EDGE FINDING, SEGMENTATION OF EDGES AND RECOGNITION OF COMPLEX OBJECTS

Yoshiaki Shirai
Electrotcchnical Laboratory
Tokyo, Japan

ABSTRACT

This paper describes an approach to the recognition of real-world objects such as books or a telephone on a desk. The system consists of (l) edge finding process which extracts edges of curved objects from light intensity data, (2) segmentation of the edges into straight lines or elliptic curves, (3) recognition of objects by matching the liner, to the models, and (4) simple supervisor. The module (1), *(2)* and (3) interact with each other through a simple supervisor so that the system can locate given objects quickly- First, most reliable edges are found and segmented, and then recognition ir. attempted using segmented lines. If recognition is not successful, less reliable edges are searched for, arid recognition is retried. An example of locating a lamp, a book stand and a telephone is shown.

1. INTRODUCTION

This paper describes an approach to the recognition of real-world objects such as books or a telephone on a desk. Scene analysis could be divided into two modules: (1) extraction of features and (?) recognition of objects based on the features. It is often pointed out that the two modules should not be simply connected serially, but should interact with each other to achieve reliable recognition [1].

The author made an attempt to recognise polyhedra by those two modules i.e. a line finder and a line proposer [?]. There have been many works on model-driven recognition of polyhedra [3, 4.5]. However, it is not easy to make these two modules capable of recognizing curved objects. The difficult problem is to determine what features are useful for recognition and how to Interpret? the extracted features.

There have been some works on description of curved objects using range data [6,7,8]- Garvey and Tenenbaum [9] made an experiment for locating objects in office scenes using range and color data. The work presented here uses only light intensity data, because it can easily be obtained and stored, which makes it practical to approximate unknown scenes at first.

Features of objects to be extracted for recognition depend on the objects. For outdoor scenes, regions of uniform color might be good clues for recognition of the sky, fields, lakes and so on [10,11]. For example, a large blue region at an upper part of a scene might be the sky. For office scenes composed of known objects [9], color and range of the each point in the scene might be enough to locate the table top or the door candidates.

On the other hand, recognition of every day objects on the desk using only light intensity data requires more complex features. There is less constraint on the configuration of objects on the desk than those in the outdoor scenes. IAght intensity of a point in the unknown desk scene alone can not suggest any meaningful facts. It might be necessary to look at the change of light intensity and the relation with other parts of the scene.

There have been many works on feature extraction of complex scenes [12,13], but few works have recognized the scenes, using these features. While, many papers assuming picture primitives, have dealt with the symbolic primitives to recognize complex objects. There have been few works on recognition of complex objects using raw light intensity data.

The author has proposed local features and global features of curved objects for analysis of real-world scenes [l^]. The local features represent the types of 1ight intensity change and the global ones are the connected local features of the same type. If the curved object has uniform surfaces, the global features correspond to the edges of the plane *or* curved surfaces.

Tsuji [15] made a linguistic approach to segmentation of curved object scenes into regions using picture primitives similar to the local features described above. It consists of hierarchical modules, each of which compresses the input data to finally obtain the plane and curved regions. The problem with the method is that it must completely analyze the whole scene before it extract?', some useful facts about a scene.

This paper is a further extension *of [14]* and describes a total system. The system consists of (1) edge finding, (2) segmentation of the edges into straight lines or elliptic curves, *(3)* recognition of objects by matching the lines to the models, or by usirig the raw data if necessary, and (4) top level supervising. The module (1), (2) and <3) are controlled by a simple supervisor to interact with each other so that depending on the situation, the system can locate given oblerts quickly or analyze the whole scene. The module (]) and (?) are easily controlled by a set of global parameters such as reliability level, minimum curvature, spatial resolution, etc. They are applicable to various objects. The module (3) which has at present less generality is described briefly in this paper.

P. EDGE FINDING

?.l. Edge point detection

An edge of an object surface is defined here as a series of smoothly connected edge points. Many operators for edge point extraction have been proposed [12,13,16]. The two operators (two-dimensional and one-dimensional) described here are designed to detect edge points of planes and smoothly curved surfaces. The operators extract discontinuity of both the light intensity and its gradient. If the direction of the edge is unknown, two-dimensional operator is used to compute the direction of the gradient, otherwise a one-dimensional operator detects the edge point quickly.

The former operator makes use of the gradient computed by a conventional method [14 One of three regions shown in Fig. 1 in used to get the gradient at the center element. The gradient value D and the direction a for mode 1 are defined by light intensity of the neighboring points I(i) as

follows.

$$D_x = I(8) + I(1) + I(2) - I(4) - I(5) - I(6)$$
$$D_y = I(2) + I(3) + I(4) - I(6) - I(7) - I(8)$$
$$D = |D_x| + |D_y|$$
$$\tan \alpha = D_y/D_x$$

The gradient for mode 2 or mode 3 is similarly defined. The operator for edge finding at (x_0,y_0) is defined by its gradient $D(x_0,y_0)$, $\alpha(x_0,y_0)$ and its neighbors. Let (x_i,y_i) denote a point which is i points away from (x_0,y_0) in the direction $\alpha(x_0, y_0)$. Similarly (x_{-i},y_{-i}) represents a point in the opposite direction. An edge point is defined using the following conditions.

