1988

A microcomputer-based vision system to recognize and locate partially occluded parts in binary and gray level images

Volker Peter Petersen
Iowa State University

A microcomputer-based vision system to recognize and locate partially occluded parts in binary and gray level images

Petersen, Volker Peter, Ph.D.

Iowa State University, 1988

A microcomputer-based vision system to recognize and locate partially occluded parts in binary and gray level images

by

Volker Peter Petersen

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major: Industrial Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1988

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This paper presents a microcomputer-based machine vision system to recognize and locate partially occluded parts in binary or gray level images. The recognition process is restricted to untilted, two-dimensional objects.

A new edge-tracking technique in conjunction with a straight-line approximation algorithm is used to identify the local features in an image. Corners and holes serve as local features. The local features identified in an image are matched against all the compatible features stored for the model parts. The algorithm computes, for all image and model features matches, a coordinate transformation that maps a model feature onto an image feature. A new clustering algorithm has been developed to identify consistent coordinate transformation clusters that serve as initial match hypotheses. A hypothesis verification process eliminates the match hypotheses that are not compatible with the image information.

The system performance was compared to a vision system restricted to recognize nonoverlapping parts. Both systems require the same hardware configuration and share the basic image processing routines.

STATEMENT OF PROBLEM

Introduction

In the past 20 years, work has been done at universities and independent research centers to develop computer vision systems that can deduce the three dimensional visual information present in the environment to meaningful data for a computer. Machine vision is the automatic acquisition and analysis of images to obtain desired data for interpreting a scene or controlling an activity. The main emphasis lies in the application of computer vision technology to control an activity, that is, a production process.

Machine vision adds a high level of flexibility to automation equipment. For example, the first robotics applications had to be planned very carefully to assure that the parts to be handled were at the right place at the right time. Robots with vision capabilities are making expensive fixture equipment obsolete [1], because the robot movements can be corrected based on the information from the vision system. Common machine vision applications are part counting, sorting, locating, safety, quality control, process control, and robot guidance.

The task of machine vision systems can be divided into two major ones [2]. The first task, called image processing, deals with image digitization, noise reduction, contrast enhancement, and the conversion of gray scale images into binary ones. The second task is a classification process that groups images into predetermined categories. This process is usually referred to as pattern recognition or image analysis. Figure 1 shows the information flow through a typical machine vision system.

Objectives of the Research

This research concentrated on the image analysis task and resulted in the development of a microcomputer-based machine vision system designed to recognize partially occluded objects. The motivation for this research stems from the work by Petersen and Even [3,4], which led to the development of a binary machine vision system. This research, as well as work by fellow researchers [5,6,7], proved the feasibility of low-cost, microcomputer-based machine vision systems with a price tag of $5,000 to $7,000.

Microcomputer-based machine vision systems evoke increasing interest as powerful micros become available. The reason can be seen primarily in the high costs of traditional machine vision equipment. A survey of commercial vision systems showed (in 1984) a price range from $10,000 to $120,000 [8]; the average price in 1986 is around $50,000 [9]. Congiliara [9] predicts a trend toward two pricing groups: one group for the more complex turnkey systems with an average price of $50,000 to $70,000 and another group for around $10,000 to $15,000 for relatively

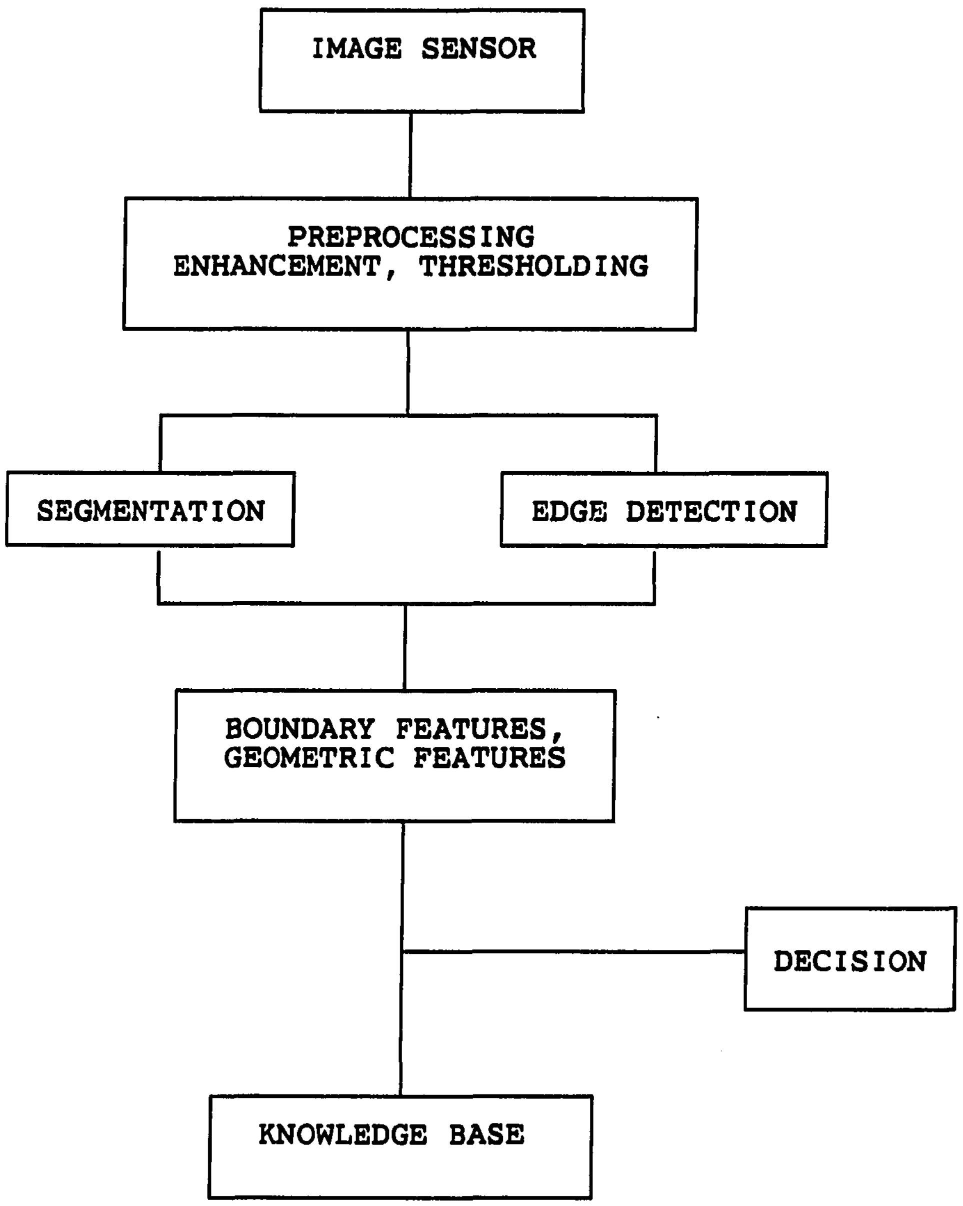


FIGURE 1. Information flow through a machine vision system

standard systems with minimal customization options. Low-cost machine vision systems can open this technology to applications for which a vision system is desirable but not cost feasible yet.

However, the scope of vision systems such as the ones developed by Petersen and Even or Weilert [3,4,6] is limited to handle only binary images showing the full outline of the part to be recognized. These restrictions resulted from the selection of a connectivity algorithm [10,11] that uses descriptive figures based on the geometry of the whole part to classify the objects under consideration. Such a vision system will fail to identify and locate an occluded object, because descriptors of part of the shape may not have any resemblance to the descriptors of the entire shape.

A relaxation of the above assumptions is necessary to open the scope of machine vision systems for a wider range of applications. Often parts are only partially visible due to overlap, low contrast, or noise in the image. A particular application in mind is the bin-of-parts problem which has received wide attention over the past years [12,13]. The challenge of this problem lies not only in recognizing and locating partially visible parts but also in dealing with a truly three dimensional task of combining machine vision and robotics technology to acquire randomly oriented parts from a tote bin. This problem has been reported as one of "the most difficult problems of automatic assembly" [13].

Furthermore, gray level image processing capabilities are needed for applications that cannot be dealt with by binary images. Examples are the processing of parts where interior features other than holes are important (see Fig.

2), where contrast between object and background is small or even variable, where objects have varying brightness, and where objects are jumbled together or perimeter information is otherwise obscured.

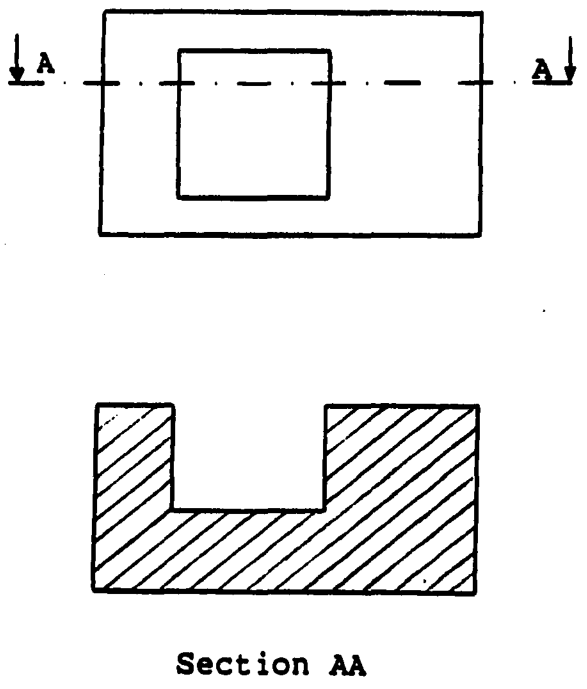


FIGURE 2. Sample part which requires gray level image processing to recognize all features

The first goal of this research was to develop an edge-tracking process which provides the input information for the existing algorithm to recognize nonoverlapping parts. This step would eliminate the restriction to binary images since the edge-tracking algorithm processes only the outline of the parts in the image which can be found using an edge detection algorithm [14,15].

The second goal was to use the very same edge-tracking algorithm as starting point for the overlapping part recognition process. This approach has the advantage that the resulting machine vision software is using the same front end routines containing the necessary algorithms to capture a gray level or binary image, to apply an edge detector, and to identify and trace the edges in the image. The occluded part recognition process will be restricted to untilted, two-dimensional objects viewed from a constant distance. That is, the parts to be dealt with are required to have a small height compared to their width and length dimension.

Finally, the third goal was to compare both, the overlapping parts recognition system and the nonoverlapping parts recognition system. This comparison should also give indication whether or not the recognition of partially occluded parts is feasible using a microcomputer-based machine vision system.

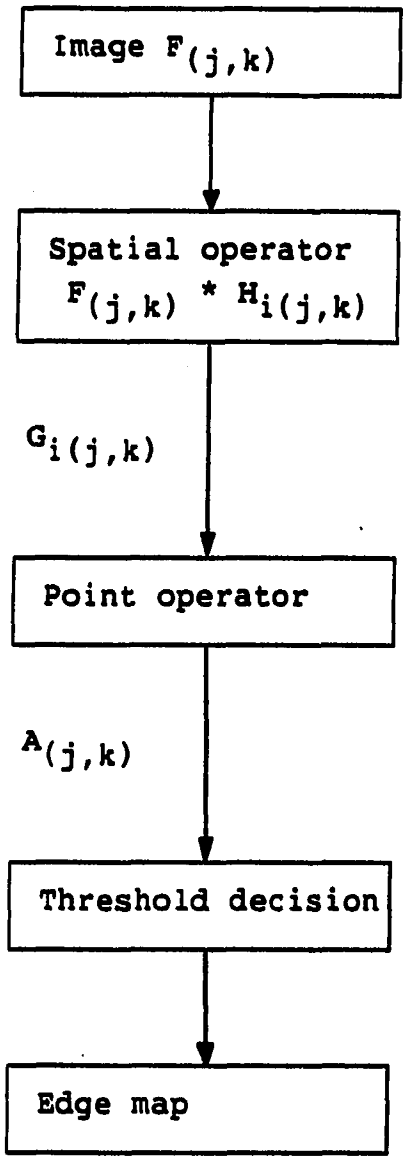
RELEVANT LITERATURE

Computer vision and pattern recognition research started in the early 1960s with research primarily directed toward the development of edge detection [16,17,18] and other image segmentation (or shape description) techniques such as the connectivity algorithm [11], and region growing [19].

Edge Detection

The edge detection is an image analysis technique based on the detection of discontinuities in the gray level intensities in an image. An edge or boundary is defined as a place in an image where there is a more or less abrupt change in the gray level intensity. This definition suggests the use of gradient estimators to detect those changes. Figure 3 shows a typical template matching approach as suggested by Sobel cited in [20], Prewitt cited in [21], and Roberts [22]. Figure 4 shows the gradient or spatial operator as defined by Sobel and Prewitt to estimate the image intensity gradient using a three point average.

Many researchers have evaluated the various edge detection techniques and, depending on the applications, have favored different algorithms. Abdou and Pratt [15] clearly recommend the Sobel or Prewitt operator for their amplitude response invariance to the edge orientation and lack of bias in orientation measurement. Bulloch [23], on



$$G_{1(j,k)} = F_{(j-1,k+n)}\, H_{1(j-1,k+n)} + F_{(j+1,k+n)}\, H_{1(j+1,k+n)} + F_{(j,k+n)}\, H_{1(j,k+n)}$$

$$G_{2(j,k)} = F_{(j+n,k-1)}\, H_{1(j+n,k-1)} + F_{(j+n,k+1)}\, H_{1(j+n,k+1)} + F_{(j+n,k)}\, H_{1(j+n,k)}$$

$n = -1,\ 1$

$$A_{(j,k)} = \left[G_{1(j,k)}^{2} + G_{2(j,k)}^{2}\right]^{0.5}$$

$A_{(j,k)} >$ threshold
$===>$ edge is present

1: edge is present
0: no edge at this position

angle of the edge with respect to the horizontal axis:

$$\theta_{(j,k)} = \arctan\left[\frac{G_{2(j,k)}}{G_{1(j,k)}}\right]$$

FIGURE 3. Edge detection using a gradient estimator

$$F = \begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array} \qquad H1 = \begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline m & 0 & -m \\ \hline 1 & 0 & -1 \\ \hline \end{array} \qquad H2 = \begin{array}{|c|c|c|}\hline -1 & -m & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & m & 1 \\ \hline \end{array}$$

$$G_1 = \frac{a + m\,d + g}{3} - \frac{c + m\,f + i}{3} \qquad \text{Prewitt: } m = 1$$

$$G_2 = \frac{g + m\,h + i}{3} - \frac{a + m\,b + c}{3} \qquad \text{Sobel : } m = 2$$

FIGURE 4. Image intensity gradient computed using the Prewitt or Sobel gradient operator mask

the other hand, recommends the Hueckel gradient operator [24].

