MODEL BASED OBJECT RECOGNITION USING MODIFIED CODED BOUNDARY REPRESENTATION(MCBR) METHOD

Katsunori Inoue,
Welding Research Inst.of
Osaka University
11-1 Mihogaoka,Ibarakai-shi
Osaka,567,Japan

Shuichi Fukuda,
Tokyo Metropolitan Inst.
of Technology
6-6 Asahigaoka,Hino-shi
Tokyo,191,Japan

Masashi Okubo,
Welding Research Inst.of
Osaka Univ.
11-1 Mihogaoka,Ibaraki-shi
Osaka,567,Japan

Tong Qin
Harbin Inst.
of Technology

ABSTRACT

This study aims at building the object recognition system which uses the Coded Boundary Representation (CBR) method to store shape data in computers. The CBR method was originally developed as a simple 2-D CAD model, and it can represent a shape as a simple form of lists and operate the shape qualitatively in a non_numerical way(1). However, it can't represent a shape in detail because of these features, it is difficult to use the method in the object recognition system without modification.

Therefore this paper describes the modification of the CBR method (which is called the MCBR method), the structure of shape models stored in computers, and shows the 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 using Hough transformation. The role of the latter is to reconstruct the shape using the segments in the MCBR method. Then we demonstrate the effectiveness of the MCBR method for object recognition with some illustrative examples.

INTRODUCTION

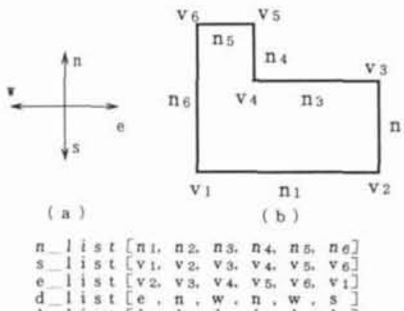
Many model based recognition systems which are performed until now, use the B-reps method to represent the shape stored in computer(2). The B-reps method has been developed as 3-D solid models for CAD systems. On the oter hand, the system which is descrived here uses the CBR method. The fundamental idea of the CBR method is the same as that of the B-reps method, except on two points.

One point is that the CBR method represents a shape in the form of lists. Therefore the CBR method has such advantages that it is very easy to get rotated, mirrored or conjugated shapes by simple list operations.

Another point is that the CBR method utilizes the idea of Freeman's directional codes which are frequently used in the field of image processing. To express it more concretely, one of the lists shows the direction of the each line segment. Therefore, it is very simple to extract or compare the topological features of shapes using the directional codes lists. Because of these, it is considered that this method is suitable to develop a flexible model based object recognition system. For example, it will be very efficient to make aspect graphs because that the CBR represents the topological features of shape very simply and can decrease the number of graphs.

MCBR METHOD

Figure 1 shows the CBR method which uses the 4 directional codes and represents a shape with the edge name list(n_list), the edge start and the edge end vertex name lists(s_list,e_list), the edge length list(l_list) and the directional code list(d_list) of those edges. In this method, the shape loop is in the counterclockwise. The features of this CBR method include that it is easy to operate the lists for carrying out the figure processing such as rotating, mirroring, decomposing and composing; and it is also easy to get the topological features of a shape using the d_list.



```
n_list [n1, n2, n3, n4, n5, n6]
s_list [v1, v2, v3, v4, v5, v6]
e_list [v2, v3, v4, v5, v6, v1]
d_list [e , n , w , n , w , s ]
l_list [l1, l2, l3, l4, l5, l6]
```
(c)

Fig.1 CBR method
(a) Directional codes
(b) Shape example
(c) Shape lists in CBR method

To apply the CBR method to the recognition of objects, it has to be improved. The CBR method's directional codes have been expressed only in 4 values: s,e,n,w. Although the clear and simple shape representation is easy for computers to recognize the features of shapes and can carry out the shape operations in a non-numeric way, it is not convenient for recognizing objects if used without any modification.

