i < nm)
      while (j < nt)

          /* model edge i is formed by joining vertices */
          /* i and i+1. Object edge j is formed by joining */
          /* vertices j and j+1 */

          compare types of vertices i,i+1 and j,j+1;
          If(vertex types of object and model agree)
            compute orientation, $\theta_i$, of object edge j;
            compute difference between $\theta_i$ and orientation
            of model edge i, $\theta_m$. Call this $\theta$;
            if ($\theta$ within threshold)
              compute scale change, s, between object
              edge,j, and model edge, i;
              if (s within threshold)
                compute $\Delta x$, $\Delta y$;
                if ($\Delta x$ and $\Delta y$ within threshold)
                  add in the parameters s, $\theta$, $\Delta x$, $\Delta y$ to the 4-D space;
                  count = count + 1;
                end if
              end if
            end if
          end if
      end while
  end while
END
```

```
PROCEDURE bin_up_data
BEGIN

  /* initialise no. of iterations, nit, and maximum */
  /* number of iterations, nitmax, allowed */

  nit = 0;
  nitmax = 5;
  while ((nit < nitmax) or (standard deviations of transform
                  parameters exceed thresholds))

     /* divide up parameter space into bins and */
     /* allocate the data quads to them */

     allocate_data_to_bins;
     for (each selected bin)
       compute mean and standard deviation of transform
       parameters;
       allocate_data_to_bins;
     end for;
     nit = nit + 1;
  end while
END
```

```
PROCEDURE allocate_data_to_bins
BEGIN
 find min, max, range of each parameter;
 set up number of bins and bin widths;
 allocate the data quads to the table;
 find the number of unique object-model edge vector pairs in
 each record of the table;
 sort table in order of decreasing count of unique pairs;
 Examine table records with significant counts in order to see
 how close together they are - compute distance of nearest
 neighbour, ndist, to each selected bin;
 for (each bin where ndist > threshold)
   flag this table record as candidate transformation;
 end for
END
```

The use of object features and heuristics means that less than one third of the entries actually reach the stage where this hypothesis verification, or inverse transformation, is required. This is in itself made more efficient by the use of yet another heuristic. In order to prevent complete computation for the worst matches an initial threshold on the weighted r.m.s. value is set. As soon as this is exceeded the summing and squaring operations are ceased. Once a complete pass through a data set has been achieved and a new minimum weighted r.m.s. value obtained this then becomes the new threshold value.

Experience suggests that as little as 5% of the database entries selected reach this final stage in the matching process.

Object recognition is realised through matching, in exactly the same fashion, to a single, multi-object database where the object descriptions for each individual object are written to the image plane simultaneously. In this case the best match both identifies the object as well as determining its orientation.

In some instances, especially where objects are heavily occluded, the matching process may fail because the area/perimeter^2 ratio of the sensed object directs the search to incorrect regions of the database. This is quite a rare occurrence but is easily remedied; if a satisfactory match is not found then matching is reinitiated with a global search and comparison of the database.

7.4. Error Analysis

An analysis of how errors in the detected positions of the object features affect the calculated RST transformation has been performed. Assuming that the model feature positions are known exactly, the errors in the transformation parameters will be due only to errors in the coordinates derived in the image frame. Errors in the image rotation parameters can best be reduced by considering the rotation of the longest line segments available in the image. Values are given for the mean inter-feature distances (line segment lengths) which are required in order to obtain certain accuracies for given errors in the position of detected features. In the light of these results, a means of reducing the effect of these errors to a minimum is given.

Consider the case in which it is required to determine the orientation and scaling of a 2-D image of some object with respect to a model view of that object. The object can be shifted, scaled and rotated in the plane of its image frame. It is possible to solve for these transformation parameters if any two known correspondences of model and image features can be identified. Given that features at $\overrightarrow{r_{1m}} = (x_{1m}, y_{1m})$ and $\overrightarrow{r_{2m}} = (x_{2m}, y_{2m})$ in the model map onto features at $\overrightarrow{r_{1i}} = (x_{1i}, y_{1i})$ and $\overrightarrow{r_{2i}} = (x_{2i}, y_{2i})$ in the image, the rotation, θ, scaling, s, and shift of the image, Δx, Δy, relative to the model can be determined from the four simultaneous equations

$$x_{ij} = s_1 x_{mj} - s_2 y_{mj} + \Delta x \qquad (45a)$$

$$y_{ij} = -s_2 x_{mj} - s_1 y_{mj} + \Delta y \qquad (45b)$$

for $j=1,2$ and where $s_1 = s\cos(\theta)$ and $s_2 = s\sin(\theta)$.

These are solved to give

$$s_1 = \frac{(x_{2i} - x_{1i})(x_{2m} - x_{1m}) + (y_{2i} - y_{1i})(y_{2m} - y_{1m})}{(y_{2m} - y_{1m})^2 + (x_{2m} - x_{1m})^2} \qquad (46a)$$

and

$$s_2 = \frac{(y_{2i} - y_{1i})(x_{2m} - x_{1m}) + (x_{2i} - x_{1i})(y_{2m} - y_{1m})}{(y_{2m} - y_{1m})^2 + (x_{2m} - x_{1m})^2}. \qquad (46b)$$

Assuming that the model feature positions are known exactly, then the errors in the transformation parameters will be due to errors in the coordinates determined in the image frame. In particular, the errors in s_1 and s_2 will depend only on the errors in the separation $\overrightarrow{r_{2i}} - \overrightarrow{r_{1i}}$. This error, $\delta(\overrightarrow{r_{2i}} - \overrightarrow{r_{1i}})$, is a vector of length, ε, oriented at some angle e, (see figure 21), and has components ($\varepsilon\cos(e)$, $\varepsilon\sin(e)$). The separation, $\overrightarrow{r_{2m}} - \overrightarrow{r_{1m}}$ has components ($L_m \cos(\theta_m)$, $L_m \sin(\theta_m)$). (The errors in the shifts, Δx and Δy, will be affected by the absolute errors in $\overrightarrow{r_{1i}}$ and $\overrightarrow{r_{2i}}$).

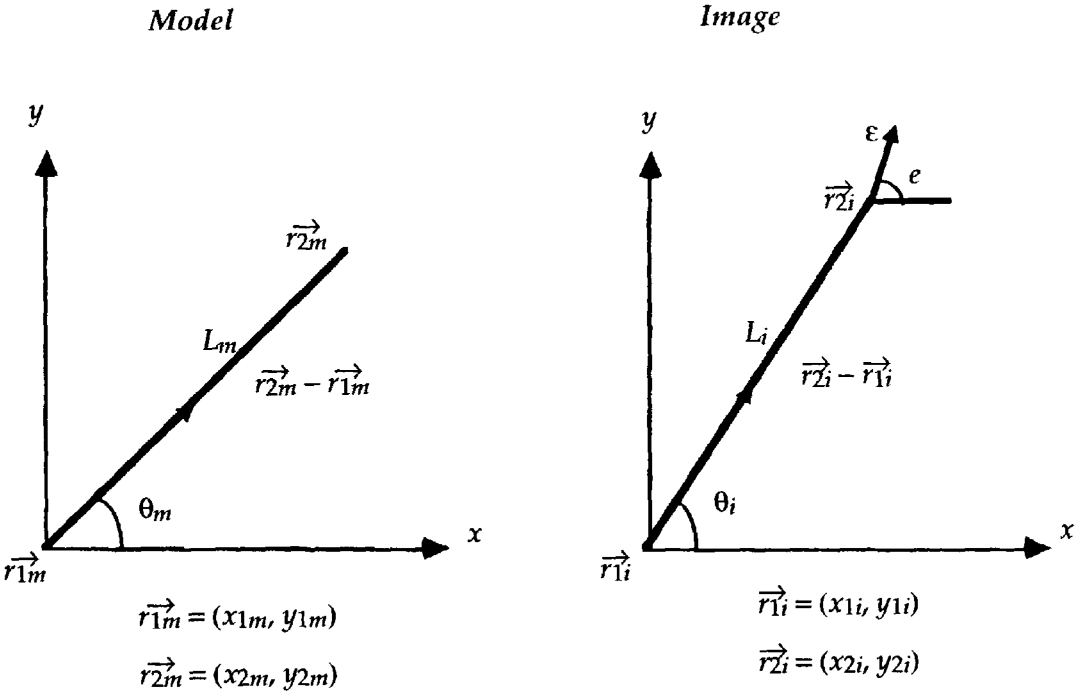


Figure 21. Definition of the error in the separation of two image features.

Then, the resulting errors in s_1 and s_2 are found to be

$$\delta(s_1) = \frac{\varepsilon \cos(e)\, L_m \cos(\theta_m) + \varepsilon \sin(e)\, L_m \sin(\theta_m)}{L_m{}^2}$$

which reduces to

$$\delta(s_1) = \frac{\varepsilon \cos(e - \theta_m)}{L_m} \qquad (47)$$

and

$$\delta(s_2) = \frac{\varepsilon \sin(e)\, L_m \cos(\theta_m) - \varepsilon \cos(e)\, L_m \sin(\theta_m)}{L_m{}^2}$$

which reduces to

$$\delta(s_2) = \frac{\varepsilon \sin(e - \theta_m)}{L_m}\,. \qquad (48)$$

These errors in s_1 and s_2 need to be translated into errors in the actual transformation parameters, s and θ.

Firstly, consider the scale factor, s, given by $s^2 = s_1{}^2 + s_2{}^2$. It can be seen that, to first order,

$$2s\ \delta(s) = 2s_1\ \delta(s_1) + 2s_2\ \delta(s_2)$$

$$s\ \delta(s) = s_1\ \delta(s_1) + s_2\ \delta(s_2)\ .$$

Using equations (47) and (48) gives

$$s\ \delta(s) = s\ \cos(\theta)\ \frac{\varepsilon}{L_m}\ \cos(e - \theta_m) + s\ \sin(\theta)\ \frac{\varepsilon}{L_m}\ \sin(e - \theta_m)$$

$$= \frac{s\ \varepsilon}{L_m}\ \cos(\theta + \theta_m - e)\ .$$

Therefore, from (41),

$$\frac{\delta(s)}{s} = \frac{\varepsilon}{L_i}\ \cos(\theta + \theta_m - e)\ . \qquad (49)$$

Now, consider the rotation angle, θ, given by $\tan(\theta) = \frac{s_2}{s_1}$. Then, to first order,

$$\sec^2(\theta)\ \delta(\theta) = \frac{s_1\ \delta(s_2) - s_2\ \delta(s_1)}{s_1{}^2}$$

which, using (47) and (48), gives

$$\delta(\theta) = \frac{\varepsilon}{L_m}\left(\frac{\cos(\theta)\ \sin(e - \theta_m) - \sin(\theta)\ \cos(e - \theta_m)}{s}\right).$$

Using equation (41)

$$\delta(\theta) = \frac{\varepsilon}{L_i}\sin(e - \theta_m - \theta)$$

or

$$\delta(\theta) = -\frac{\varepsilon}{L_i}\sin(\theta + \theta_m - e)\,. \qquad (50)$$

7.5 The Effect of Errors

In general, an estimate of the accuracy of the estimated scaling and rotation parameters is required, given some expected uncertainty in determining the positions of features used for registration.

From the analysis given in section 7.4 it can be seen that, to first order, the fractional error in the scale factor, $\frac{\delta(s)}{s}$, and the absolute error, in radians, in the rotation angle, $\delta(\theta)$, have r.m.s. values given by $\frac{\varepsilon}{\sqrt{2}L_i}$ with ranges of $\pm\frac{\varepsilon}{L_i}$, where ε and L_i have the same definitions as before.

It will, even under near perfect conditions, be difficult to locate features to much better than ±1 pixel simply due to the discretisation of the image. Under more realistic conditions, with noise, blurring etc., a more realistic estimate would be about ±2 pixels. However, since ε represents the error in the separation $\vec{r_{2i}} - \vec{r_{1i}}$ a more representative value would be about $2\sqrt{2}$ or ≈ 3 pixels.

Then, for a length, L_i, of 10 pixels it is found that $\frac{\delta(s)}{s} \approx 15\%$ and that $\delta(\theta) \approx 9°$ with a range of $\sim \pm 17°$. These errors reduce to $\frac{\delta(s)}{s} \approx 6\%$ and $\delta(\theta) \approx 3°$ with a range of $\sim \pm 7\,°$ for a length, L_i, of 25 pixels. For a length, L_i, of 50 pixels $\frac{\delta(s)}{s} \approx 3\%$ and $\delta(\theta) \approx 2°$ with a range of ~ ± 3 °. For a length, L_i, of 100 pixels $\frac{\delta(s)}{s} \approx 1.5\%$ and $\delta(\theta) \approx 1°$ with a range of~ ± 2 °. Assuming typical image dimensions of some 200 x 200 pixels with, on average, of the order of 20 features to be identified then the expected inter-feature separations will be ~22 pixels. (The expected distance to the nearest neighbour of a given feature is $\frac{1}{2\sqrt{f}}$ where f is the number of features per unit area, $5\text{x}10^{-4}$ in this case[63]). Thus typically $\frac{\varepsilon}{L_i}$ might be expected to be ~3/22 ≈ 0.14. To first order then, the expected error in θ will be $\delta(\theta) \approx 0.14 \sin(\theta + \theta_m - e)$ giving an r.m.s. error of $\sim \sqrt{0.5} \times 0.14 = 0.1$ radians or 6° with a range of some ± 8°. Similarly, the error in the scaling factor, s, will be $\frac{\delta(s)}{s} \approx 0.14 \cos(\theta + \theta_m - e)$ with again an r.m.s. of 0.1. In other words, ±10% errors can be expected in s. Obviously, features separated by smaller distances will have even larger errors.

However, the parameters of the RST transformation are not derived from single image-model edge segment pairs, but rather from many different combinations of model and image edges. The final result is improved through the number of independent estimates thus obtained. For n independent estimates a $\sqrt{n}$ improvement in the size of the errors results. For example, in an image containing 20 point features, these will be connected by 20 edges. However, since the features are each connected by two edges not all of the 20 estimates obtained are strictly independent; a large error in the position of one point feature will produce changes in the lengths and orientation of the two connecting edges. Since the resultant rotations will tend to be of opposite sign, however, a simple $\sqrt{n}$ improvement is likely to be an adequate approximation. This could improve the mean errors in rotation angle and scale to ~1.2° and ~3% respectively for the given example.

However, it has been shown that shorter lines give rise to larger errors. In fact, assuming a $\sqrt{n}$ improvement from n independent estimates, a single edge of length $L_i = 100$ will produce errors of the same size as approximately 4 edges of length $L_i = 50$, 16 edges of length $L_i = 25$ and 100 edges of length $L_i = 10$.

The practical implications of this are as follows. In order that errors are minimised the longest line lengths available should be used. Since the errors in the transformation parameters depend upon $\frac{\varepsilon}{L_i}$, then each estimate could be inversely weighted with its error, i.e. in proportion to its length, L_i. Indeed, if little weight is given to the parameters derived from short edges, then there is actually little value in including them. Simply deleting short lines is inadvisable since by doing so some features which are 'close' to one another will be removed from consideration by the clustering algorithm; these may be important in deriving orientation.

The procedure which has been adopted is to form a series of virtual edges, that is edges which connect features that are not directly linked by real edges in the image. In this way every detected feature can be included, whilst choosing virtual edges such that the edge lengths are maximised. In this way errors in the parameters of the resulting RST transformation are minimised and thus the estimate of object orientation is more reliable.

The magnitudes of the likely errors in the transformation parameters have implications for the minimum bin widths selected when clustering. Obviously, these should never be smaller than twice the magnitude of the expected error.

CHAPTER EIGHT

Reconstructing Partially Occluded Object Boundaries and Refining the Match

Having found a match as described in chapter 7 it is a simple task to reconstruct those parts of the sensed object which are either missing or unmatched with model data[64]. Whilst it may seem a somewhat superfluous activity and that the resulting object recognition and orientation determination is sufficient by itself, there is no method of checking whether the final match is actually a correct interpretation of the sensed object, as chapters 9 and 10 will show. Reconstruction of partially matched shapes to produce complete object descriptions, as described in this chapter, may be combined with subsequent matching to a database of global shape descriptions, e.g. Fourier descriptors, in order to answer this question and to refine the match.

8.1 Information Preserving Methods of Describing Shape

Many different methodologies have been developed to describe the shape of regions and these can be classified according to several different criteria. A fairly fundamental distinction may be made between those techniques which allow some degree of shape recovery from the shape descriptors and those which do not. These are termed information preserving and infonnation non-preserving, respectively[61].

Quantities such as width, elongation, the area/perimeter^2 ratio and other simple measures of shape were amongst the first to be used. These methods are simple to implement but are limited since widely different shapes can produce identical descriptors. They do not allow recovery of the original object shape and are thus information non-preserving.

Two widely used information preserving methods are moment-based techniques and Fourier descriptors[1,2,3,4,5,6,7,8,9,10]. The latter allow reconstruction of the original shape to any degree of accuracy, depending upon the number of coefficients of the Fourier series that are retained as shape descriptors. However, it is not easy to relate higher order moments to shape and the disadvantages of such global methods, see chapter 3, have already been mentioned.

The method used here, which describes object shape in terms of the local geometry of the boundary results in a piece-wise linear approximation which is, by its very nature, information preserving and, since it is derived from an analysis of local features, is able to produce accurate object descriptions for both noisy data and cases of partial occlusion.

Its chief advantages lie in the fact that the procedure is robust to poor feature detection i.e. 'missing' features and 'false' features. In this way, a match can still be found even if some model features are missing from the image and if spurious features are found. It is this fact which enables the method to cope with both noisy and partially occluded objects.

8.2 Reconstructing the Object Boundary

The matching algorithm produces an estimate of object orientation along with a list of correspondences between the matching object features and model features. The associated list of quads of transform parameters provides estimates of the mean and standard deviation of the parameters of the RST transformation which maps detected image features onto the model description. The list of correspondences is used to determine which model features have not been matched with features of the object. These unmatched features are then split into a series of groups, each group lies between two matched model features. For each group the following procedure is then adopted. The unmatched model features are to be transformed into the frame of the image such that a complete description of the object exists in the image frame. In order to do so the derived transformation, given by equation (44), is applied to the model features.

An iterative technique is adopted, minimising the distance between the end points of each group of transformed model features and the corresponding matched object features. The actual parameters of the transformation used may take on any values in the range of ±3 standard deviations away from the mean. Thus, local estimates of the global transformation are used in order to achieve the best possible reconstruction.

Where object features have been matched the original boundary is retained, being replaced by appropriately transformed model data where no match is achieved due to corrupt or missing data. The result is a reconstruction consisting partly of real and partly of model data.

Example results of reconstruction are shown in figures 22 to 25. In each case the left hand image shows the noisy occluded object, in these examples with ~60% of boundary pixels perturbed. The central image shows the reconstructed object and, to enable subjective comparison, the right hand image shows the original object.

Experimental results given in table 3, show that Fourier descriptors are unreliable when applied to partially occluded boundaries. However, when all of the boundary is present they produce acceptable results. In an attempt to quantitatively evaluate the quality of the reconstruction Fourier descriptors are used to determine object orientation both prior to reconstruction and on the reconstructed images; these are illustrated in table 3. The poor performance of Fourier descriptors on occluded objects can be seen. The precision of the line fitting and the accuracy of the edge segment matching algorithms enables accurate reconstruction of the object; the improvement in the errors α, β is at once obvious. These represent the error, in degrees of arc, of the orientation of the major, (or longitudinal) and minor axis of each object, respectively. (Reasons for their use are given in chapter 9).

