コンピュータビジョンと
イメージメディア　105－3
（1997．5．15）

曲線の部分マッチング手法に基づくモデルの学習と物体認識

賈縉
静岡大学 理工学研究科
jia@cs.inf.shizuoka.ac.jp

阿部圭一
静岡大学 情報学部
abe@cs.inf.shizuoka.ac.jp

はじめに

本研究では、異なる三次元物体のいくつかの二次元ビューから物体のモデルを自動抽出し、実際の複雑なシーンから物体を認識する方法を提案した。モデルの抽出と物体の認識は曲線の部分マッチング手法に基づいて実現した。認識の対象物は部品に分解しやすい、かつ部品間の連結関係を記述しやすいものとなっている。部品の形と部品間の連接関係が似ている異なる三次元物体に対して、曲線の部分マッチング手法を用いて、数少ないモデルで記述することができる。実際のシーンに対して実験して、良い結果を得た。

Learning and recognizing 3D objects by using partial planar curve matching method

Jin JIA
Graduate School of Science and Engineering
e-mail: jia@cs.inf.shizuoka.ac.jp

Keiichi ABE
Dept. of Computer Science
e-mail: abe@cs.inf.shizuoka.ac.jp

Abstract

A scheme for learning and recognizing 3D objects from their 2D views is presented. The scheme proceeds in two stages. In the first stage, we try to learn a model automatically from 2D training images of different objects which have similar shape of components and similar adjacency relation between the components. In the second stage, the generated model is used to recognize the learned object. We use partial planar curve matching method to implement our scheme. We tested the approach on recognizing objects from an image of a complex real scene and we got satisfactory results.

1 Introduction

Learning and recognizing 3D objects is a complex subject in computer vision. A large variety of methods have been proposed for the task of visual object learning and recognizing. Among these methods, three main classes can be distinguished, as summarized by Ullman[1]:

Invariant properties and feature spaces The approach to objects recognition has been to assume that objects have certain invariant properties that are common to all of their views.

Parts and structural descriptions This approach relies on the decomposition of objects into constituent parts and the capturing of the invariant properties which are relations among parts.

Alignment approach The basic idea of this approach is to compensate for the transformations separating the viewed object and the corresponding stored model, and then compare them.

Our approach lies in parts and structural description approach. This approach assumes that each object can be decomposed into a small set of generic components. There are several reasons for us to go along with this approach. Firstly, since we try to generate a model automatically from training examples of several kinds of objects, to find invariant properties is difficult. Secondly, since we deal with non-rigid objects, we can not use alignment approach either.

We try to develop a system which can learn a model of similar objects from 2D training images automatically, and then use the model to recognize the learned object or some objects similar to the learned ones from a complex scene. In our system, we just handle those objects such that they can be decomposed into parts and relations among the parts are easily extracted and easily described. Chairs have been chosen here as objects for processing.

The outline of the system is shown in Figure 1. As shown in Figure 1, inputs of the system in the learning stage are gray level images of chairs. These images are then fed to a preprocessing procedure which consists of segmenting images and putting the results of segmentation into order. Then all 2D images of a 3D object are divided into several clusters. Thus only several 2D clusters are used to describe the object instead of using all images. In the next step, the clusters of different objects are matched each other and a common model (prototype) representing different objects is constructed. This model is used to recognize the objects from an image of a real complex scene.

The rest of this paper is arranged as follows. In section 2, we will describe preprocessing procedure. In section 3 we describe how to divide input images into clusters. We will introduce model construction in section 4 and recognition approach in section 5. Finally, we show some results of experiments.

2 Preprocessing

Preprocessing includes segmentation[2] and putting results of segmentation into order. In Figures 2(a) and 2(b), two segmentation results of an object observed from two directions are shown. From these figures, we can explain why we do not use the results of segmentation directly.

Because of the lighting condition and colors of the object itself, we get different segmentation result. The "difference" means that after matching two results of segmentation, a pair of matched regions may have very different appearance features and the change of appearance features is not due to change of observing directions. For example, the region A in Figure 2(a) matches region A' in Figure 2(b). But they have very different shapes, because regions A' includes parts of the pillar and the leg. For constructing a model from these images, we have to "correct" the results of segmentation.

