MODEL-BASED VISION SYSTEM FOR SCENES CONTAINING MULTIPLE PARTS

W. A. Perkins

Computer Science Department
Research Laboratories
General Motors Corporation
Warren, Michigan 48090

ABSTRACT

A model-based computer vision system has been developed which can determine the position and orientation of several parts in the same scene. The parts can be of the same type or of different types. Furthermore, the system can work with visually noisy scenes and some occlusion. Abstract two-dimensional models are formed by having the system view the parts under favorable conditions. Parts are found by matching the abstract models to the scene data, after it has been highly organized.

I. INTRODUCTION

The present work extends an earlier system, which found single parts [Perkins, 77], into a system which can recognize and determine the position and orientation of several parts in the same scene. The geometric configuration of these parts must allow only a limited number of stable positions because the program uses two-dimensional models. The program works with gray-level picture data and is not sensitive to visual noise. It can find objects which are partially out of the field of view or occluded by other objects.

Although some block -world programs [Grape, 73; Griffith, 73] could handle scenes containing several blocks and occlusion, the present system can handle a much greater variety of objects with more complex shapes. However, since the three-dimensional problem is more difficult for these objects than for polyhedra, the present program requires that the parts have only a few stable positions and be at a known distance from the camera.

Several other systems have been developed for locating industrial parts in similar environments [Tsuji and Nakamura, 75; Yachida and Tsuji, 75; Baird, 76]. Each system has its strong points or advantages: The system of Tsuji and Nakamura can find objects from oblique views if an ellipse or trapezoid is present in the surface being viewed; The system of Yachida and Tsuji forms models by viewing the objects with an operator interactively pointing out features; The system of Baird has simplicity, is relatively fast, but does not use any specific models.

The present system differs from the others mostly in that it performs more detailed image analysis (which is similar to Shirai's system [Shirai, 75]) before matching with models. Its strong points are that it can handle partial views and occlusion; and its models are formed by viewing the objects.

Multiple objects in the field of view are located as follows: The program is shown objects individually, and it forms a model of each. Then a scene, composed of an unknown number of objects, is shown to the program; it is asked to find one or all of the objects in the scene. (The program is not expected to find objects which it has not previously seen.)

The program puts the model and image data in a highly organized form [Perkins, 77] which is essential for any reasonable efficiency in matching several models to a complex scene. The basic data structure is a set of connected curves (straight lines and circular arcs) called "concurves" [Perkins, 77]. These concurves usually form the physical boundaries of the object. Only a few concurves are needed to describe a model.

To find an object, the program must match model concurves to image concurves. For five objects there may be 500 possible combinations of model and image concurves, and only 10 correct ones. Since only a few of the combinations need to be tested in detail, it is important that the program order the search process. The program orders the combinations from most likely to least likely by comparing general properties of the concurves. This ordered list of combinations is used for deciding both the order in which models will be matched and the order in which concurves for a particular model will be matched.

For a particular model the program proceeds down the list of combinations using one or two combinations at a time to determine a tentative transformation from model to image coordinates. After the program has determined a possible location for a model, it can verify or reject this choice by comparing a representation of the model with the image data.

If a part is found, the program stores information about it (such as its location) and marks any image concurves which were used up in the matching process. If the program is asked to find <u>all</u> parts, it will continue matching each model against the available image data until it fails to find one of each model after testing a reasonable number (six times the number of model concurves) of combinations.

II. METHOD

The programs are written in PL/I and run on an IBM 370/168 Computer. Digitized pictures (256 x 256 x 32 intensity levels) are obtained with a closed-circuit TV camera [Baird, 75].

We shall illustrate the method by following the analysis of the program for the ideal case (a high-contrast scene with dark parts on a white sheet of paper) shown in Fig. la. Similar analyses for visually noisy conditions are presented in Section III.

The five models shown in Fig. 2 were formed by having the program view the parts under favorable conditions as described in [Perkins, 77]. These five castings are a universal yoke, connecting rod, compressor body, bracket, and gear blank.

