Title	Object Recognition System for Incomplete Image Data : Application of Coded Boundary Representation (Report II)(Physics, Process, Instruments & Measurements)
Author(s)	Inoue, Katsunori; Fukuda, Shuichi; Ohkubo, Masashi et al.
Citation	Transactions of JWRI. 1992, 21(1), p. 69-75
Version Type	VoR
URL	https://doi.org/10.18910/9675
rights	
Note	

Object Recognition System for Incomplete Image Data†

—Application of Coded Boundary Representation (Report II)—

Katsunori INOUE*, Shuichi FUKUDA, Masashi OHKUBO***, Tong QIN******

Abstract

This study aims to build an object recognition system that uses the Modified Coded Boundary Representation (MCBR) method[1] to store shape data in computers. We have shown the MCBR method and how to use this method for the recognition system fundamentally and have shown that the features of the MCBR method were suitable to build an object recongnition system[2]. This paper describes the whole system scheme and features. The system consists primarily of two parts, which are called the Image Processing Part and the Shape Understanding Part. The role of the former is to extract segments from the original data. The role of the latter is to reconstruct the shape using the segments.

KEY WORDS: (3-K System), (Shape Recognition), (Modified Coded Boundary Representation), (Data Structure of Shape Model)

1. Introduction

Integrating the computer graphics (or the image processing) with the knowledge processing has been thought as a hot direction in the computer applications. A kind of integrated knowledge processing system, specialized in 3-K (Know How, Know Why, Know What), is paid attentions by the industrial field. The so called 3-K system, shown in **Fig. 1,** is focused on the interfaces among the knowledge processing, the data processing and the geometry processing. Its goal is to provide a system with the abilities of the problem solving, the learning and the explaining in the industrial design applications.

Taking the idea of the 3-K system, the work presented here aims at the experiment on the practical techniques between the knowledge processing and the computer graphics.

The system presented here is divided into two parts: Image Processing and Shape Understanding. The former is written by C language, and the latter by Prolog language. Its scheme is shown in **Fig. 2.**

The function of Image Processing Part is as follows:

(1) simplifying the original image obtained from a camera into a binary-thin outline image;

(2) transforming the outline image to its Hough Transformed lines;

(3) extracting out image-relative line segments from Hough lines;

(4) integrating the segments to the shape line segments.

The function of Shape Understanding Part is:

(1) simple understanding;

(2) compensating understanding.

The connection of the knowledge processing with the

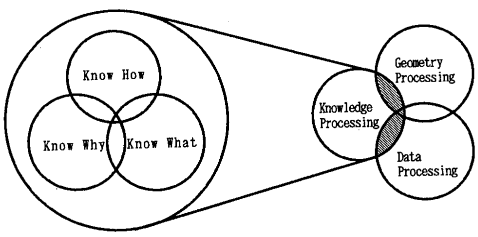


Fig. 1 3-K system and concept of this study

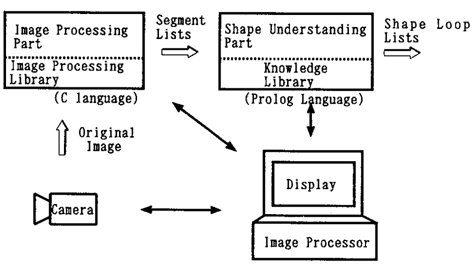


Fig. 2 Scheme of the recognition system

† Received on May 6, 1992
* Professor
** Professor, Tokyo Metropolitan Institute of Technology
*** Research Associate
**** Assistant Professor, Harbin Institute of Technology, China.

Transactions of JWRI is published by Welding Research Institute of Osaka University, Ibaraki, Osaka 567, Japan

image processing system provides a way of analyzing the inference results that are relative to images, so it makes the results more reasonable. And the system provides surroundings for the inference explaining and results showing. It makes the results believable, and easy to be accepted.

