SCENE 지식을 사용한 휘도 배열분석

이 한 식
충 북 대 학 교

ANALYZING INTENSITY ARRAYS USING KNOWLEDGE ABOUT SCENES

HAN SIK LEE
CHUNGBUK NATIONAL UNIVERSITY

Abstract

Our work is an attempt to recognize objects without a rigid ordering of steps and with fuller use of previous results as analysis proceeds. We assume that the difference in brightness between the objects and the background is large enough to detect the background boundaries easily.

Lines are mostly proposed insted of found by exhaustive search in the scne,the program is relatively efficient.

1. Introduction

This paper describes what Minsky and Papert[1)] call a heterarchical program,one organized like a community of expert. The purpose of the program is to transform information from an image dissector camera into line drawings of polyhedra.

Most previous programs first find feature points in an entire scene and then make a complete line drawing using those feature points. But it has proved very difficult to work out a complete line drawing this way without using any knowledge about constraints that limit what can possibly be in the scene. If the line drawing has some errors,further analysis based on it's likely to lead to serious mistakes.

The program is based upon the strategy of recognizing objects step by step, each time making use of previous results. The order of the lines to be detected is countour lines which separate the bodies from the background, other boundary lineswhich separate two bodies, and internal lines which lie at the intersection of two faces of the same body.

Among other boundary lines and among internal lines, the most plausible lines are proposed at each stage and an attempt is made to find them. To find a line, the range where a line segment may exist is proposed, and it's detected in a way suitable for the proposed range. If a proper line segment is found, the end of the line is determined by tracking along the line. When the line is determined, the program tries to undestand the scene taking this line into consideration. At present, this program works well on moderately complicated configurations of blocks and wedges.

2. Hypothesizing Lines

Extracting the most obvious information first dictates the following order for for the ten heuristic line proposing steps:

1, If two boundary lines make a concave point, try to find collinear extensions of them. If only one extension is found, trac along this line. Most of such cases are one body hides the other. It is easy to see to which body this line belongs.

2. If no extensions of two concave - lines are found,try to find another line which starts from the concave point. if only one lineis found, track along this line. Most of these case are not clear locally to which body this line belongs.

3. If both extensions of the two lines are found at a concave point, try to find a third one. If only one new line is found,track along this line.

Whenever tracking terminates, an attempt, is always made to connect the new line to the other lines already found. If more than one line segment is found in (1) .(2)or (3), the tracking of all those lines is put off, hopefully to be clarified by the results obtained in simpler case. Figure 1 illustrates two extensions found at concave point p. The interpretation of thetwo lines is put off to treat simpler cases first. That is,one would continue examing the contour and lines AB and CD might be found next; then, by a circular search at point B(which is explained later), line BP would be found. At this stage it is easier to interpret lines AB and BP as boundary lines which separate two bodies.Then line DP wwould be found similarly and interpreted correctly. If no line is found in case, extend the line by a certain length as in case and test if it is connected to other lines. If not connected, try circular search again as in case. This process can also be repeated until successful. figure 2 illustrates this process. In Fig.1(a) line MN' is not connected to others at N', thus step case is tried at N' and fails. The line is extended to P1 and step case is again applied. This process is repeated until the line is connected to line KL at N.Figure 2(b) shows that line HI is extended by this process to P2 where a new line is found by circular search. Similarly line CG is extended to p4. This helps to find bodies sitting on obscured edges.

Whenever one of the above steps is finished, considerable information is stored.It is then available to help guide further search. For instance, if tracking along a line terminates, a test is made to see if the line is an extension of other known lines or if the line is connected to a known vertex.If a new boundary line connects two known boundary lines, the body is split into two. In fig3, line N'P' is obtained by tracking from point N. This line is interpreted as an extension of HN, and HN and N'P'are merged into one straight line using the equations of these two lines. It is then connected to CO and Fig 3(b) is obtained. Before the line was connected to CO, there were two bodies B1 and B2 as in Fig.3(a). Now body B2 is split into two bodies, B2 and B3. We can interpret line NO as the boundary of B3 which hides a part of B2. Other properties of lines and vertexes are obtained similarly at this stage.

