RECOGNITION OF PARTIALLY OCCLUDED PARTS

by

Jerry Lee Turney

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer, Information and Control Engineering)
in The University of Michigan
1986

Doctoral Committee:

Associate Professor Trevor N. Mudge, Chairman
Professor Daniel E. Atkins
Associate Professor Ramesh C. Jain
Professor Ronald J. Lomax
Professor Richard A. Volz

For Margo, the girls, and the new baby.

ACKNOWLEDGMENT

I wish to thank my parents for their encouragement, my advisior for his help and advice, and the guys in Rm. 1504 for their friendship.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

CHAPTER I

INTRODUCTION

1.1. Problem Statement

In many practical applications of automated vision the basic vision task is that of recognizing a *known* industrial object in a digitized image. When the object is known, recognition is generally more successful if it uses a model of the object as a guide, an approach called model-based recognition. While it is true that all vision algorithms incorporate some type of model of the world, we use the term "model-based" in its strictest sense, where the model describes an exact object in as much detail as necessary for recognition.

Model-based recognition is not a well defined problem. Analytically, it is an attempt to find the inverse mapping from a 2-dimensional projection of an object as it appears in an image to the 3-dimensional object itself. In general, this is an underdetermined problem [Pal81]. Under certain constraints the problem can be reduced to one that will yield a simple solution. For example, if it assumed that objects are only presented in one of a finite number of states and if the objects are unoccluded then it is possible to extract a set of features of the object, e.g., area, length of perimeter, and moments of the profile of the object, that can be used together with standard statistical pattern recognition methods to recognize the objects with a high degree of success (see [Kan74, GlA79]). Without these constraints, however, the problem is much more difficult.

Partially Occluded Parts. When overlapping objects are present in the image the problem becomes more complex. It is difficult to untangle correspondences between projections and the objects that cause them because complete 2-dimensional projections no longer occur. In

addition, features such as area, length of perimeter and moments of the profile of the part can no longer be used in recognition. This problem of overlapping parts is sometimes named for a paradigm, the *bin of parts problem,* which involves recognizing parts piled in a bin, a common way in which parts are presented for batch assembly. The bin of parts problem has been described as "the most difficult problem in automatic assembly." A solution is said to be worth "tens of millions of dollars a year in the U.S."[Mat76]. The bin of parts problem is common to tasks such as part sorting, part retrieval, and part assembly, and, as yet, there is no satisfactory solution to this problem.

Partial occlusion is also found in images where objects are obscured by dirt, where objects are defective, or where objects are partially outside an image. Images of this type can be handled in the same way as parts occluded from overlap, although they are less difficult to deal with because there is no problem of segmentation, i.e., deciding which area of the image corresponds to which part.

In this discussion objects will be restricted to industrial parts, and the general recognition problem will be called the *partially occluded parts* problem or the POP problem for short. Overlapping, obscured, defective, and incomplete views of parts are instances of the generic POP problem.

This thesis presents a model-based method for solving a subproblem of the general POP problem in which the set of parts that are allowed to appear in an image are *2-dimensional.* For our purposes, a part is 2-dimensional if two of the dimensions of the part are much larger than the third (Fig. 1.1 shows an example). A solution to the 2-dimensional POP problem is nearly as valuable to industry as a solution to the general POP problem; applications involving stamped, cast, and forged flat metal parts are found everywhere in industry.

Most industrial systems do not allow parts to overlap or be occluded. In fact, most systems are based on incomplete models of the parts that they are to recognized. In addition to presenting a method for solving the 2-dimensional POP problem, this thesis will present extensions to deal with the following two problems.

Fig. 1.1. Overlapping parts.

Reflective Parts. One example, common to industry, where a model of a part is often incomplete is when the part is reflective—most smooth metallic parts are reflective. Reflective parts have surface irradiance that varies with position, causing the surfaces of the part to appear differently at different positions. This is not modeled. The shape of the boundary contours of the part does not change, but areas of the part undergo *contrast reversals*, i.e., some sections of the part appear lighter than the background (positive contrast) while other sections appear darker than the background (negative contrast). Contrast reversals confuse many vision systems, and, therefore, many systems eliminate the problem by back-lighting the part to give a uniform contrast. This thesis proposes a system which works with parts that are of either contrast with respect to the background.

Scale Changes. Another common way in which a model may be incomplete is by not representing the part at different scales. It is possible that the image of the part has a different

scale than the model upon which recognition is based. With the capability to compensate for scaling, a system is not restricted to a fixed viewing distance.

1.2. Model-based Recognition

Most model-based vision systems consist of two stages: an off-line precomputation, or *training* stage that extracts information about the parts necessary for their recognition; and a run-time, or *recognition* stage that recognizes parts in an image. The information obtained in training makes up the models of the parts. It is desirable to maintain as rich a model of each part as possible. The more information extracted during training, the more powerful the run-time recognition is likely to be. On the other hand, cost and performance considerations require that vision systems use incomplete models of parts. Unfortunately, this constraint, coupled with an injudicious selection of the type of model information, has in the past resulted in many models which consist of small *ad hoc* sets of features that do not even preserve shape information—one cannot reconstruct the shape of parts from these features. For example, the SRI Module uses part area, perimeter, number of holes and other features that characterize shape but that even taken together do not contain sufficient information for the part's reconstruction [GIA79]. These features, in addition, are not preserved under partial occlusion. More powerful recognition algorithms require richer models.

This thesis proposes a model-based system that assumes that the objects to be recognized are a set of rigid parts. More information is extracted *a priori* from the parts than is usually done. In particular, the relative importance of certain aspects of their shape is extracted by an off-line training step and embodied in the models of the parts. Not only does a part's model preserve a description of the shape of the part, but it also contains a measure of the importance of each component of the model in fixing the identity of the part.

1.3. Research Overview

This thesis will examine a vision system that provides solutions to many problems in industrial vision that current systems cannot handle. The system can recognize 2-dimensional parts under the following conditions:

- Parts located at any spatially position, or under any rotation about the viewing axis.
- Parts that touch, overlap, lie partially outside the image.
- Reflective parts and parts viewed under poor lighting conditions.
- Parts with any scale within a wide range.

To realize these capabilities the vision system incorporates two concepts: a *segmented boundary* representation in which the boundary of the part is set of overlapping segments; and a *saliency-based* recognition strategy, where the saliency measures the importance of segments in fixing the identity of the part.

1.3.1. Segmented Boundary Representation

In the segmented boundary representation, boundaries of parts are partitioned into equal length overlapping segments with a segment centered on each point of the the boundary. The overlap of the boundary segments makes the recognition system more dependable because there is enough redundancy in the representation so that, except in extreme cases, a few complete segments are visible even when a large percentage of a part is occluded. Two representations are used for the segments of the model. The first is a function, $\theta(s)$, where s is the arclength of each point along the boundary from some origin point on the boundary, and θ is the corresponding tangent angle of the point. We term this function the θ–s representation, and it is used for matching segments. The second is a cartesian space representation that is used to combine results of segment matches. Figure 1.2 shows an example of the cartesian and θ–s representation of the boundary of a part taken from a car's door lock. We refer the part as door lock part no. 1 in the text. The cartesian space representation has the advantage that it maintains the spatial relation-

Fig. 1.2. Cartesian and θ–s representation of door lock part no. 1.

ships between boundary segments. The θ–s space simplifies comparisons between model and image segments for variations in orientation, for scale changes and for contrast reversals.

1.3.2. Saliency-based Recognition

During training the proposed vision system extracts more information than has been typical in past model-based recognition. In particular, it automatically identifies segments of a part that have the highest probability of fixing the pose of the part and of distinguishing the part from other parts that may be present in the image; these segments will be called *salient features*. Salient feature can locate a part with great certainty even when little of the part is visible.

1.4. Thesis Organization

• Chapter II surveys the literature relevant to the problem of recognizing partially occluded parts from single views. The chapter includes descriptions of both 2- and 3-dimensional part recognition algorithms based on a single view. The chapter does not include algorithms based on range maps or multiple views of an part.

• Chapter III examines in detail several alternative algorithms proposed by other authors, and describes results obtained from these algorithms.

• Chapter IV describes the proposed method. The presentation is divided into two sections: one section describes the segmented boundary representation, and the second section describes saliency-based recognition.

• Chapter V describes of some of our past work that led to the development of the proposed method.

• Chapter VI examines experimental results obtained from saliency-based recognition and compares these results with those obtained from the methods examined in Chapter III.

• Chapter VII presents the concluding remarks.

CHAPTER II

LITERATURE SURVEY

2.1. Introduction

In this chapter we will present a survey of POP recognition approaches. We classify these approaches into one of three basic categories, *correlational*, *structural*, or *relational*.

More explicitly, we define a correlational approach as one in which features of a part are correlated to features of an image. Features can be things or attributes of a part such as color, regions, surfaces, or contours. In a correlational approach a part is typically moved around in an image and features of the part are compared with the local features that they overlay. Each matching pair of features is assigned a value that measures the degree of *local correspondence* between features. These values are summed to obtain a *global correspondence* of the part to the image at each location. After all comparisons are done, the location with the greatest global correspondence is assumed to be the location of the part.

We define a structural approach as one in which the structure of the part is iteratively built in the image. A part is represented as a set of substructures called primitives. A set of similar substructures are located in the image and are fit together to build an instance of the part in the image using a set of *production rules* as a blueprint. If the fraction of the part that can be built exceeds a predefined threshold, the part is assumed to be located.

Finally, we define a relational approach as one in which features and relations between features are abstracted from a part and compared to similar features and relations abstracted from an image. The location in the image with the greatest consistent set of features and

- **Correlational**
  - Hough Transform
  - Axial
  - Region
  - Slope Angle/Curvature
- **Structural**
  - Syntactic
  - Synthesis
- **Relational**
  - Relational Structures
  - Invariants
  - Relaxation

Table 2.1 Recognition strategies.

relations, i.e, consistent with those of the part, is selected as the location of the part.

We further divide these three approaches into classes (see Table 2.1). For completeness a list of common terms is given below.

Common terms. The 3-dimensional location and orientation of a part will be referred to as its *pose*. This gives us a terse way of expressing a frequently used concept. A 2-dimensional pose of a part will often be represented by the triple (θ, u, v) where θ is the relative orientation of the part about its centroid and u and v are the cartesian location of the centroid of the part.

A digitized view of a set of parts in their various poses in a scene is called an *image*. An image is a sampled array of picture elements called *pixels*. Pixels represent the light intensity of the image and typically take on values that are quantized into one of 256 *grey level* values.

For an image to be of interest, pixel values in the image must change in grey level value from location to location; these changes in pixel values correspond to intensity changes in the original scene. An *edge point* is characterized by having rapid changes in the pixel values in its neighborhood.

Edge points are often linked into pixel-wide *edge contours* to allow the edge points to be ordered and thus more easily compared to other sets of edge points. A connected fragment of an edge contour is called a *segment* (a fixed length segment is the basic primitive in our algorithm). By convention the edge points in a contour are ordered in such a way that when a contour is traced the higher intensity side (in the original image) of the contour is kept on the left. Furthermore, edge points are often located by an arclength along the contour measured from an arbitrary point of origin on the contour.

In addition to arclength, an edge point on a contour may also be assigned a value of *slope angle* and a value of *curvature*. The slope angle is the angle of a tangent to the contour at each edge point, and the curvature is the derivative of the slope angle with respect to the arclength at an edge point. Only one value of slope angle and curvature are associated with each arclength, therefore both slope angle and curvature can be viewed as 1-dimensional functions of arclength. We will represent the slope angle function by $\theta(s)$ and the curvature function by $\alpha(s)$.

Strong local changes in curvature are called "critical" points. They have been shown by Attneave to be important to humans in recognizing objects, and, therefore, perhaps are equally important to machines [Att54]. Critical points play a role similar to the "dots" in a children's "connect-the-dots" picture: they act as markers on the contour that define its essential structure. In addition to this role, their location on a contour tends to be insensitive to noise and insensitive to tilt and scaling. Therefore, they are useful in locating parts that are tilted and scaled.

Finally, the contours of a part collectively form the part's *boundary*. A boundary includes both internal and external contours.

In the following, the approaches in each of the classes in Table 2.1 are presented in the order in which the relevant publications occurred (with the exception that closely related work by

the same author(s) appears together). Furthermore, the review of each class is preceded by a tutorial that summarizes the basic approach of the class. We begin with a discussion of correlation approaches.

2.2. Correlational

Correlational approaches belong to four distinct classes: Hough transforms, axial correlation, region correlation, and slope angle/curvature correlation. To begin with, we start with a tutorial on Hough transforms.

2.2.1. Hough Transform

The Hough transform is a very general histogram approach for extracting information; however, we will only describe its use in part recognition. For this purpose consider a boundary of a part, $B(\mathbf{p})$. The parameter $\mathbf{p}$ of the function is a vector corresponding to the pose (and possibly the scale) of the part in the image. Assume that a set of features, $f_i^{B(\mathbf{p})}$ are found in the boundary. Appropriate choices for features are edge points, straight line segments, circular arc segments, critical points, or corners of the boundary. Furthermore, assume that a similar set of features, f_j^I, are found in the image. In the Hough transform the features of the boundary are compared to features of the image. The set of parameter vectors, $\mathbf{p}$, that would permit the $f_i^{B(\mathbf{p})}$ to match the f_j^I are determined. This set of $\mathbf{p}$ can be considered as a set of *local hypotheses* for the part's pose because they are obtained by matching local part and image features. These local hypotheses are combined to form a *global hypothesis* for the part's pose. More precisely, the global hypothesis is selected as the most frequently occurring local hypothesis. To determine the most frequent hypothesis the set of parameter vectors, $\mathbf{p}$, are tabulated in a histogram. After all features of the part and image are matched, the most frequently occurring parameter vector, which we will denote as $\mathbf{p}^*$, is selected as the pose of an instance of the part in the image. In the

more general case where the histogram has m large modes at locations $\mathbf{p}_i^*$ $i = 1 \cdots m$, the $\mathbf{p}_i^*$ are assumed to correspond to m different instances of the part.

As a concrete example, consider a boundary of a triangle (see Fig. 2.1) that is parameterized by the location of its centroid, $\mathbf{p} = (u, v)$. Let the edge points of the boundary be the features of the triangle. If the image consists of the five edge points shown in Fig. 2.1b, then the set of $\mathbf{p}$ that would be determined by the Hough transform are the lines shown in Fig. 2.2a. In other words, if the centroid of the triangle's boundary were placed at any of these (u, v) points, at least one edge point of the boundary would match an edge point of the image. The most frequently occurring parameter vector, $\mathbf{p}^*$, is the one at the center of the circle shown in Fig. 2.2a. With this choice for $\mathbf{p}^*$ the triangle's boundary would be placed as shown in 2.2b.

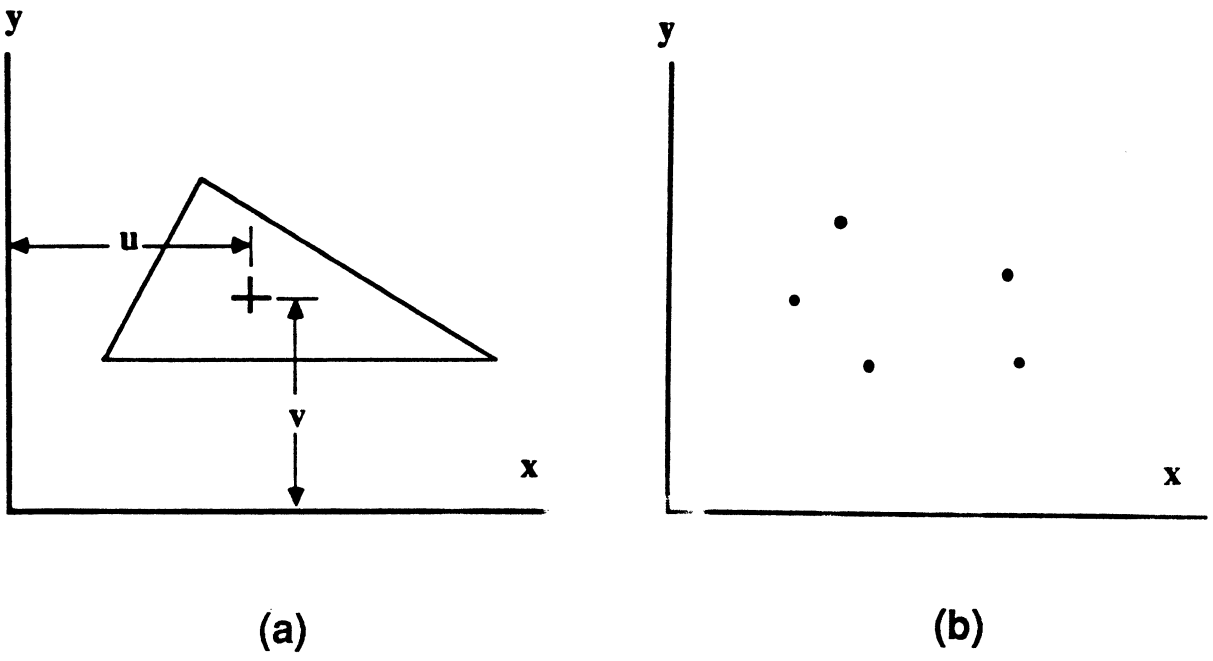

Fig. 2.1. Parameterization of a triangle boundary and image of edge points.

In the more general 2-dimensional part recognition problem, a boundary is usually parameterized by its orientation, θ, its scale, S, and its centroid location, (u, v). In this case, the parameter vector **p** is the quadruple (S, θ, u, v). As before, features of the part are matched to those in the image but now the sets of parameter vectors **p** are tabulated in a 4-dimensional $S-\theta-u-v$ histogram, rather than the 2-dimensional $u-v$ histogram.

The strength of the Hough transform is that incomplete information, i.e., local hypotheses generated from local feature matching, can be combined to form a global hypothesis for the pose of a part in an image. The shortcoming of the Hough transform is the method by which the local hypotheses are combined. Selecting the most frequent local hypothesis, as is done in the Hough transform, is not an effective method for the following reasons.

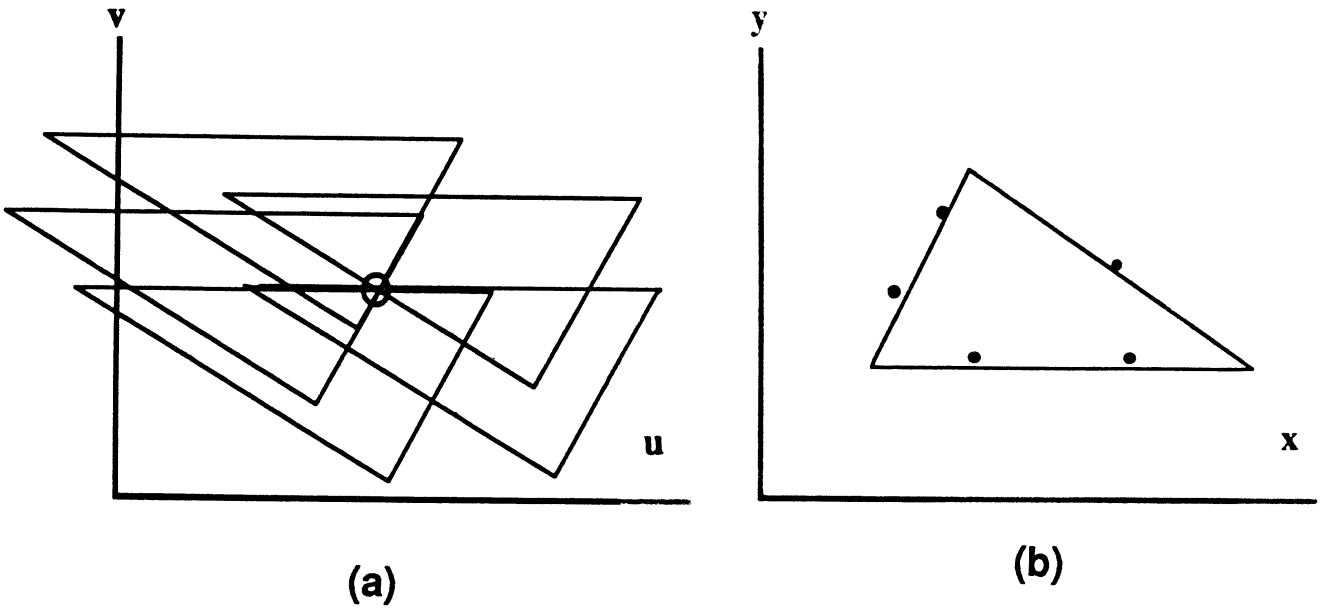

Fig. 2.2. Parameter histogram and result.

There is often a relationship between local hypotheses which is totally ignored in the Hough approach. For example, consider the POP image shown in Fig. 2.3 in which an industrial part (see Fig. 2.3a) appears in a POP image (see Fig. 2.3b). In this example, we will assume that fixed length segments of the boundary are its the features. Two segments on the part's boundary, labeled A and B, are shown in the image. Two corresponding segments, labeled A' and B' are shown in the image boundary. The spatial relationship of the two segments A' and B' is adequate to eliminate all but one hypothesis for the pose of the part—the remaining hypothesis associates A with A' and B with B'. (This spatial relationship is explicitly used in the approach proposed by this thesis.)

In the Hough transform, however, the two local hypothesis generated by matching A to A' and B to B' are simply tabulated in a histogram. For the approach to select the correct

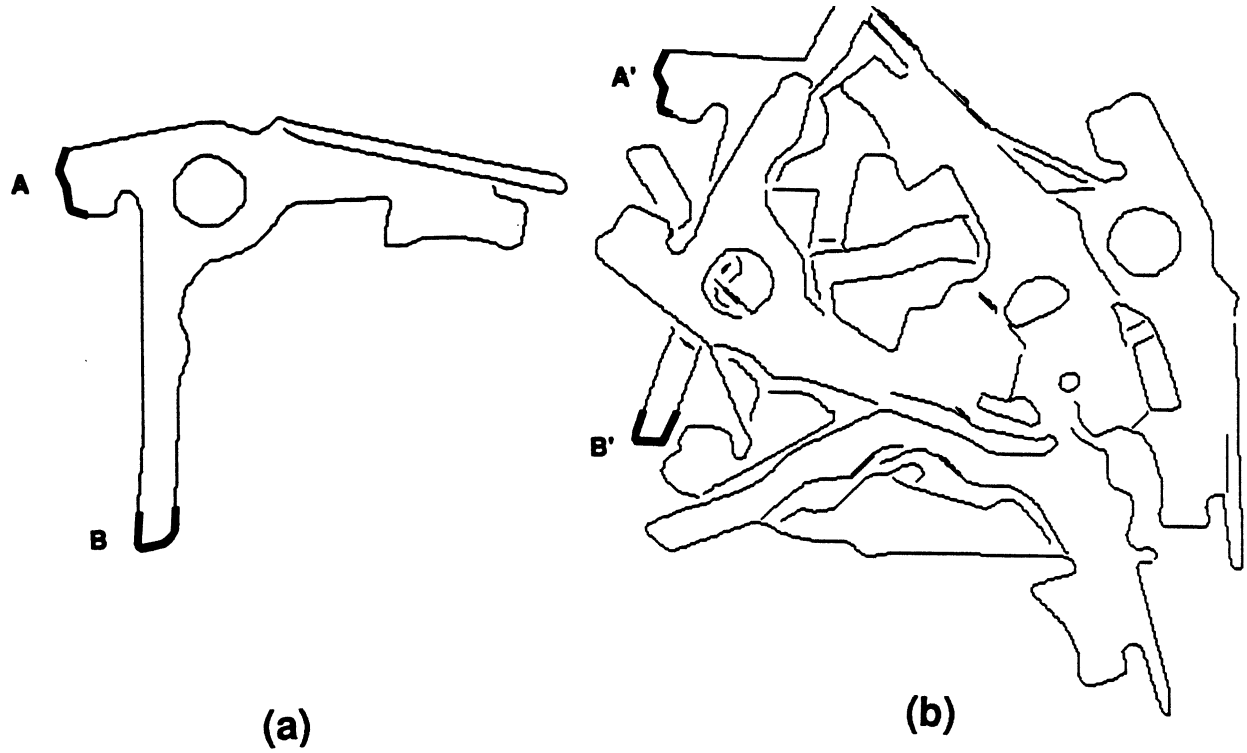

Fig. 2.3. Part and occluded parts.

global hypothesis, this tabulation must be larger than that generated by any other local hypotheses. No use is made of the relationship between the two local hypothesis. This is a much weaker method of combining local hypotheses than the method described above. It is not inconceivable that a few incorrect local hypotheses could easily cause the approach to select a incorrect global hypothesis.

In fact, the generation of incorrect local hypotheses often arises when features are not carefully chosen. In these cases a large number of false matches between features of the boundary and the image may occur with the result that many incorrect local hypotheses are tabulated.

Now that we have examined the Hough transform, we will review several algorithms based on the strategy. We begin with the generalized Hough transform of Merlin and Farber.

P. M. Merlin and D. J. Farber, "A Parallel Mechanism for Detecting Curves in Pictures." Merlin and Farber propose a part recognition strategy similar to the example given above [MeF75]. They assume that the boundary of a part is parameterized by its orientation and centroid location and that the features of the boundary are its edge points. They call the strategy the "generalized Hough transform" since it can be used to recognize arbitrarily shaped boundaries. Unfortunately, edge points are indistinguishable from one another and many false matches occur between edge points. This often results, as discussed above, in an incorrect global hypothesis for the pose of the parts in an image.

D. H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes." Ballard modifies the generalized Hough transform to improve the performance [Bal81]. Edge points of the boundaries of the part and of the image are assumed to correspond only when they have the same slope angle. With this modification fewer incorrect local hypothesis are tabulated, but false global hypotheses are still often generated.

G. Stockman, S. Kopstein, and S. Benett, "Matching Images to Models for Registration and Object Detection via Clustering." Stockman et al. use a Hough transform with *vectors* as features [SKB82]. Vectors are formed by connecting subfeatures, e.g., corners and holes in the part. More precisely, the tail of the vector is placed at one subfeature and the head of the

vector at a second subfeature. Vectors are assumed to match if they have the same length and the same subfeatures at their heads and tails. The parameters vectors, $\mathbf{p} = (\theta, u, v)$, ($S$ is assumed to be 1) of the boundary that allow a match between vectors of the part and vectors of the image are tabulated in a θ–u–v histogram.

D. H. Ballard and D. Sabbah, "Viewer Independent Shape Recognition." Ballard and Sabbah also use vectors as features. The vectors are derived from the straight line segments taken from a polygon approximation of the boundary of a part. The normal S–θ–u–v histogram, however, is divided into two disjoint histograms, an S–θ histogram and a u–v histogram [BaS83]. Parameters are determined in two stages. The values of S and θ are tabulated first, and the most frequent S and θ are selected as the scale and orientation of the part respectively. The values of u and v are determined, again through tabulation.

This strategy reduces the dimensionality of the problem from 4 dimensions to 2 dimensions with the obvious advantage that less effort is needed to search the two 2-dimensional histograms for modes than one 4-dimensional histogram. Unfortunately, reliability suffers. Separating the 4-dimensional S–θ–u–v space histogram into two 2-dimensional histograms increases the likelihood that modes in either of the two histograms will overlap. This overlap can cause a mode itself which in turn may be incorrectly interpreted as a global hypothesis for the pose of the part.

J. Segen, "Locating Randomly Oriented Objects from Partial View." Segen separates the S–θ–u–v histogram into four disjoint histograms, one for each parameter [Seg83]. He computes the parameters in the sequence, S, θ, u, and v. Parts are approximated by polygons and corners of the polygon are chosen as features. Unfortunately, overlapping modes in each of these histograms is again likely, again making the determination of the most frequent parameter vector, $\mathbf{p}^*$, unreliable.

M. W. Koch and R. L. Kashyap, "A Vision System to Identify Occluded Industrial Parts." Koch and Kashyap [KoK85] use an approach similar to that of Segen in that corners are used as features. However, unlike Segen the parameter vectors, $\mathbf{p}$, are tabulated in one θ–u–v histogram (S is assumed to be 1). Local hypotheses for parameter vectors are determined

by fitting corners of the part to similar corners in the image. Because the strategy appeared to be one of the better correlational strategies, we investigated it in the more detail than the others. Problems with the approach soon became apparent (see results presented in Chapter III).

2.2.2. Axial

An axial representation is a shape descriptor of a part. We begin with the SAT shape descriptor due to Blum and Nagel.

H. Blum and R. N. Nagel, "Shape Description Using Weighted Symmetric Axis Features." Blum and Nagel propose an axial shape descriptor called the symmetric axis transform (SAT) [BlN78]. An SAT can be constructed using a circular primitive. More precisely, a closed boundary of an part is described as by the collection of its maximal discs. Maximal discs are discs that are tangent to the boundary at least at two points and have the property that they are not contained in any other disc inside the boundary (see Fig. 2.4a). In general, the shape of any object is the union of its maximal discs. The SAT consists of two parts: the locus of the centers of the maximal discs, called the symmetric axis, and the radius of the disc at each point, called the radius function. Figure 2.4b shows the symmetric axis for a rectangle. In practice the radius information is discarded leaving only the symmetric axis as a descriptor.

Unfortunately, SAT's are extremely sensitive to boundary noise, that is, small noises can introduce considerable distortion in an SAT, making the descriptors impractical for use in most occluded part recognition problems.

R. B. Kelly, H. A. S. Martins, J. R. Birk, and J. D. Dessimoz, "Three Vision Algorithms for Acquiring Workpieces from Bins." Kelly et al. explore three algorithms for the location of grasp points of a part in a pile. These are a "shrinking" algorithm, a "collision fronts" algorithm, and a "parallel-jaw filter" algorithm [KMB83]. (The latter will be discussed in the section on region correlation.)

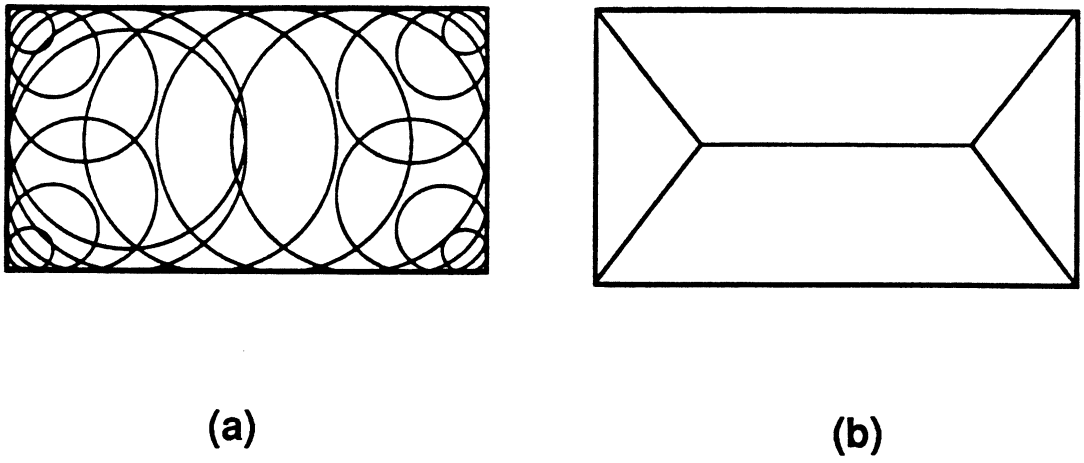

Fig. 2.4. Maximal discs representation of SAT.

The primary use of the shrinking algorithm is to locate planar surface patches where a vacuum cup gripper can be applied. The strategy is to locate exposed areas by eroding a binary image created from the original gray level image to an axis. With this method, areas can be found that are sufficiently broad so as not to be totally eroded. These are candidates grip points.

The collision front algorithm, on the other hand, propagates the edges of the image towards the middle of the parts. When a propagating edge encounters an edge being propagated from an opposite direction a collision point is formed. A set of collision points form a collision front which gives an axis of the part.

These algorithms are intended to circumvent part recognition in the POP problem by locating grasp points of occluded parts from which parts can be picked up and then recognized.

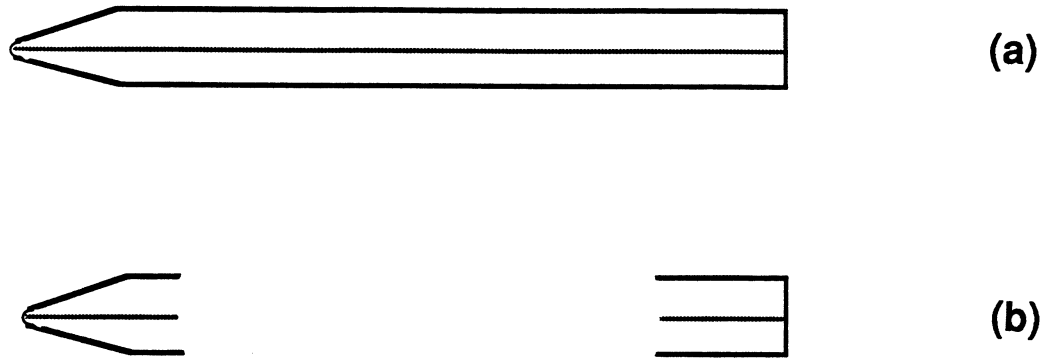

Fig. 2.5. SLS of normal and partially occluded part.

M. Brady and H. Asada, "Smoothed Local Symmetries and Their Implementation." Brady and Asada describe a representation of two-dimensional shapes called "smoothed local symmetries" (SLS) [BrA84]. SLS's are descriptors proposed for inspection and part recognition and are closely related to SAT's. The SLS descriptor is an axis constructed as the midpoint of the local symmetries of a part. It is, therefore, not affected by defective or occluded portions of the boundary that are not part of the symmetry. The SLS tends to be a more robust axial description of a part than the SAT.

However, an SLS descriptor throws away much of the structural information of a part, often leaving an ambiguous representation. For example, consider the boundary of a pen shown in Fig. 2.5a. The SLS of the pen is shown along the axis of the pen. When the pen is occluded the SLS of both the pen nib and pen top are identical straight lines (see Fig. 2.5b). It is impossible to tell from the SLS descriptor which part of the pen is visible in the image, even though there is

sufficient structural information in the original boundary to locate the pen.

The SLS descriptor is useful for locating grasp points for parallel grippers, but because it can often be interpreted ambiguously, it is not the best representation for partially occluded parts.

2.2.3. Region

Region correlation is basically a brute force approach to part recognition. Grey level regions taken from an image of the part are directly correlated with regions in the image.

Region correlation, although fairly direct to apply, is not always a workable approach. For example, correlation of a large region of a part to an image is not practical—in most POP images large regions of a part are not visible. On the other hand, correlation of a small region of the part to an image yields poor results because many regions in the image often correlate to the region with values greater than or equal to that of the correct region in the image.

In the following we will review an algorithm based on the matched filter approach. (Matched filter is just region correlation compensated for noise.)

R. B. Kelly, H. A. S. Martins, J. R. Birk, and J. D. Dessimoz, "Three Vision Algorithms for Acquiring Workpieces from Bins." Kelley et al. apply a matched filter in their "parallel jaw algorithm" to locate heuristically chosen hold-sites for parts [KMB83]. A region of the part that corresponds to a good hold-site is correlated to the image. Kelly et al. sample each of the three independent viewing angles at m views. They correlate the m^3 views of an $m \times m$ region of a part to an $n \times n$ region of the image. This requires $O(n^2 m^5)$ computations. It is an impractical number unless the regions are very small. However, as discussed above, good results are not obtained unless the regions are large.

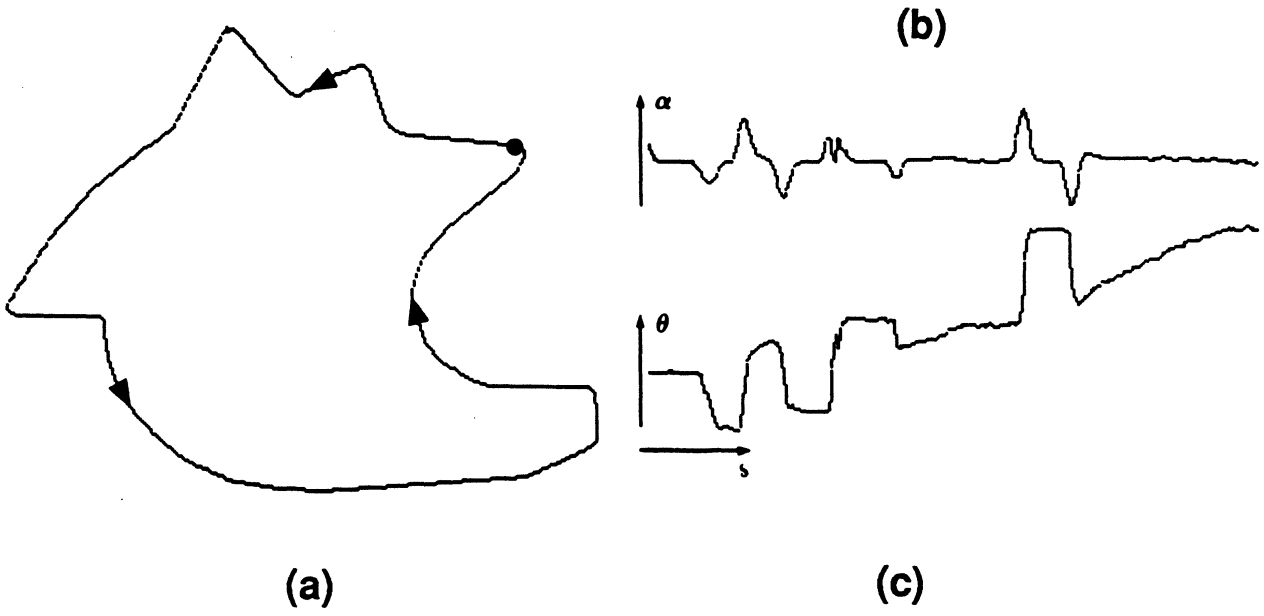

Fig. 2.6 Cartesian, α–s , and θ–s representations of a part.