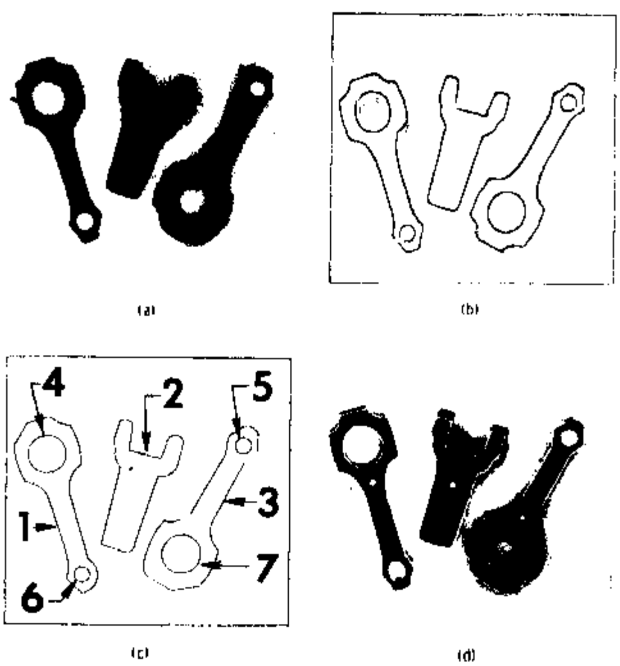


Fig. 1. High contrast scene with two connecting rods and a universal yoke, (a) Digitized Picture, (b) Edge Points, (c) Concurves. (d) Models superimposed on gray-level picture.

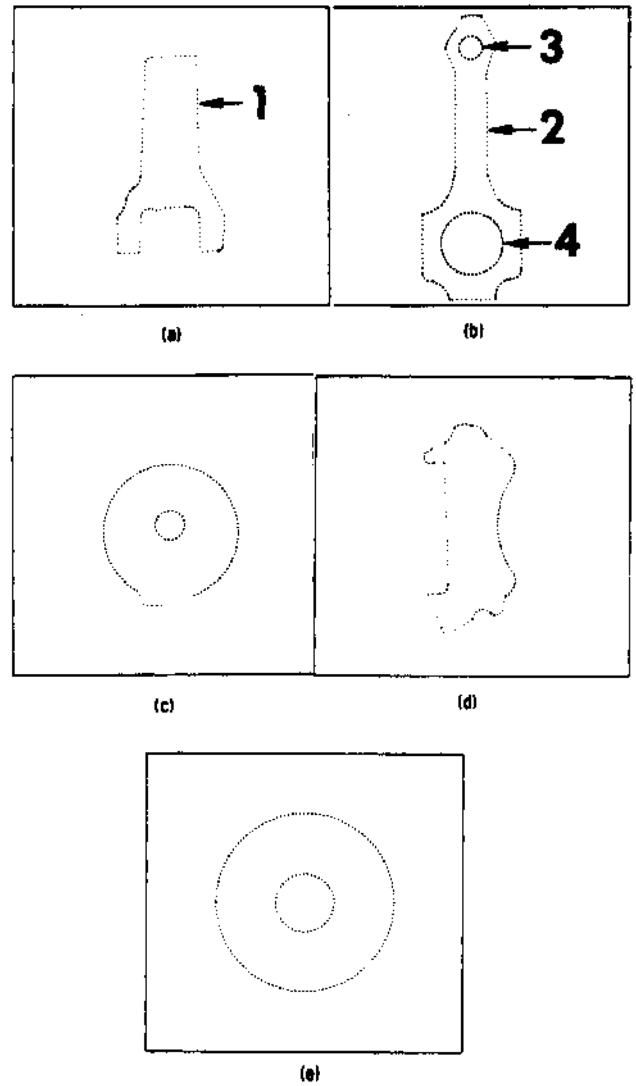


Fig. 2. Five models. (a) Universal yoke, (b) Connecting rod. (c) Compressor body, (d) Bracket, (e) Gear blank.

At the top level a model has the following set of properties: 1) name, 2) rotational symmetry (n-fold value), 3) number of concurves (connected boundary curves), A) number of multisectors (a set of vectors, equally spaced along the concurves with directions perpendicular to the concurves), and 5) pointers to concurve and multisector data.

The above properties refer to the model and are not modified in processing images. Tokens are used to describe particular instances of an object in an image [Anderson and Bower, 73]. Tokens have all the general properties of the model with some specific properties that depend upon the image data such as: location (x,y,θ), scale factor, and a pointer to a particular set of multisectors.

Each of these multlsectors has a number indicating whether it found a match with the image data and, if so, which image concurve it matched.

