1-1-1983

Recognition of three-dimensional objects from two-dimension line drawings.

Linda Kay Ludwigs

RECOGNITION OF THREE-DIMENSIONAL OBJECTS FROM TWO-DIMENSION LINE DRAWINGS

by

Linda Kay Ludwigs

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Computing Science

Lehigh University

1983

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

May 12, 1983
____
Date

____
Professor in Charge

____
Chairman of the Department

Table of Contents

List of Figures

List of Tables

Abstract

The development of scene analysis with respect to interpretation of two-dimensional projections of three-dimensional scenes is reviewed. Advantages and inadequacies of line labelling techniques are illustrated through examples. The use of gradient space to test the validity of labellings is discussed. Linear algebra and sidedness reasoning are introduced as accurate and understandable approaches to partitioning. The results of partitioning are used in various ways to identify the objects whose images are in the picture under analysis. Some of these ways are presented. The development of a small attributed grammar demonstrates an algorithm for identification of three-dimensional objects.

1. Introduction

As automation and robotics become ever expanding fields, it becomes increasingly necessary to develop a system by which a computer can interpret the picture it sees of a three-dimensional scene. Computers must be capable of visual processing. For a robot to carry out an order like "pick up the cube", it must be able to recognize the cube from the world it sees with its electronic eye. Its world is built from many parts and that fact becomes a useful tool in interpreting the scene. The initial goal of scene analysis is to identify the parts of the scene. Using those parts the process advances to the ultimate goal of naming those objects in the scene on the basis of descriptions and known constraints. This paper deals with work done on line drawings representing three-dimensional polyhedral scenes.

The original work in scene analysis was done solely by model recognition. There was no analysis of the various parts of the scene in context with each other. It was simply done by attempting to locate given shapes in a scene. L.G. Roberts, who began his work in 1963, is thought to have been the first pioneer in the area of scene analysis. Roberts took as the basis for his analysis three polyhedral models shown in figure 1-1. He attempted to interpret polyhedral scenes as transformations and/or compositions of his three models [1]. Later studies pointed out that this technique failed on some basic polyhedra. It was much too restrictive in what it could find in a scene. The need for a more general system led to advancement in the field of scene analysis as it is known today.

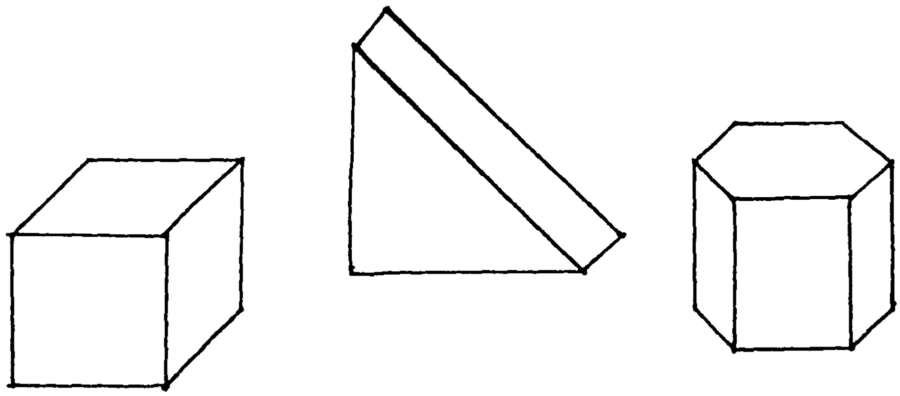


Figure 1-1: Roberts' Models [1]

2. Scene Analysis: What is it?

A scene can be considered to be an arrangement of things in three-dimensional space. The analysis of a scene by computer involves analysis of the two-dimensional image it projects. Analysis of that image involves the interaction of the input from the image itself and the knowledge associated with the visual world or space [10]. The polyhedral world provides a clear domain to study. There is a clear set of constraints to which the parts of the scene must conform. These constraints define how parts may interact, and lead to the ability to systematically recover shapes.

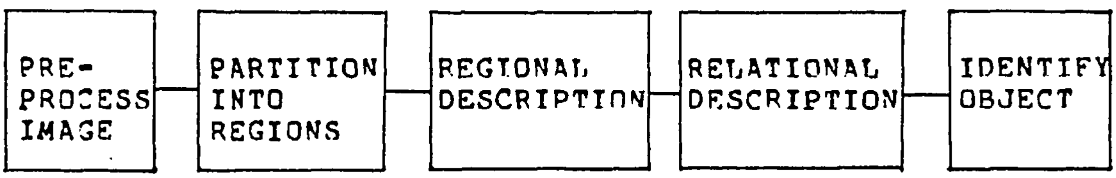


Figure 2-1: Scene Understanding System [11]

It is necessary to process the input picture of the scene. This processing begins with transforming the picture into a line drawing. The transformation is a complicated process based on things like light intensity and tone changes. This processing, though necessary, is not discussed in this paper. Once the line drawing is obtained the partitioning and recognition from that line drawing can be dealt with.

The process of scene analysis from line drawings began as the matching of model patterns and nothing else. These simple schemes were inadequate, and the process is now more complex. It basically involves two parts. The first part is the partitioning of the scene into objects or what Guzman calls bodies. The second part involves the analysis of those bodies by some means so as to arrive at an actual classification of them by name [19]. The concern here is to find properties which are of value to look at and compare. Kanade points out that there are two basic levels of analysis involved in these two steps. The first level is the qualitative analysis which deals with labelling junctions and lines to recover possible geometric shapes. The second level involves quantitative analysis which expands on the qualitative description. Here assumptions including accidental alignment and picture regularities are exploited to recover probable shapes [16].

The actual process by which a scene is partitioned requires the establishment of descriptions of the primitives of the scene leading to construction of a

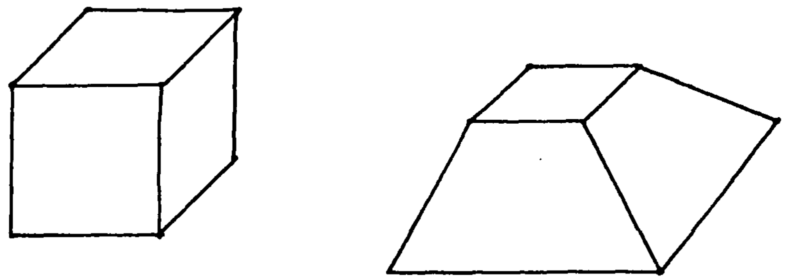


Figure 2-2: Line Drawings: Qualitatively Equal [16]

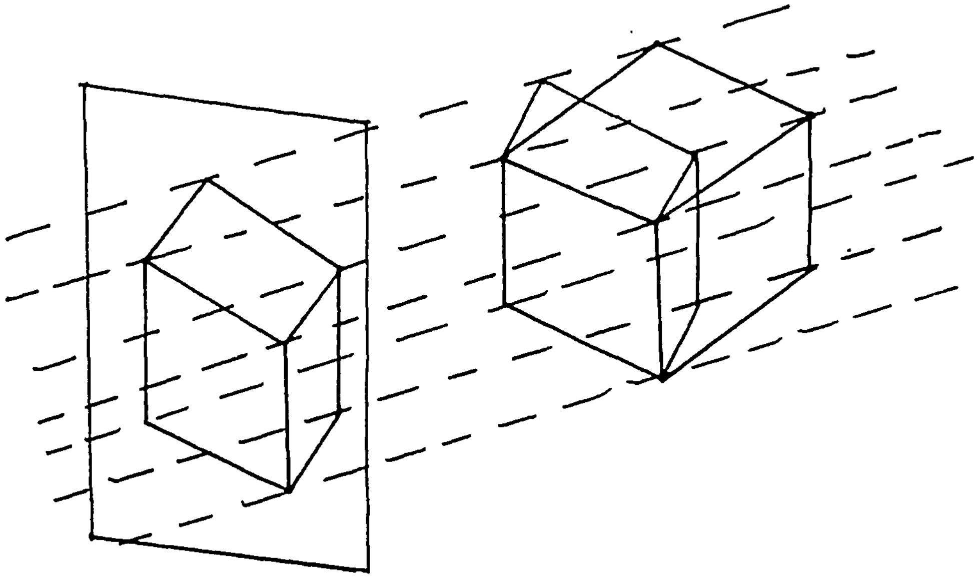


Figure 2-3: Different Shape - Same Line Drawing [16]

description of the scene as a whole. The primitives talked about here are the junctions, edges, and regions. These are labelled with qualifiers which must be consistent with the constraints of the given environment. From these primitives a description of the scene can be constructed. This description is then used to establish the number of objects in the scene, the spacial relationships between objects, and any relevant properties of those objects [26].

The process of identifying the objects in the scene by name is accomplished by matching properties of those objects in the scene to properties of objects in the world of knowledge. It places a great demand on the shape analysis algorithms. Decisions concerning which method is most accurate and least time consuming must be made. The methods of recognition can be classified in three general categories: masking, decision-theoretic matching, and syntactic pattern matching. The variations in the methods above deal with what kinds of things should be matched, how to go about matching them, and how exact the match must be. In all methods the decision as to what should be matched for recognition is vitally important.

3. Partitioning the Scene into Bodies and Descriptions

3.1 Partitioning

The recognition process for plane-surfaced three-dimensional objects involves the partitioning of a scene for identification by analyzing local and global information from the line drawing determined from the image of the three-dimensional objects. There are four basic steps.

1. Determining the characteristics of the primitives of the line drawing (i.e. junctions, lines).

2. Determining the regions formed by these primitives in the picture. (Regions are the two-dimensional bounded areas in a line drawing.)

3. Combining regions to form possible bodies. The bodies correspond to objects in the scene and regions to their surfaces.

4. Providing descriptions of the various possible bodies in terms of properties and relationships involving the regions and the primitives.

Because these are the steps involved in partitioning the scene for identification, the goal of work in scene analysis has been and is to accomplish and refine the use of these steps. Partitionings are not always unique but are based on certain heuristics to achieve the "most likely" analysis.

3.2 Initial Work with Picture Labelling

Guzman [10], in his PhD thesis in 1968, began work along these lines. His was the first work which pertained to using the characteristics of the image to describe it. Specifically, the classification of vertices shown in figure 3-1 was used. He used this classification, and the knowledge that the world under consideration consisted of polyhedra, to partition the scene into bodies. His goal was to find relationships between regions and use them to group regions into sets which might represent the images of actual objects.

Guzman accomplished this task by use of a program called SEE [10]. SEE performed a tree search creating links between regions thought to have good associations. These associations were based on a variety of heuristics concerning the junction types [17]. Links resulting from the heuristics could be strong or weak. Links may also be inhibited or prohibited by evidence found which suggests two regions should not be part of the same body, see figure 3-2. Once links were established regions satisfying the necessary conditions were put together to form nuclei. The process of merging was continued until there were no more regions which could be merged under the given set of rules. In this way SEE could partition a scene in a way similar to the way in which a human

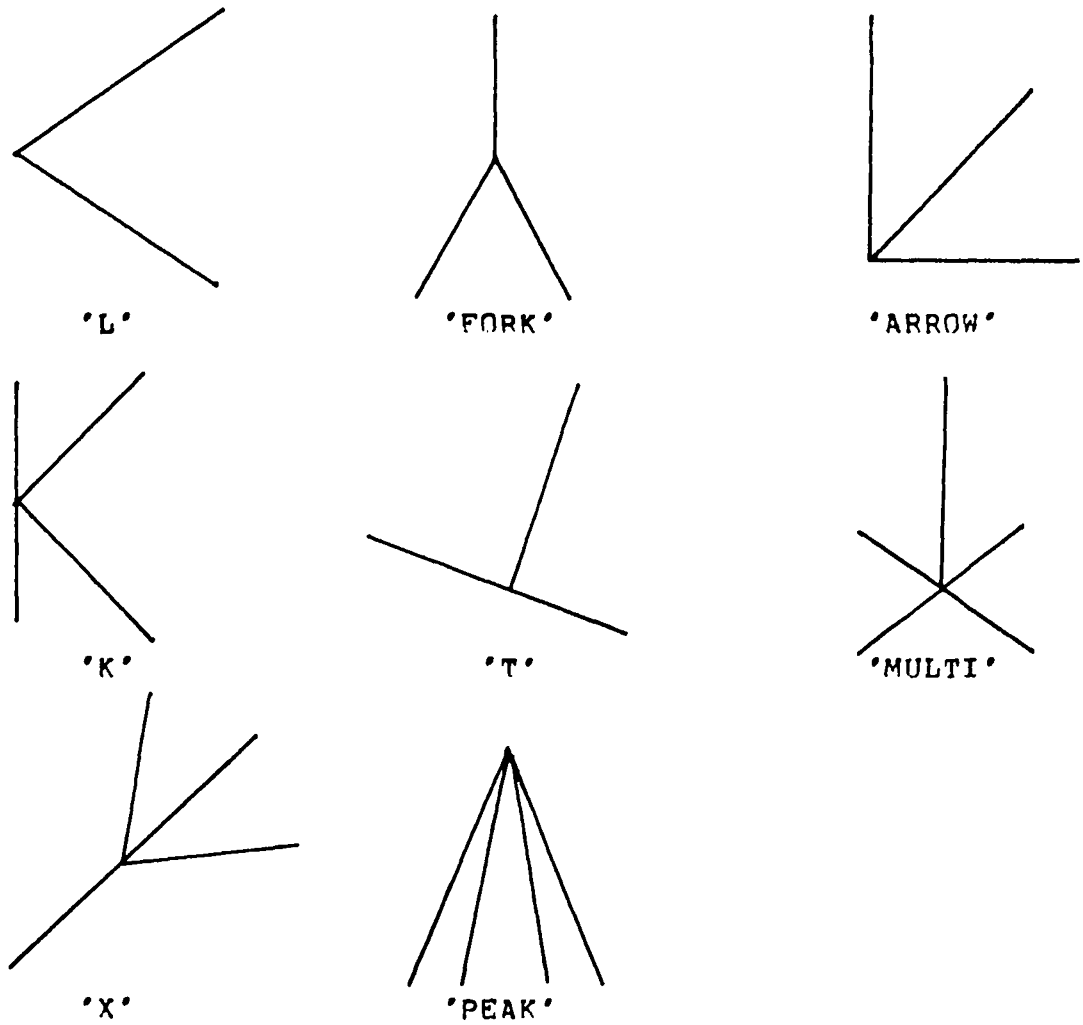


Figure 3-1: Guzman's Possible Vertices [10]

looking at the line drawing would.