2.2.4. Slope Angle/Curvature Representations

The slope angle-arclength (θ–s) and curvature-arclength (α–s) representations of boundaries are often used in part recognition. These representations have been discussed in the Introduction. Figure 2.6a-c shows equivalent cartesian and α–s and θ–s representations of a boundary. The dot on the contour represents the point of origin from which the arclength is measured. Positive arclengths are measured counterclockwise around the contour.

Some of the properties of these two representations follow. The θ–s and α–s representations are intrinsic coordinate representations and are thus invariant to translation. The α–s representation is also invariant to rotations in the plane of the boundary. Furthermore, the θ–s representation itself is invariant, within an offset, to rotations, i.e., rotations by a positive angle cause the slope angle, θ, to be incremented by the rotational angle. Because of these invariant properties the θ–s and α–s representations are efficient representations in which to compare

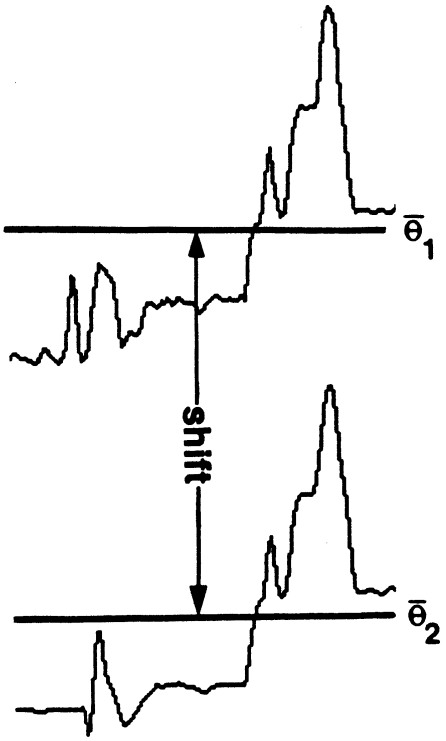

Fig. 2.7. Comparing boundaries in the θ–s representation.

boundaries of a part to those in an image with unknown orientation. To compare two boundaries, their $\theta(s)$ functions $\theta_1(s)$ and $\theta_2(s)$ (see Fig. 2.7) are shifted along the θ axis so that their means $\bar{\theta}_1$ and $\bar{\theta}_2$ have the same value. Values of θ for the two functions are then compared at uniformly spaced values of s. Because the arclength may not be measured from the same point of origin, the comparison, in general, must be performed for all possible choices of origin for one of the boundaries, i.e., over all different s offsets between the two boundaries. In comparing two boundaries represented by their $\alpha(s)$ functions no shift in θ is required but the boundaries must still be compared at all values of s. One of the shortcomings of both of these approaches is that the θ–s and α–s representations of partially occluded boundaries do not closely resemble the original θ–s and α–s representations of the unoccluded boundaries. Occlusions often introduce wide arclength separations between features of the θ–s and α–s representations of the original contour. As a consequence the representations can only be guaranteed to to be locally the same for the visible

segments of the boundary.

Another use for the α–s representation is in locating critical points. These points are invariant to rotations, translations, scale changes and are insensitive to arbitrary tilt changes (see [Mar84]). The change in curvature at a critical point is generally substantially greater than the noise in the curvature function, $\alpha(s)$. Therefore, even when the $\alpha(s)$ function of a boundary is noisy, critical points on the boundary can still be accurately located.

Now that we have examined the slope angle and curvature representations, we will review several algorithms based on these representations. We begin with a paper on critical points by Freeman.

H. Freeman, "Shape Description Via the Use of Critical Points." Freeman uses an augmented set of critical points. To the normal set he adds end points, intersections, and points at which boundaries are tangent to each another [Fre77]. The critical points are connected with chords and a set of features of the chords and of their corresponding boundary sections are computed. The features are those shown in Fig. 2.8: the length of the boundary section between critical points, the total "bay" area lying between the boundary section and the left of the chord, the total "peninsula" area lying between the boundary and the right of the chord, the maximum "bay" depth, and the maximum "peninsula" depth.

To make the features scale invariant, Freeman determines the chord length between critical points and divides linear features by the chord length and area features by the chord length squared. He determines the pose of the part in an image by finding corresponding invariant features of the part and image. Shortcomings of the approach are that features calculated from closely grouped critical points are unlikely to have accurately calculated values, and features calculated from widely separated critical points in the image are likely to be distorted by occlusions.

J. W. McKee and J. K. Aggarwal, "Computer Recognition of Partial Views of Curved Objects." McKee and Aggarwal build a library of complete contours coded in the θ–s representation [McA77]. They allow only one part to be present in the image. However, the

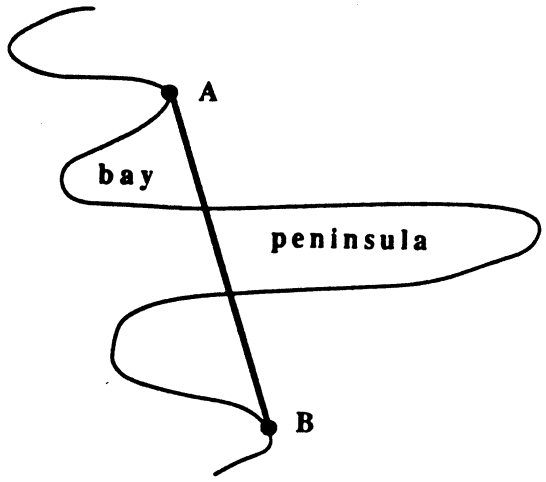

Fig. 2.8. Freeman's features.

part's boundary may be composed of several contours. During recognition, they compare contours in the image to all the contours in the library. The part whose contour in the library best matches an image contour is assumed to be the part. One shortcoming of the approach, beside the limitation of one part in the image, is that complete contours of a partially occluded part may be distorted so severely that not even the best match of an image to the library will be a correct match.

K. R. Yam, W. N. Martin, and J. K. Aggarwal, "Analysis of Scenes Containing Several Occluding Curvilinear Objects." Yam, Martin, and Aggarwal [YMA80] use the same library approach as McKee and Aggarwal in [McA77] for a slightly more general problem. They allow only one *type* of part to be present in the image, although there may be multiple instances of the part. In addition, they assume that every contour in the image is a closed contour and that at least one contour is completely visible. By comparing every image contour to every part

contour in the library, they attempt to find a match for one of the completely visible image contours. Finding a match identifies the type of part in the image. Once the type of part is identified the poses of the other contours in the image are determined by correlating their θ–s representations to the θ–s representation of the identified contour. Shortcomings of the approach are that every contour of the image must be compared to every contour in the library, and, as before, the contours may be so severely distorted that the best match to the library may be incorrect.

W. A. Perkins, "A Model-based Vision System for Industrial Parts." Perkins [Per78] models a contour by *concurves*, a concatenation of straight line and circular arc segments that approximate a contour. To recognize a part, Perkins compares concurve features such as the total concurve length, radius of arcs in the concurve, magnitude of total angular change, number of straight lines in the concurve, and number of arcs in the concurve of the part to those of the image. If there is sufficient agreement between the part and image concurve features, the algorithm correlates the original contour of the part to the image contour in θ–s space to find the location of the part in the image. This last step is similar to that used by McKee and Aggarwal in [McA77].

Shortcomings of the approach are that it relies on the assumption that a large fraction of the contours are visible in order to make the concurve approximation, and it inherently assumes that parts are well approximated by circular arc and straight line segments.

W. A. Perkins, "Simplified Model-based Part Locator." Perkins simplifies his previous approach by correlating the $\theta(s)$ representations of a part contour and the image contours to find points at which the contours have a strong correlation [Per80]. After aligning the contours in cartesian space at one of these points, his algorithm constructs a circle centered about the point of alignment. He then follows the circle around until it crosses at least one point on the image boundary and a point on the part boundary. These points are aligned by rotating the contour of the part about the center of the circle. From this final alignment the pose of the part is found.

B. Neuman, "Interpretation of Imperfect Object Contours For Identification and Tracking." Neuman approximates part and image boundaries by polygons and compares corners

formed by adjacent sides of the polygons [Neu78]. Neuman assigns to corners attributes corresponding to the interior angles of the corners and the lengths of the sides of the corners. If the attributes of a corner of the part matches those of a corner in the image, a hypothesis for the pose of the part is generated based on the match. The corners are aligned and the complete boundary of the part is fit to the image. This step is used to verify the hypothesis. If the part fits sufficiently well, the section of the polygon in the image which correspond to the part is removed, and the algorithm then attempts to locate other parts in the image. If several part corners fit the image corner, the part that best fits the image polygon is selected as the correct part.

J. D. Dessimoz, M. Kunt, and J. M. Zurcher, "Recognition and Handling of Overlapping Industrial Parts." Dessimoz, Kunt, and Zurcher correlate the curvature representations of contours of the image to that of the part [DKZ79]. They assume that a few parts are completely visible at the top of a pile with little overlap so that significant correlation can be obtained.

A. H. Bond, R. S. Brown, and C. R. Rowbury, "The Effect of the Environmental Variation Upon the Performance of a Second Generation Industrial Vision System." Bond et al. [BRB83b] segment part boundaries into straight lines and circular arcs using an approach similar to that of Perkins in [Per78]. Image boundaries, on the other hand, are not segmented but left as chains of edge points. They correlate adjacent pairs of segments of the part at every location on the image boundary in the θ–s representation. Finally, the part is fit to the image in cartesian space to confirm the match.

D. H. Marimont, "A Representation for Image Curves." Marimont determines a set of critical points that he proposes to use to locate tilted parts over a wide range of scales [Mar84]. Scaling of a part is simulated by smoothing the boundary of the part with Gaussian filters with different standard deviations. He argues that the reduction in resolution of a boundary obtained from smoothing the boundary with a Gaussian filter can simulate the natural reduction in the resolution of the boundary that occurs when the boundary is scaled down. The Gaussian with small standard deviation simulate small reductions in scale while those with large standard devia-

tion simulate large reductions in scale. A set of critical points are extracted from contours filtered by Gaussians of different standard deviations. From this set of critical points a fixed number of critical points that best characterize the contour at each scale, i.e, that are well separated and represent different critical points on the contour, are obtained using a dynamic programming algorithm. These points are used to recognize the part over a range of scales in the presence of occlusion.

Marimont also discusses results that he has obtained on the insensitivity of critical points and zero crossing in the curvature function, $\alpha(s)$, to arbitrary changes in orientation; a property useful for locating tilted parts.

This concludes our review of correlation approaches. In the following we begin our review of structural approaches.

2.3. Structural

Structural strategies fall into two classes: syntactic and synthesis. We begin with a description of the class of syntactic approaches.

2.3.1. Syntactic

Syntactic approaches are an attempt to apply formal language theory to pattern recognition problems [Fu74, Pav77]. To accomplish this goal, boundaries are regarded as sentences in a *language* defined by a formal grammar. More precisely, each boundary of a part has an associated language and grammar. In addition, an image boundary is assumed to be a sentence—an image sentence—in the language of one of the parts. In recognizing a part, the grammar associated with the part is used to *parse* the image sentence to check if it is a valid sentence. Parsing an image sentence is similar to parsing an English sentence. Image sentences are parsed into grammatical

primitives and combinations of primitives to determine if they are valid sentences for the part.

A grammar, $G = (N, \Sigma, P, S)$, for a part, consists of a finite set of nonterminals, N, a finite set of primitives, Σ, a finite set of production rules, P, and a starting symbol, S. For example, for a part such as the hammer shown in Fig. 2.9, the primitives would be a, b, c, d. The nonterminals are substructures of the hammer that are not primitives such as *hammer*, *head*, and *handle*. The production rules are rules that tell how the nonterminals are related to the primitives and other nonterminals:

$$hammer \rightarrow handle + head,$$

$$handle \rightarrow a + b + c + d + e.$$

where "+" represents the operation of concatenation. Finally, the starting symbol is *hammer*.

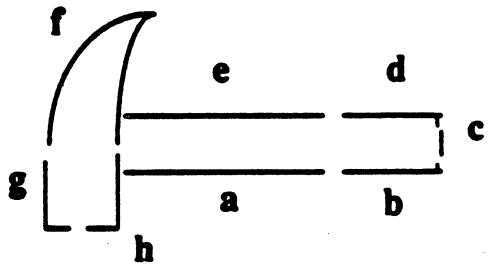

Fig. 2.9. Syntactic hammer.

A *parser* is used to find a sequence of production rules that can derive an image sentence from a starting symbol. One method of parsing is "bottom up" parsing. The primitives found in the image are examined and the grammar is used to build a representation of the part from the primitives. The production rules are used backwards—in the reverse direction of the arrow—to combine primitives and nonterminals in an attempt to derive the starting symbol. To locate the hammer, a bottom up parser would combine the *a*, *b*, *c*, *d*, and *e* primitives to construct a *handle*, and then combine the *handle* with the *head* to form the *hammer*.

Sometimes attributed grammars are used in a syntactic strategy to add more decision making capability. An attributed grammar consists of two sets of rules: first a set of syntactic production rules as described above to define the structure of a part, and second a set of semantic attribute rules for computing attributes of the part. An image sentence is insufficient to recognize the part unless the attributes of the image sentence are also the same as the attributes of the part. For example, attributes of length could be associated with each primitive of the hammer in the previous example. An attribute rule would be one that summed the lengths of each primitive to compute a total length of the perimeter. In the previous example, with an attributed grammar the hammer would be recognized only if the lengths of the primitives summed to an acceptable value.

With partial occlusion, error correcting grammars are needed. Given two sentences of primitives that match, there are three types of *single* primitive errors that can cause a mismatch: inserting an extra primitive, deleting a primitive, and substituting an incorrect primitive for a correct primitive. To account for single primitive errors, all possible sentences with these errors must be allowed by the parser. As a consequence the language of the part must be expanded to include all possible sentences with single primitive errors. For example, for the hammer the correct image sentence for the *handle* is *abcde*. To accommodate a single primitive substitution error the sentences *bbcde, cbcde, dbcde, ebcde, aacde, accde, adcde, aecde, abade, abbde, abdde, abede, abcae, abcbe, abcce, abcee, abcda, abcdb, abcdc,* and *abcdd* must be added to the language defining the part. If more than a one primitive error is allowed, the language must be further

expanded. Clearly, incorporating even simple error detection results in a combinatorial explosion in the size of the language. Handling more complex errors would make the approach even more intractable.

Now that we have examined the syntactic strategies, we will review several algorithms based on these strategies. We begin with algorithm using attributed grammars due to You and Fu.

K. C. You and K. S. Fu, "Distorted Shape Recognition Using Attributed Grammars and Error-Correcting Techniques." You and Fu [YoF80] extend the work of attributed grammars of Tsai and Fu [TsF80] to include error correction so that parts with distortion or partial occlusion can be recognized. A modified error correcting parsing algorithm for single errors is used.

T. Pavlidis and F. Ali, "A Hierarchical Syntactic Shape Analyzer." Pavlidis and Ali [PaA79] do not parse a complete boundary from image primitives as You and Fu [YoF80] but rather a set of high-level attributed nonterminals. These nonterminals are called "arcs" and have attributes of type, size, and orientation. Types were sharp protrusion, sharp intrusion, convex corner, concave corner, convex arc, and concave arc. Sizes were small, medium, large, and huge. Orientations corresponded to the points of the compass.

A part is recognized by comparing arcs of the boundary of the part to those of the boundary of the image. Weights are assigned to pairs that match based on the number of attributes shared by the pair. The weights from all the pairs of matching arcs are summed to give a measure of correspondence between the part and the image. If the measure exceeds a predefined threshold, the part is assumed to correspond to the image.

2.3.2. Synthesis

Conceptually, synthesis strategies are descendents of the syntactic strategies discussed in the previous section. Synthesis strategies preserve the structural descriptive power of the syntactic approach, while attacking problems of noise and partial occlusion. As a consequence, synthesis strategies give more robust solutions to the two-dimensional POP problem than syntactic strategies.

As with syntactic strategies, synthesis strategies rely on a set of primitives extracted from a part's boundary. For example, they may include primitives such as corners, line segments, circular arcs and holes. A set of production rules are applied to the primitives found in the image, and a part is "synthesized."

More precisely, a synthesis strategy can be viewed as a tree search where the goal of the search is to find a part in the image. For example, we will assume there exists a set of corner primitives common to the part and the image. A synthesis strategy has a set of production rules that determine the order in which primitives are selected. Suppose primitive 1 (see Fig. 2.10) is selected as the first primitive. Primitive 1 is compared to all similar primitives in the image and a tree is constructed in which the children of the root node represent the sites in the image that match primitive 1. In order to determine which node in the search tree will be expanded a measure of the similarity between part primitives and image primitives is estimated.

One measure of similarity often used is the maximum attainable similarity (MAS). The MAS is the maximum similarity that would be attained if the remaining unmatched primitives of the part perfectly matched primitives in the image. Figure 2.10 shows an example where primitive 1 is matched to three different image sites. Estimates of 98%, 99%, and 95% are assigned at the first level indicating the similarity that the notched rectangle would have to the image if all the remaining primitives, 2, 3, 4, and 5, perfectly matched the image. Production rules select the next primitive, in this example primitive 2, to be matched. The location of the part-boundary (shown lighter in Fig. 2.10) established by the match of primitive 1 serves as a guide to the search for a match to primitive 2 and subsequent match attempts.

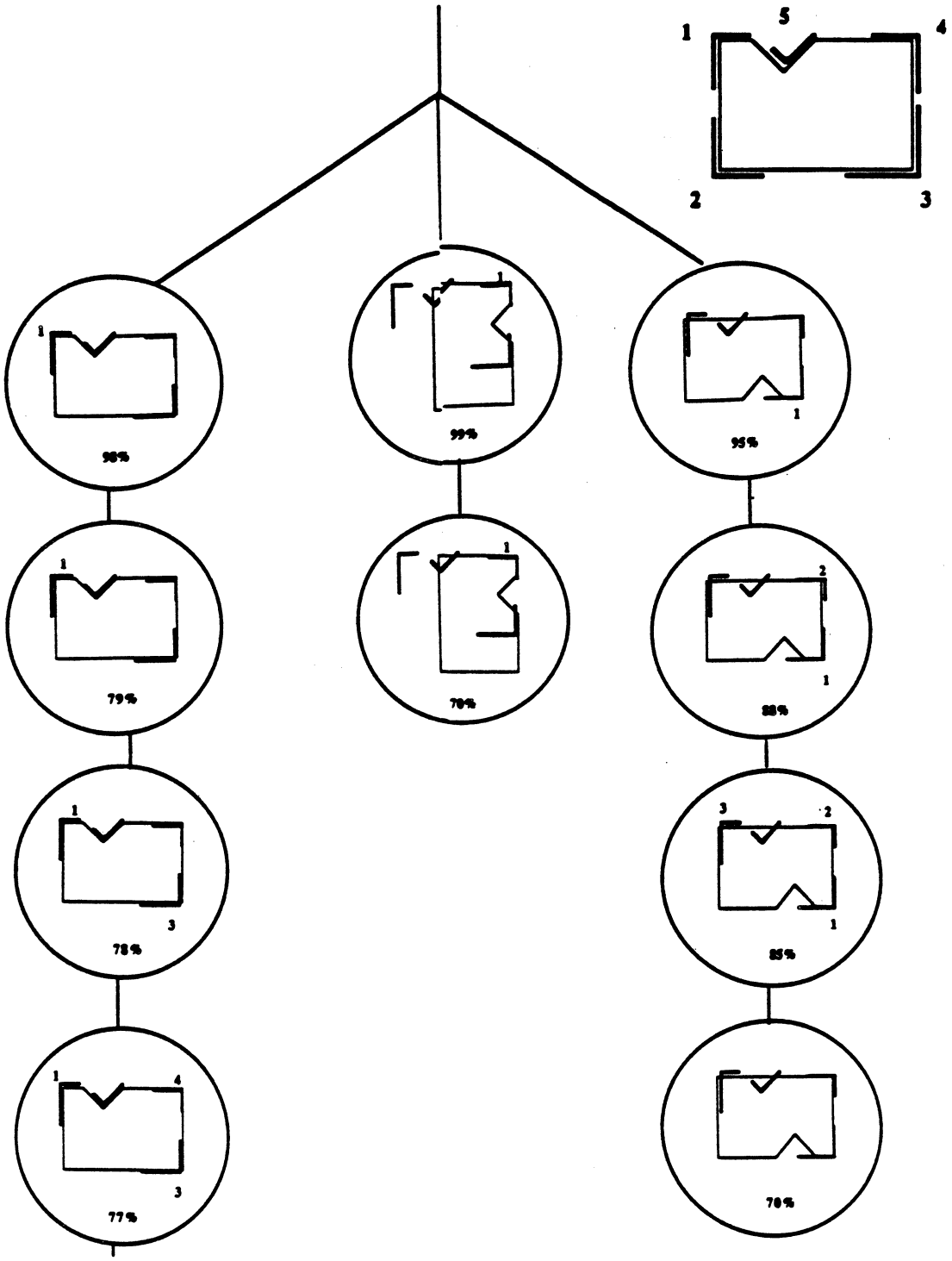

Fig. 2.10. Search tree for synthesis.

Different search strategies are possible depending on the order in which nodes of the search tree are expanded. A purely syntactic recognition strategy results if only the nodes with complete similarity, i.e., 100% similarity, are expanded, while a hill climbing or best fit strategy results if the node with the highest similarity is expanded. Finally, an A^* heuristic search [Nil80] strategy results if the expansion of the nodes is directed by the estimate of the maximum attainable similarity. Whichever strategy is selected, the desired result is to reach a goal node where a complete boundary of a part is located from primitives at some image site.

The major weakness of the synthesis strategies is in obtaining a good starting feature, or *seed feature*, for the tree search and in the methods used in verifying the best image site. To obtain a good seed feature, the strategy must choose some special feature such a corner or longest line segment of the part. If a more common feature were chosen too many image sites would have corresponding features and the tree would have too many children for an efficient search. However, by limiting the choice of seed feature, there is a danger that many of the chosen seed features will be occluded or distorted. A second weakness is that in the synthesis approach the amount of perimeter matched is generally used to measure the quality of a fit. This measure has the shortcoming that sometimes common features of many parts such as straight line segments or circular arcs have more weight in deciding the quality of a fit than features that are unique to the part that is to be located.

Now that we have examined the synthesis strategies, we will review several algorithms based on these strategies. We begin with a strategy proposed by Tropf which incorporates a heuristic search.

H. Tropf, "Analysis-by-Synthesis Search for Semantic Segmentation — Applied to Workpiece Recognition" Tropf's strategy, "analysis-by-synthesis," is essentially the heuristic search strategy based on maximum attainable similarity outlined above [Tro80]. The shortcoming of this approach is its reliance on special set of features, i.e., corners.

W. Haettich, "Recognition of Overlapping Workpieces by Model-Directed Construction of Object Contours." Haettich uses the same synthesis strategy as Tropf with line

segments as primitives [Hae83]. His search strategy, however, is a "best first" strategy. With this strategy fewer nodes are expanded, and only those in which the primitives at the image site bear a strong resemblance to the ideal part are enlarged.

P. Rummel and W. Beutel, "Workpiece Recognition and Inspection by a Model-Based Scene Analysis System." Rummel and Beutel base their work on Tropf's with a few differences [RuB84]. The major difference is that a larger set of features than Tropf's are employed: corners, circular arcs and straight lines serve as primitives. The algorithm orders the primitives by the degree of their importance, and compares them in this order to similar primitives in the image. The importance of a primitive is assigned manually.

A. Riad and M. Briot, "Identification and Localisation of Partially Observed Parts."

Riad and Briot use a heuristic search strategy, but in addition to syntactic information, they incorporate relational structures in the description of an object and perform both synthesis recognition and subgraph matching [RiB81].

N. Ayache and O. D. Faugeras, "A New Method for Recognition and Position of 2-D Objects." Ayache and Faugeras [AyF84] propose conceptually the same approach as that of Rummel and Beutel [RuB84]. Contours are approximated by polygons and line segments that form the sides of the polygons are used as primitives. Differences are that Ayache and Faugeras's strategy uses "preferred" segments as seed features and Kalman filtering to update estimates of pose. Preferred segments are the longer line segments of the polygon approximations of the boundaries. These segments do not occur often in the image, and hence fewer image sites are chosen, making the approach more efficient. The approach is one of the better structural approaches, and will be examined in more detail in Chapter III.

2.4. Relational

Relational strategies tend to use more of the information available in an image than the correlational approaches (see Sec. 2.2.1). Relational approaches, however, are sometimes overwhelmed by the combinatoric growth in the number of relations between features. In the following review, relational strategies are broken into three distinct classes: relational structures, invariants, and relaxation. In the first class we will see how the incorporation of relational information has led to a relational approach that is intractable. We begin with a discussion of relational structures.

2.4.1. Relational Structures

A *relational structure* is commonly defined as a set of features together with properties of and relations between features. Figure 2.11a shows an industrial part with its relational structure represented by a graph. The features of the part are corners, holes, and line segments, These features, in turn, have properties of type and size and have relations of relative distance and relative orientation. The nodes of the graph in Fig. 2.11b represent the features, and the arcs of the graph represent relations between features.

A relational graph is convenient method of mapping a recognition problem into a computer. For instance a part can be matched to an image by graph-matching. More precisely, a subgraph derived from the part is matched to a subgraph derived from the image.

The problem of subgraph matching is often formulated in terms of a *compatibility graph*. A compatibility graph is a graph with nodes that represent *assignments*. In our application an assignment would be an assignment of a part feature to an image feature that had the same properties. Figure 2.12a shows an image containing features of the industrial part of Fig. 2.11a. In Fig. 2.11a corner C_1 has the same properties of type and size as corners C_A and C_B in the image and, therefore, possible assignments are C_1–C_A and C_1–C_B. Figure 2.12b shows the compatibility graph of the image and the part features where the arcs in the graph correspond to compati-

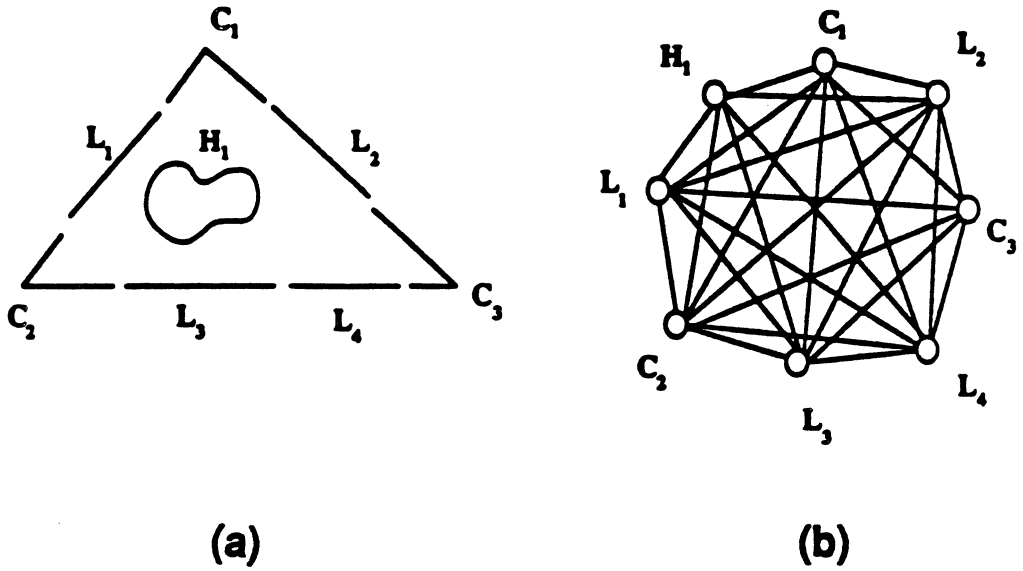

Fig. 2.11. A part with its relational structure.

bilities between assignments. A *compatibility* is defined as a pair of assignments that preserves relations between features. For example, the assignment of corner C_1 to corner C_A is compatible with the assignment of hole H_1 to hole H_A because these assignments preserve the distance and orientation relations between C_1 and H_1. The assignment of corner C_1 to corner C_B, however, is not compatible with the assignment of hole H_1 to hole H_A.

With a relational strategy a part is assumed to be at the location of the part in the image where the most feature assignments are compatible. In the compatibility graph this location is represented by the largest totally connected subgraph, i.e., the subgraph in which every node is connected to every other node. This type of subgraph is called a *clique*. A clique that is not properly contained in another clique is called a maximal clique. In general, there may be more than one clique in a compatibility graph so that in locating a part we are interested in finding the largest maximal clique. Unfortunately, this problem is NP complete [GaJ79]. Johnson [Joh74]

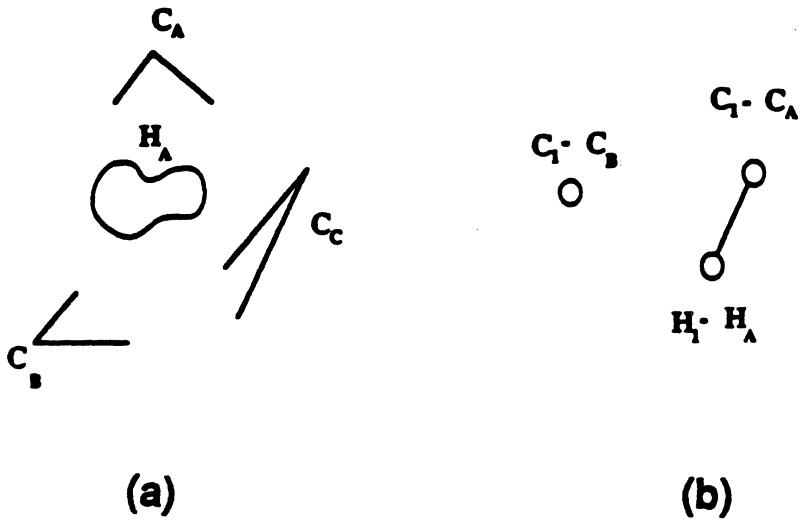

Fig. 2.12. Image and part-image compatibility graph.

has surveyed several fast heuristic methods for finding the largest cliques and has noted that their worst case performance is extremely bad. Heuristic techniques have been developed but in all cases only a few features can be used if the approach is to remain tractable.

Now that we have examined the basic concepts behind relational structures, we will review several algorithms based on their use. We begin with a algorithm due to Ambler et al. that uses a largest maximal clique approach

A. P. Ambler, H.G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone, "A Versatile Computer-Controlled Assembly System." Ambler et al. are among the first to treat the POP problem as a subgraph matching problem [ABB73]. To identify a part, they locate the largest maximal clique in a compatibility graph between the part and the image. They use this approach to locate industrial parts with imperfections, but in practice do not attempt to

locate overlapping parts.

R. C. Bolles and R. A. Cain, "Recognising and Locating Partially Visible Objects: The Local-Feature-Focus Method." Bolles and Cain base their algorithm on a simplified maximal cliques strategy called "local feature focus" During training, their algorithm isolates "focus" features that are unique to one pose of the part. A focus feature is a configuration of a central feature, such as a prominent hole or corner, surrounded by nearby features, holes and corners. To recognize the part, they build a compatibility graph between the set of focus features and features extracted from the image and finds the largest maximal clique. Then, using the location of the clique in the subgraph, they hypothesizes a pose of the part in the image. They test this hypothesis by fitting a template for the part to the image. The shortcoming of this approach is that, because it relies on a maximal clique algorithm, only a small set of features can be used in order to obtain reasonable performance. However, as a general rule, the fewer features incorporated into the algorithm the less robust the algorithm. This approach is one of the more interesting relational approaches, and will be examined in more detail in Chapter III.

W. E. L. Grimson and T. Lozano-Perez. "Model-based Recognition and Localization from Sparse Range or Tactile Data." Grimson and Lozano-Perez propose a strategy in which a few segments of the boundary of a part can be used to locate the part [GrL84]. Parts are modeled as polygons. Pairs of polygon segments of the part are compared to pairs of polygon segments of the image. Using geometric relations between segments, e.g., distances between centers of segments, angles between normals to segments, and angles relative to the segment normals of vectors between corresponding segments, they attempt to find configurations of segments pairs in the image that are unique to one pose of the part and thus located the part. This approach is similar in many ways to the one proposed by this thesis.

2.4.2. Invariants

One strategy for general object recognition is to find a set of invariant features of an object and to compare these invariant features to those taken from an image. While many features have been discovered that are invariant to rotation and translation, few features (color is one) are invariant to partial occlusion. One solution is to work with invariant features of subobjects of an object.

R. Brooks, "Model-Based Three-Dimensional Interpretations of Two-Dimensional Images." Brooks in his algorithm ACRONYM decomposes objects into structures of subobjects modeled by "generalized cylinders" [Bro83]. Generalized cylinders have properties that make them invariant over a wide range of views. Therefore, subobjects, modeled by generalize cylinders, should be recognizable over a wide range of views.

Generalized cylinders, introduced by Binford in [Bin71], describe three-dimensional volumes. Each cylinder is specified by a space curve, called a spine, a two-dimensional cross section, and a sweeping rule. The cross section is kept at a constant angle to the spine, and is deformed according to the sweeping rule (see Fig. 2.13). Brooks choses generalized cylinders with straight spines, either circular, square, or rectangular cross-sections and sweeping rules that requires the cross-section be deformed linearly. Generalized cylinders with this choice of spine, cross-section, and sweeping rule always project into images of ellipses and trapezoids. Therefore, to recognize the generalized cylinders, Brooks need only look for features of ellipses and trapezoids in the image. In other words, the shapes ellipses and trapezoids are his invariants.

Given a set of object models, Brook's ACRONYM contains geometric reasoning rules that are used to predict shapes and relationships that might be observed in an image. For example, if a subobject can be described by a right cylinder, ACRONYM can predict that parallel lines from the sides of the cylinder and an ellipse from the end of the cylinder should appear in the image. ACRONYM also predicts "quasi-invariants" that change slowly with changes in viewpoint, e.g., the length of the cylinder would be quasi-invariant.

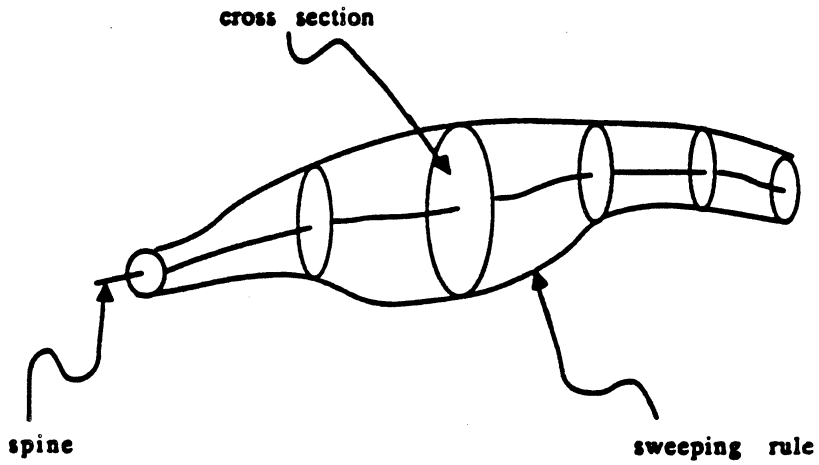

Fig. 2.13. Generalized cylinders.

To recognize an object ACRONYM looks for shapes that it predicts should be in the image. It builds an relational graph for the objects based on its predictions and an relational graph for the image, consisting of shape and their poses found in the image. ACRONYM interprets images by subgraph matching between the predicted graph of the object and the graph of the image.

C. K. Cowan, D. M. Chelberg, and H. S. Lim, "ACRONYM Model Based Vision in the Intelligent Task Automation Project." Cowan et al. point out that the success of the original implementation of ACRONYM is due to its concentration on objects that had structure that could be well approximated by skeletal descriptions, that is, objects that could be cleanly decomposed into subobjects and that exhibited clear relationships between subobjects [CCL84]. Unfortunately, industrial objects do not generally exhibit this structure. Cowan et al., therefore, extended ACRONYM by adding several new capabilities to help solve industrial problems. They increase the set of permissible generalized cylinders to a slightly larger set. They add helixes to

model springs and cylinders with holes to model tubular shapes. In addition, they add new reasoning rules to ACRONYM to account for the new subobjects. However, even with the new extensions ACRONYM is limited in its ability to recognize objects.