First we use Wang's method[3] to match two results of segmentation. Two matching results are generated. One is mapping all regions in the first image to the ones in the second image and the other is vise versa. Two operations may be applied to the results of segmentation based on the results of matching: *Splitting* and *Merging*. Between a pair of matched regions, splitting means to divide the larger region into several parts and among generated parts, there is one which is similar to the smaller region. Merging means to merge some parts connecting with the smaller region so that the merged region is similar to the larger one. We use several rules listed bellow for deciding whether merging or splitting should be applied. For two candidate images, S_1 denotes regions in one image and S_2 denotes regions in the other image. Here we assume that the images depict the object observed in two near directions.

Rule 1 The splitting operation is only applied to "stable" matched regions. "Stable" matched regions means those regions that matched each other both in left to right mapping and in right to left mapping.

Rule 2 If features of two matched regions are obviously different, one of them needs to be split. Here we pay attention to those features such as convexity, area, length of major axis and minor axis.

Rule 3 If two regions in one image match the same region in the other image, then these two regions will be merged if the convexity of merged region is not less than the convexity of any region before merged.

Rule 4 If there is a region in one image which does not match any region in the other image, it will be merged with one of regions connecting with it, if the convexity of the merged region is not less than the convexity of any regions before merged. Otherwise the region remains unchanged.

We can notice that *convexity* plays an important role on deciding whether two regions are to be merged or to be split. For this purpose we use a convexity measure which takes a continuous value in $[0,1]$; its definition and the way of calculating convexity can be found in [4].

The merging operation is easy to be implemented relatively, so we discuss splitting operation in detail. The method of cutting a predefined shape from a given region is implemented by using partial planar curve matching (PPCM) method[5]. This method is a way of detecting common parts in two 2D curves. The reason why we can use it in our approach is that the shape of two regions observed from different directions does not change much if the observing direction does not change much. The main idea of PPCM method is to use dynamic programming technique to detect common parts between two planar curves from their total-curvature graphs. For example, there are two regions shown in Figures 3(a) and 3(b). Their total-curvature graphs are shown in Figures 3(c) and 3(d). Curve segment ab in Figure 3(a) matches curve segment pq in Figure 3(b). See [5] for detail.

The cutting of a region is implemented by using PPCM method and feature points extraction method. After finding matched segments in two regions, a transformation of the form shown in Eq.(1), can be decided uniquely by each matched segment.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = s \cdot \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (1)$$

where s is the scaling factor and θ is the rotation angle. Because there may be several matched segments between two regions, so the transformation is not unique. The best transformation for cutting is chosen as follows. Each transformation is applied to all points on one region curve and calculate the difference between two region curves R_1 and R_2 by using Eq.(2). The transformation which gives the least $d(R_1, R_2)$ is used for cutting.

$$d(R_1, R_2) = df(R_1^t, R_2)/s \quad (2)$$

where s is the scaling factor in Eq.(1). R_1^t denotes R_1 after being transformed and $df(X,Y)$ is defined as

$$df(X,Y) = \sum_{x \in X} (\min_{y \in Y} dist(x,y)) \quad (3)$$

$dist(x,y)$ denotes the distance between points x and y. Notice that $d(R_1, R_2) \neq d(R_2, R_1)$. If $d(R_1, R_2) > d(R_2, R_1)$, then R_2 will be split and otherwise R_1 will be split.

Then we extract feature points from the region that is to be split, suppose R_2 here. The result of feature points extraction using N-code method[6] is shown in Figure 4(a). The contour of the region is divided at each feature point into several curve segments. Then we calculate the distance between each curve segment C_i and R_1 by calculating $d(C_i, R_1)$. The segment whose distance to R_1 is obviously large will be split from R_2. The result of splitting is shown in Figure 4(b).

The corrected segmentation results are shown in Figures 2(c) and 2(d). We use corrected segmentation results for further processing.