In determining a possible partitioning of a scene Guzman's SEE fails in some aspects of achieving the final goal of partitioning. SEE does not consider all the constraints and characteristics of the polyhedral environment. SEE looks at links between regions based

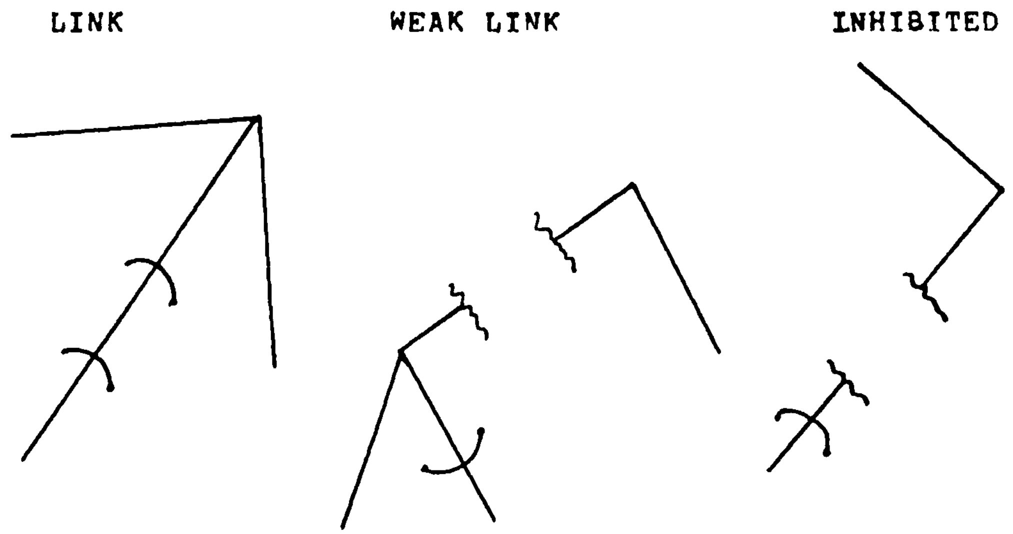


Figure 3-2: Guzman's Links [10] : 'ARROW'

only on the shape at the vertices. Because Guzman does not extend this to consider how two surfaces (the sides of the object in the three-dimensional scene) may connect, SEE is unable to recover the actual three-dimensional shape. When considering the steps in partitioning where Guzman fails is in the fact that SEE does not produce the necessary descriptions for indentification. Figure 3-3 shows SEE's result. It only shows that there is one body not anything more conclusive about that body.

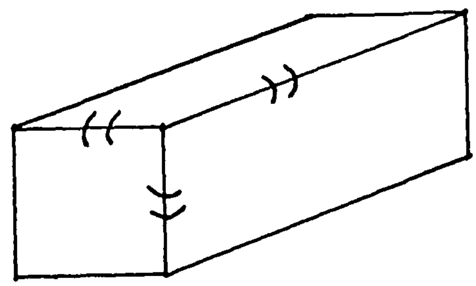


Figure 3-3: Is it an inset corner or a cube ?

3.3 Labelling Methods

The desire to use to a fuller extent what is known about the trihedral world led David Huffman of MIT and Max Clowes of the University of Sussex to elaborate on Guzman's classification of junctions. They dealt with labelling lines, categorizing junctions, considering occluded faces and consistency in labelling. Their method of labelling provided a clear definition of the object world. This provided a systematic approach to what was known and allowed them to use some clear heuristics. They compiled a junction dictionary and established an efficient labelling procedure which used filtering. Huffman presented the labelling scheme but did not write an actual program. Clowes developed an algorithm and program for the scheme.

The line labelling scheme is based on knowing how vertices may appear in a line drawing of, in this case, a trihedral world. Lines must be labelled in one of the following four ways: +(convex) or -(concave) for connect edges, and --> or <-- for occluding edges. The arrow in the last two labels is directed so that if you face in the direction of the arrow, the face to the right is visible and the face to the left is not visible in the picture.

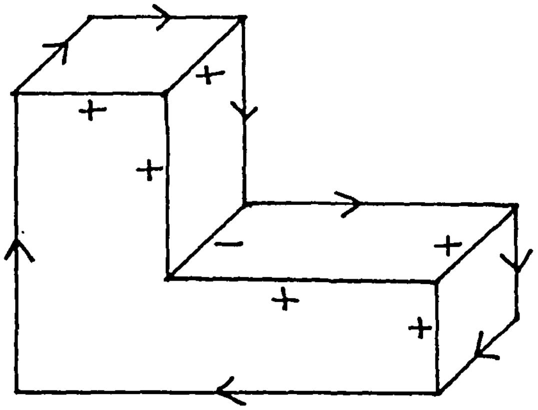


Figure 3-4: Huffman-Clowes Labelling [17]

A figure can only be a projection of an object in the trihedral world and thus realizable if all its vertices can be labelled consistently from the set of sixteen allowable corner labels shown in figure 3-6. A figure is labelled consistently only if every vertex is labelled and every line has only one label such that the line is not ambiguously defined [26]. The demand that the line labelling be consistent implies that only certain kinds of corners are allowed to be adjacent to one another. For example vertex 3 cannot be adjacent to vertex 7 because there are no lines coming into either corner that have matching labels. One way to achieve a consistent labelling is shown by Waltz. He suggests that the labelling be done using a pairwise tree search. He compared labels of adjacent vertices and eliminated any which contain the kind of inconsistency shown in the previous example. Waltz [26] repeated this pairwise search until no more labels could be eliminated. A case where no consistent labelling for all lines can be found is shown in figure 3-5.

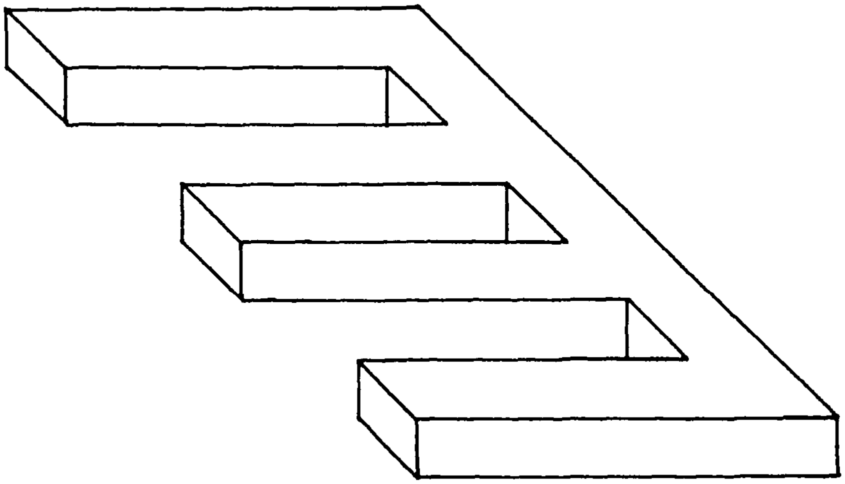


Figure 3-5: Non-labelable Drawing [21]

Clowes [3] realized that it is important to adopt a notation in which corners are described by more than their class membership. He described them in such a way so as to specify the corners in terms of surfaces and edges at the corner and their relationships. A given picture fragment may correspond to a variety of scene

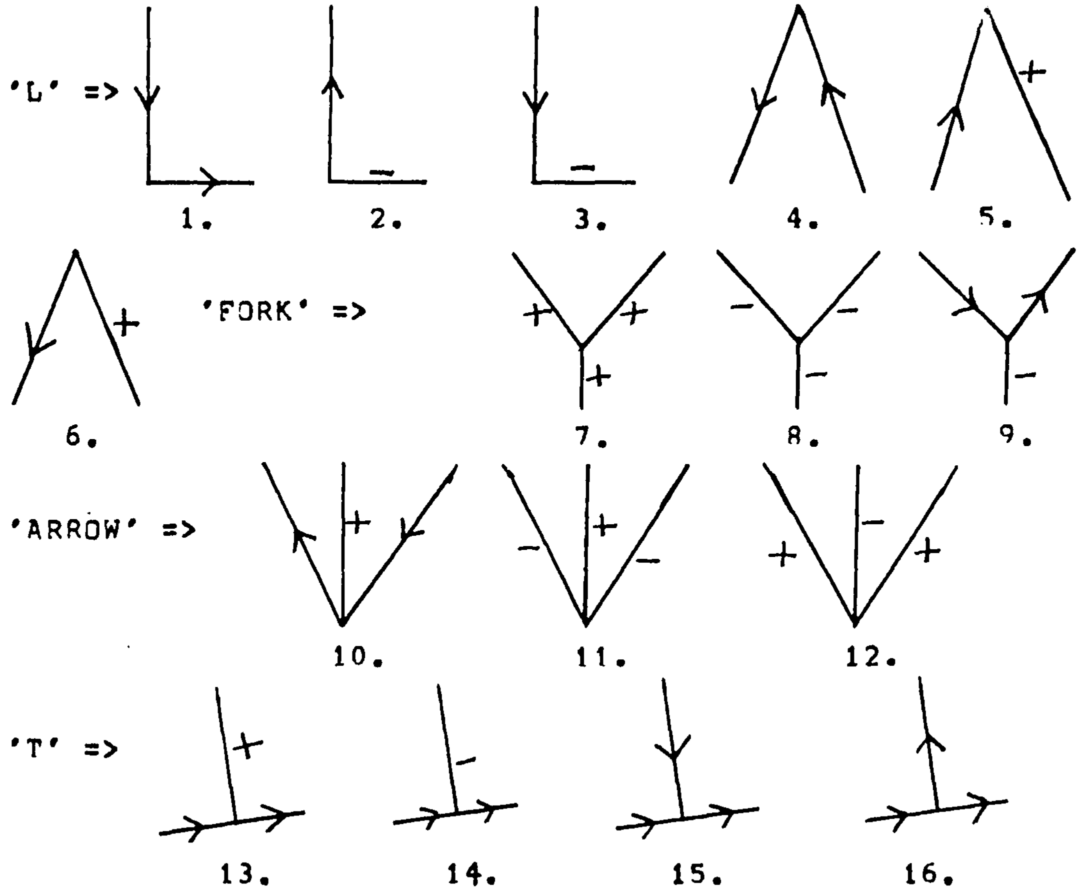
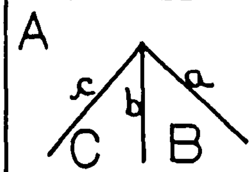


Figure 3-6: Vertex Labels: Huffman-Clowes [26]

fragments (see figure 3-3) and thus labelling is not always unique. Clowes was sure to consider all possible scene fragments for the various corners when he set up a predicate table. The table includes information about the scene fragments. A vx predicate refers to a convex edge. The cv predicate refers to a concave edge. A hind predicate refers to the fact that an edge at the corner is hidden by those which are visible. Clowes then used this table to describe a method of scene analysis based on the labelling process.

JUNCTION	CORNER	CANONICAL NUMBER	SCENE FRAGMENT
'ARROW'		2	vx(ABa) cv(BCb) vx(ACc)

vx: convex
cv: concave

Table 3-1: Clowes Predicate Table Entry [3]

The partitioning method described by Clowes [3] involves finding a labelling which is compatible with respect to all its parts. Clowes does this by beginning at the junction level and building up from there by grouping parts under certain rules to form possible larger pieces. The first step is to look at the picture and consider the junctions along with Clowes' predicate table (see table 3-1). This information is used to assign to each junction a set of descriptions called scene fragments in the table. The descriptions describe the surfaces and lines at the corners: see figure 3-7. The figure in 3-7I can be thought to represent the edge of a box where B, C, and D are visible sides and A is the table or backround. In this interpretation surfaces B and C are considered to be part of corner 2 because they are two of the three surfaces which form the corner of the box. Surface A is not part of corner 2. B and C are also visible surfaces while there is another surface which is part of corner 2, E, that is not visible. It is also possible that a junction may not even represent a corner as in the case of some "T" junctions not pictured in this figure.

After the descriptions have been assigned, pairs of corners are constructed such that the pairs could represent edges in the scene. The creation of these edges is based on the fact that for the common line the predicate relations must match. For example in figure 3-7 the following descriptions apply.

For figure 3-7I:

1. vx(BCb) vx(CDc) vx(DBd)

2. vx(BDa) vx(BDd) vx(DEe) hind(BAa) hind(DAe)

For figure 3-7II:

3. vx(BCb) cv(BDd) vx(CDc)

4. same as 2 in figure 3-7I

Line d in 3-7I is acceptable as an edge because the description containing the d component in both corner 1 and 2 is vx(BDd), thus the descriptions are compatible.

Line d in 3-7II is not a possible edge because the label vx(BDd) at 4 says d is convex and the label cv(BDd) at 3 says d is concave.

In a similar way, edge descriptions can be connected "end-to-end" so that they form a closed area to find possible surfaces. If descriptions that form such an area also describe the common region they enclose with a compatible descriptions, the region may be considered to be the surface of an object. The hind predicate is useful in determining when surfaces are partially occluded.

