SHADOWS AND CRACKS

VISION FLASH 14

by

Mark Dowson and Dave Waltz

Massachusetts Institute of Technology

Artificial Intelligence Laboratory

Vision Group

June 1971

ABSTRACT

The VIRGIN program will interpret pictures of crack and shadow free scenes by labelling them according to the Clowes/Huffman formalism. This paper indicates methods of extending the program to include cracks and shadows and shows that such an extension makes available heuristics which allow the program to be less simple minded.

Work reported herein was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported by the Advanced Research Projects Agency of the Department of Defense, and was monitored by the Office of Naval Research under Contract Number N00014-70-A-0362-0002.

The Clowes/Huffman algorithm for interpreting line drawings has been adequately described elsewhere. (e.g. Clowes 1971) Briefly it consists of interpreting each line in a picture as one and only one of four varieties of scene edge. This process is constrained in that each particular shape of junction permits only a small number of the possible combinations of interpretations of its lines. Mark Dowson has written a MICROPLANNER program, VIRGIN, which will 'parse' a picture according to the algorithm by assigning to each line a label which indicates its interpretation. The program embodies (as does the original algorithm) some tight restrictions on the nature of both the scene and picture domains. They are as follows:-

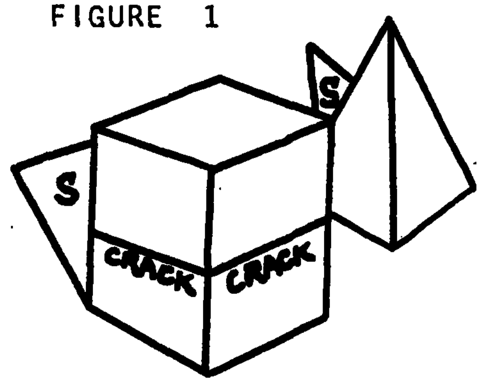
The Picture Domain

Pictures consist of straight lines, two or three line junctions and closed regions.

The Scene Domain

Scenes contain collections of opaque solid polyhedra whose vertices consist of exactly three intersecting surfaces. There are no cracks or shadows in the scene (see Fig. 1) and the texture of each surface is homogeneous. Scenes are viewed from a 'generalised position' such that small changes in the viewing position leave the geometry of the resulting picture unchanged and there are no ' nasty coincidences ' of vertices (see Fig. 1).

FIGURE 1



A 'cracked' body caught at a nasty coincidence with a tetrahedron while casting a shadow.

Extending the VIRGIN program means relaxing some of the restrictions on the kind of scene it will deal with. Some restrictions like 'generalised position' and 'homogeneous surface texture' do not seem very important at the moment. We will accept others such as 'only trihedral corners' and 'no nasty coincidences' because an attempt to relax them would lead to a much more complex program with little compensating gain in generality. The restrictions we would really like to get rid of at the moment are those which exclude scenes with 'cracks' and 'shadows' as most of the scenes that the vision system has to deal with are plagued by both. (We would, of course, like to be able to deal with curved surfaces, but conjecture that a quite different approach may be neccessary.)

The VIRGIN program embodies its knowledge about pictures and what they represent in a collection of PLANNER theorems about the possible labelings of junctions. These labelings are exhibited in Figure 2. To extend VIRGIN to handle cracks and shadows it is only neccessary to add to the program junction theorems which correspond to the kinds of picture junction which arise from scenes with cracks and shadows.

Dave Waltz has recently catalogued the types of junction involving shadows which will be found in pictures of scenes illuminated by a single light source. They are shown in Figure 3. Figure 4 shows the set of junction labelings for scene vertices involving cracks. These were constructed by Mark Dowson as an extension of his account of how the set of junction labelings embodied in VIRGIN arose in the first place. (See Vision Flash 13)