R. B. Fisher, "Using Surfaces and Object Models to Recognize Partially Obscured Objects." Fisher uses regions as quasi-invariant features [Fis83]. Objects are described as structures of connected surfaces. Each surface contains its own coordinate system. Connections between surfaces are specified by a relative translation, rotation, slant and tilt between pairs of coordinate systems. A set of heuristics are used to estimate a transform that would map the surfaces of the object into image regions. From this transform the pose of the part can be estimated. Unfortunately, most of the heuristics break down when the surfaces are partially occluded.

B. Bamieh and R. J. P. De Figueiredo, "Efficient New Technique for Identification and 3D Attitude Determination of Space Objects from a Single Image." Bamieh and Figueiredo approximate objects by *polyhedra* and compare invariants of the faces of the polyhedra to invariants calculated from the image [BaF85]. The transformation from a face of a polyhedron to the orthogonal projection of the face onto the image plane is an affine transformation. Therefore, Bamieh and Figueiredo combines moments of the faces to obtain invariants to affine transformations [Hu62]. The part and the image are represented by attributed graphs: the nodes represent faces of polyhedron with properties of moment invariants and angle between faces. Arcs between nodes represent adjacencies. Recognition is accomplished by matching a subgraph of the part to subgraphs of the image. The main shortcoming of this strategy is the assumption that the image can be represented as a projections of polyhedral faces and that the interior boundaries between faces exist. With non-polygonal parts, this assumption is invalid. The boundaries do not exist in the image since they are imaginary boundaries resulting from the polyhedral approximation.

2.4.3. Relaxation

Relaxation is an iterative parallel strategy in which a globally consistent solution to the problem of matching a part to an image can be obtained by guaranteeing local consistencies, e.g., guaranteeing that local primitives of the part match primitives of the image.

W. S. Rutkowski, "Recognition of Occluded Shapes Using Relaxation." Rutkowski introduces a probabilistic relaxation labeling approach as a solution to the POP problem [Rut82]. Rutkowski assigns a probabilistic labeling vector to each boundary point in the image. The elements of a vector represent the probability that the edge point can be labeled or interpreted as either one of the edge points of the boundary or as a "noise" point. The noise point is a catch-all for edge points of the image with no correspondence in the boundary. Recognition of a part, therefore, is formulated as a problem of finding the most probable interpretation of each edge point of the image.

Each edge point in the image is considered to be a node in a graph. The arcs in the graph represented *relations* between pairs of edge points. Relations between nodes included distance, symmetry, orientation of each edge point relative to a line joining their centers, and relative orientation. Relations either positively or negatively reenforced label assignments at the pair of edge points that they connect. If the label assignments are consistent with the relations, the reenforcement is positive, otherwise it is negative.

The probabilities of assigning node n a label l is denoted $P(n,l)$. The total probability of assigning a label to a node is normalized to one. Probabilities for nodes n_1 and n_2 are updated by a "reenforcement" functions, $R(n_1,l_1,n_2,l_2)$, where R takes on high value when the relations between node n_1 and node n_2 supports the conclusion that n_1 and n_2 should be labeled with l_1 and l_2 respectively and a low value otherwise.

The updating strategy is to replace the probability of a label assignment, $P(n_1,l_1)$, by a value that reflects the consistency of the label assignment with label assignments at other nodes, that is

$$P(n_1,l_1) \leftarrow \sum_{n_2,l_2} P(n_1,l_1)\, P(n_2,l_2)\, R(n_1,l_1,n_2,l_2).$$

and then to renormalize the updated probabilities,

$$P(n,l) \leftarrow \frac{P(n,l)}{\sum_l P(n,l)}.$$

The approach is essentially heuristic although it has a strong probabilistic flavor. After several iterations the relaxation algorithm arrives at a probable labeling for each image point.

B. Bhanu and O. D. Faugeras, "Shape Matching of Two-Dimensional Objects." Bhanu and Faugeras use a variation of probabilistic relaxation labeling in which a *global fit* criterion is optimized to guarantee the most consistent labeling [Bha84]. Boundaries of parts and the image are approximated by polygons. Their algorithm assigns two probability vectors to each line segment of the polygon approximation of the part. The elements of the first vector, $\mathbf{p}_i$ estimate the probabilities of assigning image segments as labels to the part segments. The part segments are nodes in a relaxation graph as in Rutkowski. The elements of the second probability vector, $\mathbf{q}_i$, represent the compatibilities of each label assignment with the two neighboring segments of a part segment. Vectors, $\mathbf{p}_i$ and $\mathbf{q}_i$ are again more heuristic than probabilistic.

Bhanu and Faugeras argue that the inner product of $\mathbf{p}_i$ with either $\mathbf{q}_i$ represents a good local measure of ambiguity or inconsistency between how segment i "thinks" it should be labeled and how its neighbors "think" segment i should be labeled. Based on this argument Bhanu and Faugeras derived two global criteria based on the two $\mathbf{q}_i$'s with which they can maximize the consistency of label assignments. The algorithm optimizes these criteria with a gradient projection algorithm to obtain a labeling for each line segment in the image.

J. K. Cheng and T. S. Huang, "Recognition of Curvilinear Objects by Matching Relational Structures." Cheng and Huang [ChH82] use *chords* between critical points on the boundary as features. If a chord connects two points A and B on the boundary, they choose the following properties of the chord: the length of the boundary from A to B, the area enclosed between the boundary and the chord, and the coordinates of the center of mass of the enclosed

area. They also select a set of relations between chords such as antiparallelness, adjacency, collinearity and symmetry. Relations also contain the attribute of the distance between chord centers. This set of features and relations are used to form a relational structure.

Cheng and Huang match "stars" of the relational structure of the part to those of the image. A star is a node along with its relational arcs and the neighboring nodes to which it is linked by an arc. A assignment between a star of a part and a star of an image is a "starpair." Their algorithm uses relaxation to compute the probability of matches between starpairs.

2.5. Conclusion

This concludes our review of existing approaches. In the next chapter we will examine three of the approaches in more detail and describe the results of applying these algorithms to a test image. These results will be compared to those we obtain with our approach in Chapter VI.

CHAPTER III

ALTERNATIVE ALGORITHMS

3.1. Introduction

In Chapter 2 we selected three algorithms for detailed analysis. One was a correlational algorithm due to Koch and Kashyap [KoK85], another was a structural algorithm due to Ayache and Faugeras [AyF84], and finally the third was a relational strategy due to Bolles and Cain [BoC84]. These algorithms are characteristic of each of the three categories of POP recognition approaches. In this chapter we will describe the algorithms in more detail and report results obtained from applying the algorithms to a set of test images. The test images that we have chosen are images of a bin of identical parts. The parts are those used in a car's door lock assembly (see Fig. 3.1 and Fig. 3.2). We used this test set in order to gain some insight into exactly how the algorithms worked. In addition, the results we obtained gave us an idea of how the algorithms perform on typical industrial images. We compare the performance of our algorithm on the same set of images in Chapter VI.

3.2. Correlational Strategy

Koch and Kashyap locate parts by correlating corners of the part to those in the image. To accomplish this goal, the boundary of the part is approximated by a polygon and the vertices of the polygon with large curvature are selected as corners. Curvature at a vertex is calculated as the ratio of the exterior angle (β in Fig. 3.3) of the vertex to the average length of the polygon line segments that form the vertex.

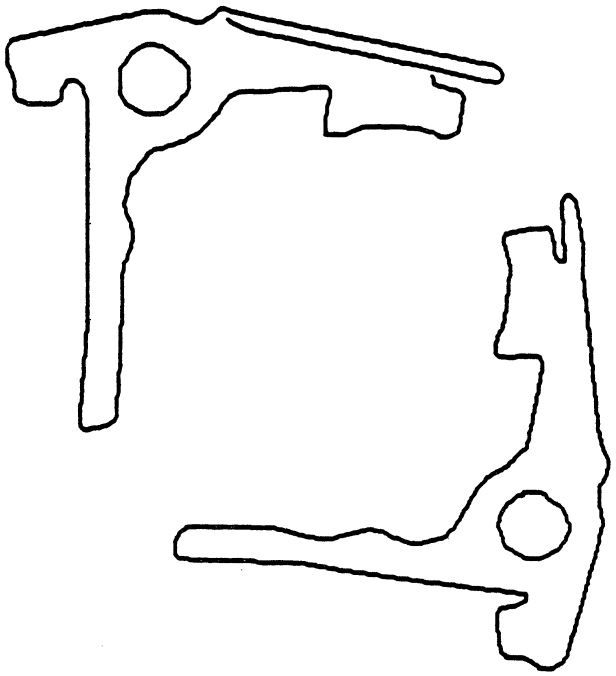

Fig. 3.1. Two sides of a door lock part no. 1.

Once the corners of the part and image have been determined, the algorithm generates hypotheses as to the location of a part in the image by matching corners of the part to corners in the image with similar attributes. Corners have associated attributes of curvature, sign of convexity, and the values of the first two moments about the vertex. The moments are calculated in the standard way from the location of points uniformly sampled on the polygon line segments, S_i and S_{i+1}, that form the vertex.

Let (x_i^P, y_i^P) and (x_i^I, y_i^I) represent respectively the corresponding points of the corners of the part and image polygon boundary. Koch and Kashyap determine a transformation, i.e., a rotation θ and translation (u, v), which, when applied to the points (x_i^P, y_i^P), minimizes the distance between the transformed points and the corresponding image points, (x_i^I, y_i^I). The values of θ and (u, v) that satisfy the above criterion are given by

$$\theta = \tan^{-1}\left[\frac{\sum_i x_i^P y_i^I - \sum_i x_i^P \sum_i y_i^I - \sum_i y_i^P x_i^I + \sum_i y_i^P \sum_i x_i^P}{\sum_i x_i^P x_i^I - \sum_i x_i^P \sum_i x_i^I + \sum_i y_i^P y_i^I - \sum_i y_i^P \sum_i y_i^P}\right],$$

and

$$u = \sum_i x_i^I - \cos(\theta)\sum_i x_i^P + \sin(\theta)\sum_i y_i^P$$

$$v = \sum_i y_i^I - \sin(\theta)\sum_i x_i^P - \cos(\theta)\sum_i y_i^P,$$

where i is summed over the number of points sampled in the corners.

In order to combine the different hypotheses Koch and Kashyap use a Hough transform. More precisely, the (θ, u, v) obtained from matching corners are tabulated in a 3-dimensional θ–u–v histogram. The triplets (θ, x, y) that correspond to clusters (modes) in the histogram

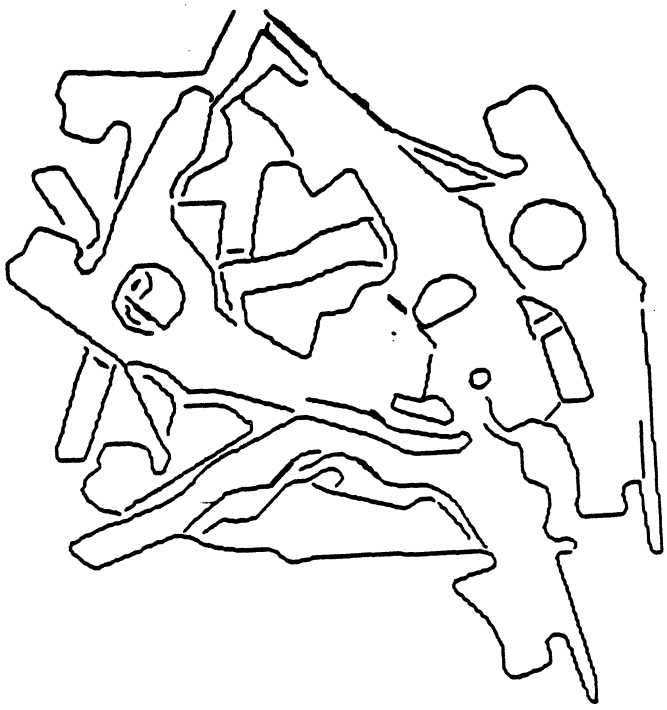

Fig. 3.2. Bin of door lock parts.

are interpreted as the best hypotheses for the part in the image.

The following are some of the insights we have gained from implementing this algorithm.

• The corner detector used by Koch and Kashyap is a poor corner detector. It does not give intuitive corners for parts in the image. In fact, the corners it typically selects do not correspond to those in the part. For example, for the images on which we executed this algorithm, many vertices in the image had approximately the same curvature (see Fig. 3.4). The failure of their corner detector can be traced to a basic flaw in the method in which the corners are determined. In the example shown in Fig. 3.5 the segments that form the corner are long. Therefore dividing the exterior angle by the average of the lengths of these segments, as in [KoK85], results in a low value of curvature even though the curvature at the corner is obviously high. On the other hand, if the segments are short the curvature will appear to large even if the exterior angle β is small.

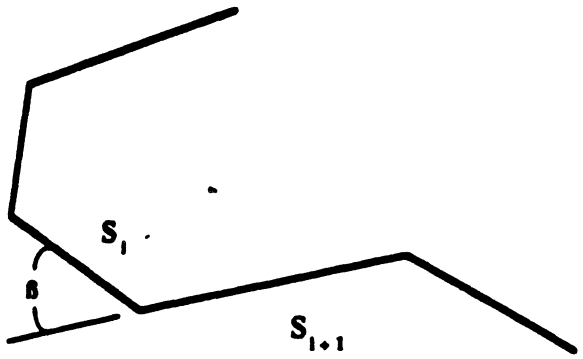

Fig. 3.3. Determining the curvature.

• A second insight is that the Hough does not perform properly when applied to a small set of features. The Hough approach relies on clustering a sufficient number of points to support a hypothesis. Accidental clusters of a few points are typical, therefore unless a cluster has a large number of points, it cannot be interpreted as supporting a correct hypothesis. Typically there are *few* visible corners in a part, and thus clusters in the $(\theta - u - v)$ histogram, even for a correct hypotheses, will contain few points. Clearly, the Hough approach is inappropriate when a few corners are used as features.

• Finally, the Hough transform as discussed in Sec. 2.2.1, because it chooses the most frequent hypothesis as the correct hypothesis, fails to fully use the information available in an image. A Hough strategy does not make use of much of the information available in an image and is therefore a much weaker approach than a relational strategy.

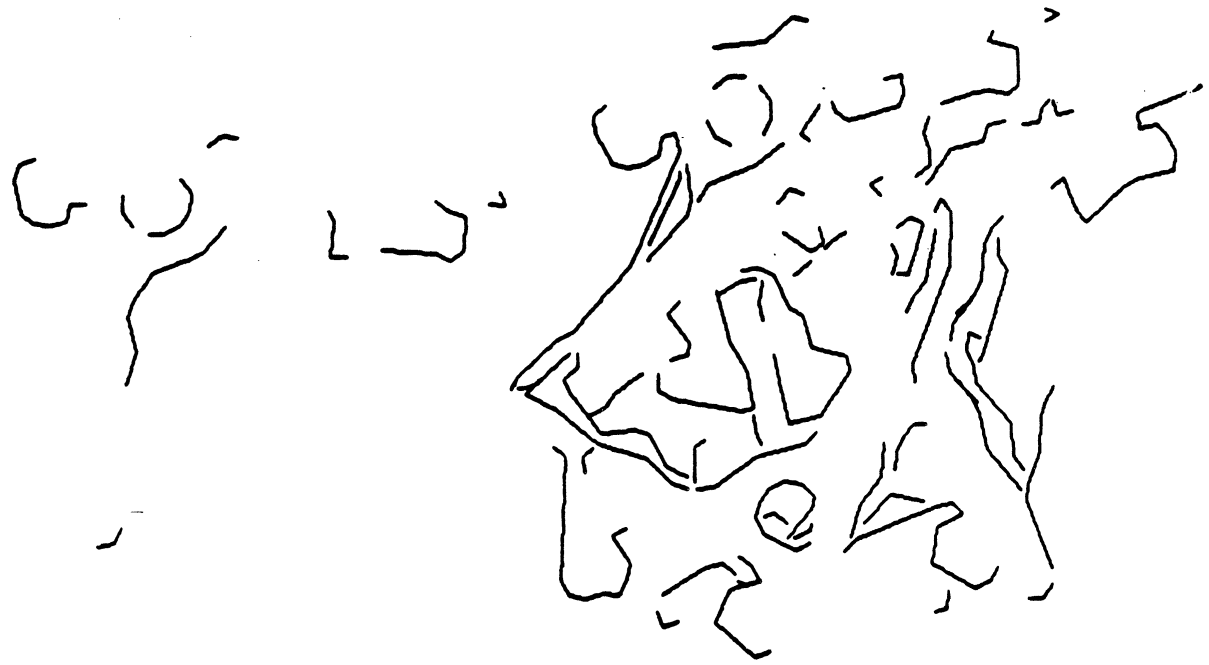

Fig. 3.4. Corners found in the part and image.

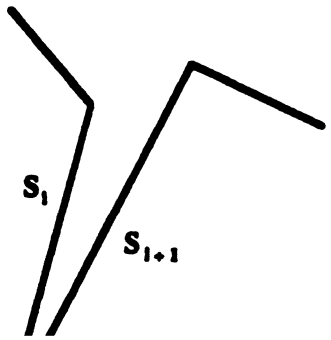

Fig. 3.5. Corner with high curvature.

Figures 3.6-7 shows the results of applying Koch and Kashyap's algorithm to the standard POP image. Boundaries taken from both sides of the part were correlated to the image. The results for the obverse side of the part are shown cross hatched in Fig. 3.6 and those for the reverse side of the part are shown cross hatched in Fig. 3.7. The cross hatched parts have been given poses that correspond to the (θ, u, v) at which the largest clusters in the $\theta-u-v$ histogram occurred, containing 7 or 8 points each. As evident in the image, the algorithm of Koch and Kashyap fails completely for this particular image.

Further experiments were run a test set of fourteen 256×256 pixel wide images of a bin of seven door lock parts. The algorithm was successful in locating the obverse side of the part 2% of the time and successful in locating the reverse side of the part 4% of the time.

3.3. Structural Strategy

The algorithm of Faugeras and Ayache is known as "HYPER." HYPER is typical of the structural strategies and yields good results for many POP images (see results presented in [AyF84]).

HYPER locates parts in the following way. The boundary of the part to be located is approximated by a polygon which we will refer to as a "template." A template consists of a list of line segments of the polygon. From this list HYPER chooses the ten longest segments as "preferred" segments. Starting with the longest preferred segment, HYPER searches the image boundary, also approximated by a polygon, for a corresponding segment that has approximately the same length as the preferred segment and the same relative angle to a adjacent image segment as the preferred segment has to it adjacent segment in the template. We will call the location of corresponding image segments *image sites*.

HYPER transforms the template so that the preferred segment, T_0, is aligned with an image segment, I_0 at an image site (see Fig. 3.8). The rotation in this transformation is taken as the angular difference, θ_0, between the two segments, T_0 and I_0. In addition, the translation (u_0, v_0) that aligns T_0 and I_0 is determined by rotating T_0 by θ_0 and taken the difference in position of the centers of segments. Let (x_{T_0}, y_{T_0}) and (x_{I_0}, y_{I_0}) represent the centers of segment T_0 and I_0 respectively. The translation (u_0, v_0) is computed from

$$u_0 = x_{I_0} - x_{T_0} cos(\theta_0) + y_{T_0} sin(\theta_0)$$

and

$$v_0 = y_{I_0} - x_{T_0} sin(\theta_0) - y_{T_0} cos(\theta_0)$$

After finding an initial transformation of the template into the image, HYPER attempts to iteratively fit the remaining segments of the transformed template with corresponding image segments. Segments are fit in the order indicated by the subscript of T_j shown in Fig. 3.8. During every iteration i, HYPER tries to locate a segment of the image boundary that has approximately the same length and orientation as a segment of the transformed template. If it succeeds,

HYPER fits the coordinates of the the midpoint of the template segment to the coordinates of the corresponding image segment to update its estimate of the template transformation. This update is performed with a Kalman filter. The equations for the Kalman filter are the following. Let $V^{(i)}$ represent the 4-dimensional vector

$$\begin{bmatrix} \cos(\theta^{(i)}) \\ \sin(\theta^{(i)}) \\ u^{(i)} \\ v^{(i)} \end{bmatrix}.$$

where $(\theta^{(i)}, u^{(i)}, v^{(i)})$ represents the estimated orientation and centroid location of the part after the i^{th} iteration. Let $C^{(i)}$ represent the 2×4 matrix

$$\begin{bmatrix} x_{T_j}^{(i)} & -y_{T_j}^{(i)} & 1 & 0 \\ y_{T_j}^{(i)} & x_{T_j}^{(i)} & 0 & 1 \end{bmatrix}$$

which contains the centroids $(x_{T_j}^{(i)}, y_{T_j}^{(i)})$ of template segment T_j. Let $Y^{(i)}$ represent the 2-dimensional vector, $(x_{T_j}^{(i)}, y_{T_j}^{(i)})$, of the centroid of the image segment that is found to correspond to the T_j during the i^{th} iteration. Then the matrix equations for the Kalman filter are defined recursively by

$$V^{(i)} = V^{(i-1)} + K^{(i)} \times (Y^{(i)} - C^{(i)} \times V^{(i-1)}),$$

$$K^{(i)} = S^{(i-1)} \times (C^{(i)})^T \times (W^{(i)} + C^{(i)} \times S^{(i-1)} \times (C^{(i)})^T)^{-1}$$

$$S^{(i)} = (I - K^{(i)} \times C^{(i)}) \times S^{(i-1)},$$

where I is an identity matrix. In addition, $S^{(i)}$ is a covariance matrix, the initial value of which is essentially guessed. $W^{(i)}$ is a noise variance, again a quantity that is guessed. $K^{(i)}$ is an intermediate matrix.

The results obtained from the Kalman filter, at least in this application, are almost exactly the same as those obtained from an iterative weighted least squares fit (see Nahi [Nah69]). There is really no benefit in using the Kalman filter over a more conventional fitting algorithm.

After each iterative update by the Kalman filter, a measure of the quality of the fit, is computed. The quality is simply the sum of the lengths of the image boundary segments that

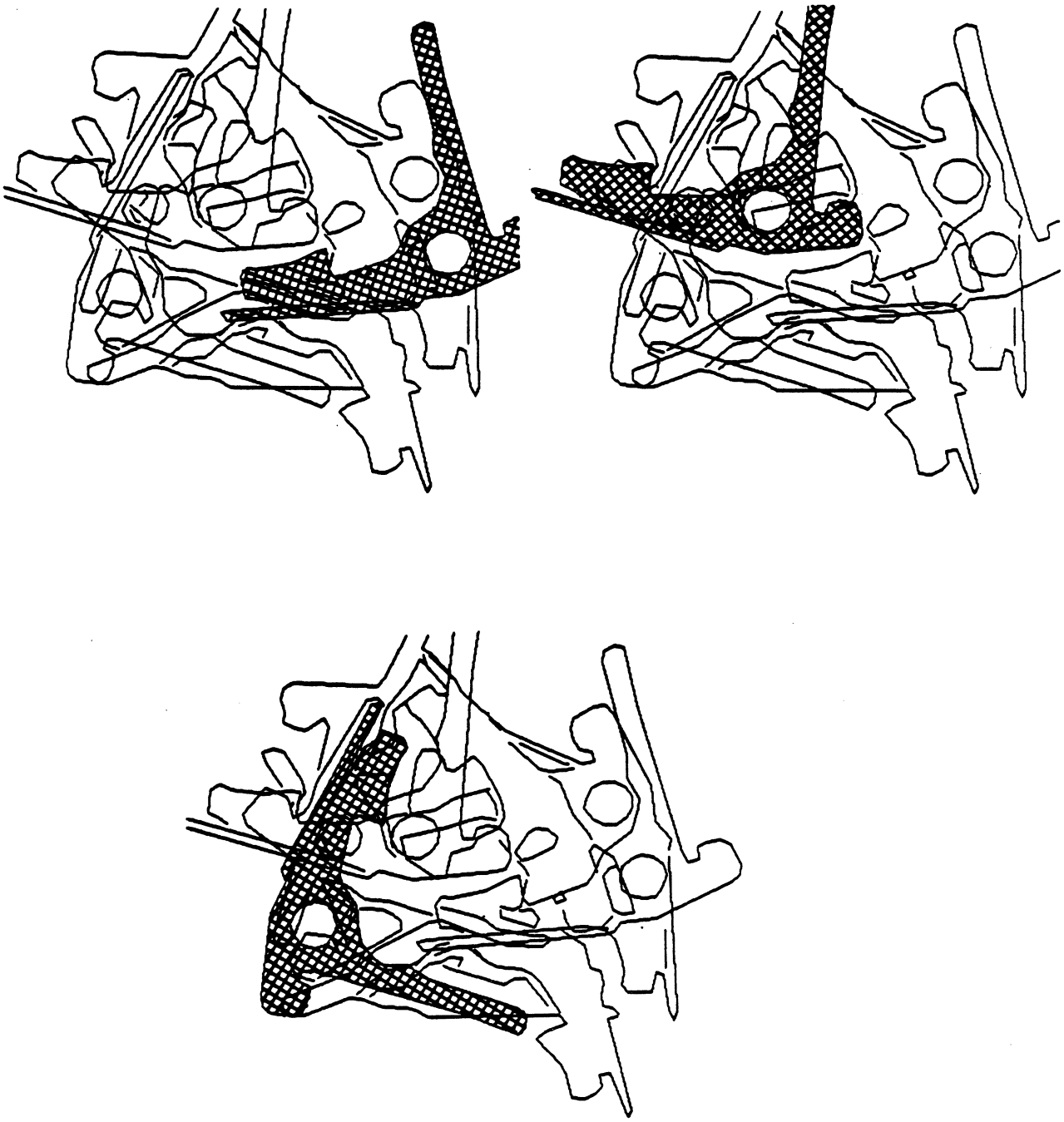

Fig. 3.6. Results of Koch and Kashyap's algorithm for POP test image.

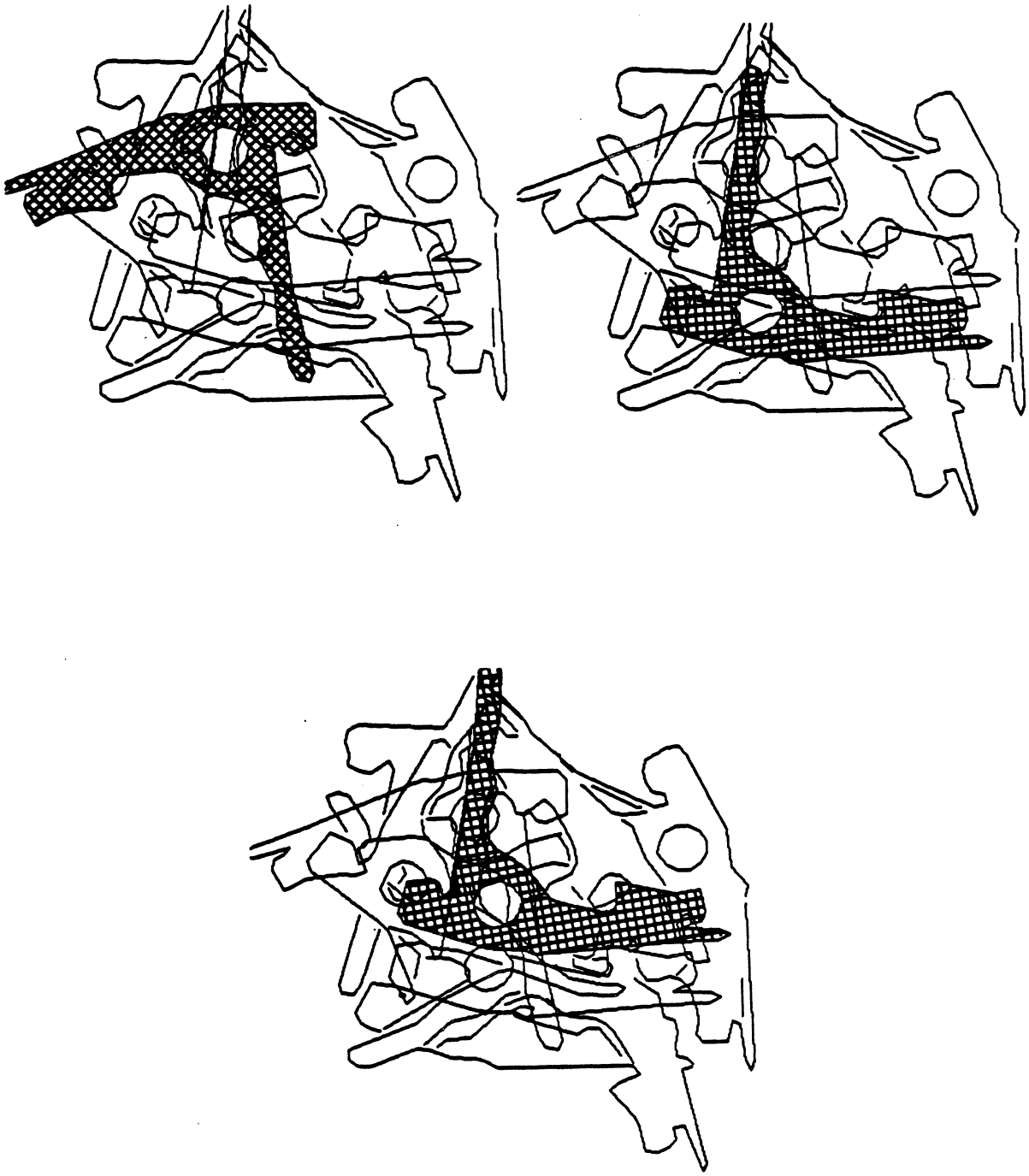

Fig. 3.7. Further results of Koch and Kashyap's algorithm for POP test image.

correspond to template segments in the fit. This sum is called the "Q" of the fit.

The process of fitting the template to the boundary is repeated at every location in the image boundary at which a segment is found that corresponds to one of the preferred segments. The fit with the highest quality measure, Q_{max}, is assumed to be the correct location of an instance of the part in the image.

However, because there are typically several hundred image sites in the image boundary, fitting the complete template at each site would be extremely inefficient. Instead, HYPER saves a running value of the highest quality measure and uses it to bound latter fits. At each new image site segments of the image are fit to the template. After each segment is fit a quantity called the maximum achievable Q is determined. The maximum achievable Q represents the largest Q that can be expected from a fit, given the segments that have already been matched to the template. It is estimated by summing the current Q of the fit plus the length of the template segments that have yet to be fit, on the assumption that these latter segments could possibly fit perfectly. If the estimate of the maximum achievable Q of the fit falls below Q_{max}, the fit is abandoned.

This strategy is very much like that of Rummel and Beutel [RuB84] except that they also include corners and circles as matching primatives.

The following are some of the insights we have gained from studying HYPER.

- Many of the unique features of the boundary, i.e., features that allow easy recognition of the part, are distorted in HYPER. For example, features around high curvature portions of the boundary are not well approximated by polygons.

- The use of Kalman filtering is essentially the same as a recursive least squares fit of the template segments to the image segments. There are considerably simpler ways to perform this fit.

- HYPER chooses the long line segments of the part boundary because these segments are generally the least common segments in a template. Unfortunately, they are also the most likely to be occluded in a POP image.

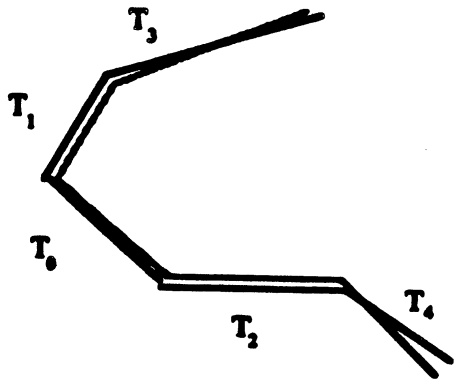

Fig. 3.8. Iteratively fitting the template.

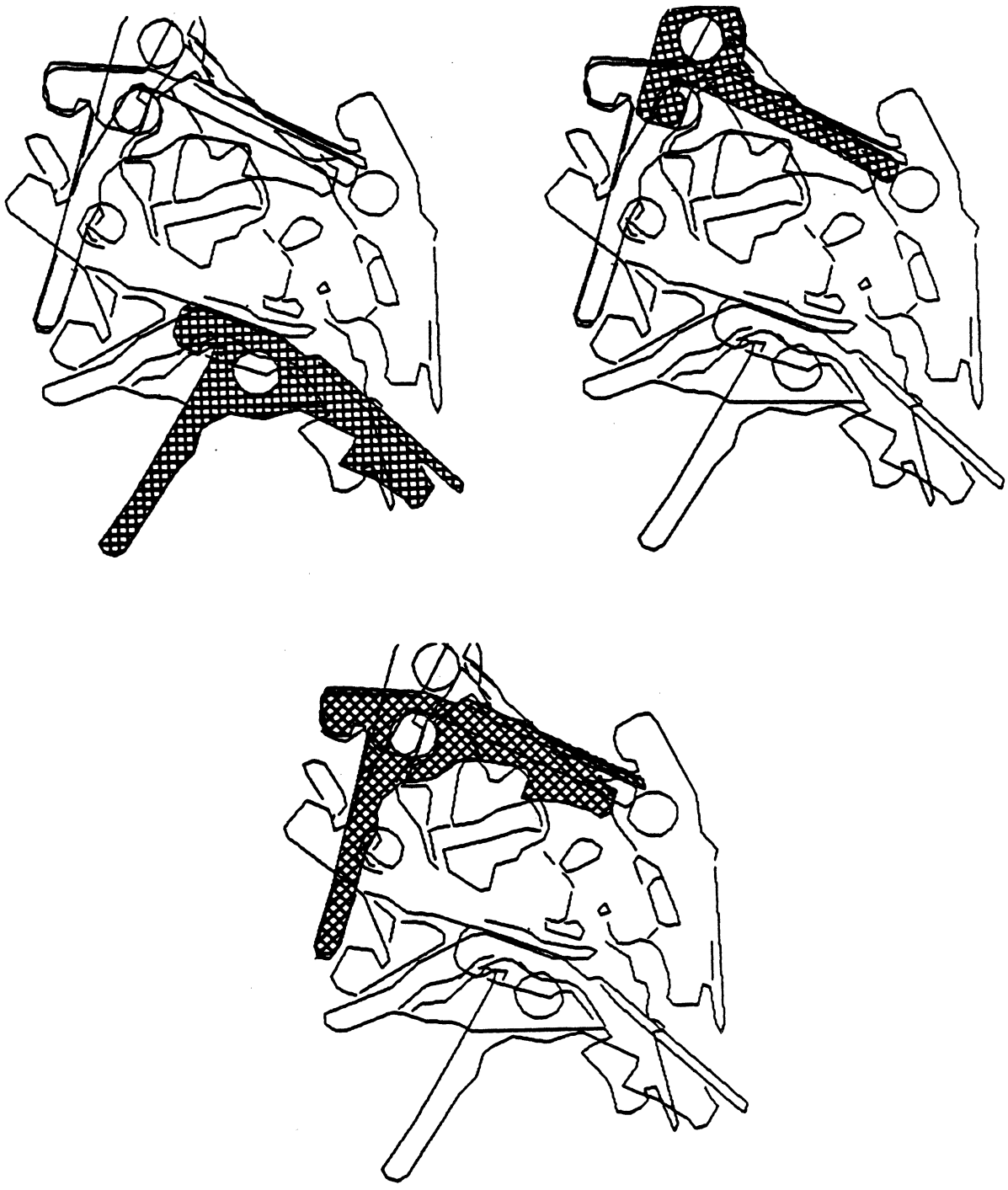

Fig. 3.9. Results of HYPER for POP test image.

• One shortcoming of the approach is the strategy that it uses to determine the measure of the quality of the fit, Q. This tends to give more weight to unimportant features of parts. HYPER rates a fit by the length of the image boundary that corresponds to the template. This does not discriminate between a fit to unique features of a template and to commonly occurring features of the template. For example, HYPER would report a better fit if a long line segment of the template matched the wrong part in the image, than it would if a shorter but unique set of segments matched the correct part.

Figures 3.9-10 shows the results of applying HYPER to a standard POP image. Both sides of the part were matched to the image using HYPER. HYPER finds one instance of the part in Fig. 3.9 but fails to find any in Fig. 3.10. The three templates shown cross-hatched in the image are the template fits with the three highest Q's. These poses are all incorrect.

In further test on the fourteen images mentioned above, HYPER found the obverse side of the part correctly 14% of the time, and found the reverse side of the part 18% of the time. It was in error in the remaining attempts.

3.4. Relational Strategy

As a example of a relational strategy, we chose the algorithm of Bolles and Cain [BoC84] known as "Local Feature Focus" or LFF for short. Of the relational strategies, this strategy appears to be the most effective.

LFF. The LFF algorithm recognizes a part in an image from features and relations between features of the part. In the approach suggested by Bolles and Cain features are regions and corners. More precisely, regions are connected areas of the part that share some common property such as a single grey level value. Corners, on the other hand, are locations on the boundary that can be fit by a pair of line segments. As an example of these two types of features, Fig. 3.11 shows a region (cross-hatched) and two neighboring corners (highlighted) for door lock part no. 1.

