STRUCTURED DESCRIPTIONS

VISION FLASH 48

by

R. P. Gabriel

August 1973

Massachusetts Institute of Technology

Artificial Intelligence Laboratory

Abstract

A descriptive formalism along with a philosophy for its use and expansion are presented wherein descriptions are of a highly structured nature. This descriptive system and the method of recognition are extended to the rudiments of a general system of machine vision.

Work reported herein was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under Contract Number N00014-70-A-0362-0005.

Introduction

The following report introduces the reader to a category of visual descriptions that, by virtue of their simplicity and elegance, may represent a major step in vision research. First attempts at descriptive formalisms almost unerringly produce systems that bog down in their own generality and apparent power; restrictions to classes of objects are viewed as a compromise of reality in its evasive detail. Unfortunately this leads to the eventual mapping out of important structural information from the aspiring formalism and, hence, a need for other powerful proceedures for recovering this loss.

This presentation is of a simple, specific structural descriptive formalism which will hopefully convince the reader of the existence of other descriptions of the same flavor and disposition. This particualr type of description has been considered by others, notably J.M. Hollerbach and G.J. Agin (Stanford); the fact that this work proceeded indendently of the above can be offered as evidence for the ripeness and importance of these ideas.

This Vision Flash reproduces a portion of my term paper for 18.436J-6.544J (Minsky, Papert). The reader will notice that roughly the first 20 paragraphs plus all three appendices are missing-since these paragraphs were of an expository-introductory nature (intended for those totally unfamiliar with vision research). The appendices, on the other hand, were of a more technical nature and were judged auxilliary to the main point of the paper.

In reading this report it is suggested that paragraphs 22-25 not be taken seriously since they deal with a, hopefully, minimally functional aspect of the visual system. Paragraphs 26-34 present a paradigm of a type of description that appears to be of considerable importance in vision research.

7. The major theme of this paper is that:

a), complex visual descriptions are built from a small number of primitive shapes, using a very small number of simple constructions

b), these primitive shapes are used as a kind of indexing system by which the high-level descriptions are accessed

c), when people view a scene they recognize these primitive shapes, and perhaps clues about the possible construction, thus conditionally recognizing the object via the indexing system

d), the primary process of visual recognition is description verification rather than description building followed by pattern matching (that is, the primitive shapes access a description via the indexing which results in verification or further accessing-perhaps using a kind of Winston semantic memory as well as the primary indexing)

e), typically people have sufficient reason to access a description prior to viewing, or soon thereafter, which allows them to rapidly verify the correspondence between and description. Thus a recognition is made wherein the feeling of "seeing everything" is actually a feeling of "having accessed the correct description"

f), people have a very good understanding of the 3-dimensional consequences of their descriptions and find it quite difficult to locate objects unless they can understand "what they look like"

g), this understanding is partly built into the structure of the construction and partly due to a sort of internal model of the object

h), therefore "seeing" is "recognizing".

10. The simplest form of visual perception is verification- simple both in terms of computational complexity and time requirements. Let us illustrate this point somewhat:

Suppose we have a husband coming home after his wife has retired for the night. Though he is interested in knowing whether she is indeed in bed, but fearing to turn on the light, he looks at the white pillow and sees a dark, undefined profile. Since this is where his wife usually is, and since he saw what he expected to see, he concludes that he has "seen that his wife is in bed". Although there are imaginatively many possible things that could have caused him to be deceived in this judgement, he stands firm in his conviction.

Here the verification proceedure is very simple: if feature f_1 is present, return T, otherwise NIL.

20. In general any object can be adequately represented as a line drawing where the lines are indications of certain features that we want to emphasize:

a), boundaries of maximun contrast form lines
b), color changes create lines
c), edge and roof effects form lines
d), shadows cause lines.

Lines are natural representations for objects in that they allow for a great deal of simplicity and translate into reasonable visual detection programs (line finders etc.)

21. It is usually assumed that people employ a sophisticated proceedure for recognizing objects since they can identify many different objects as examples of a single description. For example Figure 21.1 is identified as a box even though it has a protruding lid:

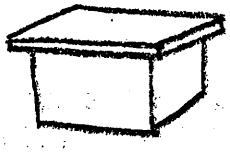


FIG. 21.1

We conjecture, and this is perhaps one of the main points of the paper, that people have a built in mechanism for approximation via certain pseudo-collinearity heuristics and the like. Also some characteristics of an object are ignored during the recognition phase.

Line drawings (and their close relatives to be discussed) can be an appropriate vehicle for this type of behaviour. Therefore we will now begin a presentation of a formalism for shape description

22. (Suggestion: do not get discouraged by the next few sections as they contain the low level shape primitives) First we have the definition of a line as follows:

$$L=(n\ (L_1)\ F_1, F_2\ F_3\ (L_2))$$

n is the line number, for use in other constructions

$L_1=(v_1, v_2)$ which are the initial and final vertices of the line L

F_1 is a feature indicating either a straight line or a curved line. Here we have several attractive possibilities:

F_1 can be an ordinary straight line which need only be realized in an actual visual scene as an approximation to a straight line (i.e. a wavy line). Denoted by 'S'

F_{11} can be an exact straight line. In other words L has to be straight to within a pre-determined tolerance. Denoted by 'St'.

F_1 can indicate a curved line in several ways. Though people have the general feeling that they have a great deal of sophistication with respect to curved lines, experience indicates that only a small number of curved line types are distinguishable. That is, even with circles only a few radii are outstanding; and almost all non-circles can be satisfactorally approximated by a sequence of circle arcs. Thus the following

formalism:

if F_1 is a circle we write
$F_1=(C,r,f_1)$ where C indicates a circle and r either singles out one of the pre-determined radii or is equal to the atom '?' which allows the line to be a circle of any radius. f_1="x" or "v" for convex or concave with the obvious line orientation.
if F_1 is a non-circle we write
$F_1=(N, L_1, (r_1,f_{r_1},f_1),\ldots,(r_n,f_{r_n},f_n))$ where

N indicates non-circle, L_1 gives the line orientation (Typically the line orientation is the order in which the vertices appear in the line definition; but we may want to change that here), r_1 is the radius indicator for the first circle in the sequence as above, f_{r_1} is the approximate fraction of arc-length for the first circle in the sequence (which can be equal to the atom '?'), f_1 is as above, and the remaining elements are the members of the sequence (n in all). Thus the third through the $n+3^{rd}$ list elements are the circles in the approximation.

F_2 is a feature indicating line equalities (within a tolerance)
$F_2=(E_i,R_1L_1,\ldots,R_nL_n)$
E_i indicates that all lines labelled E_i have the same distance between their vertices
E_i^p indicates that all lines labelled E_i^p have the same length along their paths
R_i is a relation $\angle,>$, or a rational number r_i; $\angle(>)$ indicates that the length of $L\angle(>)$ the length of L_i
r_i indicates that $r_i|L| = |L_i|$. (along their paths)

F_3 is a feature indicating parallel lines (within an approximation)
$F_3 = P_i$ or P_i^p

P_i indicates that all lines so labelled are s.t. the straight lines connecting their vertices are parallel.

P_i^p indicates that all lines so labelled are parallel along their paths.

Example:

L_2=a list of angle specifications. Each element of this list has the format:

$(L_i A_i)$ which indicates that the angle between the line L_i and L is A_i degrees , (within an approximation). Or A_i can be RA_i where R= and indicates that the angle is less than or greater than.

23. Now we can present the general definition of a line drawing description.

$D=((L^1)\ (L^2)\ (R_1))$

L^1 is the list of lines

L^2 is the list of sharp corner vertices (here we assume that we do not want sharp corners unless we specify them).

R_1 is the list of lines that constitute an opaque surface type.

$R_1=((L_1,\ldots,L_n)(S))$

where $L_1,\ldots,L_n$ are the lines defining the region and S is either "concave" or "convex". We feel that surface types need not be very complex since people are outstandingly poor at describing them. We hope that the final recognition routine will be able to allow this specification to work naturally.

R_1="implied" which means that the surface is curved the way that its defining lines indicate.

For example, suppose we have the three regions shown in Figure 23.1, and we attach them as indicated. Then we will get the profiles in Figure 23.2.