2. Image Processing Part

The role of the Image Processing Part is to make the lists of line segments of the shape from the original image data picked up with a camera. The procedure of this part is shown in **Fig. 3.**

In the image simplifying routine, the system executes the processes shown in the following to the original image:

(1) changing density
(2) differentially conversing
(3) dynamically binary conversing
(4) thinning

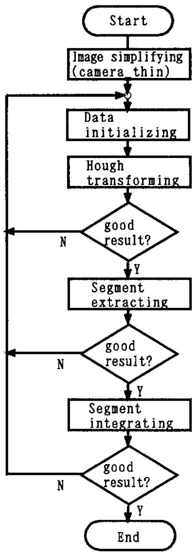


Fig. 3 Procedure of Image Processing Part

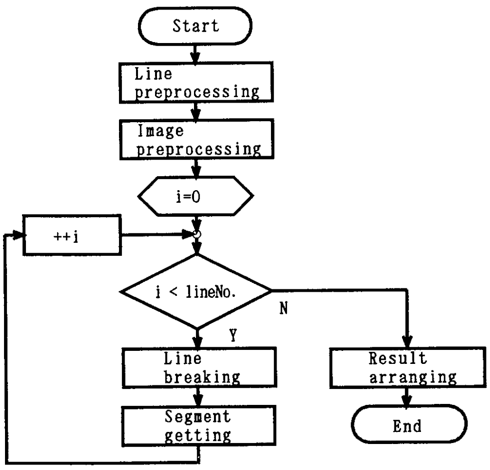


Fig. 4 Flow chart of segment extracting

(5) rejecting small points

After simplifying, Hough Transformation is executed and the line components are extracted. Then the number of extracted lines is examined, because too many or too few lines confuse next routine and the threshold number, by which the number of extracted line is changed, is defined. Some Hough lines that are of few relations with outline image and some noises on the line are rejected to extract successful segments in this routine.

2.1 Segment extracting

The flow chart for the segment extraction is shown in **Fig. 4.**

In the line preprocessing routine, the system rejects some of Hough lines that have a little relation with the outline image. In the next routine, noises on the out line image are rejected. In the line breaking routine, all the crossed points of one Hough line (it is crossed by all other Hough lines, except a small angle that less than a compiling constant number) are obtained. In the segment getting routine, the useful segments from the lines are picked out as follows. Then the relationship rates $RR(S)$, between Hough line segment S and outline image, is used. First, $p(x, y)$ is a point on a segment S of Hough line. And $pl(x,y)$ is a point beside the point $p(x, y)$, it is defined as

$$pl(x, y) \in p(NEAR,x,y) \qquad (1)$$

, where

$NEAR$ is a compiling constant, and should belong to {1, 4,8}.

$$P(1,x,y)=\phi$$

$$P(4,x,y)=\{p(x+1, y), p(x, y+1), p(x-1, y), p(x, y-1)\}$$

$$P(8,x,y)=P(4, x,y)\cup\{p(x+1, y+1), p(x-1, y+1), p(x+1, y-1), p(x-1, y-1)\}$$

And $G(p)$ is the graylevel of the outline image on the point $p(x,y)$, then the relativity $r(p)$ on the point $p(x,y)$ is defined as:

$$r(p)=\begin{cases} 1 & \text{if } G(p)=0 \\ 0.5 & \text{if } G(p)=0 \text{ but } G(p1)\neq 0 \\ 0 & \text{if } G(p)\neq 0 \text{ and } G(pl)\neq 0 \end{cases} \qquad (2)$$

The relativity $R(S)$ of the segment S is defined as

$$R(S)=\sum_{ponS} r(p) \qquad (3)$$

Furthermore, the relative rate is defined. A segment $S(ps,pe)$ starts from $p(xs,ys)$, ends at $p(xe,ye)$. Then pixels $pix(S)$ is as:

$$pix(S)=\max\{ | xe-xs| , | ye-ys| \}$$
$$(pix(S)>0) \qquad (4)$$

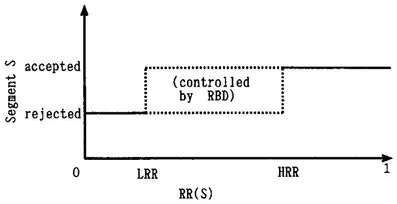


Fig. 5 Decision method for segment S

And then the Relative Rate $RR(S)$ is defined as:

$$RR(S)=\frac{R(S)}{pix(S)} \quad (0<RR(S)<1) \tag{5}$$

Figure 5 shows how the system decides whether it accepts the segment S or rejects. *HRR* is the higher boundary of the relative rate, *LRR* is the lower boundary of the relative rate and *RBD* is the boundary of relativity in Fig. 5.