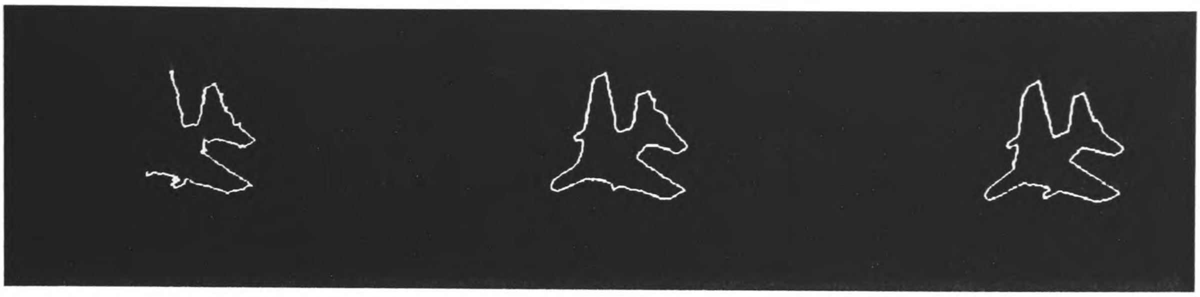


Figure 22. Occluded, reconstructed and original images of the Hawk jet trainer at 45, 11, 39.

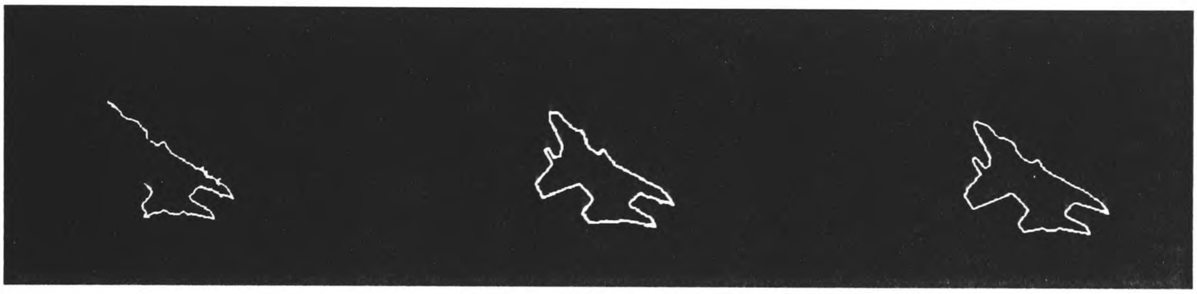


Figure 23. Occluded, reconstructed and original images of the Hawk jet trainer at 42, 44, 300.

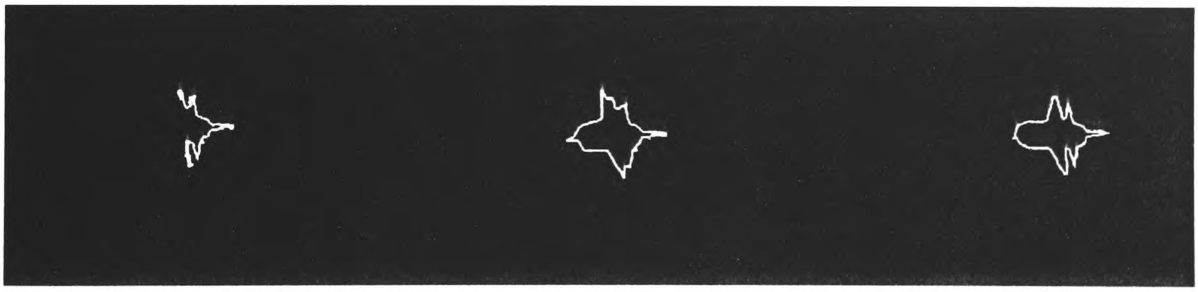


Figure 24. Occluded, reconstructed and original images of the missile at 82, 39, 359.

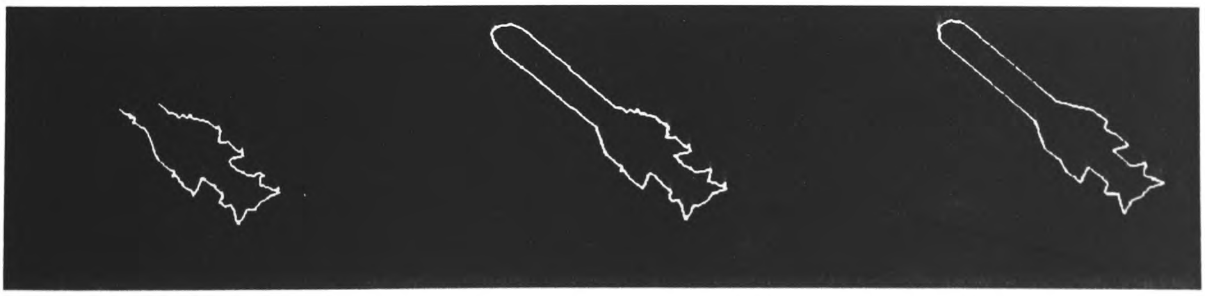


Figure 25. Occluded, reconstructed and original images of the missile at 33, 3, 315.

object	true orientation			% occ.	sensed orientation of occluded object using Fourier Descriptors			error in degrees of arc		sensed orientation of occluded object using line-segment matching			error in degrees of arc		sensed orientation of reconstructed object using Fourier Descriptors			error in degrees of arc	
	yaw	roll	pitch		yaw	roll	pitch	α	β	yaw	roll	pitch	α	β	yaw	roll	pitch	α	β
Hawk	45	11	39	30	70	15	30	25.4	15.4	50	5	34	6	4.7	55	13	55	10.3	8.5
Hawk	42	44	300	35	40	80	305	4.3	32.9	40	40	296	3.6	1.6	45	45	297	3.7	4.8
missile	82	39	359	40	60	25	97	32.1	112.9	80	40	356	2.1	4.1	85	15	327	4.6	9.0
missile	33	3	315	30	50	5	318	17.1	1.1	30	5	317	3.5	2.1	35	5	316	2.2	1.6

Table 3. Experimental results for the objects shown in figures 22-25.

As results to be presented in the next chapter will show, the line segment matching method can determine the orientation of objects with ~60% of boundary pixels perturbed by added noise and with up to 30% of the boundary removed to within an average uncertainty of 7 degrees of arc in both of the objects major and minor axes. This enables accurate reconstruction of noisy and occluded objects.

Furthermore the method is inherently information preserving; by maintaining a list of all of the unmatched edges in the selected model data these can easily be restored, leaving the matched edges unchanged. The result is an accurate reconstruction of the partially occluded object which comprises both real and model data.

CHAPTER NINE

Experimental Results

A series of experiments was conducted to test the performance of the system described compared to that for geometric moments and Fourier descriptors. Moment invariants of both the silhouette and the boundary curve of objects and Fourier descriptors based both upon magnitudes only and the inclusion of phase information have been employed. Test images of the wire-frame models of the three objects in increasing levels of noise and occlusion have been utilised. Three different levels of applied noise have been used, resulting in an average of 25%, 46% and 60% of the boundary pixels being disturbed when compared to the original clean boundary. Four different levels of occlusion were examined, resulting in the removal of 10%, 20%, 30% and 40% of the boundary. These levels of occlusion were tested for 0%, 25%, 46% and 60% noise. In total, over 650 different test images were generated; just over half of these were used to test the system in conditions of partial occlusion. Each system was tested with identical sets of images. In addition, the line segment matching algorithm has been tested with several real images of both the Hawk and Concorde.

In the following analysis the error in the determined orientation is expressed in terms of the two angles α and β. These represent the error, in degrees of arc, of the orientation of the major, (or longitudinal), and minor axis of each object, respectively. This is done since there are non-uniquenesses in the three rotation angles used to generate the test views of the wire-frame models. In particular, for yaw angles of ±90 degrees the two rotations, roll and pitch, reduce to rotations about the same axis; at these yaw angles the longitudinal axis of the object is rotated into coincidence with the line of sight, or the z-axis of the coordinate system described in section 2.2. Simply using the difference in the three angles yaw, roll and pitch means that some apparently disparate orientations, at high yaw angles, are in fact quite close.

One of the problems with the use of moment invariants is the ambiguity of 180 degrees in the determination of pitch angle. This is a real uncertainty, resulting from the ambiguity of 180 degrees in the orientation of the approximating ellipse (see figure 4). The orientation of the principal axes of both model and test objects is constrained to lie within ±90 degrees of the horizontal. This means that for test objects whose principal axes differ in orientation from the model principal axes by greater than 180 degrees will be in error by this amount.

There is an additional ambiguity in roll angle about 45 degrees for the missile. This is because the missile possess an axis of symmetry about ±45 degrees roll which is not true of the Hawk or Concorde.

Fourier descriptors suffer from the same ambiguity in pitch and roll angle as moment invariants. For example, the orientations (yaw,roll,pitch) 70,10,45 and 70,80,45 are reflections about the longitudinal axis of the object(see figure 34, section 10.1). The only way in which the method of Fourier descriptors can distinguish between these two orientations is with the use of phase components. Using magnitudes only, roll angles are only correct roughly 50% of the time. With the use of phase information this percentage increases to about 90% for clean images. This percentage falls as noise levels increase from 84% for 25% noise, to 80% for 46% noise and 76% for 60% noise.

There exist no ambiguities in the determination of orientation using the line segment matching method; pitch and roll are uniquely defined for all objects tested at all orientations. Of course, in common with the human eye, the method will not be able to distinguish between orientations which result from real ambiguities due to object symmetry.

Figures 26 to 28 show bar graphs illustrating, for each study object, the mean error in the angles α and β, for the various techniques investigated in conditions of noise. Figure 29 compares the mean error in orientation determination, over all the study objects, for the methods under study.

Similar results for the performance of each technique under partial occlusion are shown in figures 30 to 33. The results shown are for an applied noise level of 46%. This gives a representative assessment of how each method stands up to the partial loss of shapes. Figures 26 to 29 give a good impression of how noise affects the precision of orientation determination; noise levels lower than 46% tend to produce slightly smaller errors and higher noise levels tend to produce slightly larger errors, as would be expected. However, the effects of noise are really secondary when substantial parts of an object are missing.

No results are shown for Concorde at 30% and 40% occlusion since the line segment matching algorithm began to break down for this object at these levels of occlusion and noise. Reasons for this are discussed in chapter 10. Thus, overall averages shown in figure 33 are formed only from the errors for the Hawk and missile at higher levels of occlusion. This should not necessarily be regarded as a negative result since it is ostensibly a purely academic question as to whether errors in orientation of 30 degrees or more constitute a superior result than one of failure to determine orientation at all.

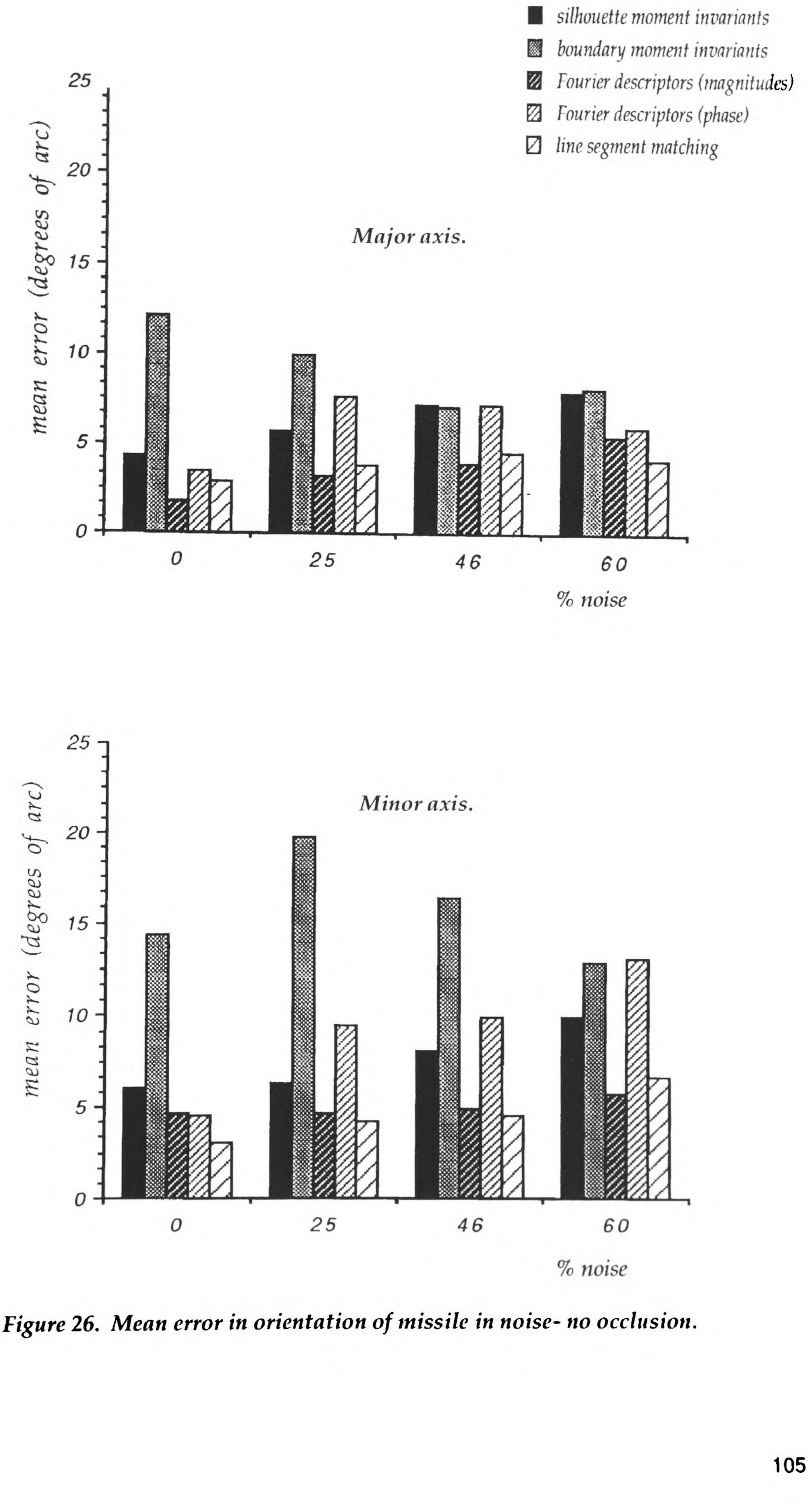


Figure 26. Mean error in orientation of missile in noise- no occlusion.

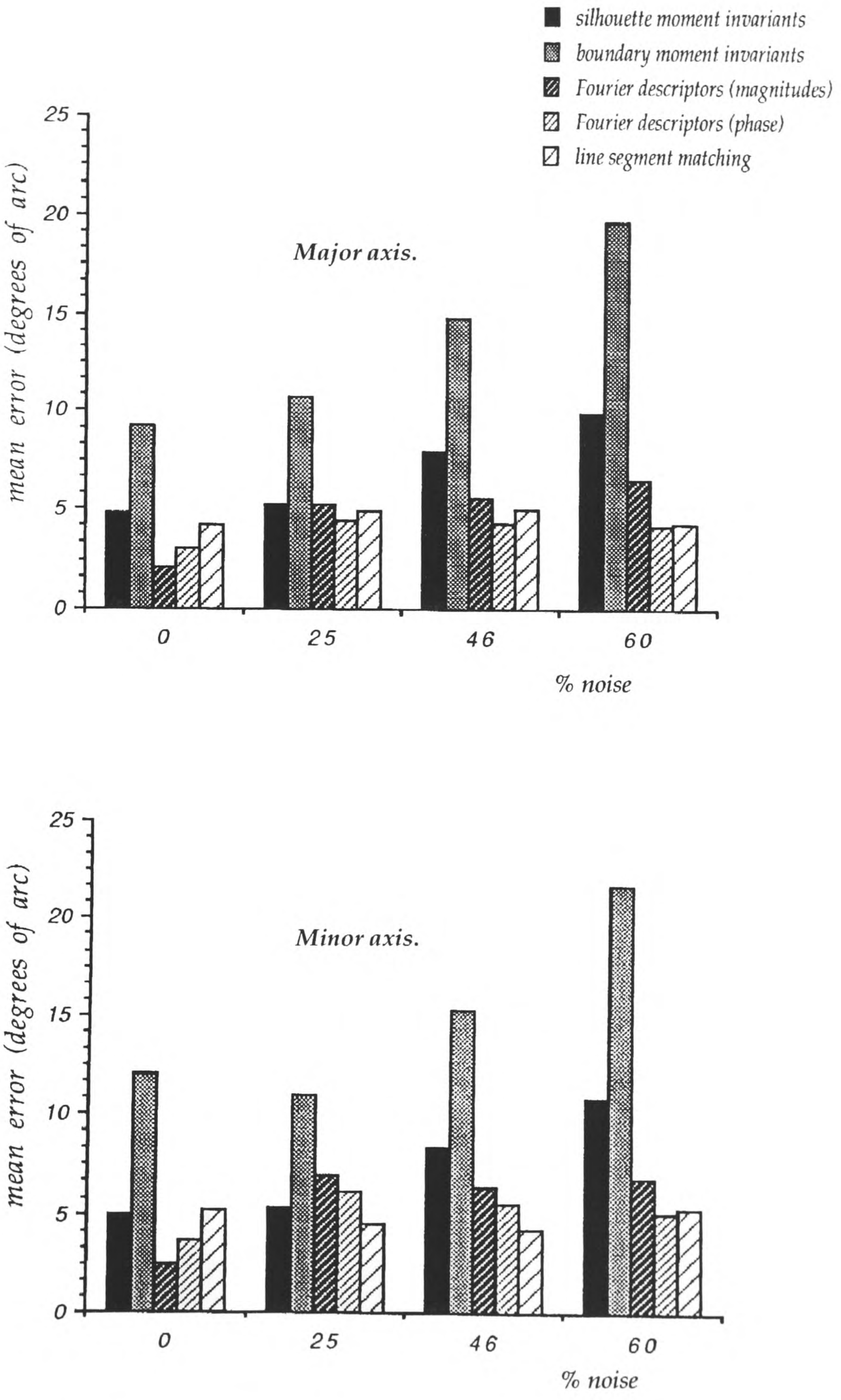
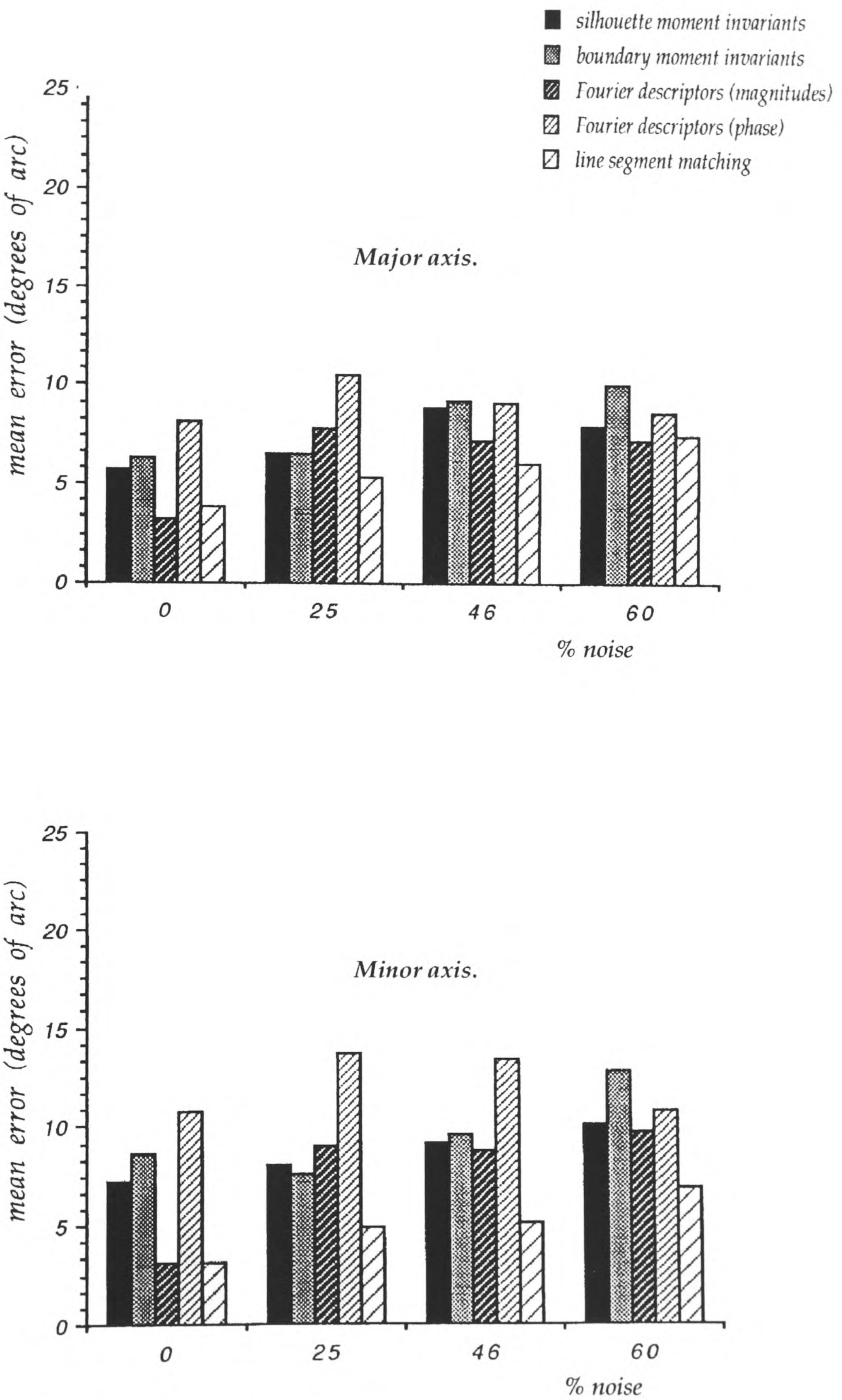


Figure 27. Mean error in orientation of Hawk in noise- no occlusion.