(C1) $D(x_0,y_0) > D_t$

(C2) $\{|\alpha(x_i,y_i)-\alpha(x_o,y_o)|>\alpha_t\}$
$\vee \{D(x_i,y_i)<f(D(x_o,y_o))\}$
for some i in the neighbors $(-i_w \le i \le i_w)$.
where $f(D(x_0,y_0)) < D(x_0,y_0)$ is a linear function of $D(x_0,y_0)$

(C1) meanr, that the light intensity at an edge must change. (C2) means that the neighbor of an edge in direction a must have a different light intensity change from that at the edge. Usually if (C?) holds for both $i_1>0$ and ip^O, then the edge corresponds to the boundary between two planes. If only one side of the edge (i<0 or i>0) satisfies (C?), it corresponds to an edge of a curved surface. Fdgr points extracted by (CI) and (C?) are sometimes broad along an edge of a surface. In checking (C?), even if some i ^ is found at one side of the edge, (xQ,yrj) is considered as a transient point if for some i? at the other side,

(C3) $D(x_0,y_0) - D(x_{i_1},y_{i_1})$
$< D(x_{i_2},y_{i_2}) - D(x_0,y_0)$

Thus edge points must satisfy (C1) and (C2), and must not satisfy (C3).

The edge point, is classified into three categories: (1) type B which satisfies *(C2)* for both i >0 and *io*<0> (?) type L which satisfies (C?) for only i >0 , and (3) type R which satisfies (C?) for only *i*<0. Fig. ? shows an example of the edge point extracted by this operator.

The one-dimensional operator is used if the direction of the edge is predicted. It assumes that the direction of the gradient *a* at the neighboring points is perpendicular to the predicted edge direction. This operator detects an edge point using an intensity profile of the picture elements on the line centered at the point in the direction a, and classifies it into the three categories in the similar way described above.

2.?. Edge finding program

Fig. 3 shows a simplified flow of the edge finding program. The program has two main tasks: l) find an edge kernel above a given reliability level, and P) track along the edge starting from the kernel.

The edge kernel is a set of edge points of the same cateogry which have similar gradient directions. A kernel finder first detects an edge point using the two-dimensional oeprator described above, and detects other edge points along the estimated direction of the edge which is perpendicular to the gradient direction of the first point. If enough edge points are found, the edge kernel is classified into three categories (B, L and R) based on the types of the edge points. The reliability level corresponds to the threshold of the gradient (Dt) in (C1) for the edge point detector.

A reference map is used to speed up the kernel finding. The original picture is divided into rectangular regions (e.g. 6x6 picture elements) and the map contains an approximate gradient value of the each region. The kernel finder searches for an edge point in the regions corresponding to the points in the map with larger values than a given reliability value. After an edge kernel is searched for in the candidate region, the corresponding point in the reference map is updated in order to give an accurate value or to avoid the repetition in the later search.

The kernel thus found is extended in both directions by tracking the edge. The purpose of the tracking is to obtain a smoothly curved edge which consists of the edge points of the same type. Intuitively the smooth curve means that it is continuous and the tangent direction does not change abruptly. The type and the current parameters of the edge such as the position X_c. , direction 6_c and gradient value D_c are initially given by the kernel finder. The current parameters are updated as the tracking goes on. The main steps of the tracking at one cycle are

1. Predict the position of the edge point using X_c and β_c.
2. Find a proper edge point.
3. Update X_c, β_c and D_c, and go to 1.

The prediction of the edge point P at n-th cycle is based on the current edge position X and the direction B^n as shown in Fig. U. The distance £ between XJ? and P^n is one of the parameters of the edge finder which determines the minimum curvature of the edge tracker. When a proper edge point E^n is found, Xp+1 is advanced on the line passing through X£ and F^n, and B£+^ in updated as shown in Fig. *h*. If the predicted position of the edge is regarded as being on the other edge already found, or on the other end of the same edge, the tracking terminates. For this check, a binary map of the edges is used. When the edge point E^n is found, the map is updated at the point corresponding to E^{n-m}(ra depends on £ described above) and its two neighbors in the direction perpendicular to $8^{n"m}$.

In step ?, the one-dimensional operator is used to find an edge point of the same type near the predicted position. The direction of the gradient of the edge point is assumed here to be perpendicular to the current direction of the edge. If the gradient value of the detected edge point is close to D_c, it is determined as an extension of the edge. If no such points are found, a back up routine determines whether or not to stop tracking as follows.

a) If step 2 is the first attempt of the edge point finding, a different gradient direction is proposed to find a proper edge point again, and go to step *2*.
b) If an edge point is found in the previous cycle, then insert a siinulated edge point at the predicted position and go to step 3-
c) Othervise, stop tracking.

The purpose of the back up is to avoid the effects of a small texture of an object, various kinds of noise and digitization of directions.

When the tracking in both directions terminates, each end of the edge is checked to connect to the other edges already found. If the tracking stopped at the other edge of the same type and the both edge can be connected smoothly, they are merged.

Fig. 5 shows an example of the edge finding. Usually, edge finding is repeated over the picture, each time changing the reliability to a lower level The tracking of a less reliable edge can be blocked by more reliable edges which cross to it.

3- SEGMENTATION OF EDGES

Each edge obtained by the edge finder is represented by a set of smoothly connected points. Tfce segmentation of edges is to describe them by several parameters useful for recognition.

Tfcere have been many works on the mathematical approximation of curves by polygonal lines [17]. piecewise polynomial functions or spline functions [18]. They defined an error norm and attempted to find an optimal solution. However, these are not easily applied to the description of the complex scenes.

In this paper, straight lines and elliptic curves are used to describe the edge curves. The segmentation of the each edge is divided into two parts: (l) segment the edge into lines and curves, and (?) approximate each segment by a straight line or an elliptic curve.