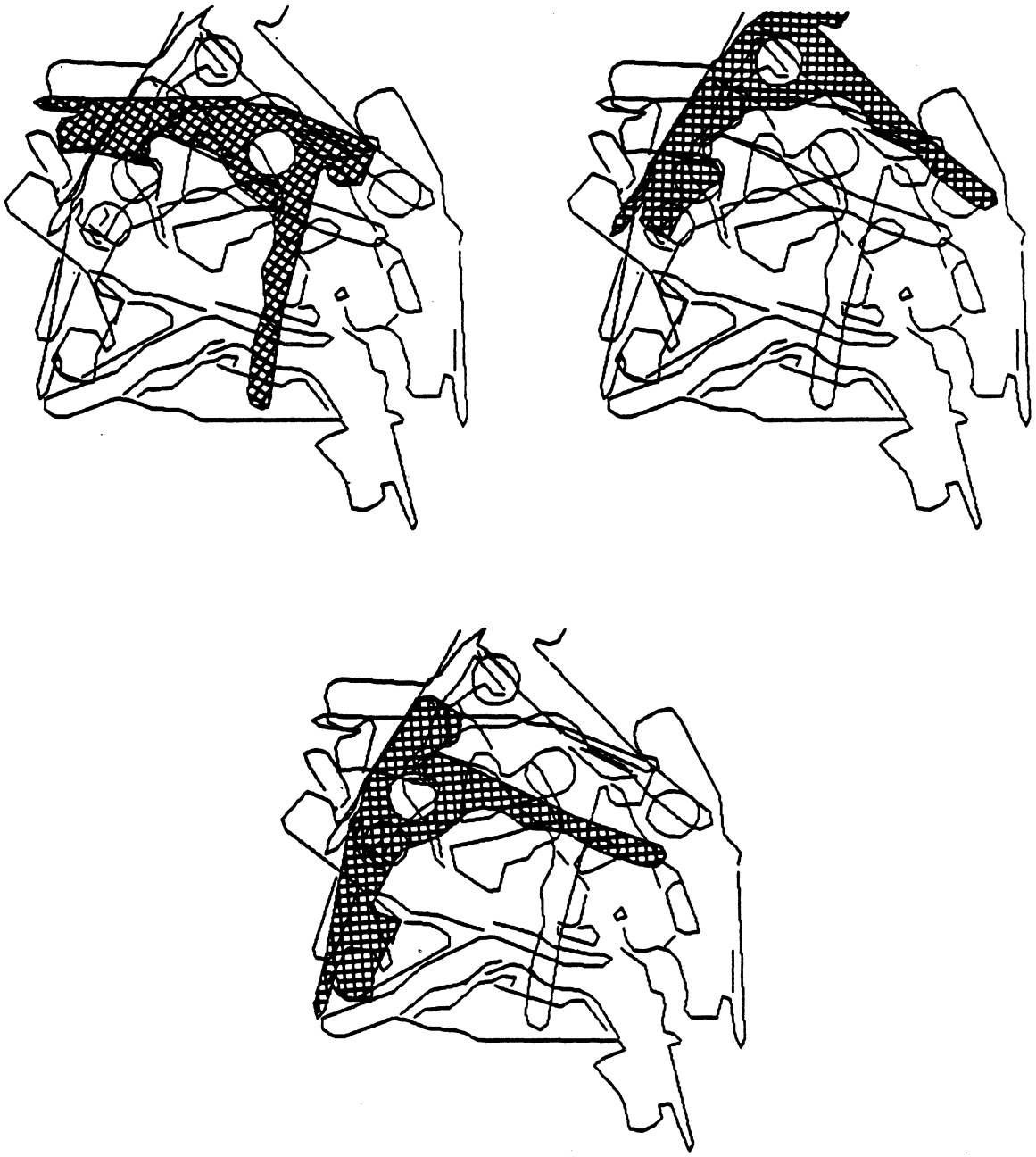

Fig. 3.10. Further results of HYPER for POP test image.

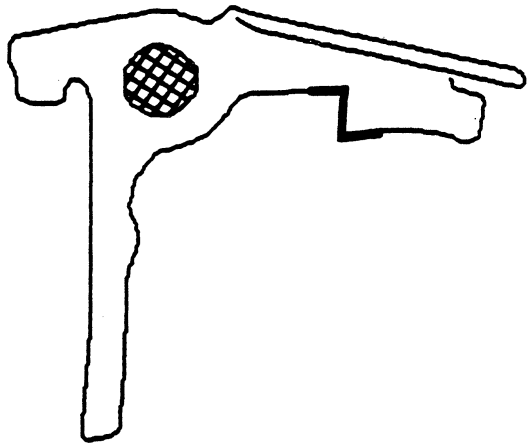

Fig. 3.11. Features of door lock part no. 1 used by LFF algorithm.

From the set of features, the LFF algorithm selects a subset of preferred features called "focus" features. A focus feature has the property that, together with a list of nearby features called "co-features," it identifies a unique pose of the part. In other words, if a focus feature of a part is found and enough of its neighboring "co-features" are found, the pose part can be uniquely identified in the image.

Each co-feature has a set of characteristics. These characteristics include type, distance from the focus feature and orientation with respect to the focus feature. If any instance of a focus feature is located in an image, the LFF algorithm searches for occurrences of co-features. If co-features are found, the algorithm builds a list of possible assignments of features of the part to those of the image. From this list a compatibility graph is constructed. The largest maximal clique of this graph is assumed to represent a correct set of feature assignments.

Focus features are selected during a training stage. Given a set of object models, the LFF algorithm selects a random feature as a potential focus feature. It then selects a set of nearby features for each focus feature and then ranks the focus features according to the predicted costs of using them to identify an object.

The following are some of the insights we have gained from studying the LFF algorithm.

- In the LFF algorithm, features are chosen to be easily identified primitives such as regions and corners. These sets of special features will not, with great likelihood, be found undistorted in sufficient numbers to be useful in locating a part in a POP image. For example, regions are prone to occlusion. In addition, corners built from pairs of straight line segments are not well defined unless they include a considerable amount of the perimeter. The approach can be made more robust by selecting fragments of these features, i.e, segments of the perimeters of regions or segments around corner. This approach adds some redundancy to the feature set.

- Recognition is accomplished by locating the largest maximal clique in a compatibility graph. This problem is NP complete. For this approach to be reasonably efficient, only a few features may be selected as focus features, but this reduces recognition capability.

Figure 3.12 shows the results of applying Bolles and Cain's algorithm to the test POP image of Fig. 3.2. The hole in the part served as the focus features and the co-features were the corners. As is apparent from the image, only one hole was undistorted and therefore only one part was found. From the test set of 96 images mentioned above, 14 parts (14.5%) of the parts were recognized using Bolles and Cain's algorithm with the focus feature shown in Fig. 3.11. All parts had the hole exposed.

An improvement on this algorithm would be to use critical points as the features. In this case, the features are again easily found. Unfortunately, most images contain a large number of critical points. This along with the fact the recognition algorithm is NP-complete makes the approach impractical for even a modest number of features.

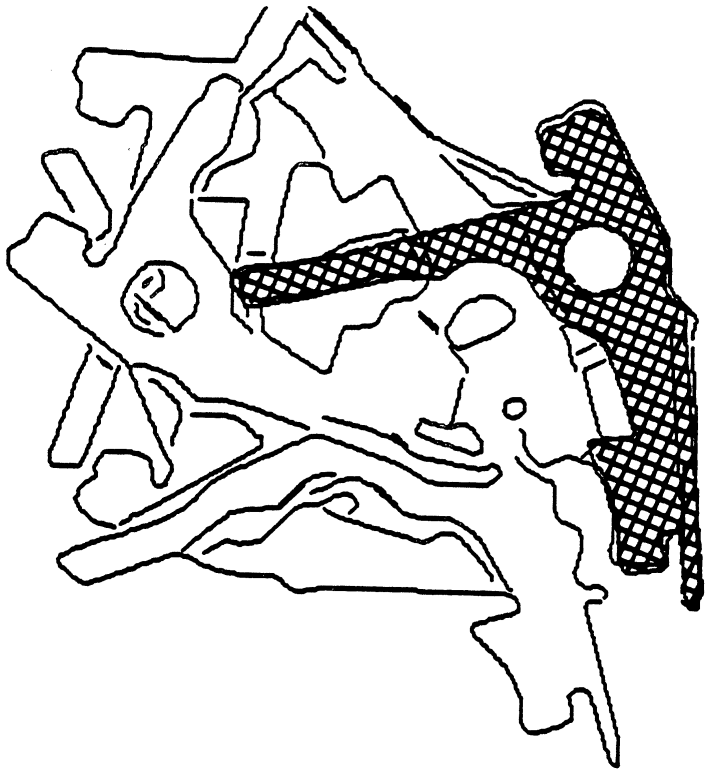

Fig. 3.12. Results of LFF for POP test image.

CHAPTER IV

RECOGNIZING PARTIALLY OCCLUDED PARTS

4.1. Introduction

In this chapter we will present our method for solving the POP recognition problem. We start by presenting arguments for constructing part-models from pairs of fixed length overlapping boundary segments. These pairs of segments, which we term configurations, form the primitive features of our part-models. An efficient method is presented for determining matches between configuration-segments from the part-models and configuration-segments in the image. It makes use of a dual representation for segments: a cartesian representation and a θ–s representation similar to that discussed in Chapter 2. To further improve the recognition procedure we associate a number in the half-open interval (0, 1] with each configuration that indicates the degree to which it distinguishes the part containing the configuration from other parts in the set of parts. This number is the saliency of a configuration. The concept of saliency is formalized. Next, a training method for the automatic generation of salient configurations is presented. The configurations above a certain saliency threshold are used in the part-models. Finally, a strategy is presented for solving the POP problem that uses the part-models.

4.2. Part-models

A model-based POP recognition algorithm consists of two components. First, a set of *models* containing geometrical information about features of the parts to be recognized. These models are embodied in the data structures of the algorithm. Second, a *strategy* that uses the models to

recognize the parts. This strategy is embodied in the code of the algorithm. The degree to which a model-based algorithm solves the POP problem depends largely on how much information the models represent about the parts and their features. No matter how ingenious the algorithm is, it cannot create information and, therefore, the performance of any algorithm is bounded by the amount of information in the models. Thus, we can set some limits on the potential performance of algorithms for the POP problem by simply examining the information that the algorithms use in their part-models. With this in mind, the types of features we have chosen to embody in our models can better be appreciated if we first briefly review some characteristic models used in the algorithms surveyed in Chapter 2.

Part-models based on *edge points* are proposed by Merlin and Farber [MeF75] and Ballard [Bal81]. Edge points are a good representation in the presence of occlusion, because, short of total occlusion, some edge points of a part-model will always appear in the image. Unfortunately, as a representation, edge points have the disadvantage that they are relatively indistinguishable from one another: edge points can belong to any boundary of any part in the image. (Ballard uses the slope angle of an edge point to distinguish it, but this is very little added information). For this reason, as we saw in Chapter 2 (see Sec. 2.2.1), the generalized Hough transforms of Merlin and Farber, and Ballard often find incorrect locations for parts. The edge points of the part may correlate better with edge points that belong to other parts, or to the wrong edge points on the same part, than to the correct set of edge points.

Part-models based on edge points with associated *probabilistic labeling vectors and relations between edge points*, are used by Rutkowski [Rut82]. Rutkowski associates a probabilistic labeling vector $\mathbf{p}^i$ with each edge point i of the image. Element p_j^i represents the probability that image point i can be labeled as edge point j of the part. Rutkowski uses the labeling vectors and spatial relations between edge points as input to a relaxation algorithm. Each edge point in the image boundary is assigned a vector of labels, one label corresponding to edge point of the part boundary. For n edge points in the part boundary and m edge points in the image boundary there are $n \times m$ labels. Each of these labels updated in an iterative fashion, using a set of of

heuristic relations, until a final label is assigned to each edge point (see Sec. 2.4.3). The part-models are unnecessarily complex and result in undesirably long recognition times.

Part-models based on *grey level regions* are used by Kelly et al. [KMB83]. A grey level region is no more than a contiguous set of gray valued pixels. By correlating certain regions with an image, one can estimate the location and identity of parts in the image (see Sec. 2.2.3). If we adopt the approach in [KMB83], region correlation requires $O(n^2m^5)$ operations, when images have $n \times n$ pixels, regions have $m \times m$ pixels, and m^3 different views of each region are used. Region correlation is sensitive to occlusions because it is unreliable when based on small regions. In a POP image the visible regions of a part will generally be small, and, therefore, regions are inadequate features for the POP problem.

Part-models based on *axial representations*, such as the symmetric axis transform (SAT) of Blum and Nagel [BlN78] and the smoothed local symmetries (SLS) of Brady and Asada [BrA84], are simple models to compare. However, they contain much less of the structural information present in a part's boundary than the original boundary. SAT's derived from occluded parts are often completely distorted, because SAT are sensitive to occlusion of the the boundaries. SLS's are much less sensative to occlusion. However, SLS's derived from occluded parts, because they contain much less structural information than the original boundary from which they are derived often have ambiguous interpretations (see Sec. 2.2.2).

Part-models based on limited sets of *special features*, such as corners and holes, are used by Bolles and Cain [BoC84], Tropf [Tro80], Koch and Kashyap [KoK85], and Stockman [SKB82]. In contrast, our algorithm selects a set of features that are special in the sense that they distinguish the part. The type of special features in the above references have the advantage that they are easy to locate and are thus useful for quick recognition. However, these features occur infrequently on the contours of typical parts, making them vulnerable to occlusion. In general, algorithms that rely on a limited set of features, such as holes and corners, will often not find enough of the features in the image for reliable recognition. In addition, they tend to be problem specific. Without an automatic way to select special interesting features from a set of parts, it is necessary

to redesign an algorithm for each problem domain.

Part-models based on *line segments* that form the sides of a polygon approximation of the boundary of a part are used by Ayache and Faugeras [AyF84] (see Sec. 2.3.2). Line segments are more frequent in typical images than special features, but are distinguishable from each other only by their length. When recognizing a part, Ayache and Faugeras compare only the longer segments from the part-model to the segments in the image. This dramatically reduces the number of comparisons from the number required if they had compared line segments of arbitrary length. Unfortunately, in POP images the longer line segments are more likely to be occluded, and, thus, are generally not a good choice of features.

Our brief survey of part-models suggests that probabilistic labeling vectors, regions, axial representations, special features, and line segments, are not particularly suitable features for use

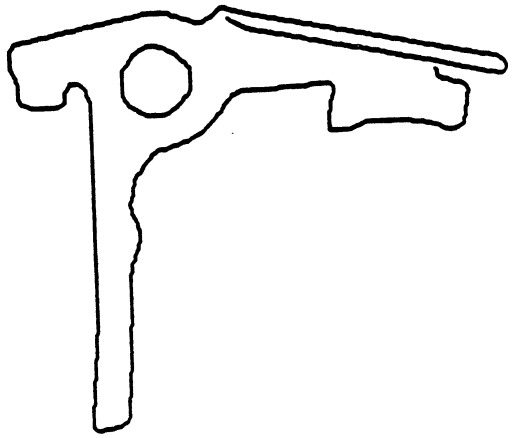

Fig. 4.1. Door lock part no. 1.

in solving the POP problem. By comparison, edge points appear better suited to the problem. Their main shortcoming is that they have no local structure to differentiate one edge point from another. This can be overcome by using sequences of linked edge points, or *segments*. Moreover, to maximize the number of visible segments in the presence of occlusion, segments may be overlapped. Therefore, we have chosen fixed length overlapping segments of the boundary as components in the features for our part-models. The fixed length is a design parameter and depends on the set of known parts. If the length is too short, the segments will be indistinguishable. On the other hand, if the length is too long, the segments are less likely to be visible, although longer segments are implicitly compared when shorter fixed length segments that cover the longer segments are compared. The fixed segment length should be related to the curvature of the parts: if there is frequent occurrences of high curvature, segments should be short.

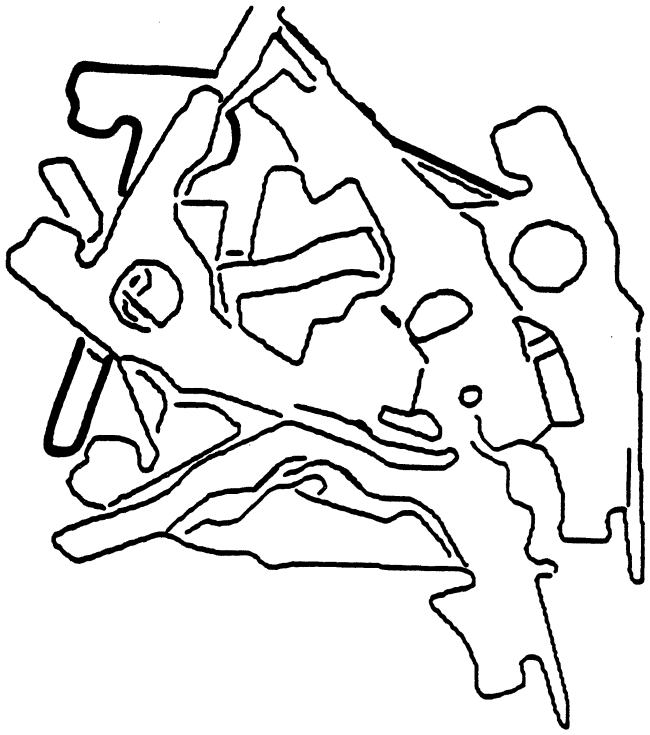

Fig. 4.2. POP image and visible segments of door lock part.

The set of overlapping segments are obtained from all the boundaries from all of the part's stable positions. Since the part is 2-dimensional, we make the assumption that it will always appear in quasi-stable position, that is to say, tilted little from one of its true stable positions, even when it is in a pile with other parts. Figure 4.1 shows a part, and Fig. 4.2 shows a POP image with the visible segments of one instance of the part highlighted. Finally, to minimize the possibility that a segment, resulting from the random alignment of two or more contours from different parts in an image, can match one from a part-model, we have chosen to use *configurations* of segments as features. Figure 4.3 shows a configuration. It is simply two fixed length segments in a fixed relative position. The relative position of two segments can be defined by the angle between the vectors from the mid-points of the segments to the centroid of the part's boundary from which the segments are taken. The notion of a configuration could be extended to allow an arbitrary number of components or segments. This extension is similar to Bolles and Cain's *local*

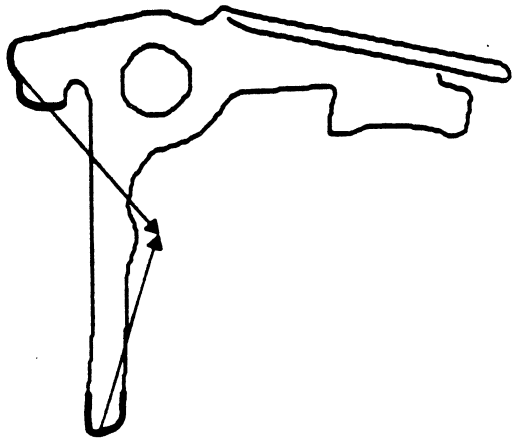

Fig. 4.3. A configuration of segments.

feature focus, which is a configuration of special features [BoC84]. In the following we will not consider configurations with other than two components.

Configurations form the primitive features of our part-models. However, using them indiscriminately for recognition would result in an inefficient algorithm, since each part has a large number of configurations which are nearly indistinguishable. To counter this we have adopted a measure for the distinctiveness of a configuration which we term the *saliency* of the configuration. Informally, the saliency of a configuration is the inverse of the frequency of occurrence of the configuration in the set of known parts. The idea behind this notion is that the more often a configuration is found in the set of parts, the less important the configuration is in distinguishing a part and its pose. An efficient recognition strategy begins by trying to identify the most salient configurations first. The concept of saliency must be modified slightly when

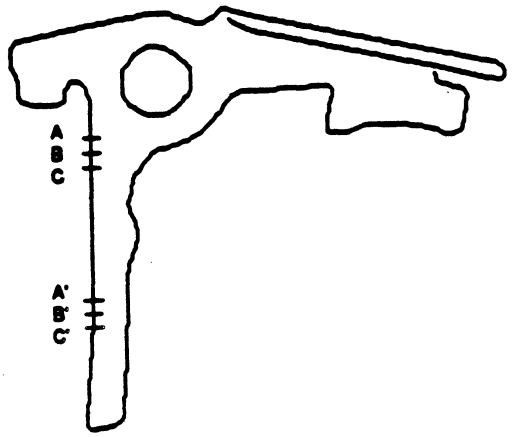

Fig. 4.4. Segmented boundary representation of a part.

noise is taken into account, as we shall see. However, before exploring saliency more fully, we first present an efficient method for determining matches between configuration-segments from the part-models and configuration-segments in the image.

4.3. Segment Matching

A part-model may be viewed as a boundary partitioned into overlapping fixed length segments together with saliency information for configurations (Fig. 4.4 shows overlapped segments AA' , BB' , and CC'). An implementation need not store all the segments; it can simply store the boundary as a linked list of edge points from which segments may be taken during run-time.

4.3.1. Dual Representation

In our part-models we keep a dual representation for the configuration-segments. The first is a straightforward *cartesian* representation and the second is the θ–s representation discussed in Chapter 2 (see Fig. 4.5). The θ–s representation is a parameterization of the slope angle, θ, of the part's boundary by its arclength, s, where arclength is measured from an arbitrary starting point on the boundary. The slope angle can be represented as a function of arclength, $\theta(s)$. The θ–s representation allows us to compare segments of the part-model with segments in the image that are flipped, segments in the image with contrast reversals, and segments in the image that are scaled, in each case more efficiently than in a cartesian representation. The θ–s representation does not, however, preserve 2-dimensional distances between segments. Therefore, to compare configurations of segments of the part-model to the those in the image, it is first necessary to compare segments of the configuration individually in θ–s space and then to check the relative poses between segments in cartesian space. We will assume that two configurations matched if their segments have the same relative pose in cartesian space and if the corresponding segments of the configurations match in θ–s space.

Fig. 4.5. A part and its θ–s representation.

4.3.2. The θ–s Representation

The principal reason for using the θ–s representation is that segments can be compared more efficiently in this representation than in a cartesian representation.

Comparing Configurations. In a cartesian representation, segments are compared by fitting a segment of the part to a segment of the image. Fitting involves three parameters: x and y components of translation, and a relative rotation angle, ϕ. In the θ–s representation the segments can be fit with one parameter, the relative orientation, Θ.

To compare a part-model segment with an image segment, we select the sum of the squares of the differences between corresponding slope angles as a measure of the closeness of the fit. The centers of the segments are aligned and the θ values of the part-model segment, $\theta_M(s_i)$ for $i = [-n, \cdots, n]$, are least squares fit to the corresponding θ values of the image segment, $\theta_I(s_i)$

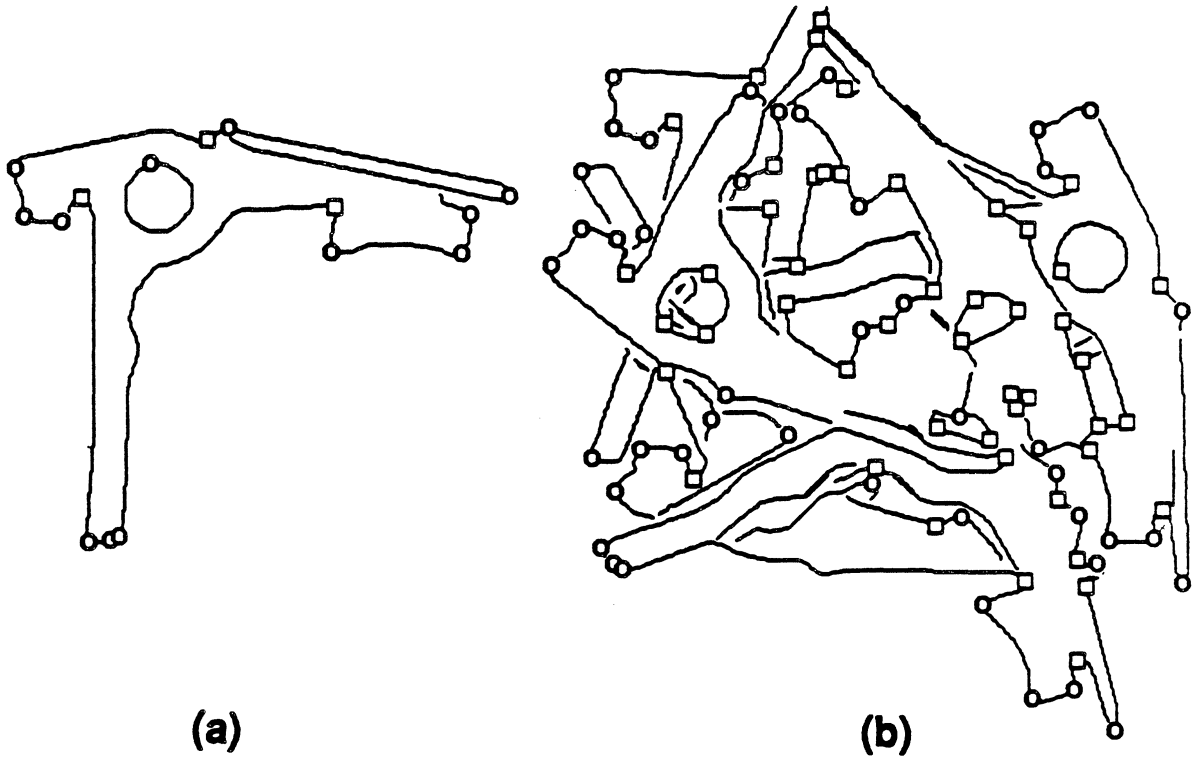

Fig. 4.6. Critical points of part and POP image.

for $i = [-n, \cdots, n]$. We assume that both segments have been sampled at equal arclengths at n points on either side of their centers. The fit parameter, Θ, is chosen to minimize the following

$$\frac{1}{2n+1} \sum_{i=-n}^{n} (\theta_M(s_i) - \theta_I(s_i) - \Theta)^2.$$

The minimum occurs when

$$\Theta = \frac{1}{2n+1} \sum_{i=-n}^{n} (\theta_M(s_i) - \theta_I(s_i)) = \bar{\theta}_M - \bar{\theta}_I,$$

in other words, when Θ is simply the difference between the mean tangent angles of the two segments. The minimum residue

$$R = \left(\frac{1}{2n+1} \sum_{i=-n}^{n} (\theta_M(s_i) - \bar{\theta}_M - (\theta_I(s_i) - \bar{\theta}_I))^2 \right)^{\frac{1}{2}}, \tag{4.1}$$

can be used as a measure of the similarity of the segments, and is used by us to decide whether the segments match. Equation (4.1) is all that must calculated to compare segments once θ and s are determined. In practice we assume that two segments match if R is less than a fixed threshold, D. The value of D is chosen to reflect the noise anticipated in the images under consideration. We assume that two configurations match when

1) the relative poses of the segments of the configurations in cartesian space are equal and,

2) the segments of one configuration individually match corresponding segments of the second configurations in θ–s space, i.e., are within a tolerance D of each other.

Critical points. Critical points in a boundary, if they exist, can be used to further improve the efficiency of comparison. We define critical points as the maxima and minima of the curvature of the boundary that have curvatures above a fixed threshold. These are the larger valued maxima and minima. The curvature of a contour is the first derivative of $\theta(s)$ with respect to s, or $\frac{d\theta(s)}{ds}$ (see [Lip69]). If a segment of the part-model contains critical points, as is often the case, it need only be compared to segments in the image that contain similar critical points. By comparing a part-model segment to only those image segments with similar critical points, we substantially reduce the number of comparisons needed to locate matches. The location of critical points in the contour is readily obtained by applying a 1-dimensional edge detector to the function $\theta(s)$.

Figure 4.6a shows the critical points of a part; curvature maxima are shown as circles and curvature minima are shown as squares. Figure 4.6b shows the critical points of the boundary in a POP image. Note the correspondence of critical points.

In addition to a providing a simple method for comparing contour segments, the θ–s representation can also be used to compares segments of flipped parts. This provides a savings when the flip-side of a part is a mirror image of the obverse side. This often happens with 2-dimensional parts.

Flipped parts. Figure 4.7a shows the flip-side of the part shown earlier in Fig. 4.5. The tangent angle θ at each edge point on the boundary of the flipped part in Fig. 4.7 is the negative of the tangent angle at a corresponding point on the boundary of the original part, and the contour of the flipped part is traced in the opposite direction to the contour of the original part. Thus, the θ–s representations of the two views are related by,

$$\theta_{flipped}(s) = -\theta_{original}(-(s - s_0)) + \theta_0 , \tag{4.2}$$

where s_0 represents a possible offset if the two contours have different starting points, and, θ_0 represents a possible rotational offset if the two contours have different orientations after the flip. Thus, to search for the segment of a flipped part, the corresponding segment of the part-model is reverse in direction and its θ values are negated before the segment is compared to the image boundary. Compare the θ–s representation of the part in Fig. 4.5b to that of its flip-side in Fig.

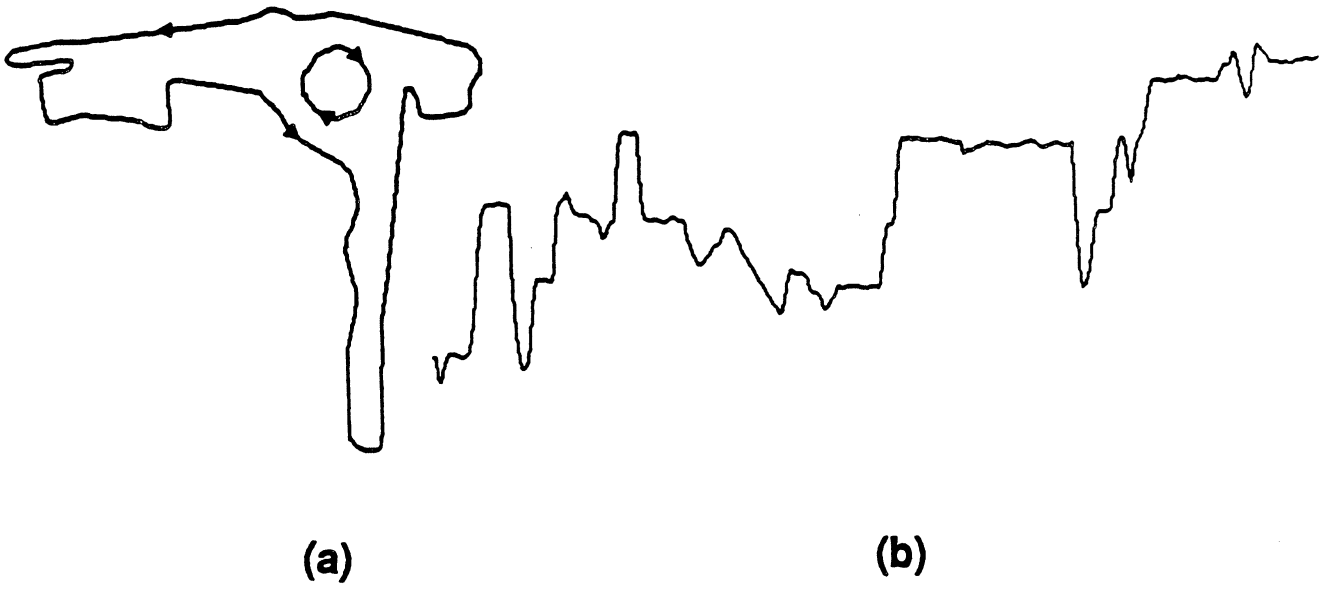

Fig. 4.7. Flipped part and θ–s representation.

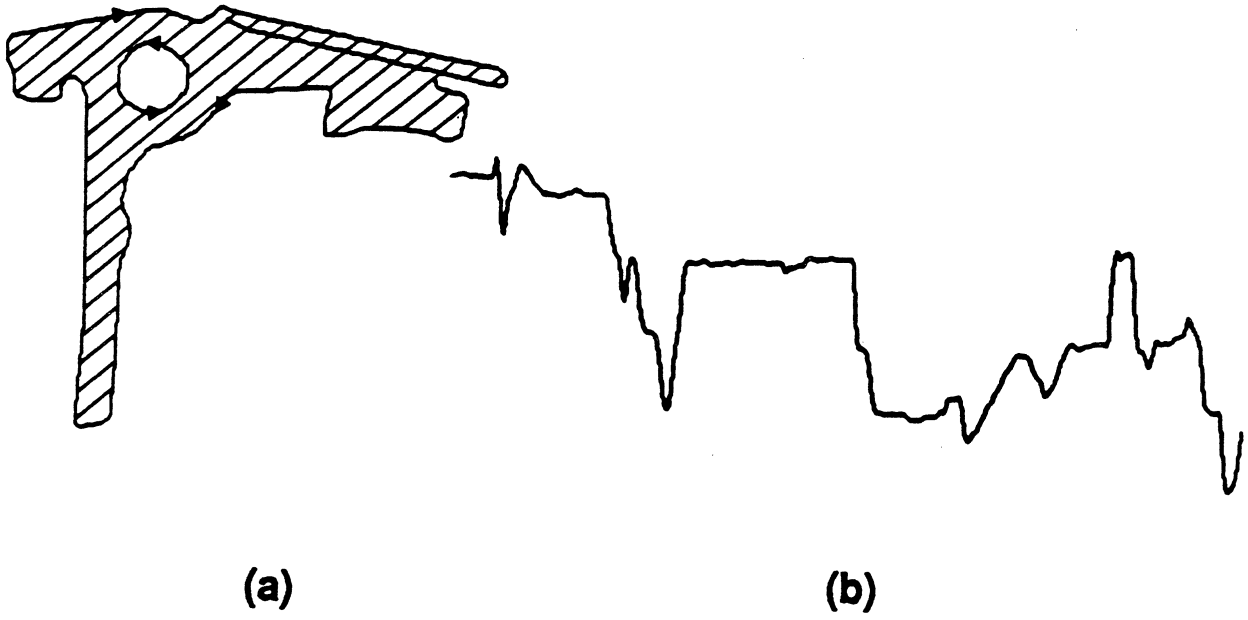

Fig. 4.8. Contrast reversal of a part and θ–s representation.

4.7b.

The θ–s representation can also be used to compare segments of opposite contrast. This is important when parts are reflective, as we noted in Chapter 1.

Contrast reversals. Reflective parts can have contrast reversals. By convention the direction of the boundary is traced by keeping the higher contrast pixels on the left. Therefore, contrast reversals can cause direction reversals in segments of the boundary. This information cannot be easily stored in a model of the part because it depends on the lighting. However, it can still be accommodated by our part-models if segment matching is performed in both directions. The segments are compared in the direction consistent with the part-model's contrast and then in the reverse direction to search for areas of the part that are reversed in contrast from the model. This allows segments with contrast reversals to be found, and, as a consequence, reflective parts

can be recognized in an image. Figure 4.8a shows the contrast reversal of a part. The θ–s representation of the part in Fig. 4.5 is related to the θ–s representation of the part in Fig. 4.8 by,

$$\theta_{contrast\ reversed}(s) = \theta_{original}(-(s - s_0)) + \theta_0 , \tag{4.3}$$

where s_0 represents a possible offset if the two contours have different starting points, and θ_0 represents a possible rotational offset if the two contours have different orientations. Combining (4.2) and (4.3), we see that the θ–s representation of the contrast reversal of a flipped segment is related to the θ–s representation of the original segment by,

$$\theta_{flipped,\ contrast\ reversed}(s) = -\theta_{original}(s - s_0) + \theta_0 .$$

Thus, contrast reversals of segments in both flipped and unflipped views can be compared to the segments in the part-model by appropriately changing the direction of trace and/or negating the values of θ obtained.

Finally, the θ–s representation of a part can also be used to locate scaled partially occluded parts.

Scaled parts. We consider only parts that appear scaled down, or shrunk, from the model's scale. Magnification can be handled in an analogous fashion. Shrinking a part decreases the arclength between points on the contour of its boundary but does not change the slope angle of the points. Its θ–s curves are compressed uniformly along the s axis, but the slope angle structure remains the same. Figure 4.9a shows an example of a boundary that was extracted from an image of a half-scale version of the part in Fig. 4.5. Notice that the boundary is scaled down in the s direction by a factor of two, but maintains much of the same structure as the original θ–s representation. The θ–s representation of the two parts are related by

$$\theta_{scaled}(s) = \theta_{original}\left(\frac{s - s_0}{K}\right) + \theta_0 ,$$

where K is the scaling factor. Again, s_0 represents a possible offset if the two contours have different starting points, and θ_0 represents a possible rotational offset if the two contours have

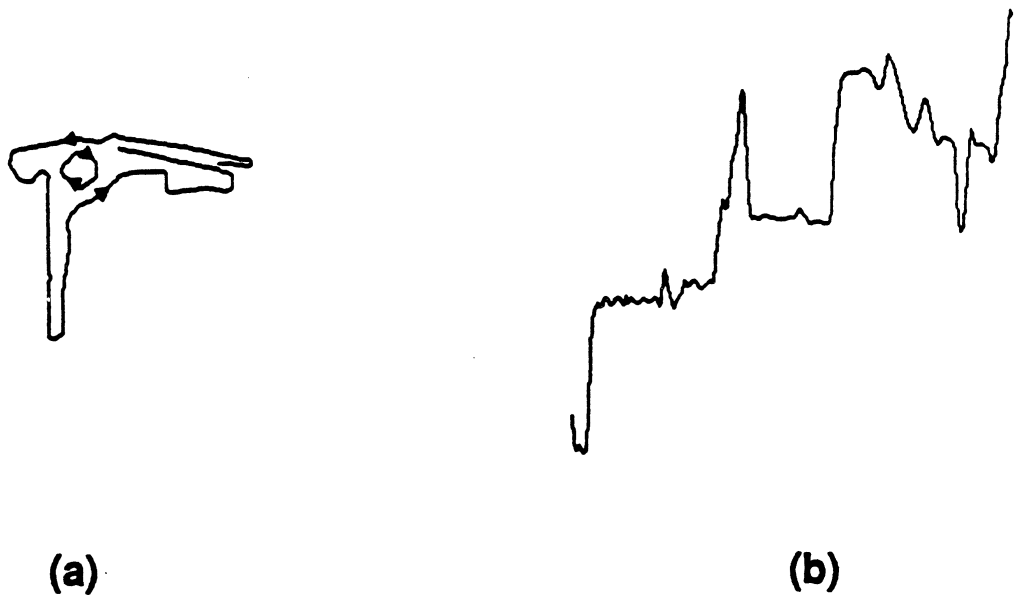

Fig. 4.9. Scaled boundary and θ–s representation.

different orientations.