Major axis.

Minor axis.

Figure 28. Mean error in orientation of Concorde in noise - no occlusion.

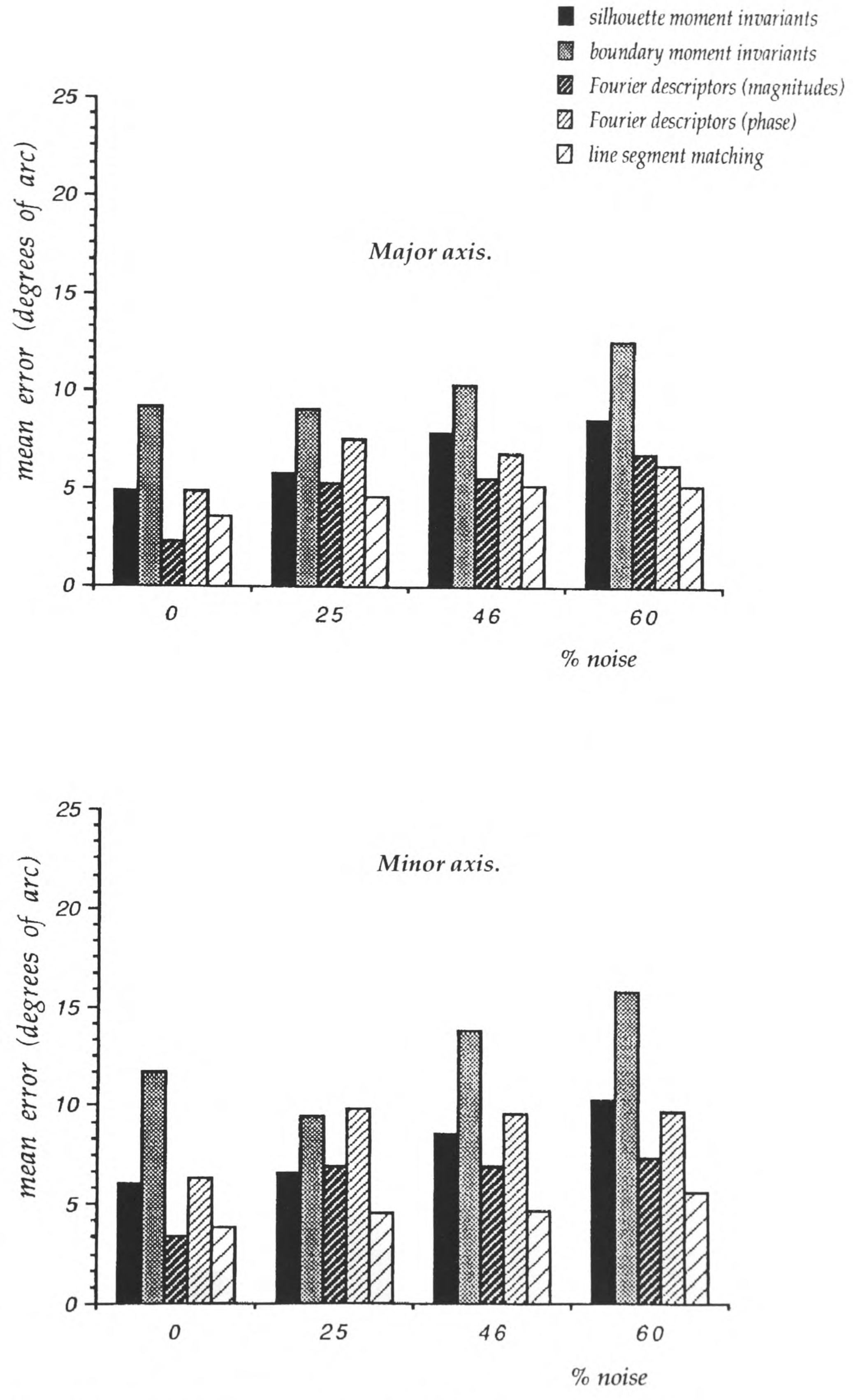


Figure 29. ***Overall mean error in orientation in noise - no occlusion.***

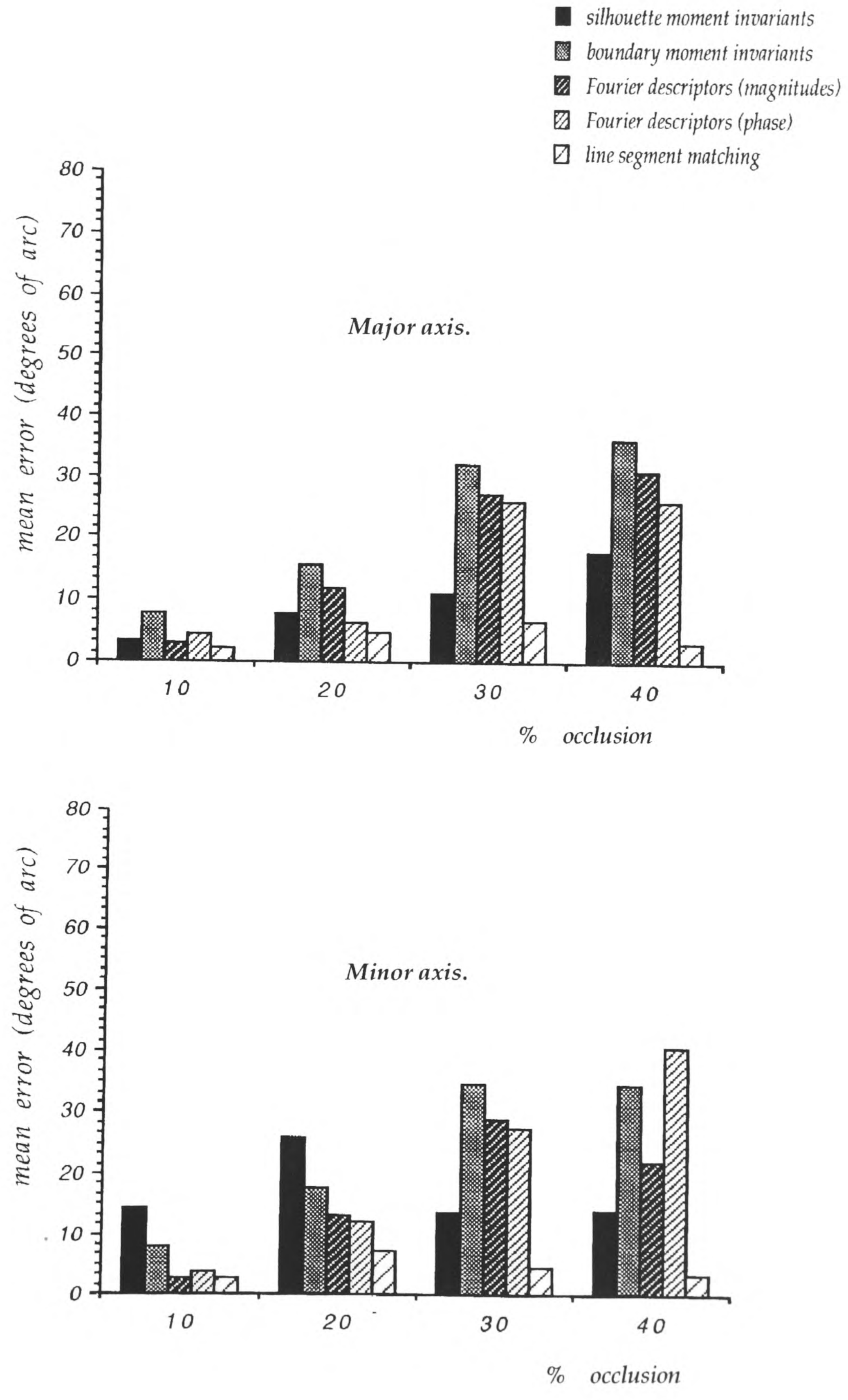


Figure 30. Mean error in orientation of missile in occlusion - 46% noise.

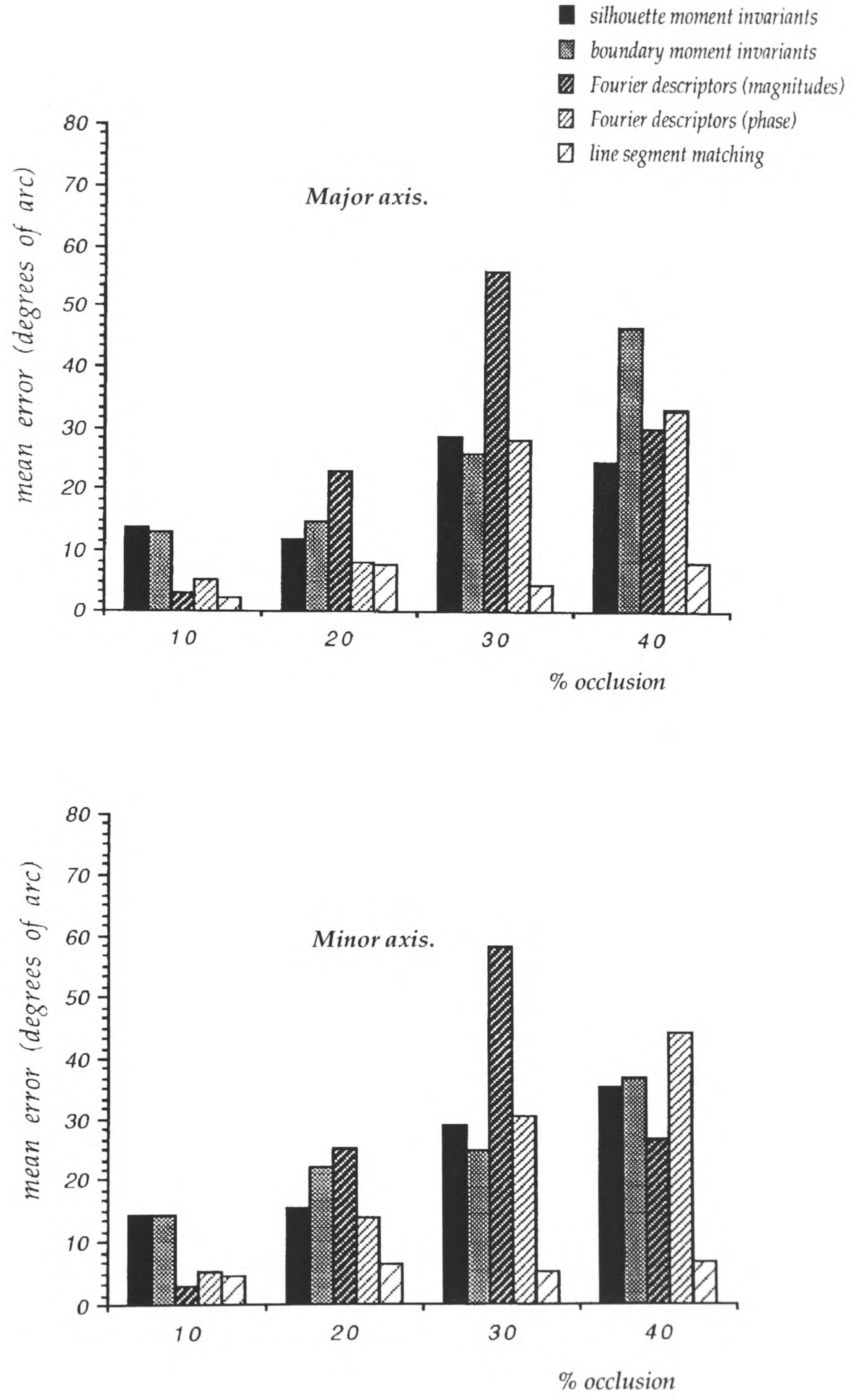


Figure 31. Mean error in orientation of Hawk in occlusion - 46% noise.

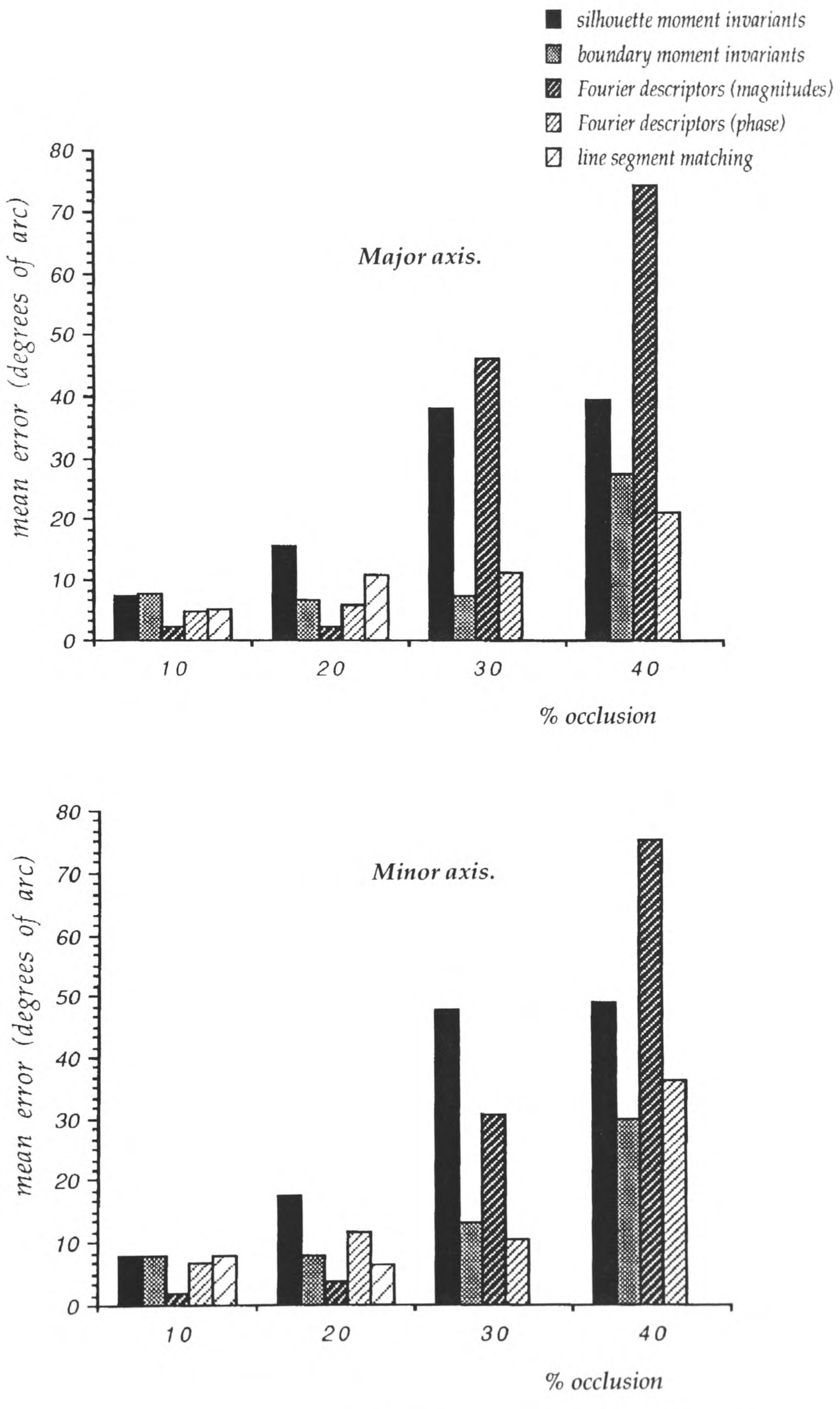


Figure 32. Mean error in orientation of Concorde in occlusion - 46% noise.

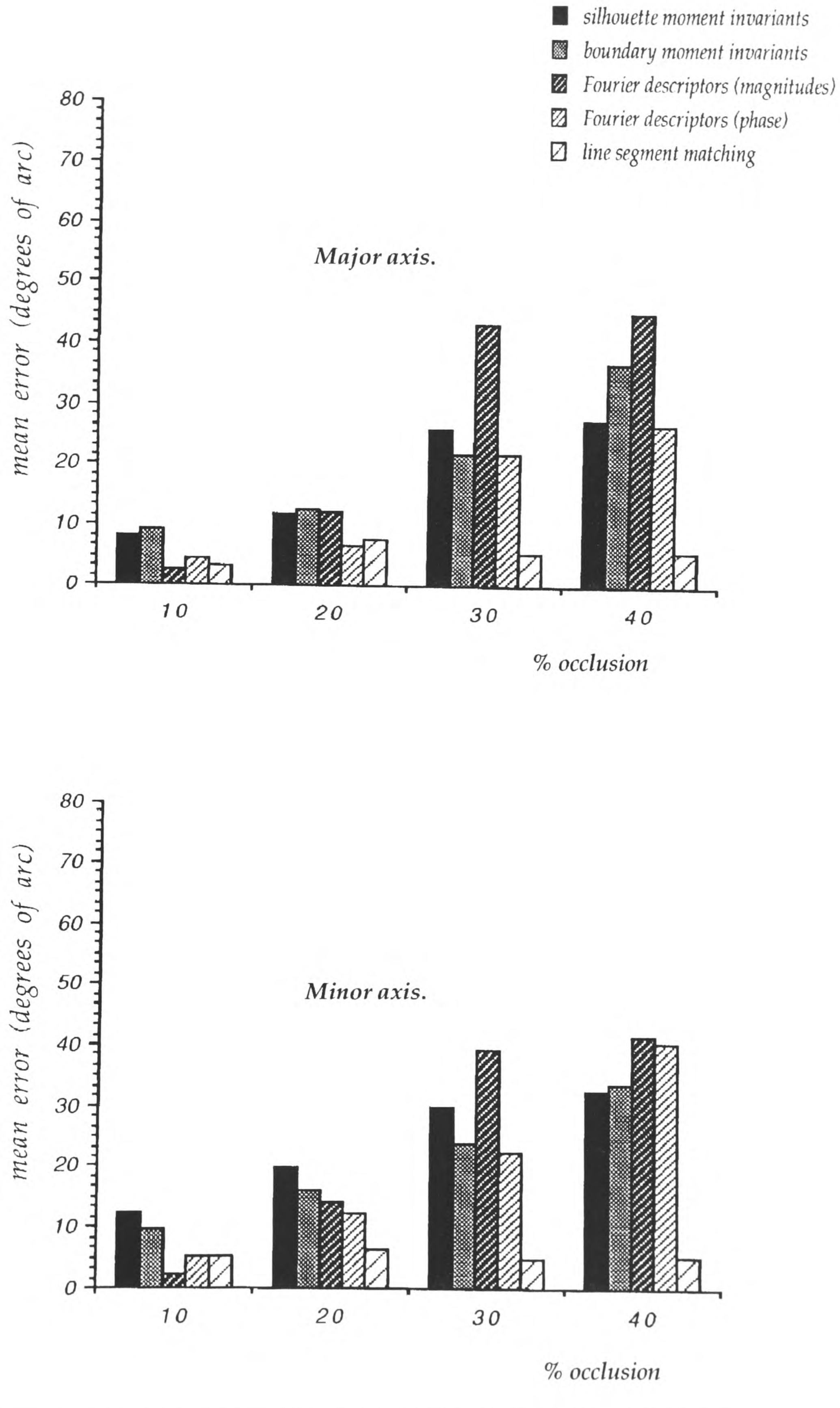


Figure 33. Overall mean error in orientation in occlusion - 46% noise.