3 Clustering

In our system, a 3D object is described by its 2D views. We noticed that the shape of parts of an object and the relation between these parts do not change much if the object is observed from two near directions, so we can divide all 2D images into several groups. In each group, we generate a cluster and save it in memory for recognizing. The clustering is implemented by considering both shape change of parts and adjacency relations change between parts.

For each part of the object, we can define measurement of shape change between two views. Let P_1 and P_2 be one part observed from two near directions and define their difference as

$$diff(P_1, P_2) = df(P_1^t, P_2)/s + df(P_2, P_1^t) \quad (4)$$

$df(X,Y)$ is defined in Eq.(3).

The measurement of change of adjacency relations is defined similarly to the relational distance[7] between two views V_1 and V_2. Let $L_1 = \{r_1, \ldots, r_m\}$ and $L_2 = \{l_1, \ldots, l_n\}$ be the relational descriptions among the parts in V_1 and V_2, respectively. Here only "connecting" relation is considered at the current stage. The relational distance between V_1 and V_2 is defined as follows.

$$\begin{aligned} rd(V_1, V_2) &= \textstyle\sum_{i=1}^{m} \Psi(r_i \circ f, L_2) \\ &\quad + \textstyle\sum_{i=1}^{n} \Psi(l_i \circ f^{-1}, L_1) \end{aligned} \quad (5)$$

where f is the mapping function from V_1 to V_2 and f^{-1} is vise versa. In practice, $r_i \circ f$ means mapping two regions which included in relation r_i in V_1 to the regions in V_2. $\Psi(r_i, L_j)$ equals to 0 if relation r_i is also true after being mapped in relation description L_j, equals to 1 otherwise. Thus $\Psi(r_i \circ f, L_2)$ is equal to 0 if two regions included in relation r_i still connecting each other after being mapped from V_1 to V_2.

After defining the measurement of changes of shape and adjacency relation, we can make clusters from 2D images of an object. Here we suppose 2D views of an object is arranged in order of viewing direction. Firstly, if $rd(V_m, V_n)$ is larger than a predefined threshold, then the views from V_m to V_n will be grouped into a cluster. Secondly, if the next equation is satisfied, the views from V_m to V_n will be grouped into a cluster.

$$\sum_{i}^{N} \Upsilon(d(R_{m,i}, R_{n,i})) \times w_i > \alpha \quad (6)$$

where N is the number of matched parts in two views, α is a predefined threshold, $R_{m,i}$ denotes region i in V_m and w_i is the weight for region i. $\Upsilon(d(D_{m,i}, D_{n,i}))$ is equal to 1 if $d(D_{m,i}, D_{n,i})$ is larger than a threshold. At the moment, the weights for all regions are set equal.

An example of obtained groups are shown in Figure 5.

After clustering, let C be a cluster generated from view V_m to view V_n and let P_i be a component of it. We use PPCM method again to generate the shape of P_i in C. First the component in one view is transformed according to (1), here we suppose P_i in V_m is transformed to V_n. Let Cur_m and Cur_n denote the total curvature of P_i in V_m after being transformed and V_n, respectively. The total-curvature of the component in the cluster is calculated simply by Eq.(7).

$$Cur_C = \frac{Cur_m + Cur_n}{2} \quad (7)$$

Then according to Cur_C, the shape of component P_i in the cluster can be generated. See Figure 6 for an illustration.

4 Model generation

Till now, one 3D object is described by several 2D clusters. These clusters can be regarded as a model of the object. We want to extend the model by combining clusters generated from similar objects so that the new model can be used not only to recognize one kind of object, but also to recognize various similar objects. This part of work can be regarded as an extension of R. Basri's work[8], but we deal with more complex cases than he did.

The model generation is also implemented by using PPCM method. After 2D views of each object having been grouped into several clusters, each cluster is described by a hierarchical graph structure. If there is an edge between nodes n_i and n_j, it means two regions stored in the nodes are connecting. A common model is generated by matching among these hierarchical graphs of each kind of objects.