Fig. 6(a) illustrates the purpose of the first part, i.e. to divide the edge AF into five segments (AB, BC,...,EF). The method is a heuristic one which maker, use of the curvature of the edge as described in [2] (i.e. the curvature of an edge point F is defined as the angle 6 between PR and PQ, where Q and R are On the edge and constant number (m^) of points away from P). Thus, the curvature of AF is defined on the interval between A* and F' as in Fig. 6(b) which is ir^ points inside of AF. The main steps of the method are

1. Find the candidates of the knots of the edge.
?. Classify temporary segments into straight lines or curves.
3. Merge segments if possible, and determine the positions of the knots.

First, to locate the knots where the direction of the edge changes abruptly, the regions are found which have large curvatures. The candidate regions of the knots are illustrated in Fig. 6(b) as intervals [Li,Rj]. The interval is defined with the curvature 6(P) as follows. Let fij^x denote the maximum of |6(P)| in [Li,Ri], then the interval satisfies the following conditions.

$$\delta_{max} \geq \delta_t$$

$$|\delta(P)| \begin{cases} \geq h(\delta_{max}) & \text{for } P \in [L_i, R_i] \\ < h(\delta_{max}) & \text{for } P = L_i - 1 \text{ and } P = R_i + 1 \end{cases}$$

where h(x) is a linear function of x(h(x)<x).

The intervals may contain either real knots or curved segments. If the interval is short, it is regarded as a real knot, region, otherwise a curved segment. New knots are inserted at the ends of curved segments. Now the temporary segments are defined an the intervals between the knot candidate regions (i.e. [A',L_1], [R_1,L_2], [R_2,L_3], [L_3,R_3] and [R_3,F'] in Fig. 6(b)).

The next step classifies the temporary segments (excluding the curved intervals found in step 1) into straight lines or curves using the constant (m_d) of the curvature, sum of δ's (S_δ) and the total number (N) of points in the segment. If the segment PQ is assumed to be an arc as in Fig.7. the maximum distance d between the line PQ and the arc, and the fan angle θ are approximated as follows.

$$d = (N/\theta)\,(1 - \cos\theta/2)$$

$$\theta = S_\delta(N - 1)/m_d(N - 2m_d)$$

Based on this arc assumption, the classification algorithm is: if N≤?m; then the segment is undefined; if $d>d_u$, then curve; if $\theta<\theta_t$, then line; if N<m/2 and $d\leq d_\ell$ ($d_\ell<d_u$), then line; otherwise curve.

The third step tries to merge undefined and curved segments. If sums of the curvatures S51 and S,52 of the adjacent segments and that of the knots are all similar, then the segments are merged into one segment. This procedure is repeated until no more merge is applied. Small segments of a smooth curve created in the step 1 because of the effect of noise or digitization, might be merged here. The position of each knot is determined now accurately as the border of the adjacent segments.

The second part of the segmentation fits a straight line or curve to a set of edge points (Xi, Yi) (i=l,2,...,n) in each segment. Conventional methods of least squares assume that only Y^ is accompanied with error. But the position of each edge point has error of both Xj and Yj. The Deming's method [l°] is employed to find the parameters of the curve which minimize the sum of square of the distance from the curve to earn point.

Let the equation of the curve vith m parameters be denoted by f(x,y; a],a£, . . .,$a_{T!1}$) = 0, then the method, starting from the approximated parameters, estimates better parameters iteratively until they converges within a given error ratio. The initial parameters are given by an appropriate number of edge points which are sampled from the segment. If the segment is classified into a curved line, an elliptic curve is at first tried to be fitted to it. The parameters are the center position, major and minor axes and rotation. If the parameters do not converge in the iteration, the number of the parameters is decreased. Fig. 8 shows an example of the results of the segmentation and curve fitting.

k. RECONGNITION

A strategy of recognizing an object in a scene depends on how much information is already known. If a desk area is known, the fastest way to locate a telephone is to search in the left front part of the area, and if a dial of the telephone in found, recognition of the receiver is easy. The best way to analyze the whole scene is to start from the most obvious object first, and proceed to the next obvious object making use of the previous facts. To locate a given object, however, the best way might be to try to find it first. If it fails, the other object might be tried which could help to locate the given object.

We have been taking multiple approaches to recognition of objects. One is matching models of

objects described by the graphs to a scene []**]. Now a scene is described by the equations and types of the edges. Models of an object are two-dimensional and represent its views from typical directions. Because a scene is represented by straight lines and elliptic curves, usually a few models are enough for an object. However, this method can not discriminate shapes which are sensitive to rotation, such as a circle from an ellipse. Each model is described by the edges and their relations. For analysis of a scene of multiple objects occluding one another, matching of subgraphs is required. An algorithm which extracts maximal common subgraphs between two colored graphs has been developed [?0l. Scenes of simple objects such as blocks, cylinders and spheres are recognized by the graph matching technique. The method can easily add new models of objects by declaring their descriptions. There are, however, many difficulties in recognition of complex objects, e.g. description of models and efficient matching procedure. The graph matching could be used as a part of *recognition* procedure.

Another approach is a procedural one, i.e. to provide with a procedure for recognition of each object. For recognition of a telephone, for example, the procedure is first to search for the dial, and then to find the contour of the telephone around the dial. The candidates of the dial is easily found by examining the equations of the curved edges. The telephone contour is searched for in a certain region around each dial candidates using a map of the edges. Serially connected lines or curves which surround the dial are identified as the contour. If enough proof is obtained, the recognition is successful. If proof is not sufficient or many candidates of a telephone are found, further i river, tigation is required which might ask more edges or check the inside of the dial. Sometimes the investigation requires a facility for v.ooming-up a scene although this feedback loop has not at present been implemented.