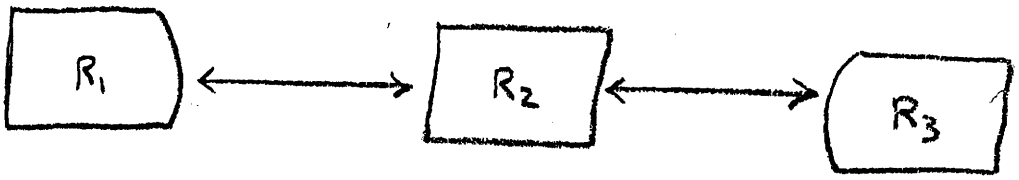


FIG. 23.1

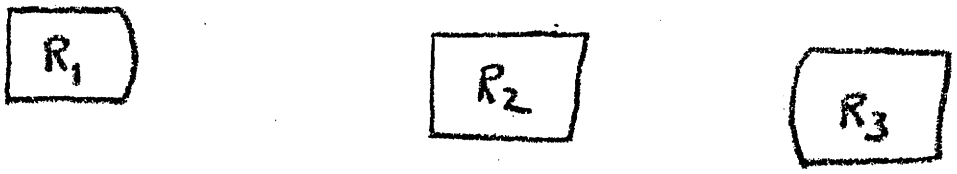


FIG. 23.2

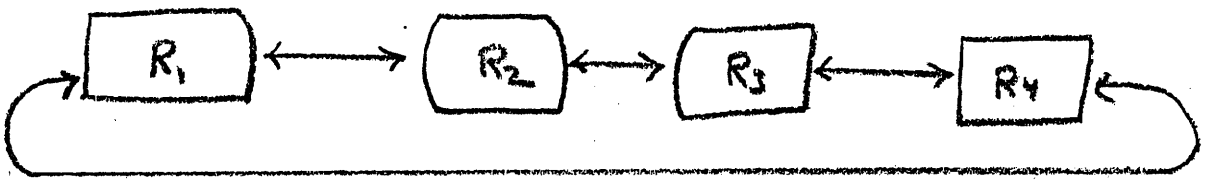


FIG. 23.3

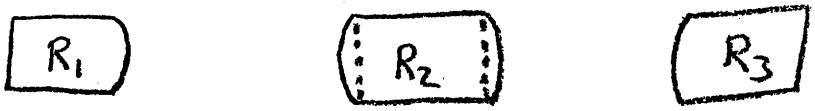


FIG 23.4

Thus R_2 is convex (i.e. comes up out of the page). If R_1, R_2, R_3, R_4 were as in Figure 23.3, the profiles would be given by Figure 23.4, where R_1 and R_3 are convex at one end and flat at the other.

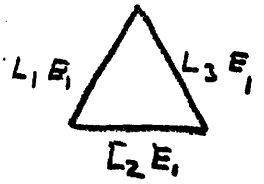
24. Once we have the possibilty of line descriptions we potentially have the power tp provide a representaion for any specific object we would want to identify. However, it would be inconvenient to be required to spell out in detail every line of every description even though the one we want is a simple combination of descriptions already available.

For this purpose we need a series of subroutines with the following capabilities:

1. Given D_1, D_2 attach them along given lines;
2. given D_1,D_2 attach them at given surfaces;
3. given D_1,D_2 attach a given line of D_1 to a given surface of D_2;
4. given D_1,D_2 attach them at given points; and
5. given D_1,D_2 attach a given point of D_1 to a given line or surface of D_2.

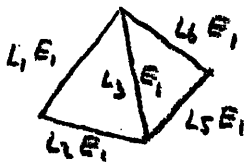
Towards these ends one of these routines has been conceived; namely, to accomplishll. : this routine is called ATTACH, and the details are given in Appendix II. Essentially what this subroutine does is to disjointify copies of the same decription, if necessary, and append them along the appropriate lines; the bulk of the computation is disjointification and the correct propagation of the features of the identified lines.

For example:

$D_1 =$ 

We want to attach another copy of D_1 to this D_1: say we want to identify line L_3 of the first copy with line L_1 of the second copy. First we disjointify:

Now we attach D_1 to D_1' by identifying L_3 with L_4. We get:



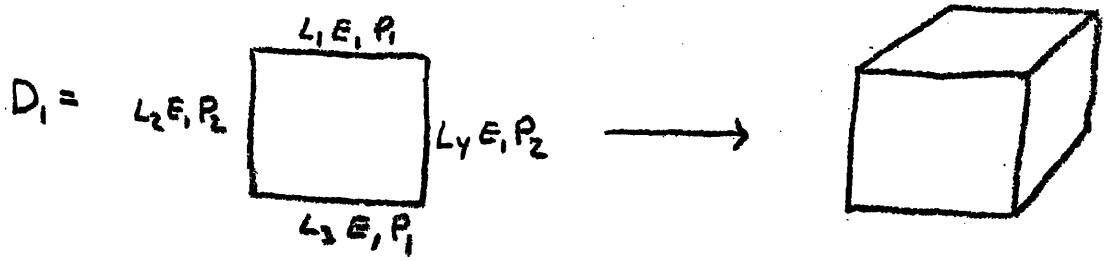
The notation used to accomplish this is:

(lambda(x,y)(ATTACH(xL_3,yL_1))(D_1,D_1)

We can attach more than two descriptions with the same call; for instance:

(lambda(x,y,z)(ATTACH(yL_3,xL_1)(zL_1,yL_4)(zL_2,xL_4))(D_1,D_1,D_1)

where D_1 is a square produces the familiar oblique view of a cube:



25. At this juncture we reach a sort of theoretical dilemma: we find that very few objects with which we are visually familiar are, seemingly, represented by complex lists of lines and relations between them; yet when we are pressed further and further towards the primitives of our visual world, we find that lines (representing contrasts, edges, etc.) are the only means for description.

We also find the great difficulty for people to deal with Test E, which requested the subject to decipher a line description of the type mentioned in Section 24. However, in this case it is difficult to shift the blame from the shoulders of possible internal representations: that is, very little visual knowledge appears to be organized into sentences (or schematics), and when presented with such a descriptions we are forced to translate it into a suitable internal representation before we can 'understand' the object. (Remark: subjects were observed drawing the figure in the air with their fingers; some asked to be allowed to draw it.)

The solution seems to be to introduced certain 'compiled' primitive descriptions and a set of construction mechanisms for building with them; we have already seen the first of these mechanisms in the ATTACH group of Section 24. Next we will extend them and begin to supply a set of modifiers for the descriptions.

26. There is an extremely important group of constructions for objects, of which Hollerbach's projective approach is, perhaps, a special case; we have chosen to call them "suspensions" due to their similarity to certain topological constructions.

We have noticed that, when asked, people agree that the essence of a cylinder is a circle: that if we slice a cylinder perpendicular to its axis, the cross section is a circle. Similarly, the essence of a box is a rectangle. In fact this is basically the foundation for the claim that the projective approach is natural.

However, this naturality, though genuine, is not expressed in its most general terms. For example, a cone is again based on

a circle in a simple manner, as is a tetrahedron on a square. Furthermore, if we slice off the top of a cone and look at the remainder, we are still left with the essential "circleness".

Thus we can proceed to give the 0^{th} order approximation to the definition of a suspension:

Definition: Let D_1, D_2 be two descriptions such that D_1, D_2 are homeomorphic to the unit square (in suitable topologies), then Susp(D_1, D_2) is the following object:

D_1 is one face, D_2 is the opposite face, and the middle is 'filled in' in the simplest manner.

Unfortunately this definition leaves quite a bit to the imagination, and, admittedly, much needs to be worked out to complete the definition. We will attaempt to make the intention of the definition clear via examples:

Example 1. Susp(circle,circle)=cylinder

Example 2. Susp(point,circle)=cone

Example 3. Susp(point,square)=tetraheron

Example 4. Susp(rectangle,rectangle)=box or block

Example 5. Susp($\text{circle}_1, \text{circle}_2$)= cone with the top sliced off. $\text{diam}(\text{circle}_1) < \text{diam}(\text{circle}_2)$

Example 6. Susp(square,triangle)= 

Sometimes it might be useful to have some additional notation or footnote to a suspension. For instance, a cube is an example of a special suspension of two squares. But if we wrote:

Susp(square,square)

we would not know, first of all, that the first square is the

same as the second square. We could either write the desired description as

(lambda(x))(Susp(x,x))(square) or as

Susps(square) where 'Susps' denotes a self-suspension

The first notation is more general and adaptable, so we choose it.