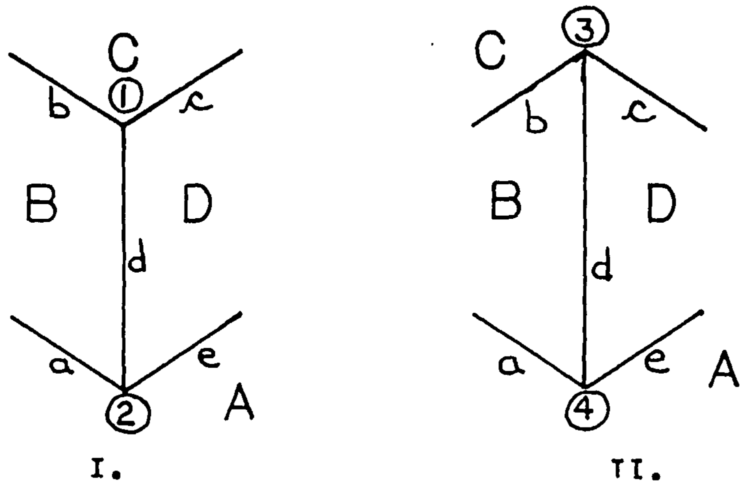


Figure 3-7: Edges with Clowes Labelling

A body description can be found by connecting adjacent surfaces. The same rules apply as before. The connection must be cyclic thus the body is closed and common edges must have compatible descriptions. The labelling process for a picture is complete when all picture regions are accounted for. Each picture may have more then one possible labelling based on combinations possible when building from the junctions.

OBSCENE is the name of a program in which the progression of unions derived by Clowes is applied. OBSCENE begins with the corner level description. The unions for building are based on coherence rules between the edges and the surfaces. OBSCENE provided an advancement from SEE but still did not take into account the viewing angle and depth.

The advancements of Huffman and Clowes were important contributions to scene analysis. They made effective use of the necessity for consistency in a scene. There are several inadequacies involved in the labelling process and OBSCENE. These inadequacies stem from the fact that the labelling scheme is based on two assumptions: that the world is trihedral and that there is a "general viewpoint" camera position [5]. In the trihedral world corners can only have three surfaces, picture junctions must be the intersection of two or three lines and all surfaces are planar such that surfaces which share an edge are not coplanar. Figure 3-8 shows some junctions which could not be accounted for by this method.

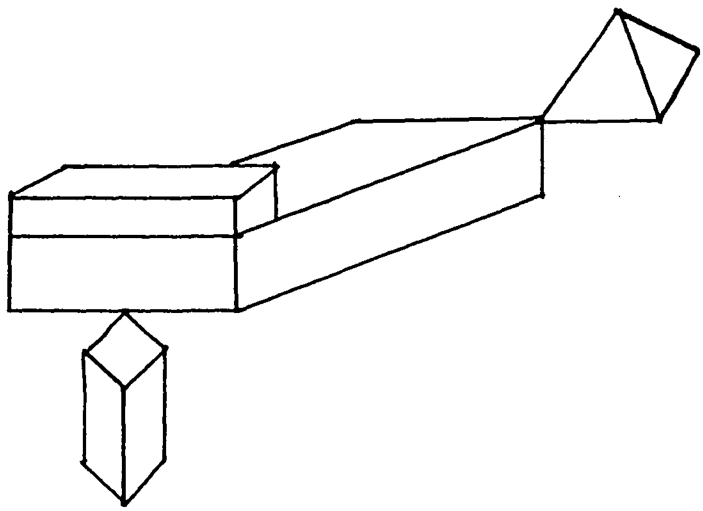


Figure 3-8: Junction Not Allowed in Trihedral World

Clowes' OBSCENE did allow for more than three lines at a junction but the other restrictions remained. A "general viewpoint" is necessary to avoid accidental junctions. Huffman [13] demonstrated the problem caused by this assumption as it allows unlikely pictures to be labelled in a unique way. He also showed how some unrealizable objects can be consistently labelled. These failings made clear the fact that further work was needed towards a goal of geometric adequacy. Waltz introduced shadows and cracks to the things he analyzed. He allowed some non-trihedral vertices and accidental junctions though his method could not handle them in general [5]. The method Waltz used is not however discussed here. A goal to establish still a more quantitative basis for partitioning is what prompted further work.

3.4 Quantitative Approach: Dual and Gradient Space

The work in gradient space came as the result of a desire to lift some of the restrictions formerly placed on scenes for analysis. In association with this work pictures involving straight line segments and scenes made of opaque polyhedra could be considered. Mackworth [5] created a program for this world trying to achieve geometric adequacy. Gradient space is based on coherence rules between surfaces and edges which are determined by the knowledge of gradient space. Previously the partitioning results had been based on the categorizing of junctions [18]. This method is used to check the labelling from a geometric standpoint. It is helpful to define several terms here for clarity. A picture is the two-dimensional representation made up of line segments and regions. A scene is the actual three-dimensional object made up of edges, vertices and surfaces. A connect edge will correspond to concave and convex edges in the previous work. The goal is to determine the relationships between the various regions from a valid geometric standpoint.

The basis for this geometric approach is a set of properties belonging to gradient space. Gradient space is a two-dimensional projection of dual space. It is important to note some of the characteristics of dual space. The representation of a figure in conventional space by a figure in dual space is such that points map into planes, lines into lines, and planes into points. The origin is to be the viewing point, and planes through the origin are not considered. The equation of any plane not through the origin can be written as

$$ax + by + cz + 1 = 0$$

The plane is mapped into the dual point (a,b,c). Since points map into planes and connecting lines of two planes must pass through the points which represent the planes connected by that line, a corner made of n surfaces is represented in dual space by a planar n-gon. The major concern here deals with the projection of a three-dimensional body into a two-dimensional picture. This image can be considered in terms of a viewing position and the picture plane [18] (see figure 3-9). Any edge, its projection, and the viewing point must lie in the same plane called the plane of interpretation.

Gradient space then is a two-dimensional projection of dual space. A dual point (a,b,c) corresponds to the gradient point (a/c,b/c). The line a=0, b=0 is taken to

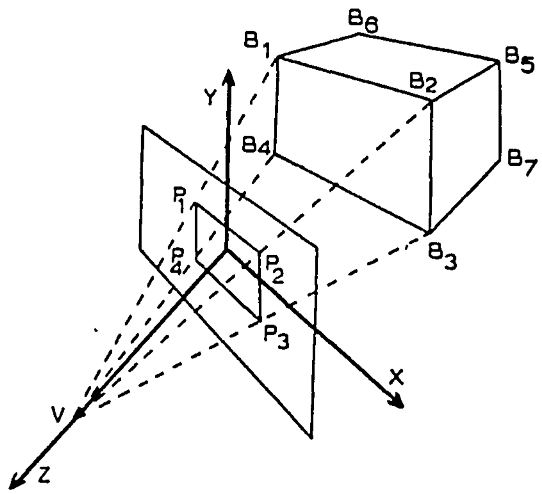


Figure 3-9: Picture Taking Process [18]

e the viewing axis. Planes with c=0 correspond to planes parallel to the viewing axis and are not represented in gradient space. See figure 3-10. The projection is made with the center of the projection at O, the origin of the dual space, and (0,0,1) the origin of the gradient plane. Note that in figure 3-10, -c is the distance from point I to the x-y plane of the dual space. Since parallel planes have the same a:b:c ratio, they correspond to the same point. If a point S in the gradient plane is chosen, the length of the vector from O_G to that point S is the tangent of the angle between the picture plane and the plane corresponding to S. In the case where the gradient point is G=(0,0) the tan(a) = 0 and thus a=0 so G represents planes where z=-c which are perpendicular to the view line. A gradient point G1=(p,0), p>0, would give tan(a) = p. Thus 0<a<90 and the point G1 would correspond to planes with a dip of between 0 and 90 degrees relative to the picture plane. The larger p, the more slanted the surface being considered [16]. This knowledge may be useful in reverse. Knowing the equation of a surface plane and the picture plane allows the calculation of the distance the gradient point representing a surface plane should be from the gradient origin.

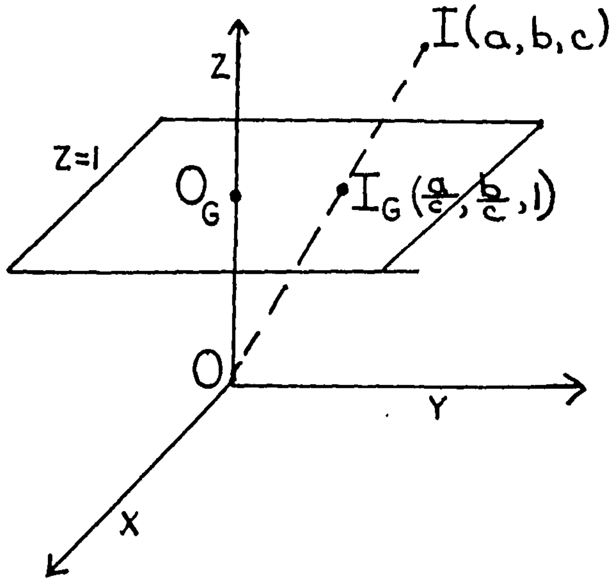
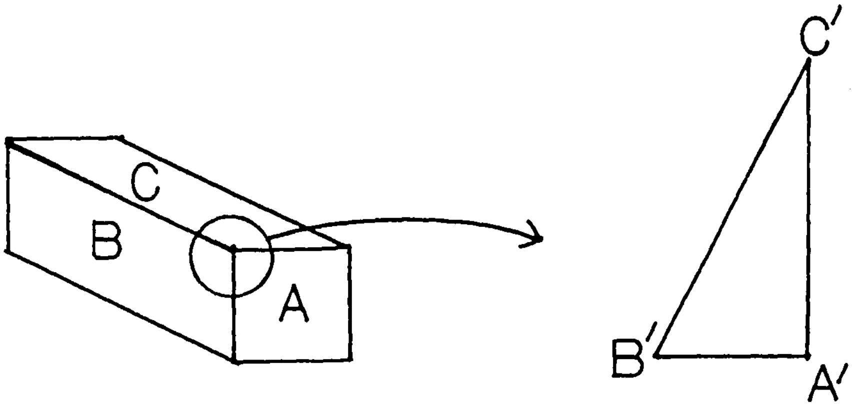


Figure 3-10: Projection: Dual to Gradient Space

Along with the points and the planes, the edges must be considered. The gradient line is simply the projection into the gradient space of the dual line. As a result it carries most of the characteristics of dual lines: specifically, the property that says if a line represents a connect edge, the gradient line must pass through the points which correspond to the surfaces it connects. It may be noted here that the points will be ordered in the same order as the surfaces on an edge which is convex and in reverse order on an edge which is concave. The orientation of a gradient line is determined by the picture line since they must be perpendicular. Thus a gradient diagram like the one in figure 3-11 can be obtained. It has been pointed out that gradient diagrams are not unique, because a gradient diagram may move or change sizes. It may not however change shape. That is determined by the line drawing [21].

With a background of this theory Mackworth [18] constructed a program which used the coherence rules established in gradient space. The program is called POLY the outline of which is in figure 3-12. The rule stating that the points representing two connected surfaces lie on a line perpendicular to the connect edge played a major role in Mackworth's program. POLY's goal



a. Line Drawing b. Gradient Diagram

Figure 3-11: Gradient Space Diagram

is to determine which edges are connect, the details about the edges, and the orientation of surfaces and edges. From the two-dimensional line drawing it is not possible to calculate the equation of each surface to determine the exact gradient point, since the third coordinate is unknown. As a result, there are some arbitrary selections POLY must make during its analysis.

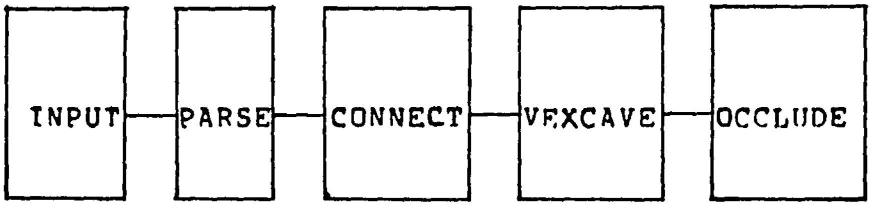


Figure 3-12: Organization of POLY [18]

CONNECT is the portion of POLY which finds the connect edges. It does this by trying to make all the edges connect edges at first guess. Upon finding that this is inconsistent with the coherence rules, CONNECT reduces the number of connect edges and tries to establish a labelling again. It continues this process until it obtains a coherent labelling. The exact process can be explained in conjunction with figure 3-13.

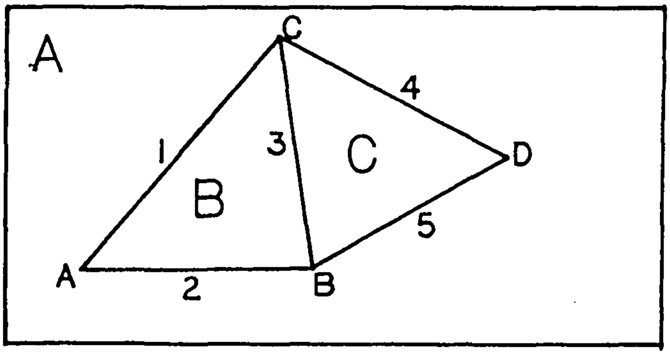


Figure 3-13: CONNECT example

The first point is chosen arbitrarily and will represent the origin for the gradient space diagram. In this example G_A (gradient point A) will be the origin. If 1 is to be a connect edge then, to comply with coherence rules discussed previously, G_B must lie on a line through G_A perpendicular to 1. Since G_B may lie anywhere on such a line, G_B is also arbitrary to some extent. This early use of arbitrary selection is usually referred to as arbitrary choice of origin and scale [5].

If 2 were now to be established as a connect edge, then G_B must also lie on a line perpendicular to 2 through G_A. Since this is impossible both 1 and 2 cannot be connect edges. 3 could be a connect edge if G_C were on a line perpendicular to 3 through G_B. Now 4 may also be established as a connect edge because placement of G_C can be made so it is also on a line perpendicular to 4 through G_A. The progression for constructing this gradient diagram with CONNECT is shown in figure 3-14. This method is called triangulation.