All these edge detectors have the major disadvantage of using a global threshold value to decide whether or not an edge is present (see Fig. 3). McIlroy, Linggrad, and Monteith [25] suggested a variable threshold based on the local area brightness of the part of the image currently investigated. This new approach resulted from the understanding that the perceived contrast between regions does not only depend on the gray level gradient, but also on the average light intensity of the region.

Another edge detection approach is the application of the Laplacian operator to the image [26]. This approach, however, does not give any useful directional information about the edge and, being an approximation to the second derivative, doubly enhances any noise in the image.

Pattern Recognition Based on Global Features

These early image processing research efforts were applied in the late 1970s with the introduction of the first machine vision systems by General Motors and the Machine Intelligence Corporations [27]. Both systems use a straightforward bottom-up approach, that is, they reduce the amount of data needed to represent the image information from step to step. Figure 5 shows the knowledge hierarchy for such a global feature based image recognition system. The most crucial step is the generation of the so-called blob list, which was done using a connectivity or blob analysis algorithm.

Lowest level	Pixel
	Blobs
	Global features
Highest level	Parts

FIGURE 5. Knowledge hierarchy for a global feature based pattern recognition system

This kind of an algorithm was developed at the Stanford Research Institute [10] and groups the image into blobs, i.e., groups of pixels of the same color. When this algorithm is applied to binary images, it identifies blobs representing the background and blobs representing the part in the image. The system then computes characteristic values based on the geometry of the whole blob. These characteristic values are called global features and can be the boundary length of the object, its area, the moments of invariant [28,29,3], or Fourier transform parameters [30]. These global features can then be used to locate, identify, and guide the manipulation of industrial parts. Many of today's machine vision systems are using global features for pattern recognition tasks because of the robustness of the algorithm [1].

Pattern Recognition Based on Local Features

Recognition algorithms based on global features (i.e., the geometry of the whole part) are limited to applications generating images containing only nonoverlapping parts. This restriction resulted in the development of feature based pattern recognition methods in the late 1970s [31], which allow the problem of recognizing partially occluded parts to be addressed.

The input to most of these local feature based algorithms is the edge information of the image under consideration. Most edge detectors generate an edge map

(Fig. 3) which contains in the simplest case a "1" at positions where an edge is present and a "0" where no edge was detected. The next logical step in the image analysis process is a further reduction of the information in order to identify and locate the parts in the image. Figure 6 shows the knowledge hierarchy for local feature based pattern recognition systems.

Lowest level	Pixel
	Edge points
	Edge approximation
	Local features
	Match hypotheses
Highest level	Parts

FIGURE 6. Knowledge hierarchy for a local feature based pattern recognition system

Edge approximation

The edge approximation is a crucial step in the pattern recognition process since it replaces the original image information with a set of lines to represent the detected edges in the image. Various methods have been discussed and their performance is evaluated in terms of the accuracy and the speed [32]. Gordon and Seering [32] suggest a least squares approximation which minimizes the distance parallel to the pixel axis most nearly perpendicular to the approximating line.

Pavlidis and Horowitz [33] developed a split and merge algorithm which has been used by a number of researchers. This algorithm is an extension of work by Urs Ramers [34]. Ramers algorithm was designed to represent a boundary using polygons with a minimum number of vertices. The fit criterion is the maximum Euclidean distance of the boundary points to the approximating polygon.

Freeman [35] developed a chain encoding algorithm to represent edge boundaries. The linear interpolation scheme by Ballard [36] realizes an important space saving by not representing all points explicitly and without approximating the boundary by polygons or curves.

Curve fitting algorithms such as B-Splines are not suitable for this application since they require that the curve pass through all the data points [37]. Because the boundary approximation is only done to help to identify local features it is sufficient to use an approximation algorithm as suggested by Ramers [34] which allows the approximated polygon vertices to be close to or at the actual data points.

Feature extraction and hypothesis generation

The next data reduction step in a typical bottom-up recognition approach is the identification of local features such as boundary line segments [38-41], holes and corners [31-45], curves [40,41,46,47], or moments of invariant [48]. Researchers tried to attack this problem in a variety of ways. The main distinction between the various methods lies in the definition of local features and the effort spent to extract those features before a match hypothesis is formulated. There is a constant trade off between the cost to identify more features to generate a more unique hypothesis and the cost to verify various match hypotheses. This trade off must not only be examined with respect to the execution time of the algorithm but also with respect to its robustness. There might be no match hypothesis for a heavily occluded part if too many features are required for the formulation of such a hypothesis.

The hypothesis generation can be classified into two extreme approaches: The one of least commitment exists where all possible matches between image features and stored model features are examined. From these matches, sets of consistent matches are extracted to form match hypotheses that are verified to try to identify one unique match hypothesis. Stockman et al. [49], Bolles [31], Bolles and Cain [44], Bhanu and Ming [39], and Koch and Kashyap [42,43] have used this approach. The advantage is that less time is spent to identify and classify local features but more match hypotheses have to be evaluated before a part can be identified. The matching scheme mostly used in conjunction with this hypothesis generation approach can be described as follows:

1. Match all model features against all image features based on the numerical attributes of the local features.

2. Determine a coordinate transformation that transforms the model features onto the image features. This coordinate transformation is the current match hypothesis.

3. Using coordinate transformation evaluate the current match hypothesis via some similarity measure to accept or reject the hypothesis.

4. If match hypothesis similarity measure is sufficient stop, else go to step 2 to form a new match hypothesis.

The other approach is that of most commitment where, based on the match of a highly distinguished feature, a hypothesis is generated and the remaining features are used to verify that hypothesis. This method, which concentrates on the identification of local features, was used by Perkins [47], Knoll and Jain [50], and Turney et al. [40,41]. The main disadvantage of this approach is that it will fail if a highly distinguished feature happens to be occluded or the part does not have any distinguished local features. The matching scheme for this approach:

1. Identify a set of highly distinguished local features from the image.

2. Match all pairs of model features with the image features using some numerical attributes of the local features and compute a coordinate transformation that maps the model features onto the image features.

3. Extract a consistent set of matching pairs (i.e., coordinate transformations) by detecting a cluster in the coordinate transformation space.

4. Verify this match hypothesis by predicting and identifying other features in the image.

The following occluded parts recognition approaches are the most widely published ones and they differ primarily in the way they identify characteristic parts of the image and how the match hypotheses are formulated.

The Local Feature Method [31,42-45,51,52] characterizes objects by their distinctive corners and arcs (local features). Thus this approach uses spatially interrelated boundary features to model objects. By comparing features in the image with features in the pre-taught models (prototypes), objects in the image are recognized. Since recognition is based only on parts of the boundary, overlapping or touching parts may be recognized. A problem with this method is the selection of the right number of boundary features. If this number is too small, an object might be erroneously recognized. A large number of features results in long computation times due to the exponential nature of the matching process [31]. Furthermore, a large number of local features increases the occurence of false matches which can distort a match hypothesis.

Koch and Kashyap [42,43] do not use an edge detection algorithm to obtain the outline of an object but a boundary tracking algorithm as suggested by Montanari [53]. For the feature selection they use polygon moments as estimators of the similarity of scene and model corners and an association graph to represent match and compatibility constraints. The match hypothesis is verified by computing a coordinate transform that maps the model features onto the scene.

Bolles [31] and Bolles and Cain [44] use corners and holes as local features and uses a maximal clique algorithm to find the largest set of mutually consistent matches between the local features in the image and the model features. A match between an image and a model feature represents a node in a graph. Two nodes are connected by an arc if the matches are mutually consistent, where the consistency criterion is based on the geometry of the model part. The maximal clique algorithm finds the largest completely connected subgraph and the so generated hypothesis is verified using the coordinate transformation to predict the presence of other features and to check for boundary consistency.

The Theta-S Representation Method [40,41,47] computes a curvature function of the boundary. The curvature is defined as the rate of change of Theta, the angle of the tangent to the boundary with the horizontal axis, with respect to the arc length. Perkins [47] extracts high level features called concurves for which he computes 11 numerical attributes. Those concurves attributes are matched against the model feature attributes. The best match give the coordinate transform that is used to verify the model selection.

Turney et al. [40,41] split the Theta-S representation into subtemplates which are matched against model subtemplates by minimizing the mean square errors between them. A Hough transform [20,54] type approach was used to find a group of matching subtemplates in the scene. This group was assumed to be the match hypothesis.

Yet another method is the Edge Cue Analysis as suggested by Shirai [55]. He also approximates the edges by straight lines or elliptic curves to recognize the objects using a hierarchy of features. This method combines features of the two above mentioned approaches in that it uses the Theta-S representation to describe the features and it uses a distinctive local feature to formulate a hypothesis which is verified with the help of the other features in the image. The identification of the local features required 80% of the total recognition time since a high accuracy in the edge recognition process is necessary.

Summary

Most machine vision research during the recent years concentrated on the problem of recognizing partially occluded parts with the overall goal of a general purpose image processing system. Suggested algorithms are, however, often restricted to specific parts (e.g., they require a hole in the parts [51], allow only one part at a time in the image [46], or are computationally very complex and require at least a minicomputer [31,43,41].

This research concentrated on the development and evaluation of a microcomputer-based vision system to recognize partially occluded parts. The software was designed such that the same front end image processing routines could be used as data input to either the global or local feature pattern recognition algorithm. Both, the local and global feature recognition algorithm assume no a priori knowledge about the image (e.g., which parts to expect in the image) and the hypothesis generation was done for the local feature based pattern recognition system using a combination of the two extreme commitment approaches. All image local features where considered valid matches if two feature consistency conditions where satisfied.

THE VISION SYSTEM

The hardware used to develop this occluded parts recognition process fits easily within the lower cost end of vision processing environments. The computer used was an IBM AT operating at 6 MHz, equipped with 512 kBytes of RAM, an 80287 coprocessor, and a CGA monitor. The camera used was a MicronEye camera by Micron Technology, Inc. This camera is capable of generating images with a resolution of 64 by 128 pixels or 128 by 256 pixels. The camera comes with an interface board that fits one of the PC AT expansion slots. Thus, no additional hardware (e.g., frame grabber) is needed to receive images from the camera.

The software was written in a combination of 80286 Assembly code and the "C" language. The low level image preprocessing routines (to control the camera, grab an image frame, and apply the Sobel edge detector) were implemented in Assembly code to achieve a maximum processing speed and to have access to all system resources. The high level pattern recognition routines were implemented in "C". The "C" language was selected because it guarantees a high degree of portability for these generic pattern recognition routines, it allows for dynamic memory allocation during program run time, and it is efficient enough to achieve execution times similar to Assembly code.

The software allows the processing of images up to 256 by 256 pixel resolution and 256 gray levels. This restriction is due to the maximum data segment size of 64 kBytes (256 x 256 x 8) for an 8086 or 80286 processor based computer.

Edge Detection and Identification

The first step in the pattern recognition process is the separation and identification of the objects and the background. This was done using a Sobel edge detector [20] with a local threshold as suggested by McIlroy et al. [25]. The local threshold is the average image intensity in the current image window. The output of this edge detection process is a simple edge map that contains a "one" at positions where an edge was detected and a "zero" where no edge is present.

A recursive edge-tracking algorithm [56] was used to identify which edges belong to which part in the image and to determine the parent-child relationships. This edge-tracking process works similarly to the chain code described by Wilf [57] except that all processing is done in one single image scan.

The algorithm starts by scanning the edge map generated by the Sobel edge detector to find the top left edge pixel. The next pixel on the part boundary can be found by checking if any of the eight surrounding pixels are on the boundary. These eight neighboring pixels (the numbering scheme follows the one suggested by Wilf [57]) relative to the current edge pixel (X) are shown in Fig. 7.

3	2	1
4	X	0
5	6	7

FIGURE 7. Direction numbering used by the edge-tracing algorithm

To assure that the algorithm traces the edges in a clockwise manner the number of pixel locations checked is limited to 5 of the 8 possible locations. The locations are the ones 90 degrees left, 45 degrees left and right, directly forward, 45 degrees right and forward, and 90 degrees right of the last edge pixel. For example if the pixel (X) in Fig. 7 was detected coming from pixel (4), the algorithm would check if the pixels 2, 1, 0, 7, and 6 are on the part boundary. The so identified edge pixels are marked with a unique number pertaining to that part boundary. When the entire edge is traced, the edge-tracking process finds a pixel set numbered with that unique edge identifier. This signifies that the edge forms a complete chain.

Thus, the output of the edge-tracking algorithm is an updated edge map where all edges belonging to the same part are labeled with the same number. Furthermore, the algorithm constructs an array (BLOBPARENT) to describe the parent-child relationships. A value BLOBPARENT[i] = i indicates that the edges numbered i in the edge map belong to a parent part. A value BLOBPARENT[i] = j indicates that the edges numbered i belong to a child (hole) in parent part j.

If nonoverlapping parts are to be recognized it is relatively easy to compute the moments of invariant as descriptive figures for part identification during the edge-tracing process. These values are invariant with respect to the angle at which one looks at a part. Petersen and Even [3,4] and Petersen et al. [56] have demonstrated the use of these values for part identification in binary and gray level images.

Edge Boundary Approximation

The occluded parts recognition process continues with the approximation of the outer part boundary by polygon line segments. The reason for this step is a further reduction of the image information and the need to identify local features in the image. As above mentioned, global features such as the moments of invariant, are not adequate descriptive figures for the recognition of overlapping parts. The recognition must be based on a number of local features so that the absence of a local feature due to an occlusion can be explained by the matching of the remaining features.

Ramers [34] iterative polygon approximation procedure was used because of its ease of implementation, its robustness, and its approximation quality using a small number of vertices. Furthermore, more sophisticated algorithms like spline curve fitting require all data points to be on the curve [37]. This is not necessary and not even desired for this application.

Ramers suggests as a fit criterion the maximum Euclidean distance between the approximating polygon line segment and the curve. The distance is found where the tangent to the curve is parallel to the straight line segment. The algorithm generates a new vertex if the fit criterion exceeds a set threshold value.

The computation of this fit criterion needed to be changed to avoid the definition of the tangent to the edge boundary. The implemented algorithm computes the Euclidean distance D between the approximating straight-line segment (connecting points 1 and 2 in Fig. 8) and each point (e.g., point 3 in Fig. 8) on the actual part boundary to determine the maximum distance D_{max}.