The concurves themselves have a set of properties which are used in comparing model and Image:

1) Simple description (circle, arc, complex curve,...);
2) Total length or radius of arcs;
3) Magnitude of total angular change;
4) Number of straight lines;
5) Number of arcs;
6) Bending energy (curvature function [Chang, 76]).

And If both concurves are closed, the system uses in addition:

7) Intensity direction (if given for model);
8) Area inside outer border;
9) Area of Internal holes;
10) Compactness (area / perimeter*); and
11) Ratio of minimum to maximum moment of inertia.

Properties such as these can be used for classification [Duda, 76]. The program calculates a likelihood for each combination of one model concurve and one image concurve from a comparison of the above properties. Table I shows the seven combinations for the universal yoke model and the image concurves of Fig. Ic and the 21 combinations for the connecting rod model and the image concurves of Fig. Ic.

From an average of the highest likelihood for all concurves of each model, the program can obtain a likelihood for each model being present in the scene. This is used to set the order in which models are matched against the image data. This ordering can save considerable time, especially if the program has several models and is asked to recognize which object is in the scene (classification).

The likelihood for a universal yoke being in the scene is taken to be the highest likelihood for its one concurve being in the scene, namely 97 (see Table I). The likelihood for a connecting rod being in the scene is taken to be the average of the highest likelihoods for all its concurves ■ (108 +94+87) / 3 - 96.

If we compare all five models with the analyzed image data of Fig. 1, we obtain the ordered list of likelihoods shown in Table II. If the program were asked to find <u>all</u> objects. It would start with the most likely model and proceed down the list (see Table II). Then for each model it would proceed down lists such as those of Table I. After the program finds an object in the scene (which involves matching one or two combinations of model and image concurves to determine the transformation from model to image coordinates and a verification process [Perkins, 77]), the program marks image concurves that are used up. This reduces the choices for the next search process and keeps the program from wasting time with consumed image data.

Identical objects cause some new problems. Suppose we are trying to find all connecting rods in Fig. 1. If no precautions were taken, the

TABLE I.

Likelihood of Different Combinations

Combinations	Likelihood	Model No.	Image No.
1	97	1 (yoke)	2
2	62	1 (yoke)	1
3	53	1 (yoke)	3
4	-49	1 (yoke)	4
5	-51	1 (yoke)	7
6	-62	1 (yoke)	6
7	-62	1 (yoke)	5
1	108	2 (rod)	3
2	107	2 (rod)	1
3	94	4 (rod)	7
4	87	3 (rod)	6
5	84	4 (rod)	4
6	78	3 (rod)	5
7	60	2 (rod)	2
8	19	3 (rod)	4
9	17	3 (rod)	7
10	11	4 (rod)	5
11	10	4 (rod)	6
12	-44	4 (rod)	2
13	-47	4 (rod)	1
14	-47	4 (rod)	3
15	-48	2 (rod)	7
16	-49	3 (rod)	2
17	-51	3 (rod)	3
18	-51	2 (rod)	4
19	-53	3 (rod)	1
20	-62	2 (rod)	6
21	-62	2 (rod)	5

rABLE II.

Likelihood of Different Models

Likelihood	Model Name
97	universal yoke
96	connecting rod
76	bracket
59	gear blank
52	compressor body

program would keep finding the same connecting rod over and over again. To avoid this undesirable occurrence, the program utilizes two changes that it makes after it finds a connecting rod: 1) the information that It stores about this token, and 2) the information concerning concurves that have been used up. In determining tentative transformations on subsequent searches for connecting rods, the program works only with image concurves that have not been consumed. After the program has determined a tentative location for an object in the scene, this location is compared with that of previously found tokens. If all of the coordinates (x,y,0) of this new location are within certain limits of the coordinates of any token, the tentative model transformation is rejected.

For the scene of Fig. 1 and the five models of Fig. 2, the program first looked for and found a universal yoke. After storing the token information and marking the consumed image concurve, the program tried to find another universal yoke and failed. Next it found the two connecting rods. At this point all the image concurves were used up so it was unnecessary for it to try matching any more of the combinations. Thus the program worked with only 9 of the 63 possible combinations (one to find the universal yoke, six to find that a second yoke was not present, one to find each of the two connecting rods). This is an ideal case, but the same kind