After establishing connect edges and constructing a diagram, POLY continues to work with the edges. It uses VEXCAVE to determine which connect edges are concave and which are convex. Because picture lines are perpendicular to their gradient duals, it is known that if the faces in the picture are ordered the same as the corresponding points the edges are convex. If they are in reverse order, the edges are concave. Figure 3-15 shows an example of this. Finally POLY uses OCCLUDE to determine details regarding the non-connect edges using two rules. The first rule is if two non-connect edges intersect at a connect edge it can be determined which

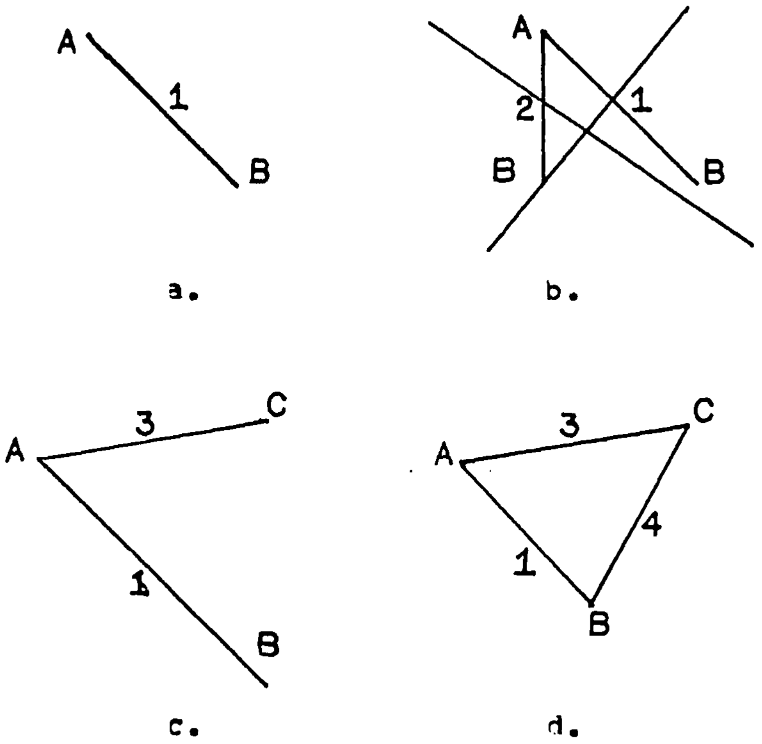
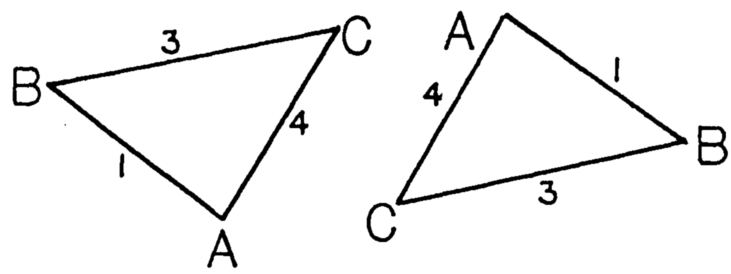


Figure 3-14: Gradient Diagram Construction

surface is in front of the other and to which surface the edge belongs. This rule is used to add the hidden surface at the edge being considered. The diagram is then completed by use of the second rule which adds the minimum number of hidden surfaces to satisfy the fact that a corner made of n surfaces is represented in gradient space by an n-gon. After one interpretation progresses through OCCLUDE other possibilities are looked for in CONNECT and the progression of analysis continues until all coherent labellings are established.



a. Convex b. Concave

Figure 3-15: Convex - Concave Labellings

Mackworth's work was based on properties of gradient space and thus dealt with surface properties as opposed to junction properties. From a gradient interpretation it is possible to determine information about the tangent of the angle between a surface and the picture plane. This is information about the orientation of the surface not about its actual position. The position as such cannot be determined because the depth parameter is missing. No absolute values for the unknown coordinates in the gradient space can be found, they can only be calculated in terms of each other [5].

Huffman [12] wished to expand work done with dual and gradient space to set up conditions of closure for a set of lines in a scene and thus determine the realizability of configurations. Thru a sequence of calculations he demonstrated the following. If (x,y,z) represents the coordinates of a point in a scene and (u,v,w) represents the coordinates of a point in dual space, the rate of change of z with respect to the distance along a picture line is equal to the distance from the origin to the corresponding line in the gradient space. Along the same lines, the rate of change of w with respect to the distance along a gradient line is equal to the distance from the origin to the corresponding line in the scene. It is also observed that the distance from the origin to the point in the gradient diagram is equal to the tangent of the angle of tilt of the surface corresponding to that point as mentioned, before. Using these facts a path which appears closed in a picture can be analyzed to see if it is really closed or not. The amount z changes along a picture line l_i with slope s_i is $l_i s_i$, where l_i is the length of the line. A closed path must start and end at the same point, therefore for a path to be closed the net change in z must be

$$0 = \sum l_i s_i .$$

If this is not the case, the path is not closed.

With these concepts and the idea of cut sets Huffman tried to establish a method to determine the realizability of a scene. A cut set of picture lines is the set of picture lines that "enter a simple closed region of the picture from outside that region". It should be noted that movement along the surface of a polyhedron generates a corresponding path in the dual scene called a trace. This trace consists of a sequence of points associated with surfaces of polyhedra connected by lines associated with the edges. Huffman's [14] claim is that a certain labelling is realizable if and only if the trace corresponding to it is a closed path. The closure of the path could be shown to be false by finding a region phi, containing possible picture origins, to the right of all directed line segments in a cut set. The path is not closed in this situation because if phi exists all changes in w are positive and thus the sum

$$\sum w_i * l_i \quad \text{does not equal zero.}$$

Huffman [14] states that testing all possible subsets in this way is a necessary but not sufficient condition to determine the realizability of a configuration. Draper [5] finds that this inadequacy is the result of a movement from the global characteristics beginning to be dealt with in gradient space back to line labelling and use of restricted areas of analysis.

The use of dual and gradient space came about as the result of a desire to make the partitioning process based more on geometric, quantitative information. This approach worked on a less restrictive environment. The approaches developed along these lines were more surface oriented then the line labelling achieved previously. Mackworth began his work by finding and using the coordinates of the end-points. POLY, his program, took these end-points and searched over connect and non-connect edges for possible labellings. Improvements have been suggested for the work done and new approaches outside of the dual space concept have been motivated by this work.

3.5 Sidedness Reasoning

Stephen Draper [5] presents a new approach which he claims combines the ability to do the partitioning task competently (like the plane equation approach) while still using the geometric theory (the geometric approach). Some specifics must be recalled here. Any plane divides space into two halves. Line labels on lines where two planes intersect allow the determination of which plane is in front of the other. This is the basic geometry on which Draper's proposal is based.

The characteristics of Draper's theory are these. Properties of planes and knowledge of connect edges allow certain claims to be made. The rule generated here is that two faces A and B which meet across a concave edge are such that everywhere on A's side of the line (extended if necessary) A is in front-of B and on B's side A is behind B [5]. For convex edges the rule is reversed. This rule leads to certain statements about the relationship between planes. It makes possible the rejection of incorrect labelling. This is illustrated in figure 3-16 below by considering how C relates to A and B. Label 1 implies that visible A is behind extended C therefore 2 is behind C. Label 3 implies the visible B is in front of C therefore 2 is in front of C. There a contradiction has been reached and the labelling is deemed incorrect [5]. The consistency of geometry provides this logical sideness reasoning.

Sidedness reasoning can reason without depth equations. It can also determine valid labellings for figures when considering concurrent or non-concurrent "cyclic sets of four". A cyclic set of four is a set of four faces each of which shares an edge with two of the other faces. The set is concurrent if the edges meet at a point. Sidedness reasoning demonstrates the impossibility of the concurrent, cyclic set of four in

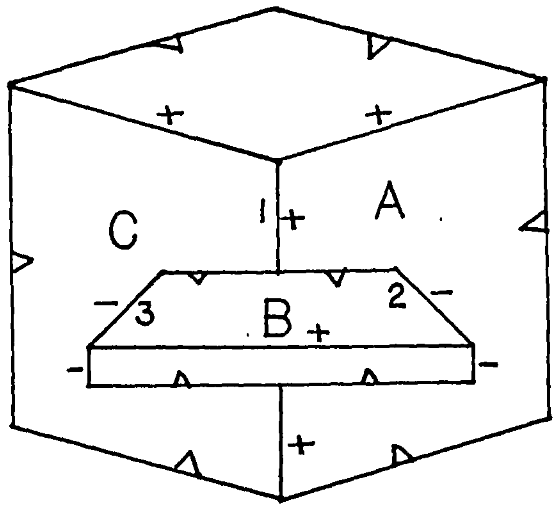


Figure 3-16: Sidedness Relations [5]

figure 3-17. 2 is behind A since 1 is convex making the visible part of A in front of B. 2 is in front of D since 3 is concave making the visible part of C in front of D. 2 must have the same depth value on both A and D since it is collinear with 4 and 4 is on both A and D. Therefore there is a contradiction because 2 cannot have the same depth in terms of A and D and still be behind A and in-front of D.

This reasoning combines occlusion information with that of connect edges and therefore is not totally dependent on the connect edges as Clowes' method was. The basic development for an actual program would involve

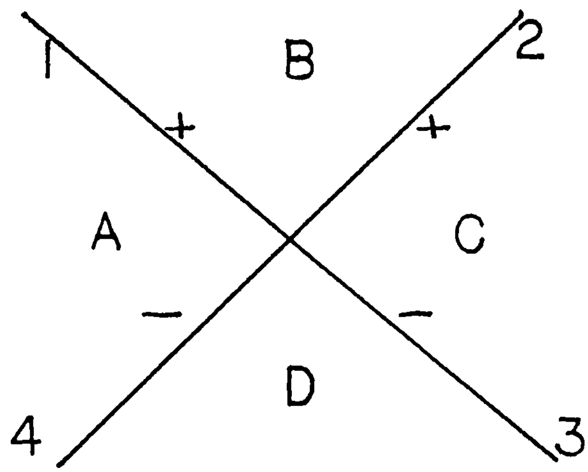


Figure 3-17: Concurrent Cyclic Set of Four

taking the possible line labellings and translating them into sidedness constraints. It would check for consistency against those already accepted. An inconsistency would cause the program to reject the labelling and go on. Sideness appears to be a promising look at a method of partitioning. It has the quality of being geometrically correct and easy to understand.

3.6 Linear Algebra for Realizability

Even though extensive work has been done along with partitioning in the area of realizability of configurations, the problem of recognizing incorrect line drawings is still being examined. The method in which line labels are checked for consistency is one of local analysis and results in probable interpretations. These interpretations may contain "global inconsistencies" [22]. It is desirable to be able to tell whether a line drawing correctly represents a polyhedron. Methods previously discussed seem to provide necessary but not sufficient conditions. The purpose as discussed by Sugihara and Shirai [22, 21] of the linear algebra is to provide those sufficient conditions.

The discussion has been on labelling line drawings which are projections of three-dimensional scenes. All labelable pictures however are not necessarily representations of polyhedra or unique. Once the labelled picture is given it is possible to determine its realizability using linear algebra. The drawing is considered to have n vertices and m faces. Assume in a line drawing (x_i, y_i, z_i) are coordinates for vertex V_i, where z_i is unknown since two-dimensional space is the domain.

$$a_j x + b_j y + z + c_j = 0$$

is the equation for face F_j where a_j, b_j and c_j are also unknown. It is possible to obtain L_j linear equations for the L_j vertices on face F_j. There are $L = L_1 + L_2 + \ldots + L_m$ equations where the number of unknown variables is n+3m. Let $\vec{w}$ be a vector whose components correspond to the n+3m unknown variables. Then a fundamental system of equations for the picture is

$$A\vec{w} = 0$$

where $\vec{w} = (z_1 \ldots z_n\ a_1\ b_1\ c_1 \ldots a_m\ b_m\ c_m)^t$
and A is a coefficient matrix determined by the picture [22]

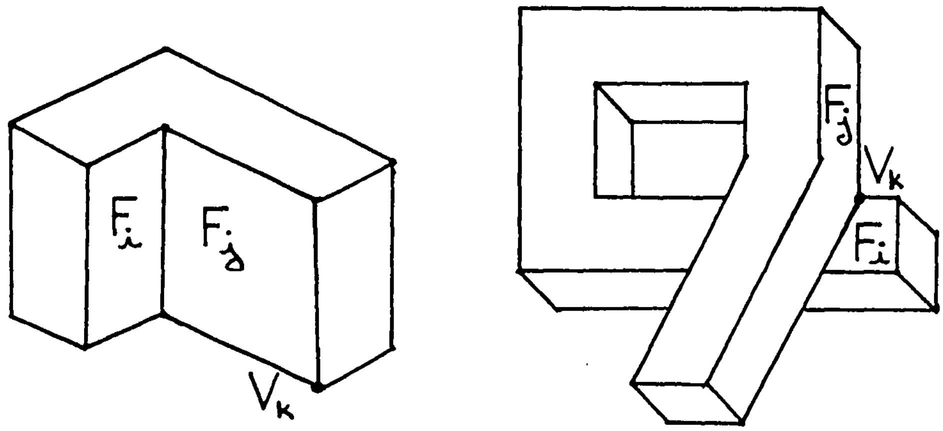
Some observations about the system can be made. One important observation is that there is no difference between the algebraic structure of an orthographic and perspective projection so this system does not depend on the eye position of the camera.

The picture can give clues as to the depth of parts of the scene. Three kinds of cues give information along these lines. First the physical edge, second the thickness of the body and third the existence of occlusion can all be used. In the first clue if the edge where F_i and F_j meet is concave and V_k is on F_j as in figure 3-18, then

$$a_i x_k + b_i y_k + z_k + c_i > 0.$$

This kind of inequality could be determined for any of the faces and edges. The inequality above becomes >= because V_k might be on F_i since it intersects with F_j. This is related to a range of values pertaining to the thickness of a body. If there are 'T' junctions along occluding lines a new vertex is introduced on the appropriate face as in figure 3-18 and V_k should be closer than F_i. This can be represented by the inequality just developed and a set of these obtained for all 'T' junctions. This set of inequalities for vertices and faces is reduced to

$B\vec{w} > 0$ where B is a coefficient matrix.



a. Edge Properties b. Occlusion

Figure 3-18: Cues for Algebraic Analysis [22]

Finding a $\vec{w}$ where $A\vec{w} = 0$ and $B\vec{w} < 0$ is a necessary and sufficient condition for realizability. [21] It is possible then to determine the correctness of a line drawing. This work accomplishes what Huffman failed to do in his work with cut sets. Developments from here with linear algebra lead to the ability to correct line drawings based on algebraic rules. This is not discussed in this paper.