Suppose C_i and C_j denote graphs of two clusters of different objects. Firstly, we start to find the most similar part in C_j for a component P_i stored in C_i by using Eq.(2). The condition of judging whether two nodes are matched well is done not only by examining Eq.(2), but also examining the features of two regions stored in the nodes. We pay special attention to these features such as convexity, length of major axis and minor axis. then we construct adjacency relations. For a pair of matched regions, a new region is generated by using Eq.(7). The adjacency relations between two pairs of matched nodes are stored into model graph by using *or* relation, if there are unmatched nodes between them. See Figure 7 for an illustration.

After a model being generated based on two objects, it can be used to match with other kinds of object. Figure 8 shows a few views of three similar objects, which are used to generate a model. The result of model generation is illustrated in Figure 9. Only those components and adjacency relations which exist in most of objects are stored in the model.

5 Recognition

In this section we describe recognition procedure. The recognition works in two stages. In the first stage, every component stored in the model is identified and in the second stage the entire object is then recognized in terms of distinctive relations among the already identified components.

The input is a gray level image and it is segmented first. Then a component can be identified from the result of segmentation of input image, denoted R, by using PPCM method. Among all regions in R, we chose several potential candidates P_j in R which minimize $d(C_i, P_j)$, where $d(X,Y)$ is defined in Eq.(3) and C_i is the component in the model that we want to detect. After all components stored in the model have been detected, those regions that the relations between them are similar to the relations between components in model is regarded as a result of recognition. How similar are the relations of detected regions to the relations of components in model can be measured by using Eq.(5).

Figure 10 shows the segmentation result of an input image. Figure 11 shows three regions in the image which are most similar to a region stored in the top node of the model. Figure 12 shows result of detecting another component in the middle node of the model. Figure 13 shows the final result.

Reference

[1] S. Ullman, *High-level vision*, The MIT Press, 1995

[2] T. Sugiyama and K. Abe, "Image segmentation using both edge and region information", *Proc. of IAPR Workshop on Machine Vision Application, MVA'94*, pp.127-130, Kawasaki, Japan, 1994.

[3] C. Wang and K. Abe, "Region correspondence by inexact attributed planar graph matching", *Proc. 5th ICCV*, pp. 440-447, 1995.

[4] A. Held and K.Abe, "On approximate convexity", *Pattern recognition letters*, Vol. 15, No.6, pp. 611-618, 1994.

[5] A. Pikaz and I. Dinstein, "Matching of partial occluded planar curves", *Pattern Recognition* Vol.28, No.2, pp. 199-209, 1995

[6] K. Abe, R. Morii, K. Nishida, and T. Kadonaga, "Comparison of methods for detecting corner points from digital curves — a preliminary report", *Proc. 2nd Int'l Conf. on Document Analysis and Recognition*, pp. 854-857, 1993

[7] R. M. Haralick and L.G. Shapiro, *Computer and robot vision*, Chap. 18, Addison-Wesley Publishing, 1993.

[8] R. Basri, "Recognition by prototype", *Proc. Conf. on Computer Vision and Pattern Recognition*, pp. 161-167, 1993.