However, even 1. loses, since it describes a box with square ends, not necessarily a cube. We could add the proper notation:

(lambda(x))(Susp(x,x))(square)(all E_1)

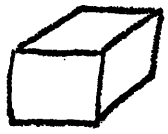
which means that all construction lines used to fill out the description have the notation 'E_1' attached, where E_1 is the length notation required for the sides of the squares.

For the sake of some generality we can modify the notation to:

(lambda(x))(Susp(xE_1,x))(square) which points out that the equality feature is inherited from the first description

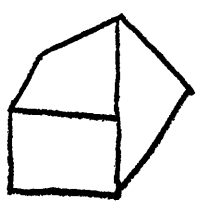
We now list a set of further modifications to this notation.

A. When we say Susp(square,square) we think of the situation:



rather than:

or:



We could state which vertices of the first description are closer to the plane of the second, or we could use the deviance from the perpendicular of the descriptions from the axis between them, as well as the torsion away from the "normal" position.

B. Given two descriptions in the suspension we might to indicate which is larger:

Susp($D_1 < D_2$) or Susp(x,y)($D_1 < D_2$)

(Note: (lambda(x))(Susp(xE_1,x))(square)=Susp($squareE_1$=square)

C. Suppose we want to describe the following:

1.  or 2. 

A natural way might be:

1. Susp(point,circle,point)
2. Susp(circle,point,circle,point,circle)

   or perhaps:

   (lambda(x,y))(Susp(x,y,x,y,x))(circle,point)

D. Finally we might want to describe: 

Here we would want separate descriptors for the individual filler lines which could be appended to the suspension itself. Or we might want to be able to make statements about the shape of the profiles (with symmetry modifiers).

More on suspensions will be said later.

26.1 We can give a precise mathematical representation of the notion of a suspension as it now stands:

Definition: Let D_1, D_2 be two descriptions as above. Let D_1', D_2' be realizations of them in three-dimensional Euclidean space such that the best approximating planes are parallel. $\forall$ point, x, in D_1' and $\forall$ point, y, in D_2', let $L_{x,y}$ be the set of points in the straight line joining x and y. Let $L=U(L_{x,y})$ over all such x, y. Then the Euclidean realization of $Susp(D_1, D_2)$ is given by the points $D_1' \cup D_2' \cup L$.

27. We are now ready to begin discussing perhaps the most important of our descriptive notions, the skeleton. Now we want to be careful not to confuse this with the skeletal pairs of Calabi and Hartnett. They have developed a mathematical theory of skeletal pairs which extends the approach of Blum (and partially explored by Krakauer). Very informally, a skeletal pair is a pair (S,q) where S is a set, called the skelton, and q is a map, called the quench function. Given a set A, its skeleton S consists of those points, x, in A such that the minimum distance from x to the complement of A, $\bar{A}$, occurs at two points. In other words, let y be one of the points in $\bar{A}$ such that no other point in $\bar{A}$ is closer to x than y; if y is not unique, then $x \in S$. The quench function, q, is merely this minimum distance map. Calabi has shown:

Definition: Let A be any set. The Convex Hull of A (denoted CH(A)) is the smallest convex set containing A.

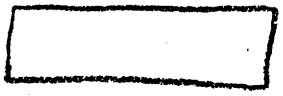
Definition: Let A be any set. The Convex Deficiency of A (denoted by CD(A)) is the set A-CH(A) (or CH(A)/A).

Theorem: A and A' have the same skeletal pair iff they have the same convex deficiency.

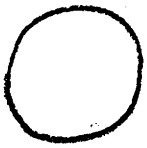
Theorem: Let (S,q) be a skeletal pair. Then the interior

of S is empty.

Examples of Calabi Skeletins:

A= 

S= 

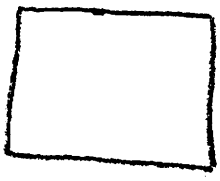
A= 

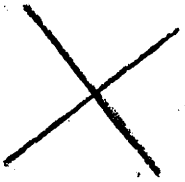
S=

(Note: this example illustrates our informal presentation: actually S is empty here, but this fact does nott effect us. Furthermore, the skeleton of a set is empty iff its convex deficiency is.)

A= 

S= 

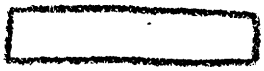
A= 

S= 

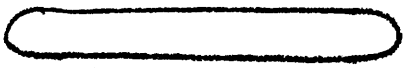
Montanari has developed distance algorithms that compute Calabi skeletons of gray-scale pictures, but here we want to diverge.

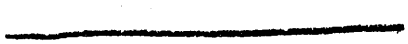
The notion of skeleton we want is simpler, but not as precise.

Examples:

A= 

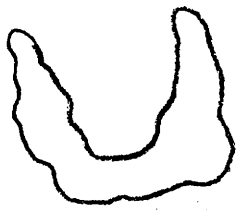
S= 

A= 

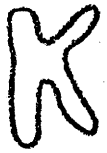
S= 

A= 

S= • or empty

A= 

S= 

A= 

S= 

The 0^{th} order approximation to the definition of a skeleton is:

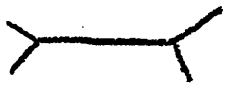
> Definition: Let A be any set. If A 'can be approximated' by a set of 'long, thin cylinders', let S be the set of center lines of them. Otherwise S is empty. We stipulate that these approximations be from within the set when possible.

Exactly how such approximations are to be specified is not clear at this point; perhaps it is best to give a second, motivational definition of the concept:

> Definition: Let A be any set. Then the set of "eye scans" of A is the skeleton of A.

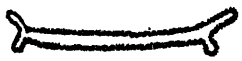
However, no matter what method is decided upon for the computation of skeleta, it should be flexible enough to allow us to expect both of these interpretations:

A* 

S_1* 

S_2* 

With the former being preferred.

But if A= 

then we might want the preference to be reversed.
In any case, a suitable computational definition of skeleton needs to (and we believe can be) decided upon to capture our intuitive notion of it.

For now we will only concern ourselves with the naturality of skeleta and their uses.

First consider Tests D and G in Appendix III; every subject located the "X" immediately, which would seem to indicate that people have a natural skeleton mechanism. Likewise, the "K" in the second test was rapidly located even though a key section of it was obscured. Also peoples' ability to read unusual styles of script points out that this notion is reasonable.

28. Now we return to suspensions, which can be generalized

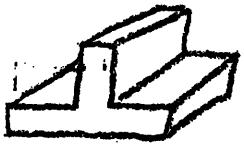
quite a bit now that skeleta have been introduced.

Sometimes we might want to define an object in terms of a skeleton but with a certain quench function. Often this function can be adequately described by means of a suspension. For example, suppose we want to describe the object in Test I.c. Clearly it is both a "U" and a cylinder. Thus:

=(lambda(x))(Susp(skel("U"),x,x))(circle)

=(lambda(x))(Susp(skel("‿"),x,x))(circle)

(lambda(x))(Susp(skel(line),x,x))(skel("⊥") is a description of



and of

And (lambda(x))(Susp(skel(circle),x,x))(circle) is a doughnut !
Thus the general form of a suspension is:

$((\text{lambda}(x_1,\ldots,x_n))(\text{Susp}(\text{skel}(S_1),x_{i_1},f_1,A_1,x_{i_2},A_2,\text{tor}_1,\text{skel}(S_2),\ldots$

$\ldots,x_{i_j},A_{i_{j-1}},\text{tor}_{j-1}))(D_1,\ldots,D_n))(x_{j_1},R_1,x_{j_2},\ldots,x_{j_q},R_q,x_{j_{q+1}})$

where S_i are skeleta
D_i are planar decriptions
R_i are length relations

f_i are length relationson the suspension lines (we are referring to the length along the center lines of the skeleta)
And $f_{i_j} \neq$ empty implies $f_{i_{j+1}}$ is empty, j odd.