FIGURE 2

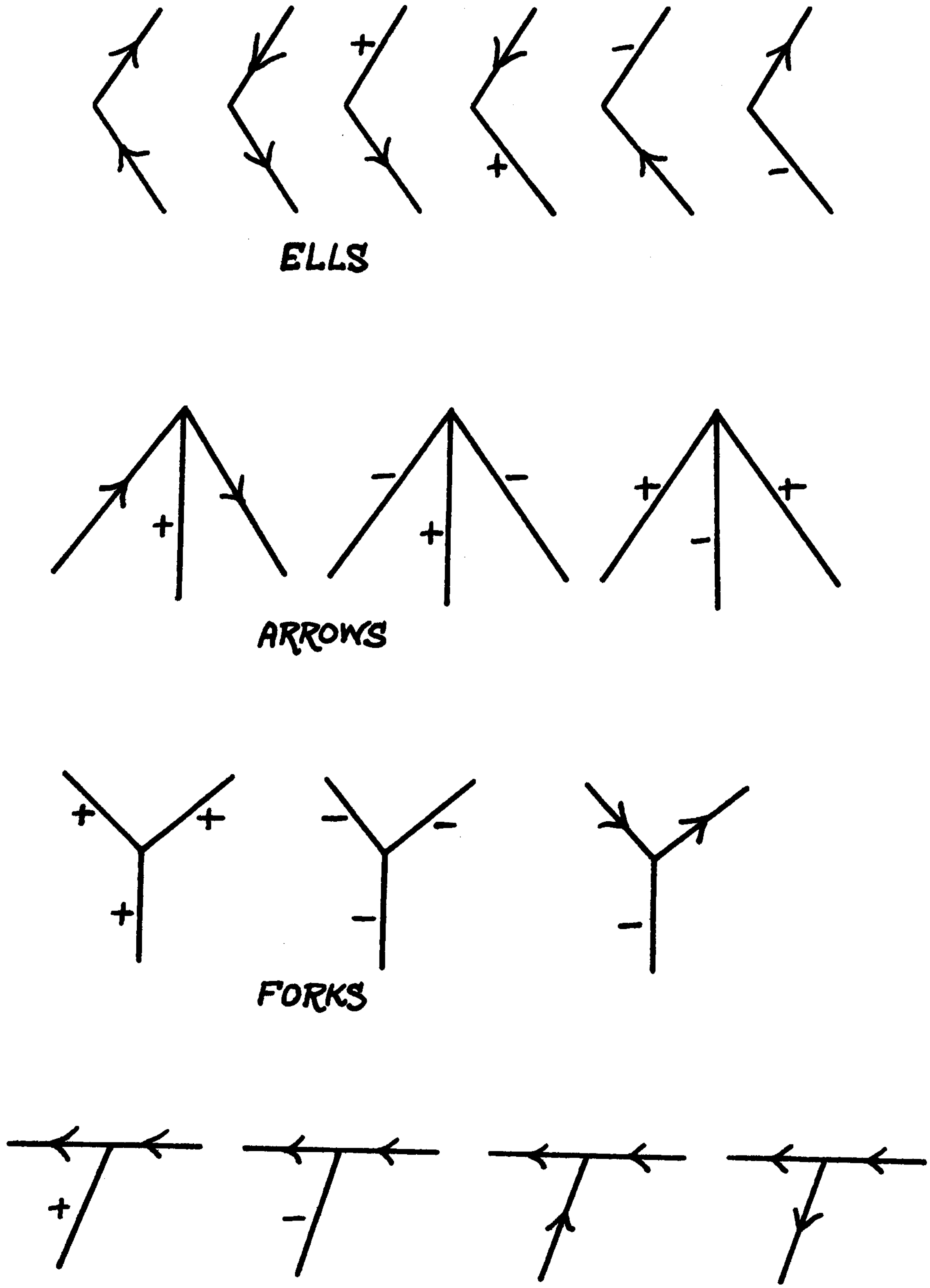
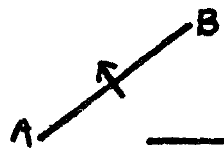


FIGURE 3



AB is a shadow edge, both sides of the edge coplanar; arrow head in the shadowed region.