Therefore, in the MCBR method, the directional codes are extended to real numbers ($0.0 \leq D < 8.0$) in Fig.2.

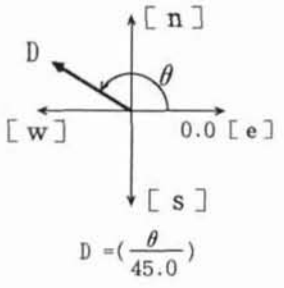


Fig.2 Directional code in MCBR method

Because of this change, the processing such as mirroring, rotating, etc., which used elements of d_list, has also to be modified. Figure 3 shows the 'rotation' as the example of figure processing in the MCBR method. In the CBR method, rotation angle is defined with every 90 degrees. On the other hand, in the MCBR method, it is defined with all degree. With the edge direction D and the rotation angle θ, the edge direction D' obtained after rotation can be described as:

$$D' = Rm(D + \frac{\theta}{45.0}, 8.0) \qquad (1)$$

where,

$$Rm(x,y) = \begin{cases} x & if(0.0 \leq x < y) \\ Rm(x-y,y) & if(x \geq y) \\ Rm(x+y,y) & if(x < 0.0) \end{cases} \qquad (2)$$

In this way, with the MCBR method, it is possible to represent a line in any direction. It is also possible to express position relations between several lines using the equalities or the inequalities. Thus, a more flexible figure processing can be defined.

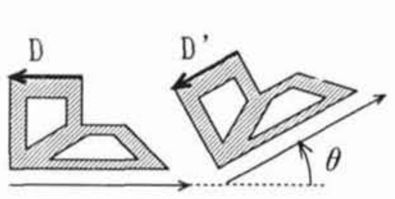


Fig.3 Figure processing 'Rotation'

SYSTEM

The recognition system presented here is divided into two parts: Image Processing and Shape Understanding Part.

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;
(5) sending the data to Shape Understanding Part.

The function of Shape Understanding Part is:
(1) understanding simply;
(2) understanding using mending rules.

Image Processing Part

The role of the Image Processing Part is to make the lists, in the MCBR method, 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.4.

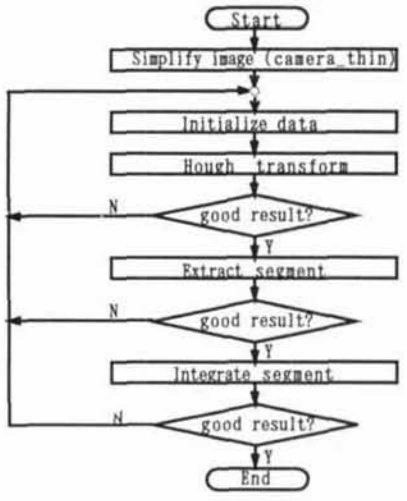


Fig.4 Procedure in Image Processing Part

After simplifying, Hough transformation is executed and the line components are extracted. In the 'Extract segment' routine, the system rejects some of Hough lines that have a little relation with the outline image. And 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 and cut in pieces. In the 'Integrate segment' routine, the useful segments from thes lines are picked out.

In this part, 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 $p1(x,y)$ is a point beside the point $p(x,y)$, it is defined as

$$p1(x,y) \in P(NEAR, x, y) \tag{3}$$

,where

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

$$\begin{aligned} P(1,x,y) &= \emptyset \\ P(4,x,y) &= \{p(x+1,y), p(x,y+1), \\ &\quad p(x-1,y), p(x,y-1)\} \\ P(8,x,y) &= P(4,x,y) \cup \\ &\quad \{p(x+1,y+1), p(x-1,y+1), \\ &\quad p(x+1,y-1), p(x-1,y-1)\} \end{aligned}$$

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

$$r(p) = \begin{cases} 1 & if(G(p) \neq 0) \\ 0.5 & if(G(p) = 0 \ but \ G(p1) \neq 0) \\ 0 & else \end{cases} \tag{4}$$

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

$$R(S) = \sum_{p on S} r(p) \tag{5}$$

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:

$$\begin{aligned} pix(S) &= max\{|xe - xs|, |ye - ys|\} \\ &\quad (pix(S) > 0) \end{aligned} \tag{6}$$

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

$$RR(S) = \frac{R(S)}{pix(S)} \quad (0 < RR(S) < 1) \tag{7}$$

The system decides whether it accepts the segment S or rejects by this number $RR(S)$.