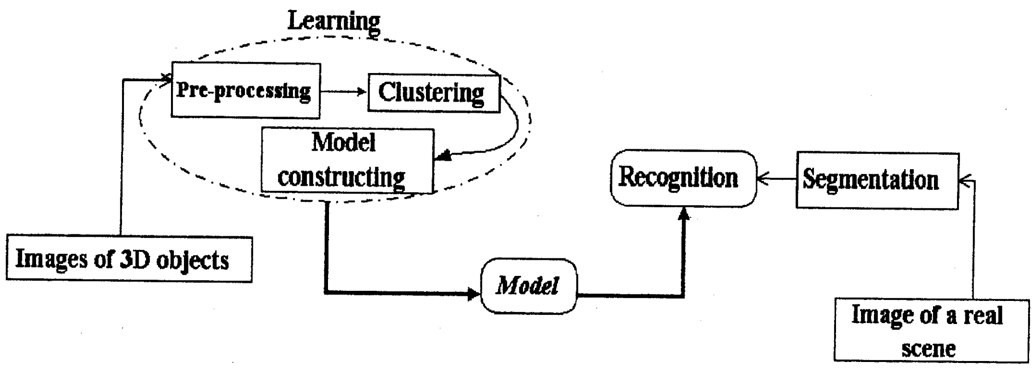

Figure 1: The outline of the system

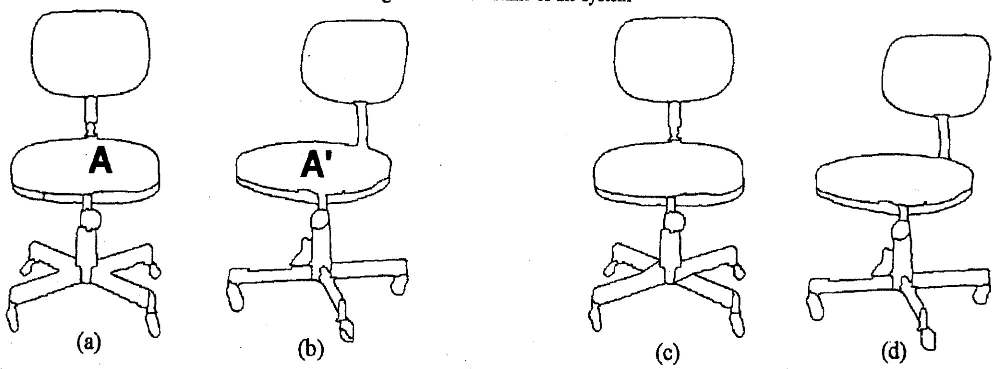

Figure 2: (a) and (b) show the segmentation results; (c) and (d) show the segmentation result after "correction"

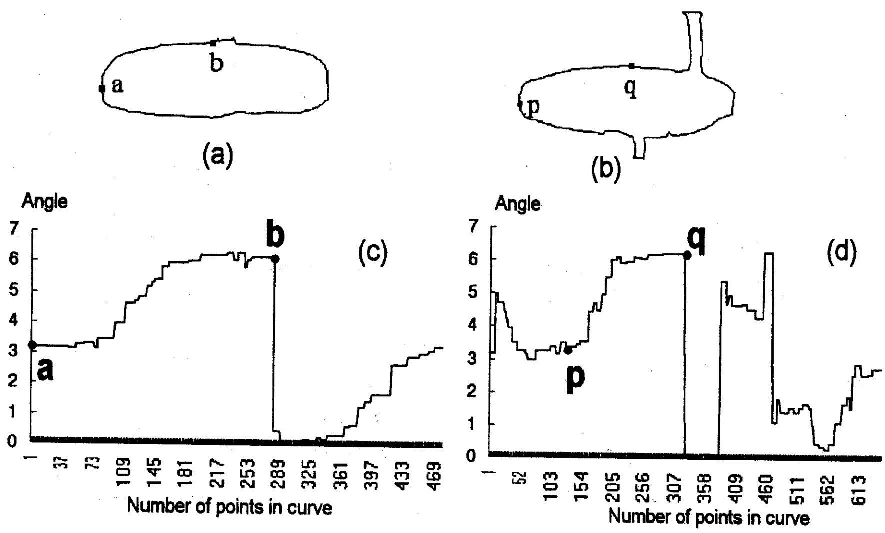

Figure 3: Two regions and their total-curvature graphs. In the graphs, *ab* matches *pq*

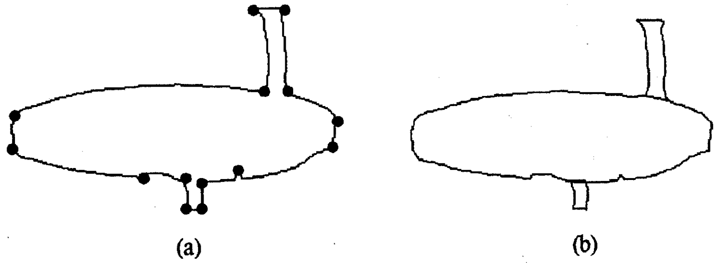

Figure 4: (a) shows result of feature points detecting by using N-code method;
(b) shows result of the region being cut.