A simplified organization of the present syste.ti Li-, uhown in Fig. 9. Given a set of object names, the supervisor asks the edge finder to get edges. If enough number of edge points or edges are found, then the edges are segmented. Otherwise, the supervisor changes parameters of edge finding and more edges are searched for. The rerognition consists of subroutines for the given objects. Tf not al1 the objects are recognized successfully, the supervisor asks the edge finder again to collect more edges.

Fig. 10 shows an example of locating a lamp, a book stand and a telephone. Recognition of a telephone is as described above. To locate a lamp, a lamp shade is searched for first as edges of a bright strip region of an allowable dimension, and then the trunk is searched for by finding a pair of parallei vertical edges whose direetions are opposite. A book stand is located by finding a set of vertical lines clustered in a certain region. The computing time for locating the three objects using a picture (6 bit level, 256x256 points) stored in the disk is about *h* minutes. More than half of the time is spent for finding edges.

On the other hand, in order to recognize as many objects in a scene as possible, the edge finder is given parameters of low reliability level to get many edges. Even if recognition of some objects is not successful, edge finding can not be retried in this case. An experiment is mode Tor the same scene as in Fig. 10 in this manner, Thf; upper bound of edge number (TO) is determined by the storage limitation of the computer. Although the same objects are successfully located, the computing time for finding edges is more than twice of the above example (Fig. 10). This shows that in order to locate given objects, the system organization in Fig. 9 is efficient.

CONCLUSIONS

An approach to the recognition of real-world objects using light intensity data has been described. The recognition is mainly based on the edges of curved objects. The edge is a set of smoothly connected edge points where light intensity *or* its gradient changes abruptly. The edges are segmented into straight linen or elliptic curves so that they are easily matched to the models of objects. Recognition of complex objects with a simple matching procedure has been successful.

The method proposed in this paper would be applicable to recognition of every day objects which are identified by the edges of straight linen or ellptie curves. Further research is required to check on doubtful objects by zooming up the regions of interest in the scene. The other objective to be studied is an efficient way of making many models of complex objects.

REFERENCES

1. Winston, P. The MIT robot. Machine Intelligence 1 (1972).
2. Shirai, Y. A Context Sensitive Line Finder for Recognition *of* Polyhedra. Artificial Intelligence *h (2)* (1973).
3. Falk, 0. Tpterpretation of Imperfect Line Data as a Three-Pimensional Scene. Artificial Intelligence 3 (?) (1972).
4. Perkins, W. and Binforri, T. A Corner Finder for Visual Feedback. Computer Graphics mnd Image Processing *?* (*1/k)* (1073).
5. Grape, G. Model Rased (Intermediate-Level) Computer Vision. Artificial Intelligence Project Memo No. POT, Stanford University, Stanford, Calif. (1970).
6. Agin, G. and Binford, T» Computer Description of Curved Objects, Proc. 3rd Intern. Joint Conf. on Artificial Intelligence (August, 1973).
7. Nevatia, R. and Binford, T. Structured Description of Complex Objects, ibid.
8. Oshima, M. and Shirai, Y. Representation of Curved Objects Using Three-Dimensional Information. Second USA-Japan Computer Conf. Proc. (August, 1975).
9. Garvey, T. and Tenenbaum, J. On the Automatic Generation of Programs for Locating Office Scenes. Proc. 2nd Intern. Joint Conf. on Pattern Recognition. (August, 197**).
10. Preparata, F. and Kay, S. An Approach to Artificial Nonsymbolic Cognition. Information Science *h* (3) (1972).
11. Yakimovsky, Y. and Feldman, J. A Semantic Based Decision Theory Region Analyzer. Proc. 3rd Intern. Joint Conf. on Artificial Intelligence. (August, 1973).
12. Prewitt, J. Object Enhancement and Extraction. Picture Processing and Psychopictorics, Academic Press, New York (1970).
13. Rosenfeld, A., Thurston, M. and Lee, Y. Edge and Curve Detection: Futher Experiments.

IEEE Trans. on Computers, C-20 (5) (1971).
14. Shirai, Y. A Step toward Context Sensitive Recognition of Irregular Objects. Computer Graphics and Image Processing 2 (3/4) (1973).
15. Tsuji, S. and Fujiwara, R. Linguistic Segmentation of Scenes into Regions. Proc. 2nd Joint Conf. on Pattern Recognition. (August, 1974).
16. Hueckel, M. An Operator Which Locates Edges in Digitized Pictures. J. ACM 18 (1) (1971).
17. Pavlidis, T. and Horowitz, S. L. Segmentation of Plane Curves. IEEE Trans. on Computers, C-23 (8) (1974).
18. Braess, D. D. Chebyshev Approximation by Spline Functions with Free Knots. Numer. Math. 17, (1971), 357-366.
19. Honma, H. and Kasuya, N. Dimensional Analysis Method of Least Squares and Experimental Formula. Lectures on Applied Mathematics 5, Corona Publishing Co., Tokyo (1969) (in Japanese).
20. Tamura, K. Maximal Isomorphic Subgraphs between Two Graphs. Bul. Electrotech. Lab., 37 (9) (1973) (in Japanese).