To compare a part-model contour to the image contour, initially one critical point, C_1 (see Fig. 4.10), in the part-model segment is aligned along the s axis with a critical point, C_1' , on the image contour. A second critical point on the part-model contour, C_2, is aligned with a second critical point on the image boundary, C_2' . This is done by contracting the θ–s representation of the part-model until the two sets of critical points are aligned. In practice this is accomplished by uniformly resampling the θ–s representation of the part-model so that it has the same number of samples between the first and second of its critical points as the image boundary has between the first and second of its critical points. The θ–s representation of the image boundary and of the contracted part-model boundary are compared using the sum of the squares of the differences as a measure of fit. If the two representation match, a scaled version of the part is assumed to have been found (see Sec. 6.2.3 for results).

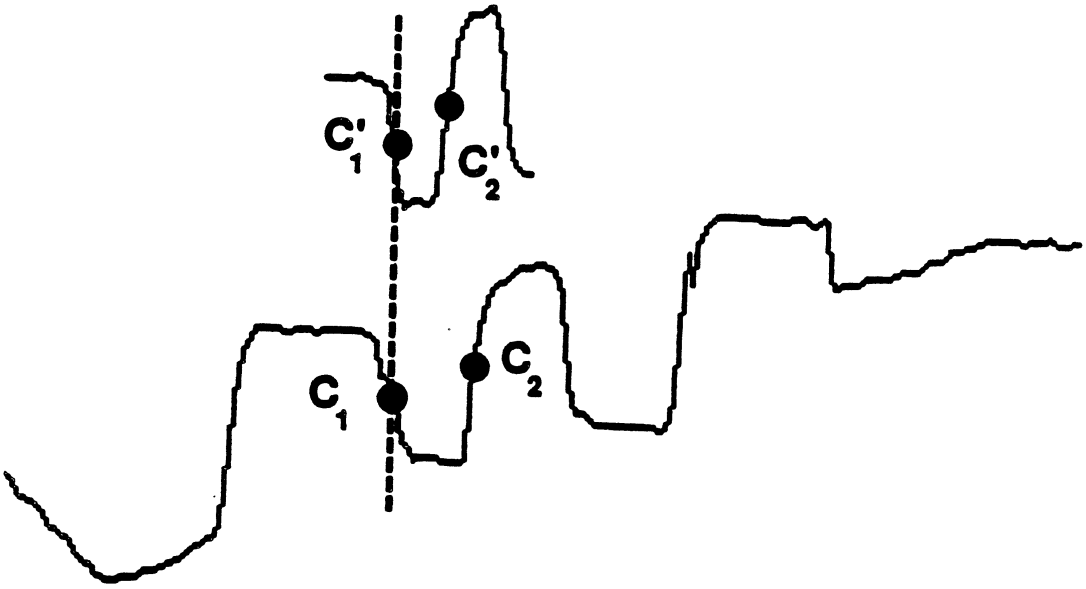

Fig. 4.10. Comparing segment to scaled segment.

In general, a critical point may have no corresponding critical point in the other boundary. This may be caused by occlusion or by curvature changes due to scaling. Therefore, it is necessary to try to match several consecutive pairs of critical points of the part-model to consecutive pairs of critical points of the image boundary.

4.4. Saliency

A configuration of a part can be assigned a saliency that measures the importance of the configuration in identifying the part. The saliency of a configuration is learned during off-line training from the boundaries of the complete set of parts that may appear in the image. The concept of saliency is central to our recognition algorithm. To simplify the presentation of the concept we begin by considering the the case where there is no noise in the image. We then show this

is a special case of saliency in the presence of noise.

Without noise. If parts are viewed without noise and all parts appear with equal likelihood in an image, we define the saliency of a configuration of segments to be the inverse of the frequency with which identical configurations appear in the set of parts. For example, assume that the notched rectangle and the square shown in Fig. 4.11 are the set of parts that may appear in an image, and that both have equal probability of appearing. The configuration of corner A and corner B has a saliency of $\frac{1}{2+4}$ or $\frac{1}{6}$, since identical configurations appear twice in the rectangle and four times in the square. Figure 4.12 shows how saliency is computed for the configuration of corners A and B. The dashed outlines indicate the six alignments of the set of parts that yield matches. The notation X-Y means that segment X from one of the parts is matched with segment Y of the rectangle. Note that, in effect, both parts are moved around to find matches with the configuration A and B. We could just as easily have imagined the matching process as moving the configuration while the parts were held fixed. Continuing our example, we see that the configuration of corner A and corner C has a saliency of $\frac{1}{2}$ since identical configurations appear twice in the rectangle. Finally, the configuration of corner A and the notch E has a saliency of 1 since this configuration appears only once in all the parts: it uniquely characterizes the rectangle and its pose in the image. If this configuration appears in the image, the pose of the rectangle is known with probability one, barring accidental alignments (see below). The calculation of saliency depends on knowing the set of all parts that may appear in an image, consequently part-models that make use of saliency implicitly incorporate comparative information about the particular part set.

Clearly, saliency is highly dependent on the set of parts. To illustrate this consider a set that contains only the notched rectangle of Fig. 4.11 and a similarly notched triangle (see Fig. 4.13). Reexamining the saliency of the configurations in the rectangle reveals that the configuration of A and B now has a saliency of $\frac{1}{2}$; the configuration of A and C still has a saliency of $\frac{1}{2}$; but the configuration of A and the notch E now has a saliency of $\frac{1}{2}$ and thus no longer

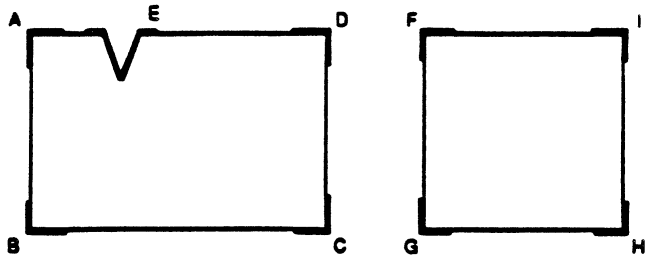

Fig. 4.11. Parts without noise.

uniquely identifies the pose of the rectangle. However, the pose of the rectangle is still uniquely identified by C and E.

The saliency of a configuration is in some sense indivisible; it cannot, in general, be determined from the frequency of occurrence of its component segments. For example, in Fig. 4.11 the saliency of the configuration formed by corners A and B is $\frac{1}{6}$, and the saliency of A and C is $\frac{1}{2}$, while the inverse of the frequency of the individual segments A, B, and C are all $\frac{1}{8}$.

We now turn to consider noisy images. If noise is present in an image the saliency of a configuration may be defined by generalizing the probabilistic viewpoint introduced informally above.

With noise. In the following, we will adapt the simple Bayesian argument of [TMV85b] to formalize the concept of saliency for parts when noise is present in the image. First some notation. Let $B_p(x, y, \theta)$ describe the boundary of part p from the set of known parts. The

parameters x and y are the coordinates of the centroid of the boundary and θ is the orientation of the boundary about the centroid—different values of x, y and θ correspond to different poses. In practice x, y, and θ are restricted to a finite set of values as a result of digitization. Thus, it is more accurate to represent $B_p(x, y, \theta)$ as $B_p(x_i, y_j, \theta_k)$ where i, j, and k are indices for the finite set of values to which x, y, and θ are restricted. Assume that $B_p(x_i, y_j, \theta_k)$ is partitioned into a set of overlapping segments. If $B_p(x_i, y_j, \theta_k)$ consists of u segments there will be $\frac{u(u-1)}{2}$ configurations (pairs of segments). Let $C_p^r(x_i, y_j, \theta_k)$ be the r^{th} configuration of boundary $B_p(x_i, y_j, \theta_k)$, where r ranges from 1 to $\frac{u(u-1)}{2}$.

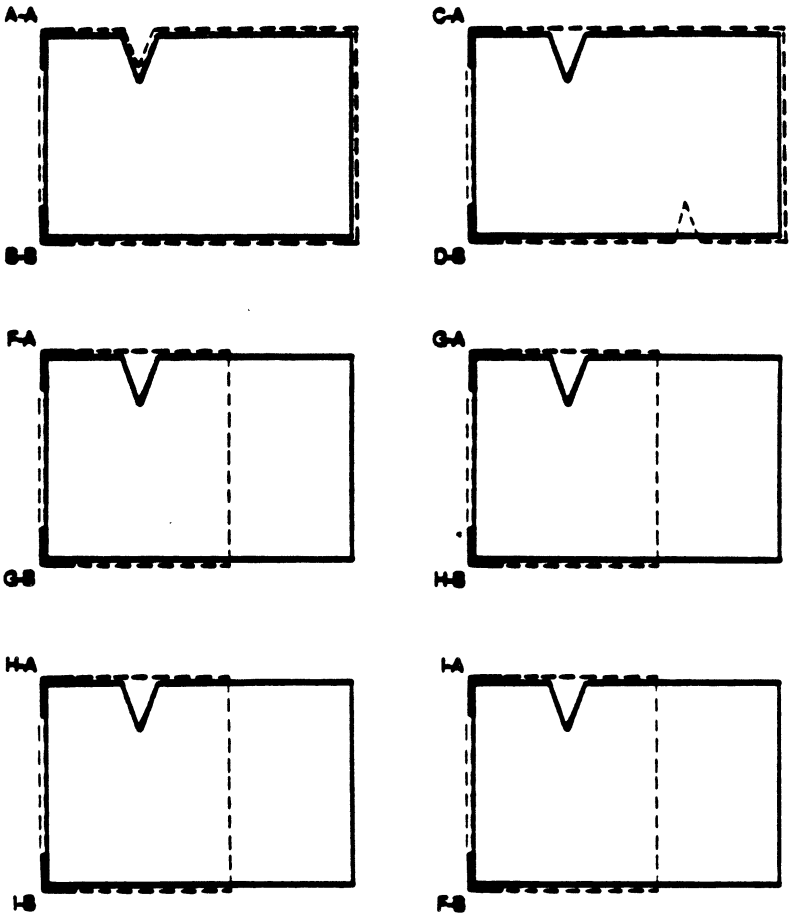

Fig. 4.12. Computing saliency.

Assume configuration C_p' is present in an image at pose x_i, y_j, and θ_k. In general, $C_p'(x_i, y_j, \theta_k)$ will be distorted by noise so that it will appear as some configuration $\mathbf{C}$. Let the probability that C_p' is distorted into $\mathbf{C}$ be represented by $Pr[\mathbf{C} \mid C_p'(x_i, y_j, \theta_k)]$. Without knowing in advance which configuration caused $\mathbf{C}$, the configuration $\mathbf{C}$ may be interpreted in several ways. Let the probability that $\mathbf{C}$ will be correctly interpreted as $C_p'(x_i, y_j, \theta_k)$ be represented by $Pr[C_p'(x_i, y_j, \theta_k) \mid \mathbf{C}]$. Then, the product

$$Pr[C_p'(x_i, y_j, \theta_k) \mid \mathbf{C}] \times Pr[\mathbf{C} \mid C_p'(x_i, y_j, \theta_k)],$$

is the probability that $C_p'(x_i, y_j, \theta_k)$ is present in the image, appears as a configuration $\mathbf{C}$, and is correctly interpreted as configuration $C_p'(x_i, y_j, \theta_k)$.

It is impossible to know *a priori* the form in which $C_p'(x_i, y_j, \theta_k)$ will appear in a image due to noise distortion. It is, however, still desirable to determine the probability with which the presence of $C_p'(x_i, y_j, \theta_k)$ will be correctly interpreted. We therefore define the saliency of configuration $C_p'(x_i, y_j, \theta_k)$ as the probability that $C_p'(x_i, y_j, \theta_k)$ will be present in an image and will be correctly interpreted, given that $C_p'(x_i, y_j, \theta_k)$ could be distorted into any possible configuration $\mathbf{C}$. This definition can be written as

$$SA(C_p'(x_i, y_j, \theta_k)) \triangleq \sum_{\mathbf{C}} Pr[C_p'(x_i, y_j, \theta_k) \mid \mathbf{C}] \times Pr[\mathbf{C} \mid C_p'(x_i, y_j, \theta_k)].$$

Saliency is a measure of how unambiguously a configuration will be recognized, given all the possible distortions it can undergo due to noise. If the noise is independent of x, y, and θ, the saliency is also independent of x, y, and θ, in which case, the above definition can be rewritten as,

$$SA(C_p') \triangleq \sum_{\mathbf{C}} Pr[C_p'(x_i, y_j, \theta_k) \mid \mathbf{C}] \times Pr[\mathbf{C} \mid C_p'(x_i, y_j, \theta_k)]. \tag{4.4}$$

The probabilities still depend on the pose x_i, y_j, and θ_k. However, it is not important which pose, only that a particular one be chosen. Varying x_i, y_j, and θ_k changes the terms that contribute in (4.4), but the summation remains constant because the it is taken over all possible configurations $\mathbf{C}$. We will further explore this definition by individually examining the terms on the

right-hand side.

The term $Pr[\,\mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,]$ is a noise distribution for the image. It is the probability that configuration $C_p^r(x_i, y_j, \theta_k)$, when present in the image, will be distorted by noise and appear as configuration $\mathbb{C}$. The noise distribution is intrinsic to the image and not to the configuration $C_p^r(x_i, y_j, \theta_k)$.

The term $Pr[\, C_p^r(x_i, y_j, \theta_k) \mid \mathbb{C}\,]$ is the probability that the appearance of $\mathbb{C}$ in the image will be interpreted as $C_p^r(x_i, y_j, \theta_k)$. If there are many configurations that can appear as $\mathbb{C}$ the probability will be low. If, however, only a few configurations, including $C_p^r(x_i, y_j, \theta_k)$, can appear as $\mathbb{C}$ the probability will be high. The expression for $Pr[\, C_p^r(x_i, y_j, \theta_k) \mid \mathbb{C}\,]$ can be rewritten as

$$Pr[\, C_p^r(x_i, y_j, \theta_k) \mid \mathbb{C}\,] = \frac{Pr[\,\mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,] \times Pr[\, C_p^r(x_i, y_j, \theta_k)\,]}{Pr[\,\mathbb{C}\,]}, \tag{4.5}$$

where $Pr[\, C_p^r(x_i, y_j, \theta_k)\,]$ is the *a priori* probability that configuration $C_p^r(x_i, y_j, \theta_k)$ is present in the image, and $Pr[\,\mathbb{C}\,]$ is the total probability that configuration $\mathbb{C}$ appears in the image. Applying Bayes' rule to $Pr[\,\mathbb{C}\,]$ yields

$$Pr[\,\mathbb{C}\,] = \sum_{q,s,l,m,n} Pr[\,\mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,] \times Pr[\, C_q^s(x_l, y_m, \theta_n)\,], \tag{4.6}$$

where $C_q^s(x_l, y_m, \theta_n)$ is the s^{th} configuration of boundary $B_q(x_l, y_m, \theta_n)$. Part q is any of the set of known parts, including p, that can appear in the image. The term $Pr[\,\mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]$ is the probability that configuration $C_q^s(x_l, y_m, \theta_n)$ is present in the image, it appears as $\mathbb{C}$. The term $Pr[\, C_q^s(x_l, y_m, \theta_n)\,]$ is the *a priori* probability that $C_q^s(x_l, y_m, \theta_n)$ is present in the image. We have assumed in this expansion that the segments in a configuration come from the same part, and not from the accidental alignment of segments of two or more parts. More precisely, an *accidental alignment* occurs when a configuration of segments from two or more parts happens to fall in a relative position that resembles a configuration of segments from a single part. Strictly speaking (4.6) should also include terms of the form $Pr[\mathbb{C} \mid S_{q_1}^i S_{q_2}^j] \times Pr[S_{q_1}^i S_{q_2}^j]$ and higher order joint probabilities, where $S_{q_1}^i$ and $S_{q_2}^j$ are the i^{th} and j^{th} segments of different parts q_1 and q_2. Accidental alignment would cause some of these terms to be non-zero. We assume that

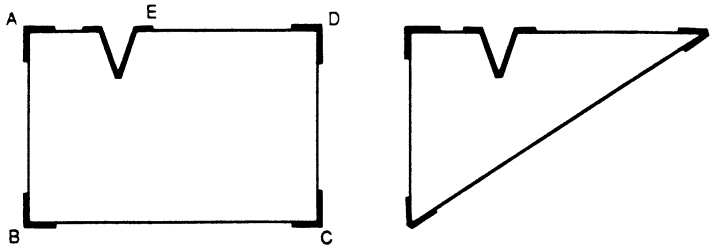

Fig. 4.13. Notched rectangle and triangle without noise.

accidental alignments have negligible probability. With this assumption (4.5) becomes

$$Pr[\ C_p^r(x_i, y_j, \theta_k)\ |\ \mathbb{C}\ ] = \frac{Pr[\ \mathbb{C}\ |\ C_p^r(x_i, y_j, \theta_k)\ ] \times Pr[\ C_p^r(x_i, y_j, \theta_k)\ ]}{\sum_{q,s,l,m,n} Pr[\ \mathbb{C}\ |\ C_q^s(x_l, y_m, \theta_n)\ ] \times Pr[\ C_q^s(x_l, y_m, \theta_n)\ ]} \tag{4.7}$$

If we assume that a part is equally likely to be at any pose in the image then $Pr[\ C_q^s(x_l, y_m, \theta_n)\ ]$ is a constant independent of x, y, and θ. This constant is proportional to,

$$\frac{\textit{the frequency of parts of type } q \textit{ in images}}{\textit{number of digitized poses}}$$

To simplify the discussion we will assume that the parts occur equiprobably; therefore, the constant terms are all equal and, thus, cancel one another. This results in the following,

$$Pr[\, C_p^r(x_i, y_j, \theta_k) \mid \mathbb{C} \,] = \frac{Pr[\, \mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,]}{\sum_{q,s,l,m,n} Pr[\, \mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]} . \tag{4.8}$$

Substituting (4.8) into (4.4), the saliency of $C_p^r(x_i, y_j, \theta_k)$ becomes

$$SA(C_p^r) = \sum_{\mathbb{C}} \frac{Pr[\, \mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,]^2}{\sum_{q,s,l,m,n} Pr[\, \mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]} . \tag{4.9}$$

If we assume, for the moment, that no noise is associated with the boundaries of the image, equation (4.9) can be greatly simplied. Let $I(e \in A)$ represent an indicator function whose value is 1 when e is an element of the set A and whose value is 0 otherwise. In the noiseless case, if configuration $C_q^s(x_l, y_m, \theta_n)$ is present in an image it will appear as $C_q^s(x_l, y_m, \theta_n)$; thus, $Pr[\, \mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]$ becomes an indicator function $I(\mathbb{C} \in \{C_q^s(x_l, y_m, \theta_n)\})$ defined for the singleton set $\{C_q^s(x_l, y_m, \theta_n)\}$. Substituting for $Pr[\, \mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,]$ and $Pr[\, \mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]$ in (4.9) we obtain,

$$SA(C_p^r) = \sum_{\mathbb{C}} \frac{I(\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\})^2}{\sum_{q,s,l,m,n} I(\mathbb{C} \in \{C_q^s(x_l, y_m, \theta_n)\})} = \frac{1}{\sum_{q,s,l,m,n} I(C_p^r(x_i, y_j, \theta_k) \in \{C_q^s(x_l, y_m, \theta_n)\})}$$

By summing over all x_l, y_m, and θ_n we are, in effect, moving part q so that each of its configurations, s, is compared with the fixed configuration $C_p^r(x_i, y_j, \theta_k)$. The resulting saliency is the inverse of the frequency with which configurations $C_q^s(x_l, y_m, \theta_n)$, that are identical to configuration $C_p^r(x_i, y_j, \theta_k)$, occur in the set of parts. This agrees with our earlier informal discussion of saliency.

Now, returning to the noisy situation, we observe that if the noise distribution $Pr[\, \mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]$ is known, the saliency for configuration $C_p^r(x_i, y_j, \theta_k)$ can be calculated from (4.9). On the other hand, if the noise distribution is not known but is characterized by a fit tolerance, D, the expression on the right in (4.9) can be approximated as follows.

Denote by $\{\mathbb{C}\}_D$ the set of configurations with segments which are in the same relative pose as the segments of a configuration $\mathbb{C}$ and which are individually within a tolerance D in θ–s space of corresponding segments of $\mathbb{C}$. This set is simply the configurations that match $\mathbb{C}$ in the

sense defined after (4.1). The set can also be viewed as a radius D sphere in configuration space centered on $\mathbb{C}$. Configuration space contains the set of all possible configurations. Points in this space include all the configurations C_q^s at all poses. The metric is the fit given by (4.1). We assume a constant density of configurations in configuration space, and let the number of configurations in a radius D sphere in configuration space be N, a constant. Then, we can approximate $Pr[\,\mathbb{C} \mid C_q^s(x_l, y_m, \theta_n)\,]$ by the term $\frac{1}{N} \times I(\mathbb{C} \in \{C_q^s(x_l, y_m, \theta_n)\}_D)$. In other words, we have assumed that the probability that a configuration, $\mathbb{C}$, can be distorted enough to fall outside of the radius D sphere about $C_q^s(x_l, y_m, \theta_n)$ is 0. Similarly, we approximate $Pr[\,\mathbb{C} \mid C_p^r(x_i, y_j, \theta_k)\,]$ by $\frac{1}{N} \times I(\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\}_D)$. Substituting the above two terms into (4.9) we obtain,

$$SA(C_p^r) \approx \sum_{\mathbb{C}} \frac{\left(\frac{I(\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\}_D)}{N} \right)^2}{\sum_{q,s,l,m,n} \frac{I(\mathbb{C} \in \{C_q^s(x_l, y_m, \theta_n)\}_D)}{N}} \tag{4.10}$$

$$\approx \frac{1}{N} \times \sum_{\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\}_D} \frac{1}{\sum_{q,s,l,m,n} I(\mathbb{C} \in \{C_q^s(x_l, y_m, \theta_n)\}_D)}$$

In the denominator of (4.10) configuration $\mathbb{C}$ is held fixed while the sum over q, s, l, m, and n selects sets $\{C_q^s(x_l, y_m, \theta_n)\}_D$ that contain $\mathbb{C}$. Thus, the summation counts the number of spheres that contain a particular value of $\mathbb{C}$. This sum is equal to the sum

$$\sum_{q,s,l,m,n} I(C_q^s(x_l, y_m, \theta_n) \in \{\mathbb{C}\}_D). \tag{4.11}$$

The equality follows from the observation that the number of radius D spheres centered on the possible values of $C_q^s(x_l, y_m, \theta_n)$ that contain a particular $\mathbb{C}$ is the same as the number of centers $C_q^s(x_l, y_m, \theta_n)$ within a sphere of radius D centered on the particular value of $\mathbb{C}$. Substituting (4.11) into (4.10) yields,

$$\frac{1}{N} \times \sum_{\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\}_D} \frac{1}{\sum_{q,s,l,m,n} I(C_q^s(x_l, y_m, \theta_n) \in \{\mathbb{C}\}_D)} . \tag{4.12}$$

Diagrammatically, the C_q^s terms counted in the denominator of (4.12) are shown in Fig. 4.14. The solid circles represent the radius D spheres. Therefore, configurations within the circle centered on $\mathbb{C}$ are those within a D tolerance of $\mathbb{C}$.

If we assume that the density of $C_q^s(x_l, y_m, \theta_n)$ configurations is locally constant within a $2D$ sphere centered on C_p^r (the dashed circle in Fig. 4.14), then we can obtain an estimate of the number of C_q^s's within D of $\mathbb{C}$ by the number of C_q^s within D of C_p^r. Substituting this number in (4.12) yields,

$$\frac{1}{N} \times \sum_{\mathbb{C} \in \{C_p^r(x_i, y_j, \theta_k)\}_D} \frac{1}{\sum_{q,s,l,m,n} I(C_q^s(x_l, y_m, \theta_n) \in \{C_p^r(x_i, y_j, \theta_k)\}_D)} .$$

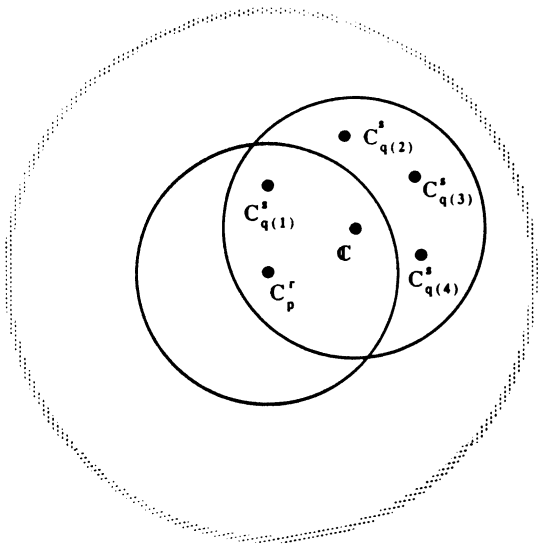

Fig. 4.14. Spheres in configuration space.

Since, the sum in the denominator is now approximated by a sum independent of $\mathbb{C}$ and the cardinality of $\{C_p^r(x_i, y_j, \theta_k)\}_D$ is simply N, we can eliminate the summation over $\mathbb{C}$ to yield the following modified definition of saliency:

$$\mathbf{SA}(C_p^r) \triangleq \frac{1}{\sum_{q,s,l,m,n} I(C_q^s(x_l, y_m, \theta_n) \in \{C_p^r(x_i, y_j, \theta_k)\}_D)} \tag{4.13}$$

This is the working definition that we use to calculate saliency during training. In other words, we approximate the saliency of configuration $C_p^r(x_i, y_j, \theta_k)$ as the inverse of the frequency of all configurations of the set of parts that have segments in the same relative pose as those of $C_p^r(x_i, y_j, \theta_k)$ and which are within a D tolerance of $C_p^r(x_i, y_j, \theta_k)$.

There are two points to discuss before concluding this section on saliency. First, in the special case of symmetric parts we are generally uninterested in which of the equivalent symmetric poses the part is found. For example, if a part has n rotational symmetries, n of its poses are equivalent to us. In this case the saliency of any configuration of the part should be modified by multiplying its normal saliency by the symmetry of the part. In other words, the saliency of a configuration C_p^r would become $n \times \mathbf{SA}(C_p^r)$. Clearly, this saliency is defined with respect to the part—the same configuration in the other parts in the part set, i.e. parts without n rotational symmetries, would not have the same value of saliency. Second, if there is *a priori* knowledge about the frequency of occurrence of each part in typical application images, the saliency of part-configurations can be weighted by a normalized frequency of occurrence factor. This factor can be accounted for by rederiving (4.13) from (4.7), but with the term $Pr[\, C_q^s(x_l, y_m, \theta_n)\,]$ now a function of the part q.

4.5. The Automatic Generation of Salient Configurations

Training is the process of determining the saliencies of the configurations of a set of parts. Assume that we wish to determine the saliencies of the configurations of boundary, B_p. An obvious approach is to start by comparing all of the configurations of B_p to those of another boun-

dary, B_q, as was done in Fig. 4.12. This is inefficient. If there were u_p segments in boundary B_p and u_q segments in B_q we would compare $\frac{u_p(u_p-1)}{2}$ configurations in B_p with $\frac{u_q(u_q-1)}{2}$ configurations in B_q for a total of $\frac{u_p(u_p-1)}{2} \times \frac{u_q(u_q-1)}{2}$ comparisons. Since all parts must be compared there would be a grand total of,

$$\sum_{p \in P} \frac{u_p(u_p-1)}{2} \times \sum_{q \in P} \frac{u_q(u_q-1)}{2}, \tag{4.14}$$

where P is the set of known parts. Even though this is a one-time off-line computation, it is unacceptably inefficient.

Instead the following approach is taken. The segments of B_q are compared to the segments of B_p. If a segment of B_q at pose (x_l, y_m, θ_n) matches a segment $S_p^r(x_i, y_j, \theta_k)$ of B_p, the pose (x_l, y_m, θ_n) and the identity of S_p^r, i.e., the index r, are stored in a *match table*. The match should satisfy the D tolerance. As before, we consider the configuration to be fixed at (x_i, y_j, θ_k) while the segments of B_q are moved. The match table is implemented as a hash table with the ordered triple (x_l, y_m, θ_k) as the primary key. A key may have multiple indices stored at its associated table location. After all segments of B_q have been matched to those of B_p, the match table is searched for pairs of segments of B_p which matched B_q at the same pose. These are simply pairs of indices at the same key. If two segment of B_p match two segments of B_q at the same pose (x_l, y_m, θ_k) then the configuration of the two segments of B_p must match a configuration of segments of $B_q(x_l, y_m, \theta_n)$. Thus, searching the match table for pairs of indices at the same key is equivalent to searching for matching configurations.

A 2-dimensional array, indexed by the identities of pairs of segments in B_p, is used to record the frequency with which configuration-pairs match the boundary B_q. The array elements are initially zero. Each time a configuration of B_p is found which matches a configuration of $B_q(x_l, y_m, \theta_n)$, the element of the array corresponding to the pair is incremented.

After the match table has been completely searched, it is cleared and the segments of another boundary, B_q, are matched to the segments of B_p. This is repeated for all q, including p

itself. Each time, the array of frequencies is updated to reflect the number of matching configurations. When the segments of all parts have been matched to the segments of B_p, the reciprocals of the elements of the array yields the saliencies of the configurations of B_p. The configurations, C_p^r, with associated saliencies, $\mathbf{SA}(C_p^r)$, form the part-model of p. In most applications only those values C_p^r with $\mathbf{SA}(C_p^r) \approx 1$ are retained for the the part-model. The whole procedure is repeated for each part. It is straightforward to show that the number of comparisons required by the training procedure is given by,

$$\sum_{p \in P} \sum_{q \in Q} u_p u_q \ .$$

This compares favorably with (4.12).

4.6. A Strategy for the POP Recognition Problem

If a particular part is sought, an efficient strategy searches for configurations from the associated part-model in order of decreasing saliency. As an example, consider searching for the rectangle of Fig. 4.11 in a scene with the rectangle partly obscured by the square from the same figure (the scene is depicted in Fig. 4.15). Configurations which include the notch as a segment have the largest values of saliency and should be compared first to the POP image. If no such pair can be found, a configuration with less saliency, e.g., corner A and corner B, would then be compared to the image. This process is continued until the part is found or no untried configurations are left in the part-model. If the latter situation occurs we assume the part is not present or totally hidden. If more than one part, for example a subset of the set of parts, is sought, a efficient strategy is to search for all the configurations from the subset in order of decreasing saliency.

Searching for configurations of a particular saliency can best be done by first searching for the segment that occurs most often in the configurations. The search for individual segments can be done in the manner outlined in the previous section, and illustrated by the following.

If the segment has a curvature extremum, we align the curvature extremum of the segment with an extremum in the image boundary before comparing the segments. On the other hand, if

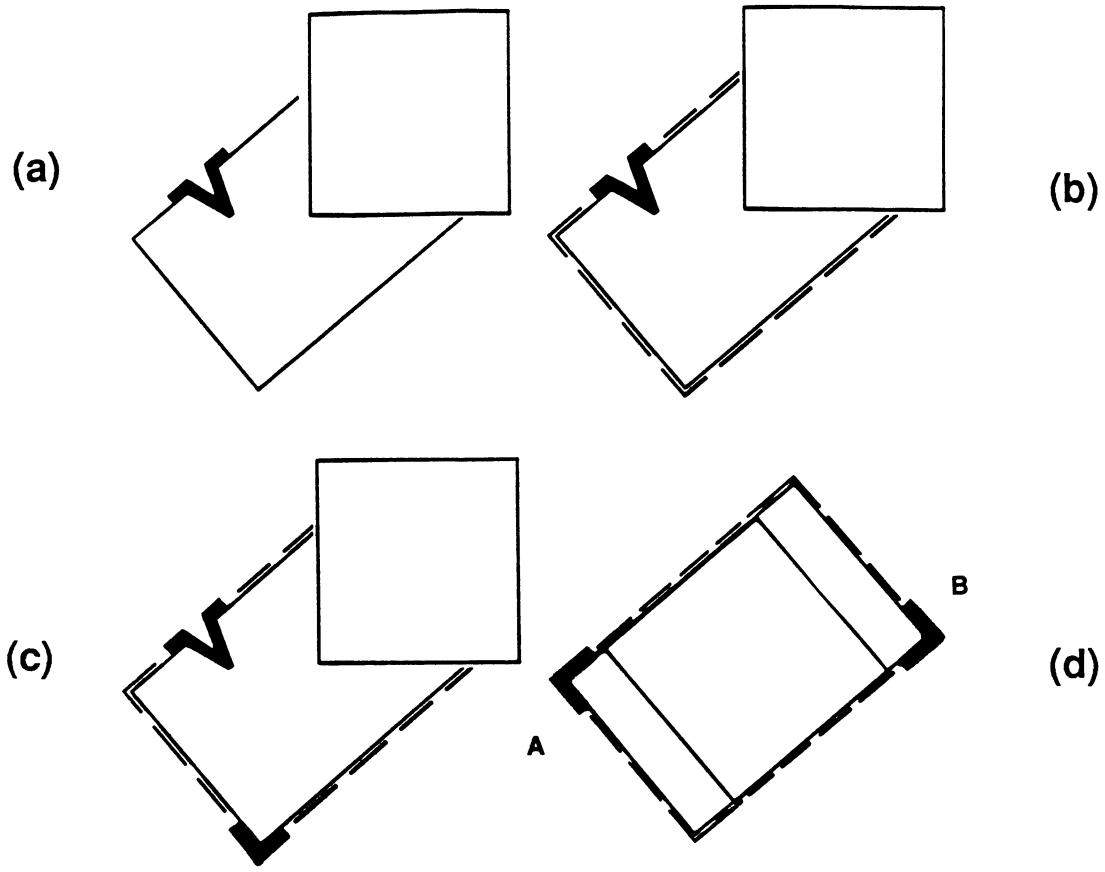

Fig. 4.15. Locating a part.

the segment has no extremum, it is necessary compared to all segments of the image boundary. In both cases, comparison is performed in the θ–s representation. If a good match is found between the part-model segment and a segment of the image boundary (see notch in Fig. 4.15a), the rotation and translation necessary to align the two segments is computed by performing a least squares fit of the two segments in cartesian space. The rotation and translation are applied to the entire boundary of the part (see dotted outline in Fig. 4.15b) and the transformed boundary is used as a guide in searching for the second segment of a configuration with high saliency (see the lower corner in Fig. 4.15c). The saliency provides an estimate of the probability that the correct part at the correct pose has been located. In our example, if the notch in the rectangle

were not visible in the POP image (see Fig. 4.15d), the algorithm locates the rectangle using a less salient pair (corner A and corner B in Fig. 4.15d) and reports that the probability of it having found the correct pose is $\frac{1}{2}$, i.e., the saliency of A and B.

A variation on the above search strategy, would be to store pointers back to all the parts that contained each configuration. The search starts with configurations (the configurations used in consist of single segments) that have non-maximal saliencies. When such configurations are located in the image their possible interpretations are determined by fitting all the boundaries from which the configurations may have be taken to the image, and then selecting the boundary and pose with the best fit as the correct interpretation. This only works if there are a large number of parts in the set and only a few are expected to appear in any image; and if fitting the entire boundary can be done efficiently.

CHAPTER V

PREVIOUS POP WORK

5.1. Introduction

In this chapter we will describe the approaches that we have developed in the past for the POP recognition problem. This material, perhaps, more properly belongs to the review of Chapter II. However, we have chosen to isolate it here in order to justify some of the design features of our current approach and to show how our approach evolved.

In the following we will summarize the papers in which our work appeared, and, in addition, we will discuss some of our unpublished results.

5.2. Earlier Algorithms

J. L. Turney, T. N. Mudge, R. A. Volz, and M. D. Diamond, "Experiments in Occluded Parts Recognition Using the Generalized Hough Transform." In this paper [TMV83a] we presented an improved version of the generalized Hough transform (GHT) discussed in Sec. 2.2.1 (see also [MeF75, Bal81].)

The GHT normally determines the pose of a part in a POP image by matching the edge points of the boundary of the part to the edge points of the boundaries in the image. The major shortcoming of the GHT is that it often incorrectly matches a large fraction of the edge points and as a consequence determines an incorrect pose for the part.