The shape line segments obtained from the segments extraction routine are reconstructed in the segment integration routine. If two segments are in parallel or approximately in parallel, very near each other, then they are considered to be one cluster. And if two clusters are considered that they are connected at the same point then the ends of these clusters are remained.

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 don't include any necessary information. So, in this part, the system can delete some noisy information and add some necessary.

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

$line(N_i, D_i, L_i, [S_{xi}, S_{yi}, E_{xi}, E_{yi}])$

,where

N_i	: identity line segment number
D_i	: directional code (defined in Fig.2) $(0.0 \leq Di < 8.0)$
L_i	: segment length
(S_{xi}, S_{yi})	: start point coordinate
(E_{xi}, E_{yi})	: end point coordinate

Now here is an example of an image of rectangular solid obtained from a camera. Figures 5(a)(b) are the two patterns from the image which has a couple of face loop. We show the structure of the shape model stored in computers in Fig.5. At first each face loop on the rectangular solid is assigned with a quadrilateral type, then it is represented in the MCBR method as:

$$\begin{array}{llllll} n\_list & (Loop_i, & [N_{i1}, & N_{i2}, & N_{i3}, & N_{i4} \ ]) \\ s\_list & (Loop_i, & [V_{i1}, & V_{i2}, & V_{i3}, & V_{i4} \ ]) \\ e\_list & (Loop_i, & [V_{i2}, & V_{i3}, & V_{i4}, & V_{i1} \ ]) \\ d\_list & (Loop_i, & [D_{i1}, & D_{i2}, & D_{i3}, & D_{i4} \ ]) \\ l\_list & (Loop_i, & [L_{i1}, & L_{i2}, & L_{i3}, & L_{i4} \ ]) \\ & (i = 1, 2, & 3, ...) & & & \end{array}$$

,where

'$-N_{ij}$' means opposite to 'N_{ij}'

$Loop_i$: identity loop number or name

(if $Loop_i$ = 'shape', they show the outside loop of the shape)

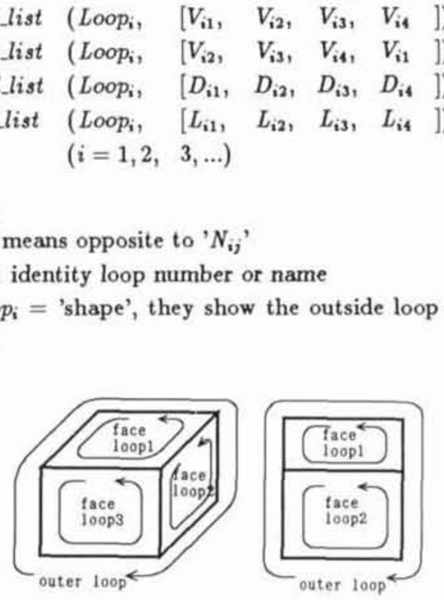


Fig.5 Two image patterns of rectangular solid
(a) 3 loops pattern
(b) 2 loops pattern

All the loops are the tetragons and they should be nearly in the shape of parallelograms. The conditions of one tetragonal loop being a parallelogram can be described as:

$$\begin{aligned} D_{i1} &= Rm(D_{i3} + 4.0, 8.0) \\ D_{i2} &= Rm(D_{i4} + 4.0, 8.0) \end{aligned} \tag{8}$$

In Figs.5(a)(b) the numbers of face parallelogram loops are 3 and 2 correspondingly, and the condition of a rectangular solid is that these parallelograms must contact each other. These conditions can be described as:

$$\begin{aligned} &{}^{\forall}i \in [1, 2, 3, ..., I], j = (i+1) \ mod \ I + 1, \\ &{}^{\exists}m, n \ that \ N_{im} \in [N_{ik}]_k, N_{jn} \in [N_{jk}]_k \\ &make \ N_{im} = -N_{jn} \end{aligned} \tag{9}$$

The Shape Understanding Part's role is almost to construct loop clusters. This procedure is shown in Fig.6.