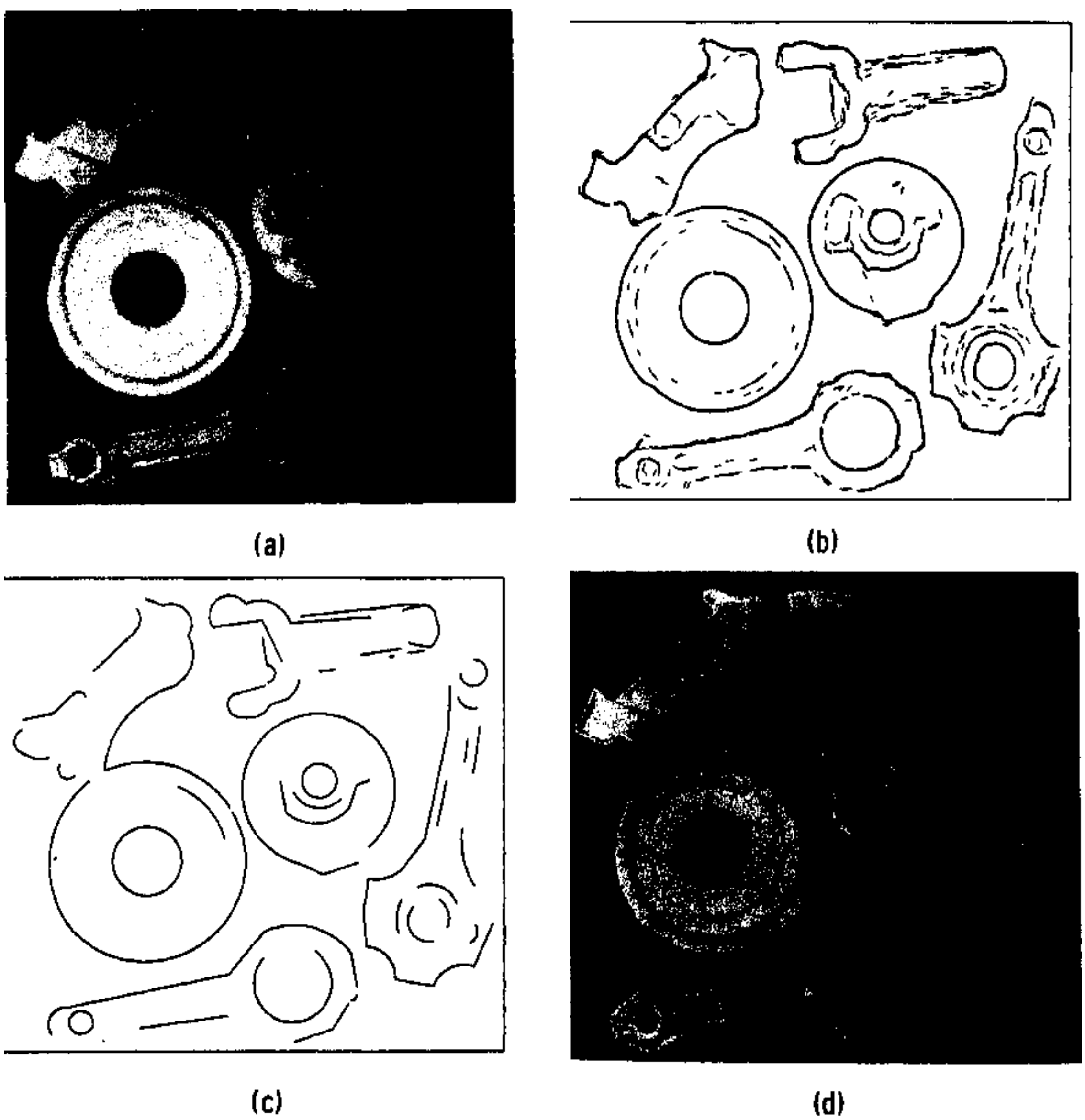


Fig. 3. Six parts on a conveyor belt, (a) Digitized Picture. (b) Edge Points. (c) Concurves. (d) Models superimposed on gray-level picture.

of economy occurs in visually noisy scenes. The CPU time for finding the three parts in Fig. 1 was 18.2 seconds (image analysis, 16.9 sec; matching, 1.3 sec).

III. RESULTS

The results for visually noisy conditions are presented in this section. In these scenes the dark gray parts were placed on a dark conveyor belt typical of those used in industrial environments. The lighting consisted of room lights only (fluorescent bulbs in the ceiling). The scenes chosen illustrate the capabilities of the system to find: 1) all parts in the field of view with several different parts and an unknown number of each, 2) objects which are partially out of the field of view, and 3) several occluded parts in the same scene.