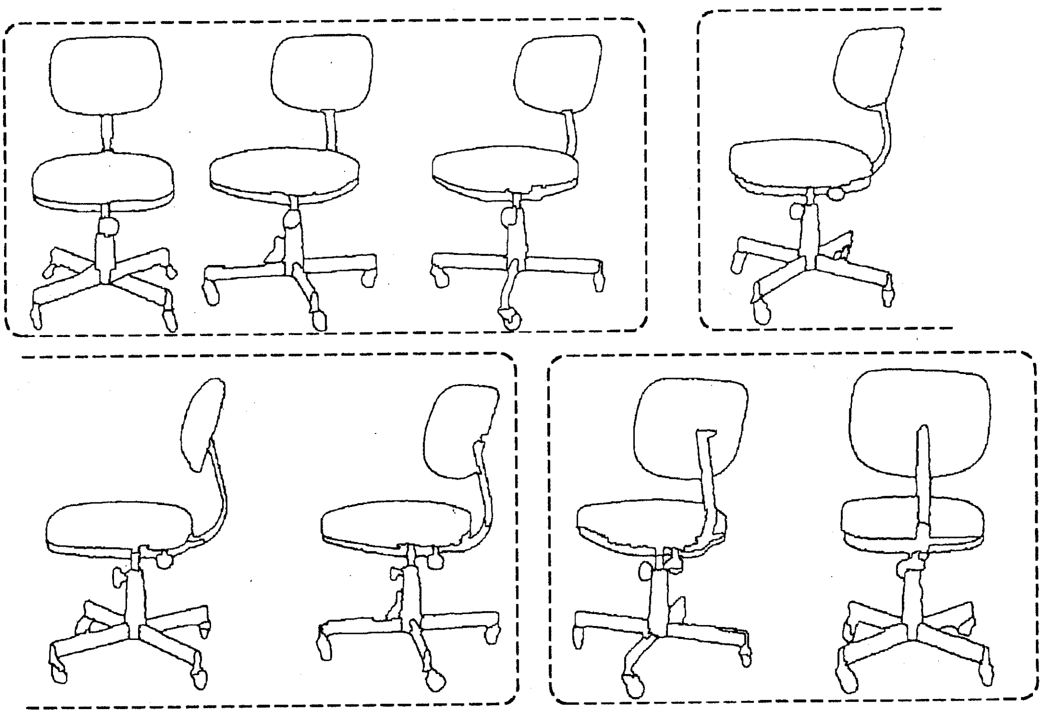

Figure 5: Result of clustering of several views of a chair.

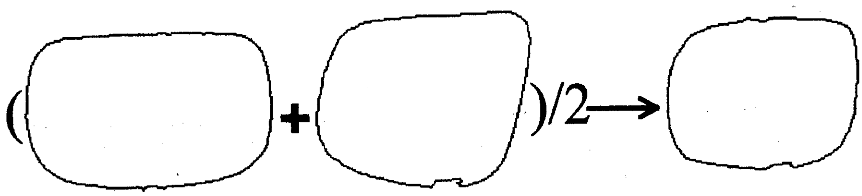

Figure 6: Generate a cluster component from two sample parts.

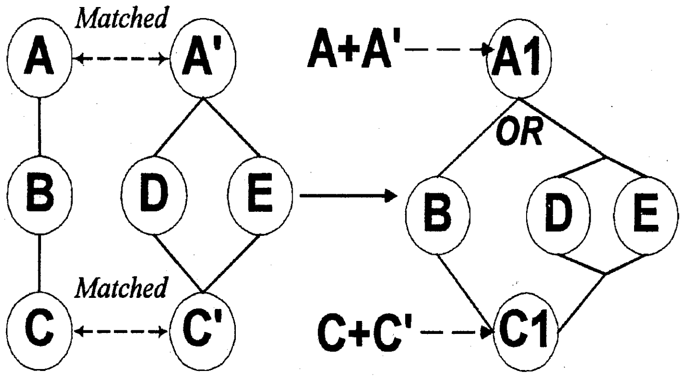

Figure 7: Generate a model based on matching two different kinds of chairs.