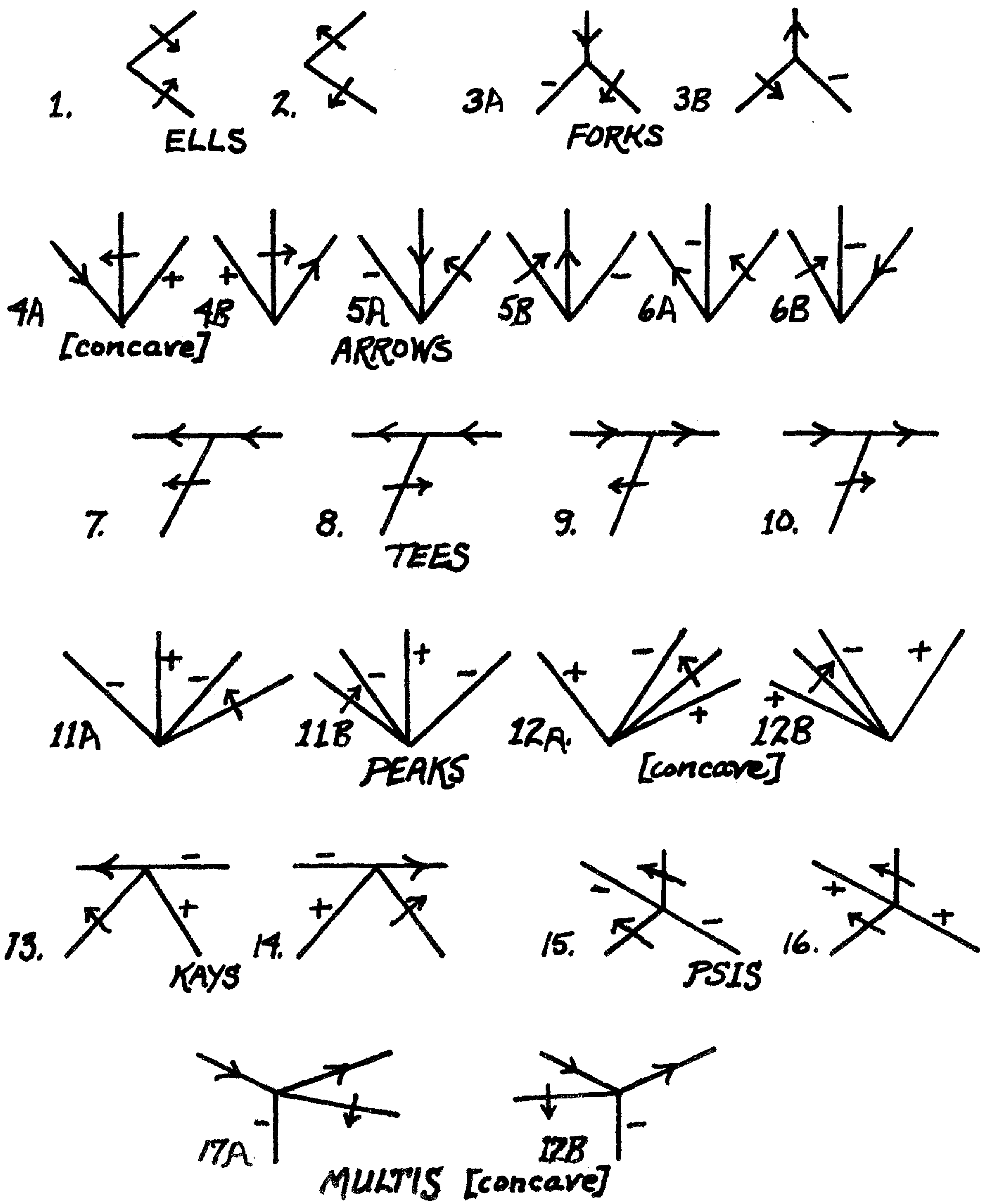