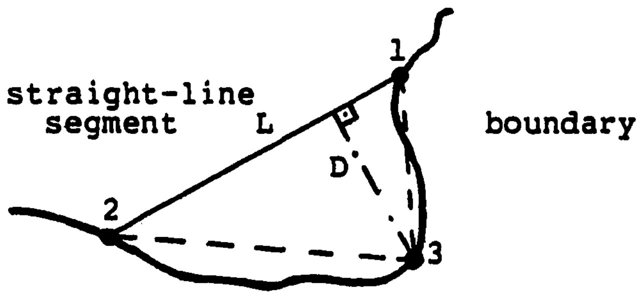


FIGURE 8. Definition of the fit criterion D

The area of the triangle 123 in Fig. 8 can be defined as

$$A = \frac{L\,D}{2} = \frac{X_1Y_2 - X_2Y_1 + X_2Y_3 - X_3Y_2 + X_3Y_1 - X_1Y_3}{2} \tag{3-1}$$

where L is the length of the straight-line segment between the points 1 and 2 and (Xi, Yi) are the coordinates of the triangle corners i. Solving (3-1) for the Euclidean distance D.

$$D = \frac{X_1Y_2 - X_2Y_1 + X_2Y_3 - X_3Y_2 + X_3Y_1 - X_1Y_3}{L} \tag{3-2}$$

Rearranging and defining the constant (for one straight-line segment) terms

$$\Delta X = X_2 - X_1 \tag{3-3}$$

$$\Delta Y = Y_1 - Y_2 \tag{3-4}$$

$$UV = X_1\ Y_2 - X_2\ Y_1 \tag{3-5}$$

yields

$$D = \frac{UV + \Delta X\ Y_3 + \Delta Y\ X_3}{L} \tag{3-6}$$

Equation (3-6) is computationally more efficient than (3-2) since the terms defined by (3-3), (3-4), and (3-5) need only be computed once for the identification of the point of maximum distance from the straight-line.

The other question of concern for the implementation of this algorithm is the selection of the initial vertices, since we are dealing only with closed curves. (An open curve would be a line in the image which would not be labeled by the edge-tracing algorithm.) Ramers suggests the selection of two oppositely located extremal points as initial vertices. This approach was found not to be practical due to the additional time spent identifying those extremal points.

Instead, the closed boundary is divided into 5 equal parts and the endpoints of these parts serve as initial vertices. Experiments with the algorithm showed that the division into 5 equal parts and a fit criterion threshold value of 1 to 2 pixels would result in the shortest processing times in most cases. Tables 1, 2, and 3 give the execution times for the implemented polygon approximation algorithm for different numbers of initial vertices and values of the fit criterion threshold value. Threshold values greater than 2 pixel did not guarantee a sufficient approximation quality for the following processing steps.

Figure 9 shows the edge approximation of the model part 1 shown in Fig. 13 using a large threshold value (8 instead of the recommended 1-2 pixels). Note that the algorithm fails to detect corners 5 and 6.

Furthermore, Ramers algorithm was modified such that the previous to last vertex and not the last vertex of the curve segment under consideration is the starting point of

TABLE 1. Execution times in seconds for the polygon approximation algorithm using a fit criterion threshold of 0.8 pixels

Initial vertices	Image in Fig. 13	Image in Fig. 20	Image in Fig. 23
2	0.112	0.210	0.223
3	0.115	0.185	0.200
4	0.110	0.175	0.198
5	0.105	0.177	0.165
6	0.108	0.170	0.197
7	0.121	0.179	0.182

TABLE 2. Execution times in seconds for the polygon approximation algorithm using a fit criterion threshold of 1.0 pixels

Initial vertices	Image in Fig. 13	Image in Fig. 20	Image in Fig. 23
2	0.100	0.169	0.190
3	0.102	0.133	0.165
4	0.098	0.142	0.157
5	0.096	0.151	0.154
6	0.099	0.163	0.154
7	0.110	0.168	0.160

the next curve segment. This modification assured that the arbitrarily selected initial vertices are not automatically included in the set of polygon vertices used to approximate the edge boundary.

TABLE 3. Execution times in seconds for the polygon approximation algorithm using a fit criterion threshold of 2.0 pixels

Initial vertices	Image in Fig. 13	Image in Fig. 20	Image in Fig. 23
2	0.098	0.160	0.187
3	0.102	0.146	0.145
4	0.095	0.137	0.139
5	0.099	0.135	0.133
6	0.096	0.137	0.133
7	0.110	0.124	0.136

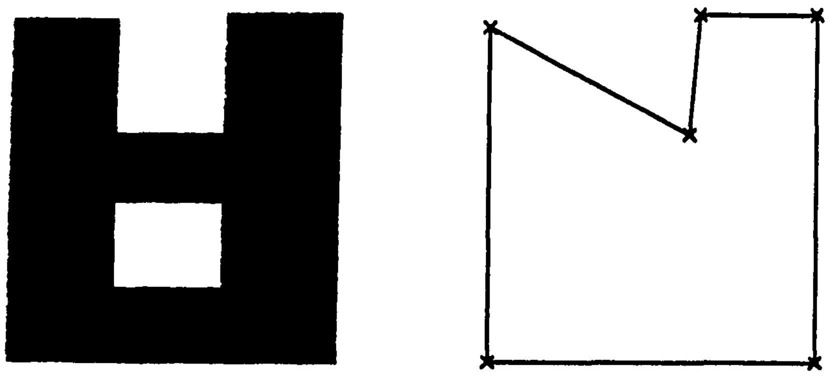
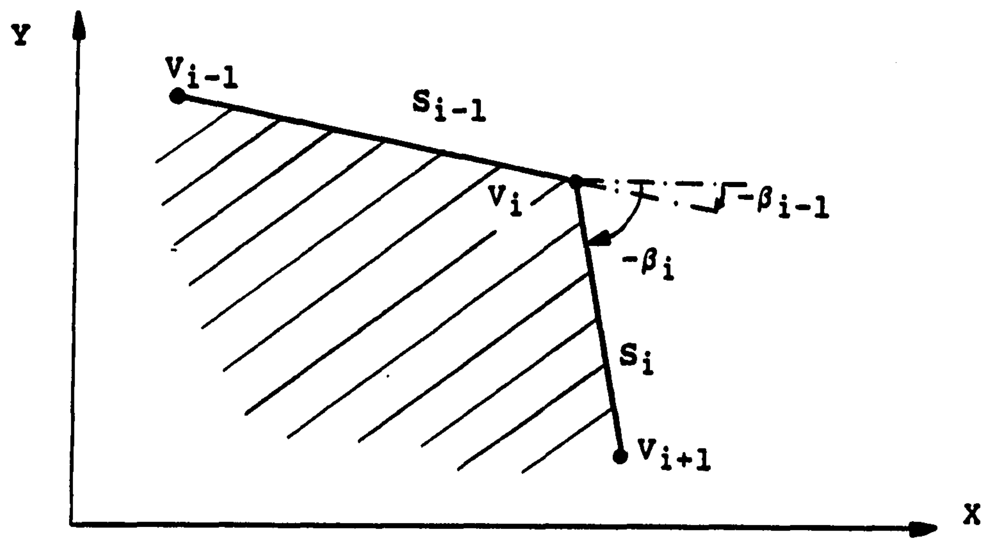


FIGURE 9. Camera image and line approximation of the model part 1 using a threshold value of 8 pixels

Local Feature Definition and Identification

Corners and holes are used as local features in this occluded parts recognition system. Corners have been selected as local features for two reasons. First, humans use such terms as sharp corner, sharp notches, or protrusion to describe the shape of an object; thus it seems logical to use the same kind of descriptors in a vision system. Second, using polygons to describe the boundary of a part makes the identification of corners easy, since the polygon approximation algorithm places vertices at or close to parts of the boundary with a big change in curvature, i.e., corners.



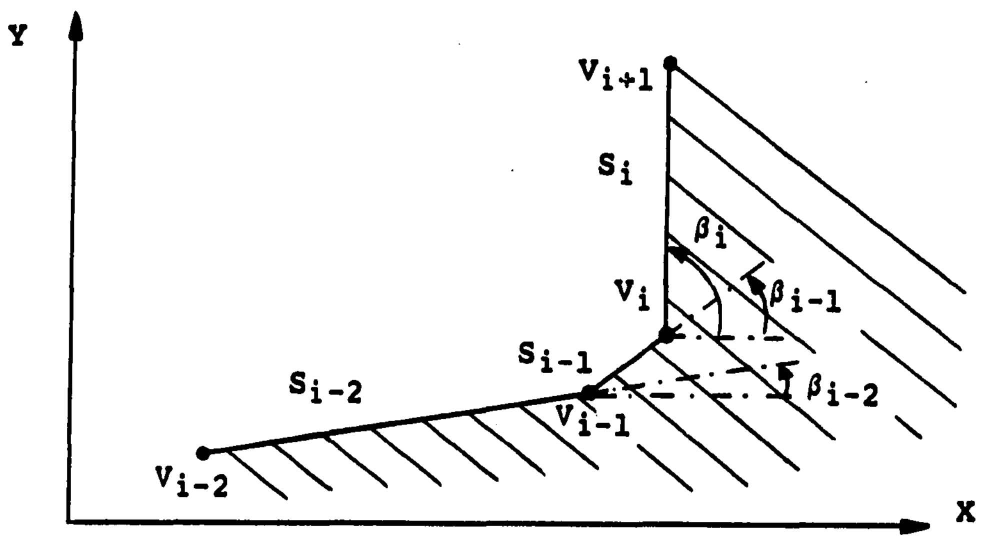
Exterior angle $\theta = \beta_i - \beta_{i-1}$ > 60 degrees

Length of polygon segment S_{i-1} > 4 pixels

Length of polygon segment S_i > 4 pixels

FIGURE 10. Definition of a concave first order corner

The corners are classified into convex or concave corners and are defined as points on the part boundary where



Exterior angle $\theta = \beta_i - \beta_{i-2}$	$>$ 60 degrees
Length of polygon segment S_{i-2}	$>$ 4 pixels
Length of polygon segment S_{i-1}	$<$ 4 pixels
Length of polygon segment S_i	$>$ 4 pixels

FIGURE 11. Definition of a convex second order corner

the angle between two adjacent line segments is greater than 60 degrees. Figures 10 and 11 show the definition of first or second order type corners. The second order corner definition was necessary to account for inaccuracies in the boundary approximation process and to ensure that all corners in the image can be identified as local features. Corners have only two attributes, namely their type (i.e., convex or concave) and the magnitude of the exterior angle θ.

Holes are defined in terms of their area and the location of their centroid, relative to the top left corner of the image. The centroid coordinates are computed using the moment calculation equation developed by Hu [28], Wong and Hall [29], and Wilf [57]. Defining the coordinate differences ΔX and ΔY of two adjacent edge pixels as

$$\Delta X_i = X_i - X_{i-1} \qquad (3\text{-}7)$$

$$\Delta Y_i = Y_i - Y_{i-1} \qquad (3\text{-}8)$$

where (Xi, Yi) represent the coordinates of the edge pixel i. The interior area of a closed curve is given by equation (3-9).

$$\text{Area} = \frac{1}{2} \sum_{i=1}^{n} (X_i \Delta Y_i - Y_i \Delta X_i) \qquad (3\text{-}9)$$

and the centroid coordinates can be computed using (3-10) and (3-11).

$$Cx = \frac{1}{3} \sum_{i=1}^{n} \{(X_i \Delta Y_i - Y_i \Delta X_i)\ (Y_i - \frac{1}{2} \Delta Y_i) \ / \ \text{Area}\} \qquad (3\text{-}10)$$

$$Cy = \frac{1}{3} \sum_{i=1}^{n} \{(X_i \Delta Y_i - Y_i \Delta X_i)\ (X_i - \frac{1}{2} \Delta X_i) \ / \ \text{Area}\} \qquad (3\text{-}11)$$

Thus, the area and centroid coordinates of a closed curve can be computed by simply walking around the boundary and updating the moment equation (3-9, 3-10, 3-11) at every edge pixel. Equations (3-12) to (3-16) can be used in the manner to compute the descriptive figures for a global feature based recognition system as proposed by Petersen et al. [56].

$$AL_i = X_i \, \Delta Y_i - Y_i \, \Delta X_i \qquad (3\text{-}12)$$

$$I_y = \frac{1}{4} \sum_{i=1}^{n} AL_i (X_i^2 - X_i \, \Delta X_i + \frac{1}{3} \Delta X_i^2) \qquad (3\text{-}13)$$

$$I_x = \frac{1}{4} \sum_{i=1}^{n} AL_i (Y_i^2 - Y_i \, \Delta Y_i + \frac{1}{3} \Delta Y_i^2) \qquad (3\text{-}13)$$

$$I_{xy} = \frac{1}{4} \sum_{i=1}^{n} AL_i (X_i Y_i - \frac{1}{2} X_i \Delta Y_i - \frac{1}{2} Y_i \Delta X_i + \frac{1}{3} \Delta X_i \, \Delta Y_i) \qquad (3\text{-}14)$$

$$I_1 = I_x + I_y \qquad (3\text{-}15)$$

$$I_2 = I_x \, I_y - I_{xy}^2 \qquad (3\text{-}16)$$

Equations (3-15) and (3-16) evaluate the so-called moments of invariant which are invariant to the angle at which one looks at a part. Thus these values are suited for pattern recognition of nonoverlapping parts.

The reason for using the area and the centroid coordinates as hole features and not using the corners as local features as done for the outer part boundary is threefold.

1. The hole perimeter is relatively small compared to the perimeter of the whole part. The part shown in Fig. 13 for example has an outer boundary length of 204 pixel and the hole boundary is only 45 pixel long. Noise in the image has a far greater effect on the approximation quality of a short boundary than of a longer boundary. Thus, greater errors must be expected if the boundary of holes is approximated using the implemented algorithm.
2. It is computationally more expensive to find a polygon approximation of the boundary and to identify corners as local features than to compute the area and the centroid coordinates of an hole.
3. Given the above local feature corner definition, circular holes could not be represented by this system.