object	true orientation			sensed orientation			error in degrees of arc	
	yaw	roll	pitch	yaw	roll	pitch	α	β
Hawk	30	0	90	20	5	90	10	5
Hawk	90	0	270	85	10	281	5	1.4
Hawk	0	30	235	5	40	238	5.8	9.8
Concorde	0	45	210	10	45	219	13.4	1.1
Concorde	45	0	180	55	10	182	10.1	8.4
Concorde	45	0	0	55	0	1	10	1

Table 4. Results for the line segment matching algorithm on real images of the Hawk jet and Concorde.

Table 4 shows a set of results which were obtained from real images of the model of the Hawk jet and Concorde. Images were captured as described in section 2.1.

The method always correctly recognises the object when orientation is determined. When more than 30% of the boundary is removed the line segment matching algorithm begins to break down. At 40% occlusion and with ~ 60% of the boundary pixels perturbed with added noise some 15-20% of sensed objects are unrecognised and therefore no estimate of orientation result. Table 5 gives overall average results for recognition accuracies in ~60% noise.

% occlusion	% recognised
10	100
20	91
30	87
40	81

Table 5. Recognition accuracies in 60% noise.

CHAPTER TEN

Discussion

10.1 Comparison of the Average Performance of the Methods in Noise

Analysis of figure 29 reveals that overall, the line segment matching algorithm produces the most accurate estimates of orientation for complete shapes in noisy conditions. In the noisiest of conditions, the 60% noise level, average errors of just over 5 degrees of arc in both the major and minor axes of the test objects result. This should be compared to errors of 6-7 degrees for Fourier descriptors using magnitudes, 6-10 degrees for Fourier descriptors including phase components, 9-10 degrees for moment invariants of the solid silhouette and 13-16 degrees for moment invariants of object boundaries. The method is surpassed only by that of Fourier descriptors using magnitudes of coefficients in noise-free conditions. In this case, errors of between 2.5-3.5 degrees result compared to approximately 3.5 degrees from the line segment matching algorithm.

The method of boundary moment invariants has the poorest performance of all; even in noise-free conditions mean errors of the order of 10 degrees result. This is hardly surprising since, despite the fact that most shape information is concentrated at the boundary, the influence of noise is also at its greatest here. This can be illustrated by the magnitude of the errors using this method for the Hawk jet (see figure 27). They rise from about 9 degrees for noise-free conditions to over 20 degrees for noise at the 60% level. The Hawk jet is by far the most complex study object and as such has the most complex boundary curve. Therefore, boundary moment invariants are subject to large errors. By contrast, the orientation of Concorde, which has a very simple shape, (which has been further simplified in the model such that most orientations are represented by around 10 vertices) can be determined to within ±13 degrees, on average, at the 60% noise level (although this is still poor in comparison to the other methods).

Generally, errors for Fourier descriptors using phase are larger than for Fourier descriptors using magnitudes only. This reflects the susceptibility of phase components to noise on the boundary. Since normalisation of coefficients for start point and rotation angle in the plane of the image is based on phase information, sub-optimal normalisation can result. Normalisation is also non-unique. Therefore, as noise levels increase, the likelihood of normalising about different phase coefficients increases. This in turn will lead to increased errors in the determination of orientation.

However, this should be viewed bearing in mind the fact that using magnitudes only means that for objects with more than one axis of symmetry, eg. the missile, some orientations are indistinguishable (figure 34). This ambiguity in orientation determination can be resolved with the use of phase components. This is successful up to 76% of the time when 60% of the boundary pixels are noisy, even if the magnitudes of the resulting errors increase slightly.

Missile at 70, 10, 45. *Missile at 70, 80, 45.*

Figure 34. Some orientations are merely reflections about an axis of the object.

Examination of figures 26 through 28 reveals that all methods, except for boundary moment invariants, produce smaller errors for the Hawk than for the Concorde at all levels of noise. The orientation of the missile can also be determined more accurately than that of Concorde, again the only exception being for the method of boundary moment invariants. This is explained by the relative levels of complexity in the shapes of the projected 2-D silhouettes of the various objects.

Comparing the errors for the missile and the Hawk, it can be seen that in general, they are larger for the Hawk than for the missile, except for the method of Fourier descriptors using phase. This reflects the effect of ambiguous normalisation in noise due to the extra axis of symmetry that the missile possesses. This affects roll angles rather more than yaw angles. For example, it can be seen in figure 26 that, for the method of Fourier descriptors using phase information, the errors in orientation about the minor axis of the object are much larger than those about the major axis. The increased precision in orientation determination for the missile when compared to the Hawk cannot be explained by the relative complexity of the two shapes; it would be expected that the Hawk jet, with its complex shape, would produce more accurate estimates of orientation. However, the only real 3-D element to the Hawk is the tail plane. This is not really an important feature in fine-tuning orientation estimates, it merely indicates the direction of yaw and roll angles (i.e. positive or negative). By contrast, the missile with its four wings and four tail fins extending form the body of the missile at right angles is inherently more 3-dimensional, and so produces more accurate estimates of orientation.

10.2 Comparison of the Average Performance of the Methods Under Partial Occlusion

Figures 30 to 33 demonstrate how global methods of describing shape fall down as objects become partially hidden. The line segment matching algorithm, by contrast, produces orientation estimates which are reasonably stable and within about 5 degrees of the true value, even when almost half of the boundary pixels are noisy and 40% of the boundary is missing. The only exception is for the Concorde when at these levels of noise and occlusion the matching algorithm begins to break down. This again may be explained by the simplicity of the model of Concorde. With only 10 vertices describing any orientation, removal of 40% of them leaves only 6 vertices with which to match. Corruption of just one or two of the remaining data points makes finding a match more unlikely than for the Hawk, say, with typical orientations having over 30 vertices. Indeed, accurate estimates of orientation, i.e. to within 7 degrees, can still be obtained for both the Hawk and missile when 60% of the boundary pixels are corrupted by noise and with 40% of the boundary removed.

For small levels of occlusion Fourier descriptors can still produce accurate estimates of orientation. The degree to which they fall down depends largely upon which particular portion of the shape is missing. The loss of much of the low frequency content, e.g. the fuselage of the missile, produces larger errors than the loss of higher frequency portions of the shape e.g. a wing or tail fin, even though these features may have more importance in fine-tuning orientation estimates.

Figures 35 and 36 display plots of cumulative frequencies versus error in yaw and roll angles of the model missile, for various portions of the shape being removed. No noise

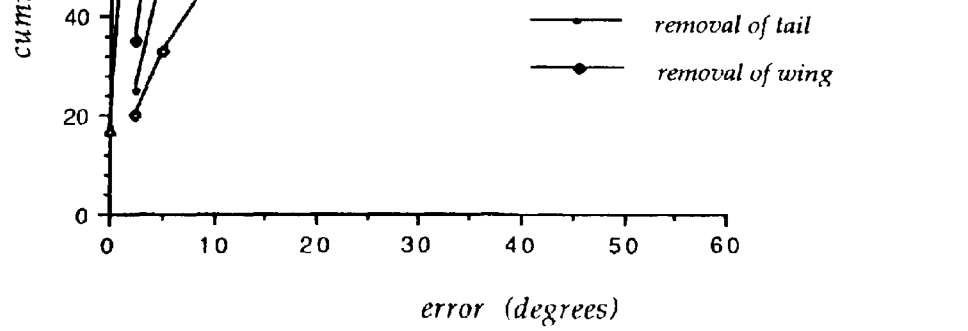


Figure 35 Plot of cumulative percentage of orientations versus error in yaw angle for the missile. (Plots show the percentage of orientations which were determined to within a specified accuracy).

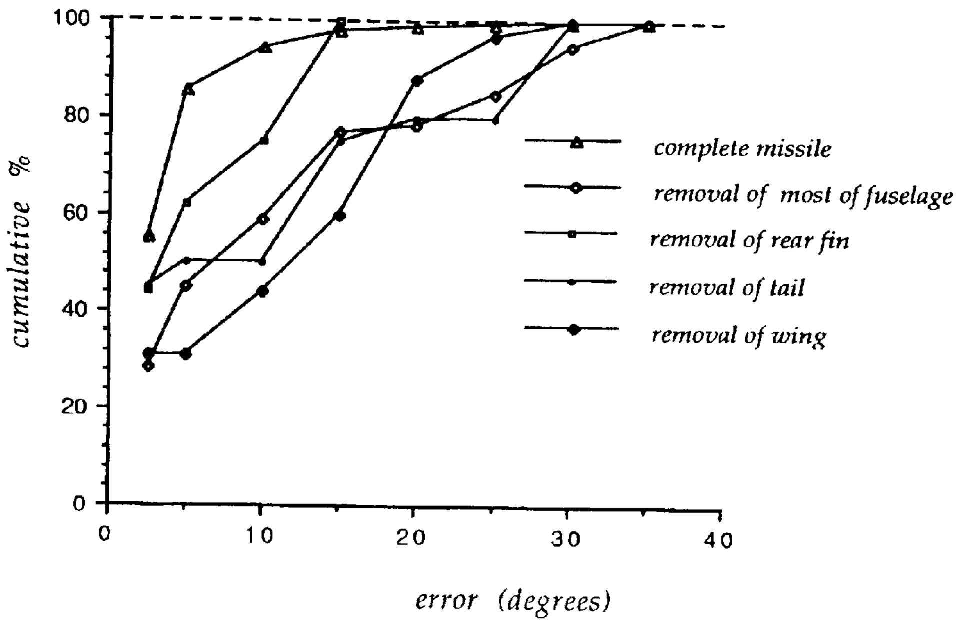


Figure 36 Plot of cumulative percentage of orientations versus error in roll angle for the missile. (Plots show the percentage of orientations which were determined to within a specified accuracy).

has been added to the boundaries. The plots show the percentage of orientations which were determined to within a specified accuracy. High yaw angles (over 75 degrees) have been excluded from the study in order to avoid the problems involved with using simple differences in rotation angles as error estimates at yaw values approaching 90 degrees. Each graph has been generated using the data from several hundred runs of the method of Fourier descriptors using magnitudes only. There are five curves which correspond to the complete shape, the removal of most of the fuselage, the removal of a wing, the removal of a tail fin and the complete removal of the tail, respectively. The graphs show that, when the complete shape is present, 98% of orientations were determined to within 5 degrees of yaw and 85.5% of orientations were determined to with the same accuracy in roll angle. The removal of a wing reduces these figures to 80% and 31% for yaw and roll respectively. This low percentage for roll angles reflects the loss of symmetry about the longitudinal axis that the removal of a wing produces; most of the resulting errors are in roll rather than yaw angles. The loss of a tail fin reduces the percentage of orientations which can be determined to within 5 degrees of yaw to 68% and to within 5 degrees of roll to 62%. The removal of the whole tail produces percentages of 50% for both yaw and roll. The importance of the low frequency content of shapes, or their 'bulk' properties is shown by the loss of the fuselage of the missile. Only 33% of orientations can be determined to within 5 degrees of yaw and only 45% of orientations are accurate in roll angle by this amount or less.

The match is resolved on the basis of two or three of the lowest frequency coefficients alone. These are of much larger magnitude than the higher frequency coefficients and so contribute proportionally much more to the sum of differences squared. True orientations are rejected on the basis of these two or three coefficients matching badly whilst the match over the remaining coefficients is very good. Thus selected orientations are not always the best match over the entire range of coefficients.

The line segment matching algorithm, being based on local features, is not susceptible to partial occlusion in the same way as Fourier descriptors. Object features responsible for fine-tuning orientation estimates are the most important factor. When these are occluded, orientation estimates necessarily degrade. The partial occlusion of 'bulk' portions of the shape, e.g. the fuselage of the missile, does not necessarily mean that large errors in orientation will result. As an example, consider the results given in chapter 8 (table 3 and figure 25) for the occluded missile at an orientation of 33, 3, 315.

Obviously, the orientation of the principal axis of an object will change if part of the shape becomes hidden for any reason. The amount by which it changes will depend upon which part of the shape is occluded (and by how much). The use of a threshold on the difference in the selected model and sensed principal axes compared to the computed rotation angle between the two is therefore not an 'infallible' method of decreasing or limiting the number of possible object-model correspondances.

In fact, this is perhaps the major reason for mis-matches to database records, or indeed for failure to match at all, since for heavily occluded objects insufficient object-model edge segment pairs will be admitted to the 4-D transformation parameter space.

The threshold value could be relaxed slightly to allow more object-model correspondances to be considered, although the disadvantage is that false clusters are more likely to occur.

There is no real pattern to the accuracy of orientation estimates with variations in yaw and roll. One observation is, however, that at low yaw and roll angles, where objects are viewed almost flat on, errors tend to be larger. Conversely, at high yaw angles, where objects project a much smaller area, the errors in orientation determination are consistently at their lowest. Even in noise at the 60% level and with up to 40% of an object boundary occluded, errors of 5 degrees or less are not uncommon. Also, for the Hawk and Concorde, yaw estimates tend to become more error prone at high roll angles. For the missile, with its extra axis of symmetry, errors in yaw tend to be symmetric about 45 degrees in roll, with a maximum occurring at 45 degrees, and minima occurring at around 22 or 23 degrees and 67 or 68 degrees roll.

10.3 The Line segment Matching Algorithm: Results from Real Images

The results given in table 4 show that reasonable estimates of orientation can be obtained from real images. The accuracies are good, bearing in mind that the wire-frame models used are, in fact, only approximations to the model objects used. In the case of the Hawk jet some object features are purely conjectured since the tail fins were in fact absent from the plastic model used in this study. Thus these results represent accuracies which could be realised in real world conditions where poor object segmentation, errors in feature detection and partial occlusion result.

10.4 The Line segment Matching Algorithm: Considerations on Effectiveness and Efficiency

One approach to the problem of object recognition involves the use of transform clustering, in which the underlying idea is to accumulate independent pieces of evidence for a match. Each possible pair of object-model features defines a possible transformation between the model and the object in the sensed image. Pairs that are part of the same correct match will produce similar transformation parameters whilst random combinations of model and object features will produce random transformation parameters. Thus, clusters of similar transformation parameters correspond to correct mappings between model and object.

The Generalised Hough Transform[19] represents one method of locating these clusters of transformation parameters. It works by quantising the parameter space into a series of discrete cells, buckets or bins. Each possible transformation is then mapped to a bin in this Hough space, indexed by its parameters. Thus, to locate clusters of transformation parameters all that is required is to search the Hough space to determine which bin, or possibly bins, contains the most entries.

In practise, however, several problems arise. Similar transformation parameters can end up in different bins if they lie on different sides of a quantisation boundary. As the dimensionality of the transformation increases, the size of the associated Hough space can quickly become unmanageably large making the search for clusters both time consuming and awkward. The success of the method depends upon there being a low likelihood of large clusters occurring at random; the quantisation of the Hough space makes this more likely to occur.

Two approaches can be adopted in order to reduce the effects of the first two problems. Either clustering is performed over small neighbourhoods centred upon each bin, or each parameter is distributed to a series of bins, often depending upon the probable magnitude of any errors associated with the computation of the transformation. Both of these approaches increase the number of entries in the Hough space and so increase the likelihood of large clusters occurring at random. Reducing the size of the Hough space to make the problem more tractable, i.e. making the bins larger, also has this effect. Often, especially for high dimensional parameter spaces, the Hough space is decoupled into a series of sub-spaces[31,32,35]. This also increases the likelihood of large random clusters and there is no guarantee that clusters in one projection of the space correspond to global clusters in the whole space.

The approach presented here keeps the dimensionality of the problem to a manageable size, using the Hough transformation parameter clustering approach to solve for the simple case of a 2-D transformation using point features. The inclusion of scale as a degree of freedom increases the dimensionality of the problem to four, solving now for the transformation between a model and the sensed object in its most general form. This is in contrast to most other work which usually assumes a known scaling between object and model.

Solving for 3-D orientation is realised through comparison to a database of 2-D model descriptions. This follows a hypothesise and test type of approach in which simple object features are used to direct the search of the database. Each selected database record represents a hypothesis on object orientation. Hypothesised orientations constitute, on average, less than 20% of the total database. Extensive use of object features controls the combinatorial explosion, reducing the *mn* possible transformations from *m* model and *n* object features by about 90%.

Using a spare quadrant of the image plane to store the database of model descriptions is both effective and efficient since the whole of the database is effectively held in RAM.

Each selected model database record is compared to the sensed description using a modified Hough transformation parameter clustering approach. A hierarchical approach, binning initially on a coarse scale and then sub-binning only those bins which contain large numbers of entries, immediately reduces the size of the accumulator array required. For example, a typical comparison between sensed and model data might lead, in the first level, to the creation of 10 bins for scale factor, 5 for rotation angle, 9 for Δx and 5 for Δy. This results in a parameter space of 10 * 5 * 9 * 5 = 2,250 bins in number. The second level of binning splits the selected bin into 3 * 3 * 2 * 3 = 54 bins and then, in the final level, into 3 * 1 * 3 * 3 = 27 bins. This should be compared to the 'brute-force' approach which would require

$$(10 * 3 * 3) * (5 * 3 * 1) * (9 * 2 * 3) * (5 * 3 * 3)$$

$$= 90 * 15 * 54 * 45$$

$$= 3{,}280{,}500 \text{ bins.}$$

The implications for this, both in terms of reduced storage requirements and the associated benefits in searching much reduced size parameter spaces, are self-evident.

Further, at each level of binning, rather than using a 4-D accumulator, a simple 2-D table structure is used. This has the effect of allocating storage only when new combinations of transformation parameters are encountered. Using a 'brute-force' approach and forming a 4-D histogram and incrementing the counts in the appropriate bins as each new object-model combination is considered leaves the vast majority of bins empty. This is obviously inefficient. The tabling approach presented here produces only 85 entries in the table at the first level of binning. This can be compared to the 2,250 bins of the hierarchical 4-D accumulator which would be required. In the second level 26 table entries result, compared to the 54 bins which would otherwise be required. In the final level the resulting 7 table entries can be compared to the 27 histogram bins which would be required. These figures represent 3.78%, 48.15% and 25.93% of the space which would be required at each level if the simple hierarchical binning approach was adopted. The advantages of searching for clusters of parameters in these even further reduced size spaces are indisputable.