A_i are the angles bteween the plane of the description and the center line of the skeleton.

tor_i are the torsions of the suspension lines.

(Note: we have not addressed the considerations in Section 26. D.)

For example:

(lambda(x))(Susp(skel(line),x,45°,x,90°,45°))(square)

is a twisted box with the front tilted back 45°.
In general default values will be the obvious ones.

(lambda(x))(Susp(skel(line),x,E_1,x,45°))(square) is a simple twisted cube.

(lambda(x))(Susp(skel(line),x,45°,x))(square) is

(lambda(x))(Susp(skel("U"),x,x))(square) is

However, even though we have a great deal of power and generality, most uses of suspension and skeleta will only exploit a small fraction of this power, and we will often drop many of these features. (We can indicate these normally dropped features as properties of the dummy variables of the suspension, which can be left empty. Thus we can alter the general form of the suspension to:

$$((\text{lambda}(x_1,\ldots,x_n))(\text{Susp}(\text{skel}(S_1),x_{i_1},x_{i_2},\text{skel}(S_2)..))$$

$$(D_1,\ldots,D_n)(D_{j_1},R_j,D_{j_2},\ldots,D_{j_q},R_j,D_{j_{q+1}})$$

where everything is as above and

$\text{prop}(x_1)=f_1$
$\text{prop}(x_2)=f_2$
$\text{prop}(f_1)=A_1$
$\text{prop}(A_1)=\text{tor}_1$ etc.

We also allow the final list to be dropped.)

28.1. We can, if we want, add some modifiers to a suspension, the effect being to emphasize either the skeleton or the base descriptions (the D_i above). So suppose we have:

D=(lambda(x))(Susp(skel(line),x,x))(circle)

Then Long(D) implies that skel(line) is the dominant feature. While Short(D) implies that 'circle' is important, and that skel(line) is indeed short.

Likewise these modifiers can be applied to other constructions; perhaps some others should be added to this list.

29. A further means of construction is the Rotation, whose usefulness has not yet been determined except as a shorthand for certain classes of suspensions. Suppose we have a planar description, D; then we can roatate it about a pre-determined axis. For instance:

Rot(circle)=sphere
Rot(square)=

Rot(triangle)=

The suspension to which Rot(circle) corresponds is:

$$\lim_{diam(circle_1)\to 0} ((lambda(x))(Susp(skel(circle_1),x))(circle_2))$$

Rot(circle)is a much better notation!

30. Lastly we have the very important modifier, Hole, which inserts a hole into an object. The data necessary are:

a. which face the hole enters
b. which face the hole exits (or how far into the object the hole extends)
c. how big it is
d. where in the entry and exit faces the hole is located
e. what shape the hole is.

The final condition is easily specified by describing the hole as if it were a solid object (i.e. the hole could be cylindrical) and by noting which regions are invisible (the "hole" part).

Thus we have:

$Hole(D_1, D_2, F_1, F_2, Size, Loc_1, Loc_2)$

where D_1 is the object description
D_2 is the hole description; =(D, R_1, R_2) -D is the description and R_1, R_2 are the invisible regions of D
F_1 is the entry face in D_1 (R_1 is in F_1)
F_2 is the exit face (can be "opposite" when D_1 is a suspension and F_1 is one of the end descriptions in the suspension) or a percentage of the distance through the object that the hole extends
Size is the size of the hole at the entry face (the hole description implies the exit size)
Loc_1 is the location of the hole in the entry face
Loc_2 is the location of the hole in the exit face.

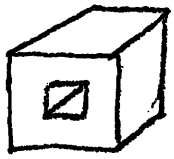
If D_1 is a suspension, F_1 usually refers to one of the end descriptions; or it is "susp" to indicate one of the suspension faces

(we also append, if desired, the line and the description which forms part of the entry and exit regions. We suspect that this specificity is not useful in many cases).
Size is in terms of a percentage of the entry face (default is slightly less than 50%)
Loc_1,Loc_2 are "center" by default and either simply "offcenter" or a line(s) specification to indicate the closest line in the face to the hole.

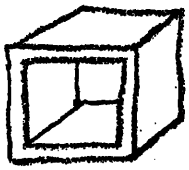
Examples:

let D=(lambda(x))(Susp(skel(line),x,x))(square)

then Hole(D,D,square,opposite,nil,nil,nil) is:

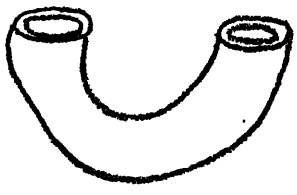


But Hole (D,D,square,opposite,90,nil,nil) is:



(We can drop the terminal nils)
Let CYL=(lambda(x))(Susp(skel("U"),x,x))(circle)

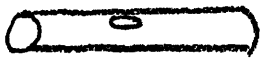
then Hole(CYL,CYL,circle,opposite,90) is:



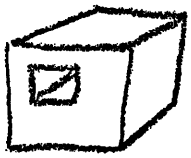
Let cylinder=(lambda(x))(Susp(skel(line),x,x))(circle)

Hole(cylinder,cylinder,circle,90,90) is:

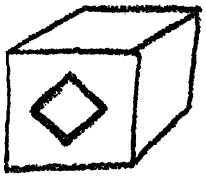
Hole(cylinder,cylinder,susp) is:



Hole(D,D,sqaure,opposite,nil,offcenter,offcenter) is:



Notice, though, that we cannot describe with the available machinery:



Thus, in our formalism we must allow for the specification of orientation of the hole and the object with respect to each other. For now we will denote this by letting D_2=(D,ang). So that the previous example is Hole(D,(D,45°),square,opposite)

30.1 Here are some more examples of suspensions:

(lambda(x))(Susp(skel(square),x,x))(square):

(lambda(x,y))(Susp(skel(line)x,y))(square,circle)

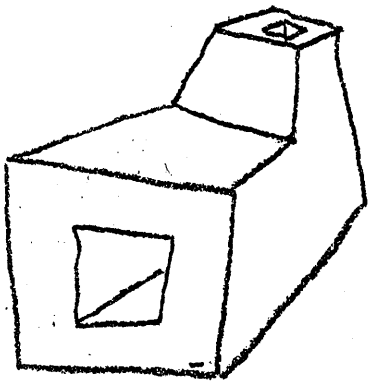
(lambda(x))(Susp(line)(x,x))(triangle)

(lambda(x,y))(Susp(skel(line)x,y))(square,square)(x,y)

(lambda(x))(Susp(skel(triangle),x,x))(triangle)

D_1=Hole(square,square)

(lambda(x,y))(Susp(skel("L"),x,y))(D_1,D_1)(x y)



31. Now we come to the topic of models, where by 'model' we restrict ourselves to the notion of a knowledge system through which we can obtain facts about a particular visual description not explicity mentioned in that description. For us these facts will only be about what views we can expect an object to have, and what shape the visible faces will have.

Since we have several types of description, we are faced with several different problems. For instance, if we describe a cube as 12 lines, 6 faces, 8 vertices with various relations between them, there is no obvious means of determining that, at most, three of these faces can be visible at one time, nor can we understand how the invisible faces are obscured (neither do we a priori know that the 'square' faces are now parallelograms).

People have some versatility in this area, but not nearly as much as commonly supposed. For if we were to present this same description to a person he would be baffled (though it is really not a fair task). But a person could do it, and, since it is one of

the basic types of description available to the machine, it will have to do it as well. Here we would expect that our visual understanding system to construct a model in E^3, much as a person would have to 'visualize' the object. It is not a trivial problem, but certainly not insurmountable. A second obvious technique for obtaining the sort of information we want is to provide, as a part of the description, a list of the possible views. This approach is redundant, though we may want to investigate it with respect to the problem of abstracting descriptions from actual views of objects.

Once the model in E^3 has been determined we can begin rotating the object systematically to compute some of the possible unique views. We need not discover all of them since the recognizer will have access to these routines if it becomes necessary to find a particualr one. Naturally the system must have the capability of deciding which lines are visible and which are invisible from a given viewpoint.