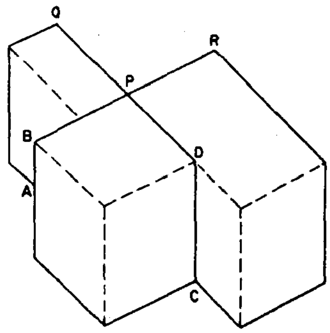


Fig.1 Two extension lines are proposed at concave point p.

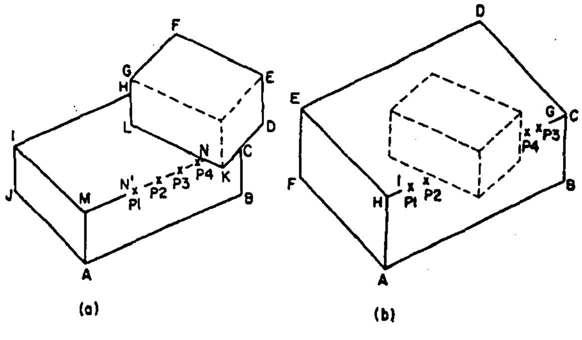


Fig.2 Continuation by circuit search.

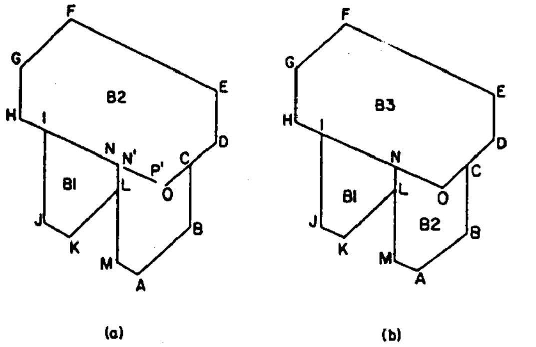


Fig.3 Splititing a body into two new bodies.

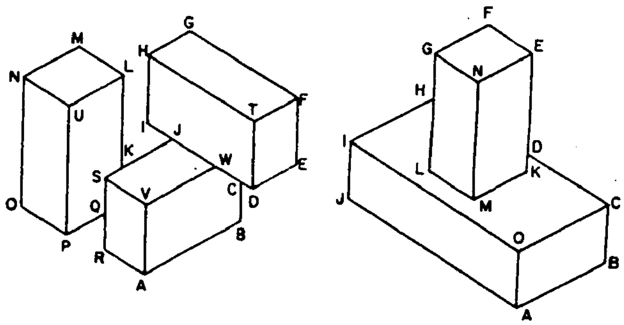
3. Experimental results and comments

To test the program, experiments were made with cubes and wedges having relatively uniform white surfaces placed on a black background. The image dissector camera,use as the input device, returns intensity information from points on a 1024 x 1024 grid. Objects occupy only a part of the scene. In a typical experimental scene, the rectangular area which includes the objects of interest may consist of only about 400X400 points.

Sound experimental practice requires that the pictures be stored to ease debugging and facilitate method comparisons. Consequently pictures are stord in mass memory in blocks which contain intensities from square patches each of which is made of64X64 points giving fast access for these programs which know a lot about where to look. When a light intensity at some point is required, a block containing the point and adjacent points are brought into core memory.

In these experiments the light is intensity is represented by a little less than 100 levels, spanning a range in intensity of about three to one. The input data for a clear bright edge in the dark background is blurred due to some optical and electo-optical limitations. If the real intensity change change is a step function , there is a transient area in the input data about 10 points wide. Thus the resolution of the picture can be regarded as 10 points. The parameters used in line segment detection and tracking are based upon this resolution.

Some results are shown in Fig.4 The difficulty or processing time of the recognition depends not only on the complexity of the object but also on the information known at each stage. In Fig.4(a), for example, boundary lines KS and QS are easily proposed as the extension of countour lines.On the other hand, it is not easy to find boundary lines KM or LM in Fig.4(b).