A series of overlapping bins is used; each bin is allowed to overlap adjacent bins by 0.25 of a bin width. Each transformation parameter can therefore be allocated to 2 bins. This means that, as a worst-case, there is a redundancy in the allocation of quads of transform parameters of $2^4 = 16$, i.e. each transformation, resulting from a single object-model correspondence, can be allocated to a maximum number of 16 bins. In practise, this rarely occurs and the average number of bins per correspondence is more usually around 3 or 4 at the first level, reducing to 2 or less in the final level. In the example above these redundancies represent 0.17%, 5.55% and 6.48% of the total parameter space respectively. Whilst these redundancies appear to be quite large for the lower levels, much of the outlying data is removed in the initial level of the binning, thus greatly reducing the risk of large clusters occurring at random.

False clusters are easily refuted as candidate orientations at the stage of back transformation and computation of error-of-fit estimates. Any random clusters of transformation parameters will produce large r.m.s. errors at this hypothesis verification stage. Extensive use of the heuristics described in chapter 7 further enhances the efficiency of the method; only one third of the selected database entries reach this stage of the matching process. As levels of occlusion increase, leading to fewer compatible object-model correspondences, the fraction of selected database records requiring hypothesis verification decreases, sometimes to just a few instances.

Encouraging to note is that computed r.m.s. values are small for a range of orientations centred around the selected value; however, this means that orientations of the order of 10 degrees in error are sometimes selected in preference to orientations which are closer to the true orientation. This is on the basis of r.m.s. values which differ only in the second decimal place. One method of fine-tuning the results would be to partially abandon the partial computation of r.m.s. values, producing complete computations for the few percent of orientations producing the smallest r.m.s. values. The match resulting from the line segment matching could then be used to reconstruct the sensed object, as described in chapter 8, and Fourier descriptors of the reconstructed shape generated. These descriptions could then be matched against a library of Fourier descriptors, as outlined in chapter 3. The final orientation would be selected on the basis of the reconstruction which produces the smallest sum of differences squared when matched in this way. The figures shown in table 3 of chapter 8 suggest that this would be a viable proposition.

Occasionally, in high levels of noise and occlusion, a match results which is hopelessly wrong. These instances are not refuted by the hypothesis verification stage of the procedure, as would happen under normal circumstances, since often only a few equally bad matches reach this stage in the matching as described above. These instances are recognisable by their relatively high r.m.s. values. Reconstruction of these shapes with transformed model data and computing a match using Fourier descriptors, as outlined, would produce large sums of squares of differences and orientation estimates incompatible with the hypothesis. Thus incorrect hypotheses could be rejected.

Despite these minor drawbacks, this unique combination of a modified version of Hough transformation parameter clustering, with a directed search of a database, is a robust method of both recognising 3-D objects and determining their orientation to within an uncertainty of less than 7 degrees of arc, even when 60% of the boundary pixels are affected by noise and when 40% of the boundary is missing, or hidden.

10.5 Practical Considerations and Future Work

The method assumes that the object of interest is segmentable from the background and that a solid silhouette or continuous edge can be extracted. Edge detectors may result in a series of unconnected edgelets rather than a continuous, complete boundary.

The method of applying noise facilitates the maintenance of a connected boundary even with high noise levels. This raises the question of just how realistic the resulting noisy boundaries are. There are several methods of adding noise to images and of these Gaussian noise is the most commonly used. This can be added using a random number generator and the resulting image thresholded to produce a binary image from which the object silhouette and boundary are extracted. There is no reason why the method described should not produce results of similar accuracy for images generated in this way.

Quantisation of the database affects the precision of orientation estimates. Storing model descriptions at 5 degree intervals means that given perfect data we could expect, as a worst case, results to within 2½ degrees.

Bin widths affect errors in s, θ, Δx and Δy and hence r.m.s. errors and hence orientation estimates. The exact relationship between all of these factors has not been determined at this point. However, it is apparent that if bin sizes are chosen too small this will lead to the splitting of clusters of transformation parameters and hence to poor estimates of s, θ, Δx and Δy or to too few object-model edge vector pairs. Conversely, if bin sizes are chosen too large then spurious, outlying data will similarly affect s, θ, Δx and Δy. Currently bin widths are fixed as described in chapter 7, but bin widths for each parameter can vary independently of one another. These bin widths have been empirically determined. Further work is required in order to find a method of automatically determining optimum bin sizes.

Clearly model descriptions must be as accurate as possible. Anything less than this may result in degradation of orientation estimates.

Mention has already been made as to a method for improving the line segment matching, or transformation parameter clustering, algorithm using reconstruction of object boundaries and Fourier descriptors. Tentative attempts have revealed that this is likely to be a viable proposition, should these added refinements be required.

The method is dependent upon a good representation of shape having been derived. Analyses presented in this thesis have shown that some traditional approaches, e.g. curvature estimators, are not robust in the presence of noise and that alternatives need to be investigated. The method described in this thesis is based upon a class of approaches typified by those of Pavlidis[56,57,58] and has been demonstrated to be more robust in conditions of noise. This is one area which would benefit from further investigation, generalising the rather simplistic approach adopted involving only the intersection points of straight line segments to include e.g. arcs of circles and parabolas in conditions of noise; perhaps a probabilistic approach to the representation of object boundaries.

Improved accuracies in orientation estimation could be obtained if it were possible to assign weightings to model features according to their importance in determining orientation. This should not really present a problem if the right CAD or solid modelling software were readily available. The procedure to be adopted would be to transform the wire-frame models into each required orientation and then to identify the respective features of the 2-D projection of the boundary, e.g where faces of the wire-frame intersect etc. A count would be kept of the number of times each model feature occurred. Those features occurring most often are not important in determining orientation, although they might well be significant in recognising the object. Thus, heavier weightings would be assigned to those model features which occurred least often and low weightings to those model features with the highest number of occurrences. This has implications for the determination of orientation of partially occluded objects. The successful matching of relatively few features which are of high importance in tying down orientation estimates should carry more significance than the matching of maybe one or two more features in total which can be seen over a wide range of orientations.

The method is easily adaptable to work for more than one object in a scene, should this be required. In this case the matching process would proceed in the same fashion, but would necessitate the following up of more cluster centres, or bins, than at present. Alternatively, once a match with a model object has been achieved, the matching features could be removed from the image and the process repeated. Matching would terminate when all features in the sensed image have either successfully been assigned correspondences with one of the database models or have been discovered to have no interpretation in terms of these stored descriptions.

There is probably a limit on the number of models which could be included in the database. This is not particularly for reasons of storage since the method of using image plane quadrants is an efficient method of holding the complete database, effectively in RAM. Rather, the problem arises when several very similar models are of interest. The method, in common with others, might have some difficulty in distinguishing between such cases, especially as levels of noise and occlusion, in particular, increase. As long as objects are sufficiently different from one another then there should be no real problems of this kind.

No attempts have been made to minimise the run-time of the algorithms used, indeed, intermittent synchronisation problems have necessitated the slowing-up of execution in many cases. However, on the PC-based equipment described in section 2.1 average recognition times range between tens to a few tens of seconds.

There is much inherent parallelism in both the generation of object descriptions and the pose-clustering modules, as well as in the search of the database. Therefore, one direction in which to move in the future would be the parallel implementation of the work described.

CHAPTER ELEVEN

Conclusions

It has been shown that whilst global descriptors of shape, such as moments and Fourier descriptors, produce satisfactory estimates of orientation for complete shapes, they are not suitable when objects are partially occluded.

Local methods of describing shape, where distortion of part of the object affects only the descriptors associated with that particular region, and nowhere else, provide a more successful solution to the problem.

The local object descriptions used here comprise point features which are derived from the analysis of the trends in boundary curves. This results in representations which are more stable in the presence of noise than those which are derived from algorithms for computing curvature directly.

This unique combination of a modified version of Hough transformation parameter clustering, with a directed search of a database, is a robust method of both recognising 3-D objects and of determining their orientation to within an uncertainty of less than 7 degrees of arc, on average, even when 60% of the boundary pixels are affected by noise and when 40% of the boundary is missing, or hidden. The pose-clustering is both effective and efficient. Heuristics and object features control the combinatorics and computational load and also direct the search of the database. A hierarchical approach, combined with the use of a simple 2-D table structure, rather than a 4-D accumulator array, substantially reduces the storage requirements of the algorithm. The table typically uses less than 4% of the space which would be required at the first level of binning if the simple hierarchical approach was adopted. Savings of 75% are not uncommon in the lower levels.

Using a spare quadrant of the image plane to store the database of model descriptions is both effective and efficient since the whole of the database is effectively held in RAM.

This work shows that several fairly simple techniques can be successfully organised to solve a very difficult problem.

REFERENCES

1. **Hu, M.K. 1962** *"Visual Pattern Recognition by Moment Invariants"*, IRE Trans. Inform. Theory, vol.IT-8, pp179-187.

2. **Zahn, C.T. and Roskies R.Z. 1972** *"Fourier Descriptors for Plane Closed Curves"*, I.E.E.E. Trans. Computers, C-21, pp 269-281.

3. **Granlund, G.H. 1972** *"Fourier Preprocessing for Hand Print Character Recognition"*, I.E.E.E. Trans. Computers, C-21, pp195-201.

4. **Uesaka,Y. 1984** *"A New Fourier Descriptor Applicable to Open Curves"*, Electronics and Communications in Japan, vol 67-A, pp167-173.

5. **Dudani, S.A., Breeding, K.J. and McGhee, R.B. 1977** *"Aircraft Identification Moment Invariants"*, I.E.E.E. Trans. Computers, C-26, no.1, pp39-46.

6. **Reeves, A.P., Prokop, R.J., Andrews, S.E. and Kuhl, F.P. 1984** *"Three-dimensional Shape Analysis Using Moments and Fourier Descriptors"*, Proc. 7th Int. Conf. Pattern Recogniton, Montreal, Canada, July 30 - Aug. 2.

7. **Smith, F.W. and Wright, M.H. 1971** *"Automatic Ship Photo Interpretation by the Method of Moments"*, I.E.E.E. Trans. Computers, C-20, pp1089-1094.

8. **Sadjadi, F.A. and Hall, E.L. 1979** *"Object Recognition by Three-dimensional Moments"*, Proc. Pattern Recognition and Image Processing Conf., Chicago, Ill.,Aug. 6-8 pp327-336.

9. **Richards, C.W. and Hemami, H. 1974.** *"Identification of Three-dimensional Objects using Fourier Descriptors of the Boundary Curve"*, I.E.E.E. Trans. Systems, Man, Cybern. SMC-4, pp371- 378.

10. **Wallace, T.P. and Wintz, P.A. 1980** *"An Efficient Three-dimensional Aircraft Recognition Algorithm Using Normalised Fourier Descriptors"*, Computer Graphics and Image Processing, vol 13, pp 96-126.

11. **Teh, C.H. and Chin, R.T. 1985** *"On Digital Approximation of Moment Invariants"*, Proc. I.E.E.E. Computer Soc. Conf. Computer Vision Pattern Recognition, pp640-642.

12. **Teh, C.H. and Chin R.T. 1988** *"On Image Analysis By The Method Of Moments"* , I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-10, no.4, pp496-513.

13. **Rice, J.R. 1969** *The Approximation of Functions,2*, Addison-Wesley,Reading.

14. **Lin, C.C. and Cellappa, R. 1983** *"Classification of Partial 2-D Shapes Using Fourier Descriptors"*, I.E.E.E. Trans. Pattern Anal. Machine Intell. PAMI-5, pp653-660.

15. **Gorman, J.W., Mitchell, O.R. and Kuhl, F.P. 1988** *"Partial Shape Recognition Using Dynamic Programming"*, I.E.E.E. Trans. Pattern Anal. Machine Intell. PAMI-2, no 2, pp257-266.

16. **Wallace, T.P., Mitchell, O.R. and Fukunaga, K. 1981** *"Three-dimensional Shape Analysis Using Local Shape Descriptors"*, I.E.E.E. Trans. Pattern Anal. Machine Intell. PAMI-3, no.3, pp310-323.

17.**Fischler, M.A. and Bolles, R.C. 1981** *"Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography"*, Commun. ACM, vol 24, no 6, pp381-395.

18.**Ballard, D.H. and Brown, C.M. 1982** *Computer Vision*, Prentice-Hall.

19.**Ballard, D.H. 1981** *"Generalising The Hough Transform To Detect Arbitrary Shapes"*, Pattern Recognition, vol.13, no.2, pp111-122.

20.**Kittler, J. and Illingworth, J. 1985** *"A Review of Relaxation Labelling Algorithms"*, Image and Vision Computing, vol. 3, pp206-216.

21.**Davis, L.S. 1979** *"Shape Matching Using Relaxation Algorithms"*, I.E.E.E. Trans. Pattern Anal. Machine Intell. PAMI-1, pp60-72.

22.**Bolles, R.C. and Cain R.A. 1982** *"Recognising And Locating Partially Visible Objects: The Local-feature-focus Method"*, Int. Journal Robotics Res., vol 1, pt.3, pp57-82.

23.**Ayache, N. 1983** *"A Model-based Vision System To Identfiy And Locate Partially Visible Industrial Parts"*, Proc. Computer Vision Pattern Recognition, June 19-23 pp492-494.

24.**Koch, M.W. and Kashyap, L. 1987** *"Using Polygons To Recognise And Locate Partially Occluded Objects"*, I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-9, no 4, pp483- 494.

25.**Grimson, W.E.L. and Lozano-Perez, T. 1985** *"Recognition and Localization of Ovarlapping Parts from Sparse Data in Two and Three Dimensions"*, Proc. I.E.E.E. Conf. Robotics and Automation, St. Louis pp61-66.

26.**Van Hove, P. 1987** *"Model-Based Silhouette Recognition"*, I.E.E.E. Workshop on Computer Vision, Miami Beach, pp88-93.

27.**Stockman, G., Kopstein,S. and Benett, S. 1982** *"Matching Images to Models for Registration and Object Detection via Clustering"*, I.E.E.E. Trans Pattern Anal. Mach Intelligence, PAMI-4, No.3, pp229-241.

28.**Stockman, G. and Agrawala, A. 1977** *"Hough Curve Detection is Equivalent to Template Matching"*, Commun. ACM, vol 20, no 11, pp820-822.

29.**Stockman, G. 1987** *"Object Recognition And Localisation Via Pose-clustering"*, Computer Vision, Graphics and Image Processing, vol. 40, pp361-387.

30.**Stockman, G. and Esteva, J.C. 1985** *"3-D Object Pose from Clustering With Multiple Views"*, Pattern Recognition Letters, vol.3, pp279-286.

31.**Ballard, D.H. 1983** *"Viewer Independent Shape Recognition"*, I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-5, pp653- 660.

32.**Thompson D.W. and Mundy, J.L. 1987**, *"Three-dimensional Model Matching from an Unconstrained Viewpoint"*, Proc. I.E.E.E. Conf. Robotics and Automation pp208-220.

33. **Turney, J.L., Mudge, T.N. and Volz, R.A. 1985** *"Recognising Partially Occluded Parts"*, I.E.E.E. Trans Pattern Anal. Machine Intell., PAMI-7, no.4, pp410-421.

34.**Dhome, M. and Kasvand,T. 1987** *"Polyhedra Recognition By Hypothesis Accumulation"*, I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-9, no.3, pp429-438.

35.**Linnainmaa, S., Harwood, D. and Davis, L.S. 1988** *"Pose Determination Of Three-dimensional Objects Using Triangle Pairs"*, I.E.E.E. Trans. Pattern Anal. and Machine Intell., PAMI-10, no.5, pp634-647.

36. **Horaud, R. 1987** *"New Methods for Matching 3-D Objects with Single Perspective Views"*, I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-9, no.3, pp401-412.

37.**Lowe, D.G. 1987** *"Three-dimensional Object Recognition from Single Two-dimensional Images"*, Artificial Intelligence, vol.31,pp355-395.

38.**Lowe, D.G. 1985** *Perceptual Organisation and Visual Recognition*, Kluwer.

39.**Foley, J.D. and Van Dam, A. 1982.** *Fundamentals of Interactive Computer Graphics*, Addison-Wesley.

40.**Pavlidis, T. 1982** *Algorithms for Graphics and Image Processing*, Springer-Verlag.

41.Theoretical Weapon Attitude Study, Report No. 92.53/4, M.O.D.(PE), R.A.E., Jan. 1986.

42.**Teague, M.R. 1980** *"Image Analysis via the General Theory of Moments"*, J. Opt. Soc. Am., vol 20 no. 8, pp920-930.

43.**Abu-Mostafa, Y.S. and Psaltis, D. 1983** *"Image Normalisation by Complex Moments"*, Proc. I.E.E.E. Soc. Conf. Computer Vision and Pattern Recognition, June 19-23 1983, Washigton D.C., pp114-120.

44.**Abu-Mostafa, Y.S. and Psaltis, D. 1984** *"Recognitive Aspects of Moment Invariants"*, I.E.E.E. Trans. Pattern Anal. Macine Intell., PAMI-6, no.6, pp698-706.

45.**Otsu, N. 1979** *"A Threshold Selection Method from Gray Level Histogram"*, I.E.E.E. Trans. Sys., Man, Cybern. SMC-9, pp62-66.

46.**Attneave, F. 1954** *"Some Informational Aspects Of Visual Perception"*, Psychological Review 61, pp183-193.

47.**Rosenberg, B. 1972,** *"The Analysis of Convex Blobs"*, Computer Graphics, Image Processing, vol.1, pp183-192.

48.**Rosenfeld, A. and Johnston, E. 1973** *"Angle Detection On Digital Curves"*, I.E.E.E. Trans. Computers, C-22, pp875-878.

49.**Rosenfeld, A. and Weszka, J.S. 1975,** *"An Improved Method Of Angle Detection On Digital Curves"*, I.E.E.E. Trans. Computers, C-24, pp940-941.

50. **Davis, L.S. 1977** *"Understanding Shape: Angles and Sides"*, I.E.E.E. Trans. Computers, C-26, no.3, pp236-242.

51.**Shirai, Y. 1978** "*Recognition Of Real-world Objects Using Edge Cues*", in Computer Vision Systems (A Hanson, E.M. Riseman, Eds.), Academic Press, New York. pp353-362.

52. **Chien, C.H. and Aggarwal, J.K. 1989** "*Model Construction And Shape Recognition From Occluding Contours*", I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-11, no.4, pp372-389.

53. **Langridge, D. 1972**, "*On the Computation of Shape*", in Frontiers of Pattern Recognition, S. Wanatabe (Ed.), Academic Press, New York.

54.**Teh, C.H. and Chin, R.T. 1989** "*On The Detection Of Dominant Points On Digital Curves*", I.E.E.E. Trans. Pattern Anal. Machine Intell., PAMI-11, no 8, pp859-872.

55.**Ramer, U. 1972** "*An Iterative Procedure for the Polygonal Approximation Of Plane Curves*", Computer Graphics, Image Processing, vol.1, pp244-256.

56.**Pavlidis, T. 1973**, "*Waveform Segmentation Through Functional Approximation*", I.E.E.E. Trans. Computers, C- 22, pp689-697.

57.**Pavlidis, T. and Horowitz, S. 1973** "*Piece-wise Polygonal Approximation of Plane Curves*", Proc. IJPCR, vol.1, pp396-405.

58. **Pavlidis, T. 1982**, *Structural Pattern Recognition*, Springer-Verlag.

59. **Horn, B.P.K. 1986**, *Robot Vision*, M.I.T. Press.