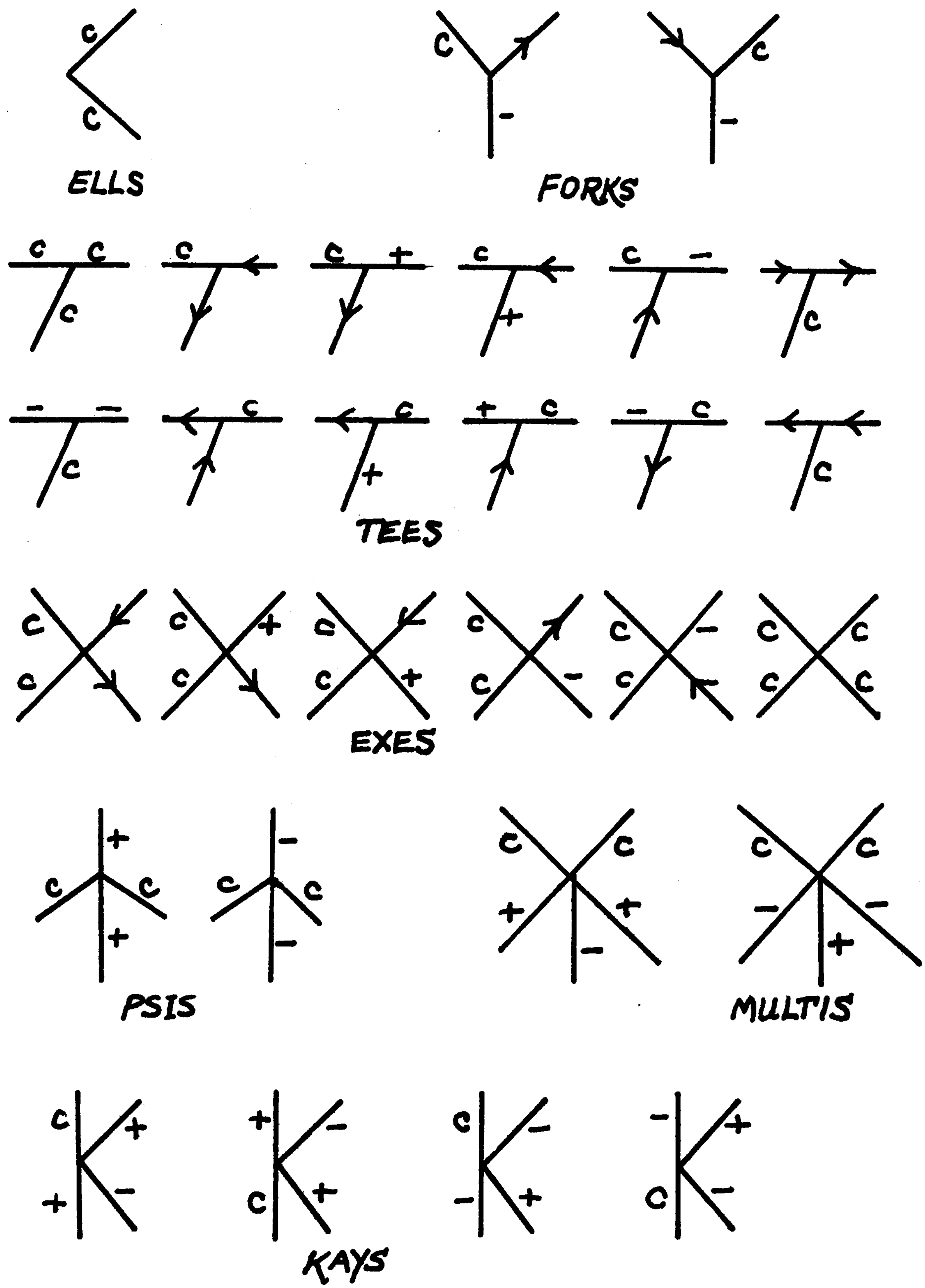