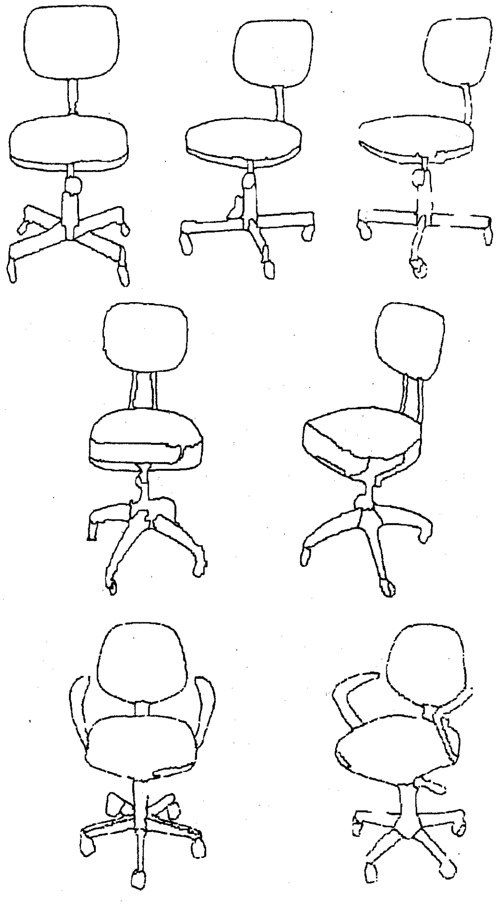

Figure 8: Objects used to generate the model shown in Figure 9.

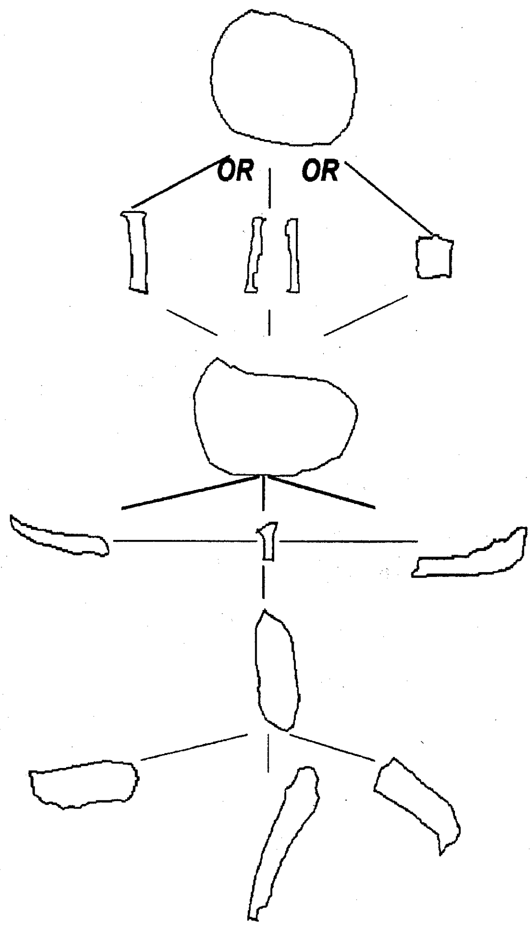

Figure 9: The model generated from the objects shown in Figure 8.