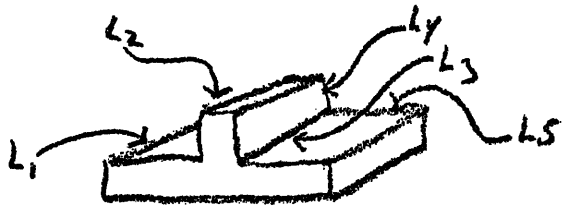
In Appendix I we have developed a fairly clever predicate for visibility (which is perhaps the only possibly elegant result obtained). It is currently designed to work with straight line drawings, however the general algorithm is easily extendable to curvilinear drawings with a modification of an intersection subroutine.

Thus the general strategy is to systematically "imagine" the object from various viewpoints and note the unique views until we get no new views between two unique ones (a binary type search).

Hopefully we would not be required to determine visibility from scratch with each new view, since we can pay attention to those lines that are close to possible obscuring regions. So when we make a comparatively small rotation, e°, we can leave the visibility flags as in the previous view and then spend time checking the dangerous points (one heuristic might be to watch lines partially obscured and those connected to totally obscured

lines).

For example, suppose we have the view:



when we rotate counterclockwise we need onle watch lines L_1 and L_2 (and possibly L_3, L_4, and L_5).

However, very few of our descriptions are of the line drawing variety: most are structural or involve attachments of known descriptions. Therefore we can expect that the views of some of these structures are predictable and that, if a model is needed, its construction is simplified by our knowledge of the structure of the description (which would be a more realistic model of human behaviour as well as being a simple solution to our problem).

32. The first example of the kind of savings we have in mind occurs with attchments.

Suppose that we decide to only store away visibility and views at a small number of different angles. Now assume that we ATTACH two objects for which these views have been determined, and suppose we want to determine the visibility of this compound object at one of the stored angles. At worst we need only compute the cross-visibilities; that is we need only consider how each object obscures the other. If we know certain features about each and how they are attached, even this much work is unnecessary. For example, if both objects are convex and one is on top of the

other (and we are viewing from above), we need only consider how the top object hides part of the bottom object.

33. In the case of suspensions the situation may be much simpler. Assume we have a typical suspension with a linear skeleton and only two base descriptions, D_1, D_2. Now the views we expect are quite simple:

a, we can see only D_1 unless $D_2 \not\subset D_1$

b, vice versa

c, we can expect to see D_1, some filler faces, and a rear profile of D_2

d, vice versa

e, we can expect to see the filler faces and side views of D_1 and D_2

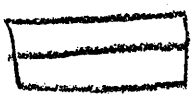
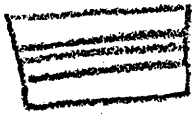
For example, let D_1, D_2 = 

Consider: (lambda(x))(Susp(skel(line),x,x))(D_1)

Then the views are:

a,b 

c,d  etc.

e,   etc.

The reader can try drawing some views of a linear suspension and see for himself how mechanical and simple it is.

(lambda(x,y))(Susp(skel(line),x,y))(triangle,triangle)(x∠ẏ)
Some of the views are:

For non-linear skeletons in a suspension we can make similar predictions about visibility.

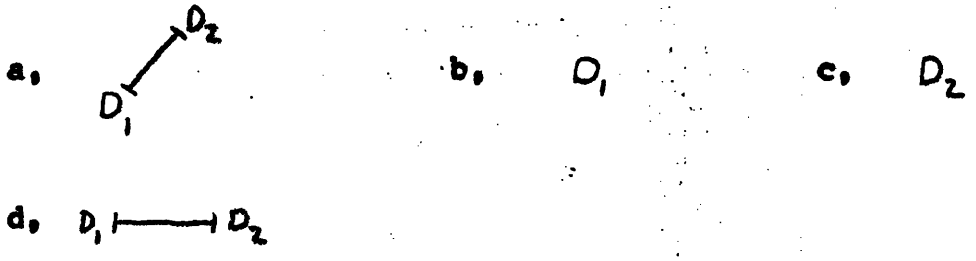
(lambda(x))(Susp(skel("L"),x,x))(square)

If the skeleton is non-linear (but not a loop) then we can expect a view where both base descriptions are visible, neither are visible (depending on the skeleton), the filler lines plus rear profiles may be visible, and etc. The filler lines are a simple product of the skeleton and the base descriptions.

In general our conjecture is that suspensions have a certain modular structure for its visibilities that are somewhat independent of the specifics of the suspension, and we need only plug in particular modules from the suspension to obtain the desired information. Especially if the system has only a small number of skeleta for use in its suspensions, these structures are likely to be very simple. Thus whenever we specify a new suspension definition to be put into the system, we create a set of view structure that have certain details filled in accordding to the actual suspension. Some of these structures can be eliminated by the sus-

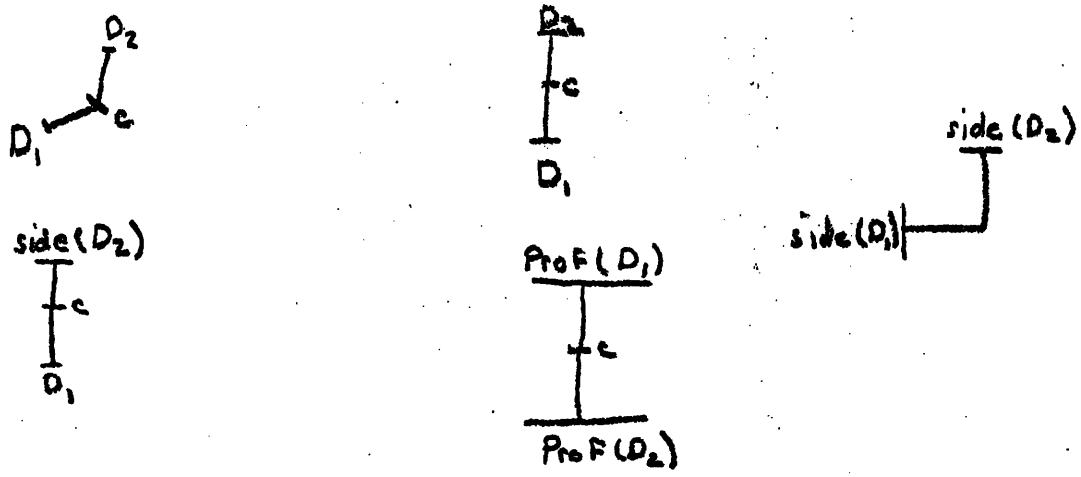
pension, others created.

For example, for linear suspensions we can expect these different views unless $D_1 \angle D_2$ or vice versa:



which get filled in by (lambda(x))(Susp(skel(line),x,x))(D_1) as shown earlier. However, in these schemes-for this example- views b and c are identical, thus we drop one; we also notice that view a contains a line in the rear profile of D_1 that is obscured, so we modify the views to include this new one:

For suspensions with skel("L"), we can expect:



side(D)=side view of D 'c' indicates a corner.
prof(D)=rear profile of D

For suspensions with a loop skeleton we can expect a top, a side, and an oblique view where the top view is the shape of the loop, the side reflects the way the base descriptions effect the filler lines, and the oblique view is a product of the two.

33.1 Sometimes we might not be willing to accept certain views though they have be found to be possible. For instance, the following are views of a cube:

But since they are seen so rarely we do not readily accept them, as was shown in Test H. Because these views can only occur for very specific angles, this may be the correct heuristic to use in eliminating them.

34. The views of a rotatation are trivial: top and bottom views are circles, all side views are the base description.

Views need not be computed for skeleta due to the proccess by which they are recognized.

Holes are essentially straightforward.

35. The final concept needed for real object recognition can be called perspective: that is, a, how the shape of a face is distorted due to viewing angle, b, how classical perspective further distorts this shape.

Thus the first part deals with the phenomena of foreshortening and angle mutilation, while the second deals with the behaviour of parallel lines and various other length distortions.

We can, of course, solve the problem explicitly by performing perspective projection in our models, but this requires a great deal of abstraction from particualrs, which is probably not done by people anyway. Recall, also, that models, thus far,

have only been used to discover the possible views of an object. Here we want to known things like what a square looks like from and oblique angle. A more likely situation is that people only understand things like "squares become parallelograms; circles become ellipses" etc.