FIGURE 4



Two points should be immediately noted. 1) Considering that three additional edge types have been added ('crack' and two directions of shadow edge) there are remarkably few extra labelings. 2) With the notable exception of TEE junctions there is little overlap between junction types which include shadow and crack labelings and the original VIRGIN set. This implies that if an extended program is set to interpret a shadow and crack free scene it will do so nearly as fast as the original program and produce few extra interpretations.

We do not know what the performance of a program which was a simple extension of the VIRGIN parsing program would be like. Clearly it would be considerably slower and require more space for the interpretation tree. We suspect that in at least some cases it would produce an inconveniently large number of (possible) interpretations of a picture since VIRGIN is innocent of which interpretations are more likely than others. Luckily, however, there are obvious ways to make the rather mindless approach of VIRGIN more efficient in the extended case. In fact, the problem is to decide which of the superfluity of such goodies to actually use. To use them all would make the program hopelessly complex and impossible to understand or extend further. When we have a deeper theoretical understanding of some of the available heuristics it will be easier to incorporate them without running into this trouble.

The available heuristics fall into distinct categories. First there are those which depend on getting extra information from the line finder/preprocessor level. If one can determine that a particular line is a shadow line or a crack and make an assertion to this effect in advance of attempting to parse the picture the search space will generally be much reduced. Even tentative identifications can be used to direct the order of selection of junction labelings.

Secondly there are heuristics which arise from the nature of the table of junction labelings e.g. any EXE junction must have at least two crack lines.

Thirdly there are global properties of particular scenes which can be determined from partially completed parsings and used to direct the remainder of the parsing. Direction of lighting probably falls into this category.

Some of these heuristics are discussed more fully below.

Shadows contain information in the sense that they enable a program to interpret otherwise ambiguous scenes in fewer ways; sometimes a shadowed scene will have only one possible parsing. Notice that the VIRGIN program described above can give four equally valid physical interpretations to an isolated cube. The lower edges of the cubes in figure 5A can be labelled either way, but as shown in figure 5B when shadows are present, the two cases are distinct from each other.

FIGURE 5A

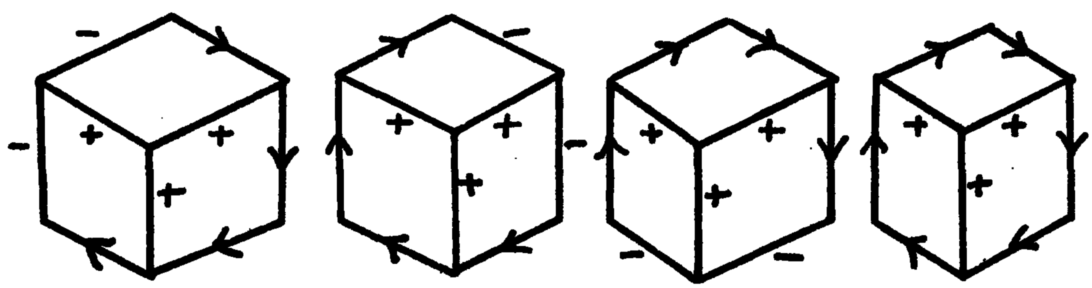


FIGURE 5B

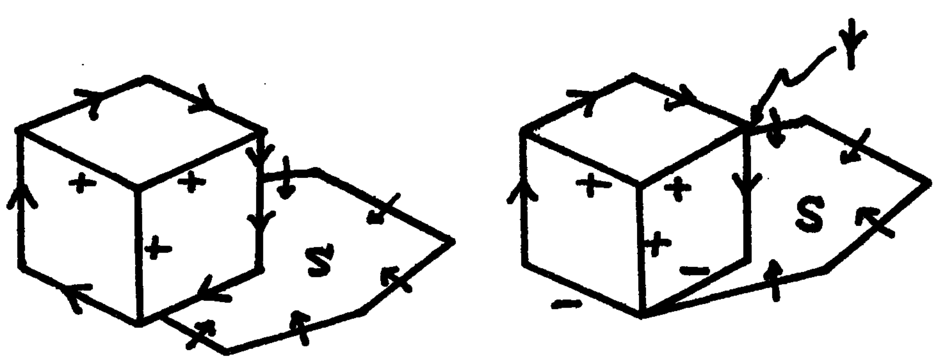
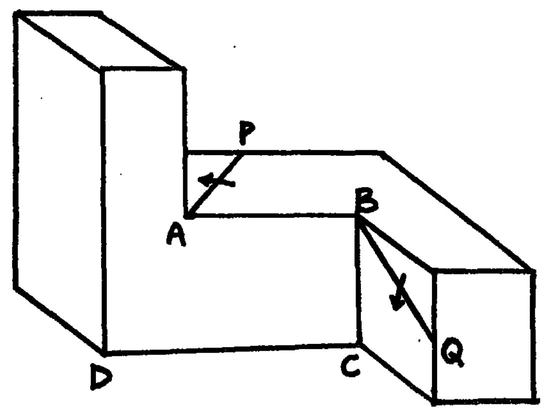