Feature Matching

The feature matching process computes, for all compatible image and model features, a coordinate transformation that maps a model feature onto an image feature; i.e., every compatible image feature is matched against every model image. This approach assumes that only rigid body motion is considered; in other words, parts are not allowed to be deformed. The feature compatibility rules are defined as

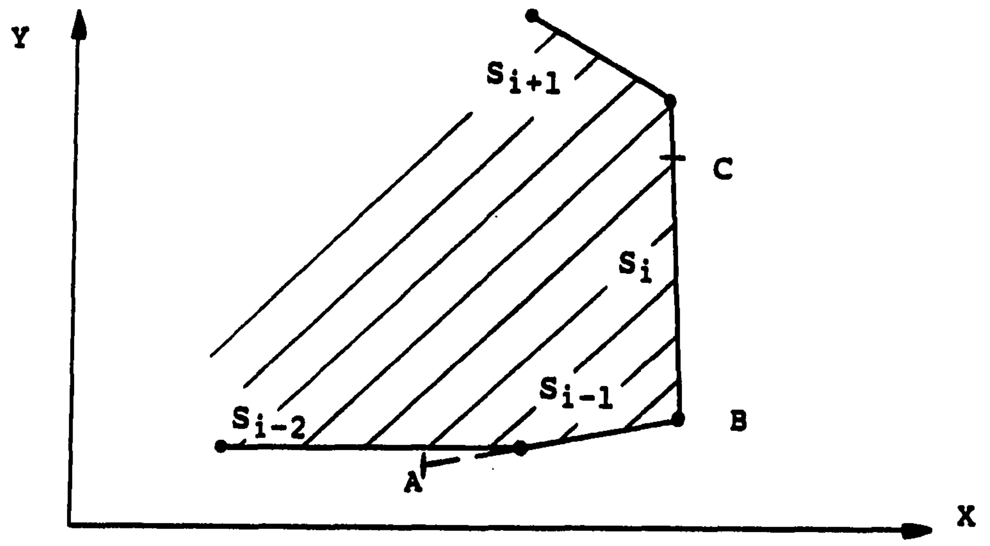
1. both the model and image feature must be either convex or concave,
2. the exterior corner angle of the image feature is not allowed to deviate more than 10% from the model feature angle.

Assume that (I_{xi}, I_{yi}) are local feature coordinates in the current image and (M_{xi}, M_{yi}) are the coordinates a of local feature of a model part. The matrix equation (3-17) maps a model coordinate point (M_{xi}, M_{yi}) onto the image point (I_{xi}, I_{yi}):

$$\begin{bmatrix} I_{xi} \\ I_{yi} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -1/R \sin(\theta) \\ R \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} M_{xi} \\ M_{yi} \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix} \qquad (3\text{-}17)$$

where θ is the rotation of the model point around the coordinate system origin and (T_x, T_y) is the translation needed to align the model point with the image point. R is the row to column aspect ratio of the image sensor. This parameter is necessary since not all cameras display an image without distortion. Thus, the image in the computer memory must be adjusted at this point to represent the real image. The MicronEye camera has a row to column aspect ratio of 2.5. However, this value is much smaller for more sophisticated cameras. For example, the GE TN2500 CID camera has a row to column ratio of only 1.38.

Computing the coordinate transform parameters (θ, T_x, T_y) based on only one feature point proved insufficient due to inaccuracies in the boundary approximation process. Instead, the transform parameters were determined based on an average value of three corner points (points A, B, C in Fig. 12). Koch and Kashyap [43] call these terms polygon moments. The coordinates of points A, B, and C are stored by the system in addition to the two corner attributes, i.e., corner type and magnitude of the exterior angle θ.



$\overline{AB} = 10$ pixels $\qquad \overline{BC} = 10$ pixels

FIGURE 12. Definition of the corner endpoints

To determine the coordinate transformation needed to translate a model point onto a given image point one can use a least square approach and minimize the sum of the squared deviations between the real image points and the transformed model points; thus

$$\text{MIN} \quad J = J_1 + J_2 \qquad (3\text{-}18)$$

where the terms J_1 and J_2 are defined as

$$J_1 = \frac{1}{n} \sum_{i=1}^{n} [I_{xi} - M_{xi} \cos(\theta) + \frac{M_{yi}}{R} \sin(\theta) - T_{xi}]^2 \qquad (3\text{-}19)$$

$$J_2 = \frac{1}{n} \sum_{i=1}^{n} [I_{yi} - M_{yi} \cos(\theta) - M_{xi}R \sin(\theta) - T_{yi}]^2 \qquad (3\text{-}20)$$

Taking the partial derivative with respect to T_x, T_y, and θ results in three equations to determine the three unknown coordinate transformation parameters. The partial derivatives with respect to T_x and T_y equated to zero yield equations (3-21) and (3-22). Note that the summation limits i = 1,...,n are implied by the summation sign $\sum$:

$$\frac{\partial J}{\partial T_x} = \frac{2}{n} \sum [I_x - M_x \cos(\theta) + \frac{M_y}{R} \sin(\theta) - T_x](-1) = 0 \qquad (3\text{-}21)$$

$$\frac{\partial J}{\partial T_y} = \frac{2}{n} \sum [I_y - M_y \cos(\theta) - M_x R \sin(\theta) - T_y](-1) = 0 \qquad (3\text{-}22)$$

Solving for T_x and T_y

$$T_x = \frac{1}{n} [\sum I_x - \sum M_x \cos(\theta) + \sum \frac{M_y}{R} \sin(\theta)] \qquad (3\text{-}23)$$

$$T_y = \frac{1}{n} [\sum I_y - \sum M_y \cos(\theta) - \sum M_x R \sin(\theta)] \qquad (3\text{-}24)$$

Taking the partial derivative with respect to θ and equating it to zero yields

$$\frac{\partial J_1}{\partial \theta} = \frac{1}{n} \sum [\{I_x - M_x \cos(\theta) + \frac{M_y}{R} \sin(\theta) - T_x\}\{M_x \sin(\theta)$$
$$+ \frac{M_y}{R} \cos(\theta)\}] \qquad (3\text{-}25)$$

$$\frac{\partial J_2}{\partial \theta} = \frac{1}{n} \sum [\{I_y - M_y \cos(\theta) - M_x R \sin(\theta) - T_y\}\{M_y \sin(\theta)$$

$$- M_x R \cos(\theta)\}] \tag{3-26}$$

$$\frac{\partial J}{\partial \theta} = \frac{\partial J_1}{\partial \theta} + \frac{\partial J_2}{\partial \theta} = 0 \tag{3-27}$$

Substituting the expressions for Tx and Ty into (3-25) and (3-26), respectively, and multiplying by n allows (3-27) to be rewritten as

$$\frac{\partial J}{\partial \theta} = 0 = R \sum(I_x M_x) \sin(\theta) + \sum(I_x M_y) \cos(\theta)$$

$$- \frac{R}{n} \sum I_x \sum M_x \sin(\theta) - \frac{1}{n} \sum I_x \sum M_y \cos(\theta)$$

$$- \sum(I_y M_x) \cos(\theta) + \frac{1}{R} \sum(I_y M_y) \sin(\theta)$$

$$+ \frac{1}{n} \sum I_y \sum M_x \cos(\theta) - \frac{1}{n R} \sum I_y \sum M_y \sin(\theta) \tag{3-28}$$

Solving (3-28) for θ yields

$$\tan(\theta) = \frac{\sum(I_y M_x) - \sum(I_x M_y) + [\sum I_x \sum M_y - \sum I_y \sum M_x]\frac{1}{n}}{R[\sum(I_x M_x) - \frac{1}{n}\sum I_x \sum M_x] + \frac{1}{R}[\sum(I_y M_y) - \frac{1}{n}\sum I_y \sum M_y]} \tag{3-29}$$

Matching each local feature of the model part shown in Fig. 13 against each compatible feature of the image shown in Fig. 14 results in the coordinate transformation parameters given in Table 4. Note that the convex corners

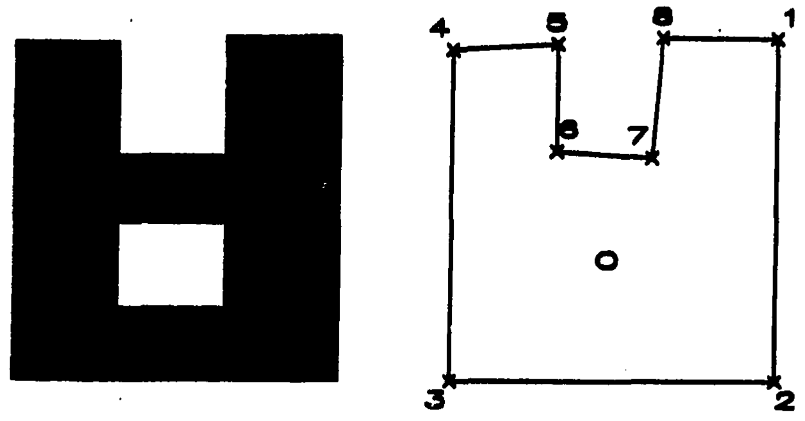


FIGURE 13. Camera image and line approximation of the model part 1

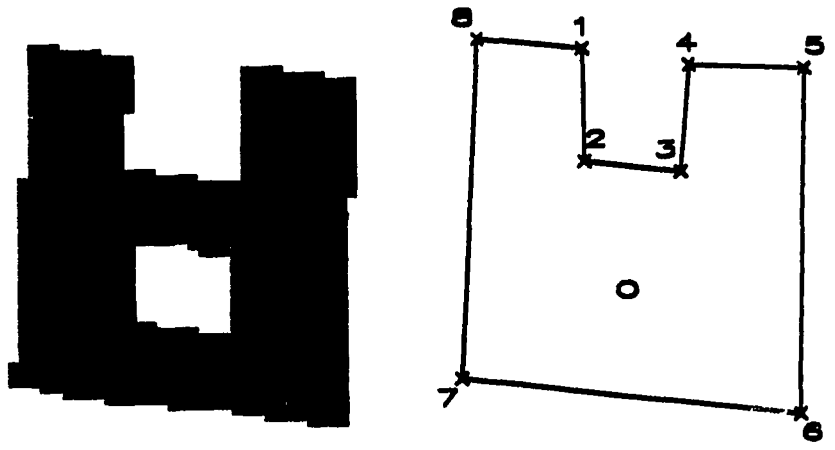


FIGURE 14. Camera image and line approximation of the rotated model part 1

of the model part (corners 6 and 7 in Fig. 13) are matched only against convex corners (2 and 3) in the image shown in Fig. 14. The corners are numbered in the order in which they were detected by the system.

This feature matching process results in the restriction of this vision system to recognize only untilted parts in the image. The coordinate transformation between the model features and the image features is assumed to be two-dimensional. Thus, both features must be in the same plane.

Hypothesis Generation

In order to generate a match hypothesis one has to identify the biggest cluster of consistent coordinate transformation parameters in Table 4. To solve this problem, a new two step clustering algorithm was developed. This algorithm applies a one-dimensional Hough transform [58] in the first step to find initial clusters in the θ space. The second step identifies and deletes stray matches from the θ clusters based on a second clustering in the Tx parameter space. The complete algorithm is as follows.

1. Initialize accumulator array with 90 bins (each 4 degrees "wide")
2. Increment the appropriate bin for each θ entry in Table 4 by one, e.g., the first entry in Table 4 would increment bin 0 by one, the second entry would increment bin 75 by one and so forth.
3. Compute for each bin the sum of its two neighbor bin entries plus its own entry.
4. Find the biggest rotational clusters in this new accumulator array.
5. For this cluster, find the maximum number of consistent Tx translation parameters. If the maximum number of consistent Tx matches is smaller than 30% of the number of local features in the model part, discard this θ cluster. Otherwise, add this cluster to the hypothesis

TABLE 4. Coordinate transformation parameters

Model corner	Image Corner	θ (degree)	Tx (pixel)	Ty (pixel)
1	1	1.6	-16.9	-3.9
1	4	303.0	21.3	115.4
1	5	3.2	4.1	-3.8
1	6	121.2	84.9	-52.5
1	7	183.7	83.1	64.7
1	8	303.3	1.4	110.8
2	1	243.5	39.9	146.8
2	4	180.9	104.1	63.7
2	5	246.5	57.9	151.1
2	6	2.6	4.2	-2.1
2	7	68.6	31.4	-91.2
2	8	183.3	83.4	64.8
3	1	181.6	63.4	60.7
3	4	123.0	85.3	-22.1
3	5	183.3	83.6	66.5
3	6	301.2	22.8	90.2
3	7	3.7	3.7	-7.1
3	8	123.2	65.5	-26.2
4	1	64.1	25.9	-61.8
4	4	1.7	23.6	11.3
4	5	67.1	48.1	-59.0
4	6	183.4	83.9	68.7
4	7	249.2	38.0	119.7
4	8	4.1	3.9	-8.7
5	1	2.4	4.1	-4.7
5	4	303.7	32.1	70.3
5	5	4.0	25.1	-3.1
5	6	121.8	74.7	-6.5
5	7	184.5	62.0	63.6
5	8	303.8	12.2	65.8
6	2	0.9	4.1	-1.4
6	3	304.9	17.7	87.3
7	2	56.9	25.8	-78.8
7	3	1.1	4.2	-1.1
8	1	57.5	11.7	-98.7
8	4	358.4	3.9	6.0
8	5	60.2	34.6	-97.6
8	6	180.0	103.7	62.8
8	7	242.0	51.2	159.2
8	8	0.8	-15.8	-3.3

list by use of an average value for θ, Tx, and Ty. Consistency of the Tx translation parameter is given if the difference between the current value of Tx and the average value of Tx for this rotation cluster is less than 8 pixels.

6. Eliminate the current θ cluster from the accumulator array and continue with step 4 if more hypotheses are desired or terminate.

This algorithm does the clustering first in the θ parameter space because Koch and Kashyap [43] have shown that comparing translational parameters (Tx, Ty) becomes useless since a change in the model coordinate system can make their difference zero even if the rotational parameter (θ) differs. The elimination of stray matches in the rotational parameter space is done based only on Tx because experiments with this algorithm have shown that one translation parameter is sufficient to confirm or reject a match.

Another reason for the initial clustering to be done for the rotational parameter is the fact that the lower and upper bounds for θ are well defined (namely 0 to 360 degrees) so that a straightforward accumulator array can be used for this task.

Table 5 shows the four hypotheses that were generated from the match data in Table 4 by use of the above clustering algorithm. Looking at the data in Table 4 and the first hypothesis, one finds that the system correctly identified a match between the model part corners 1, 2, 3, 4, 5, 6, 7, 8 (Fig. 13) and the image corners 5, 6, 7, 8, 1, 2, 3, 4 (Fig. 14), respectively.