To improve the performance of the GHT we designed an algorithm in which edge points of the part and image were allowed to match only if their *neighborhoods* matched. We chose as a neighborhood a fixed length segment of the part or image boundary, centered on the edge point of interest. For example, a neighborhood of edge point a in Fig. 5.1a would be segment A. Under our constraint an edge point, a, would only be allowed to match an image edge point, a', (see Fig. 5.1b) if the segment, A, centered on a matched the segment, A', centered on, a'. In essence we altered the problem from matching edge points to one of matching segments. In any case, it is clear that our constraint eliminated a large number of incorrect matches.

However, even with this constraint incorrect matches can still accidentally occur when the segments being matched are common to several locations on a part or common to many parts. For example, segments with little structure such as the straight line segment S and the circular

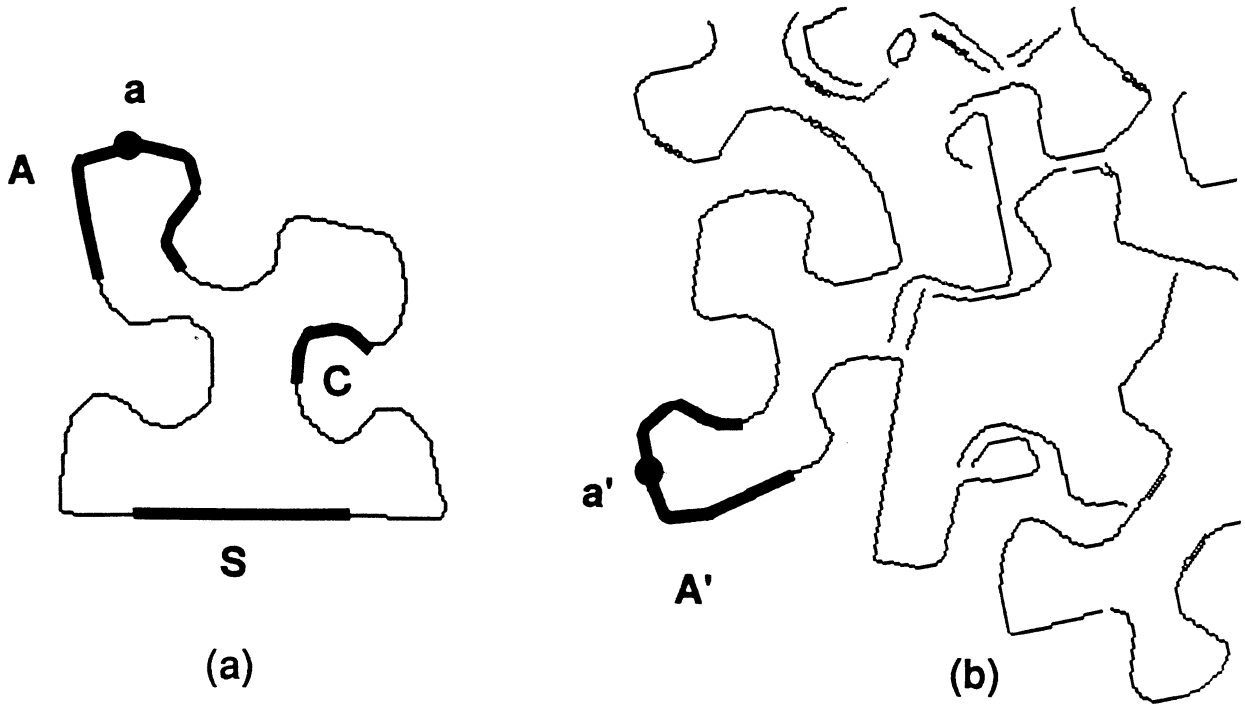

Fig. 5.1. Part-model and image boundary segments.

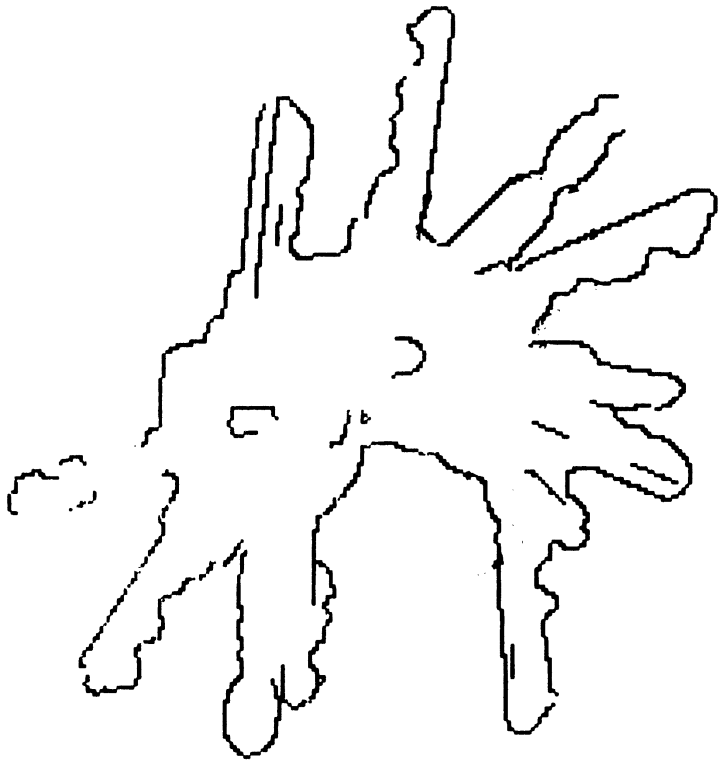

Fig. 5.2. Occluded keys.

arc segment C will often incorrectly match many straight line and circular arc segments in the image boundary (see Fig. 5.1b). To deemphasize the contribution of this type of segment in determining the pose of a part, we assigned weights to segments based on their uniqueness, i.e., the more unique the segment the more the weight it was assigned. We called this weight the "saliency" of the segment. The algorithm for determining the segment weights will be discussed in the summary of the next paper.

The segments and their associated weight form the part-model. We determined the pose of the part by comparing segments of the part-model to segments of the image boundary and by accumulating the weight for each matching pair of segments in a table indexed by the pose at which the segments matched. The pose was represented by a triple (θ, u, v), where u and v were the cartesian coordinates of the centroid of the part and θ was the orientation of the part about the centroid. This triple was used as a key into the table. The pose was determined by

rotating and translating the centroid of the part boundary by the rotation and translation necessary to bring the matching segments into alignment. After all segments of the part were matched to those of the image, the pose with the largest accumulated weight was taken as the pose of the part. By this technique, all matches between segments contributed to determining the pose of the part, but the unique segments, the segments more likely to indicate a correct match, contributed more weight.

In order to match segments, we represented them in a θ–s representation similar to that used in our current approach (see Sec. 4.3.1). In several experiments we found that the θ–s representation of contours were *less* distorted by noise than the cartesian representations of the same contours. In addition, the θ values demonstrated good invariance, within a linear shift in θ, to rotations. This invariance is necessary if rotated parts are to be located from the θ–s representation of a non-rotated part boundary.

To compare a part-model segment with an image boundary segment, we selected the sum of the squares of the differences between corresponding slope angles in the $\theta(s)$ functions of the two segments as a measure of the closeness of the fit. We assume the functions were both sampled at uniform arclength spacings. Again, this is similar to our current approach (see Sec 4.3.1). Besides its simplicity, this measure has the advantage that it allows some latitude in matching segments. This latitude is required because it is not always possible to obtain precise estimates of the arclength, s, along the boundary. With the sum of squares of the differences as a measurement of similarity, small deviations in the s values do not significantly affect the outcome in comparing two segments.

We tried other measures for comparing segments, in particular the maximum of the minimum difference between θ values of the segments. This measure proved to be more costly in the amount of time it took to compute and showed less tolerance to deviations in the arclength, than the sum of squares measure.

During recognition all segments of the part boundary are compared to the segments of the image boundary. For a part boundary containing n points and an image boundary containing m

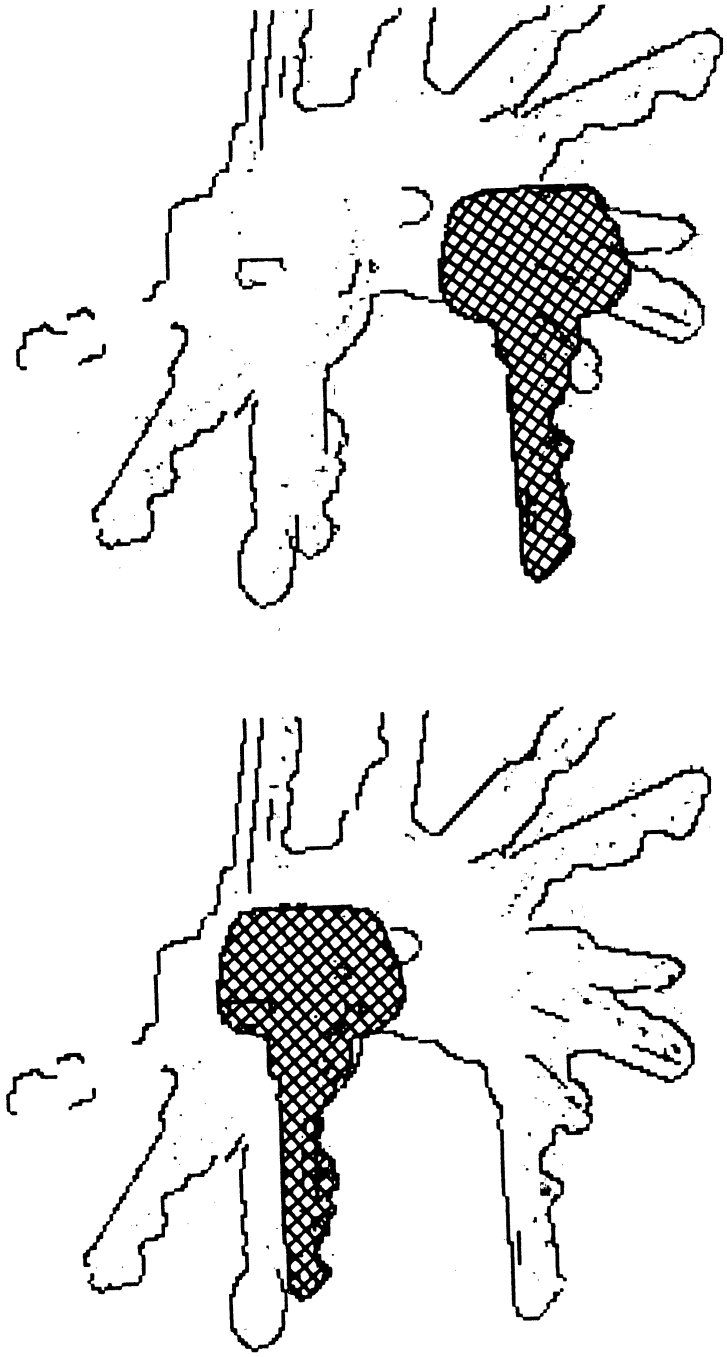

Fig. 5.3. Keys located.

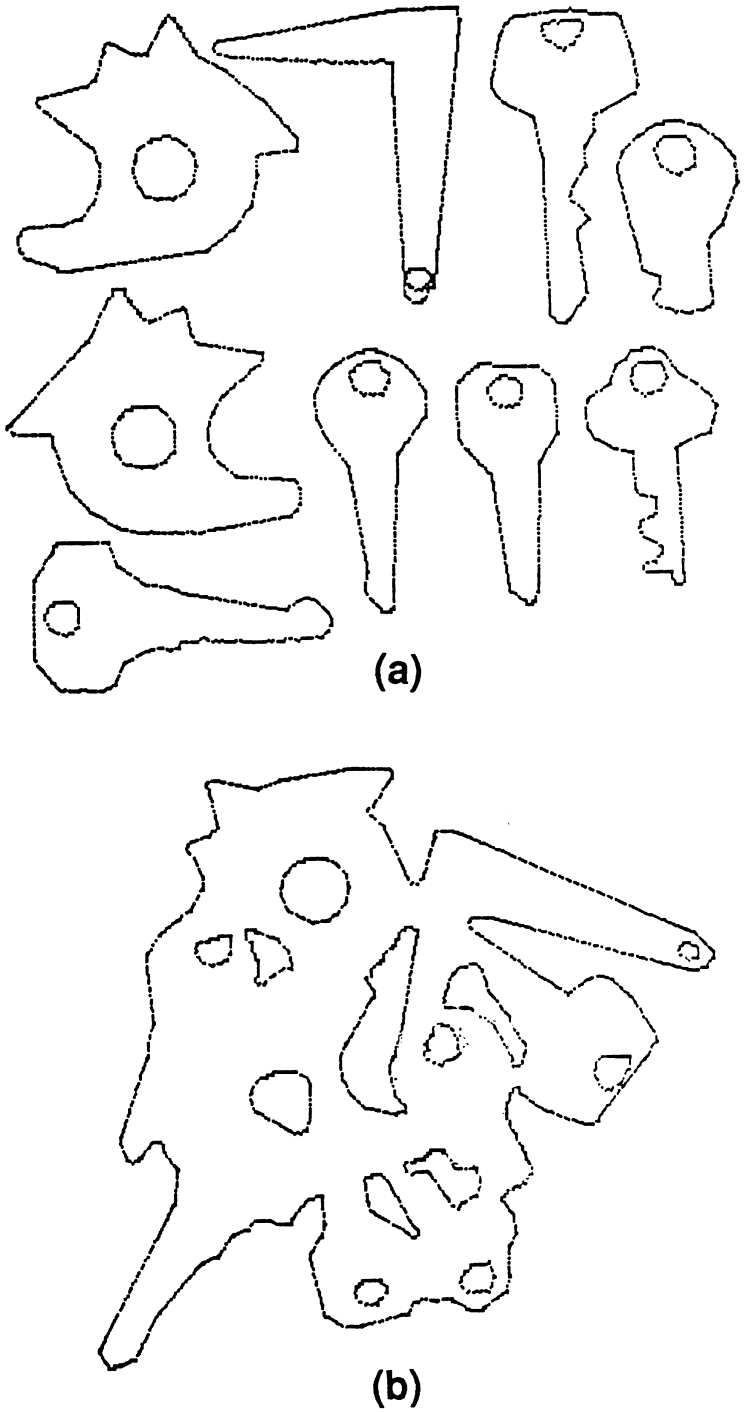

Fig. 5.4. Set of nine parts and occluded image.

points, $O(n \times m)$ segment comparisons are required for the recognition of one part.

Figure 5.2 shows a scene containing a set of occluded office keys. Using our algorithm we

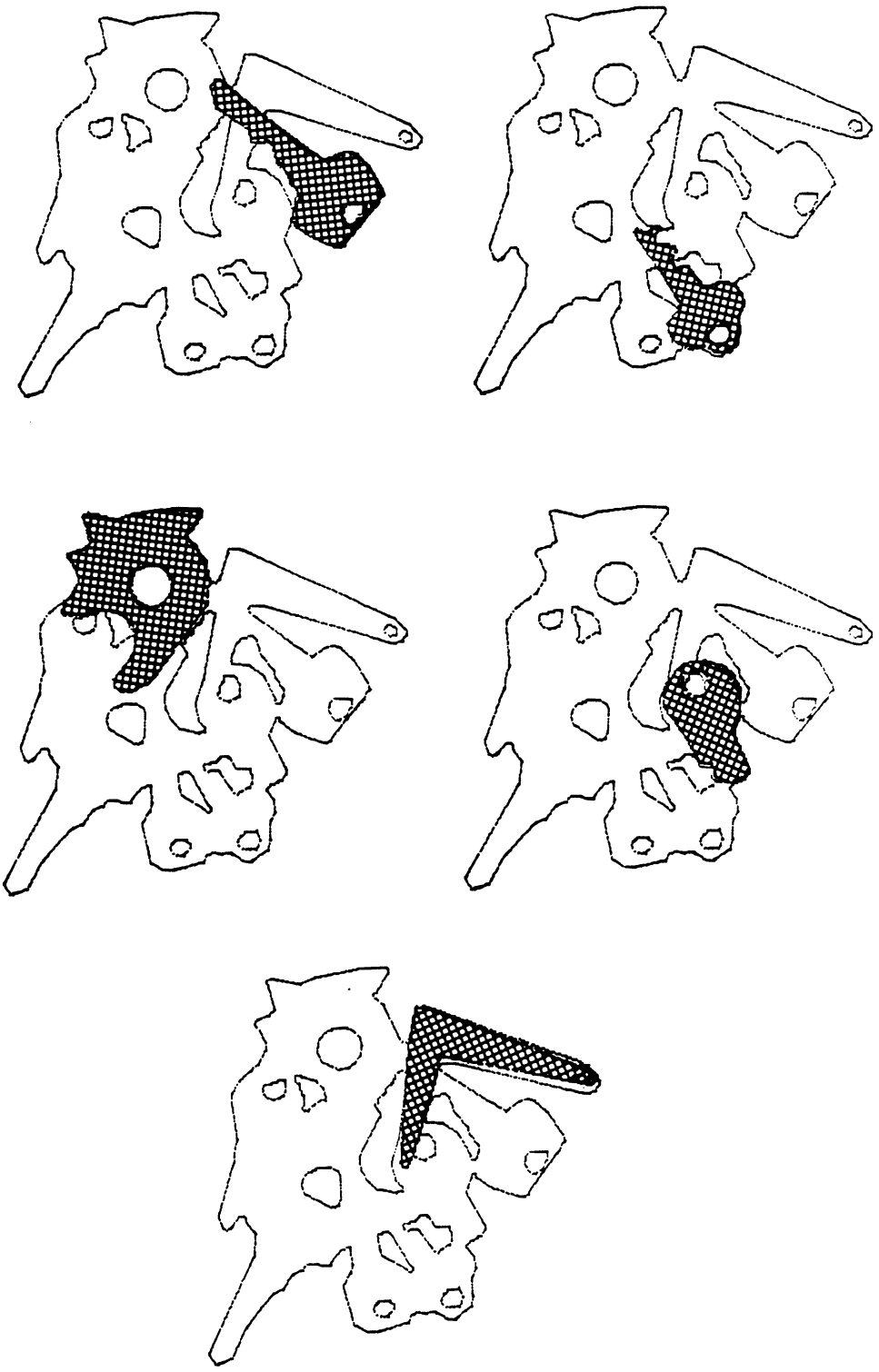

Fig. 5.5. Parts located using weighted segment approach.

were able to locate the keys that we attempted to find (the keys are shown cross hatched in Fig. 5.3a-b). In the experiment, segments of the boundary around the notches on the keys received the greatest weight and were largely responsible for determining the pose of the keys.

J. L. Turney, T. N. Mudge, and R. A. Volz "Experiments in Occluded Parts Recognition." In this paper [TMV83b] we presented further improvements to the approach discussed above. A quadratic optimization scheme was developed for determining the weights assigned to part-model segments and additional experiments were performed using the improved approach.

The weights assigned to part-model segments were automatically determined off-line as follows. For simplicity, assume that two parts are to appear in an image. Let their part-models be denoted by P_1 and P_2. Although P_1 and P_2 are both part-models, it is convenient for this explanation to consider P_1 to be a part-model and P_2 to be an image boundary. The segments of P_1 were compared to the segments of P_2 as in the normal matching scheme. If a segment i of P_1 matched a segment j of P_2, a record of the match $c_{i,j} \times w_i$ was stored in a table indexed by the pose at which the segments match. The *matching coefficient*, $c_{i,j}$, was the taken to be $(1 + \gamma_{i,j})^{-1}$, where $\gamma_{i,j}$ is the sum of the squares of the differences in the $\theta(s)$ values of the segments. Thus $c_{i,j}$ was in the half open interval (0,1]. The closer two segments matched, the closer the matching coefficient, $c_{i,j}$ was to 1. The term w_i symbolically represented the weight assigned to the segment i of P_1. The product $c_{i,j} \times w_i$ itself was symbolically added to the previous contents of the table. After all segments were matched the table contained symbolic sums of the weights accumulated at every pose at which a segment of P_1 matched a segment of P_2.

The part-model P_1 was matched in a similar way to the other part-models (including itself) with the results stored in separate tables. After the matching was completed, the weights, w_i were adjusted to minimize the symbolic sums found in the tables for all pose except the correct one. This was accomplished by minimizing the sum of the squares of the symbolic sums stored in the tables subject to the constraint that the weights be positive and sum to 1. (The last constraint was added to keep the weights from taking on values of zero.) With the set of weights determined

by this method, matching the segments of part P_1 to an image containing other parts, e.g., P_2, would result in very little accumulated weight for any pose of P_1. Thus the other parts would not be interpreted as P_1. However, if P_1 was matched to itself the accumulated weight at the correct pose would sum to 1, and P_1 could be clearly located. The segments that were unique to a part occurred in few of the symbolic sums and were, as a result of optimization, given larger weights.

Figure 5.4a shows a set of nine parts. Most of the parts are office keys with the exception of one part, which we call door lock part no. 2, which appears once on its obverse side and once on its reverse side. Figure 5.4b shows an image in which the parts occlude one another. Figure 5.5 shows the results of applying our algorithm to the image for several parts. The results were quite reasonable. The only shortcoming to the approach was its complexity.

Some of work that we performed at this time, but did not report, was concerned with scaled parts. Figures 5.6 and 5.7 shows the θ–s representation of several scaled views of two different parts. The representations have been normalized along the s axis so that they have the same total arclength. From these results it is clear that most of the structure of the θ–s representation, even for the notches of the key, is preserved under scaling. In the case of the key, less structure about the notches appears to be preserved in the cartesian representation than in the θ–s representation. In addition, critical points in curvature, corresponding to the locations on the θ–s of large slopes, were, generally, preserved. These observations led to our proposed scheme for locating scaled parts (see Sec. 4.3).

J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing Partially Occluded Parts." In this paper [TMV85a] we summarized the work that had been accomplished using the weighted segment approach discussed above. In addition, we presented a more formal development of the optimization technique used to determine the segment weights, and we reported experiments comparing the performance of our algorithm to the GHT of Ballard [Bal81].

Figure 5.8c shows the results of using the GHT in a simple experiment in which a key is to be located in an image. Three office keys A, B, and C (see Fig. 5.8a) appear in an image (see Fig. 5.8b). Keys A and C are identical except for their notches. Key B is placed on top of key A,

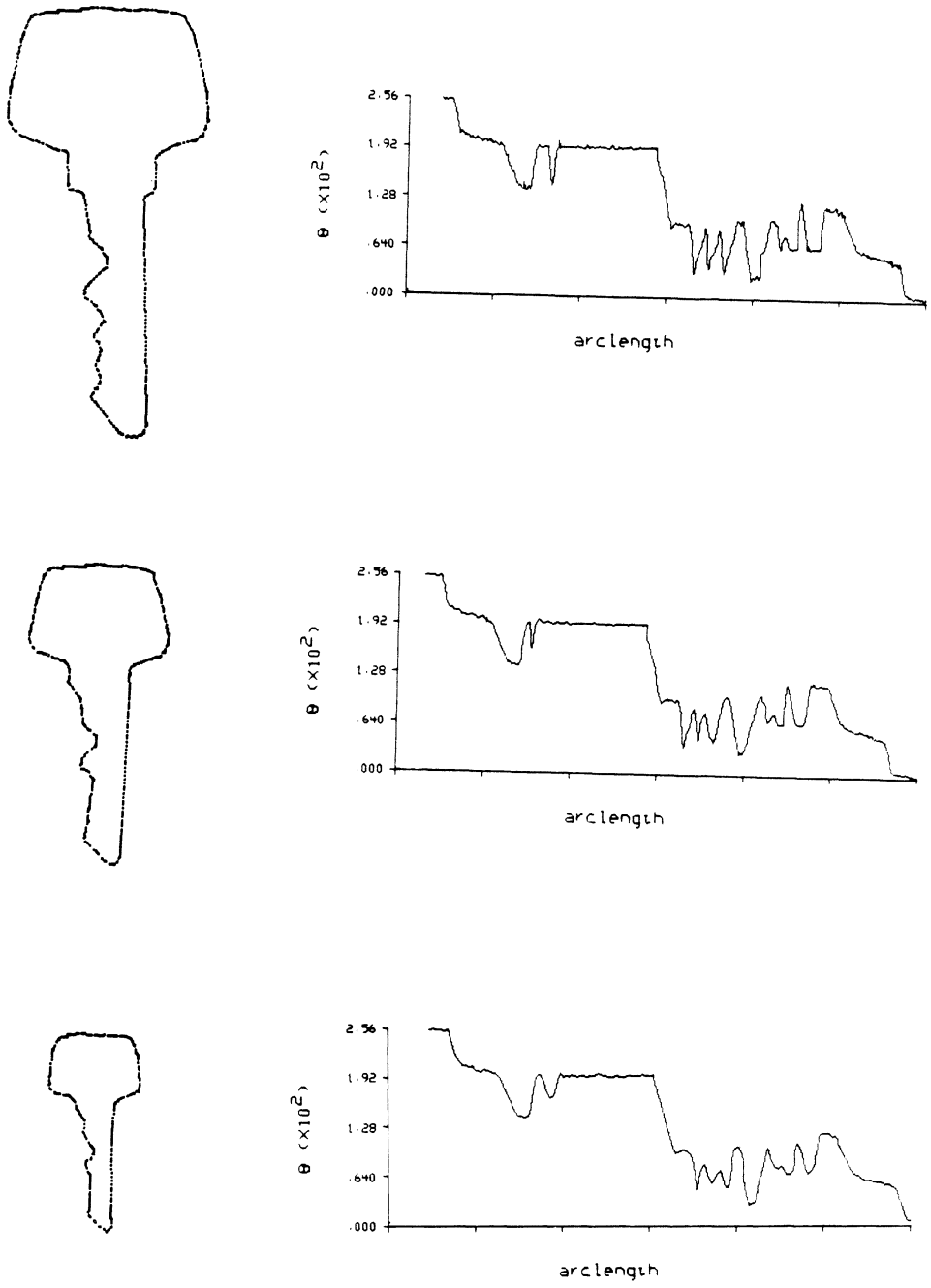

Fig. 5.6. θ–s representation of key at different scales.

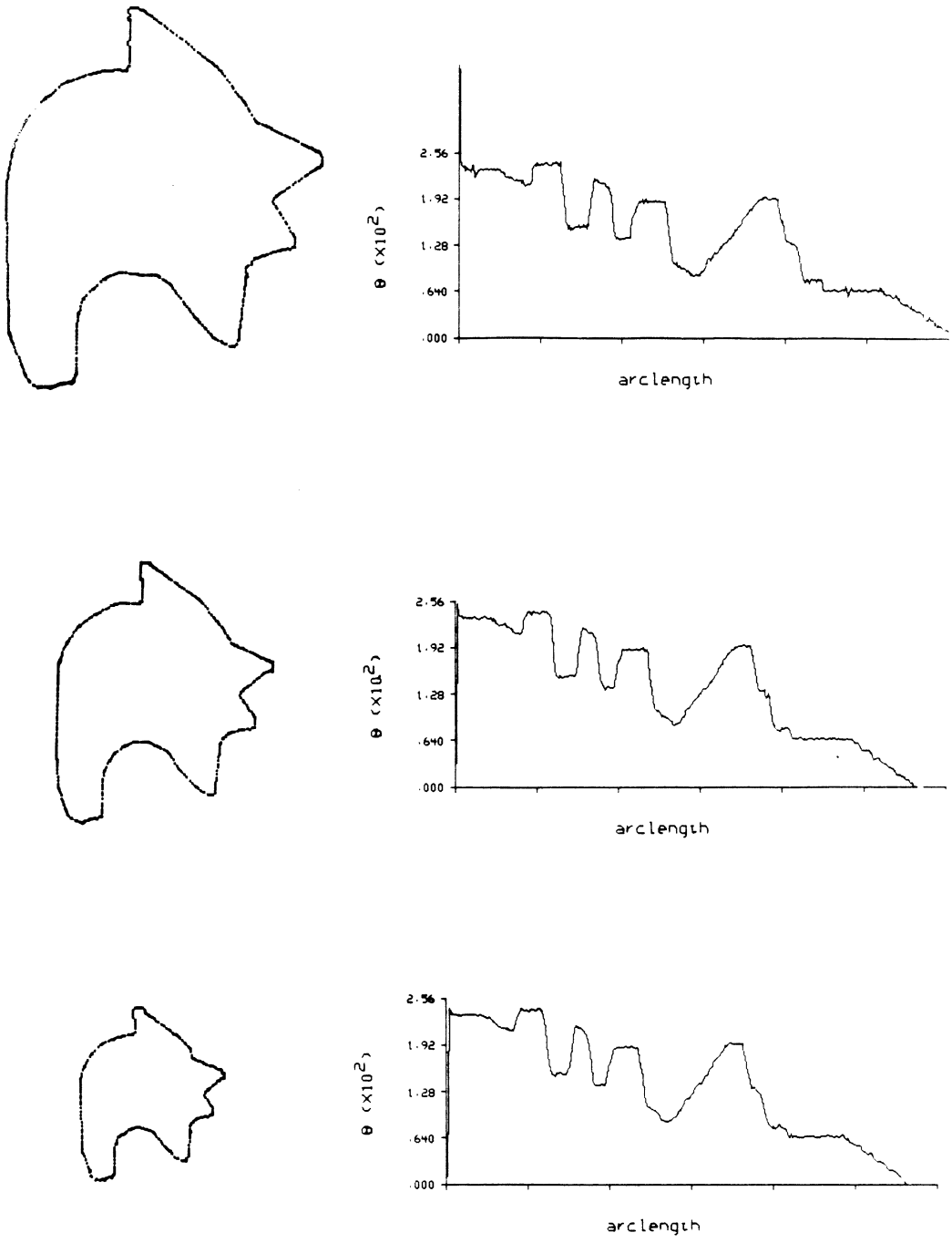

Fig. 5.7. θ–s representation of door lock part no. 2 at different scales.

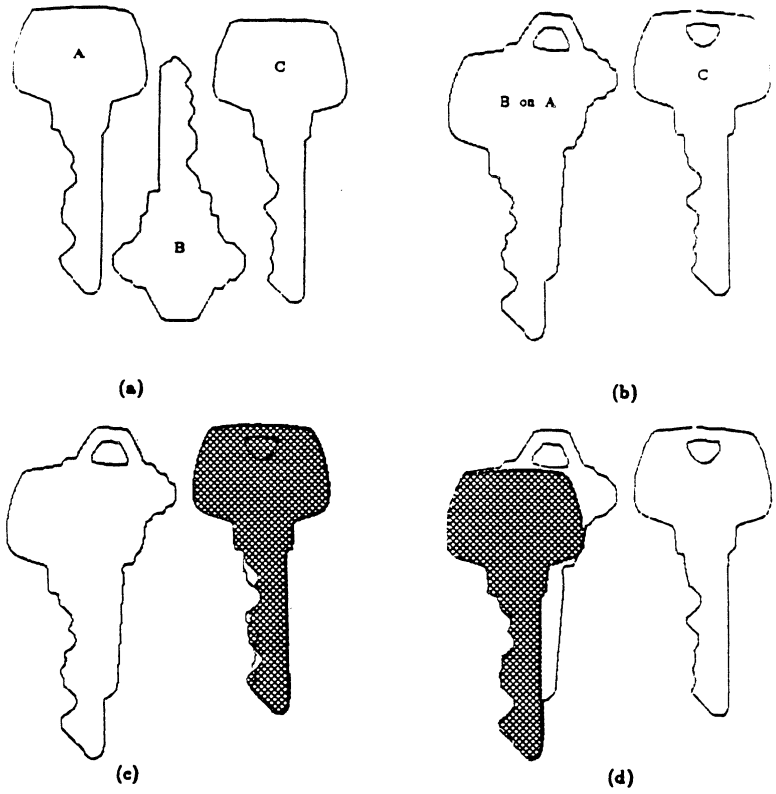

Fig. 5.8. Experiment with overlapping keys.

occluding it. The normal GHT, because it places equal emphasis on every edge point, incorrectly recognizes key C as key B, while our algorithm, because it weights segments of the key that distinguish it from other keys, correctly recognizes key B (see Fig. 5.8d). Additional experiments in applying the GHT to the image of Fig. 5.4b for set of nine parts yielded a false pose for several of the parts.

J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing Partially Hidden Objects." From our work with weighted segments, it became clear that for many images only a few visible segments of a part are needed to correctly locate a part [TMV85b]. With this in mind, we developed an algorithm that automatically selected pairs of part-model segments that could uniquely identified the part. This idea was expanded into our present approach.

The method of determining the unique configurations of a part-model was similar to the approach presented in Sec. 4.5. The recognition strategy, however, was somewhat different.

Instead of matching segments of the part-model to the image as we currently do, the reverse was done. Segments of the image were taken from sequential locations in the image boundary and compared to the segments of the part-models for parts that were to be located. In this sense the approach was *data driven.*

If an image segment matched a part-model segment, the pose of the part (determined in the same way as discussed above) at which the segments matched was stored in a table. The coordinates of the pose, (θ, u, v), were used as a key into the table. As more segments of the image were matched to the part-model the table was checked to determine if two segments stored in the table from the same part had the same pose. If they did and if the two segments formed a unique configuration of the part-model for the part, the part was assumed to be located. The pose of the two segments was taken to be the correct pose of a part in the image.

Figure 5.9 shows the results of using this algorithm to locate a key and an industrial part in the image of Fig. 5.4. As the parts were located, the boundary of the part-model was overlaid on the image. The repeated location of the parts through the use of different unique configurations of segments is indicated by the thickness of the lines shown in the overlays. More importantly, no false poses were determined for the parts.

One additional note: in this paper we defined saliency of a segment as the number of unique configurations of the part-model to which a segment belonged. This differs from our current definition where saliency is a precomputed estimate of the probability of occurrence of a part-model configuration (see Sec. 4.4).

T. N. Mudge, J. L. Turney, R. A. Volz, "Automatic Generation of Salient Features for the Recognition of Partially Occluded Parts." In this paper we presented a more formal definition of saliency and derived the results for saliency that appear in Sec. 4.4 [MTV85]. In addition, we altered the strategy of locating parts to its present form (see Sec. 4.6). We also introduced the use of critical points to speed up the algorithm (see Sec. 4.3.1). We performed experiments to determine the usefulness of the various properties of the θ–s representation discussed in Sec. 4.3.1. This led to experiments in locating flipped parts. In this paper we

Fig. 5.9. Locating parts from unique pairs of segments.

presented some initial results in locating the part (which we refer to as door lock part 1) of Fig. 5.10 in a bin of identical parts. These results along with others are reported in the next chapter.

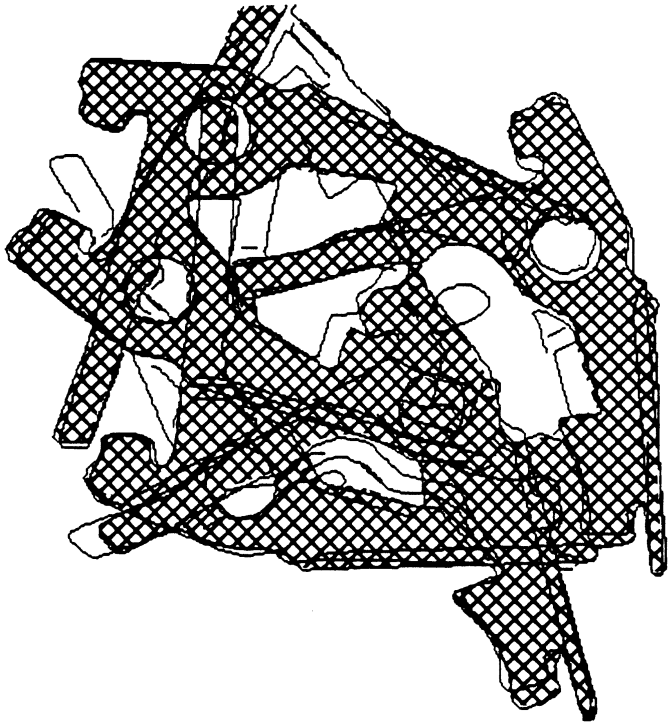

Fig. 5.10. Five no. 1 door lock parts located.

CHAPTER VI

RESULTS AND DETAILS OF IMPLEMENTATION

6.1. Introduction

In this chapter we will present the results obtained from of our model-based POP recognition algorithm. In addition, we will present some of the details of the implementation that we have postponed discussing until now.

6.2. Results

6.2.1. Identical Parts

In order to illustrate our approach, we first applied our algorithm to a set of images of a bin of identical parts, a common mode of presentation for parts in industry. There were seven identical parts in each bin. The part was of stamped metal and was reasonably flat (the smallest dimension of the part was 15% of the largest dimension). This part is door lock part 1 used elsewhere in the thesis (see Fig. 4.1).

Figure 6.1 shows an example of a typical set of image edge boundaries obtained for a bin of no. 1 parts. Edge boundaries were obtained using a Canny [Can83] edge detector (see implementation details later in this chapter). Parts were painted with a non-reflective coating to allow us to test our algorithm without the additional complication of obtaining good edge boundaries. (Unpainted reflective parts are treated later in this chapter.)