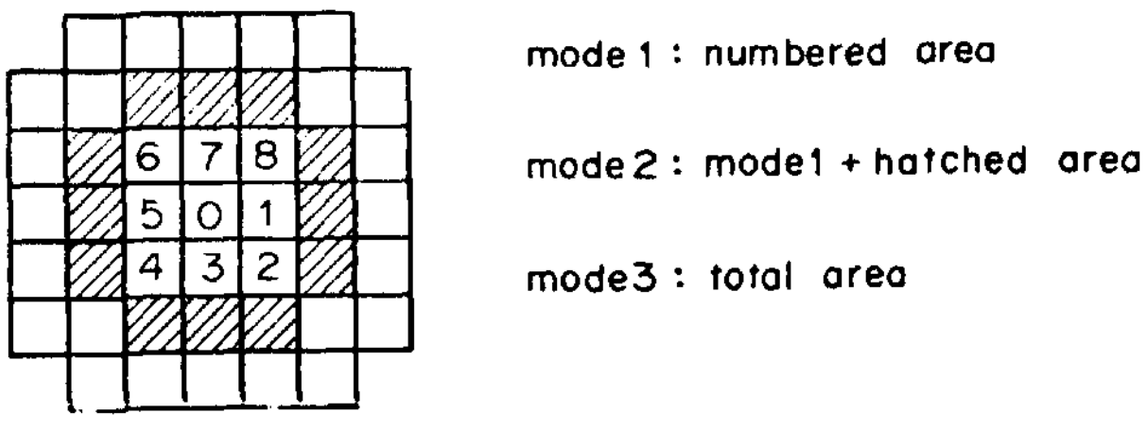
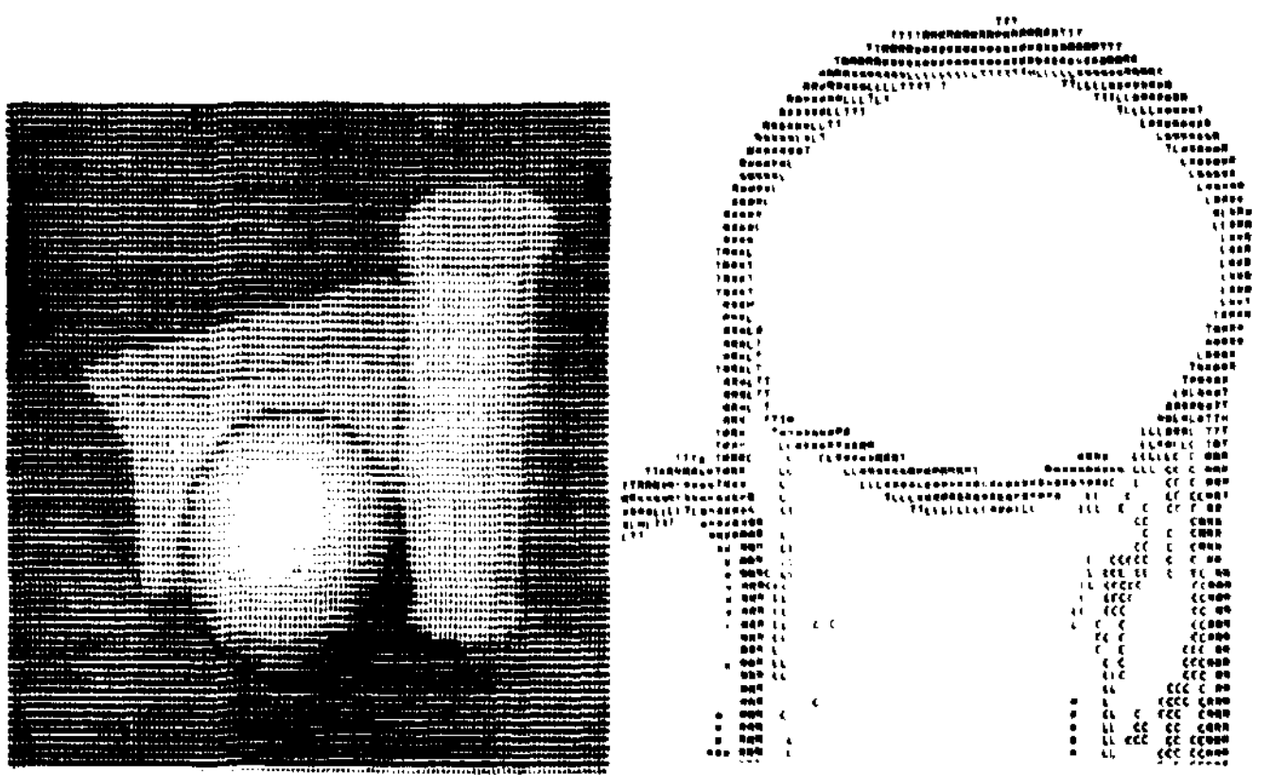


Fig. 1. Area used to compute gradient



L: type L, R: type R, *: type B, T: transient, C: curve

Fig. 2. Edge point (right) of upper right of scene (left)