A. Finding Several Parts in the Field of View

The program was asked to find all the parts in Fig. 3a using its five models shown in Fig. 2. Since there is an unknown number of each particular type of part, the program must do a reasonable search for one more of each type than exist in the scene. (This means eleven searches in the case of Fig. 3.) We have also made the problem a little more difficult by using two connecting rods that differ from the one used in forming the model. One has a square-shaped

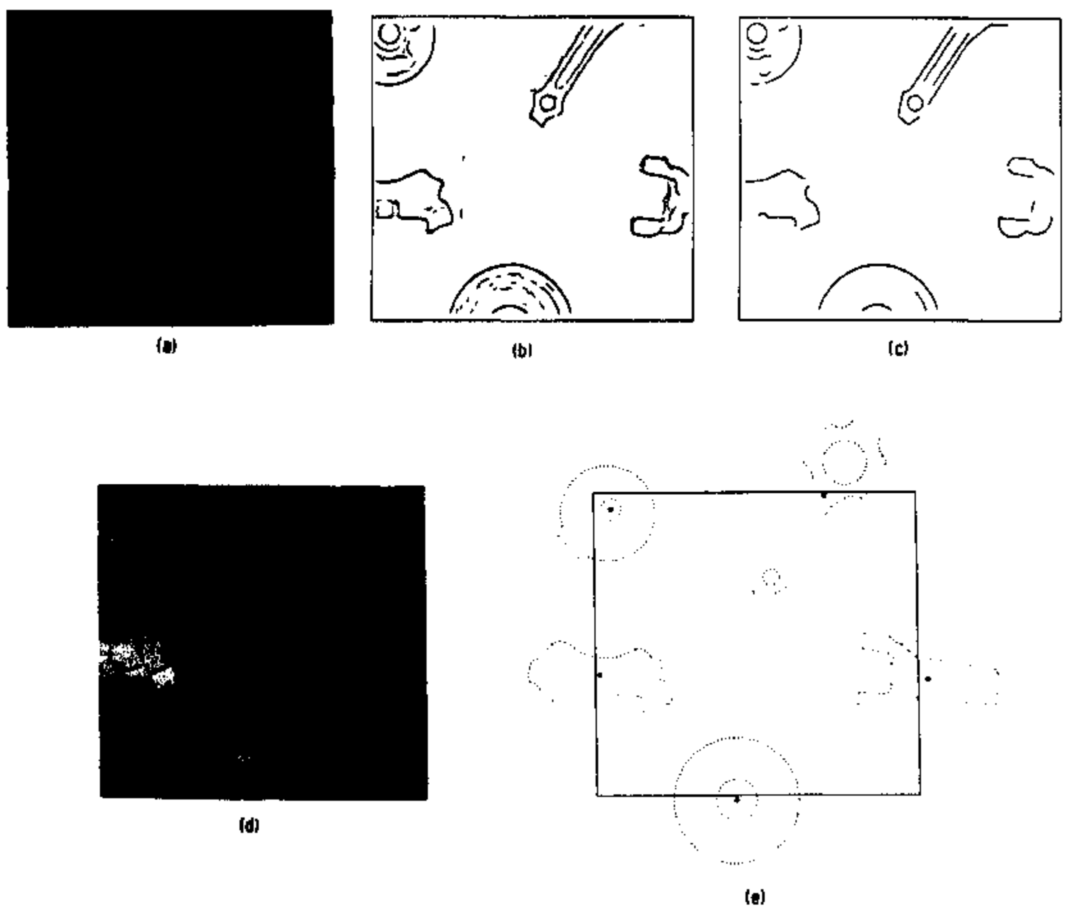


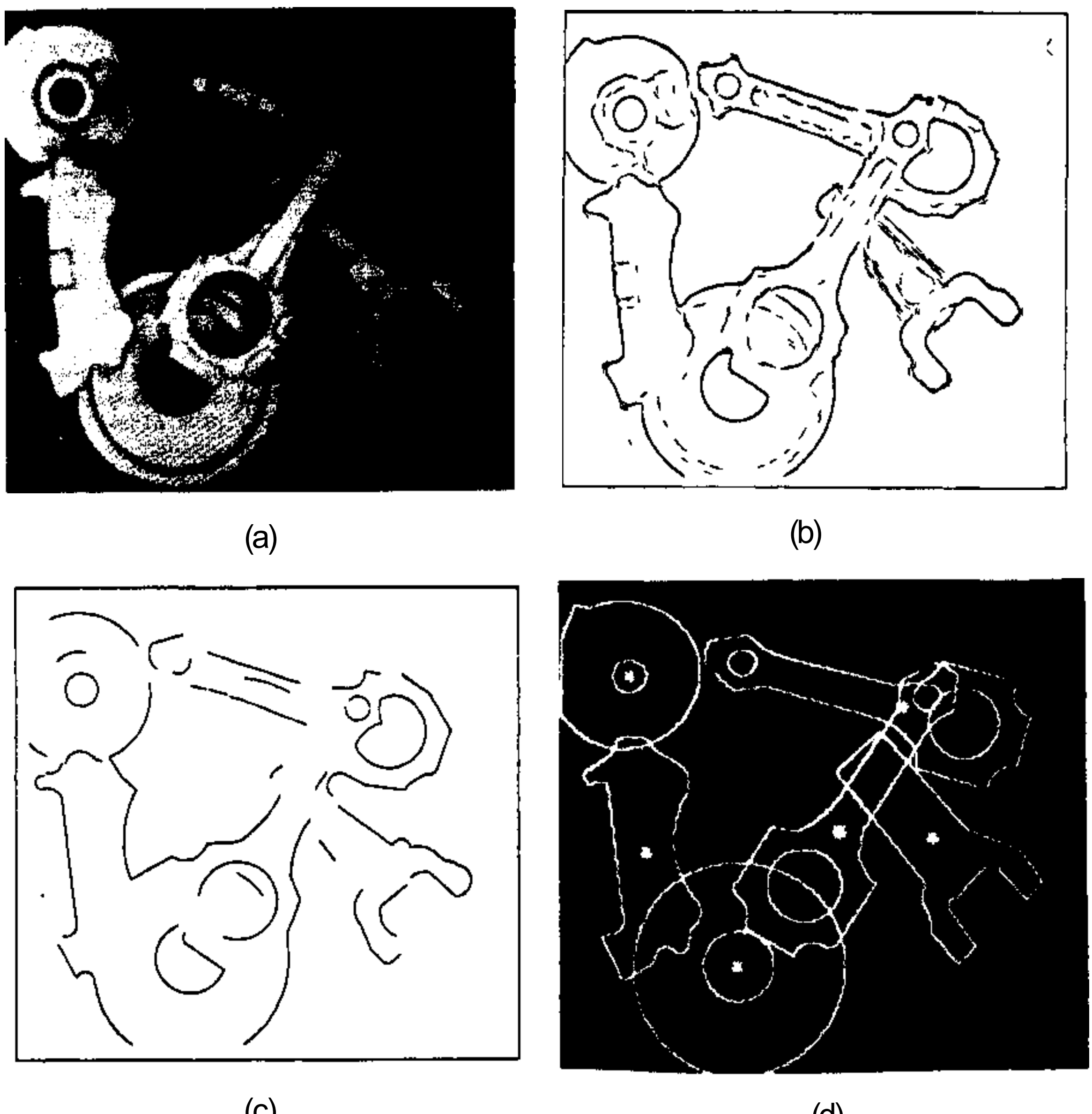
Fig. 4. Partial view of five parts on conveyor belt. (a) Digitized Picture, (b) Edge Points, (c) Concurves. (d) Models superimposed on gray-level picture, (e) Display of models.

"sprue" projection while the other is slightly different in shape and does not have the large hole punched out.

Figure 3a shows the digitized picture while Fig. 3b shows the edge points obtained using the Hueckel operator [Hueckel, 73] and Fig. 3c shows the analyzed image data (31 concurves). The results of matching the five models to the analyzed image data are shown in Fig. 3d. The CPU time for finding the six parts was 42.7 seconds (image analysis, 32.8 sec; matching, 9.9 sec).