If $R(S)>HRR$, S is accepted. If $R(S)<LRR$, S is rejected. And in case $LRR<RR(S)<HRR$ and $R(S)>RBD$, *then* S is accepted and else is rejected. When the segment is shorter than the *RBD*, it is called as a very short segment, else as a long segment. For each accepted long segment if there are some very short segments connecting to its ends, it will be extended by not more than *ENDS* very short segments where *ENDS* is a compiling constant.

2.2 Segment integrating

The shape line segments using the segment extracted from Hough lines are reconstructed in the segment integration routine. **Figure 6** shows their examples. If two segments l_i and l_j are in parallel or approximately in parallel, very near each other, the two segments are overlapped or closely in series, then they are considered to be one cluster.

Figure 7 shows the relation between clusters. These clusters are considered that they are connected at the samepoint and the ends of these clusters are extended.

2.3 Problems

The advantage of the 'segment unit' is that it can compensate any wrong information inside a segment. For an example shown in **Fig. 8**. the segment **ab** (or **ac**)in Fig. 8(b) is processed as the basic unit (or smallest unit) so that the wrong information between **ab** (or **ac**) in Fig. 8(a) is compensated as shown in Fig. 8(c). However, it is failed in an example of **Fig. 9**, because there is no Hough line across points **a, c.** The line crossed **ab** is picked out as a segment **bd** instead of segment **ba.** Figure 9(c) is a little far from the outline image.

On the other hand, the strategy about very short segments includes the following basic rules:

- all the very short segments between two accepted long segments are accepted
- all the very short segments between two rejected long segments are rejected

In most case, these rules are working very well, but they may be failed to recognize some images with high noise in very sensitive positions. For example in **Fig. 10**, the segment **ab** is entirely covered by noise lines obtained from **N**

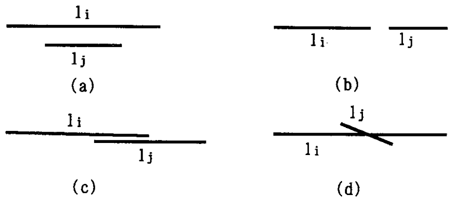


Fig. 6 Local relation between l_i and l_j

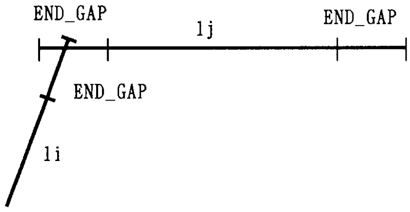
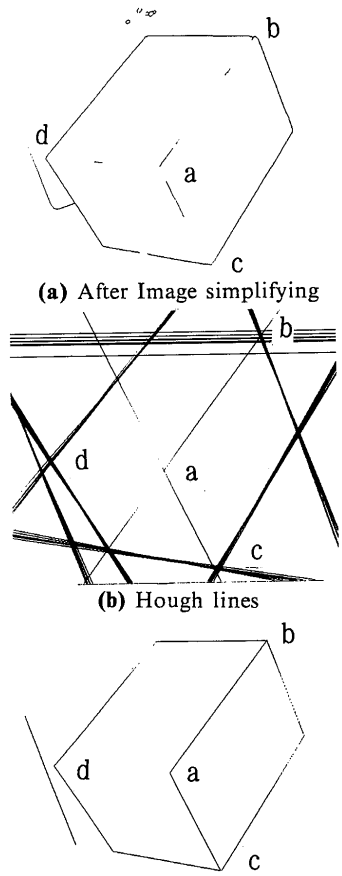