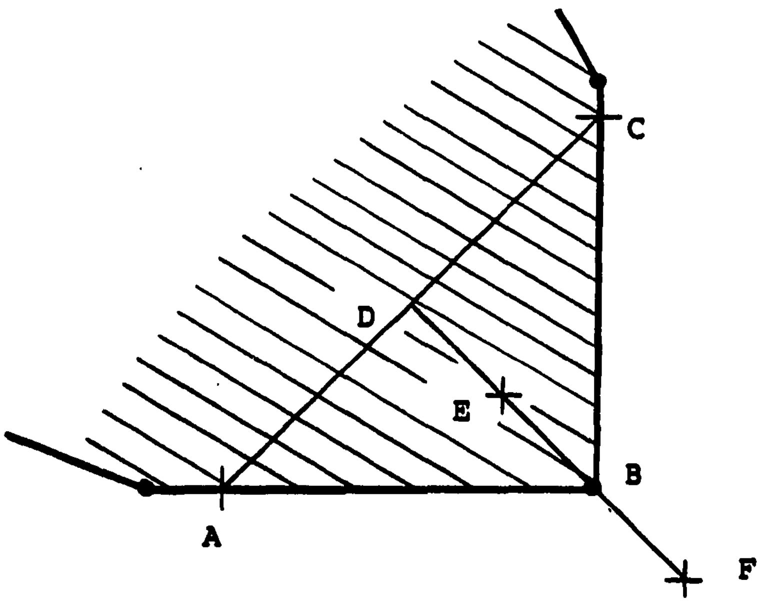
TABLE 5. Match hypotheses generated from the Table 4 data

Hypothesis	θ (degree)	Tx (pixel)	Ty (pixel)
1	2.04	4.04	-2.86
2	-177.43	83.49	66.19
3	-56.68	18.51	89.67
4	67.82	31.38	91.20

Hypothesis Verification

The hypothesis verification uses a similar approach as Knoll and Jain [50]. Given a match hypothesis with its coordinate transformation parameters (θ, Tx, Ty), each model corner point can be translated into the image space. The program then examines the neighborhood of that position to find positive or negative indication for the presence of the local feature.

Positive indication for a corner is given if the point E in Fig. 15 is within the part, that is, has the same color as the part and the point F in Fig. 15 is outside of the part, that is, has the same color as the background. Negative indication for a corner is given if the point E is outside of the part. Neutral indication (does not increase either the positive nor the negative hypothesis score) is given in all other cases. The neutral indication is for cases where a corner might be overlapped by another part so that the point F in Fig. 15 is also within a part cluster.



$\overline{AB}$ = 10 pixels $\overline{BC}$ = 10 pixels $\overline{BF} = \overline{BE} = 0.5\ \overline{BD}$

FIGURE 15. Definition of the corner verification checkpoints E and F

The verification of holes is even simpler and the program checks only at the centroid position if a background color is present. If so, the part hole count is incremented by one.

The total hypothesis score is computed using the equation (3-30). Negative scores are set to zero.

$$\text{Score} = \frac{\text{positive - negative}}{\text{number of corners}} * 0.7 + \frac{\text{Hole count}}{\text{number of holes}} * 0.3 \qquad (3\text{-}30)$$

Thus the corners make up 70% of the total hypothesis score and the holes 30%. These weighting factors were chosen because it was felt that the corners give a stronger evidence for the presence of a part in the image due to the twofold check for the existence of a corner feature in the image. A hypothesis is rejected if the hypothesis score is less than an acceptance threshold value. In the current system this value equals 70%. The program selects the hypothesis with the highest score as the match hypothesis between a part cluster in the image and a model part in the library.

Local features that have been identified and for which a satisfactory match hypothesis has been found are marked invalid by the system to assure that they are not included in the matching process for other model library parts.

Model Library

The model library is designed to store in RAM memory the data for all local features necessary to describe the model parts the user wants to identify in other images. This library is constructed by the user by simply showing the system the model parts, executing the local feature identification routine, and storing the generated feature data in the model library. Furthermore, the user has the option to store and retrieve the contents of the model library on or from disk files and to delete individual model parts from the library. The model parts are sorted in the library by increasing number of local features. Figure 16 shows the information stored in the model library for the model part 2 (Fig. 18).

```
Concave corner 1 in parent blob 1 with exterior angle: 86.95
                  row    column
                ---------------
                  9.00   82.00
                  9.00   92.00
                 18.99   92.53

Concave corner 2 in parent blob 1 with exterior angle: 93.04
                  row    column
                ---------------
                 46.01   92.47
                 56.00   93.00
                 56.00   83.00

Concave corner 3 in parent blob 1 with exterior angle: 90.00
                  row    column
                ---------------
                 56.00   78.00
                 56.00   68.00
                 46.00   68.00

Concave corner 4 in parent blob 1 with exterior angle: 90.00
                  row    column
                ---------------
                 19.00   68.00
                  9.00   68.00
                  9.00   78.00

1 hole(s) with the following parameters
       area: 127      cx: 81.01     cy: 32.03
```

FIGURE 16. RAM library data for model part 2

The size of the RAM library is limited by the computer memory (the program allocates the necessary memory for each new model part during run time) and the execution time of the occluded parts recognition algorithm. The execution time of that algorithm increases in the worst case linearly with the number of model parts stored in the RAM library since the algorithm tries to match the image features against each model part until a sufficient match hypothesis has been found.

Program Flowchart

The flowchart shown in Fig. 17 gives an overview of the sequence in which the different algorithms are executed during the overlapping parts recognition process. In addition to the algorithm description the flowchart also lists the function names used in the program source code.

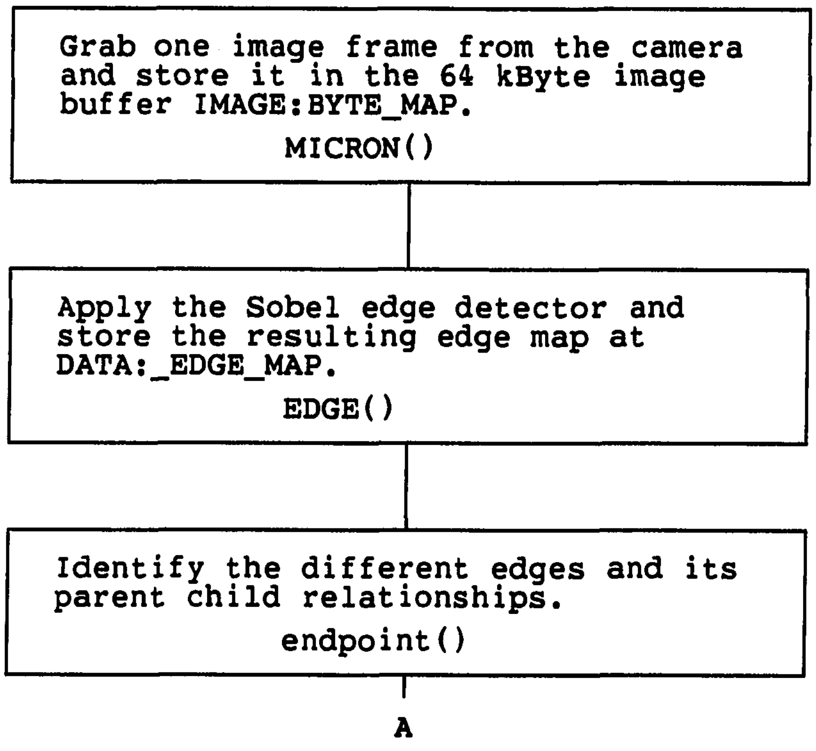

Grab one image frame from the camera and store it in the 64 kByte image buffer IMAGE:BYTE_MAP.
MICRON()
Apply the Sobel edge detector and store the resulting edge map at DATA:_EDGE_MAP.
EDGE()
Identify the different edges and its parent child relationships.
endpoint()
A

FIGURE 17. Occluded parts recognition program flowchart

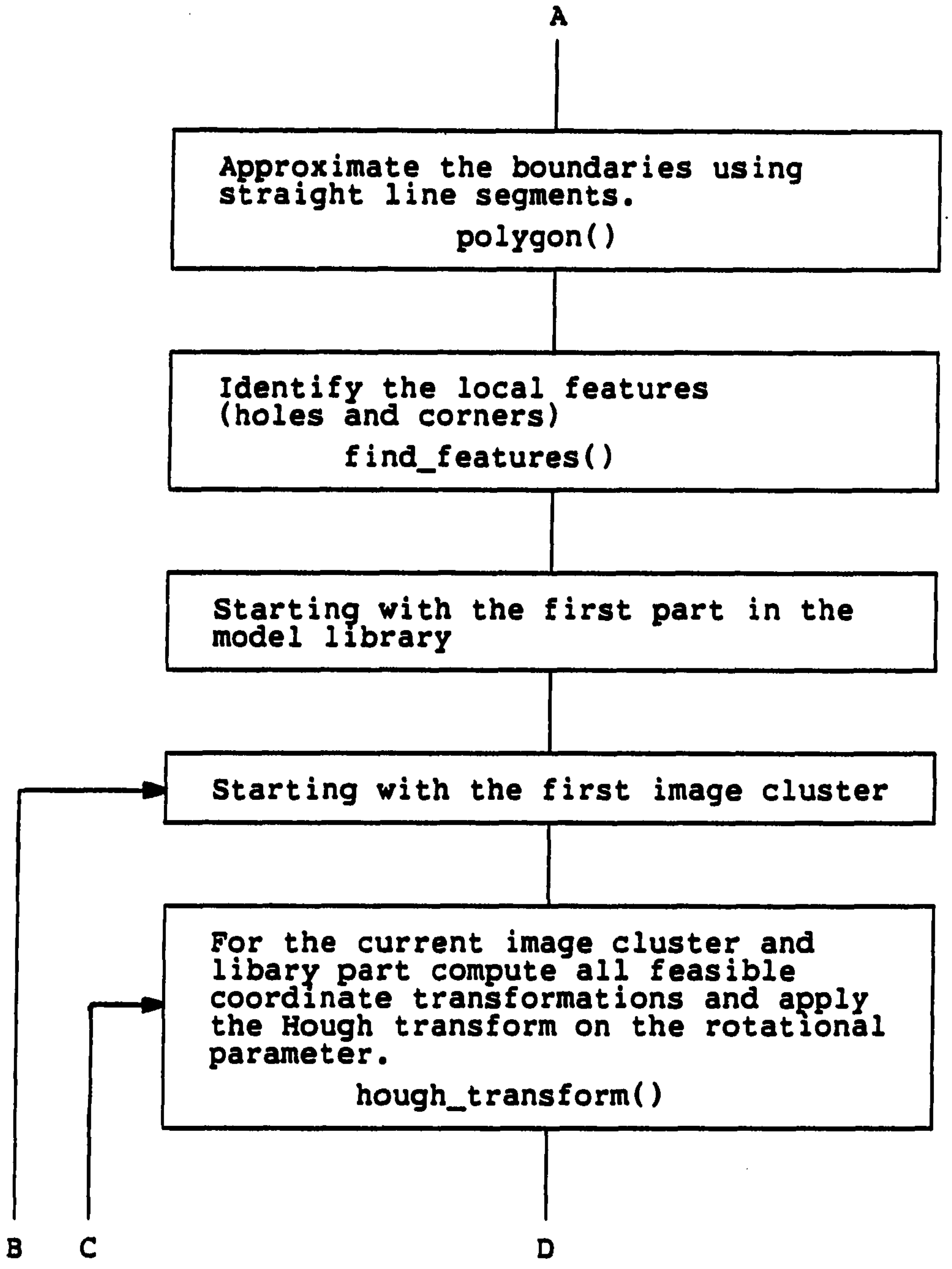

A
Approximate the boundaries using straight line segments.
polygon()
Identify the local features (holes and corners)
find_features()
Starting with the first part in the model library
Starting with the first image cluster
For the current image cluster and libary part compute all feasible coordinate transformations and apply the Hough transform on the rotational parameter.
hough_transform()
B
C
D

FIGURE 17. (Continued)

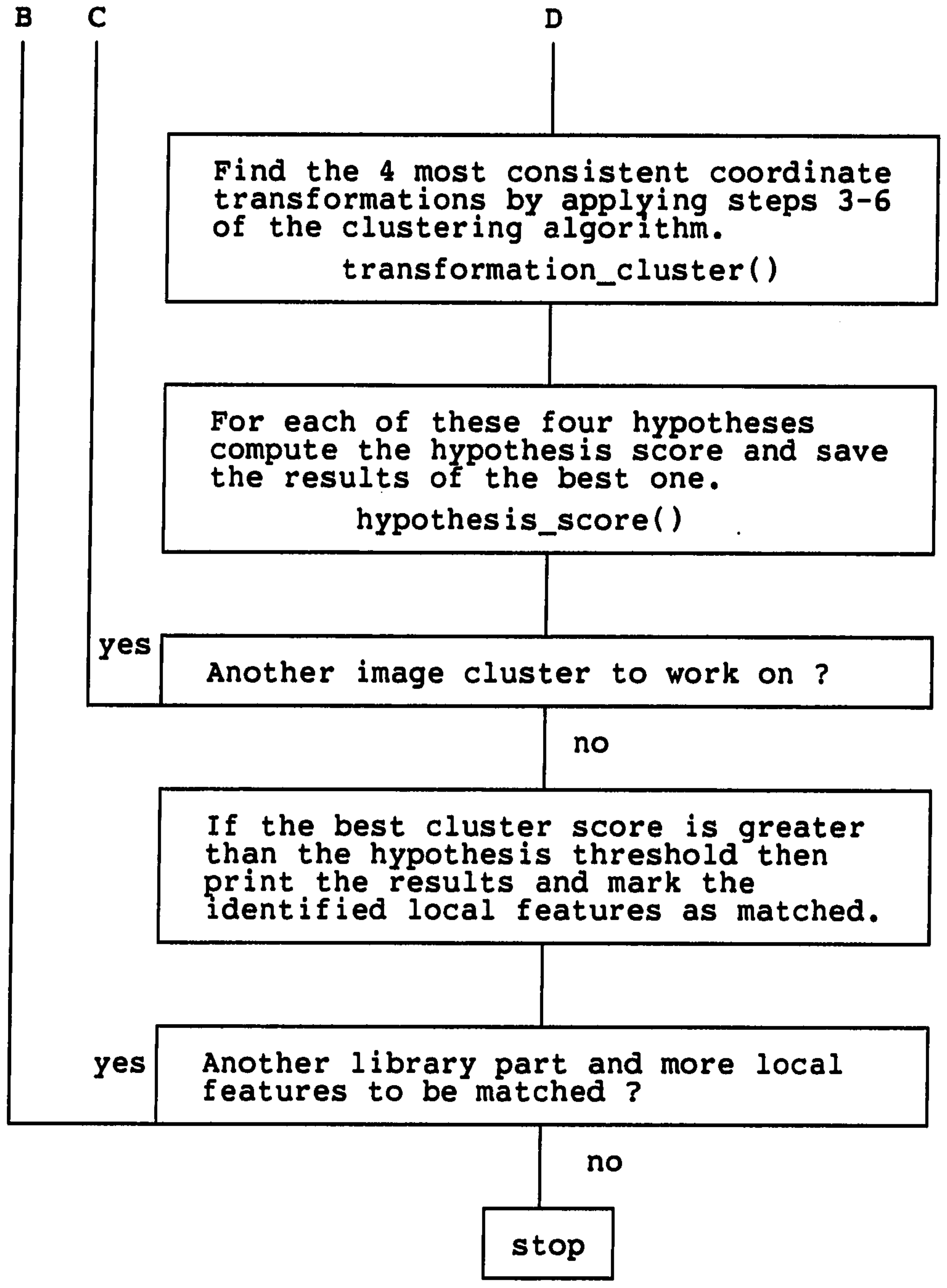

B
C
D
Find the 4 most consistent coordinate transformations by applying steps 3-6 of the clustering algorithm.
transformation_cluster()
For each of these four hypotheses compute the hypothesis score and save the results of the best one.
hypothesis_score()
yes
Another image cluster to work on ?
no
If the best cluster score is greater than the hypothesis threshold then print the results and mark the identified local features as matched.
yes
Another library part and more local features to be matched ?
no
stop

FIGURE 17. (Continued)

RESULTS

This chapter describes the experiments performed to evaluate the performance of the developed occluded parts recognition system. The main emphasis was to gain some insight about the flexibility and limitations of the proposed system and to compare it with another vision system to recognize nonoverlapping parts.