3.7 Summary of Partitioning

The goal in the analysis in scenes in this part has been the partitioning of a picture which is a two-dimensional representation of a polyhedral scene. The process for doing this has varied through time and from person to person. It is the aim of each worker to break the picture into bodies and their descriptions. The process involves the four steps that are mentioned in section 3.1. Not every researcher applied all four steps but the most recent work has made use of them. The principle of breaking the scene into descriptions of its parts and building a complete scene description has, however, always been the underlying theme. The methods by which this is done have progressed from edge and junction labelling, to surface orientation properties, to a stricter use of algebraic principles.

An important part of partitioning is a labelling process. The labels as descriptions for the parts are generated in some way and then tested for consistency and realizability. Table 3-2 indicates the various techniques discussed and the subsets with which the interpretations are involved. It can be noted that there is nothing listed under tester in the Huffman-Clowes method. Clowes [3] did however test labels for consistency along lines when using this method in

OBSCENE. More recently such labels have been tested with the use of linear algebra by Sugihara and Shirai [22, 21]. The process of providing a partitioning and description of a picture for scene analysis is the first major step in recovering the objects in a scene.

Method	Generator	Tester	Subset
Huffman-Clowes	Trihedral junction dictionary		S_{tri}
Mackworth	Sequential Generation of most connected inter-pretations	Constructive test on coherence rules in gradient space	S_{poly}
Huffman		Phi(Phi') point test for all cut sets	$S_{phi(phi')}$

Table 3-2: Labelling Schemes [15]

4. Recognition of Objects in a Scene

4.1 Elements of Pattern Recognition

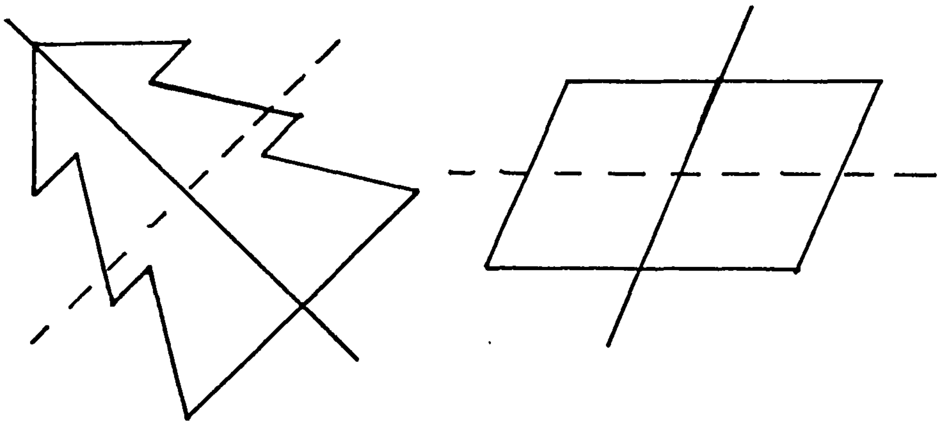
Once the scene has been partitioned into individual objects it is the goal to be able to determine exactly what those objects might be. The descriptions from the partitioning stage include sets of regions which belong to the same body, the kinds of edges which connect the regions and an analysis of which regions are "in-front" or occluded. Qualitative junction types may also be included. The first step in recognizing the objects involves expanding the descriptions to include more quantitative information. Things such as parallelism and symmetry help narrow down the search for the type of region being considered. These regions and their boundaries serve as primitives for recognition of the object as a whole.

As mentioned before, there are three general approaches to actual recognition of objects in a scene. They are the masking (template matching) method, the decision-theoretic (discriminant) method, and the syntactic (structural) method [7]. Masking or template matching is the simplest approach. It was the approach Roberts used. There is a template pattern and a match to that template is looked for in the scene. This is useful in recognizing printed characters but will not be elaborated on in this paper. The decision theoretic approach requires the matching of a given number of features between a model pattern and the decription of the object from the scene. Exact matches are not always required so the level of similarity in features which constitute a match is an issue. A diagram of this method is found in figure 4-3. The syntactic approach involves representation of a pattern by a string or tree or graph structure. This structure is built from the primitives in the scene. Because of this kind of representation it is analogous to parsing strings in a language and can be talked about in that way. The exactness of matches here is also an issue. A diagram of this approach is in figure 4-4.

4.2 Extending the Descriptions of Objects

Identification of objects in a scene requires knowledge of the parts. This knowledge comes by way of descriptions of the objects in terms of quantitative information. During partitioning some descriptions are obtained but it would be more useful to extend these descriptions and thus close in on recognition of an object. The important part of this process is choosing elements for the description which can be found and are able to identify an object. Since the scenes discussed are images of polyhedra it is necessary to recognize the polygons which are their component parts. These are such thing as squares, triangles, and trapezoids. Characteristics which identify polygons are numerous. It is worthwhile to look for edges which are equal, edges which are parallel, and measures of angles. A heuristic to be used in establishing these descriptions is that there are no accidental regularities. For example two lines parallel in the line drawing are assumed to represent two parallel lines in the scene.

Some of the traits mentioned above can be calculated directly from the coordinates of the line drawing obtained thru preprocesssing the image. Lengths and slopes of edges are two such traits. Since a non-special angle of viewing is assumed, lines parallel and equal in the line drawing can be assumed to be so in the scene. Kanade [16] points out that any time there is a skewed symmetry in a line drawing it probably corresponds to a real symmetry in a scene. Skewed symmetry means the transverse axis is not necessarily perpendicular to the symmetric axis but must be at a fixed angle (see figure 4-1). The region in figure 4-1b is assumed to be the projection of a rectangle. Because the skewed symmetry probably indicates a real symmetry, a trapezoid or a rectangle as see in figure 4-2 are the possible scene elements corresponding to 4-1b. There is a skewed symmetry in the other direction that probably represents a real symmetry, as a result 4-1b is considered to be the projection of a rectangle and not a trapezoid.



a. b.

Figure 4-1: Skewed Symmetry

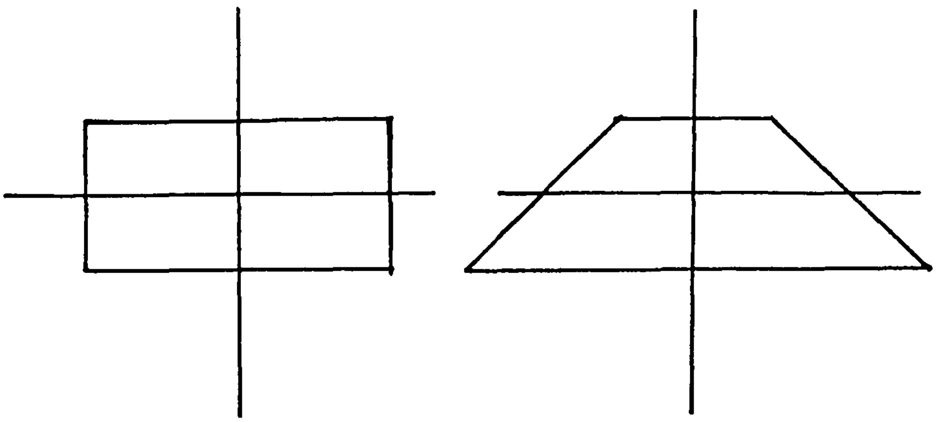


Figure 4-2: Trapezoid and Rectangle Possible

Other characteristics can also be extracted from the line drawing. The number of edges of a region is easily calculated. After partitioning, the number of visible regions (faces) on an object can be determined. The angles at which edges intersect can be calculated in the two-dimensional space. The angles at which the edges intersect in three-dimensional space would require knowledge of the third dimension. It is possible to determine the angle at which faces intersect thru use of gradient space if such third dimensional knowledge is obtainable. An earlier discussion showed gradient diagrams can be used to find the slant of a face with respect to the picture plane. These calculations can be applied to intersecting faces to determine the angle at which the faces intersect if the third dimension can be found. Details about numbers of sides and angle measure help classify regions and objects.

Polygons and hence polyhedra can be viewed as a set of definitions and constraints. Breaking down a line drawing into descriptions involving its traits becomes all important. The qualitative and quantitative geometric descriptions provide a basis from which to make comparisons and decisions as to what an object is. The question now is how those decisions are made once the descriptions are established.

4.3 Decision-Theoretic Approach to Recognition

One of the major approaches to the job of pattern recognition is the decision-theoretic or discriminant approach [9, 23]. This type of approach is based on the use of decision functions for comparison of the pattern being considered and a model or sample set of patterns. A diagram for this method is shown in figure 4-3.

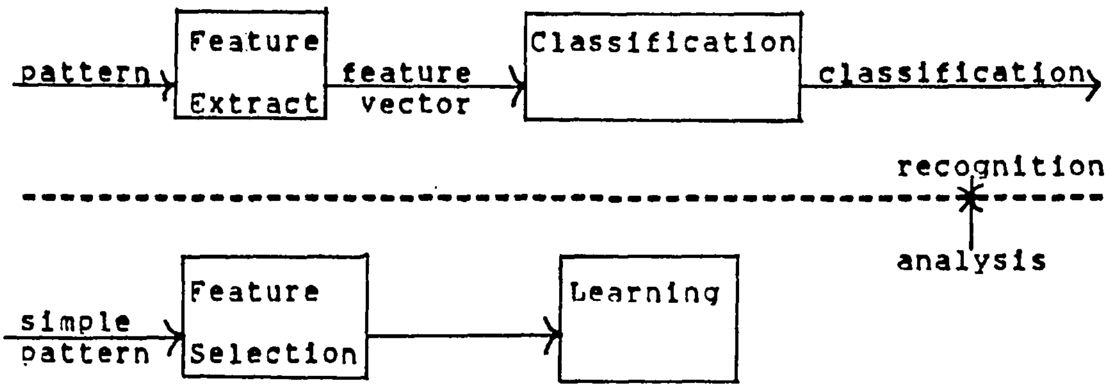


Figure 4-3: Decision-Theoretic Model [7]

For decision making, a set of features must be selected to construct a column feature vector $\vec{x}$ [23]. This is of the form

$$\vec{x} = (x_1, x_2, \ldots, x_N)^T$$

where $\vec{x}_i$ represents the i^{th} feature descriptor of a given object. x_i could be any of a number of things like the sum of the angles, the lengths of the edges, or the number of edges. The more uniquely the features chosen for the vector identify the object the better the system. It is optimal to find features which provide necessary and sufficient conditions for the objects being identified.

It is not always possible or practical to include every little feature in the vector. Because certain features are more important toward identification, they can be given weights. There is a weight vector as follows

$$\vec{w} = (w_1, w_2, \ldots, w_N)^T$$

associated with each pattern class. The features and their weights can be used in conjuction with each other to establish identification.

Once features have been chosen the problem involves extracting those features from the object under consideration and applying decision functions. If M classes are being considered, there will be M decision functions $d_1(\vec{x}), d_2(\vec{x}), \ldots, d_M(\vec{x})$ defined so that

$$d_i(\vec{x}) = \vec{w}_i^T \vec{x}$$

where each $\vec{w}_i$ is the weight vector associated with the i^{th} pattern class.

(These are linear decision functions. Other types of decision functions involve $d_i(\vec{X})$ which are of polynomial form. The number of terms needed to describe such functions increases rapidly with the degree of the poynomial but the discussion here is restricted to the linear form. The linear form can be expanded to deal with the more complicated cases.)

Class membership is determined by application of the decision functions. If the classes are $c_1,\ldots,c_M$ an object belongs to c_i if

$$d_i(\vec{X}) > d_j(\vec{X}) \quad \text{for all } i<>j \text{ [9]}$$

The selection of a weight vector is made so that the definition of class membership above holds. The process of assigning weight vectors and thus establishing decision functions involves initializing the weight vectors at something, say (0,0,...,0). Adjustments are made to the vectors until the weight vectors are such that the rule for the decision functions is satisfied. For example, suppose the procedure for establishing weight vectors is in the k^{th} step of adjustment in determining a pattern $\vec{x}(k)$ as belonging to pattern class i. If

$$d_i(\vec{x}(k)) <= d_j(\vec{x}(k)) \quad \text{for some } j, i<>j$$

then the weight vectors must be adjusted. To do this the following rule is used [23].

$$\vec{w}_i(k+1) = \vec{w}_i(k) + c\vec{x}(k)$$
$$\vec{w}_j(k+1) = \vec{w}_j(k) + c\vec{x}(k)$$
$$\vec{w}_l(k+1) = \vec{w}_l(k)$$

for all $l<>i$, $l<>j$, $l= 1,2,\ldots,M$
and $d_i(\vec{x}(k)) > d_l(\vec{x}(k))$
where k+1 indicates one more step.

What all this means is that if the decision function does not satisfy the rule for some i with respect to every j, j<>i, those weight vectors where the rule is not satisfied are adjusted. For any j, j<>i, where the rule is satisfied, the weight vectors are left unaltered. The process is completed when for every j, j=1,2,...,M and j<>i,

$$d_i(\vec{x}(k)) > d_j(\vec{x}(k))$$

When classification using decision functions is implemented the main problem is in determining the coefficients for those functions. When samples from each class of pattern being considered are available the process of recognition is said to be supervised learning or training. This process uses a classified set of training classes or patterns and a learning procedure like the weight vector adjustment where the $\vec{x}$ being considered is a training pattern. In this way the coefficients for the decision functions can be calculated. A training sample matrix for K classes is shown in table 4-1. It is possible however that the samples available have unknown classifications. The process of recognition in this case is said to be unsupervised. The system now has the problem of learning what classes are present so all objects may be classified. This involves the idea of clustering together those things which have similar pattern vectors. With the clustering, mathematical calculations are required to establish coefficients for the decision functions which will classify those items with similar pattern vectors in the same class. Fuzzy set theory has been proposed as a possible method to be used in cases such as this. The occurrence of unsupervised classification in industrial use of this technique is rare. The polyhedral world is specific enough that the classes involved are able to be identified.