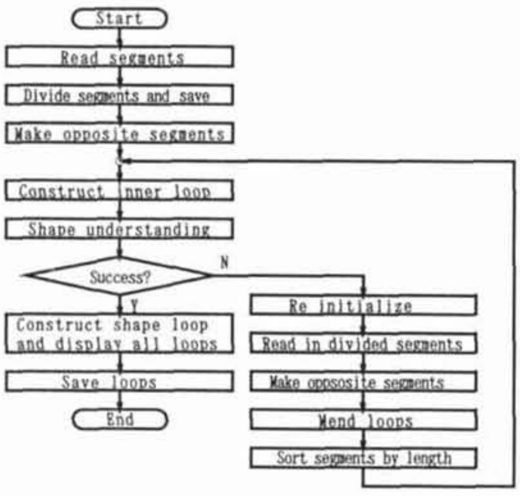
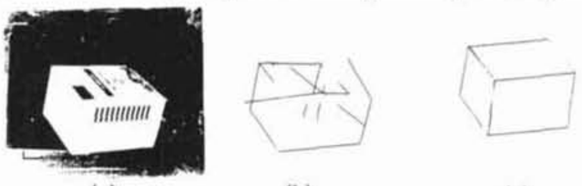


Fig.6 Procedure in Shape Understanding Part

In the divide segments routine, these lines which 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 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, the system 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.

EXAMPLE IN APPLICATION

Figure 7 shows the examples of recognition using this system, (a) is original image, (b) is the result of Image Processing Part and (c) is the result of Shape Understanding Part, (d) is the shape lists in MCBR method. It shows that the system can extract the cubic shapes tolerable correctly in spite of not so good original image.



```
n_list(11, [1, 2, 3, 4]).
s_list(11, [v1, v2, v3, v4]).
e_list(11, [v2, v3, v4, v1]).
l_list(11, [140, 109, 120, 126]).
d_list(11, [1.807721, 2.868190, 5.754900, 6.757030]).
n_list(12, [5, -2, 6, 7]).
s_list(12, [v5, v3, v2, v6]).
e_list(12, [v3, v2, v6, v5]).
l_list(12, [206, 109, 206, 109]).
d_list(12, [4.104836, 6.868190, 0.104836, 2.868190]).
n_list(13, [8, 9, -6, -1]).
s_list(13, [v1, v7, v6, v2]).
e_list(13, [v7, v6, v2, v1]).
l_list(13, [206, 140, 206, 140]).
d_list(13, [0.104836, 1.807721, 4.104836, 5.807721]).
n_list(shape, [-7, -9, -8, -4, -3, -5]).
s_list(shape, [v5, v6, v7, v1, v4, v3]).
e_list(shape, [v6, v7, v1, v4, v3, v5]).
l_list(shape, [109, 140, 206, 126, 120, 206]).
d_list(shape, [6.868190, 0.104836, 4.104836, 2.757030, 1.754900, 0.104836]).
```

Fig.7 Example (d)

CONCLUSION

In this system, the data flow from the Image Processing Part to the Shape Understanding Part is one way. If 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.

Furthermore, the idea of Computer Geometry will be helpful to the recognition, and the MCBR method may be suitable to use this idea.

REFERENCES:

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

2) For example, Masahiko Koizumi, Fumiaki Tomita, Qualitative and Quantitative Matching of Solid Models and Image of 3D Objects, IPIJ Technical Rept. on Computer Vision 54-5(1988), (in Japanese).