Fig. 7 Positional relation between segments which considered to connect

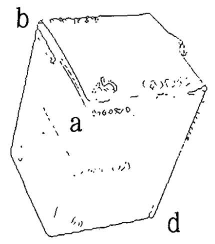


(a) After Image simplifying

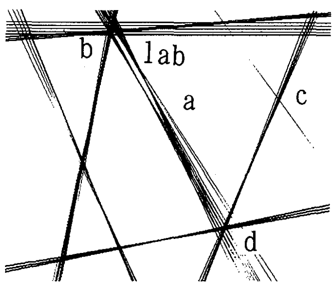
(b) Hough lines

(c) After Segment integrating

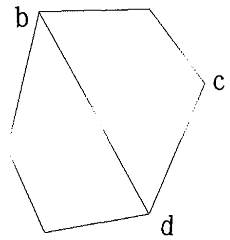
Fig. 8 Succeeded example about the 'segment unit'



(a) After Image simplifying



(b) Hough lines



(c) After Segment integrating

Fig. 9 Failed example about the 'segment unit'

part, so that this segment is missed.

The Shape Understanding Part plays its role to solve these problems correctly.

3. Shape Understanding Part

The Shape Understanding Part reconstructs and recognizes the shape with the segment cluster data obtained from the Image Processing Part. The data obtained from the Image Processing Part are not completely correct. In other words, the data sometimes include some excess information and sometimes don't include any necessary information. So, in this part, the system can delete some noisy information and add some necessary. We'll discuss about this strategy concretely, and especially about the recognition of the cubic object.

3.1 Data structure using in the Shape Understanding Part

The data structure obtained from the Image Prossing Part is expressed in the form of:

$$\text{line}(N_i,\ D_i,\ L_i,\ [Sx_i,\ Sy_i,\ Ex_i,\ Ey_i])$$

, where

N_i : identity line segment number
D_i : directional code (defined in **Fig. 11**) $(0.0 \leqq D_i < 8.0)$
L_i : segment length
(Sx_i, Sy_i): start point coordinate
(Ex_i, Ey_i): end point coordinate

Using these data, the system reconstructs some loops to recognize the shape. The loop data structure, using MCBR method, is expressed in the form of:

$$\text{n\_list}(L_i, [\text{n}_{i1},\ \text{n}_{i2},\ \cdots\cdots\cdots])$$

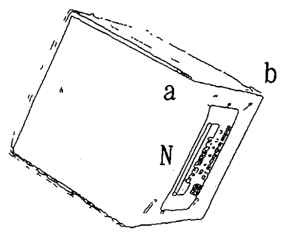
$$\text{s\_list}(L_i, [\text{p}_{i1},\ \text{p}_{i2},\ \cdots\cdots\cdots])$$

$$\text{e\_list}(L_i, [\text{p}_{i2},\ \text{p}_{i3}, \cdots\cdots \text{p}_{i1}])$$

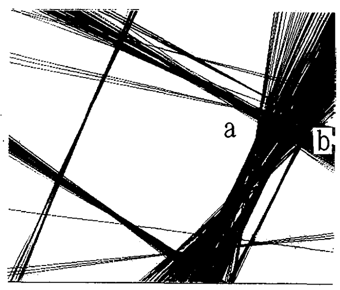
$$\text{d\_list}(L_i, [\text{d}_{i1},\ \text{d}_{i2},\ \cdots\cdots\cdots])$$