(a) (b)

Fig. 4 Experimental results

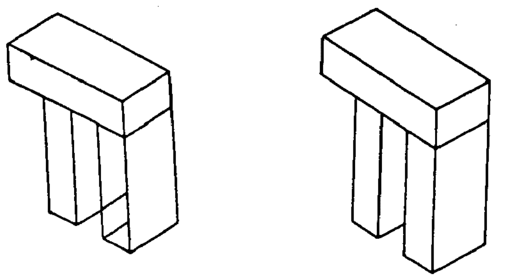


Fig.5 Comparison between hierarchical and heterarchical program.

That is,after DK and HL are found, circula search is necessary at K and L respectively. Circular search is less reliable in finding a line segment, and more time consuming. Once all the boundary lines of an object are determined, all the internal lines are proposed in both cases. But trcking along VW in Fig.4(a) and EN in Fig. 4(b) terminates in the middle.

An example of the result of an earlier pass-oriented program is shown in Fig .5 That program looks at the whole scene homogeneously and picks up feature points . Lines are found using those feature points. But it's very difficult to determine a prioei the various thresholds appropriate for detection of feature points, line fitting, and connection of lines. In the heterarchical program described here, the various thresholds are adjusted with the context of all the information obtained previously. Additionally, the particular tracking algorithm itself is changed from case to case depending on whether the line is a boundary or internal type.

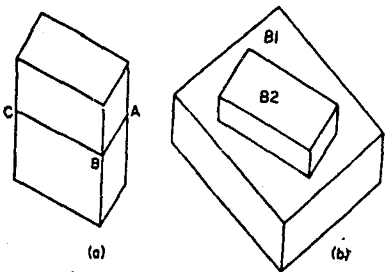


Fig.6 Comparison between hierarchical.

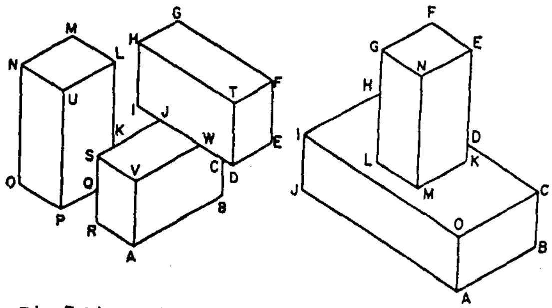


Fig.7 Situations with a lack of case.

The results of experiments with moderately complex scenes are mostly satisfactory. However, there are some limitations of this program at present. One of them is that bodies may be missed in some cases. A simple example is shown in Fig. 6 The boundary lines AB and BC in Fig.6 (a) are not proposed, though the other contour lines and internal lines are found, because the resulting-regions are such that no concave vertexes activate step 1. 1. In such a case when bodies are neatly stacked, it's necessary to seach for boundary lines which start from some points on the contour line.In Fig.6(b) body B2 is not found. To find a body that is included in a face of another body, it's necessary to search for line segments inside the region. Though these two kinds of search(search along the boundary line and search in the region) are required to find all the bodies in the scenes as shown in Fig.6, they are still more effective then the exhaustive search in the entire scene.

A more serious limitation of the present program is that it's not always applicable to concave objects. Figure 7(a) shows a simple example. Line BD is found as an extension of line CB.If all the bodies are convex, line BD IS INTERPRETED LIKE A boundary line as shown in Fig.7(b). This does not hold for concave bodies. In this program, line BD is regarded as a boundary line, and then line DE is found by circular search at D. At this stage, however,DE should be interpreted as an internal line of the same body insted of as a boundary line which separates the body into two. If DE were interpreted correctly, them line BD could be determined to be an internal line.

References

1. Minsky, Marvin, and Seymour Papert: Progress Report on Artificial Intelligence , M.I.T. Artificial Intelligence Laboratory Memo 252,1972.
2. Abelson,R.P.: The Structure of Belief Systems, in R.C.Schank and K.M.Colby (eds.), "Computer Models of Thought and Language,"W.H.Freemanb, San Frencisco," 1973.