60. **Fairney, P.T. 1989** "*Recognition and Orientation of Specular Objects from the Analysis of Plane Polarised Light* ", Ph.D. Thesis, Polytechnic of Wales.

61. **Pavlidis,T. 1980** "*Algorithms for Shape Analysis of Contours and Waveforms*", I.E.E.E. Trans. Pattern Anal. Machine Intell. , PAMI-2, 4, pp301-312.

62. **Illing, D.P., Fairney, P.T. and Wiltshire, R.J. 1990** " *3-D Object Recognition and Orientation from both Noisy and Occluded 2-D Data*", Proc. BMVC90, Oxford University, 24-27 Sept., 1990.

63. **Phillips, S.J. 1990** *Private communication*

64 **Illing, D.P. and Fairney, P.T. 1990** "*Reconstructing Partially Occluded Object Boundaries*", Pattern Recognition Letters, In Press (to be published shortly).

3-D Object Recognition and Orientation from both Noisy and Occluded 2-D Data.

D.P.Illing*, P.T.Fairney+ & R.J.Wiltshire,*

*Dept. of Mathematics and Computing,
+Dept. of Computer Studies,
The Polytechnic of Wales,
Pontypridd,
Mid Glamorgan,
CF37 1DL.

We present a method of recognising 3-D objects and determining their position and orientation from single 2-D images of both noisy and partially occluded shapes.

Our approach combines analytical and structural-type techniques, adopting a 2-D pose-clustering method combined with a comparison to a database of 2-D model descriptions for various object orientations. Object features are used both to control the amount of computation and to direct the search of the database.

Features derived from an analysis of the local geometry of the object boundary are used in matching. Local features are preferable to global techniques which perform poorly in noisy conditions and where the object boundary is partially occluded. In these circumstances, local features provide sufficient information to enable the object to be recognised and to determine orientation to within less than seven degrees of arc, on average, in both the major and minor axes of the object.

1. Introduction

Determining 3-D object orientation from single distorted 2-D image frames is a problem in computer vision/pattern recognition.

In our study we have used airborne objects, e.g. aircraft. We have assumed orthographic projection, such that very little, if any, perspective is present in the image, and that only the silhouette projection of the viewed object will be available to us. The method described is not confined to such objects and can be easily adapted to solve for different viewing geometries. However, it is not a general purpose technique because different 3-D objects may project the same 2-D silhouette. In practice this is unlikely to be the case and therefore storing a library of such projections to represent the different orientations of 3-D objects is a feasible approach.

Early work[1,2,3,4,5,6] used global descriptors of shape such as 2-D and 3-D moment invariants and Fourier descriptors; these have been used to recognise or classify 3-D objects from 2-D silhouettes. Given an arbitrary view of a known object, or one of a small set of objects, a set of shape descriptors are computed for the silhouette which are then matched against a library of precomputed descriptors.

These global methods give good results for complete shapes not distorted by noise or occlusion. Moments[7] are particularly susceptible to noise. Neither method is capable of representing local information with any degree of accuracy[8] and both degrade rapidly when part of an object's boundary is missing or occluded. As a result there has been a move towards local descriptions of shape, i.e. corners[9,10], holes[10], lines [11] and curvature[12].

The generalised Hough transform[13] is commonly used [11,14,15,16,17]. Such approaches involve an accumulation of evidence which either supports or weakens a hypothesis on the pose of an object in the image. Objects are recognised by means of a rotation, scaling, translation (RST) transformation which maps observed image features onto features in the model. This is achieved by assuming that the object in the image can be represented by an instance of the model having undergone some RST transformation and that the object is rigid. Even with such limitations the approach is viable. A match is still possible even if some model features are missing, and if spurious features are found. As a result the method is able to cope with both noisy and partially occluded objects.

Only the object's 2-D silhouette is available and we have therefore chosen boundary segments as our local features. These have been used previously to register aerial imagery with maps, to recognise instances of objects in aerial photographs and to match 2-D models to images of 2-D industrial parts[11].

We use wire frame models of a Hawk jet trainer and a missile to generate both model descriptions and test data for our system. Analytic techniques are used to derive structural descriptions of object boundaries, in terms of angles and straight line segments. These features are then used in a pose-clustering process. When this is combined with matching to a database of precomputed model descriptions we are able to recognise and determine the unconstrained orientation of a 3-D object from a single image of its 2-D silhouette projection with an accuracy of less than seven degrees, on average. The position of the object within the field of view is unimportant, as is the scale of the object with respect to its model. Given the focal length of the camera then the resulting scale factor can be used to estimate object range.

We use object features to limit the combinatorics of the clustering and to reduce the *im* combinations of *m* model segments and *i* image segments by appproximately 90%. By using simple measures of shape to direct the search of the database, less than 20% of the model database entries are selected as possible poses for our matching algorithm. Heuristics limit the number of hypothesised poses that are verified to about 30% of the selected model database entries, and of these only 15% will complete this final stage.

2. The Pose-clustering Paradigm

Consider figure 1 which shows a single image vector $\vec{i}$ and a single model vector $\vec{m}$. The RST transformation which will bring $\vec{i}$ into registration with $\vec{m}$ may be computed as follows.

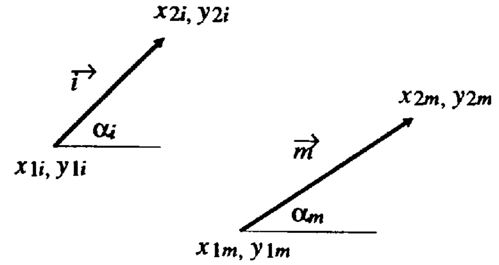


Figure 1. An RST transformation.

The rotation angle, α, is given by

$$\alpha = \alpha_m - \alpha_i \qquad (1)$$

where α_m is the orientation of the model vector, $\vec{m}$, and α_i is the orientation of the image vector, $\vec{i}$.The scale change, ρ, is given by

$$\rho = \frac{\lambda_m}{\lambda_i} \qquad (2)$$

where λ_m = length of model vector, $\vec{m}$

and λ_i = length of image vector, $\vec{i}$

The translations, Δx and Δy are then given by

$$\Delta x = \rho\, y_{1i} \sin\alpha - \rho\, x_{1i} \cos\alpha + x_{1m} \qquad (3)$$

$$\Delta y = -\rho\, x_{1i} \sin\alpha - \rho\, y_{1i} \cos\alpha + y_{1m}. \qquad (4)$$

The resulting transformation, in homogeneous coordinates, is then

$$[x_m, y_m, 1] = [x_i, y_i, 1] \begin{bmatrix} \rho\cos\alpha & \rho\sin\alpha & 0 \\ -\rho\sin\alpha & \rho\cos\alpha & 0 \\ \Delta x & \Delta y & 1 \end{bmatrix}. \qquad (5)$$

Consider an image which comprises a series of points, each of which represent a feature of the boundary of the object under scrutiny e.g. points of maximum curvature. These points are connected by straight line segments such that our image now comprises a series of directed line segments or edge vectors. Next, consider a similar image which will be our model. To bring the test image into registration with the model, we must find the RST transformation which optimally brings the object edge vectors into alignment with the corresponding model edge vectors.

We use a pose-clustering procedure to achieve this. Consider the image vector $\vec{a_i}$ in figure 2, we compute an RST transformation which will bring this into registration with the model vector $\vec{a_m}$. This transformation comprises a scale change, ρ, a rotation angle, α, and two translations, Δx and Δy. The computed transformation can be considered to contribute a count of one to the particular cell, or bin, in the 4-D space indexed by ρ, α, Δx and Δy. We can also compute an RST transformation which will bring this vector into alignment with all of the other model vectors $\vec{b_m}, \vec{c_m}$. These transformations also contribute counts to their respective bins in the 4-D space. Next image vector $\vec{b_i}$is considered and RST transformations are computed which register it with all of the model vectors. At the same time the counts associated with the corresponding bins of the 4-D space are incremented. This procedure is followed until all image vectors have been considered. The correct transformation will show up as the most significant cluster in this 4-D space. The process of assigning data quads, representing the transformation parameters, to cells in this 4-D space we term binning.

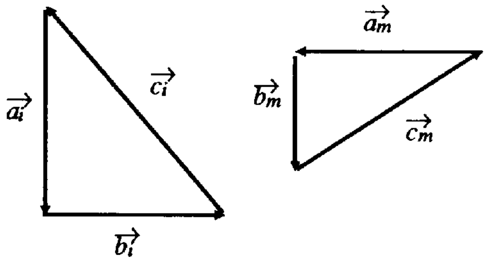


Figure 2. Both image and model data must be brought into registration.

We use a hierarchical approach and a series of overlapping bins in order to prevent the splitting of clusters. Several cluster centres may be followed up at any given level, and in the final level the best cluster is chosen from several candidates by computing "the average degree of alignment " between the image and model edge vectors. The cluster with the smallest r.m.s. distance between the vectors provides the final match which is also optimum in a global sense.

3. Determining 3-D Orientation.

Rather than just determining localisation and orientation in one plane, (i.e the plane of the image) we are able to determine the orientation of an object in 3-D space and its position within the field of view.

Orientation is given by three rotation angles, one about each axis of our coordinate system. The terms yaw, roll and pitch are commonly applied to airborne objects.The rotation about the z-axis (pitch) is a rotation in the plane of the image. The rotation about the y-axis (yaw) is a rotation in and out of the plane of the image and the rotation about the x-axis (roll) is a rotation about the longitudinal axis of the object itself.

For any fixed yaw and roll angles the problem reduces to 2-D. Envisage a database containing descriptions of a model object in terms of its edge segments for a series of yaw and roll angles corresponding to different 2-D projections, then, given an image of this object at some unknown orientation, it is possible to register this image with model elements in the database. Some combinations of yaw and roll give poor registration but for orientations close to the true orientation of the object the registration will be good. The problem now reduces to that of finding the optimum registration of the model edge vectors in the database with those of the object. A hypothesise and test type of approach is used in which simple measures of shape are used to direct a search of the database. Each selected model entry represents a hypothesised pose of the object in our image. A pose-clustering procedure is used to derive the parameters of the RST transformation which will bring the object into registration with its model description. The validity of the hypothesised pose is tested by applying the resulting transformation to the image data and comparing the r.m.s distance between the transformed image and model edge vectors. The database entry with the smallest r.m.s distance provides the final match. The yaw and roll angles are given by the yaw and roll angles of the selected entry in the database while the pitch angle is produced by the registration process.

4. Equipment Used.

We use a PC compatible fitted with an Imaging Technology Inc. FG-100-AT board. This provides a single, 12 bit image plane of 1024 * 1024 pixels. A single image occupies 512 * 512 pixels.

Unused portions of the image plane are used to store data in the form of integer values in the range of 12 bits, i.e. 0-4095. We store our database on an unused portion of the image plane.

5. Feature Extraction

Man-made objects such as aircraft, missiles etc. have well defined geometric outlines. The silhouette of such shapes can therefore be considered as a series of straight line segments separated by small regions where the curvature may change rapidly, perhaps discontinuously. One approach is therefore to locate points of high curvature and connect them by straight lines. This method has considerable support from psychological experiments[18] and as a means of shape description has become increasingly popular[9,12,19,20].

Our experiments indicate that, when noise is present, the performance of such algorithms is unreliable and therefore, locating high curvature points directly is ineffective. Instead, our approach is to detect the constituent sides of the object boundary. We search the boundary curve for linear segments and at the junctions between boundary segments we assign curvature discontinuities. The use of estimates of general trends in the curves to derive point features, rather than attempting to locate the features themselves, means that the detected positions of rapid curvature changes are much more stable in the presence of noise which is effectively filtered out.

We fit straight line segments of increasing length to the boundary curve using a least-squares minimisation. This is done for a series of points around the boundary which are equidistant in arc length. Then for each line length we determine points around the boundary where the errors of fit are locally minimal producing a sub-set of the original fitted lines. These points should correspond to regions of the boundary which approximate straight lines. The resulting lines are hierarchically assigned to an initially empty map of our object boundary, such that the longest lines are assigned first. When a line is assigned its domain on the boundary map is marked as bound; no other lines can subsequently be assigned within this domain. This process is continued until either there are no more lines to fit, or the map becomes full. We extrapolate assigned lines across any small gaps which might remain in the map and determine their intersection points. These points become the object features, corresponding to the points of maximum curvature.

6. The Model Data.

We generate a database of model edge arrays using wire frame models of a missile and a Hawk jet trainer. Each wire frame model is rotated through increments of five degrees in yaw & roll, using a standard transformation matrix[21]. At each orientation a description of the model in terms of line segments is produced. The resulting list of x,y coordinate pairs constitutes the model description at that orientation. We will refer to these model descriptions, which go to make up our database, as model arrays. The description of the object in the image we will term the image array. The model arrays are stored on a spare quadrant of the image plane.

7. Retrieving Model data.

When searching the database for a best match we wish to limit the number of model arrays we read and compare, to a small but suitable subset of the database. With this aim, we devised a method of pointers to individual model arrays based upon the area/perimeter^2 ratio, r_m. We compute the area/perimeter^2 ratio of the viewed object, r_i, and this is used to direct our search of the database. We scan those model arrays whose ratio, r_m, is within the range

$$r_i - t \leq r_m \leq r_i + t \qquad (6)$$

where t is a predetermined threshold.

The values $r_i \pm t$ represent the upper and lower bounds of the area/perimeter^2 ratio of model data we wish to compare. Having located candidate model arrays in this way we match each with the image array; this match is described next.

In practice, this limits searches to less than 20% of the database.

8. The Registration Process

The feature detection algorithm produces a list of x,y coordinates which correspond to end-points of linear segments around the boundary of the object. Each vertex connecting adjacent segments is assigned a type depending upon the angle between the segments. For angles in the range $[0, \pi]$, a type "convex" is assigned, whilst for angles in the range $[\pi, 2\pi]$, we assign a type "concave".

The area/perimeter^2 ratio of the description is computed and model database records whose area/perimeter^2 ratio lies within the specified range of this value are selected for comparison. Then, the object description is compared to each selected model record in turn through a pose-clustering procedure.

For each selected record only combinations of image and model segments with the same vertex types are included when forming the 4-D cluster space. Also, combinations of model and image edge vectors are rejected if the computed rotation angle does not fall within ±15 degrees of the difference in the principal axes of the object and model. This is a robust procedure to adopt since, even in the case of corrupt data, the principal axis should not be in error by more than this amount. This use of object features reduces the number of possible combinations of image and model arrays by about 90%. Further reductions in the number of hypothesised poses are achieved by using the difference in the coordinates of the centroids of the object and model to constrain the translations $\Delta x, \Delta y$. We also use the ratio of the square root of the projected area of object and model to constrain scale factors.

Initially the cluster space will consist of many different combinations of model and image edge vectors. It is not possible for any one image edge vector to correspond to more then one model edge vector and vice versa. The number of unique model-object edge vector pairs in our cluster space is computed and if below a predetermined threshold then the clustering process is terminated. This threshold is set at 40% of the number of model edge vectors.

The algorithm used to identify candidate clusters or transformations lies at the heart of the method and therefore needs to be carried out efficiently. An approach which is hierarchical in nature is described, and rather than clustering in 4-space directly and forming a 4-D histogram with its associated high storage requirements, a simple 2-D table structure is used.

The parameters of each computed candidate transformation are examined in turn. This information is entered into an initially empty table which comprises five columns ρ, α, Δx, Δy and a count. If a given combination of parameters is not present then that combination is entered and the count initialised to one. If already present then the count is simply incremented. In practice, a single record in the table will not represent a single transformation, but rather, is a bin containing a group of transformations whose parameters lie within a specified range of each other.

This procedure is performed in a hierarchical fashion, binning up the data initially on a coarse scale, locating the candidate clusters at that level which are to be further binned in subsequent levels. We set a maximum number of such iterations to be five and terminate the binning when this is exceeded or when the standard deviation within individual bins falls within acceptable bounds. In practice, the process converges after only two or three iterations, and cases which continue beyond this point are probably mismatches with our database. In this instance there will be very few possible model-object edge vector pairs in the final bin. To prevent clusters from splitting into different bins, we use a series of overlapping bins.

The data in the finally selected bin gives an estimate of the RST transformation which will register our object data with the database entry. We now wish to evaluate how good this registration is - poor registration indicates an incorrect match. The estimated transformation is evaluated by applying the transformation itself to the x,y coordinates which correspond to the data in the final bin. We then compute the r.m.s difference between these transformed object coordinates and those of the model; this is used to select the best match. The smallest r.m.s error, of those model arrays selected, produces the estimated orientation of our sensed object. Using object features means that less than one third of the selected database entries reach the stage of hypothesis verification. To prevent complete computation of the poorest matches, we set an initial threshold on the r.m.s. value, and once this value is exceeded the summing and squaring operations cease. When a complete pass through a data set is accomplished a new minimum r.m.s. value is obtained and this becomes the new threshold value. We find that as few as 5% of the selected database entries reach this final stage of the matching process.

When the model data for several different objects are combined to produce a single multi-object database, the system is also able to perform object recognition, as our results show. In this case, the best match also determines the objects identity.

9. Experimental Results

Wire frame models of a Hawk jet trainer and a missile were used to generate object data to test our system. The wire frame models were rotated into many orientations and object descriptions generated, (section 5). Each computed description was then matched with our database. This was performed for both clean images and noisy images. We added noise[22], in varying amounts, to the traced outline of each wire frame model and then used these noisy boundaries to generate line segment descriptions of our objects for matching to our database. The results in Table 1 are for a noise level which perturbed ≈60% of boundary pixels. Figure 3, shows a typical noisy object with ≈60% of the boundary pixels perturbed. We also tested our system with partially occluded objects. Partially occluded boundaries were generated by producing solid silhouettes from the wire frame models and then masking off selected regions. The boundaries of the

orientation						rec.	obj.	error in degrees of arc	
true			sensed						
yaw	roll	pitch	yaw	roll	pitch			θ	ϕ
3	12	11	5	5	10.4	yes	m	2.2	7.0
27	36	5	25	45	7.1	yes	m	2.7	8.5
61	37	123	60	30	121.4	yes	m	1.4	5.3
82	39	359	80	35	4.4	yes	m	2.1	9.1
79	81	315	75	80	317.8	yes	m	4.1	5.6
3	12	11	5	15	12.5	yes	h	2.8	3.2
27	36	5	25	35	4.0	yes	h	2.2	0.7
61	37	123	60	40	123.2	yes	h	1.0	3.1
82	39	359	80	30	5.4	yes	h	2.2	15.1
79	87	315	80	80	317.4	yes	h	1.1	3.1

Table 1. Results for noisy images of both the Hawk(h) and the missile(m) with ≈60% of the boundary pixels perturbed. (rec = recognised; obj = object).

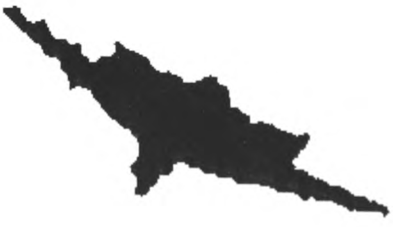


Figure 3. A noisy(≈60%) image of the Hawk jet trainer.