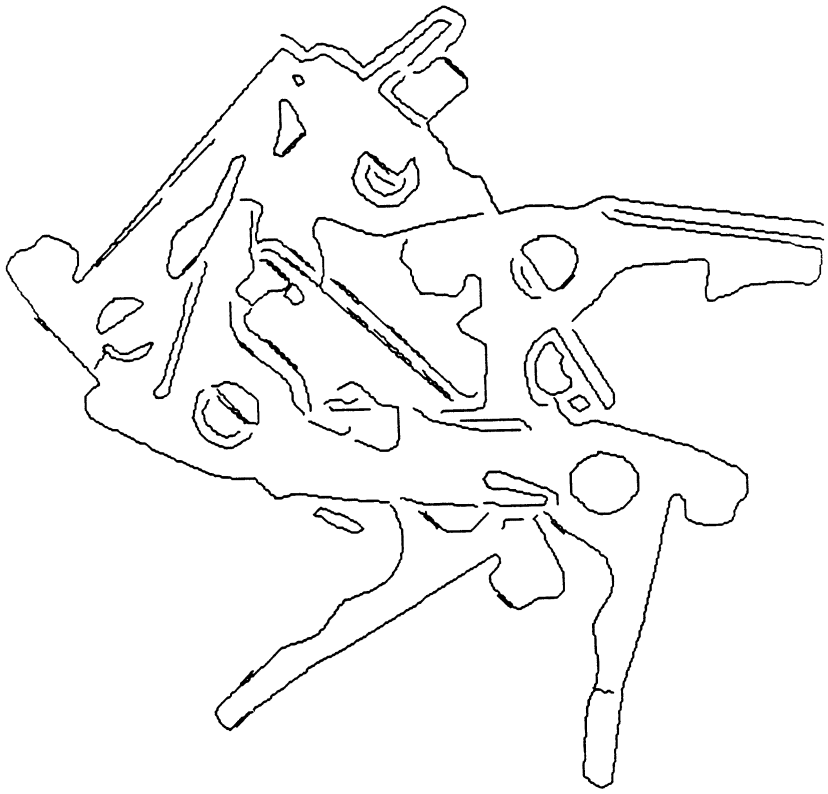

Fig. 6.1. Edge boundaries extracted from a bin of no. 1 parts.

There were fourteen 256×256 images of the bin of parts in the test set. There were, therefore, 96 instances of parts in the set of images. Figures 6.2-3 show the results obtained for locating the door lock part in one of the images. (For the test image shown in Fig. 3.2, we were able to find 5 of the 7 parts; see Fig. 5.10). We had an overall success rate of approximately 5.8 parts found per image (83%). Only one part in one image (1%) was falsely reported. Both the obverse and reverse sides of the parts were located from the part-model of the obverse side. Thus, in essence, this experiment tested the usefulness of our algorithm in recognizing both normal and flipped parts from one side of the part. (Slightly better results would be expected if both sides of the part were used in locating the parts—the parts were not completely flat and the obverse and reverse sides are not true mirror images of one another.)

The plot of Fig. 6.4 conveys more information than simply the percent of parts recognized. It shows the percentage recognition versus the percentage of a part's boundary that is exposed in

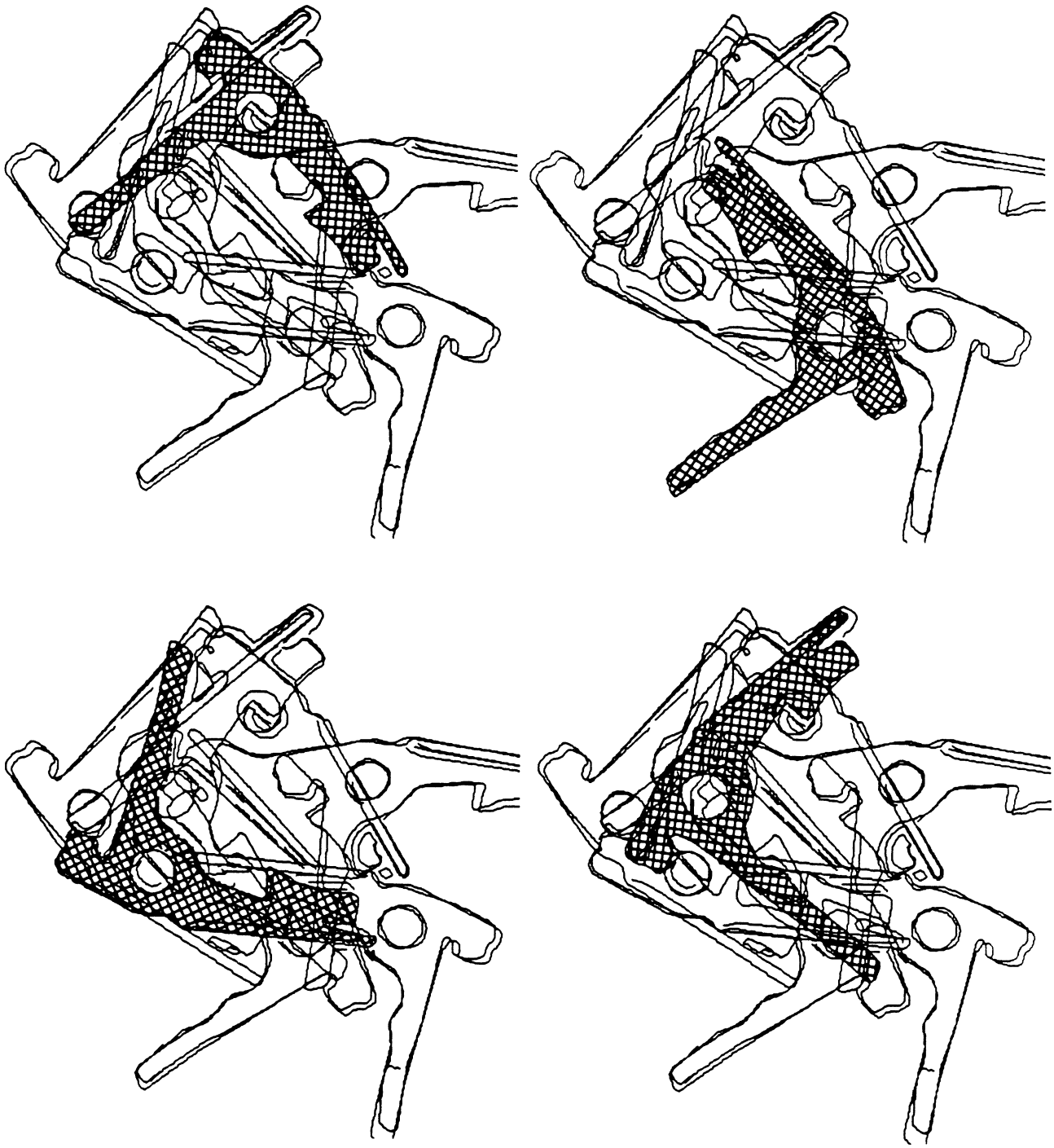

Fig. 6.2. Parts located.

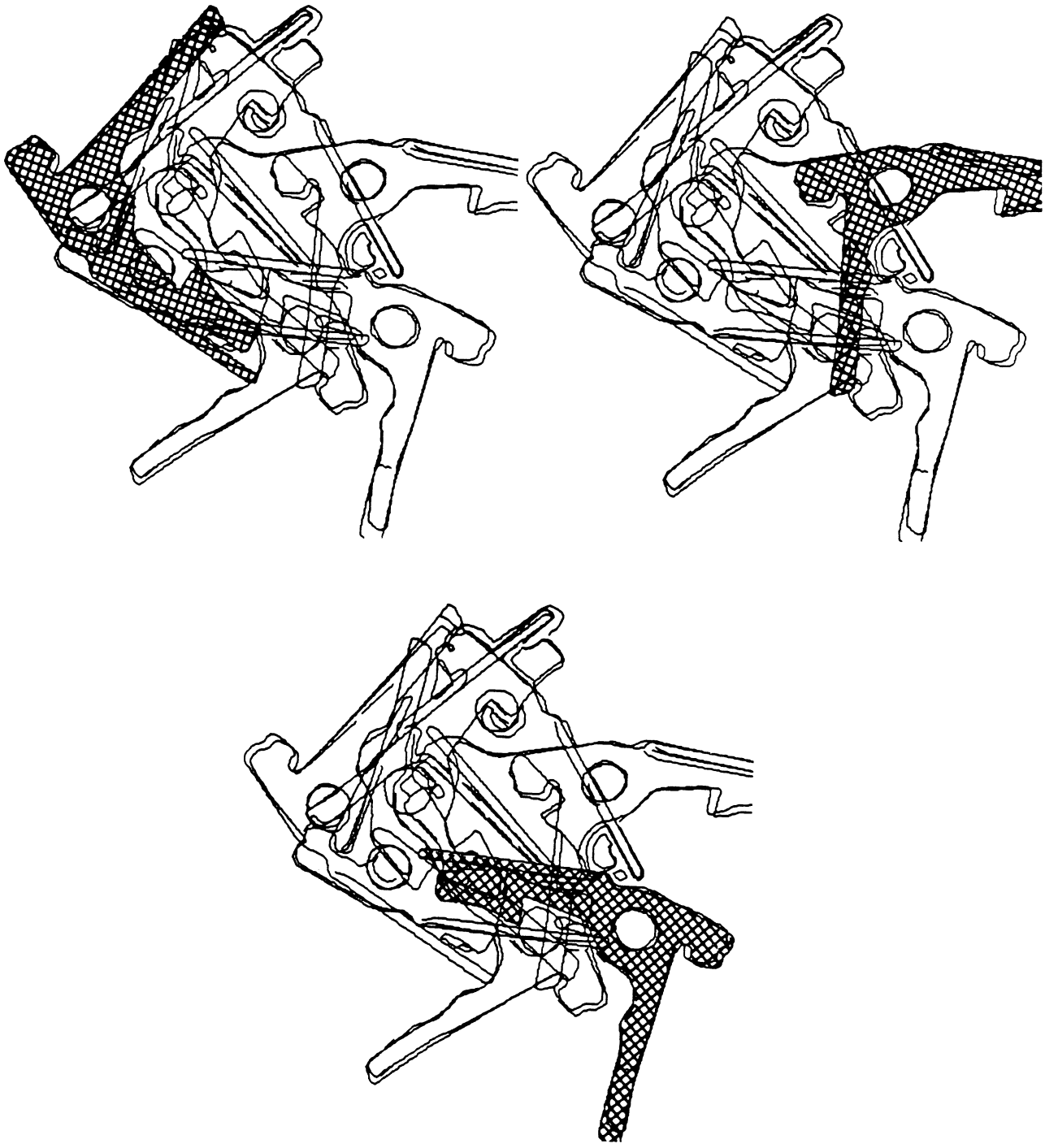

Fig. 6.3. More parts located.

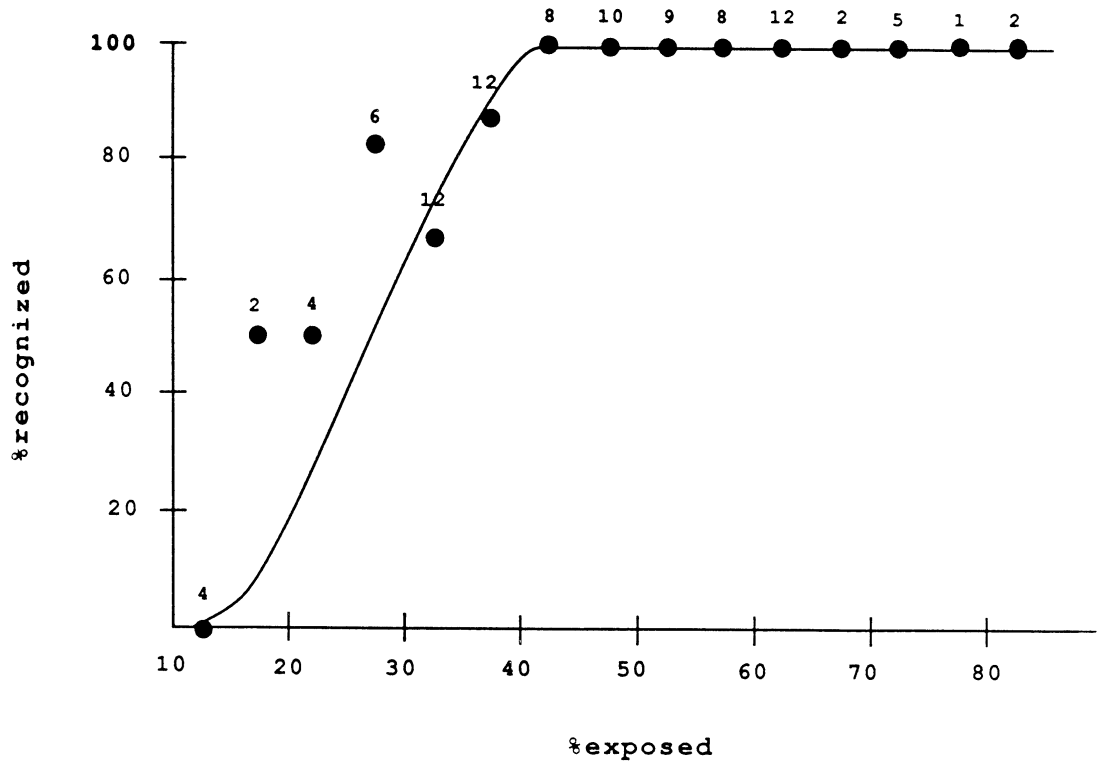

Fig. 6.4. Percentage of parts located v. percentage of part boundary exposed.

the image. We have loosely sketched a curve suggesting a possible relation between the number of parts found versus the amount of exposed boundary of a part. One would expect that as less of the part was exposed, the probability of recognizing the part would decrease. This behaviour is evident in the curve of Fig. 6.4. One can see that we obtained reasonable results even when as little as one third of a part's boundary was exposed. Of course, these results are strongly correlated to the choice of part but, in any case, reflect the robustness of the approach for a typical industrial part.

The numbers in the graph adjacent to each point show the number of instances of parts in the images supporting the data. For example, there were 12 parts with between 35% and 40% of their boundary exposed, and 83% of these parts (10) were located by our algorithm.

The percentage of boundaries exposed were obtained by manually fitting a template of the part with tick marks evenly spaced along the template. The template was fit to the image boun-

daries and the tick marks corresponded to the image boundary were counted. The fraction of tick marks that corresponded to exposed segments of the image boundary gave the percentage of the image boundary exposed. This technique also allowed us to determine the percentage of the boundary exposed for parts that were *not* found as well as for those that were found.

Parts were tilted up to approximately 30 degrees. Although the algorithm was not explicitly designed to locate tilted parts, we were able to locate many of these part by adjusting the threshold on segment matching to accommodate the distortion in the θ–s representation of the image boundary segments resulting from tilt.

For part-models of n segments and image boundaries of m segments, the worst case complexity for locating a part would be $O(n \times m)$. In practice the worst case is seldom realized. With the use of critical points to speed up recognition (see Sec. 4.3.1) very little of the image boundary is actually matched before a part is found. In addition, once a part is found in the image, the boundary corresponding to the part is eliminated from further matching. If p parts were separately sought, the complexity would be $O(p \times n \times m)$.

Recognition times were on the order of 4 to 5 seconds per part on an Apollo 660 node. This value, however, does not include preprocessing such as edge detection and edge linking.

6.2.2. Non-identical Parts

We next tested the algorithm on images of overlapping puzzle parts. These parts, shown in Figs. 6.5, were ten typical jigsaw puzzle parts. Some of the parts that we selected were very similar in appearance and had many common features. They were chosen in order to tax our algorithm. Because of the similarity of the parts, fewer configurations were unique to any one part, and, as a consequence, our algorithm needed, on average, a higher percentage of the boundary of a part exposed in order to locate the part. In addition, it was necessary to use a smaller threshold on segment matching during training and recognition so that our algorithm would be able to better distinguish between similar configurations. However, this gave the algorithm less latitude

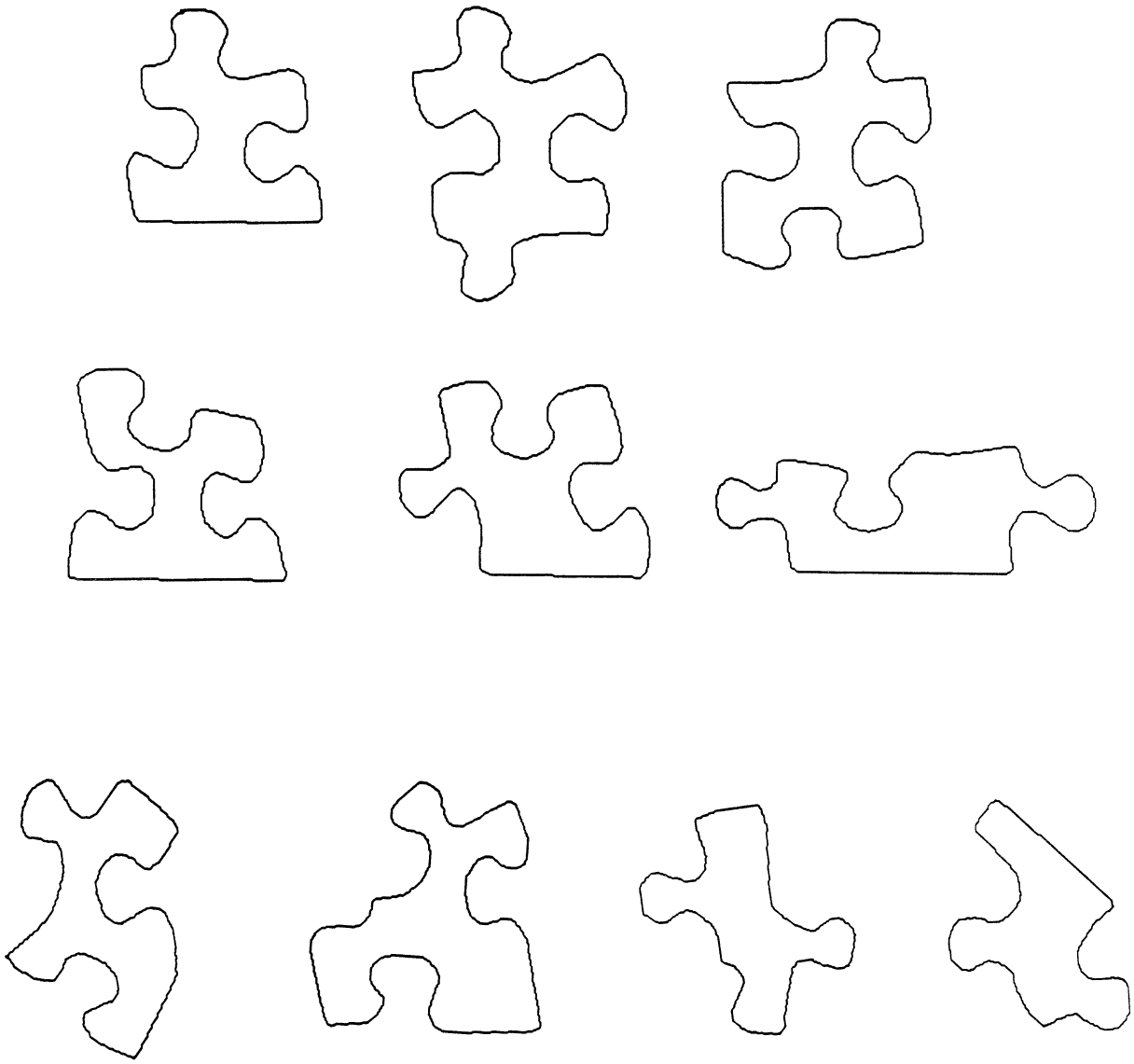

Fig. 6.5. Jigsaw puzzle parts.

in recognizing tilted parts, and for this reason the algorithm failed to locate several tilted parts

that had a large fraction of their boundary exposed.

Figure 6.6 shows the edge boundaries taken from a typical image. Parallel edges were due to shadows. Shadow edges had the opposite direction as normal edges and thus were not often incorrectly interpreted as edges of the parts. There were eight images of this type tested, or 80 instances of parts involved in the tests. Figures 6.7-8 show some of the results obtained for the image of Fig. 6.6. Overall, we had a success rate of approximately 6 parts found per image (75%).

The plot in Fig. 6.9 summarizes the results. It shows the percent of the puzzle parts recognized in the images versus the percent of a part's boundary that is exposed. The percentages of boundaries exposed were obtained in the same way as in the first experiment by manually fitting part-model templates to the image boundaries. The numbers in the graph adjacent to each point

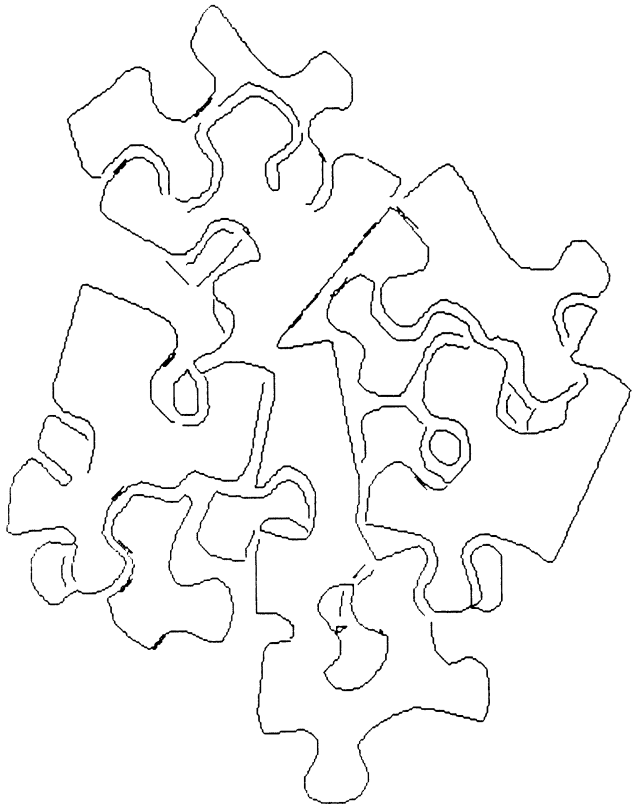

Fig. 6.6. Occluded puzzle parts.

show the number of instances of parts in the images supporting the data.

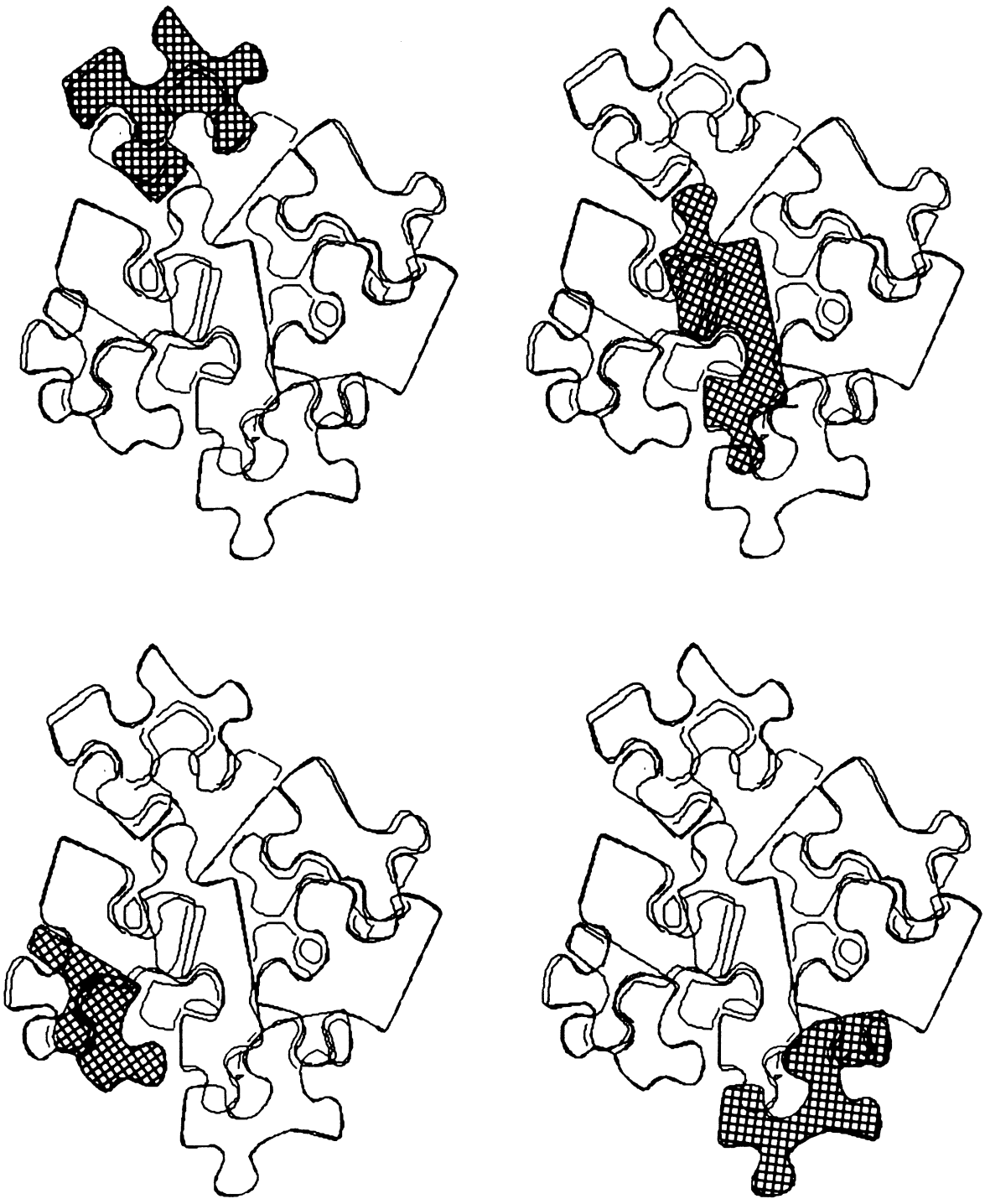

Fig. 6.7. Puzzle parts located.

We have again sketched a curve suggesting a relation between the number of parts found

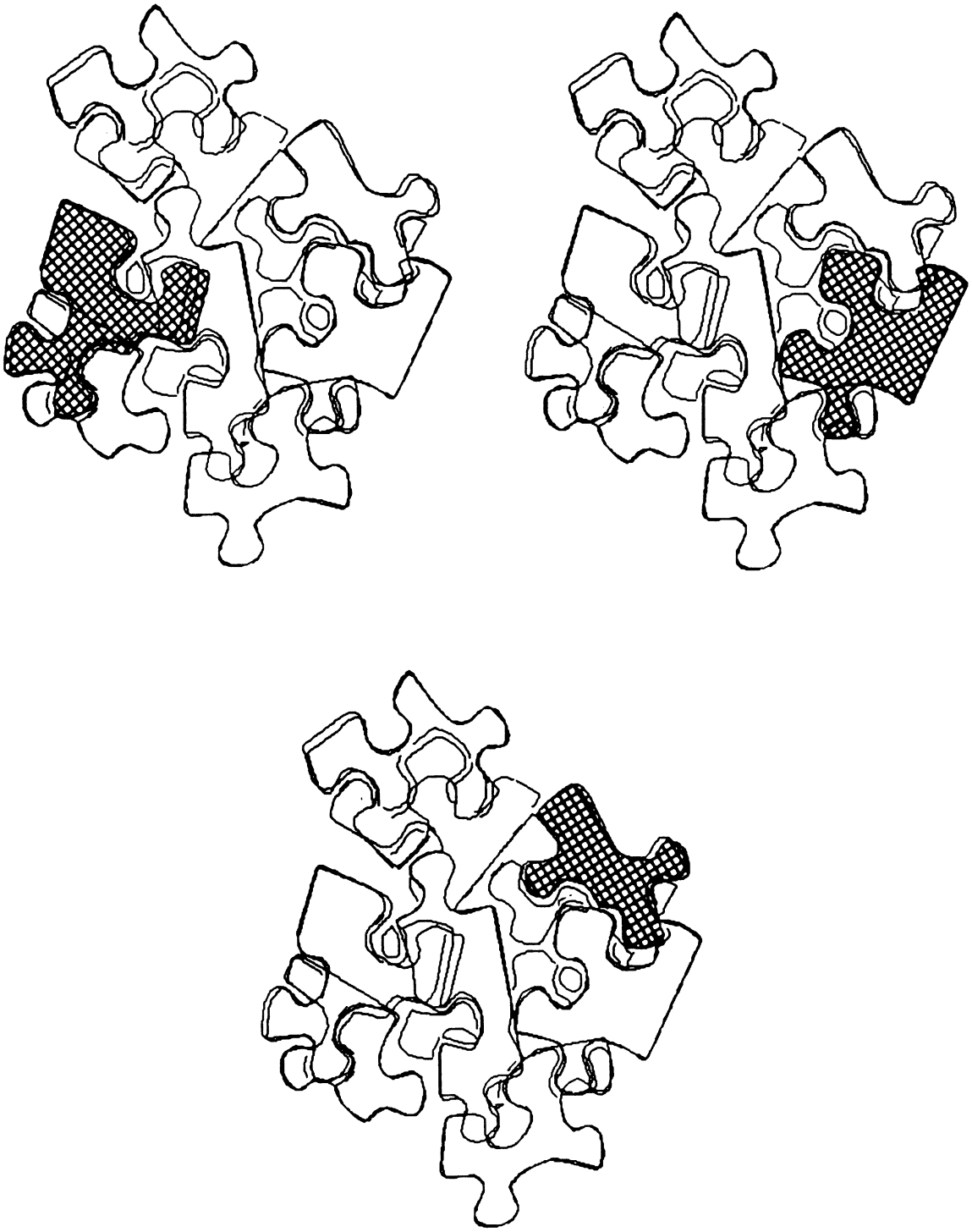

Fig. 6.8. More puzzle parts located.

and the percentage of their boundary that was exposed. One can see that considerable more exposure of the boundary on the average was necessary than in the previous experiment with the industrial part.

Recognition times were on the order of 7 to 8 seconds per part on an Apollo 660 node. This value does not include preprocessing such as edge detection and edge linking.

6.2.3. Scaled Parts

To test our approach for recognizing scaled parts, we selected the part whose boundary is shown in Fig. 6.10a. This part is also taken from the mechanism for a car's door lock, and is door lock part no. 2 from Fig. 5.4a. We obtained three images of fragments of the part at scales of roughly 60%, 40%, and 20% of the initial scale of the part.

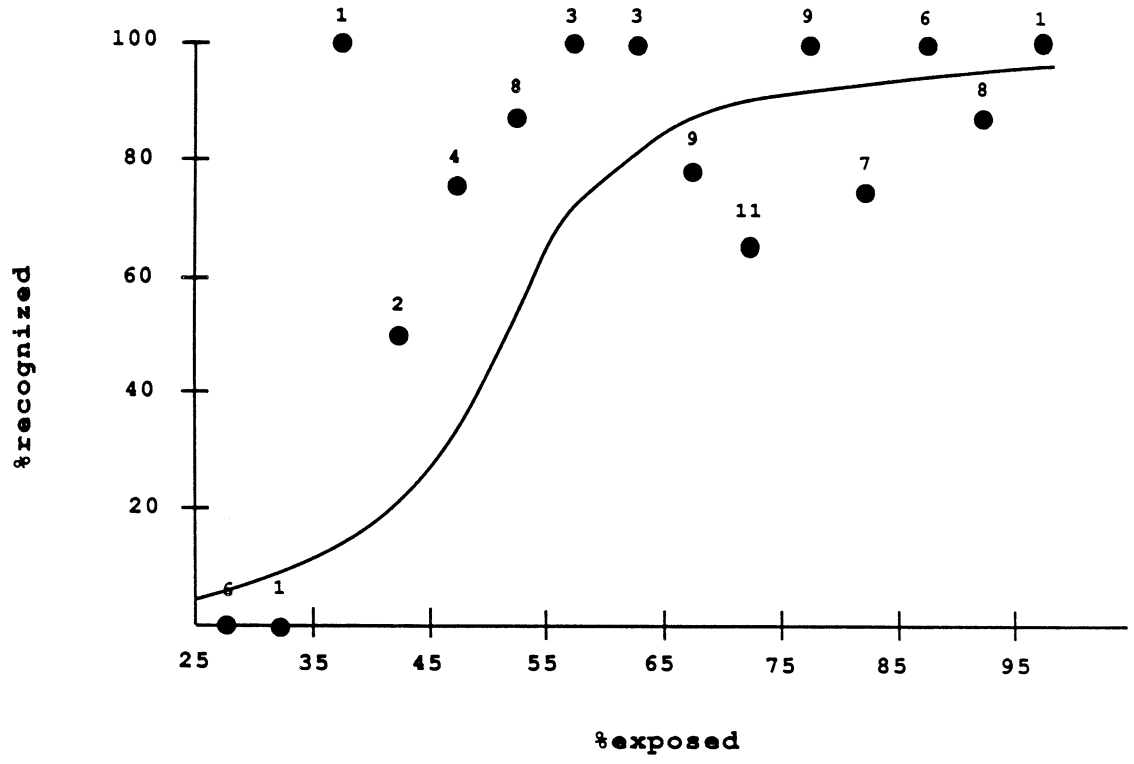

Fig. 6.9. Percentage of puzzle parts located v. percentage of part boundary exposed.

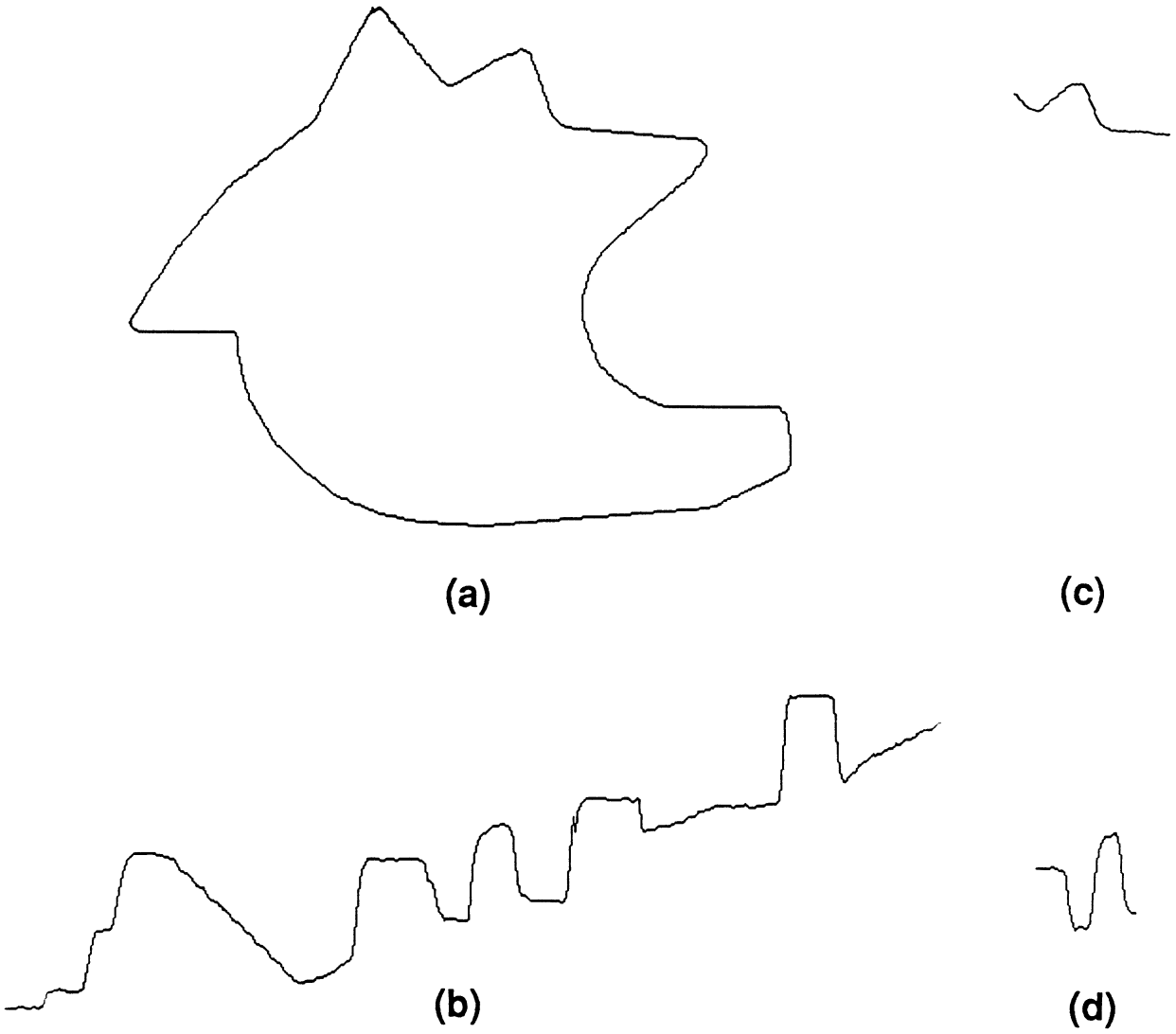

Fig. 6.10. Cartesian and θ–s representation of door lock part no. 2.