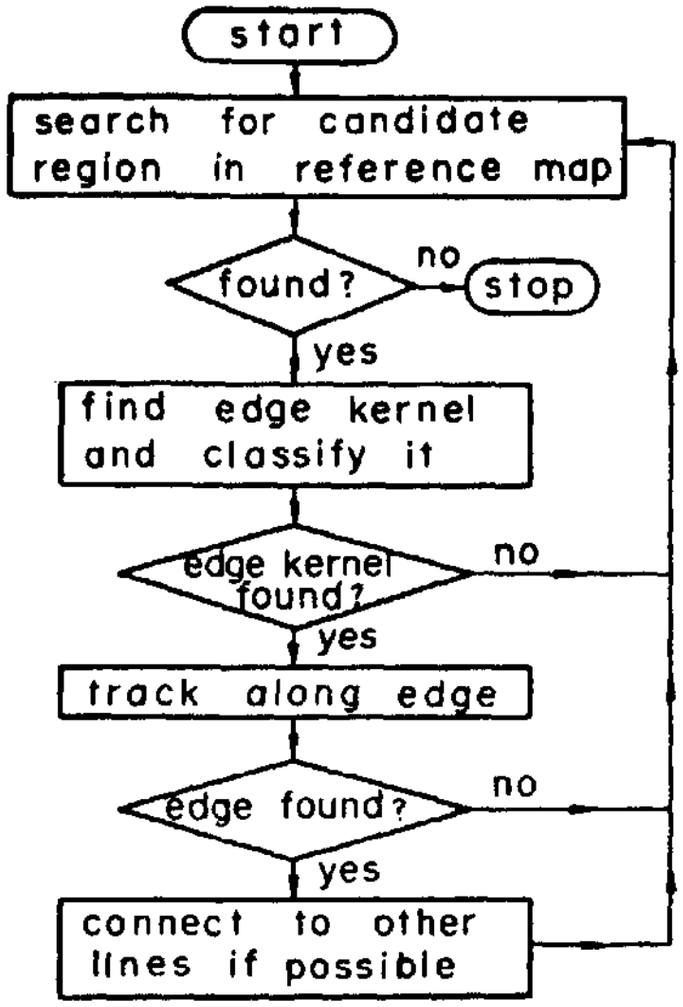


Fig. 3. Flow chart of edge finding

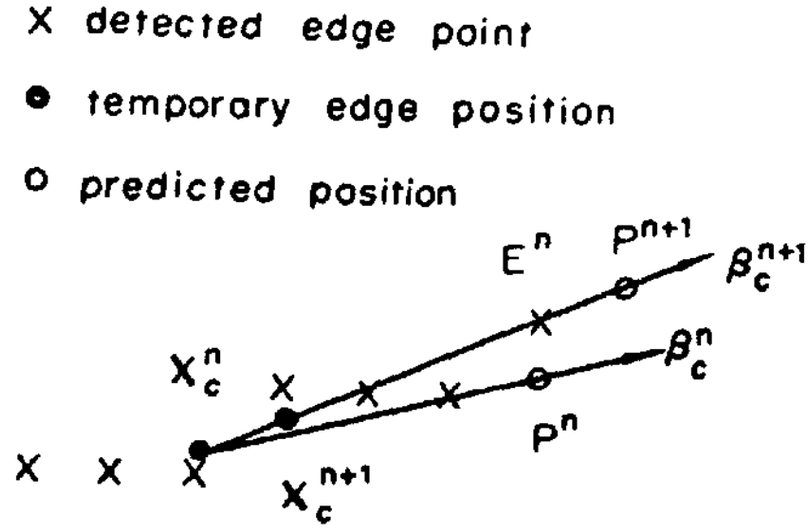
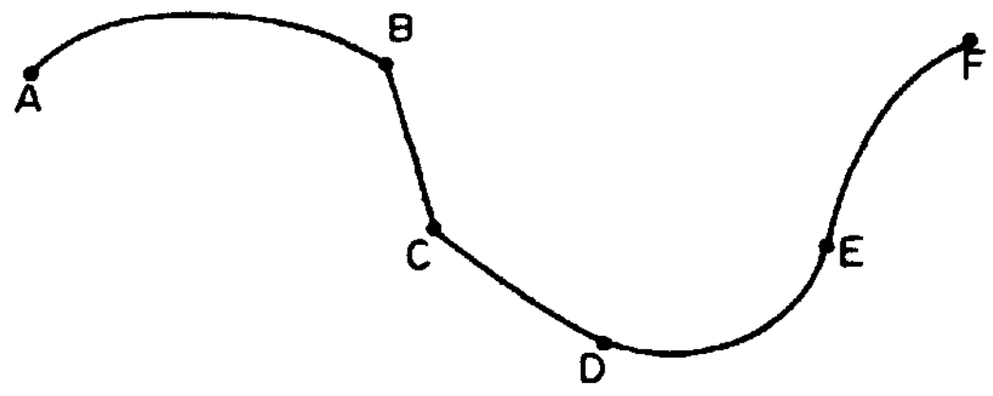
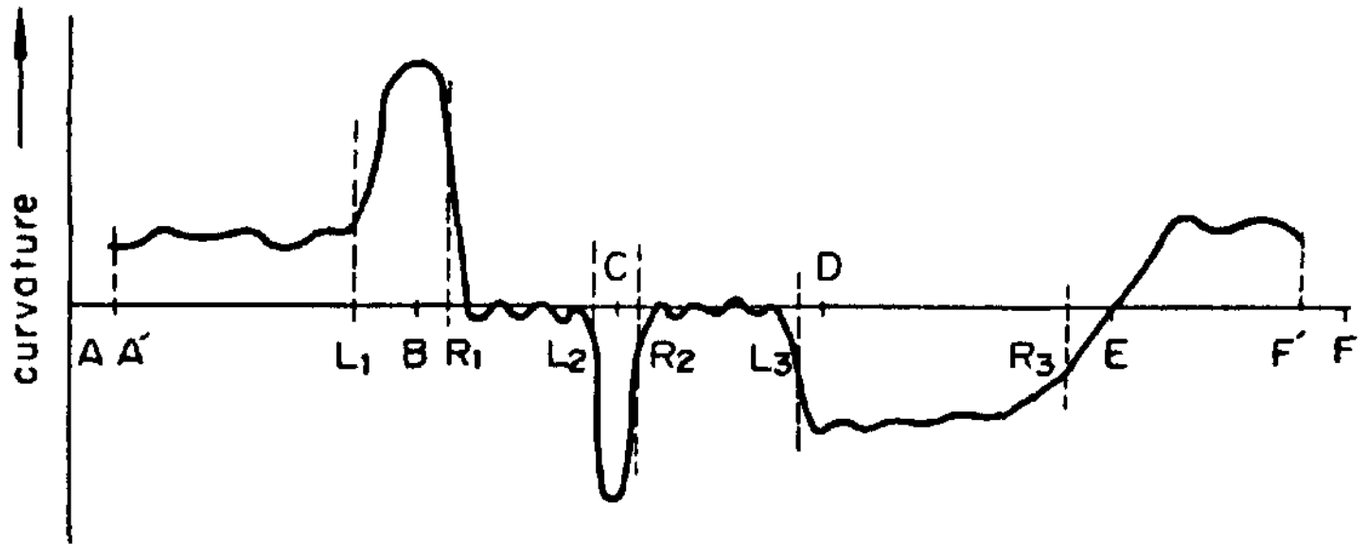


Fig. 4. Prediction of edge point

Fig. 5. Example of edges (left reliable, right less reliable).