orientation						rec.	obj.	error in degrees of arc	
true			sensed						
yaw	roll	pitch	yaw	roll	pitch			θ	ϕ
45	11	39	50	10	35.8	yes	m	5.4	3.1
82	39	359	85	40	356	yes	m	3	4.5
45	11	39	50	20	42.9	yes	h	5.7	6.2
82	39	359	85	10	328	yes	h	4.5	4.7

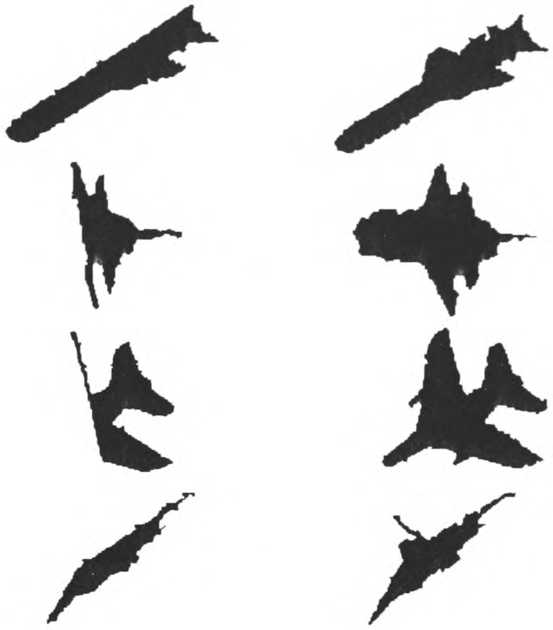


Table 2. These results are for noisy images with 40% of the boundary removed. Both the complete and partial object is shown alongside each result. The applied noise level resulted in ≈60% of boundary pixels being perturbed.(h=hawk; m=missile; rec=recognised)

resulting clipped shapes were then traced and our line fitting algorithm used to generate the object descriptions. Table 2, shows a selection of our results, in each case both the complete and the occluded silhouette is shown. When noise levels reached ≈60% and the amount of occlusion applied was 40%, the system began to break down, i.e. approximately 1 in 6 test objects were unrecognised.

Table 3. shows the average results for all test images.

Because some orientations are not unique, i.e. different combinations of yaw, roll and pitch produce identical orientations, some apparently disparate sensed orientations can in fact be quite accurate. Therefore, in order to quantify our results we define errors in terms of two angles θ, ϕ, which are the difference in degrees of arc in the major and minor axes of both sensed and true orientations respectively.

Proportion of occluded boundary.	Applied noise level			
	≈30%		≈60%	
	uncertainty in degrees of arc			
	θ	ϕ	θ	ϕ
0	4.4	4.5	4.8	4.5
10%	5.8	3.6	3.0	3.8
20%	6.4	3.7	6.5	5.9
30%	4.4	6.6	6.1	4.8

Table 3. Average results for all images

For objects with a high degree of symmetry, i.e. the missile, some different orientations may produce the same silhouette. As an example, our method will not distinguish between the orientations 0,0,0 and 0,90,0.. However, the human eye could not distinguish between these orientations without the aid of other cues. As a result, we only store and therefore detect roll angles for the missile in the range [0....85] degrees;

10. Discussion.

Noise and occlusion will distort an object's shape sufficiently to conceal, or even add, features. Local methods, which match parts of a boundary, are more likely to succeed under such conditions than global shape descriptors.

Using a least squares fit to derive a series of linear segments which represent the local geometry of the boundary of an object is effective in reducing the influence of noise and is a robust method of describing man-made shapes. The algorithm is also capable of describing small changes in the orientation of modelled objects.

The combination of a clustering algorithm to derive the parameters of the RST transformation which maps image data onto model data with a search of a database of 2-D model descriptions forms the basis of a system which is capable of determining object orientation to within an average uncertainty of 7 degrees of arc in both the major and minor axes, even when a high degree of noise is present. This method also produces accurate estimates of the object's position, and its scaling with respect to its model description.

Because the method matches individual image edge segments to model edge segments it is able to cope with occluded or partial data. We have found that the method consistently gives good results with up to 30% of the boundary missing and with noise affecting ≈60 % of the remaining edges. When 40% of the boundary is removed the method occasionally fails to recognise the object and therefore determine orientation. Nevertheless we have shown that the method is reasonably robust to both noise and to occluded objects.

Precomputing model data and storing it on an unused quadrant of the image plane is both effective and efficient, since the database is effectively held in RAM. Furthermore, the model descriptions are computed once only in a process that occurs prior to the primary functions of the system. The use of object features and heuristics which reduce the search area and the computational demands of subsequent matching further enhances the method.

We have analysed how errors in the detected positions of the object features affect the calculated RST transformation. Assuming that the model feature positions are known exactly, the errors in the transformation parameters will be due only to errors in the coordinates derived in the image frame.

Error analysis is beyond the scope of this paper. However, the practical implications are as follows. We wish to use the longest line lengths available to us in order that we minimise errors. Simply deleting short lines is inadvisable since, by doing so, we will be removing some features which are 'close' to one another from consideration by our clustering algorithm; these may be important in deriving orientation.

The procedure we have adopted is to form a series of virtual edges, that is edges which connect features that are not directly linked by real edges in the image. In this way we can include every detected feature, whilst choosing our virtual edges such that the edge lengths are maximised. In this way we minimise errors in the parameters of the resulting RST transformation, and thus improve the accuracy of the determined orientation.

11. Conclusions.

Redefining boundaries by directed edge vectors and matching them to similarly described models provides a sound basis for recognising objects and determining their orientation to within an acceptable degree of uncertainty. When a pose-clustering technique is combined with a search of a database of precomputed 2-D descriptions, the result is an effective and accurate algorithm.

References

1. **Dudani, S.A., Breeding, K.J .and McGhee, R.B.** 1977 *"Aircraft Identification by Moment Invariants"*, I.E.E.E. Trans. Computers, C-26, no.1, pp39-46.

2. **Reeves, A.P. , Prokop, R.J., Andrews, S.E. and Kuhl, F.P.** 1984 *"Three-dimensional Shape Analysis Using Moments and Fourier Descriptors"*, Proc. 7th Int. Conf. Pattern Recogniton, Montreal, Canada, July 30 - Aug. 2.

3. **Zahn, C.T. and Roskies R.Z.** 1972 *"Fourier Descriptors for Plane Closed Curves"*, I.E.E.E. Trans. Computers, C-21, pp 269-281.

4. **Granlund, G.H.** 1972 *"Fourier Preprocessing for Hand Print Character Recognition"* I.E.E.E. Trans. Computers, C-21, pp195-201.

5. **Richards, C.W. and Hemami, H.** 1974. *"Identification of Three-dimensional Objects using Fourier Descriptors of the Boundary Curve"*, I.E.E.E. Trans. Systems, Man, Cybern. SMC-4, pp371- 378.

6. **Wallace, T.P. and Wintz, P.A.** 1980 *"An Efficient Three-dimensional Aircraft Recognition Algorithm Using Normalised Fourier Descriptors"*, Comput. Graphics and Image Processing, vol 13, pp 96-126.

7. **Teh, C.H. and Chin R.T.** 1988 *"On Image Analysis By The Method Of Moments"* I.E.E.E. Trans. Pattern Analysis Mach. Intelligence, PAMI-10, no.4, pp496-513.

8. **Rice, J,R,** 1969 *"The Approximation of Functions,2 "*, Addison-Wesley,Reading.

9. **Koch, M.W. and Kashyap, L.** 1987 *"Using Polygons To Recognise And Locate Partially Occluded Objects"* I.E.E.E. Trans. Pattern Analysis Mach. Intelligence, PAMI-9, no 4, pp483- 494.

10. **Bolles, R.C. and Cain R.A.** 1982 *"Recognising And Locating Partially Visible Objects : The Local-Feature-Focus Method"* Int. Journal Robotics Res., vol 1, pt.3,pp57-82.

11. **Stockman, G., Kopstein, S., Benett, S.** 1982 *"Matching Images to Models for Registration and Object Detection via Clustering."* I.E.E.E. Trans Pattern Analysis Mach. Intelligence, PAMI-4, no.3, pp229-241.

12. **Wallace, T.P., Mitchell, O.R. and Fukunaga, K.** 1981 *"Three-dimensional Shape Analysis Using Local Shape Descriptors"*, I.E.E.E. Trans. Pattern Analysis Mach. Intelligence, PAMI-3, no.3, pp310-323.

13. **Ballard, D.H.** 1981 *"Generalising The Hough Transform To Detect Arbitrary Shapes"*, Pattern Recognition, vol.13, no.2, pp111-122.

14. **Dhome, M. and Kasvand,T.** 1987 *"Polyhedra Recognition by Hypothesis Accumulation"*, I.E.E.E. Trans. Pattern Analysis Mach. Intelligence, PAMI-9, no.3, pp429-438.

15. **Turney, J.L., Mudge, T.N. and Volz, R.A.** 1985 *"Recognising Partially Occluded Parts"*, I.E.E.E. Trans Pattern Analysis Mach. Intelligence, PAMI-7, no.4, pp410-421.

16. **Stockman, G.** 1987 *"Object Recognition And Localisation via Pose-clustering"*, Computer Vision, Graphics and Image Processing, vol. 40 pp 361-387.

17. **Stockman, G. and Esteva, J.C.** 1985 *"3-d Object Pose From Clustering With Multiple Views"*, Pattern Recognition Letters, vol.3, pp279-286.

18. **Attneave, F.** 1954 *"Some Informational Aspects Of Visual Perception"* Psychological Review 61, pp183-193.

19. **Shirai, Y.** 1978 *"Recognition Of Real-world Objects Using Edge Cues"*, in Computer Vision Systems (A Hanson, E.M. Riseman, Eds.), Academic Press, New York. pp 353-362.

20. **Chien, C.H. and Aggarwal, J.K.** 1989 *"Model Construction And Shape Recognition From Occluding Contours"*, I.E.E.E. Trans. Pattern Analysis Mach. Intelligence, PAMI-11, no.4, pp372-389.

21. **Foley,J.D. & Van Dam,A.** *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, 1983.

22. Theoretical Weapon Attitude Study, Report No.92.53/4, M.O.D(PE), R.A.E., Jan.1986.

Reconstructing Partially Occluded Object Boundaries.

D.P.Illing* & P.T.Fairney+,
* Dept. of Mathematics and Computing,
+ Dept. of Computer Studies,
The Polytechnic of Wales,
Pontypridd,
Mid Glamorgan. CF37 1DL.
G.B.

Key words.

Pose-clustering, reconstruction, RST transformation.

Abstract.

Straight line segments of varying lengths are used to redefine the projected boundary of man-made objects. The resulting directed edge segments are compared to data in a multi-object database using a pose-clustering technique. The best match both recognises the viewed object and determines its 3-D orientation. Missing portions of the projected boundary are reconstructed from the appropriate model edge segments in the matched database entry.

Address for correspondence.

Dr.P.T.Fairney
Dept. of Computer Studies,
The Polytechnic of Wales,
Pontypridd,
Mid Glamorgan. CF37 1DL.
G.B.

Introduction

Many different methodologies have been developed to describe the shape of regions and these can be classified according to several different criteria. A fairly fundamental distinction may be made between those techniques which allow some degree of shape recovery from the shape descriptors and those which do not. These are termed information preserving and information non-preserving, respectively (Pavlidis, 1980).

Quantities such as width, elongation, the area/perimeter^2 ratio and other simple measures of shape were amongst the first to be used. These methods are simple to implement but are limited since widely different shapes can produce identical descriptors. They do not allow recovery of the original object shape and are thus information non-preserving.

Two widely used information preserving methods are moment-based techniques and Fourier descriptors, (e.g. Dudani et al(1977), Reeves et al(1984) and Wallace et al(1980)). The latter allow reconstruction of the original shape to any degree of accuracy, depending upon the number of coefficients of the Fourier series that are retained as shape descriptors. However, it is not easy to relate higher order moments to shape and the method is particularly susceptible to noise. Also, in keeping with all transform techniques, these methods are unable to represent local information with any degree of accuracy, (Rice (1969)), and degrade rapidly in cases where part of an object is missing or occluded.

For these reasons there has been a move away from global towards local descriptions of shape. Object features which have been used in this way include corners (Koch and Kashyap, (1987)), holes (Bolles and Cain, (1972)), and lines (Stockman et al, (1982)). An example of the use of local features, to recognise 2-D silhouettes of aircraft, can be found in Wallace et al, (1981).

We present a method which describes object shape in terms of the local geometry of the boundary. Analytic techniques are used to derive structural descriptions of object boundaries, in terms of angles and straight line segments. The resulting piece-wise linear approximation is, by its very nature, information preserving and, since it is derived from an analysis of local features, is able to produce accurate object descriptions for both noisy data and cases of partial occlusion.

The method recognises and determines the orientation of 3-D objects from single 2-D orthogonal projections. It does this by comparing the 2-D object description to a database of similarly described 2-D projections at different orientations in a hypothesise and test type of approach. The resulting match is then used to reconstruct the complete silhouette projection, replacing any unmatched portions of the boundary with the appropriate model line segments, and adding any missing or occluded parts. This results in a reconstruction consisting of partly real and partly model data.

When comparing the image to a database entry we use a pose-clustering type of approach, (e.g. Stockman et al (1982), Stockman (1987), Dhome and Kasvand (1987)), accumulating evidence which either supports or weakens a hypothesis on the pose of an object in an image. Object orientation is achieved by solving for the RST (rotation,scaling,translation) transformation which maps observed features in an image onto features in a stored model description. This implies that the object is rigid and the object in the image can be represented by an instance of the model having undergone some RST transformation. Despite these limitations the approach is viable for many types of objects. Its chief advantages lie in the fact that the procedure is robust to poor feature detection i.e. 'missing' features and 'false' features. In this way a match can still be found even if some model features are missing from the image, and if spurious features are found. It is this fact which enables the method to cope with both noisy and partially occluded objects.

3-D Orientation from 2-D Pose-clustering

Consider an image (the sensed object) which comprises a series of points, each of which represent some feature of the object under study e.g. points of maximum curvature around the boundary. These points may be connected by straight line segments such that our image now comprises a series of directed line segments or edge vectors. Next, consider a similar image which will be our model. In order to bring the sensed object into registration with the model, we must determine the RST transformation which optimally brings the object edge vectors into alignment with the corresponding model edge vectors. A solution to this problem is provided by a pose-clustering procedure in 2-D, e.g. Stockman et al (1982).

Each of the image edge vectors can be registered with each of the model edge vectors by means of a RST transformation, i.e. a scale change, s, a rotation angle, θ, and two translations, Δx and Δy given by

$$[\, x_m' \;\; y_m' \;\; 1] = [\, x_i \;\; y_i \;\; 1] \begin{bmatrix} s\cos\theta & s\sin\theta & 0 \\ -s\sin\theta & s\cos\theta & 0 \\ \Delta x & \Delta y & 1 \end{bmatrix} \qquad (1)$$

where x_m', y_m' are the estimated coordinates of the end points of the transformed object vector and x_i, y_i are the positions of the end points of object vector.

We take all possible combinations of object and model edge vector pairs, searching for clusters of vector combinations whose values of s, θ, Δx and Δy are similar. The most significant cluster in this 4-D space will provide the best estimates of s, θ, Δx and Δy which optimally bring the object into global registration with the model; this forms the basis of the pose-clustering technique.

This approach has been extended such that rather than just determining localisation and orientation in one plane, (i.e the plane of the image) we are able to determine the orientation of an object in 3-D space and its position within the field of view from a single 2-D image.

Orientation is given by three rotation angles, one about each axis of our coordinate system, see figure 1. The origin of our system is at some notional point within the object in question, e.g. the centre of mass. The terms yaw, roll and pitch are commonly used to describe the orientation of airborne objects.

For any fixed yaw and roll angles the problem reduces to 2-D. Consider a database containing descriptions of an object in terms of its edge segments for a series of yaw and roll angles. Then, given an image of this object at some unknown orientation, it is possible to register its description with the elements of the database. For some combinations of yaw and roll the registration will be very poor, but for orientations close to the true orientation of the sensed object the registration will be good. The problem then reduces to that of finding the best registration of the model edge vectors in the database with those of the object. This follows a hypothesise and test type of approach in which we use simple measures of shape to select areas of the database for matching. Each selected model entry can be considered as representing a hypothesised pose of the object in our image. A pose-clustering procedure is used to derive the parameters of the RST transformation which will register the object with its model description. The validity of the hypothesised pose is then tested by applying the derived transformation to our image data and computing the r.m.s. distance between the transformed image and model edge vectors. The database entry with the smallest r.m.s. distance provides the final match.

Feature Extraction

The objects we consider have well defined geometric outlines e.g. aircraft, missiles etc. A boundary curve can therefore be considered as a series of straight line segments separated by small regions where the curvature varies very rapidly, even discontinuously. Since, in our experience, algorithms for computing curvature have not proved robust for noisy data, rather than searching for high curvature points directly, e.g. Shirai, (1978) and Chien and Aggarwal, (1989), the approach we have adopted is one of detecting the constituent sides of the object.

Our object boundaries are closed contours, i.e. each boundary is formed by a series of 8-connected pixels. We fit a series of straight line segnents of increasing length to the boundary using a least-squares minimisation. We do this for a series of points around the boundary which are equi-distant in arc length. Then, for each line length, the points around the boundary where the errors of fit are locally minimal are found, producing a sub-set of the original fitted lines. These lines are then assigned to an initially empty plan of our object boundary, in a top-down fashion, such that the longest lines are assigned first. Once a line is assigned to the boundary plan its extent on the plan is marked as full; no other lines are subsequently allowed to be assigned within full regions of the plan. This process of assigning lines to the boundary plan is continued until either there are no more lines to fit, or the plan is completely full. The assigned lines are then extrapolated across any small gaps which might remain in the plan and their intersection points are found. These points become the object features which correspond to the points of maximum curvature originally mentioned.

Computing the Model Data.

Wire frame models of a missile and a Hawk jet trainer are used to generate our database. Each wire frame model in turn is rotated through 5 degree increments in yaw & roll using a standard transformation matrix. From the silhouette of the rotated model we generate a description at each orientation in terms of straight line segments, producing a series of x,y coordinate pairs, each of which corresponds to the end-point of a line segment. This description is stored as part of our database on a spare quadrant of the image plane. When the data for several different objects is combined into a single multi-object database the method is also capable of object recognition. The selected match both recognises the object and determines its estimated orientation.

Registration Procedure.

The junctions or vertices between each straight line segment are assigned a type depending upon the angle between adjacent segments. For angles in the range $[0,\pi]$ we assign a type "convex" and for angles in the range $[\pi,2\pi]$ we assign a type "concave".

Combinations of edge vectors are only included in the cluster space if the vertex types at each end agree. This controls the combinatorics of the problem.

An efficient algorithm is required to identify candidate transformations. We describe an approach which is hierarchical in nature and rather than clustering in 4-space directly, producing a 4-D histogram with its associated high storage requirements, we use a simple 2-D table structure..

Initially the cluster space will consist of many different combinations of model and image edge vectors. Since it is not possible for any one image edge vector to correspond to more then one model edge vector, and vice versa, the number of unique model-image edge vector pairs in our cluster space is found; if this is too few then the clustering terminates.

Each object-model edge vector combination is examined in turn and the parameters of the RST transformation are computed. This information is entered into an initially empty table which comprises five columns, s, θ, Δx, Δy and a count. If a given combination of parameters is not present then that combination is entered and the count initialised to one. If already present then the only step required is to increment the appropriate count. In practice, a record in the table will not represent a single transformation, but rather is a bin containing a group of transformations whose parameters lie within a specified range of each other.

This process is hierarchical. The data are initially binned on a coarse scale, locating the candidate clusters at that level which require further binning in lower levels. In practice we find that the process converges after two or three iterations, and that cases which continue beyond this point are more than likely mismatches with our database. In such instances there will be very few possible model-object vector pairs in the final bin. Overlapping bins are used to avoid the splitting of clusters.