The decision-theoretic approach to recognition has its applications and its drawbacks. It is used industrially in areas of medicine, manufacturing, and

TRAINING SAMPLE [11]

class c_1 :	$x_1, x_2, \ldots, x_{L1}$
class c_2 :	$x_{L1+1}, x_{L1+2}, \ldots, x_{L1+L2}$
:	
:	
class c_k :	$x_{L(k-1)+1}, x_{L(k-1)+2}, \ldots, x_{L(k-1)+Lk}$

Table 4-1: Training Table for K Supervised Classes

archaeology. In industrial applications the method of matching the feature vector x with a feature vector of an object under consideration is used. The approach is mathematically sound but problems arise when the features being considered are not independent enough. It may become difficult to make an accurate classification. This is where the structural or syntactic approach can be applied.

4.4 The Syntactic or Structural Approach

The previous approach lacks structural considerations in a formal sense. It is dependent on the knowledge of what features are present or absent as opposed to any relationships between the features. To handle structural and relational characteristics in the scene being analyzed the syntactic approach is used. This seems much more natural in a polyhedral world. A diagram of the syntactic recognition system can be seen in figure 4-4. When using this approach a scene is generally represented by string, tree, or graph structures made up of its primitives. Primitives in scene analysis could be faces, edges, or corners. Because the basic concept behind the syntactic approach is construction of a scene from sub-objects, the process is analogous to parsing a language. Therefore much about appropriate grammar analysis can be used.

The composition of a scene can be represented in a natural way which is much like that in a language produced by a formal grammar. The syntax of an expression in a language like Pascal can be illustrated with a tree as shown in figure 4-5. The leaves of the tree are terminals in the language. In much the same form, a scene can be represented as shown in figure 4-6. In the case of the scene the leaves represent the

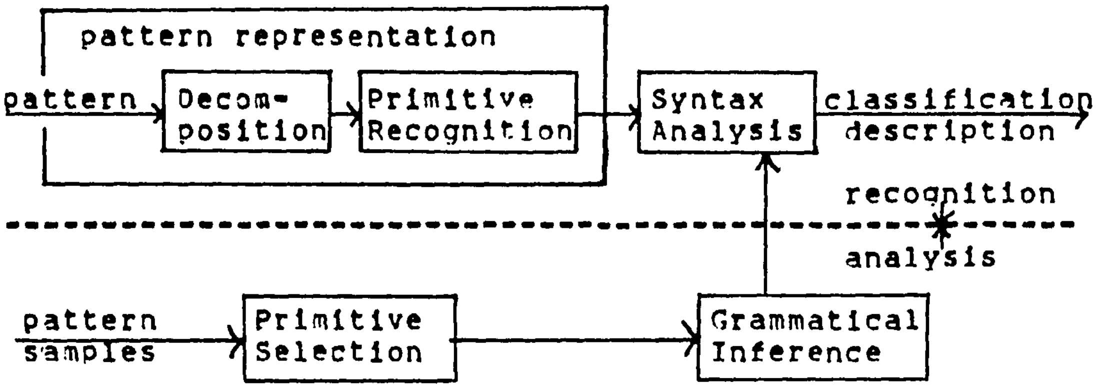


Figure 4-4: Syntactic Recognition System [7]

primitives from which the scene is built. The tree may be interpreted either top down or bottom up. Top down processing involves starting with the biggest unit (the scene) and splitting it into its elementary parts. A bottom up interpretation involves starting with the primitives (side1,side2,side3) and thinking of them as being part of, and thus replaced by, a larger unit (a triangle). In this way it is possible to develop a set of productions. With these productions, a form of parsing can be used to identify objects much like recognition of a sentence in a language.

The first question which needs to be considered is exactly what terminals(primitives) and non-terminals(subobjects) need to be included in the recognition grammar. Fu [7] says the choice of

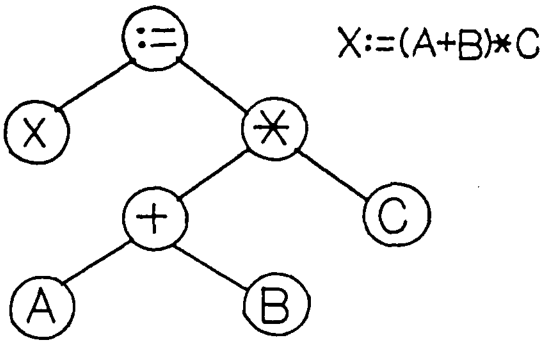


Figure 4-5: Pascal Tree

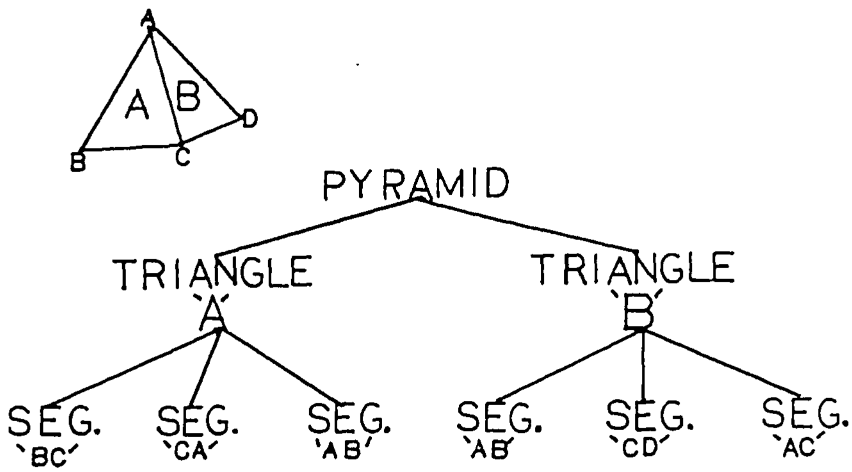


Figure 4-6: Tree Representation of a Simple Scene

primitives should satisfy the following requirements:

1. The primitives should be small basic pattern elements which provide an adequate description of the data in terms of the specified structural relations (e.g., the concatenation relation)

2. The primitives should be easily extracted or identified by existing nonsyntactic methods, since they are considered to be simple patterns and their structural information is not important.

As an example, for recognizing rectangles the following primitives might be set up.

[8]

In terms of these primitives a rectangle could be said to be any object of the form $a^m b^n c^m d^n$ where m and n are integers. It would be possible to reduce the number of primitives for the rectangle by defining -a as <---- . A rectangle could then take the form $a^m b^n (-a)^m (-b)^n$. A triangle could be represented using the above primitives as $e^n f^n g^n$. In dealing with projection of polyhedral scenes these unit vectors in various directions suffice as primitives since polygons are all concatenations of such things. However since these polygons are positioned at any of an infinite number of angles finding all such primitives for this kind of method is not quite as easy as it may first seem and a different kind of primitive may be found more useful.

Once the primitives are decided upon they are used to identify sub-objects. These sub-objects then become pieces which combine to create the objects in the scene. They are used to construct a relational tree as illustrated in figure 4-7. Again decisions must be made here about what relational properties and sub-objects best identify an object. In the polyhedral world polygonal regions and their attachment to each other should be considered. In addition various measurements which can be arrived at may be employed. To make the best use of this the syntactic approach alone is not enough.

An example of a system used involving a strictly syntactic approach with objects as strings of primitives is the Picture Description Language (PDL) of A.C. Shaw [8, 6] . Shaw used directed line segments as primitives and labelled each one with a head and a tail. Primitives could be joined under four concatenation operations +,*,-,x by the rules in figure 4-8. The remaining two operators were ~ and /. ~ is a unary operator indicating the head-tail reversal. / is used along with labelling to indicate multiple occurrences of a primitive. An example of a string representation of a given structure can be seen in figure 4-9. The correlation between formal grammar work and syntactic representations of objects is evident here. PDL can be used to generate a tree representation of a picture using the sub-structures and relational operators such as "left of".

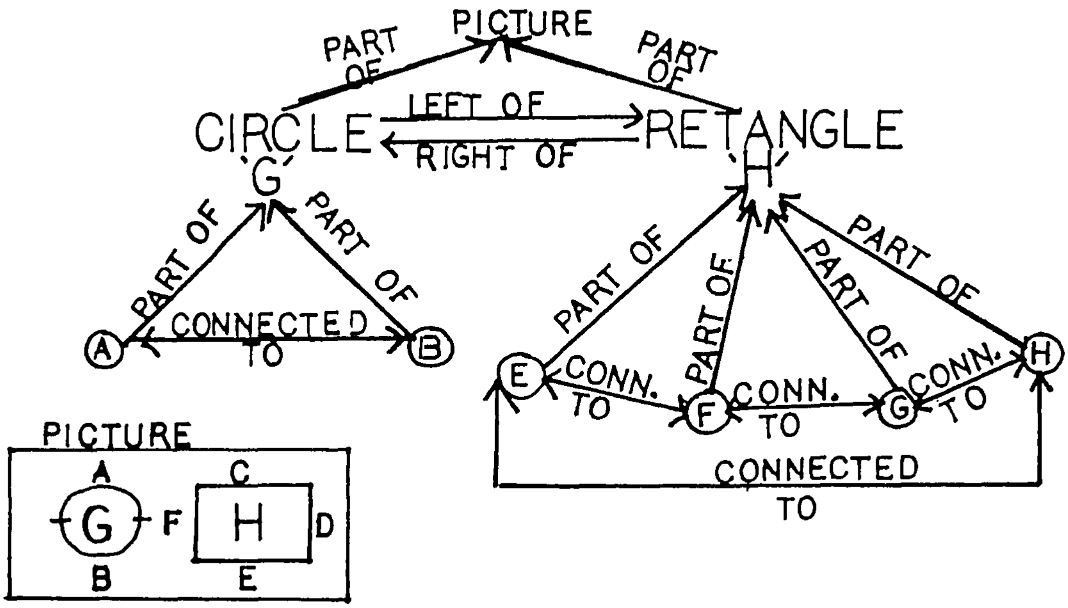


Figure 4-7: Relational Tree

This language has been used in processing pictures involved in the study of particles in physics.

The syntactic approach can be viewed as an extension of language parsing. A string of primitives can be recognized by parsing top down or bottom up. This system is alright if all that is to be considered is the construction of the object from its parts. It seems much more appropriate to extend this to include the known

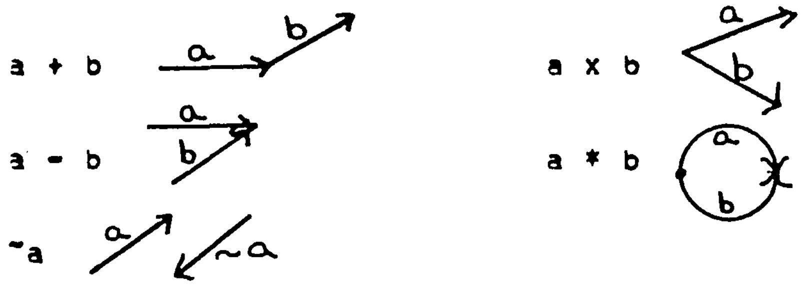


Figure 4-8: PDL Operations [8]

$$(((a * ((d^{i} + a) + (\sim d^{j}))) * ((((/ d^{i})$$
$$+ b) + ((a * ((\sim d) + (a^{k} + d))) + (\sim b))) +$$
$$(\sim(/d^{j})))) * ((b + (/a^{k})) + (\sim b)))$$

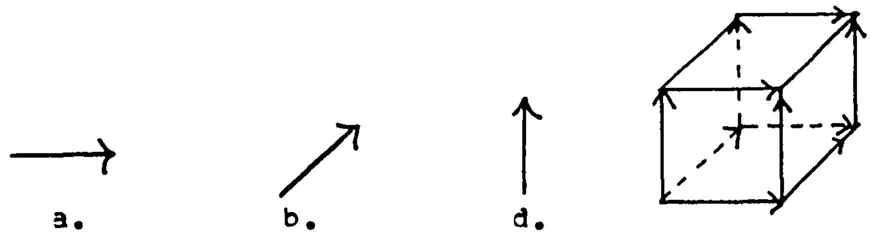


Figure 4-9: Strings in PDL [6]

attributes in certain pattern classes. The more information in use the more reliable the parsing will be. An important approach which applies these ideas is that of attributed grammars.

4.5 Attributed Grammars

The method of attributed grammars [24, 2] is an extension of the syntactic methods discussed above. These grammars include two basic elements. One of these elements is a token from the scene grammar such as an line segment or a triangle. Some of these tokens will be terminals or primitives while others will be non-terminals or sub-objects. The second element in the grammars is a list of attributes associated with each token. These attributes may include things like a measurement such as length or a relationship such as right-of. These two components can then be used along with a graph structure to better represent the objects in a scene.

It is important to give some basic definitions here. An attributed grammar consists of a 4-tuple $G=(V_n,V_t,P,S)$ [24]. V_n is the set of non-terminals. V_t is the set of terminals. S is an element of V_n and the start symbol. P is the set of productions. In the polyhedral considerations "side" might be a member of V_t, "TRIANGLE" might be a member of V_n, "scene" could be the start symbol, and a production might look something like this

TRIANGLE ----> (side)(side)(side).

What makes an attributed grammar different from the usual formal grammar is that associated with each instance of an element of V_n and V_t is a set of attributes. The set may be infinite or finite. There are two kinds of attributes those which are inherited, and those which are synthesized. Inherited attributes are those a sub-object obtains simply because one of its parts was characterized by it. For example a cube has a height of 2 inches because one of its sides is 2 inches. Synthesized attributes are those a sub-object obtains by calculating some function of the attributes of its parts. A triangle has a perimeter of 12 inches because its sides have lengths of 3 inches, 4 inches, and 5 inches. Because of these attributes productions for this grammar consist of two parts: a semantic part and a syntactic part. The syntactic part of the production is what could be considered the usual form of a production: for example,

TRIANGLE ----> (side)(side)(side).