Figure 6.10c shows a fragment of the boundary of the part at roughly a 60% reduction. Figure 6.10b shows the θ–s representation of the full-scaled part and Fig. 6.10d shows the θ–s representation of the 60% scaled part fragment. Clearly, the two θ–s representations are identical except for a scale factor along the s axis.

The critical points of door lock part no. 2 are shown in Fig. 6.11a and those of the 60% fragment are shown in Fig. 6.11b. Correspondence between locations of critical points in the contours is evident. However, not all of the critical points are the same at the two scales. As the part is scaled down, the curvature at some boundary locations increases and new critical points are created. On the other hand, as the part is scaled down, the resolution of the boundary is decreased. and some critical points are lost.

The scale for the fragments of door lock part 2 was found (see Sec. 4.3) by matching pairwise combinations of critical points of the part to pairwise combinations of critical points of the fragments. For each match a scale was hypothesized, and the θ–s representation of the part was down-scaled by this factor and fit to the θ–s representation of the fragment. Figure 6.12 shows the results of applying our scaling algorithm to the θ–s representation of part 2, overlaid on top

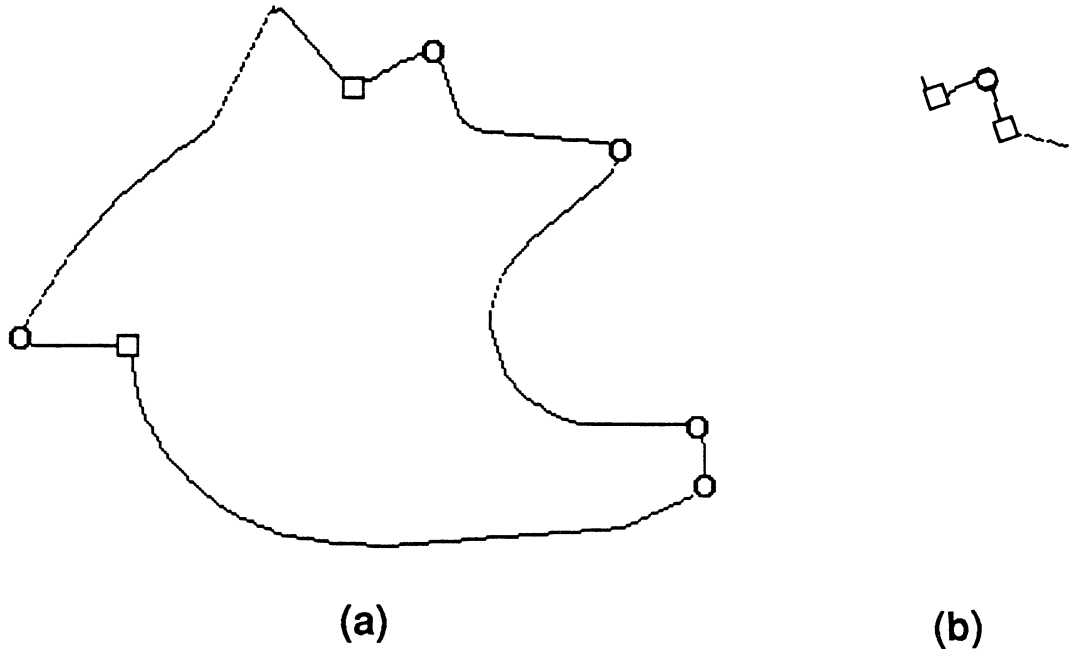

Fig. 6.11. Critical points of door lock part 2 and fragment.

of the θ–s representation of the 60% scaled part fragment. The two θ–s representations fit sufficiently well that it is difficult to distinguish between the two curves.

The figures on the left in Figs. 6.13a-c show the cartesian contours of the 60%, 40%, and 20% scaled part fragments while Figs. 6.11d-f show the results of applying our algorithm to locating the part 2 in the scaled images. The scales determined by the algorithm of Sec. 4.3.1 were within 5% of their correct values.

6.2.4. Reflective Parts

For this experiment, the non-reflective coating previous placed on the several of the door lock parts was removed. These parts are highly reflective and as a consequence the boundaries of the parts have different contrasts to the backgrounds depending on lighting and viewing

Fig. 6.12. Overlaying scaled θ–s representations.

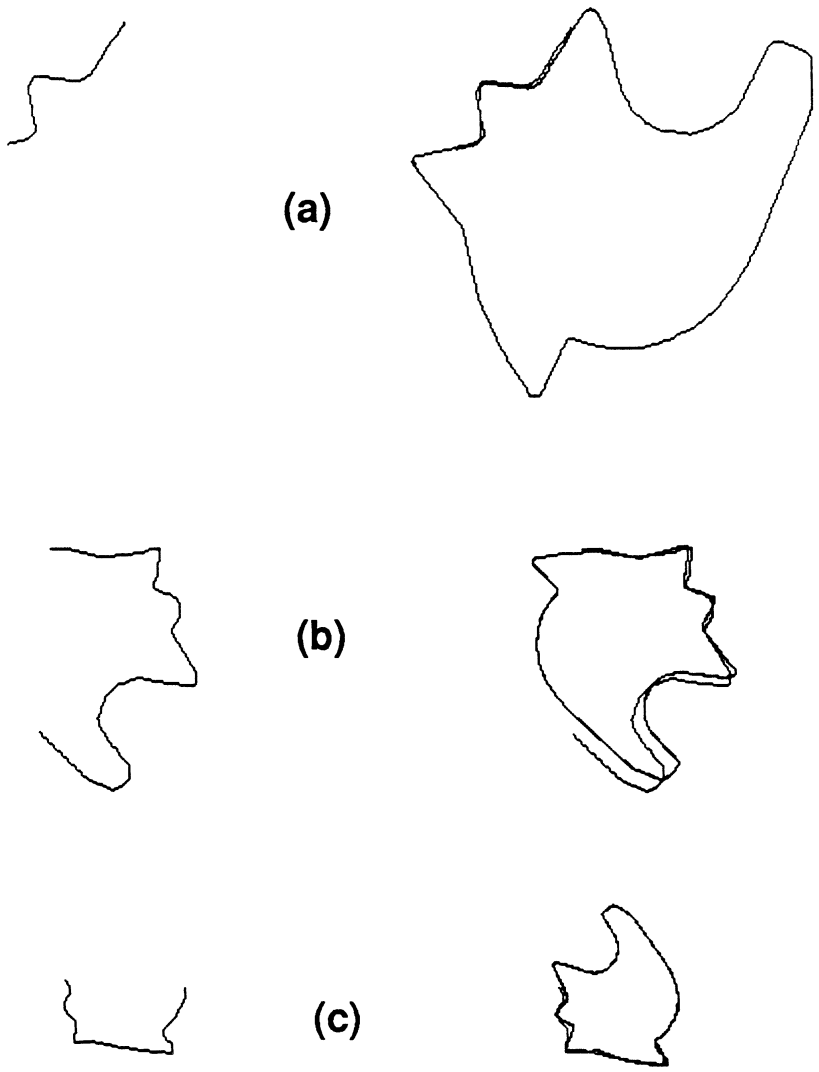

Fig. 6.13. Scaled parts found.

conditions. Figure 6.14a shows the edge boundary of the parts obtained when most of the surface

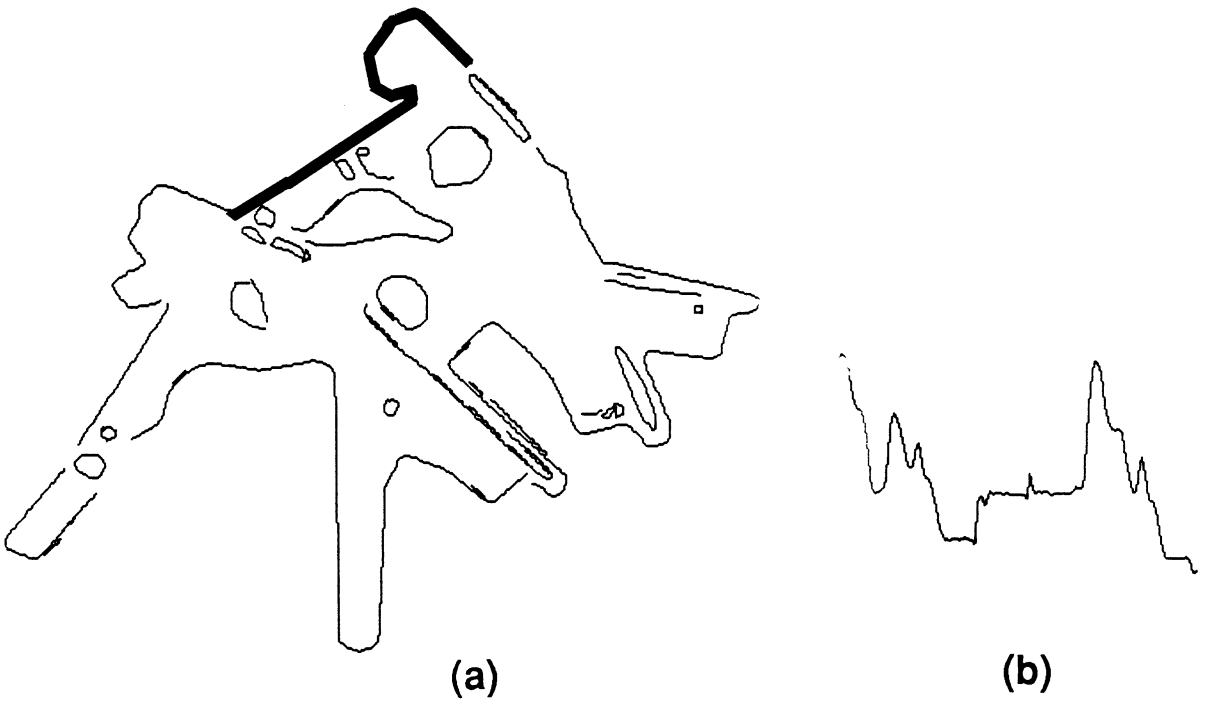

Fig. 6.14. Image boundary of reflective part.

area of the parts was darker than the background. Figure 6.14b shows the θ–s representation in the neighborhood of the highlighted segment of the image boundary. The part-model boundary of the door lock part was taken from a image in which the part was lighter than the background. Figure 4.5 shows the θ–s representation of the part-model, a segment of which corresponds to the segment of Fig. 6.14a. One can see by comparing the two θ–s representations that they are locally mirror images of one another. This agrees with our discussion in Sec. 4.3.1.

The critical points for the image boundary of the reflective parts are of opposite curvature (see Fig. 6.15a—squares represent minima and circles represent maxima in curvature) to those of the part-model (see Fig. 4.6). This reversal occurs because the curvature of the critical points is calculated under the convention that the brighter surface is on the left when tracing the boundary of a part in increasing arclength. We have previously assumed that the brighter surface belonged to the part. This assumption treats the background of Fig. 6.12a as if it were the part and

determines the critical points accordingly. Therefore, in images in which contrast reversal can occur, in order to use critical points to speed up segment matching, segments containing critical points must be compared to segments containing critical points of both positive and negative curvature.

Figure 6.16 shows the results of applying our modified algorithm to locate contrast reversed segments resulting from reflective parts. The three parts (shown cross hatched in Fig. 6.16) were all successfully located from a part-model of opposite contrast. Recognition times were again between 4 to 5 seconds.

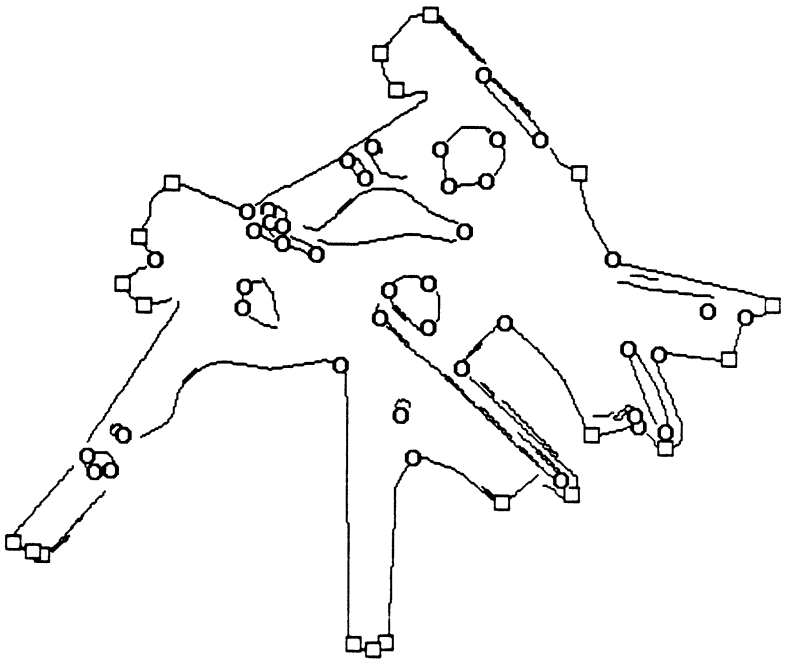

Fig. 6.15. Critical points of the reflective part.

6.3. Details of Implementation

Our algorithm consists of three components: *preprocessing*, *training*, and *recognition*.

6.3.1. Preprocessing

Preprocessing is used for building part-models and for converting images into boundary representations. Preprocessing consists of edge detection, edge linking, contour normalization, and critical point extraction.

Edge detection and linking. To detect edges we use a Canny edge operator [Can83]. We compared edge images obtained using the Canny operator to those the obtained from difference of Gaussians, Frei and Chen, and Sobel operators [Bal82] and we have found the Canny edge operator to be superior for our application.

Starting with an arbitrary edge point, we iteratively link edge points into contours. At each iteration edge points which are located within a pixel of the head or tail of an existing contour and which have roughly the same slope angle as the head or tail are added to the contour. Contours are started by selecting edge points which have not yet been linked. If more than one edge point is a candidate for linking, the one closest in slope angle to the head (tail) is added. The output of linking are doubly linked lists of edge point records with fields containing the cartesian coordinate and the slope angles of the edge points in the contour.

Contour normalization. It is necessary to perform additional processing on the contours which we call normalization. A contour can be represented as a 2-dimensional cartesian space curve $(x(s), y(s))$ or as a 1-dimensional slope angle function $\theta(s)$, in both cases parameterized by the arclength along the contour. However, contours are not in a suitable form after edge linking. For instance, contours contain noise which tends to distort the values of x, y, and θ, and indirectly the values of the arclength, s. A more significant distortion arises from the necessity of sampling edge images on a grid. The distance between edge points is non-uniform: diagonally connected edge points are further apart than horizontally or vertically connected edge points. As

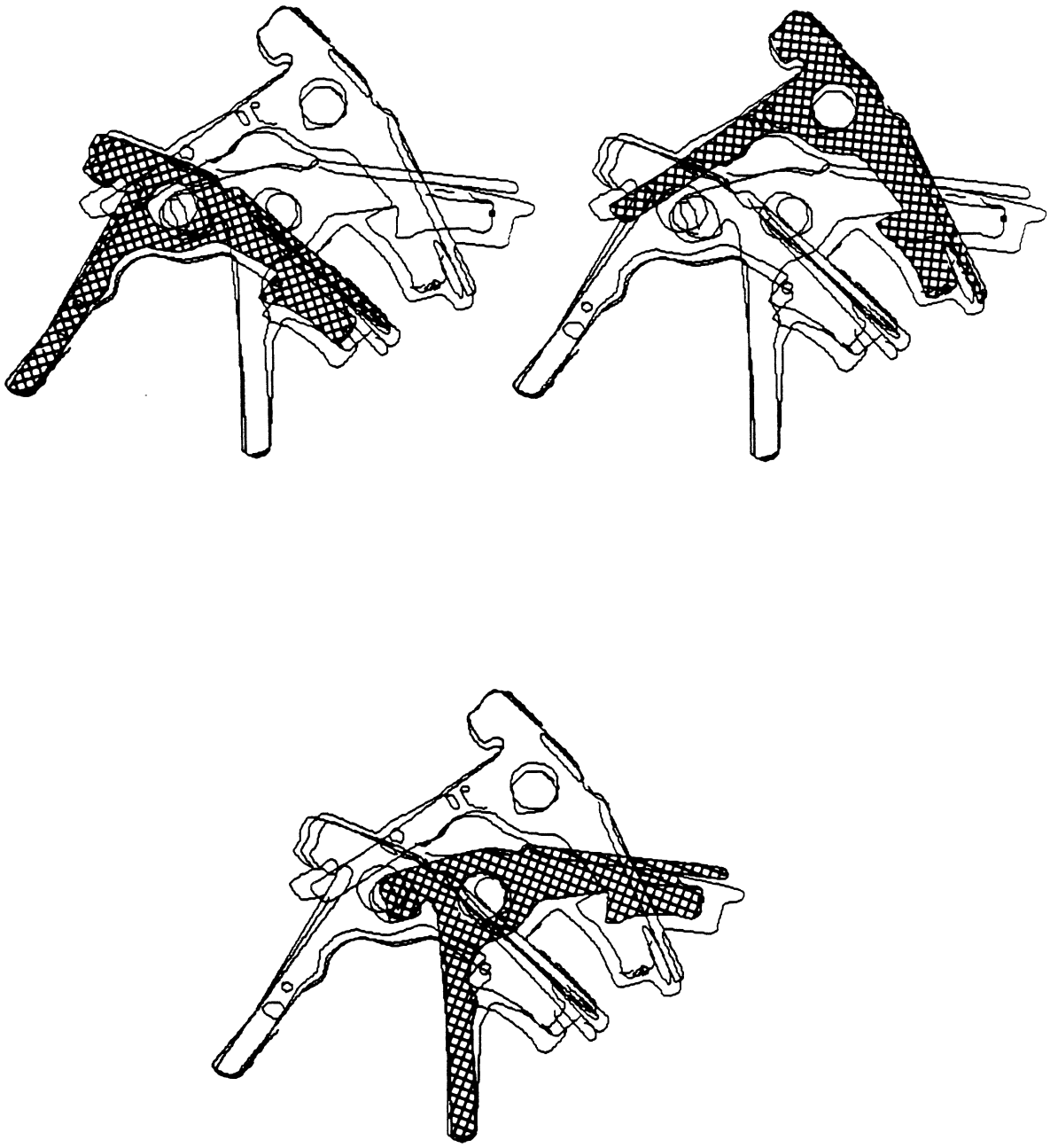

Fig. 6.16. Locating reflective parts.

a result, edge points in a contour are sampled at unequal arclengths.

To overcome the problem of noise and non-uniform sampling, we normalize the contours by smoothing them and resampled them at unit arclengths. In order to smooth a contour, we select a point on the contour as an initial *anchor* point. The contour is traced in the direction of increasing arclength until the average slope angle (averaged over 5 points) exceeds a fixed angle (10 degrees). The point at which this occurs is chosen as a second anchor point. A blending function, a Hermite cubic interpolating polynomial (see [FoV82]), which has the same slopes at its end points as the anchor points is fit to the two points. Let $\mathbf{p}_1$ and $\mathbf{p}_2$ represent position vectors of the first and second anchor points and let $\mathbf{r}_1$ and $\mathbf{r}_2$ represent the tangent vectors at these points. The curve is approximated by the vector $\mathbf{p}(t) = (x(t), y(t))$, where

$$\mathbf{p}(t) = \mathbf{p}_1 \times (2t^3 - 3t^2 + 1) + \mathbf{p}_2 \times (-2t^3 + 3t^2) + \mathbf{r}_1 \times (t^3 - 2t^2 + t) + \mathbf{r}_2 \times (t^3 - t^2),$$

and t is a parameter that ranges from 0 to 1. This polynomial is then sampled at small increments of t and the total arclength between the first and second anchor points is accumulated. The polynomial is resampled at approximately unit arclengths to obtain new x and y coordinate values for the edge points. The values of θ are found by interpolating the slope angle from the edge detection phase. Smoothing is continued by starting with the second anchor point as the initial anchor point and a new anchor point as the second anchor point. This continues until the entire contour has been smoothed.

We found our arclength resampling approach to be superior to that proposed by Dessimoz [Des78]. Dessimoz replaces the x and y coordinates for a edge point by the average x and y values of its neighbors on the contour. This approach distorts the sharp boundary features of a part by rounding them and making them less useful for recognition.

The final step in normalizing contours is to eliminate the artificial discontinuities from wrap-around that often occur in the slope angle function, $\theta(s)$. More precisely, the slope angle of an edge point is computed from the arctangent of the ratio of the edge strength in the y-direction to the the edge strength in the x-direction obtained from the edge detector. Unfortunately, the arctangent function returns values limited to the half open interval $(-\pi,\pi]$. Therefore, when the slope angle is outside of this interval the arctangent wraps the angle around so that the angle is

kept within this interval, resulting in an artificial discontinuity in the slope angle.

To overcome the problem of artificial discontinuities we trace the contour in increasing arclength and search for local discontinuities greater than π or less than $-\pi$. These discontinuities can only be artificial since they would require the contour to fold back on itself. As the contour is traced, an offset $(n_- - n_+)\times 2\pi$ is added to the slope angle at each point to correct for wrap-around. Initially n_- and n_+ are zero. Each time a negative discontinuity is found n_- is incremented by 1, and each time a positive discontinuity is found n_+ is incremented by 1. The result of adding the offset is that the slope angle function $\theta(s)$ is made continuous for the entire contour with the exception of the initial and final points of the contour. In a closed contour these two points will always have an artificial discontinuity of 2π, corresponding to tracing once around the contour. This discontinuity is handled as a special case in our code.

After normalization, the mean slope angle is calculated about each edge point by averaging the slope angle of edge points within a fixed arclength distance on each side of the point. This distance is taken to be half the length of a segment. The segment length is a design parameter. In our current implementation this length is selected as 32 pixels. The mean slope angle is stored in each edge point record and is used during training and recognition when comparing segments.

Extracting critical points. As was previously discussed critical points of a contour can be used to speed up segment matching by limiting the comparison of segments of the part-models that contain critical points to those of the image boundary with similar critical points. We have defined critical points as the local maxima and minima in the curvature of the contour with curvatures above a fixed threshold. Since curvature is the derivative of $\theta(s)$ with respect to s, the problem of locating critical points in $\theta(s)$ is identical to the problem of locating edge points in a 1-dimensional image. Critical points in the slope angle correspond to the strong edges in an image.

To locate critical points we use a 1-dimensional version of the Canny edge operator that we used in edge detection. We convolve the slope angle function of the contour, $\theta(s)$, with the derivative of a Gaussian to obtain the curvature, $\alpha(s)$. To determine a maxima or minima we

use a form of non-maximal suppression similar to that discussed in [Can83]. More precisely, the curvature at each edge point is compared to curvature values of its nearest neighbors on each side. If the curvature is greater than both neighbors or less than both neighbors we label it as a maxima or minima respectively. We threshold the values of the maxima and the minima to locate the strongest maxima and minima. These are selected as critical points.

6.3.2. Training

Training is used to determine the saliency of each configuration in a part-boundary. Section 4.5 discusses the details of training. In our current implementation only the configurations that are unique to a part-model, i.e., have saliency close to 1, are retained. Every segment, S_i, of the edge boundary is assign a Boolean vector. An element of the Boolean vector at index j is set *true* if the segments S_i and S_j of the part-model form a unique configuration for the part. Otherwise, the element is set *false*. The segments with the large number of true elements are the segments used first by the recognition algorithm to locate a part. However, in order to spread out the selection of these segments to different sectors of the boundary, segments are selected in round robin fashion from locations near the critical points on the boundary.

6.3.3. Recognition

Recognition has been discussed in Sec. 4.6. In the following we include some additional details of our implementation.

After a segment of the part-model has been located in an image, we transform the part-model boundary to the pose at which the segment matched the image boundary. We then attempt to find a second segment centered on one of the critical points of the transformed boundary that matches a segment centered on a corresponding critical point of the image boundary. If this fails, we trace the transformed part-model boundary and check for a second part-model segment that has roughly the same slope as a closely located image boundary segment. If this condition is met,

and if the second part-model segment forms a unique configuration with the first segment, then an attempt is made to match the second part-model segment to the image boundary segment. If it matches, the part is assumed to be found.

After a part is found, we mark the image boundary at uniform intervals (using the transformed part-model boundary as a template). This is done so that the boundary will not be interpreted as belonging to a different part. We also mark the critical points overlaid by the transformed part-model boundary so that they are not used again in locating a different part. As a consequence, as more parts are found, less of the image boundary is consider in matching.

Because some the parts in the standard POP problem have considerable tilt (up to 30 degrees) we perform a final fit on the parts to provide a more accurated estimate of their pose. The transformed part-model boundary is searched at uniformly spaced arclengths. The search is conducted perpendicular to the boundary on both sides for image boundary points with roughly the same slope. In our implementation, the image boundary is stored in a hash table and the search is performed in the table. If an image boundary point has the same slope as a point of the transformed part boundary they are assumed to correspond. Corresponding points are least squares fit in cartesian space to provide a refined fit of the part to the image. A transformation matrix containing six elements, four for the rotation and two for the translation, is computed from the least squares fit.

CHAPTER VII

CONCLUSIONS

The partially occluded parts (POP) problem is an important problem for industrial applications. A solution to the problem will provide a flexible vision component for many assembly and sorting tasks—tasks that are found everywhere in small parts and batch assembly. However, the general problem is, as mentioned before, very difficult to solve. Therefore, we have limited our study to a subproblem, the 2-dimensional POP problem. In this special case the contours of the parts can be approximated by planar contours. In addition, we have assumed that parts are not tilted much from their normal stable viewing position primarily in order to limit the amount of distortion permitted in the parts' contours due to the viewing angle.

Even with these limitations the problem is still difficult. When parts overlap and partially occlude one another, identifying which segment of a contour belongs to which part is a complicated puzzle. Contours are fragmented and segments of contours of different parts can merge to form segments that exist in no single part's contour. In order to solve the problem, we have chosen a boundary representation specifically for use in the 2-dimensional POP problem. This is the segmented boundary representation in which all possible segments of the boundary of the part are members of the part-model.

In a POP image, special features of the contour will not necessarily be visible, long line segments will not necessarily be visible, but some segments of the part's contour must visible if the part is to recognized. Our choice of a segmented boundary representation reflects this simple fact. In addition, segments are allowed to overlap to provide redundancy to the representation. In other words, if a segment of a part's contour is partially occluded it is likely that one of its neigh-

boring segment will be totally visible. By choosing overlapping segments we increase the chances for a segment to be found. We select a configuration of two segments as an identifying feature of part. We felt that one segment, unless it was very long, would not sufficiently characteristize the boundary of a part to locate the part, but that two segments in their relative pose would be enough.

Unlike the part's boundary, a configuration is not necessarily a unique entity that belongs to one part. By choosing configurations of segments as a basis for our part-model, we run the risk of incorrectly identifying a part from a configuration that may occur in more than one part. For this reason we needed to attach a saliency, a measure of uniqueness, to a configuration.

Through the use of a training stage, we obtained all the information necessary to compute the saliency of a part's configuration. Informally the saliency is just the inverse of the frequency with which a configuration occurs in the set of parts that are to appear in the image. A more formal definition can be couched in terms of noise distributions of the segments in the image. Unfortunately, reasonable models for these distributions are difficult to determine. We have adopted a simple model in order to get a rough estimate of saliency.

With the additional information provided by saliency, we can determine, prior to recognition, the configurations of a part that are good features, and the configurations that, when found, uniquely locate the part.

These two simple concepts of segmenting the boundary representation into overlapping fixed length segments and determining the saliency of configurations of boundary segments form the basis of this thesis.

In the future, we hope to extend our recognition approach to solve 3-dimensional POP problems. In this more complicated problem we will attempt to identify parts from their profiles. Since the number of profiles that may be generated by a 3-dimension part is large it may be necessary to restrict our part-model to segments centered around critical points on the profiles of the objects. However, we could again use highly salient configurations of these segments to uniquely identify the part.

BIBLIOGRAPHY

BIBLIOGRAPHY

[ABB73] A. P. Ambler, H.G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone, "A Versatile Computer-Controlled Assembly System," *Proceedings of the 3rd International Joint Conference on Artificial Intelligence*, pp. 298-307, August 1973.

[Att54] F. Attneave, "Some Informational Aspects of Visual Perception," *Psychological Review*, vol. 61, pp. 183-193, 1954.

[Bal81] D. H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111-122, 1981.

[Bal82] D. H. Ballard, in *Computer Vision*. Englewood Cliffs, N.J.: Prentice-Hall, 1982.

[BaS83] D. H. Ballard and D. Sabbah, "Viewer Independent Shape Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 6, pp. 653-659, November 1983.

[BaF85] B. Bamieh and R. J. P. De Figueiredo, "Efficient New Technique for Identification and 3D Attitude Determination of Space Objects from a Single Image," *IEEE Conference on Pattern Recognition and Image Processing*, pp. 67-71, 1985.

[Bha84] B. Bhanu and O. D. Faugeras, "Shape Matching of Two-Dimensional Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 2, pp. 137-156, March 1984.

[Bin71] T. O. Binford, "Visual Perception by Computer," *IEEE System Science and Cybernetics Conference*, March 1971.

[BlN78] H. Blum and R. N. Nagel, "Shape Description Using Weighted Symmetric Axis Features," *Pattern Recognition*, vol. 10, pp. 167-180, 1978.

[BoC84] R. C. Bolles and R. A. Cain, "Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method," in *Robot Vision*, A. Pugh, Ed, 1984.

[BRB83b] A. H. Bond, R. S. Brown, and C. R. Rowbury, "The Effect of the Environmental Variation Upon the Performance of a Second Generation Industrial Vision System," *Proceedings of the Society of Photo-optical Instrumentation Engineers*, November 1983.

[BrA84] M. Brady and H. Asada, "Smoothed Local Symmetries and Their Implementation," *International Journal of Robotics Research*, vol. 3, no. 3, pp. 36-61, 1984.

[Bro83] R. A. Brooks, "Model-Based Three-Dimensional Interpretations of Two-Dimensional Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, pp. 140-150, March 1983.

[Can83] J. F. Canny, "Finding Lines and Edges in Images," *Master's Thesis, MIT.*

[ChH82] J. K. Cheng and T. S. Huang, "Recognition of Curvilinear Objects by Matching Relational Structures," *IEEE Conference on Pattern Recognition and Image Processing Intelligence*, pp. 343-348, June 1982.

[CCL84] C. K. Cowan, D. M. Chelberg, and H. S. Lim, "ACRONYM Model Based Vision in the Intelligent Task Automation Project," *1st Conference on AIA*, pp. 176-183, December 1984.

[Des78] J. D. Dessimoz, "Visual Identification and Localization in a Multi-object environment by Contour Tracking and Curvature Description," *8th International Symposium on Industrial Robots*, May-June 1978.

[DKZ79] J. D. Dessimoz, M. Kunt, and J. M. Zurcher, "Recognition and Handling of Overlapping Industrial Parts," *9th International Symposium on Industrial Robots*, pp. 357-366, March 1979.

[Fis83] R. B. Fisher, "Using Surfaces and Object Models to Recognize Partially Obscured Objects," *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pp. 989-995, 1983.

[FoV82] J. D. Foley and A. Van Dam, in *Fundamentals of Interactive Computer Graphics* Addison-Wesley, pp. 516-519, 1982.

[Fre77] H. Freeman, "Shape Description Via the Use of Critical Points," *IEEE Conference on Pattern Recognition and Image Processing*, pp. 168-174, June 1977.

[Fu74] K. S. Fu, in *Syntatic Methods in Pattern Recognition.* New York/London: Academic Press, 1974.

[GaJ79] M. R. Garey and D. S. Johnson, in *Computers and Intractability.* New York: W. H. Freeman and Company, p. 202, 1979.

[GlA79] G. J. Gleason and G. J. Agin, "A Modular Vision System for Sensor-Controlled Manipulation and Inspection," *Proceedings of the 9th International Symposium on Industrial Robots*, pp. 57-70, 1979.

[GrL84] W.E. Grimson and T. Lozano-Perez, "Model-based Recognition and Localization from Sparse Range or Tactile Data," *International Journal of Robotics Research*, vol. 3, no. 3, 1984.

[Hae83] W. Haettich, "Recogniton of Overlapping Workpieces by Model-Directed Construction of Object Contours," in *Artificial Vision for Robots Applied to Workpiece Recognition*, I. Aleksander, Ed. New York: Chapman and Hall, pp. 77-93, 1983.

[Hu62] M. K. Hu, "Visual Pattern Recognition by Moment Invariants," *IRE Transactions on Information Theory*, vol. IT-8, pp. 179-187, February 1962.

[Joh74] D. S. Johnson, "Worst-Case Behavoir of Graph Coloring Algorithms," in *Proceedings of the Fifth Southeastern Conference on Combinatorics, Graph Theory and Computing*. Winnipeg: Utilitas Mathematica Publishing, 1974.

[Kan74] L. Kanal, "Patterns in Pattern Recognition: 1968-1974," *IEEE Transaction on Information Theory*, vol. IT-20, no. 6, pp. 697-722, November 1974.

[KMB83] R. B. Kelly, H. A. S. Martins, J. R. Birk, and J. D. Dessimoz, "Three Vision Algorithms for Acquiring Workpieces from Bins," *Proceedings of the IEEE*, vol. 71, no. 7, pp. 803-820, July 1983.

[KoK85] M. W. Koch and R. L. Kashyap, "A Vision System to Identify Occluded Industrial Parts," *IEEE Conference on Pattern Recognition and Image Processing*, pp. 55-60, 1985.

[Lip69] M. M. Lipschutz, in *Schaum's Outline of Theory and Problems of Differential Geometry*. New York: McGraw-Hill, p. 73, 1969.

[Mar84] D. H. Marimont, "A Representation for Image Curves," *Proceedings of the American Association for Artificial Intelligence*, 1984.

[Mat76] J. Mattill, "The Bin of Parts Problem and the Ice-Box Box Puzzle," *Technology Review*, vol. 78, no. 7, pp. 18-19, June 1976.

[McA77] J. W. McKee and J. K. Aggarwal, "Computer Recognition of Partial Views of Curved Objects," *IEEE Transactions on Computers*, vol. C-26, no. 8, pp. 790-800, August 1977.

[MeF75] P. M. Merlin and D. J. Farber, "A Parallel Mechanism for Detecting Curves in Pictures," *IEEE Transactions on Computers*, vol. C-24, no. 1, pp. 96-98, January 1975.

[MTV85] T. N. Mudge, J. L. Turney, and R. A. Volz, "Automatic Generation of Salient Features for the Recognition of Partially Occluded Parts," *Robotica (to appear)*.

[Nah69] N. E. Nahi, in *Estimation Theory and Applications*. New York: John Wiley & Sons, pp. 206-209, 1969.

[Neu78] B. Neuman, "Interpretation of Imperfect Object Contours For Identification and Tracking," *Proceedings of the 4th International Conference on Pattern Recognition*, pp. 691-693, 1978.

[Nil80] N. J. Nilsson, in *Principles of Artificial Intelligence.* Palo Alto, CA: Tioga Publishing Company, pp. 72-88, 1980.

[Pal81] S. E. Palmer, "Transformational Structure and Perceptual Organization," *Proceedings of the 3rd Annual Conference of the Cognitive Science Society,* pp. 41-49, August 1981.

[Pav77] T. Pavlidis, in *Structural Pattern Recognition.* New York: Springer, 1977.

[PaA79] T. Pavlidis and F. Ali, "A Hierarchical Syntactic Shape Analyzer," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-1, no. 1, pp. 2-9, January 1979.

[Per78] W. A. Perkins, "A Model-based Vision System for Industrial Parts," *IEEE Transactions on Computers,* vol. C-27, no. 2, pp. 126-143, February 1978.

[Per80] W. A. Perkins, "Simplified Model-based Part Locator," *Proceedings of the 5th International Conference on Pattern Recognition,* pp. 260-263, December 1980.

[RiB81] A. Riad and M. Briot, "Identification and Localization of Partially Observed Parts," *Proceedings of the 11th International Symposium on Industrial Robots,* pp. 145-150, 1981.

[RuB84] P. Rummel and W. Beutel, "Workpiece Recognition and Inspection by a Model-Based Scene Analysis System," *Pattern Recognition,* vol. 17, no. 1, pp. 141-148, 1984.

[Rut82] W. S. Rutkowski, "Recognition of Occluded Shapes Using Relaxation," *Computer Graphics and Image Processing,* vol. 19, pp. 111-128, 1982.

[Seg83] Jakub Segen, "Locating Randomly Oriented Objects from Partial View," *Proceedings of the Society of Photo-optical Instrumentation Engineers Cambridge Symposium on Optical and Electro-optical Engineering,* November 1983.

[TMV83a] J. L. Turney, T. N. Mudge, R. A. Volz, and M. D. Diamond, "Experiments in Occluded Parts Recognition Using the Generalized Hough Transform," *Proceedings of the Conference on Artificial Intelligence,* April 1983.

[TMV83b] J. L. Turney, T. N. Mudge, and R. A. Volz, "Experiments in Occluded Parts Recognition," *Proceedings of the Society of Photo-optical Instrumentation Engineers Cambridge Symposium on Optical and Electro-optical Engineering,* November 1983.

[TMV85a] J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing Partially Occluded Parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 410-421, July, 1985.

[TMV85b] J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing Partially Hidden Objects," *IEEE International Conference on Robotics and Automation,* pp. 48-54, March, 1985.