The following experiments and execution times were gathered on the above described computer hardware using the three model parts shown in Figs. 13, 18, and 19. The RAM part library required a total of 0.650 kBytes of memory, where 0.31 kBytes were required for part 1, 0.17 kBytes for part 2, and 0.17 kBytes for part 3. As mentioned above, the storage requirements for a model part increase with the number of local features of that part. For corner features, the algorithm requires the coordinates of three points, the magnitude of the exterior angle, and the direction of curvature (concave or convex). Only the centroid coordinates and the area value are needed to describe a hole feature (see Fig. 16).

The selected test images A-E (Figs. 20-24) were chosen to demonstrate the ability of the algorithm to identify partially occluded parts, to highlight some performance characteristics, and to show its limitations. The test images A-E show the model parts with varying degrees of occlusion and with one or two image clusters to demonstrate the multi cluster handling capability of the software.

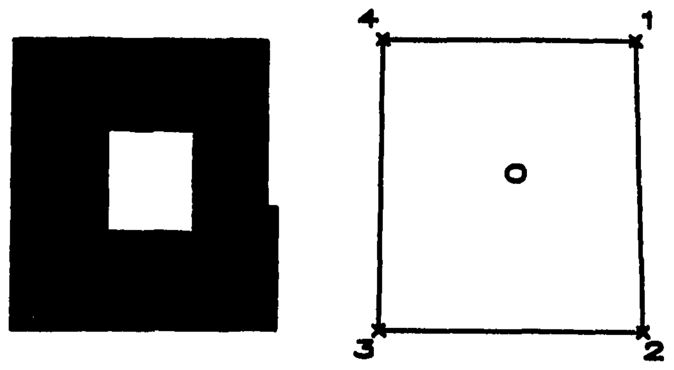


FIGURE 18. Camera image and line approximation of the model part 2

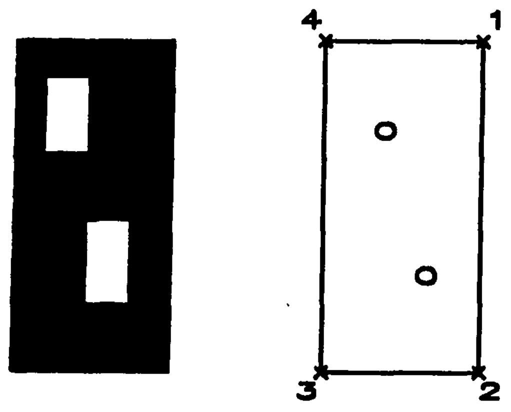


FIGURE 19. Camera image and line approximation of the model part 3

Recognition of Occluded Parts in Different Test Images

Figures 20 and 21 contain only two of the three model parts (namely part 1 and 3) with different amount of occlusion. Thus, only one part cluster is in the image and the algorithm has to determine which of the identified local image features belongs to which model part.

The Tables 6 and 7 show the match hypotheses generated by the system if the test images A and B (Figs. 20 and 21) are compared with the feature information for model parts 1, 2 and 3 (Figs. 13, 18, and 19) stored in the RAM part library.

The system correctly identifies the two model parts 1 and 3 in the image and also fails to recognize model part 2 (since the hypothesis scores for part 2 are less than 70%). Note that only two hypotheses are generated for the match with part 1. The program tries to identify a maximum of 4 match hypotheses. However, it stops processing if no more consistent coordinate transformations can be identified in the local feature match data set. Furthermore, note that the match hypotheses 3 and 4 for the model part 3 results in the same hypothesis score since the part is nearly symmetric and the program identifies it in both positions, namely 9 and -171 degrees rotated. In both cases, the program needed 1.91 sec for the recognition task.

The rotational coordinate transformation parameters for model part 3 should have been approximately the same for

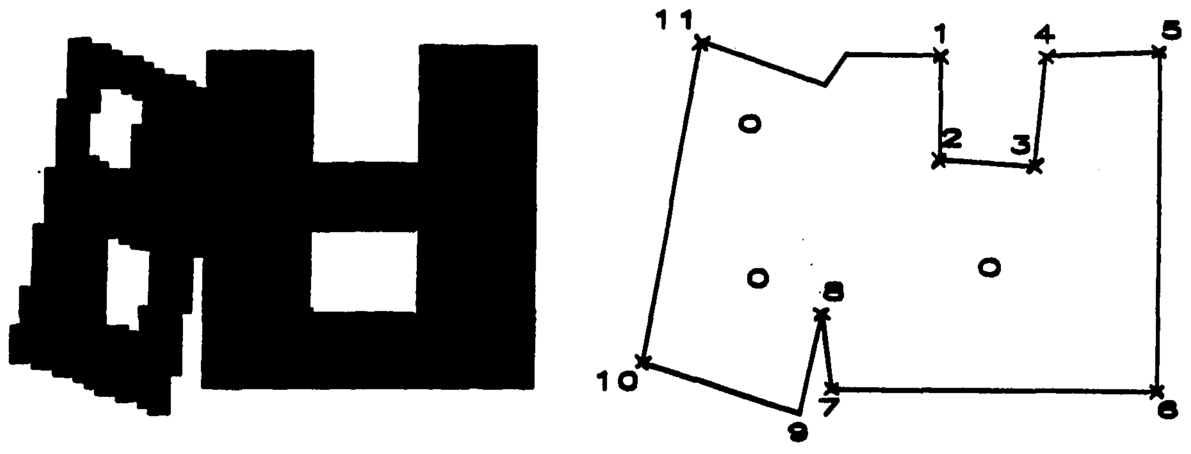


FIGURE 20. Camera image and line approximation of the test image A

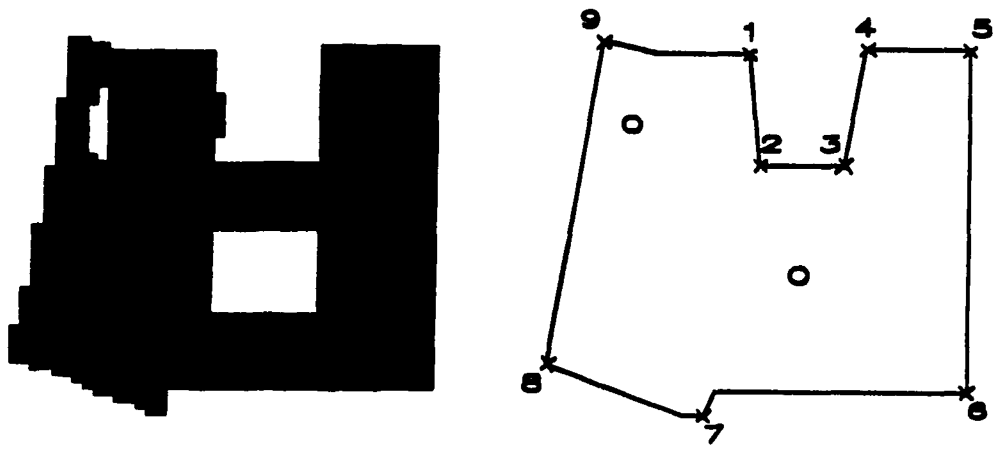


FIGURE 21. Camera image and line approximation of the test image B

TABLE 6. Match hypotheses generated for the image A (Fig. 20)

Hypothesis for part 1	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-0.61	-0.29	0.05	7	0	1	91
2	-99.98	13.11	120.46	1	5	0	0

Hypothesis for part 2	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	1.05	-36.35	-8.07	3	0	0	52
2	8.48	-46.68	-31.75	3	1	0	35
3	0.00	124.00	61.00	0	4	1	0
4	-170.32	107.71	95.68	3	1	0	35

Hypothesis for part 3	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	0.42	5.21	-1.66	2	1	0	17
2	60.42	94.02	58.95	0	4	2	0
3	9.18	-25.03	-17.41	3	0	2	83
4	-170.82	59.06	76.33	3	0	2	83

image A and B since the only difference between both test images is the degree of overlap between the two model parts. The reason why the system computed different coordinate transformation can be explained by approximation errors in the local feature identification process.

However, it is interesting to note that image B with the greater degree of overlap between the two model parts resulted in better match hypothesis scores than image A. This is due to the fact that all corners of part 3 are visible in the image and that the corners have a higher weight than the holes in the computation of the hypothesis scores.

TABLE 7. Match hypotheses generated for the image B (Fig. 21)

Hypothesis for part 1	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-0.27	-0.32	-0.67	8	0	1	100
2	-38.91	-6.02	59.20	1	5	1	0

Hypothesis for part 2	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	1.05	-36.35	-8.07	3	0	0	52
2	8.48	-46.68	-31.75	3	1	0	35
3	0.00	124.00	61.00	0	4	1	0
4	-170.32	107.71	95.68	3	1	0	35

Hypothesis for part 3	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	1.61	5.50	-4.20	2	1	0	0
2	-58.39	93.73	60.99	0	4	2	0
3	8.48	-16.18	-17.35	4	0	1	85
4	-171.52	68.35	73.79	4	0	1	85

In Fig. 22 a third model part (part 2) was added to the image shown in Fig. 20 so that this image consists of two independent part clusters. One of which is the image of the overlapping parts 1 and 3 and the other one is the model part 2.

In order to deal with this kind of images, the system generates a match hypothesis for each model part and each image cluster. That is, each model part is matched against the local features found in the image cluster 1 and 2. The algorithm indicates a match if the maximum match hypothesis score of all the cluster hypotheses is greater than the

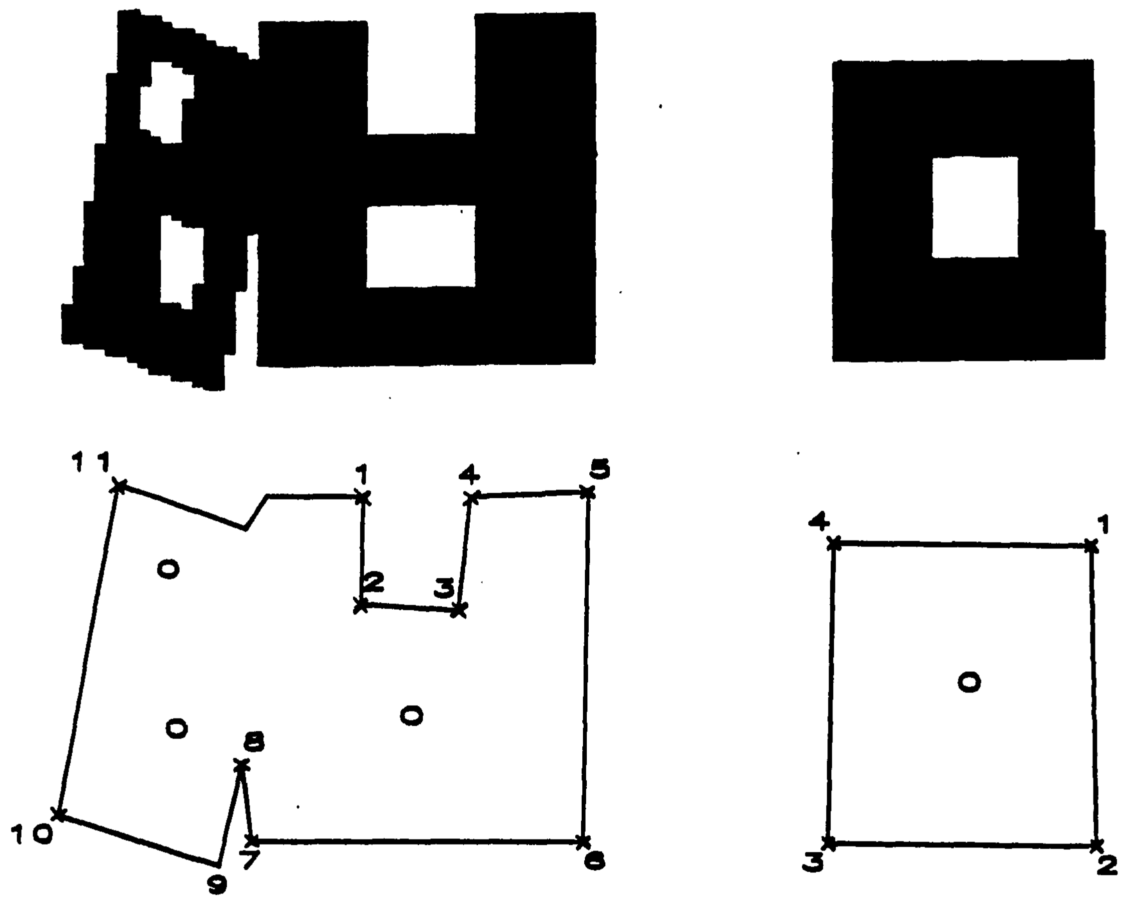


FIGURE 22. Camera image and line approximation of the test image C

hypothesis acceptance threshold value (for the implemented system 70%). The disadvantage of this approach is that a model part must be tested against each image cluster before a final match hypothesis can be formed. Thus multiple cluster in the image will increase the recognition time.