The semantic rule of the production involves using the attributes discussed before. The attribute part of a production might look like:

TRIANGLE <---- (side2.length)

where side2.length is the length of the side on the base of the triangle.

The relational properties such as the distance between two parts can be accounted for by using attributed graphs. An attributed graph is defined as a 5-tuple $g = (N, E, \lambda, \alpha, \beta)$ [2]. N are nodes made of elements from V_n and V_t above. E is the set of edges representing the connections between the nodes. λ is the node labelling function. α is a function which associates a set of node attributes with each node. These are the sets and attributes described above. β is a set of functions associating attributes with edges. An example of a scene and its associated graph is shown in figure 4-10 and 4-11. The types of nodes contained in the example are {WEDGE,COLUMN,BOARD}. The edges of the graph consist of the relations {RIGHT-OF,BELOW}. The only node attribute is height so $\alpha(n)$ = {HEIGHT} for each node n. A given node may have other attributes such as color or width, but they are not considered necessary for recognition in this example and therefore not used. Distance is the only edge attribute so $\beta(e)$= {DISTANCE} for each edge e. The graph can be used with bottom up parsing where under the guidance of the productions pieces of the subgraph can be grouped and replaced by bigger units.

A diagram for recognition involving attributed

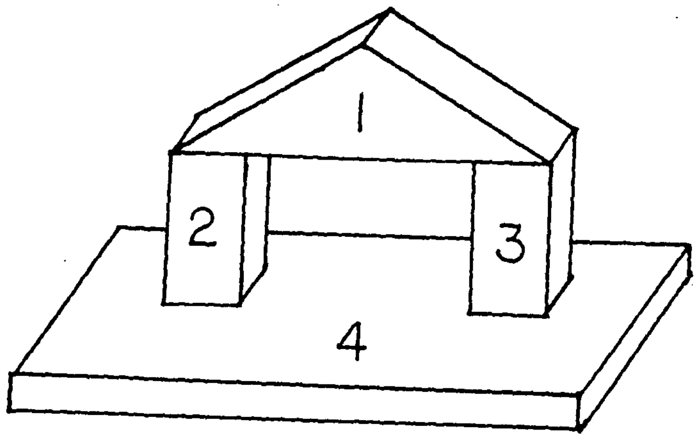
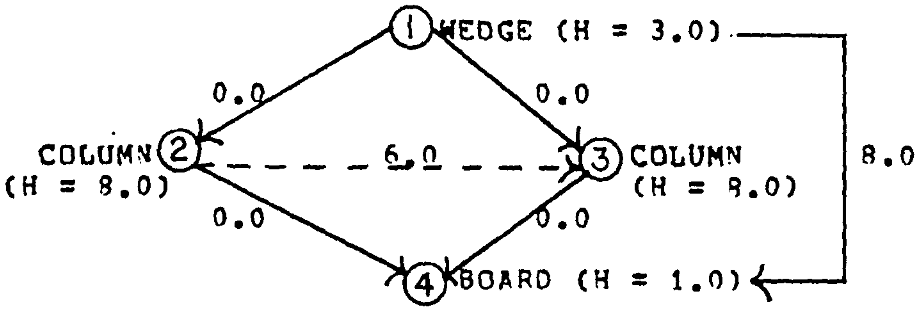


Figure 4-10: Scene [2]



solid lines=> BELOW ;   dotted lines=> RIGHT-OF

Figure 4-11: Attributed Grammar Graph [2]

grammars is shown in figure 4-12. Primitives and attributes are extracted from the picture. The next step is to take the picture and represent it as a tree or graph by establishing productions involving the primitives and sub-objects. The graph may be constructed and analyzed syntactically to build the sub-patterns or sub-objects. While these sub-objects are being constructed the semantic part of the production is used to obtain attributes of the sub-objects. The relational attributes for the sub-objects can be obtained from the attribute graph which can now be constructed from the sub-objects found and references back to the original input data or descriptions. The process can be completed by considering as part of the productions not only functions on the node attributes but also functions on the edge attributes contained in the attributed graph. The final step comes in deciding if there is a match between a known pattern and that extracted from the picture.

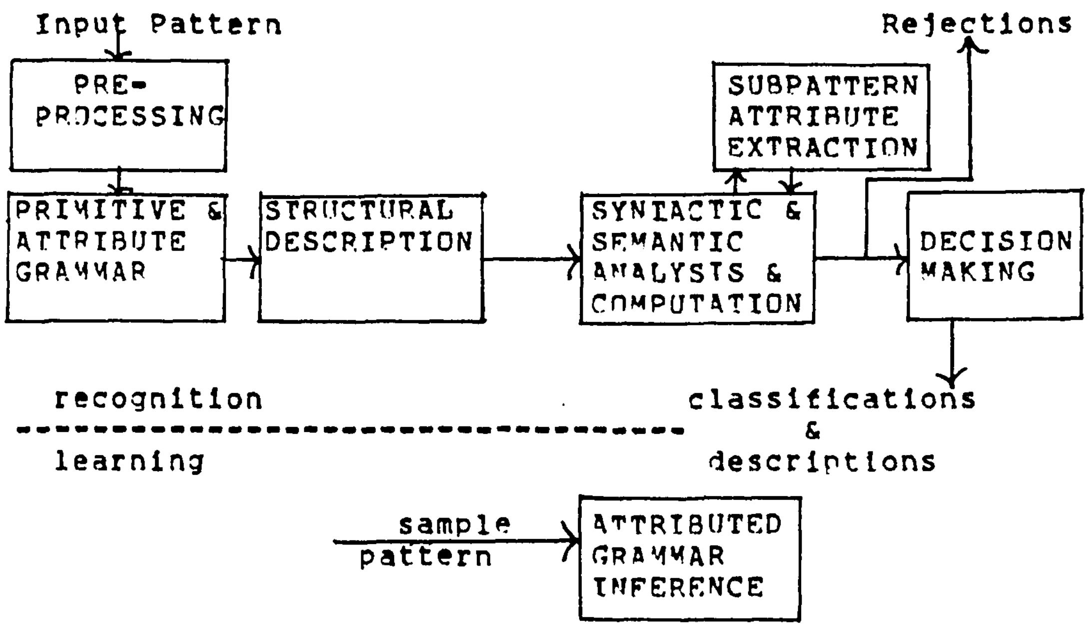


Figure 4-12: Using Attributed Grammars [24]

4.6 RECOGNIZE: An Attributed Grammar

It is possible to construct an attributed grammar for recognition of objects in a polyhedral scene. The author has developed an example of an attributed grammar found in the following pages. The grammar can be used to recognize six kinds of polyhedra from a general viewing angle: cube, rectangular solid, hexagonal solid, trapezoidal solid, pyramid, and wedge. It is assumed that faces are rectangles, triangles, hexagons or trapezoids. In conjunction with this grammar the author wrote a program called RECOGNIZE which uses the productions in the grammar to make identifications.

If this method of recognition is to be used the partitioning part of the analysis must pass forward a specific set of information. The information passed forward will contain details as to which regions belong together in one object. If a visible surface is partially blocked, those lines which are not completely visible will be completed before passing the information. In figure 4-13, object A is not uniquely determined by its visible parts. This problem is handled in the partitioning process with the use of some accepted rule for completing lines and is not the concern of RECOGNIZE. Both Clowes and Sugihara discussed a process for doing this. The partitioning process must pass the length and slope of each line of each region. Information passed to the recognition section also includes the number of regions visible and the number of sides bounding each region. For the program RECOGNIZE this information will be contained in a record, one record per object.

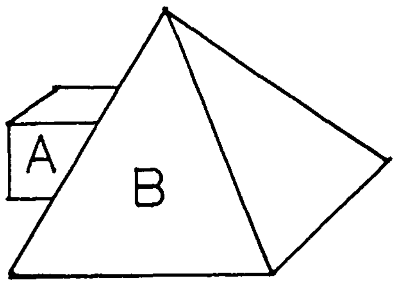


Figure 4-13: Blocked Objects

The attributed grammar being considered is based on a set of assumptions that the viewing point is in general position, and that all objects are on a flat surface. As a result of this, views of polyhedra accepted by RECOGNIZE are illustrated in figure 4-14. Only those attributes needed for recognition are considered in the grammar. The productions themselves are based on the possible combinations of visible surfaces necessary for recognition of an object. Along these lines, a pyramid is restricted to a triangular, hexagonal, or rectangular base. The example attributed grammar used has the following elements:

V_N (non-terminals) =

{SCENE,SQUARE,RECTANGLE,TRIANGLE,
TRAPEZOID,HEXAGON,
CUBE, REC_SOLID,HEX_SOLID,TRAP_SOLID,
PYRAMID,WEDGE}

V_T (terminal) =

{SIDE}

S (start symbol) = {SCENE}

Attributes of V_T and V_N :

A(SQUARE) = {HEIGHT}
A(RECTANGLE) = {HEIGHT,LENGTH}
A(TRIANGLE) = {LHEIGHT,RHEIGHT,BASE}
A(TRAPEZOID) = {LSLANT,RSLANT,TOP,BOTTOM}
A(HEXAGON) = {WIDTH}
A(CUBE) = {HEIGHT}
A(REC_SOLID) = {HEIGHT,WIDTH,LENGTH}
A(HEX_SOLID) = {HEIGHT}
A(TRAP_SOLID) = {SLANTHEIGHT}
A(PYRAMID) = {SLANTHEIGHT}
A(WEDGE) = {HEIGHT,WIDTH,LENGTH}

In the syntactic productions the notation has the following interpretation: () around a terminal or non-terminal means the item occurs exactly once, [] means the item occurs once or not at all, and {} means the item occurs any number of times including zero. The / means that a production may result in several different alternatives. These productions are based on the acceptable figures as shown in figure 4-14.

```
Production : Syntactic ==>
   SCENE --> {CUBE}{REC_SOLID}{HEX_SOLID}
             {TRAP_SOLID}{PYRAMID}{WEDGE}
   CUBE --> (SQUARE)(SQUARE)(SQUARE)
   REC_SOLID --> (RECTANGLE)(RECTANGLE)
                 (RECTANGLE)
   HEX_SOLID --> (RECTANGLE)(RECTANGLE)
                 (RECTANGLE)(HEXAGON)
   TRAP_SOLID --> (TRAPEZOID)(TRAPEZOID)
                  (RECTANGLE)/(TRAPEZOID)
                  (TRAPEZOID)(TRAPEZOID)
                  [TRAPEZOID](RECTANGLE)/
                  (RECTANGLE)
   PYRAMID --> (TRIANGLE)(TRIANGLE)[TRIANGLE]/
               (RECTANGLE)/(HEXAGON)
   WEDGE --> [RECTANGLE](TRIANGLE)(RECTANGLE)/
               (RECTANGLE)(TRIANGLE)
   SQUARE --> (SIDE)(SIDE)(SIDE)(SIDE)
   RECTANGLE --> (SIDE)(SIDE)(SIDE)(SIDE)
   HEXAGON --> (SIDE)(SIDE)(SIDE)(SIDE)(SIDE)
               (SIDE)
   TRAPEZOID --> (SIDE)(SIDE)(SIDE)(SIDE)
   TRIANGLE --> (SIDE)(SIDE)(SIDE)
```

In the semantic productions it should be noted both faces and edges are numbered in order starting at the far left and going around counter-clockwise. If the left-most face or vertex is not obvious because of the slant of a line, then the upper left-most part is used to begin numbering. The productions are straight forward except there is a problem if in the hex_solid the hexagon is face 1. In this instance the height will be considered to be undefined. In further development trigonometry could be used to calculate such a height but that is not a point of consideration at this time. It should be noted that though a pyramid or hex-solid may be seen from the base alone, RECOGNIZE does not handle these cases.

```
Production: Semantic ==>
     CUBE.HEIGHT <-- SIDE1.LENGTH
     REC_SOLID.HEIGHT <-- FACE1.LENGTH
     REC_SOLID.WIDTH <-- FACE1.WIDTH
     REC_SOLID.LENGTH <-- FACE2.WIDTH
     TRAP_SOLID.LSLANT <-- FACE1.LSLANT
     TRAP_SOLID.WIDTH <-- FACE1.BOTTOM
     HEX_SOLID.HEIGHT <-- FACE1.HEIGHT
     PYRAMID.SLANTHEIGHT <-- FACE1.LHEIGHT
     WEDGE.HEIGHT <-- FACE(TRIANGLE).LHEIGHT
     WEDGE.WIDTH <-- FACE(RECTANGLE).WIDTH
     WEDGE.LENGTH <-- FACE(TRIANGLE).BASE
     SQUARE.HEIGHT <-- SIDE1.HEIGHT
     RECTANGLE.WIDTH <-- SIDE2.LENGTH
     RECTANGLE.LENGTH <-- SIDE1.LENGTH
     TRAPEZOID.LSLANT <-- SIDE1.LENGTH
     TRAPEZOID.RSLANT <-- SIDE3.LENGTH
     TRAPEZOID.TOP <-- SIDE4.LENGTH
     TRAPEZOID.BOTTOM <-- SIDE2.LENGTH
     HEXAGON.WIDTH <-- SIDE1.LENGTH
     TRIANGLE.BASE <-- SIDE2.LENGTH
     TRIANGLE.LHEIGHT <-- SIDE1.LENGTH
     TRIANGLE.RHEIGHT <-- SIDE3.LENGTH
```

RECOGNIZE can be found on file in the Division of Computing and Information Science at Lehigh University. It works with one object at a time. The information passed from the pre-processing contains a record for each object. The record gives the number of visible regions in the object and details about each region and its parts. It uses both the syntactic productions and definitions of specific regions to establish what those regions are. The attributes that are associated with the regions are formulated by way of the semantic