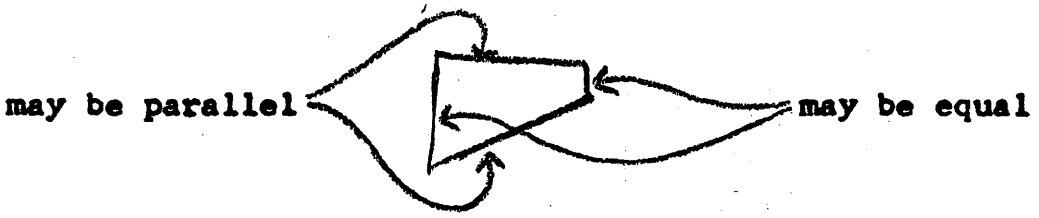
Thus we can allow a few rules such as this plus some classical perspective hints like:

a, converging lines may be parallel and are further away closer to the convergence.

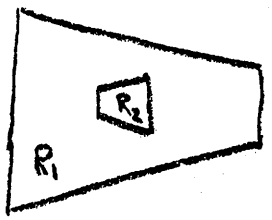
b, adjacent faces with inconsistent convergence imply separate planes and different orientation.

c, parallel lines whose length is consistent with the convergence may be of equal absolute length.

d, similar convergence implies parallel planes.



R_2 is not in the same plane as R_1



36. At last we come to the recognizer. For the purposes of this paper the recognizer will only be called upon to locate a specified object in a scene, but we will also include a scenario for how the object descriptions could be used in a general visual system.

First of all we provide the system with a set of primitive descriptions, including:

a, Triangle (several types)
b, Rectangle
c, Square
d, Parallelogram
e, Quadrilateral
f, Circle
g, Ellipse
h, n-agons $5 \leq n \leq N$ (some small N)
i, Sphere,

each of which will be referred to by name, but whose essence is a primitive line description. We also provide some useful skeleta:

a, various letters of the alphabet like O,T,L,U,X,Y,S,W,C
b, $\perp$,line,wavy lines of various kinds

Now we start inputting descriptions into the system via line description, attachments, suspensions, rotations, holes, and skeleta; at each point the system calculates some of the views possible and indexes the whole collection roughly as follows:

a, In a line description we index under the primitives mentioned plus all faces. If no primitives are mentioned, we can match for them.

b, In a simple suspension
if the skeleton is not a loop it is indexed under the end or base descriptions and the skeleton.

if the skeleton is a loop it is indexed only under the skeleton.

In a regular suspension (with more that one skeleton) it is indexed under the indices it would have if it were a string a simple suspensions.

c. In a rotation we index under "circle" plus the base description.

d. In a skeleton we index under the skeleton.

e. In an attachment we index:

let I_1 be the index of D_1

let I_2 be the index of D_2

if we have ATTACH(D_1,D_2,...) where faces $f_1,\ldots,f_n$ are identified, then we index under $I_1 \cup I_2 - (f_1,\ldots,f_n)$.

f. I

f. In a mofified suspension we index under the appropriate feature:

LONG(D) indexes under the skeleton

SHORT(D) indexes under the base descriptions.

These indices allow us to access descriptions having a particular outstanding feature. Later we may want to augment this memory by having semantic pointers from some to others structurally related. (Note: this indexing scheme may need to be modified and is only intended as a hint at the proper arrangement.)

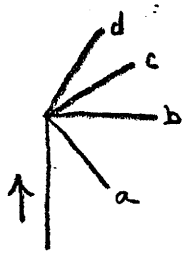
37. Here we deal with the recognizer with respect to these indexed descriptions (we have left out a section that dealt with certain unindexed descriptions-basically complex line descriptions-which used special outline heuristics that took advantage of the ability of the vidissector to track boundary lines where there is a sharp contrast. However, these techniques seemed inappropriate to the purpose of outlining a general theory of visual recognition; it may well be that it will become advantageous to pursue them further when the time comes for a system to be implemented. Essentially the proceedure was to find an outline and see if some subset of it could form the outline of the object in question. However, the proceedure was somewhat obscure and appeared to be extremely unnatural.). In particular we will consider only short suspensions and line descriptions. We will also only mention a few of the features of the recognizer.

The basic strategy is very simple: we search the scene for one of the basic descriptions occurring in the description we wish to instantiate; these basic descriptions are, of course, the ones under which the description is indexed. Thus suppose we want to find: SHORT((lambda(x))(Susp(skel),x,x))(square)). We search the scene for squares; if we find one we then verify the rest of the description: namely, the straight line projection lines, the rear profile of the second square, and the lengths of lines, parallels, etc. (Of course, in checking the projection lines we need only check that the rear profile is correct, that the lines join properly, that the correct lines are obscured, and that the proper faces appear.) Line descriptions require us to conjecture the correct view and crawl around on the lines. Notice that the description we hope to verify helps guide the tracking routine by suggesting directions and line types (boundary, edge, roof etc.), whether it is line description or a suspension.

Usually we have a subroutine whose purpose is to search the scene for these basic descriptions. This routine uses the basic description as a guide to the tracking routine in much the same way.

Suppose we are searching for a particular basic description and have found a line we hope will form one of the lines in the description. When we reach a junction where several lines diverge, we refer to our description and continue tracking along the line that seems most likely to be a continuation of the description.

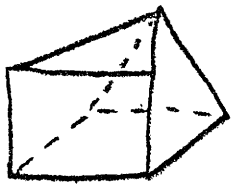
For example, assume we want to find a Square and have tracked until we find:



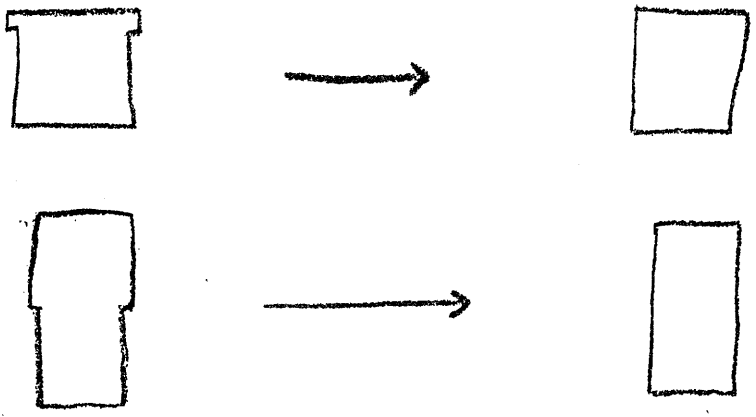
our first choice is b since it subtends 90° with respect to the tracking line. If this fails we can try using our knowledge of perspective to complete the Square. Also, we keep a record of all the lines found and, when in a bind, we can try extending collinear lines with the appropriate obscuration conditions hypothesized.

We remark that, in general, it is the duty of the recognizer to account for every discrepancy between the description and the input with a positive reason (lighting, obscuring bodies), although when we are in a hurry we will only check for the basic descriptions and an indication that the right kinds of connections exist. Also, the features located (i.e. which basic descriptions and what sort of connections exist) tend to alert us to the proper view to verify. For instance, suppose we are looking for an instance of (lambda(x,y))(Susp(skel(line),x,y))(square,triangle). We begin by looking for either a square or a triangle; if we see:

then we know that the correct view to verify will be:



Other recognizer heuristics involve approximation and frill elimination. These heuristics operate in terms of short lengths and virtual collinearity:



Similarly, if a line is almost straight (only slightly wavy), and line we are attempting to verify is not supposed to be absolutely straight, we can assume that it is straight and blame the discrepancy on poor input.

It may be (and we believe that it is desireable) that we always want to run the preliminary recognizer until it finds a complete instance of one of the basic shapes. For example, if we are attempting to locate a square and we come across a curved line, we should continue to verify that it is a circle (or whatever). The purpose is to allow us to make some kind of judgement concerning the identity of some of the objects in the scene. Thus suppose we find:

A= 

B= 

If a and a' are collinear, and b and b' are also, we might want to conclude that the two parts form a cylinder (and without this collinearity we might wish to conjecture two cylinders, though a more reasonable report would be that there are two Circles).

The advantages of this are obvious: as we go along we are gathering potentially useful information about the scene we are viewing. Since we are, a priori, admitting that these identifications are error prone we can afford some liberty at the price of later conservatism. When pressed for time this sort of recognition heuristic can be used to locate the desired object.