TABLE 8. Match hypotheses generated for the image C (Fig. 22)

Hypothesis for part 1	Image cluster 1 θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-0.61	-0.29	-0.05	7	0	1	91
2	-99.98	13.11	120.46	1	5	0	0

Hypothesis for part 1	Image cluster 2 θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-0.32	49.66	3.49	1	7	1	0
2	89.68	130.85	62.06	0	8	1	0
3	-120.28	106.86	153.09	3	5	0	0

Hypothesis for part 2	Image cluster 1 θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	4.80	-56.21	-22.87	3	0	0	52
2	178.57	120.32	59.48	2	0	0	35
3	9.11	-55.54	-33.55	4	1	0	52
4	121.95	103.32	-123.55	0	4	0	0

Hypothesis for part 2	Image cluster 2 θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	0.00	10.00	0.00	4	0	1	100
2	178.95	170.72	61.74	4	0	1	100
3	-178.95	170.25	69.07	4	0	1	100
4	-59.71	32.27	197.91	0	3	0	0

Hypothesis for part 3	Image cluster 1 θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	0.42	5.21	-1.66	2	1	0	0
2	60.42	94.02	58.95	0	4	2	0
3	9.18	-25.03	-17.41	3	0	2	83
4	-170.82	59.06	76.33	3	0	2	83

TABLE 8. (Continued)

Hypothesis for part 3	θ (degree)	Image cluster 2 Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-1.05	50.29	5.23	2	2	0	0
2	0.00	130.00	62.00	0	4	2	0
3	61.01	79.03	-78.74	2	1	0	17

The analysis of the image C (Fig. 22) resulted after 3.71 seconds in the match hypotheses given in Table 8. The system correctly identified the parts 1 and 3 in the image cluster 1 and the part 2 in the image cluster 2. Due to the symmetry of model part 2, the system offered three equally good match hypotheses for this part. The match hypotheses for the model parts 1 and 3 agree with the ones generated for the analysis of the test image A.

The test images D and E (Figs. 23 and 24) have the model part 2 added to the test images A and B, respectively. Thus they consist of one image cluster of three overlapping parts.

The results shown in Table 9 were generated for image D after 2.92 seconds recognition time. The recognition process for image E required only 2.61 seconds primarily due to the fact that fewer local features were identified than in image D. The match hypotheses for image E are listed in Table 10.

Whereas the system is still able to recognize all parts correctly in the test image D it fails to recognize the

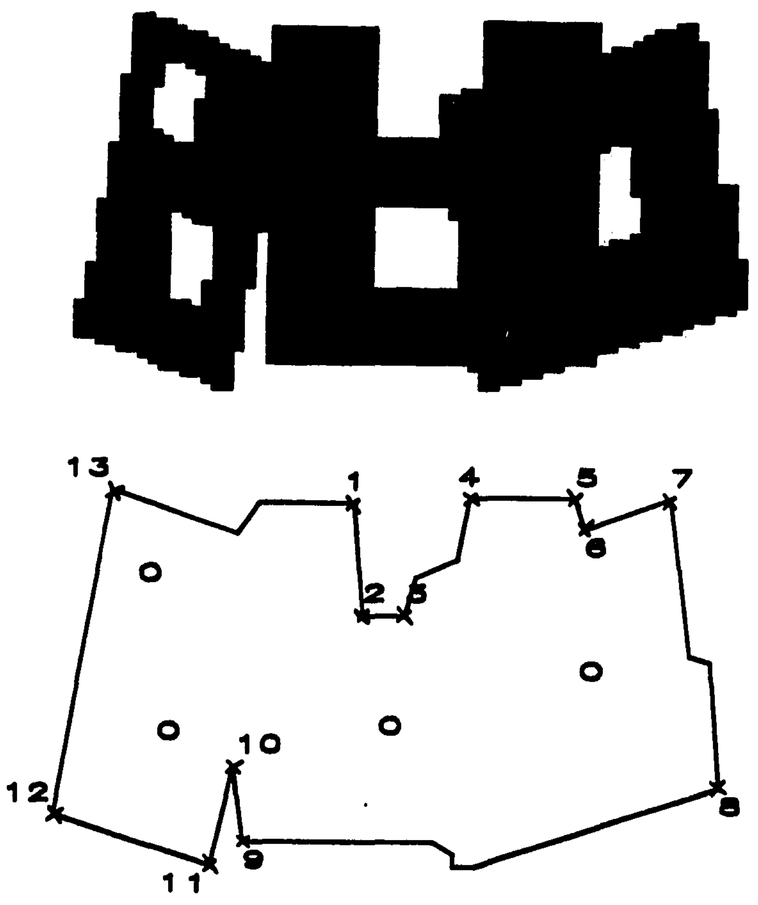


FIGURE 23. Camera image and line approximation of the test image D

model part 2 in image E. This can be explained by the fact that the clustering algorithm does not guarantee that stray matches are not included in the computation of the average coordinate transformation parameters. Looking at the data in Table 10 one can see that the system must have identified

TABLE 9. Match hypotheses generated for the image D (Fig. 23)

Hypothesis for part 1	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-0.68	-0.67	-0.32	7	0	1	91
2	-86.86	42.64	112.39	1	5	0	0

Hypothesis for part 2	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-9.21	-26.30	28.60	3	1	0	35
2	169.67	136.18	24.96	3	1	0	35
3	9.11	-55.54	-33.55	4	0	0	70
4	-169.79	98.28	98.24	4	0	0	70

Hypothesis for part 3	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	169.67	105.45	35.92	2	1	0	17
2	-132.85	75.91	129.37	1	1	0	0
3	9.18	-25.03	-17.41	3	0	2	83
4	-170.82	59.06	76.33	3	0	2	83

most of part 2's local features, since the second set of coordinate transformation parameters are close to the correct ones as listed in Tables 6-9.

The algorithm apparently included a feature match in the coordinate transformation calculation that did not belong to the model part 2. The other reason for differing transformation parameters between two similar images are approximation errors in the edge coding and local feature identification process.

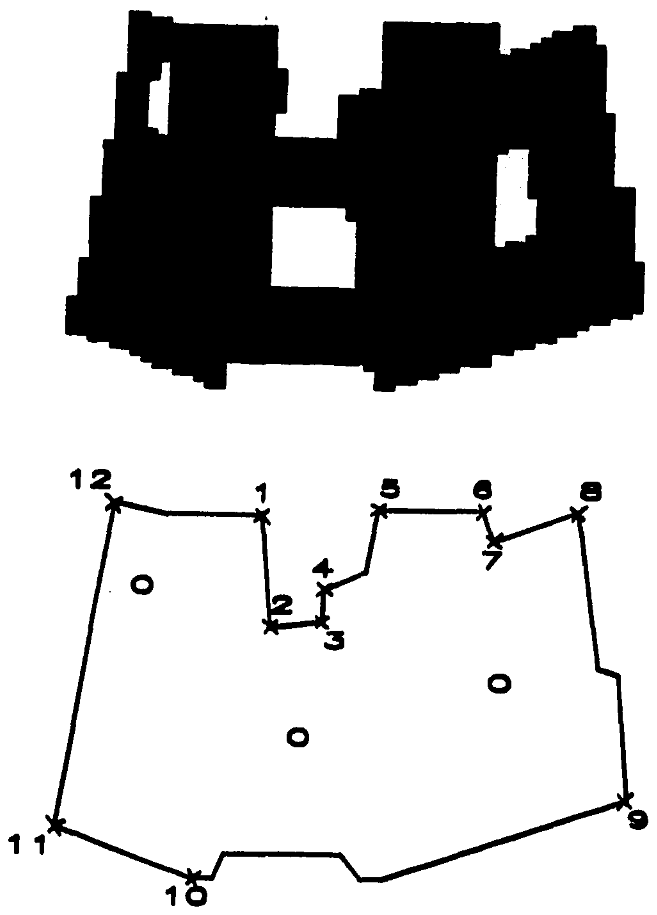


FIGURE 24. Camera image and line approximation of the test image E

Summary

The three model parts (Figs. 13, 18, and 19) were selected because they are representative of the type of images which were used during the development and testing of this parts recognition system. In general, all these parts

TABLE 10. Match hypotheses generated for the image E (Fig. 24)

Hypothesis for part 1	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-5.79	-1.03	12.79	7	0	1	91
2	-38.91	-6.02	59.20	1	5	1	0

Hypothesis for part 2	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	-6.72	-30.93	21.56	3	1	0	35
2	8.48	-46.68	-31.75	3	1	0	35
3	169.67	136.18	24.96	3	1	0	35
4	-170.43	107.69	95.77	3	1	0	35

Hypothesis for part 3	θ (degree)	Tx (pixel)	Ty (pixel)	pos.	neg.	holes	score (%)
1	8.48	-16.18	-17.35	4	0	1	85
2	169.67	105.45	35.92	2	1	0	17
3	-171.52	68.35	73.79	4	0	1	85
4	-10.33	13.72	17.79	2	1	0	17

have distinct corner features, straight edges, and none or some holes. Since this algorithm is not able to recognize round objects, parts with curved boundaries were not attempted to be recognized.

The test images A-E (Figs. 20-24) were selected to show two image sequences with the model parts under varying degree of overlap. The first sequence are the images A and B shown together in Fig. 25. Recall the system had no difficulties recognizing both model parts (1 and 3) in either image even though model part 1 was overlapping part 3 to a great extent. The second sequence was constructed by

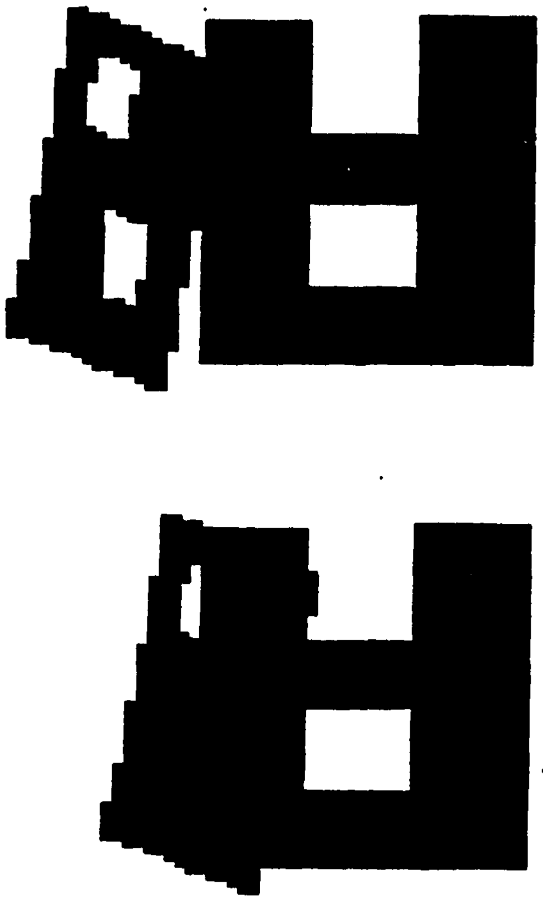


FIGURE 25. Camera images of the first test sequence

adding the model part 2 in exactly the same position to images A and B which resulted in images D and E shown together in Fig. 26. This time the system was able to recognize all parts in image D (Fig. 23) whereas it failed to recognize part 2 in image E (Fig. 24), even though part 2 is shown with the same amount of overlap in images D and E.

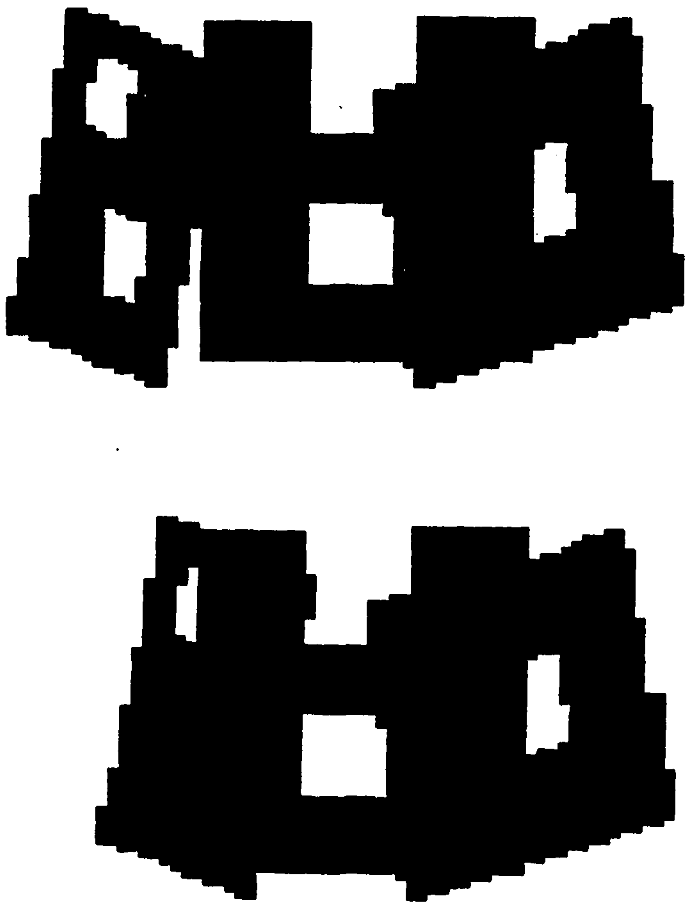


FIGURE 26. Camera images of the second test sequence

This demonstrates the problem with this kind of an algorithm. With increasing amounts of overlap the coordinate transformations computed for individual features become more and more similar for the different model parts. Thus, it can not be guaranteed that the clustering algorithm is not including a coordinate transformation pertaining to another model feature. However, if stray matches are included in the computation of the average coordinate transformation which forms one match hypothesis the hypothesis verification process is more likely to fail.

Comparing the two best match hypotheses for the model part 2 (Tables 9-10), as generated from the test images D and E, one finds that the angular rotation varies only by 0.7 degrees, the translation parameter Tx shows a difference of 8.86 pixels, and Ty a difference of 1.8 pixel. These variations are big enough to let the hypothesis verification process fail.

For this reason it is difficult to establish general rules on when the system will fail to recognize occluded parts and this research did not establish boundaries of the occluded parts recognition ability of the developed system.

Performance Comparison of Two Pattern Recognition Systems

The execution times and space requirements for the demonstrated system are relatively large compared to these performance criteria for a vision system designed to recognize nonoverlapping parts in binary or gray level images. The execution times for the nonoverlapping parts recognition system were gathered with the same hardware configuration using a software package developed by Petersen et al. [56].

TABLE 11. Execution times and library space requirements for vision systems with and without overlapping parts recognition capabilities

Recognition of	Overlapping parts recognition system Time (sec)	Overlapping parts recognition system Library space (kBytes)	Nonoverlapping parts recognition system Time (sec)	Nonoverlapping parts recognition system Library space (kBytes)
Model part 1	1.2	0.310	0.5	0.030
Model part 2	1.1	0.170	0.5	0.030
Model part 3	1.1	0.170	0.5	0.030
Image A	1.9	0.650	-	-
Image B	1.9	0.650	-	-
Image C	2.9	0.650	-	-
Image D	2.6	0.650	-	-
Image E	3.7	0.650	-	-

A comparison of the execution times and the space requirements is shown in Table 11. Three facts are worthwhile noticing. First, for the occluded parts processing system, the execution times and library storage space requirements increase substantially with the complexity of the parts to be dealt with. For the nonoverlapping parts recognition system, the library space requirements are independent of the part complexity and the execution time differences are not measurable for different parts.