(a) edge to be segmented



(b) curvature along the edge

Fig. 6. Initial segmentation of edge

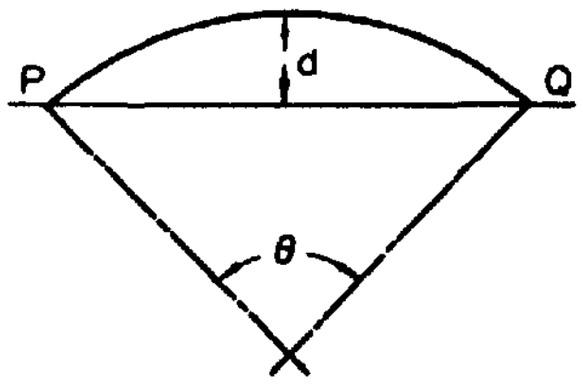


Fig. 7. Arc approximation

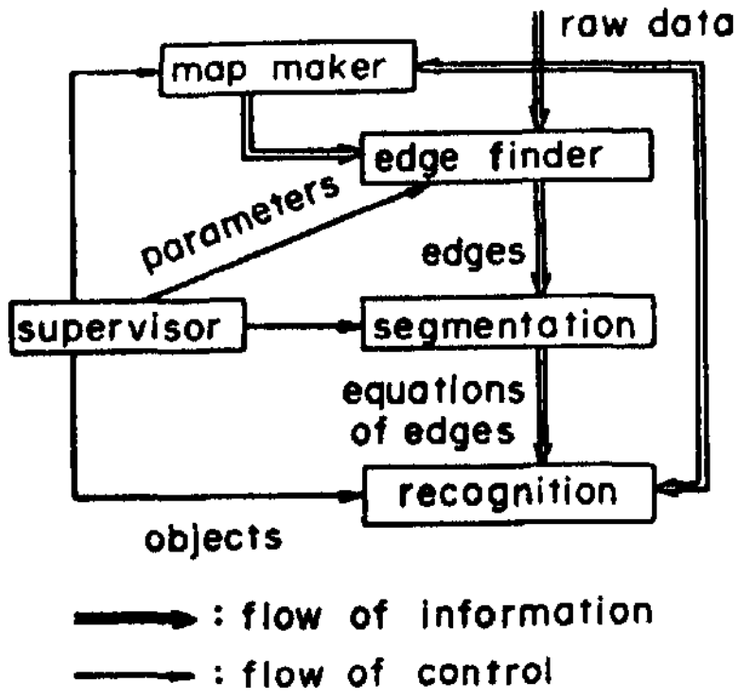


Fig. 9. Total system

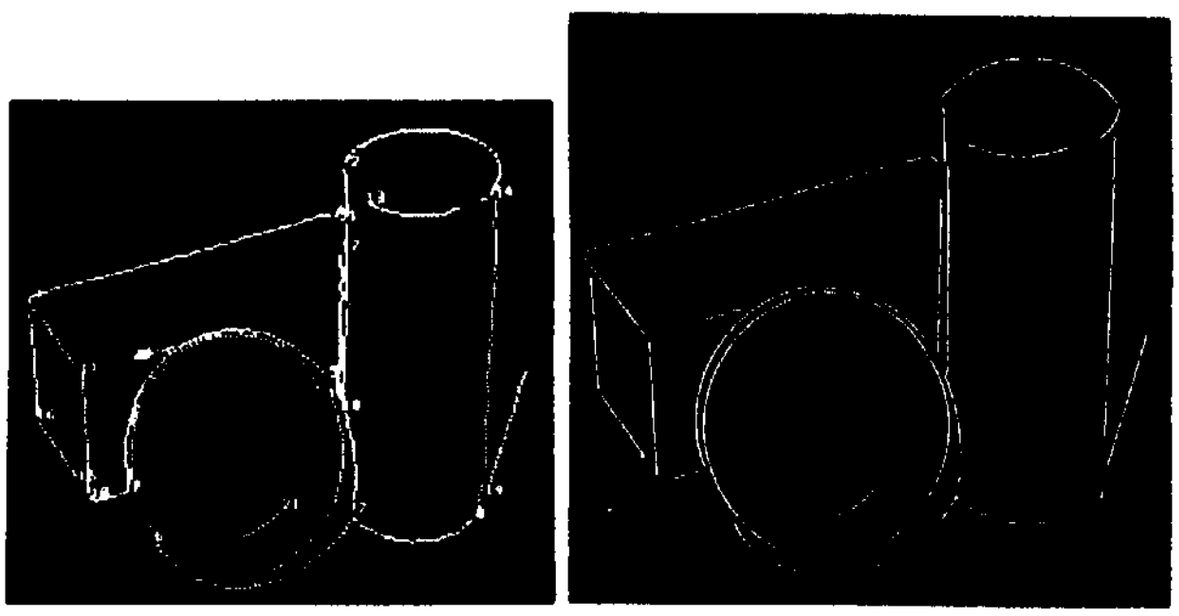
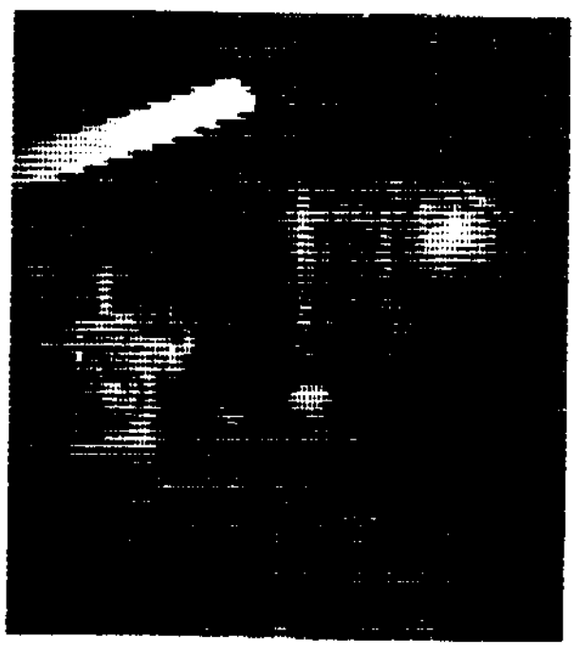