Finally, we might find ourselves in the situation where the nature of the scene fails to account for discrepancies between the description and the input. In this case we might want to re-investigate our model to determine if there is a so far undiscovered view like the one we are now studying.

38. To recognize skeleta, Loop suspensions and Long suspensions we first need to compute the skeleton of the object.

Initially we locate objects by using collinearity extension heuristics and various straight line approximations; perhaps we can use a large percentage of the mechanisms mentioned in the previous section. Once we have the skeleton we either:

a. match the skeleta
b. check the skeleton for a loop and then pass control to the suspension recognizer (for loop suspensions)
c. match the skeleta and pass control to the suspension recognizer (for Long suspensions).

The extent to which searching for basic descriptions and searching for skeleta are related has not been adequately investigated.

38.1. Rotations are found by searching for the base description or a circle.

39. We have given in amorphous form the rudiments of a theory of vision (in the form of a theory of object location) which, we feel goes a long way towards the understanding of the visual process.

We now give a brief outline of that theory:

a), Vision, as it is usually considered, is recognizing objects. That is, unless we have something with which to associate a visual image, it is useless and gives rise to confusion.

b), The most efficient, and most often used, means of recognition is verification. That is, we have a description of the object in mind, an understanding of the three-dimensional implications of the description, and a method for verifying that what we have in mind is indeed in front of us.

c), Our basic level of visual understanding is primitive shapes from which we build descriptions by means of various constructions like suspensions, rotations, and attachments. These constructions have a basic structure which allows us to understand the object visually by filling in details on a modular form. (These structures are like natural transformations in Category Theory wherein the nature of a family of maps can be assessed without reference to the sets on which they are defined.)

d), A powerful tool in our visual system is the skeleton which allows us to specify objects by only mentioning their most basic shape or structure.

e), Our basic level of visual recognition is this set of primitive shapes: we see these primitives as a structural whole without any apparent effort. These primitive shapes are a major basis for guessing what objects are and for verifying them. Thus we <u>see</u> a triangle and not three sides.

f), When we search a scene to find a particular object we attempt to see an instance of one of its basic constituents; when we find one (or several) we verify the rest.

g), As we scan the scene we recognize other primitive shapes, and, on the basis of this and other noted features near these shapes, we infer a possible identity for this object, but usually do not pursue them much further.

h), The notion of expecting what we will see and then verifying it with a few checks is central to the speed with which people must be able to see in order to survive. When we see what we do not expect, we are confused and it takes us a while to see much of anything.

i), The semantic visual memory in terms of indexing and its logical extensions is a step towards allowing us to shift our expectations of a visual scene efficiently.

Postscript

In sections 26-34 we have presented to the reader a method of description and a recipe for its use; the method and the recipe are examples of a more abstract phenomenon that deserves explicit mention. Unfortunately this phenomenon is best phrased in the language of category theory, but will be presented here in a very informal manner.

There are several ingredients that need to be introduced before the main defining properties are presented.

1. A set of descriptions called DESCR. Since we want to deal with structured descriptions we require elements of DESCR to be of the form:

$$St_i(D_1,\ldots,D_n)$$

St_i is the <u>structure</u> of the description and $D_1,\ldots,D_n$ are the <u>base</u> or <u>primitive descriptions</u>.

Examples: Simple suspensions are structured descriptions with the skeleton as the structure. General suspensions and rotations.

2. Between two instantiations of the same structure is a map:

$$g: \text{DESCR} \longrightarrow \text{DESCR}$$

such that $g(St_i(D_1,\ldots,D_n)) = St_i(g_1D_1,\ldots,g_nD_n)$, which simply tells how to change primitive descriptions to get from one instantiation to another.

3. A set, MOD, of models of objects in E^3 - that is, an element of MOD is a collection of vertices, lines and regions in 3-dimensional Euclidean space.

4. A map:

$$Mod_1: \text{DESCR} \longrightarrow \text{MOD}$$

which, when given an element of DESCR, yields <u>one</u> of its models in MOD. There are typically many satisfactory candidates for Mod_1 - i.e. since many models are rotations, dilations, or translations of a single model, there are several set-theoretic maps that do assign to a given element of DESCR one of its models (in MOD) in a satisfactory manner; however, all maps mentioned in these sections will be subject to properties A and B below.

5, A set MODV of models in E^2 of views of objects in MOD. That is, elements in MODV are collections of vertices, lines, and regions in 2-dimensional Euclidean space that represent the projections of elements of MOD onto the picture plane as viewed from various viewing positions.

6, A map:

$$f'_a: MOD \longrightarrow MODV$$

which, when given an element of MOD, yields its view (or appearance) from viewing position a.

7, A set PVIEW characterized as "prescriptions of views" of elements of DESCR.

Each element of PVIEW is a description of what a given object looks like from a particular viewing position in terms of spatial relationships and visibilities of perspective variations of the primitive descriptions (we assume that the views of the primitive descriptions are known) Thus the elements of PVIEW are <u>symbolic</u> descriptions of the views of an object, and since they are symbolic, there are only a finite number of distinct viewing positions. For instance, an elements of PVIEW might be like: "a front view of D_1, side views of D_2 and D_3; (spatial relationships)." Hence, we write the typical elements of PVIEW as:

$$PV^i_a(D_1,\ldots,D_n) \qquad a \in V$$

(V is the set of distinct viewing positions) which indicates

that these descriptions are instantiations of patterns of the form:

$$PV_a^i(x_1,\ldots,x_n)$$

8. A map:

$$f_a : DESCR \longrightarrow PVIEW$$

which, given an element of DESCR, yields the appropriate element of PVIEW.

In general we want f_a to be a proceedural map rather than a set-theoretic one. Thus we specify that f_a is composed of individual maps, each of which may depend on the particular description in DESCR to which it is applied. Thus we have:

$$St_i(D_1,\ldots,D_n) \xmapsto{f_a^{St_i(D_1,\ldots,D_n)}} PV_a^j(D_1,\ldots,D_n)$$

and: $f_a(St_i(D_1,\ldots,D_n)) = f_a^{St_i(D_1,\ldots,D_n)}(St_i(D_1,\ldots,D_n))$

Hence, though there is but one set-theoretic map, there may be as many proceedural maps composing that map as there are elements in DESCR; property B will make a statement about this and the structure of PVIEW.

9. A map, entirely analagous to Mod_1, called Mod_2

$$Mod_2 : PVIEW \longrightarrow MODV$$

10. Finally a map:

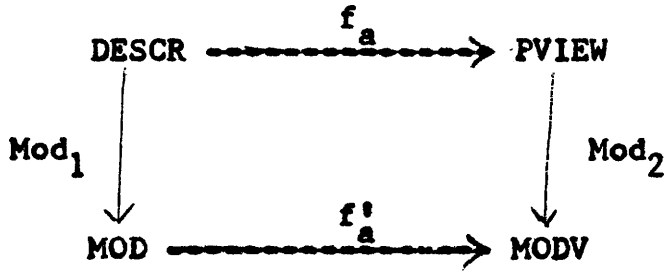
$$g^* : PVIEW \longrightarrow PVIEW$$

such that $g^*(PV_a^i(D_1,\ldots,D_n)) = PV_a^i(g_1D_1,\ldots,g_nD_n)$

where the g_j are as in 2.

Property A.

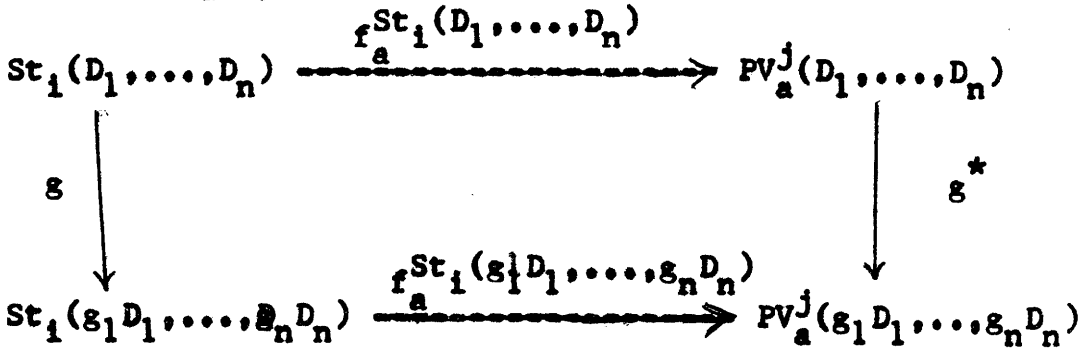
All maps above are such that for a∈V



commutes. (i.e. $Mod_2 f_a = f'_a Mod_1$)

Property B.