$$\text{l\_list}(L_i, [\text{l}_{i1},\ \text{l}_{i2},\ \cdots\cdots\cdots])$$

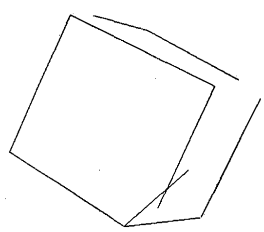
where,



(a) After Image simplifying

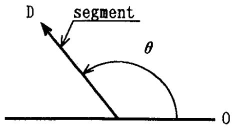


(b) Hough lines



(c) After Segment integrating

Fig. 10 Influence of the noises



$$D = \mathrm{remains}\left(\frac{\theta}{45.0}\right)$$
$$(0.0 \leqq D < 8.0)$$

Fig. 11 Directional code of MCBR method

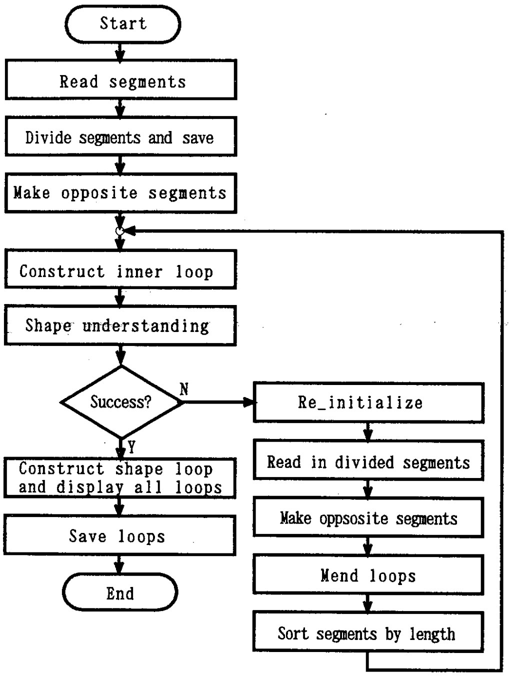
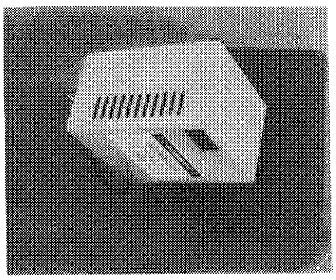
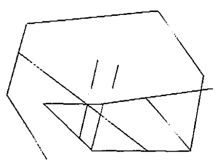


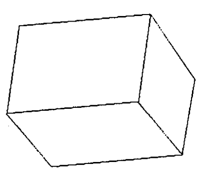
Fig. 12 Procedure of Shape Understanding Part



(a) Original image



(b) Result of Image Processing Part



(c) Result of Shape Understanding Part
Fig. 13 Example 1 of Object recognition

n_list : identity line segment number (n_{ij}) list
s_list : segment start point name (p_{ij}) list
e_list : segment end point name list
d_list : directional code (d_{ij})list
l_list : segment length (l_{ij}) list
L_i : identity loop number or name (if L_i='shape', they show the outside loop of the shape)

The end element of e_list is it as the first element of s_list. It means these lists show a closed loop.

3.2 Strategy

The Shape Understanding Part's role is almost to construct loop clusters. This procedure is shown in **Fig. 12.**

In the divide segments routine, the lines that cross another line are divided at crossing point. However, since some lines are divided owing to noise lines, they should be reinstated as temporal lines in database in their original forms. And if a temporal line is useful in some shape loop, its divided lines will be deleted, and itself will be remembered as an in-temporal line. In the make opposite segments routine, the system makes opposite segment for each segment. In the construct inner loop routine, the system connects the segments and constructs the closed inner loops which form is shown above, and in the construct shape loop routine, constructs the outer loop to recognize the shape. If the system fails to recognize the shape, it initializes the data and tries to recognize again. This means that the data sometimes can be interpreted as multiple shape.

3.3 Special knowledge to recognize cubic shape

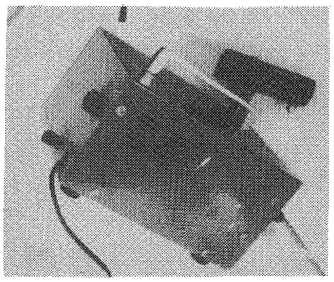
Now, we discuss about the rules of cubic shape. It is a cube if:

(a) there are 3 parallelogram loops which connect each other; or
(b) there are 2 trapezium loops which connect each other; or
(c) there is one rectangle loop; or
(d) there is a parallelogram loop and a neighbor line for shape guessing and the guessed result is good enough.