The data in the selected bin provide an estimate of the RST transformation which will register our object data with the database entry. The validity of the estimated transformation is evaluated by applying the transformation itself to the x,y coordinates corresponding to the object data in the selected bin. The r.m.s difference between the transformed coordinates and those of the model is computed and this forms the basis for the final selection of the best match. After registering the object data with each model database entry selected for comparison, the smallest r.m.s error gives us the determined orientation of our sensed object. The pitch angle falls out of the registration procedure whilst the yaw and roll angles are given by the yaw and roll angles of the best match entry in the database.

Reconstructing the Object Boundary

The matching algorithm produces an estimate of object orientation and the mean and standard deviation of the parameters of the RST transformation which maps detected image features onto the model description. Also produced is a list of correspondences between the matching object features and model features.

We use this list to determine which model features have not been matched with features of the object. These unmatched features are then split into a series of groups, each group lies between two matched model features. For each group the following procedure is then followed. We wish to transform the unmatched model features into the frame of the image such that a complete description of the object exists in the image frame. In order to do so we apply the inverse of the derived transformation, (equation 2), to the model features

$$[\, x_i' \;\; y_i' \;\; 1] \;=\; [\, x_m \;\; y_m \;\; 1] \begin{bmatrix} \dfrac{\cos\theta}{s} & \dfrac{\sin\theta}{s} & 0 \\ \dfrac{-\sin\theta}{s} & \dfrac{\cos\theta}{s} & 0 \\ \dfrac{(\Delta y \sin\theta - \Delta x \cos\theta)}{s} & \dfrac{(-\Delta x \sin\theta - \Delta y \cos\theta)}{s} & 1 \end{bmatrix} \qquad (2)$$

where x_i', y_i' are the estimated coordinates of the end points of the transformed model vector and x_m, y_m are the coordinates of the end points of the model vector.

We adopt an iterative technique, minimising the distance between the end points of each group of transformed model features and the corresponding matched object features where the actual parameters of the transformation used may take on any values in the range of ± three standard deviations away from the mean. Thus we use local estimates of the global transformation in order to achieve the best possible reconstruction.

Where object features have been matched the original boundary is retained, being replaced by appropriately transformed model data where no match is achieved due to corrupt or missing data. The result is a reconstruction consisting partly of real and partly of model data.

Experimental Results

We used the wire frame models of a Hawk trainer and a missile to generate object data for our system. The wire frame models were rotated into different orientations and descriptions in terms of line segments were generated.

The partially occluded boundaries were generated by producing solid silhouettes from the wire frame models and then masking off selected regions. The boundaries of the resulting clipped shapes were then traced. In each case we added noise, (private communication), to the traced outline of the wire frame models and used our line fitting algorithm to generate the object descriptions. These descriptions were then matched with our database.

Examples of our results are shown in figures 2 through 5. In each case the left hand image shows the noisy occluded object, in these examples with ≈60% of boundary pixels perturbed. The central image shows the reconstructed object and, to enable subjective comparison, the right hand image shows the original object.

Our experiments show that Fourier descriptors are unreliable when applied to partially occluded boundaries. However, when all of the boundary is present they produce acceptable results. In an attempt to quantitatively evaluate our results we used Fourier descriptors to determine the orientation of our objects both prior to reconstruction and on the reconstructed images. These are illustrated in Table 1.

At yaw angles approaching 90 degrees, different combinations of roll and pitch produce identical orientations, e.g. 90, 60, 30 also maps to 90, 30, 0. If we simply find the difference between the sensed and the estimated values of yaw, roll and pitch, what at first glance, appear to be inaccurate results may in fact be reasonably precise. Therefore, to enable a more accurate estimate of orientation error at high yaw angles, we define uncertainty in terms of two angles α and β which are the difference in degrees of arc in the major and minor axes of the object in both the true and sensed orientations respectively.

Discussion and Conclusions

When noise and/or occlusion affects an object's boundary, the resulting silhouette may be distorted sufficiently to hide, or even add features thus producing changes in overall shape. Local methods which match using portions of a boundary, are more likely to succeed under such conditions than global descriptions of shape.

The combination of a clustering algorithm to derive the parameters of the RST transformation which maps object data onto model data, with a search of a database of 2-D model descriptions, forms the basis of a system which is capable of accurately determining the orientation of objects and of producing accurate estimates of position and scaling with respect to model descriptions. Precomputing model data and storing it on an unused quadrant of the image plane is both effective and efficient, since the whole database is effectively held in RAM. When the model data for several different objects are combined the system is also able to perform the task of object recognition.

Using a least squares fit to derive a series of straight line segments representing the local geometry of the boundary of an object is effective in reducing the influence of noise. Since shape can thus be consistently represented with very little change in contour as imaging conditions change, we have obtained a robust method of describing man-made shapes, and one which is capable of describing small changes in the orientation of modelled objects. Furthermore the method is inherently information preserving; by maintaining a list of all of the unmatched edges in the selected model data we can easily restore these, leaving the matched edges unchanged. The result is an accurate reconstruction of the partially occluded object which comprises both real and model data.

The method can determine the orientation of objects with ≈60% of boundary pixels perturbed by added noise and with up to 30% of the boundary removed to within an average uncertainty of 7 degrees of arc in both of the objects major and minor axes. This enables accurate reconstruction of noisy and occluded objects.

When more than 30% of the boundary was removed our line matching algorithm began to break down. At 40% occlusion and with ≈60% of the boundary pixels perturbed with added noise some 15-20% of our sensed objects were unrecognised and therefore no estimate of orientation was produced to enable reconstruction.

The poor performance of Fourier descriptors on occluded objects can be seen in Table 1. The precision of the line fitting and the accuracy of the edge segment matching algorithms enables accurate reconstruction of the object. Fourier descriptors are used on the reconstructed objects in an attempt to quantify the results of our reconstruction. The improvement in the errors α, β is at once obvious.

Redefining boundaries by directed edge vectors and matching them to similarly described models provides a sound basis for recognising objects and determining their orientation to within an acceptable degree of uncertainty. When combined with a pose-clustering technique and matching to a database of precomputed 2-D descriptions, the result is an effective and accurate algorithm. The local nature of the object descriptions used, coupled with their information preserving properties means that a good match may still be achieved on degraded data and occluded portions of the boundary can easily be reconstructed.

References

Pavlidis,T. (1980) Algorithms for Shape Analysis of Contours and Waveforms. *IEEE Trans. Pattern Analysis Mach. Intelligence,* PAMI-2, 4, 301-312.

Dudani, S.A., Breeding, K.J. and McGhee, R.B. (1977) Aircraft Identification Moment Invariants. *I.E.E.E. Trans. Computers,* C-26, 1, 39-46.

Reeves, A.P. , Prokop, R.J., Andrews, S.E. and Kuhl, F.P. (1984) Three-dimensional Shape Analysis Using Moments and Fourier Descriptors. *Proc. 7th Int. Conf. Pattern Recogniton,* Montreal, Canada, July 30 - Aug. 2.

Wallace, T.P. and Wintz, P.A. (1980) An Efficient Three-dimensional Aircraft Recognition Algorithm Using Normalised Fourier Descriptors. *Computer Graphics and Image Processing,* 13, 96-126.

Rice, J.R. (1969) *The Approximation of Functions,2.* Addison-Wesley,Reading.

Koch, M.W. and Kashyap, L. (1987) Using Polygons To Recognise And Locate Partially Occluded Objects. *I.E.E.E. Trans. Pattern Analysis Mach. Intelligence,* PAMI-9, 4, 483- 494.

Bolles, R.C. and Cain R.A. (1982) Recognising And Locating Partially Visible Objects : The Local-feature-focus Method. *Int. Journal Robotics Res.,* 1, 3, 57-82.

Stockman, G., Kopstein, S., Benett, S. (1982) Matching Images to Models for Registration and Object Detection via Clustering. *I.E.E.E. Trans Pattern Analysis Mach Intelligence,* PAMI-4, 3, 229-241.

Wallace,T.P., Mitchell, O.R. and Fukunaga,K. (1981) Three- dimensional Shape Analysis Using Local Shape Descriptors. *I.E.E.E. Trans. Pattern Analysis Mach. Intelligence,* PAMI-3, 3, 310-323.

Dhome, M. and Kasvand,T. (1987) Polyhedra Recognition By Hypothesis Accumulation. *I.E.E.E. Trans. Pattern Analysis Mach. Intelligence,* PAMI-9, 3, 429-438.

Stockman, G. (1987) Object Recognition And Localisation Via Pose-clustering. *Computer Vision, Graphics and Image Processing,* 40, 361-387.

Shirai, Y. (1978) Recognition Of Real-world Objects Using Edge Cues. in *Computer Vision Systems* (A.Hanson, E.M.Riseman, Eds.), Academic Press, New York. 353-362.

Chien, C.H. and Aggarwal, J.K. (1989) Model Construction and Shape Recognition From Occluding Contours. *I.E.E.E. Trans. Pattern Analysis Mach. Intelligence,* PAMI-11, 4, 372-389.

Private Communication (1986) *Theoretical Weapon Attitude Study.* M.O.D.(PE), R.A.E., Report No.92.53/4.

object.	true orientation			% occlusion	sensed orientation of occluded object			uncertainty in degrees of arc		sensed orientation of reconstructed object			uncertainty in degrees of arc	
	yaw	roll	pitch		yaw	roll	pitch	α	β	yaw	roll	pitch	α	β
Hawk	45	11	39	30	70	15	30	25.4	15.4	55	13	55	10.3	8.5
Hawk	42	44	300	35	40	80	305	4.3	32.9	45	45	297	3.7	4.8
missile	82	39	359	40	60	25	97	32.1	112.9	85	15	327	4.6	9.0
missile	33	3	315	30	50	5	318	17.1	1.1	35	5	316	2.2	1.6

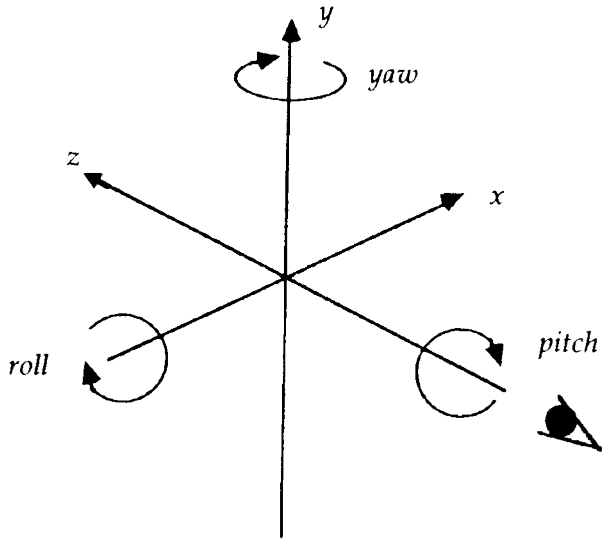
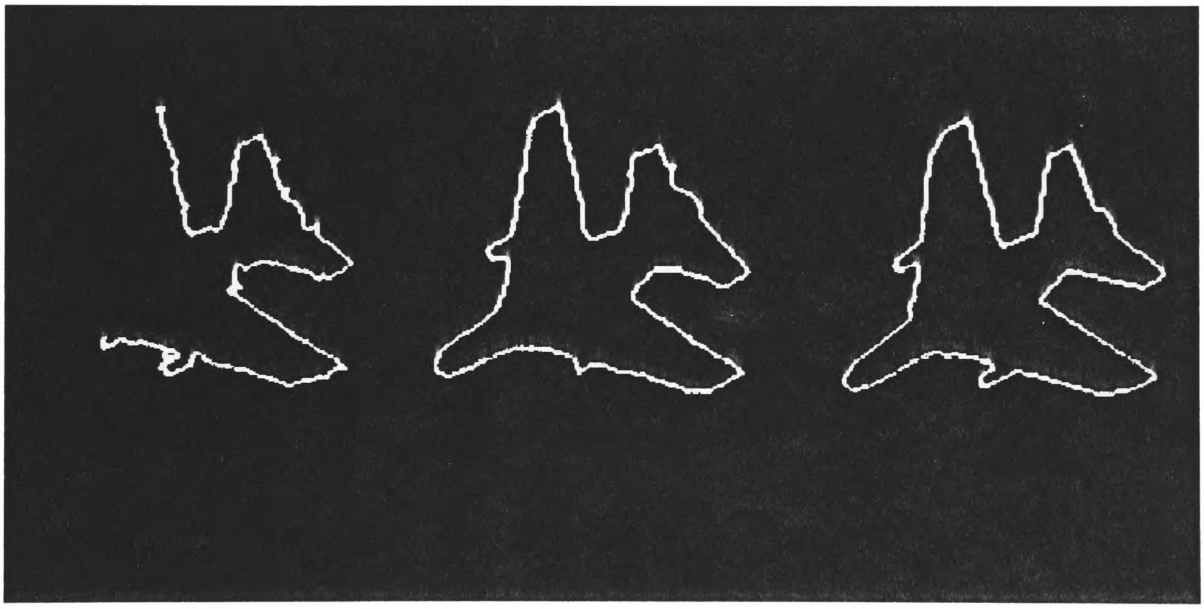
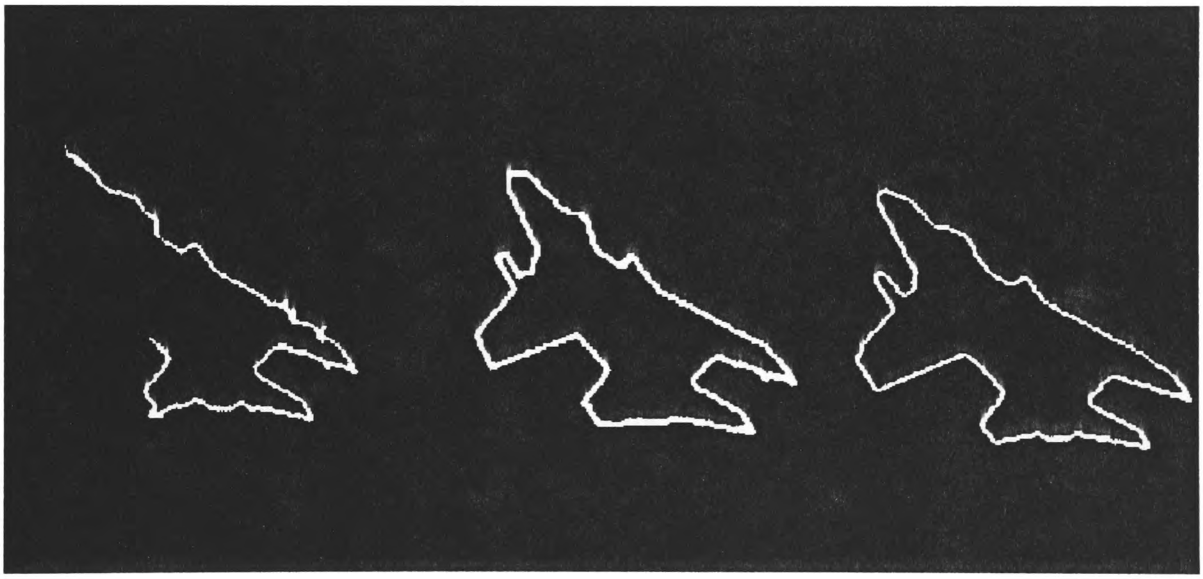
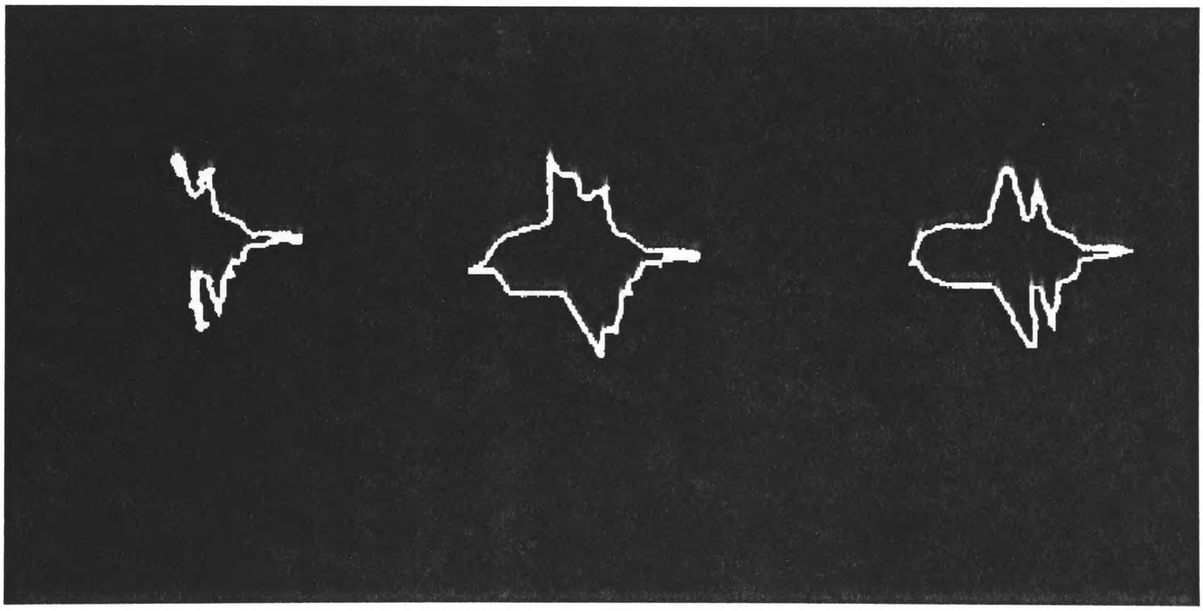


Figure 1.







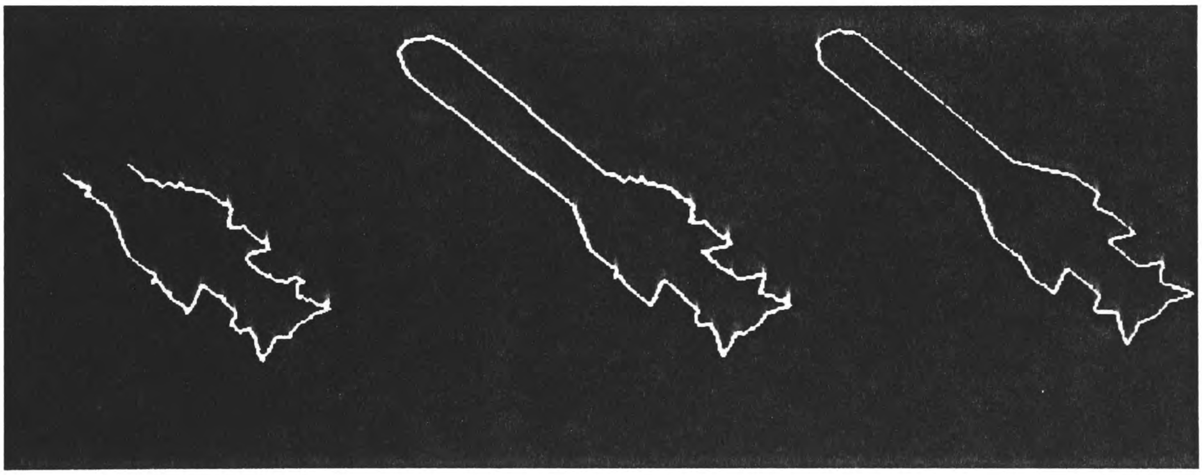


Table 1. Results, using Fourier descriptors, for the images shown in figures 2 through 5.

Figure 1. Our coordinate system showing yaw, roll and pitch.

Figure 2. Occluded, reconstructed and original images of the Hawk at 45, 11, 39.

Figure 3. Occluded, reconstructed and original images of the Hawk at 42, 44, 300.

Figure 4. Occluded, reconstructed and original images of the missile at 82, 39, 359.

Figure 5. Occluded, reconstructed and original images of the missile at 33, 3, 315.