Second, since both systems use a brute force library search until all parts (or features) in the image have been matched and the matching in the occluded parts recognition system takes about 50% of the total processing time (20% for the nonoverlapping parts software) the above execution times will deviate even more as more parts are added to the RAM library.

Third, for applications were both systems could be used, the nonoverlapping parts recognition system is about twice as fast as the occluded parts recognition system. This is primarily due to the added task of the edge approximation and the more complex feature matching process for the local feature based pattern recognition process. For tasks, for which partially occluded parts recognition capabilities are necessary (e.g., recognizing parts in the images A-E), the recognition times are quite substantial and the successful performance of the system cannot always be guaranteed.

CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

Conclusions

This research has resulted in a microcomputer-based vision system capable of recognizing partially occluded parts. However, currently the system is restricted to parts which have holes or distinctive corner features. Furthermore, the developed algorithm is also restricted to untilted, two-dimensional parts in the image that are viewed from a constant distance. That is, the parts to be dealt with are required to have a small height compared to their width and length dimension.

The recognition process is based on the development of a new edge-tracking technique in conjunction with a straight line approximation algorithm to identify local features in the image. The local feature definition, as points on the part boundary with an angle greater than 60 degrees between two neighboring straight line segments, resulted in the inability of the algorithm to recognize circular parts.

The local features of the model parts are matched against all compatible local features identified in the current image. A new clustering algorithm has been used to identify clusters of consistent coordinate transformation that serve as initial match hypotheses. A hypothesis verification process eliminates the match hypotheses that are not compatible with the image information on hand.

A performance comparison of this machine vision system with a vision system restricted to nonoverlapping parts recognition and based on the same hardware configuration showed execution times at least twice as long for the occluded parts recognizing system. Furthermore, it appeared, that with increasing complexity of the scene to be analyzed the reliability of the system decreased, due to the fact that the clustering algorithm does not guarantee that stray matches are not included in the hypothesis generating process.

If shorter processing times are required the time performance of this system can be improved using two different approaches. One way would be the implementation of this software on a more powerful computer; this should be straightforward since about 70% of the code was developed in the "C" language which is easily ported onto another computer system. The second option of execution time performance improvement of the software would be the implementation of a parallel processing approach. Since the implemented system uses a traditional bottom-up image processing approach where each step can be performed without the interaction with a previous step it would be feasible to dedicate one processor to the local feature recognition task and another processor to the match hypothesis generation and verification task.

In conclusion, one can say that the recognition of occluded parts is feasible using a microcomputer-based vision system.

Suggestions for Further Research

An extension to this project would be the definition of another local feature based on the average curvature of the polygon line segments. The local feature could be defined at a position where the average curvature changes from convex to concave or vice versa. This would allow for the recognition of round objects.

The next step would be the elimination of the constraint that only untilted parts at a fixed viewing distance can be recognized. Petersen and Even [4] have shown that a simple sonar range finding device can be used to allow parts recognition from varying viewing distances. To allow tilted parts to be recognized would widen the application scope of the vision system and solve the bin of parts problem.

Another suggestion for further research would be a quantative examination of all the parameters used by this system and their impact on the performance of the algorithm. Parameters to be evaluated include:

1. The fit criterion threshold value for the boundary approximation algorithm. What is the best compromise between approximation quality, execution time, and number of vertices needed to approximate the boundary.

2. The distance of the corner endpoints A and C (Fig. 12) which is currently defined as 10 pixels. Larger values will enhance inaccuracies in the edge approximation and smaller values will result in more similar values for the coordinates of the corner endpoints.

3. The maximum angle deviation between the image and model local features exterior angle which is one of the two feature compatibility rules.

4. The bin size of the rotational parameter accumulator array currently set to 4 degrees.

5. The consistency criterion of the Tx translation parameter. Consistency is given if the difference between the current value of Tx and the average value of Tx for this rotation cluster is less than 10 pixels.

6. The weights used in equation (3-30) to compute the total hypothesis score by combining the individual corner and the hole scores.

7. The hypothesis acceptance threshold value currently set to 70 %.

Most of these parameters have been determined individually without looking at the interrelationships. Currently three parameters (the fit criterion threshold value, the angle deviation parameter, and the hypothesis acceptance threshold) can be modified by the user without recompiling the program.

Since the quality of the local feature identification algorithm depends mostly on the performance of the boundary approximation algorithm, it would be interesting to implement another approach such as the one suggested by Beus and Tiu [59] and to compare both methods.

These suggestions for further research identify one problem in the current machine vision research approach. A variety of individual researchers try to attack the whole problem at once instead of concentrating at smaller tasks in order to develop more sophisticated image processing functions. One reason can be seen in the lack of public image processing software. Research papers usually do not publish the code. At best they provide a good outline of the algorithm being used. In order to start working in this field one has to redevelop all machine vision software components.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my major professor, Dr. John C. Even, for guiding and supporting this research effort. I would further like to thank my committee members Dr. Keith L. McRoberts, Dr. Tom J. Rudolphi, Dr. Arthur E. Oldehoeft, and Dr. Roger Berger for their helpful advice and contribution of time to this effort.

I am also indebted to Dr. Michael Wald for his continuous support of the exchange program between the Iowa State University and the Fachhochschule Hamburg which enabled me to study at this school.

Last but not least I would like to express my deepest thanks to my parents for their support and encouragement throughout my years of graduate study.

BIBLIOGRAPHY

1. E. W. Kent and M. O. Shneier, "Eyes for Automation", IEEE Spectrum, 23, No. 3, 37-45 (1986)

2. A. Pugh, Robot Sensors Volume I: Vision (Springer Verlag, New York, 1986)

3. V. Petersen and J. C. Even, "A Microcomputer-Based Binary Vision System", International Journal of Applied Engineering Education, 1, No. 6, 405-413 (1985)

4. V. Petersen and J. C. Even, "A Microcomputer Vision and Ranging System", Robotics Engineering, 8, No. 7, 26-28 (1986)

5. S. L. Supernault, "PC-Based Image Processing", Robotics Age, 7, No. 3, 20 (1985)

6. A. Weilert, "Vision Systems using Microprocessor Based Microcomputers", in Robotics and Factories of the Future, edited by S.N. Dwivedi (Springer Verlag, New York, 1984)

7. E. L. Hall, "A PC-Based Machine Vision System", in Vision'86 Conference Proceedings, p. 10-1, (SME, Dearborn, MI, 1986)

8. R. E. Keil, "Survey of Off-The-Shelf Imaging Systems", in Proceedings of the Third Annual Applied Machine Vision Conference, p. 1-10 (SME, Dearborn, MI, 1984)

9. L. Congiliara, "The Vision Industry: One Year Later", Machine Vision Association of SME Vision'86 Conference (SME, Dearborn, MI, 1986)

10. G. J. Agin and R. O. Duda, "SRI Vision Research in Advanced Automation", in Proceedings of the Second USA-Japan Computer Conference (AFIPS & IPSJ, Montval, NJ, 1975)

11. R. Cunningham, "Segmenting Binary Images", Robotics Age, 3, No. 4, 4-19 (1981)

12. D. Edson, "Vision Systems for Bin-Picking Robots Increase Manufacturing Options", Mini-Micro Systems, 17, No. 9, 177-184 (1984)

13. J. Matill, "The Bin of Parts Problem and the Ice Box Puzzle", Technology Review, 78, No. 7, 18-19 (1976)

14. A. C. Englander, "Edge Detection Techniques for Industrial Machine Vision", in Vision'86 Conference Proceedings, p. 5-85 (SME, Dearborn, 1986)

15. I. E. Abdou and W. K. Pratt, "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors", Proceedings of the IEEE, 67, No. 5, 753-763 (1979)

16. K. S. Fu and J. K. Mui, "A Survey of Image Segmentation", Pattern Recognition, 13, No. 1, 3-16 (1981)

17. M. D. Levine, "Feature Extraction: A Survey", Proceedings of the IEEE, 57, No. 8, 1391-1407 (1969)

18. W. Pratt, Digital Image Processing (John Wiley & Sons, New York, 1978)

19. H. Katz, "Region Maker", Byte, 12, No. 1, 144-153 (1986)

20. R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis (John Wiley & Sons, New York, 1973)

21. B. Lipkin and A. Rosenfeld, Picture Processing and Psychopictories (Academic Press, New York, 1970)

22. L. G. Roberts, "Machine Perception of Three-Dimensional Solids", Optical and Electro-Optical Information Processing (M.I.T Press, Cambridge, Mass., 1965)

23. B. L. Bulloch, "Finding Structures in Outdoor Scenes", in Pattern Recognition and Artificial Intelligence (Academic Press, New York, 1976)

24. M. Hueckel, "An Operator which Locates Edges in Digitized Pictures", Journal of the ACM, 18, No. 1, 634-647 (1971)

25. C. D. McIlroy, R. Linggrad, and W. Monteith, "Edge Detection in Real Time", in Applications of Digital Image Processing VII, Vol. 504, p. 445 (SPIE, Bellingham, Wash., 1984)

26. D. H. Ballard and C. M. Brown, Computer Vision (Prentice-Hall, Inc., Engelwood Cliffs, New Jersey, 1982)

27. R. K. Miller, Intelligent Robots (SEAI Institute and Technical Insight Inc., Ft. Lee, NJ, 1983)

28. Ming-Kuei Hu, "Visual Pattern Recognition by Moment Invariants", IRE Transactions on Information Theory, IT-8, No. 2 179-187 (1962)

29. R. Y. Wong and E. L. Hall, "Scene Matching with Invariant Moments", Computer Graphics and Image Processing, 8, No. 1, 16-24 (1978)

30. J. A. Coppola, "Determining Part Orientation in Machine Vision Systems via Fourier Transform", in Vision'86 Conference Proceedings, p. 5-107 (SME, Dearborn, MI, 1986)

31. R. Bolles, "Robust Feature Matching through Maximal Cliques", SPIE Technical Symposium on Imaging Applications for Automation (1979)

32. S. J. Gordon and W. P. Seering, "Accuracy Issues in Measuring Quantized Images of Straight-Line Features", in IEEE International Conference on Robotics and Automation, Vol. 2, p. 931 (Los Angeles, CA, 1986)

33. T. Pavlidis and S. L. Horowitz, "Segmentation of Plane Curves", IEEE Transactions on Computers, C-23, No. 8, 860-870 (1974)

34. U. Ramers, "An Iterative Procedure for the Polygonal Approximation of Plane Curves", Computer Graphics and Image Processing, 1, No. 3, 244-256 (1972)

35. H. Freeman, "On the Encoding of Arbitrary Geometric Configurations", IEEE Transactions on Computers, EC-10, No. 2, 260-268 (1961)

36. D. Ballard, "Strip Trees: A Hierarchical Representation for Curves", Communications of the ACM, 24, No. 5, 310-321 (1981)

37. T. Pavlidis Algorithms for Graphics and Image Processing (Computer Science Press, Rockville, MD, 1982)

38. N. Ayache, "A Model-Based Vision System to Identify and Locate Partially Visible Industrial Parts", in IEEE Proceedings Computer Vision and Pattern Recognition, p. 492 (Washington DC, June 19-23, 1983)

39. B. Bhanu and J. C. Ming, "Recognition of Occluded Objects: A Cluster-Structure Algorithm", Pattern Recognition, 20, No. 2, 199-211 (1987)

40. J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing Partially Hidden Objects", IEEE International Conference on Robotics and Automation, p. 48 (Los Angeles, CA, 1983)

41. J. L. Turney, T. N. Mudge, and R. A. Volz, "Solving the Bin of Parts Problem", in Vision'86 Conference Proceedings, p. 4-21 (SME, Dearborn, MI, 1986)

42. M. W. Koch and R. L. Kashyap, "A Vision System to Identify Occluded Industrial Parts", in IEEE International Conference on Robotics and Automation, p. 55 (Los Angeles, CA, 1985)

43. M. W. Koch and R. L. Kashyap, "Using Polygons to Recognize and Locate Partially Occluded Objects", IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-9, No. 4, 483-494 (1987)

44. R. C. Bolles and R. A. Cain, "Recognizing and Locating Partially Visible Objects: The Local Feature Method", The International Journal of Robotics Research, 1, No. 3, 57-82 (1982)

45. S. D. Roth, "Recognizing Parts that Touch", Robotics Age, 7, No. 6, 30-35 (1985)

46. J. W. McKee and J. K. Aggarwal, "Computer Recognition of Partial Views of Curved Objects", IEEE Transactions on Computers, C-26, No. 8, 790-799 (1977)

47. W. A. Perkins, "A Model-Based Vision System for Industrial Parts", IEEE Transactions on Computers", C-27, No. 2, 126-143 (1978)

48. B. Bamieh and R. J. De Figueiredo, "A General Moment-Invariants/Attributed-Graph Method for Three-Dimensional Object Recognition from a Single Image", IEEE Journal of Robotics and Automation, RA-2, No. 1, 31-41 (1986)

49. G. Stockman, S. Kopstein, and B. Sanford, "Matching Images to Models for Registration and Object Detection via Clustering", IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-4, No. 4, 229-241 (1982)

50. T. F. Knoll and R. C. Jain, "Recognizing Partially Visible Objects using Feature Indexed Hypotheses", IEEE Journal of Robotics and Automation, RA-2, No. 1, 3-13 (1986)

51. S. Berman, P. Parikh and C. S. Lee, "Computer Recognition of two Overlapping Parts Using a Single Camera", Computer, 18, No. 3, 70-80 (1985)

52. W. E. Grimson and T. Lozano-Perez, "Localizing Overlapping Parts by Searching the Interpretation Tree", IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-9, No. 9, 469-482 (1987)

53. U. Montanari, "A Note on Minimal Length Polygonal Approximation to a Digitized Contur", Communications of ACM, 13, No. 1, 41-57 (1970)

54. D. H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes", Pattern Recognition, 13, No. 2, 111-122 (1981)

55. Y. Shirai, "Recognition of Real World Objects using Edge Cue", in Computer Vision Systems, edited by A. Hanson and E. Riseman (Academic Press, New York, 1978)

56. V. Petersen, B. L. Heemsbergen, and J. C. Even, "An Edge Tracking Process for Microcomputer-Based Vision Systems", to be published in Dr. Dobb's Journal of Software Tools

57. J. M. Wilf, "Chain Code", Robotics Age, 3, No. 2, 12-18 (1981)

58. R. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures", Communications of the ACM, 15, No. 1, 11-15 (1972)

59. H. L. Beus and S. H. Tiu, "An Improved Corner Detection Algorithm Based on Chain-Coded Plane Curves", Pattern Recognition, 20, No. 3, 291-296 (1987)