For all $St_i(D_1,\ldots,D_n)$



commutes.

First consider Property A. As mentioned above there are many elements of MOD which could be the image of an element of DESCR under a proposed map Mod_1; Property A insures that we have defined Mod_1 and Mod_2, which shares the same dilemma, in a consistent manner. In light of this argument, however, we see that the left inverse of Mod_2 exists. I.e.

$$\exists\ Mod_3 : MODV \longrightarrow PVIEW \quad \text{such that}$$

$$Mod_3 Mod_2 = I_{PVIEW} \quad \text{although}$$

perhaps $Mod_2 Mod_3 \neq I_{MODV}$.

Since $Mod_2 f_a = f_a' Mod_1$ we have $Mod_3 Mod_2 f_a = Mod_3 f_a' Mod_1$

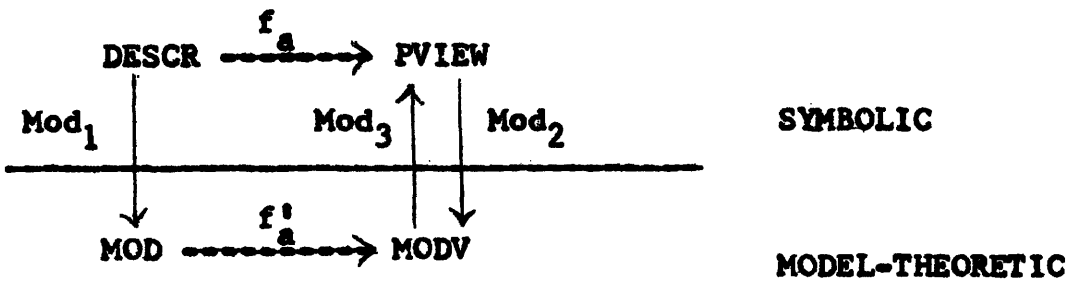
which implies that $f_a = Mod_3 f_a' Mod_1$.

This states that the system

$$DESCR \xrightarrow{f_a} PVIEW$$

is as powerful as generating a model, manipulating views with f_a', and abstracting a description with Mod_3.

So we can partition the diagram as follows:

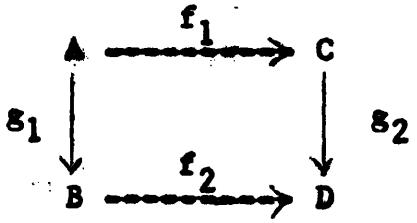


and see that one need not leave the symbolic world if property A holds

Defining Maps the Same Way

Sometimes we want to say that, mathematically, two maps are examples of the same proceedure, although they are distinct entities in a strict sense; when such a situation arises we say that the two maps are "defined the same way".

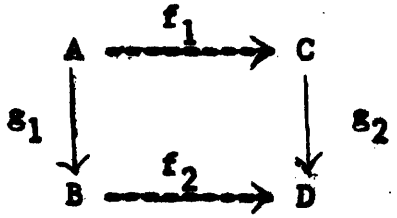
Suppose we have the following diagram:



Suppose that A,B,C, and D have some kind of structure which, 1, we are interested in, and, 2, is similar in some clear and relevant way. Formally we satisfy these requirements by demanding that A,B,C, and D be objects of the same Category.

Also assume that g_1 and g_2 are maps that, 1, preserve the structure, and, 2, are similar in some clear and relevant way. Formally this is done by demanding that g_1 and g_2 are images of the same map (in another category) under two functors (thus A(B) and C(D) are images of the same object under these two functors).

Then, if, for all such g_1 and g_2



commutes ($g_2f_1 = f_2g_1$), we say that f_1 and f_2 are defined the same way; that is, the definitions of f_1 and f_2 depend only on the structure shared by A and B. This is intuitively clear since, over all possible g_1 and g_2, it does not matter whether we alter

the details of the structure and apply f_2 or apply f_1 and then alter the details. (Those familiar with Category Theory will recognize the natural transformation in some disguise.)

—

It is now clear that property B indicates that $f_a^{St_i}(D_1,\ldots,D_n)$ and $f_a^{St_i}(g_1D_1,\ldots,g_nD_n)$ are "defined the same way" with respect to the structure St_i. $St_i(D_1,\ldots,D_n)$, $St_i(g_1D_1,\ldots,g_nD_n)$, $PV_a^j(D_1,\ldots,D_n)$, and $PV_a^j(g_1D_1,\ldots,g_nD_n)$ have the same formal structure; also g and g^* obviously preserve these structures (since each is an isomorphism) in entirely analagous ways. Thus all the conditions mentioned in the previous section have been satisfied, which means there is a single proceedure for each structure rather than a multiplicity of them; we write:

$$f_a(St_i(D_1,\ldots,D_n)) = f_a^{St_i}(St_i(D_1,\ldots,D_n)).$$

Furthermore the image of f_a is ismorphic to DESCR, and there is one such isomorphic copy for each $a \in V$.

Hence any descriptive system that obeys properties A and B is such that the maps (or proceedures) yielding the prescriptions of views depends only on the underlying structures of the descriptions.

For example, in the case of suspensions the structure is essentially the skeleton, and the proceedure for determining the basic views depends only on the skeleton. (Naturally many views dependson the concavities of the basic descriptions.)

In the case of rotations, the structure has no intuitively interesting representation, although the map, f_a^{rot}, is perhaps more transparent than those for skeleta.

Attachments accordding to a skeletal plan can possibly be a useful example of this system.

In conclusion, the power of the system is derived from property A, while the potential usefulness is evident in

property B. Moreover, if the simplicity obtained from the fact that there is only one proceedure for determining the basic views for each structure can be augmented by the internal simplicity of these proceedures, this system will indeed be of significant value in machine vision.

Bibliography

Arnheim, Rudolf, "Art and Visual Perception", University of California Press, 1954

Calabi, Lorenzo, "The Rudiments of a General Theory of Skeletal Pairs", Parke Mathematical Laboratories, TM-4, AFCRL, Contract No. F19628-69-C-0028, May 1969

Calabi, L.
Hartnett, W.E. "Shape Recognition, Prarie Fires,Convex Deficiencies, and Skeletons", Parke Mathematical Laboratories, SR-1, AFCRL, Contract No. AF19(628)-5711, Feb. 1966

Guzman, Adolfo, "Computer Recognition of Three-Dimensional Objects in a Visual Scene", MAC-TR-59 Dec. 1968

Guzman, Adolfo, "Some Apsects of Pattern Recognition by Computer", MAC-TR-37, Feb. 1967

Hollerbach, J.M., "The Projective Approach to Object Description", Vision Flash 37, Dec. 1972

Krakauer, L.J., "Computer Analysis of Visual Properties of Curved Objects", MAC-TR-82, May 1971

Montanari, U., "A Method for Obtaining Skeletons Using a Quasi-Euclidean Distance", JACM, Oct.,1968, 15:4

Roberts, L.J., "Machine Perception of Three-Dimensional Solids", Optical and Electro-optical Information Processing, James J. Tippet (ed.) et al MIT Press, 1965

Shirai, Y., "A Heterarchical Program for Recognition of Polyhedra", AI Memo 263.

Waltz, D.L., "Generating Semantic Descriptions from Drawings of Scenes with Shadows", AI-TR-271, Nov 1972

Weiss, Ruth, "Be Vision" JACM 1968

Winston, P.H., "Learning Structural Descriptions from Examples"', AI-TR-231, Sept 1970

Neurophysiology of Vision

Guyton, Arthur C., "Basic Human Physiology: Normal Function adn Mechanisms of Disease" W.B. Saunders Co. 1971, Chptrs 34,35

Hubell, D.H., "Eleventh Bowditch Lecture", PHYSIOLOGIST, 10:17 1967