B. Finding Objects from Partial Views

The program is again asked to find all the parts, but the parts in Fig. 4a are only partially inside the field of view. The edge points and the analyzed image data are shown in Figs. 4b and 4c respectively with the results of matching shown in Figs. 4d and 4e. This figure clearly illustrates that the matching techniques employed by the program [Perkins, 77] work well even with partial image data. During the verification process the program takes account of the fact that an object



(a) (b) (c) (d)

Fig. 5. Six parts on conveyor belt with considerable occlusion, (a) Digitized Picture. (b) Edge Points. (c) Concurves. (d) Models superimposed on gray-level picture.

is partly outside the field of view (some of the transformed multisectors lie outside the viewing window) in accepting a match. The CPU time for finding the five parts was 22.2 seconds (image analysis, 15.3 sec; matching, 6.9 sec).

C. Finding Occluded Parts

Figure 5 shows an image with considerable occlusion. The digitized picture of Fig. 5a is transformed to edge points (Fig. 5b) and then to concurves (Fig. 5c). Note that some of the connected curves are boundaries of several different parts. The results of the matching process are shown in Fig. 5d. This figure illustrates that the matching techniques work in the presence of serious background interference. The CPU time for finding the six parts was 31.6 seconds (image analysis, 25.4 sec; matching, 6.2).

IV. CONCLUSIONS

We have demonstrated a vision system which can find the position and orientation of all parts in a scene, even if they are partly outside the field of view or partially occluded. This system was created by adding some high-level organization to a system which found single parts [Perkins, 77].

An attempt was made to develop an efficient system by having the matching occur at a high-level and by using an intelligent control structure to order the search process. This method tends to make the image-analysis time longer and the matching time shorter, resulting in shorter total times for complex scenes.

For most of these scenes there could have been a further reduction in matching times if groups of concurves had been recognized as separate objects [Baird, 76]. However, if the parts are closely packed or there is occlusion (see Fig. 5), it is not possible to separate the different parts until they have been recognized.

The method of having a finite number of models and matching them to the image data until they have all failed or the image data is used up is satisfactory for most applications. However, if the number of models is large, the method could be extended by using a context-directed search, such as in Group Technology [Opitz, 70], with the image data being converted to a number that points to models which have similar properties.

ACKNOWLEDGEMENTS

I wish to thank L. Rossol and G. G. Dodd for their encouragement and support during the development of this system.

REFERENCES

Anderson, J. R. and Bower, G., Human Associative Memory. Winston and Sons, Washington, D. C. (1973).

Baird, M. L., Olsztyn, J. T., Perkins, W. A., and Rossol, L., "The GM Research Labs' Machine Perception Project," SIGART Newsletter No. 55, pp. 12-13 (1975).

Baird, M. L,, "Sequential Image Enhancement Technique for Locating Automotive Parts on Conveyor Belts," General Motors Research Laboratory Publication GMR-2293 (1976).

Chang, T. L., "On Similarity Measures for Oriented Line Patterns," Proceedings of the Third International Joint Conference on Pattern Recognition. Coronado, California, pp. 208-210 (1976).

Duda, R., Kremers, J., and Nitzen D., "Automatic Part Classification," in Fifth Report of Exploratory Research in Advanced Automation, Stanford Research Institute, pp. 7-22 (1976).

Grape, G. R., "Model Based (Intermediate-Level) Computer Vision," Stanford University Artificial Intelligence Laboratory Memo AIM-201 (1973).

Griffith, A. K., "Edge Detection in Simple Scenes Using A Priori Information," IEEE Trans. Comput., 22, pp. 371-381 (1973).

Hueckel, M. H., "A Local Visual Operator Which Recognizes Edges and Lines," J. ACM 20, No. 4, pp. 634-647 (1973). Also see Hueckel, M. H., Erratum, J. ACM 21, No. 2, p. 350 (1974).

Opitz, H., "A Classification System to Describe Workpieces," Pergamon Press (1970).

Perkins, W. A., "A Model-Based Vision System for Industrial Parts," General Motors Research Laboratories Report GMR-2410 (1977).

Shirai, Y., "Edge Finding, Segmentation of Edges and Recognition of Complex Objects," Fourth International Joint Conference on Artificial Intelligence, pp. 674-681, Tbilisi, Georgia, USSR (1975).

Tsuji, S. and Nakamura, A., "Recognition of an Object in a Stack of Industrial Parts," Fourth International Joint Conference on Artificial Intelligence, pp. 811-818, Tbilisi, Georgia, USSR (1975).

Yachida, M. and Tsuji, S., "A Machine Vision for Complex Industrial Parts with Learning Capability," Fourth International Joint Conference on Artificial Intelligence, pp. 819-826, Tbilisi, Georgia, USSR (1975).