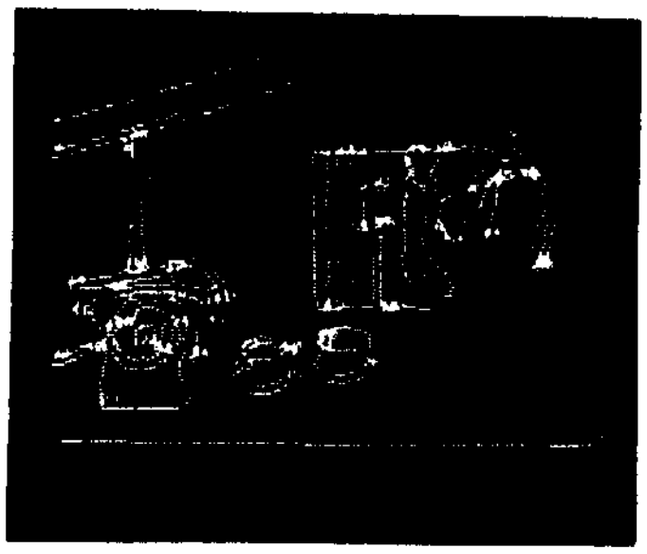


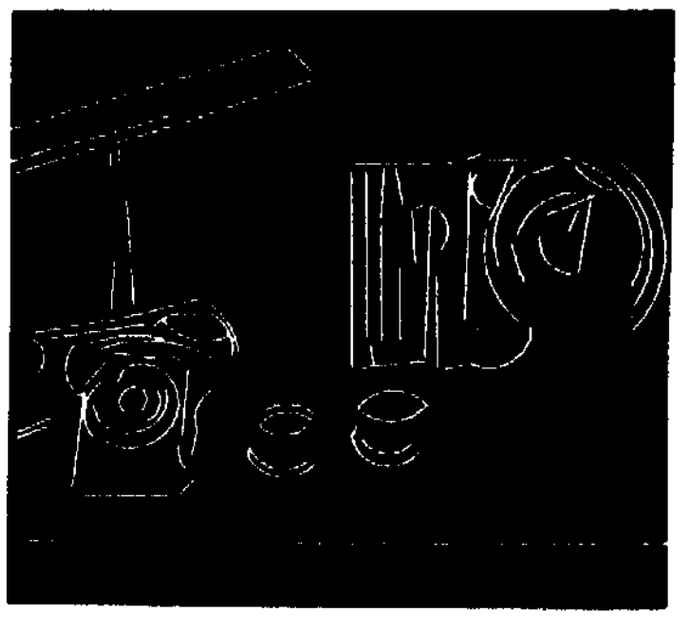
Fig. 8. Segmented edges (right) and fitted lines (left)



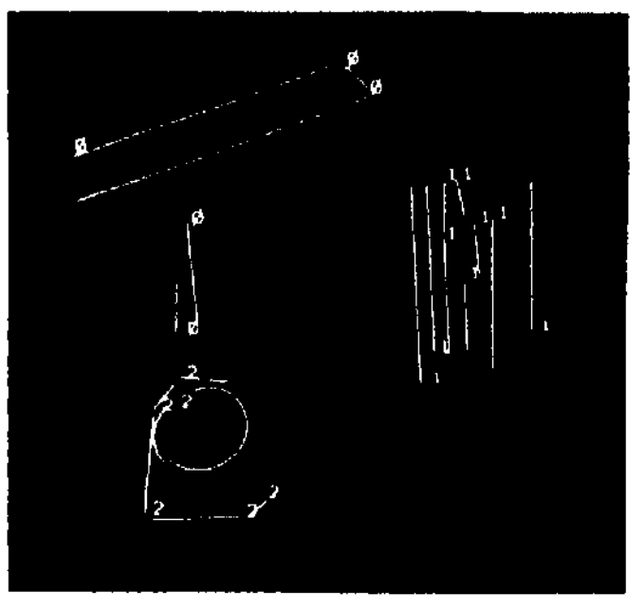
(a) original scene



(b) found edges



(c) fitted lines



(d) lines used for recognition
0; lamp, 1; book stand
2; telephone

Fig. 10. Example of locating objects