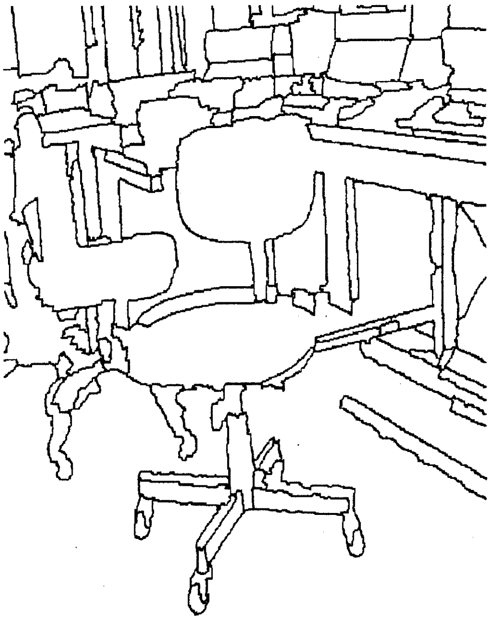

Figure 10: Segmentation result of an input image.

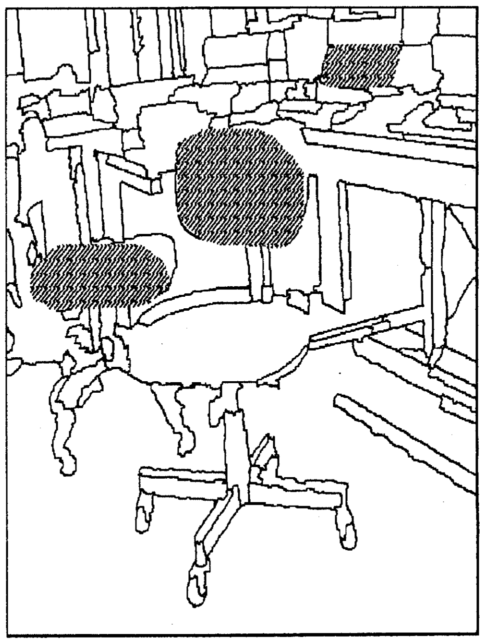

Figure 11: Result of finding one component in the model

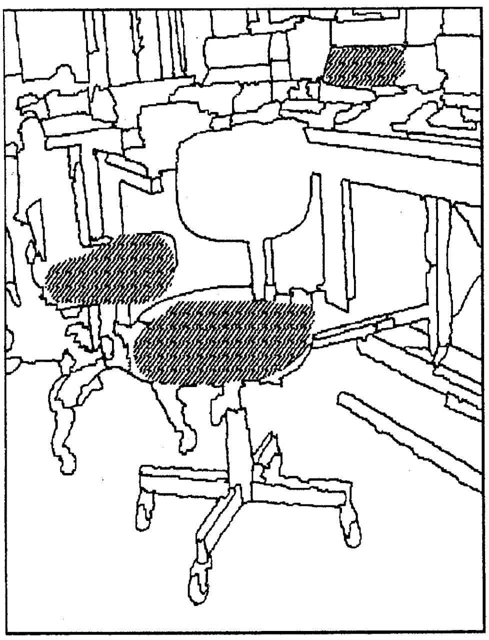

Figure 12: Result of finding another component in the model

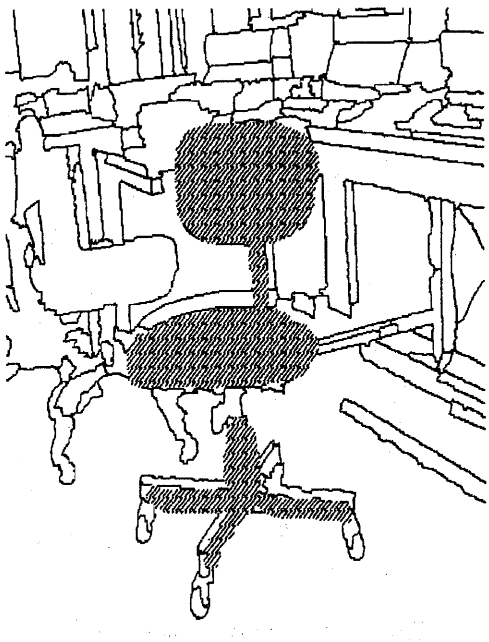

Figure 13: Final result of recognition: a chair is detected.