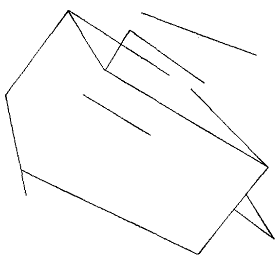
These rules seem to include some forcibly idea, but these are appropriate measure, if the scene includes the shape outline completely.

4. Examples in Application

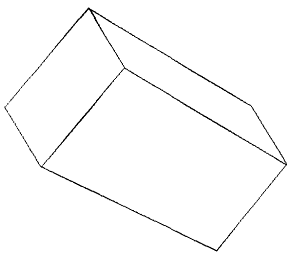
Figures 13 and **14** show the examples of recognition using this system, (a) are original images, (b) are the results of Image Processing Part and (c) are the results of Shape Understanding Part. They show that the system can extract the cubic shapes tolerable correctly in spite of not so good original images.



(a) Original image

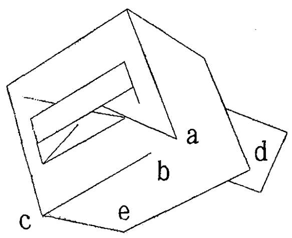


(b) Result of Image Processing Part

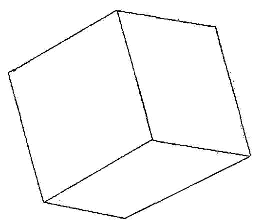


(c) Result of Shape Understanding Part

Fig. 14 Example 2 of object recognition

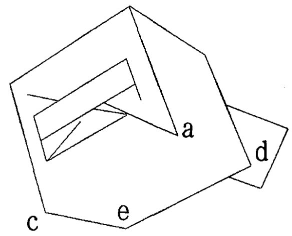


(a) After Segment integrating

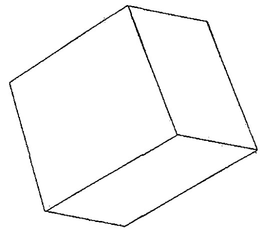


(b) Result of recognition

Fig. 15 Influence of the mending rules



(a) After Segment integrating



(b) Result of recognition

Fig. 16 Influence of the mending rules

5. Discussions

Right now, it is very often that not only one result is given by the Shape Understanding Part. It is not so good in case of a machine using the results. The way to deal with it is to modify the special knowledge with more restrictive rules, or to make a higher level of knowledge for further processing the results.

And the present loop-mending knowledge used in the Shape Understanding Part does not consider the perspective problem in visual images of 3-D objects. If considering this problem, a surface of a cube should not be an exact parallelogram, but with some change in shape. That is a reason why the present compensated results are sometimes not very exact.

In mending rules, two lines in nearly parallel and almost equal in length may be as a base for mending. The 'nearly' and 'almost' are the sensitive field for the mending. If a line is the one of missing some information but still in the sensitive field, then it will result in an unsatisfactory result. For example, in **Fig. 15**(a) line **ac** missed the information from **a** to **b,** but was still almost in same length with line **de,** then resulted in Fig. 15(b). On the other hand, if line **bc** is deleted as in **Fig. 16**(a) then the result was much better, Fig. 16(b).

Above problem will be resolved when the perspective is considered. The key perspective problem may be to resolve relations between focus distances, and object positions, with the extents of shape changing.

6. Conclusion

In this system, the data flow from the Image Processing Part to the Shape Understanding Part is one way. The system that is required better results, it is need that the system has the intimate relation between these parts. In other words, when the Shape Understanding Part can't recognize the shape using only the data obtained from the Image Processing Part, the system can require the additional data to Image Processing Part. Though, this recognition system spends so much time. So, it is fundamental problem whether the system will be made to recognize correctly or speedily.

References

1) Shuichi Fukuda: "A New Shape Model: Coded Shape Representation", The 3rd Computational Mechanics Conference, JSME, No. 900−69, 1990, 183−184, in Japanese.

2) Inoue K., Fukuda S., Ohkubo M. and Qin T.: "Recognition of Objects Using Modified Coded Boundary Representation (MCBR) Method", JWRI., Vol. 20, No. 2(1991), 49−52.