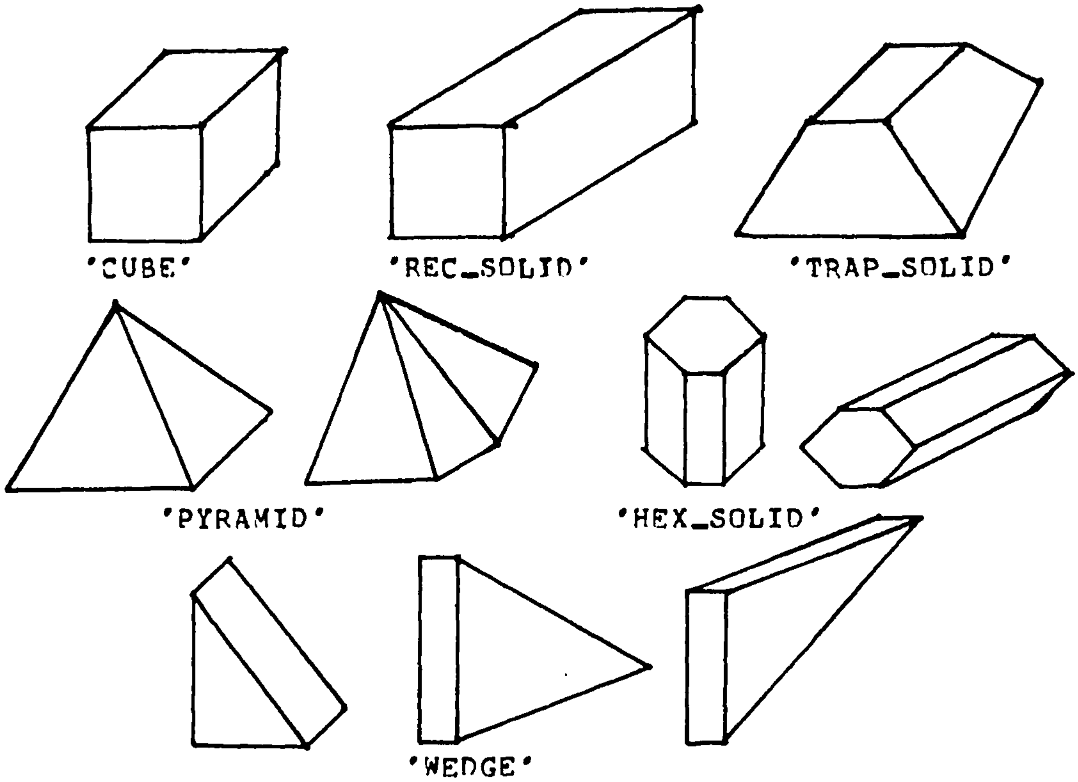


Figure 4-14: Accepted Views of Polyhedra

productions. Once the regions have been identified, the same process is used to define an object from the regions it contains. A bottom up parsing method is used in the sense that any time the right side of a production is found it is replaced by the left side. RECOGNIZE was run on a sample scene shown in figure 4-15. The input handed to RECOGNIZE for the scene is given in table 4-2 and is the kind of information that would be passed from the partitioning section of a recognition system. The first number is the number of regions a body contains. Each line then gives the number of sides for each region and the length and slope of each side. The identifications made from the input are contained in table 4-3.

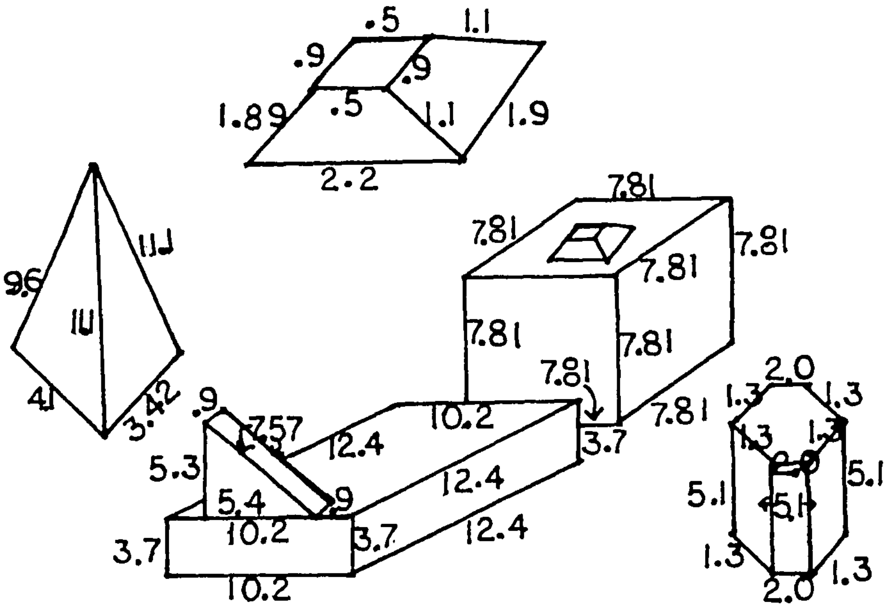


Figure 4-15: Scene Used for RECOGNIZE

This is an important current method of recognition in pattern analysis. It combines the qualities from both a syntactic and quantitative approach. Extension of this method can lead to more complete and thorough recognition with regard both to the polyhedral world and to more complex environments.

```
2 3 9.6 1.5 4.1 -0.5 11.1 -3.0 3 11.1 -3.0
      3.42 1.0 11.1 -0.33
3 4 3.7 99.99 10.2 0.0 3.7 99.99 10.2 0.0
  4 3.7 99.99 12.4 0.8 3.7 99.99 12.4 0.8
  4 12.4 0.8 10.2 0.0 12.4 0.8 10.2 0.0
2 3 5.3 99.99 5.4 0.0 7.54 -0.97
  4 7.54 -0.97 0.9 1.2 7.54 -0.97 0.9 1.2
4 4 5.1 99.99 1.3 1.2 5.1 99.99 1.3 -0.83
  4 5.1 99.99 2.0 0.0 5.1 99.99 2.0 0.0
  4 5.1 99.99 1.3 1.2 5.1 99.99 1.3 -0.83
  6 1.3 -0.83 2.0 0.0 1.3 1.2 1.3 -0.83 2.0
            0.0 1.3 1.2
3 4 7.81 99.99 7.81 0.0 7.81 99.99 7.81 0.0
  4 7.81 99.99 7.81 1.4 7.81 99.99 7.81 1.4
  4 7.81 1.4 7.81 0.0 7.81 1.4 7.81 0.0
3 4 1.89 1.0 2.2 0.0 1.1 -4.9 0.5 0.0
  4 1.1 -4.9 1.9 2.1 1.1 -0.8 0.9 2.1
  4 0.9 2.1 0.5 0.0 0.9 2.1 0.5 0.0
```

Table 4-2: Input Data from Partitioning

OBJT#	OBJECT NAME	HEIGHT	WIDTH	LENGTH	SLANTHEIGHT
1	PYRAMID				9.60
2	REC_SOLID	3.70	10.20	12.40	
3	WEDGE	5.30	0.90	5.40	
4	HEX_SOLID	5.10			
5	CUBE	7.81			
6	TRAP_SOLID		2.20		1.89

Table 4-3: Results of RECOGNIZE on figure 4-14

4.7 Conclusion: Recognition

Approaches to recognition have taken various forms. The major three have been the masking method, the decision-theoretic method, and the syntactic method. Numerous variations of each of these have been researched and developed. The problem is that each has its strong points and its weak points. The decision-theoretic approach involves the use of vectors and its decisions are based on statistical closeness. Because of this it can handle noisy patterns. It cannot however use, to any great extent, the information on relational structure. The syntactic approach on the other hand is characterized by the opposite situation. The one method which seems to make the strongest attempt to combine the best qualities of each is the method involving attributed grammars. Through these grammars both the syntactic and semantic characteristics of scenes are considered and analyzed for recognition.

5. Conclusion

The steps involved in recognition of objects in three-dimensional scenes from two-dimensional line drawings involve two primary objectives. The first is the partitioning of the drawing into sets of primitives and regions which belong to the same object. This requires using the input data arrived at by some pre-processing of the image. There have veen a variety of approaches to this both qualitatively and quantitatively. Recently the use of linear algebra has played a contributing role. The second objective is the actual recognition of the objects themselves. The recognition can be done by the masking approach, the decision-theoretic approach or the syntactic approach. The last two are the more widely used. Each has its advantages and disadvantages. Wallace [25] claims that a major disadvantage of the syntactic method is the lack of a method to create machine inferred grammars. Presently grammars must be man-made. An approach which attempts to combine the advantages of each is that of attributed grammars.

The approaches to scene analysis discussed here have been with respect to a polyhedral world. It is a well defined domain in which to work and establish concepts. These concepts can then be extended and applied to line drawings of other domains. These other domains involve objects which are curved and not so well defined as those in the polyhedral world. With the enlargement of the domain comes an enlargement of approaches. These approaches become more highly structural and mathematical in order to deal with the wider domain involved in matching and analyzation. Shapiro and Haralick [20] discuss the use of relational homomorphisms to determine class matching. Davis and Henderson [4] introduce what they call a stratified grammar to deal with syntactic analysis. Computer vision and graphics are currently areas supporting much research. The automation in all areas of production and business lead to a strong desire to develop accurate and efficient techniques to handle such things. The basics discussed in this paper are an important backround from which current research may grow.

References

[1] Brown, C.M. and R.J. Popplestone.
Cases in Scene Analysis.
In Bruce G. Batchelor (editor), *Pattern Recognition Ideas in Practice*, pages 205-227.
Plenum Press, New York, 1978.

[2] Bunke, Horst.
Attributed Programmed Graph Grammars and Their Application to Schematic Diagram Interpretation.
IEEE Transactions on Pattern Analysis and Machine Intelligence 4:574-582, 1982.

[3] Clowes, M.B.
On Seeing Things.
Artificial Intelligence 2:79-116, 1971.

[4] Davis , Larry S. and Thomas C. Henderson.
Hierarchical Constraint Processes for Shape Analysis.
IEEE Transactions on Pattern Analysis and Machine Intelligence 3:265-277, 1981.

[5] Draper, Stephen E.
The Use of Gradient and Dual Space in Line-Drawing Interpretation.
Artificial Intelligence 17:461-508, 1981.

[6] Fu, K.S.
Syntactic Methods of Pattern Recognition.
Academic Press, New York, 1974.

[7] Fu, King-Sun.
Pattern Recognition for Automatic Visual Inspection.
Computer 15:34-39, 1982.

[8] Gonzalez, Rafael and Michael Thomason.
Syntactic Pattern Recognition: An Introduction.
Addison-Wesley, Reading, MA, 1978.

[9] Gonzalez, Rafael and Reza Safabakhsh.
Computer Vision Techniques for Industrial Applications and Robot Control.
Computer 15:17-28, 1982.

[10] Guzman, Adolfo.
Computer Recognition of Three-Dimensional Objects in a Visual Scene.
PhD thesis, MIT, 1968.

[11] Hall, Ernest.
Computer Image Processing and Recognition.
Academic Press, New York, 1979.

[12] Huffman, David.
A Duality Concept for the Analysis of Polyhedral Scenes.
In E.W. Elock and D. Michie (editors), Machine Intelligence 8, pages 475-492. Ellis Horwood, Chichester, 1977.

[13] Huffman, David.
Impossible Objects as Nonsense Sentences.
In B. Meltzer and D. Michie (editors), Machine Intelligence 6, pages 295-324. Edinburg University Press, Edinburg, 1971.

[14] Huffman, David.
Realizable Configurations of Lines in Pictures of Polyhedra.
In E.W. Elock and D. Michie (editors), Machine Intelligence 8, pages 493-509. Ellis Horwood, Chichester, 1977.

[15] Kanade, Takeo.
A Theory of Origami World.
Artificial Intelligence 13:279-311, 1980.

[16] Kanade, Takeo.
Recovery of the Three-Dimensional Shape of an Object from a Single View.
Artificial Intelligence 17:409-460, 1981.

[17] Mackworth, A.K.
How to See a Simple World: an Exegesis of Computer Programs for Scene Analysis.
In E.W. Elcock and D. Michie (editors), Machine Intelligence 8, pages 510-540. Ellis Horwood, Chichester, 1977.

[18] Mackworth, A.K.
Interpreting Pictures of Polyhedral Scenes.
Artificial Intelligence 4:121-137, 1973.

[19] Newman, E.A.
Scene Analysis: Some Basics.
In Bruce G. Batchelor (editor), *Pattern Recognition Ideas in Practice*, pages 429-462. Plenum Press, New York, 1978.

[20] Shapiro, Linda G. and Robert M. Haralick.
Structural Descriptions and Inexact Matching.
IEEE Transactions on Pattern Analysis and Machine Intelligence 3:504-519, 1981.

[21] Shirai, Yoshiaki.
Image Processing for Data Capture.
Computer 15:21-39, 1982.

[22] Sugihara, Kukichi.
Mathematical Structures of Line Drawings of Polyhedrons- Toward Man-Machine Communication by Means of Line Drawings.
IEEE Transactions on Pattern Analysis and Machine Intelligence 4:458-468, 1982.

[23] Tou, J.T. and R.C. Gonzalez.
Pattern Recognition Principles.
Addison-Wesley, Reading, MA, 1974.

[24] Tsai, Wen-Hsiang and King-Sun Fu.
Attributed Grammar-A Tool for Combining Syntactic and Statistical Approaches to Pattern Recognition.
IEEE Transaction On Systems, Man and Cybernetics 10:873-884, 1980.

[25] Wallace, Timothy P. and Owen R. Mitchell and Keinosuke Fukunaga.
Three Dimensional Shape Analysis Using Local Shape Descriptors.
IEEE Transactions on Pattern Analysis and Machine Intelligence 3:310-322, 1981.

[26] Waltz, David.
Artificial Intelligence.
Scientific American 4:118-133, 1982.

Vita

Linda was born June 20, 1954 in Fort Meade, Maryland. Her parents are William A. and Eleanor B. Ludwigs. She attended Western Illinois University from September 1972 through May 1976. In May of 1976 she received her B.S. in Mathematics graduating with high honors. While at Western, Linda was a member of Sigma Zeta, Kappa Mu Epsilon, and Phi Kappa Phi honoraries. From August 1976 through May 1981, Linda taught high school Mathematics at Seeger Memorial High School in West Lebanon, Indiana. In the summer of 1978 she attended graduate school part time at Purdue University. In August 1981 she began work on her Master of Science degree in Computer and Information Science at Lehigh University.