Figure 3 lists all the possible shadows which can arise under the same restrictions used in the VIRGIN program for scenes with no cracks. The junctions labelled "concave" will only occur if non-convex objects or scenes with cracks are allowed. There are also several cases where light placement will cause some junctions involving shadow lines to be TEEs which would otherwise be forks or arrows, but these are for the time being classed as pathological since small changes in lighting angle will change them to the cases listed.

Although we do not yet know how to determine (or specify) the position of a light source relative to a scene the assumption that a scene is illuminated by a single light source provides useful information. Many junction types from Figure 3 (all the FORKS, ARROWS, PEAKS and MULTIS) represent scene vertices which are actually casting shadows (remember that 'nasty coincidences' of shadow edges with scene vertices are excluded) The simple and obvious theorem that a given shadow edge cannot be cast by two different scene edges enables us to exclude all parsings where a 'shadow ' line links two junctions of the above types.

A similar argument allows us to exclude all parsings like that shown in Figure 11.

FIGURE 11



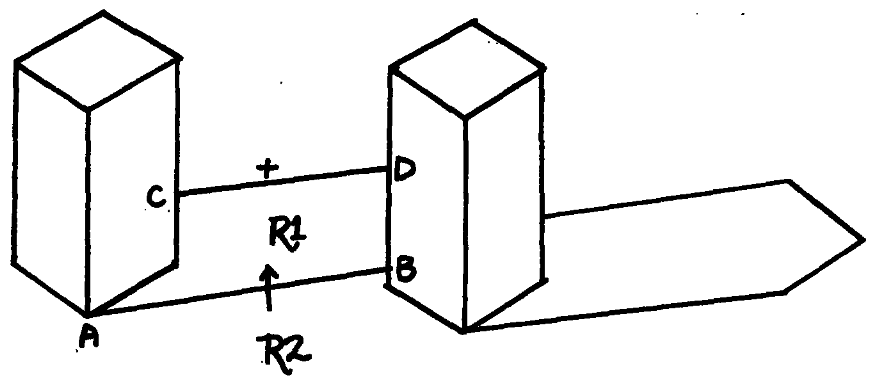
since if BQ is to be a shadow edge the light source must be behind the plane given by ABCD; thus it cannot cast a shadow such as AP on a plane behind ABCD.

This kind of constraint will be more difficult to include in the parsing program. It does, however, give an indication of how the position of the light source illuminating a scene will have to be specified to be of use in parsing scenes.

Two rather more local theorems about shadow edges allow the number of posssible parsings of a given picture to be further reduced. The lines bounding a region cannot receive interpretations which indicate that both the region and an adjacent region are shadow regions.
Or: A region cannot be labeled with a shadow arrow pointing both into and out of it. This follows obviously from the 'single light source' constraint.

The second theorem excludes interpretations such as that shown in figure 12.

FIGURE 12



The 'shadow' labeling on line AB indicates that the two regions, R1 and R2, bounding it represent part of the same surface and thus are coplanar; the '+' label on CD indicates that R1 and R2 represent non-coplanar surfaces. This argument, of course, excludes '-' or 'arrow' labelings of CD also. We are convinced that CD cannot be labeled as a 'crack' either, but do not see how to prove this yet.

Another constraint results from the following observation: If we are certain that a region is a shadow region, then all junctions of type 11 (see figure 3) bordering the region can be labelled in only one way.

(The proof depends on the completeness of the listing of junctions. We do not attempt to prove this completeness here.)

PROOF

Suppose that we have a junction as shown in figure 6 such that the shaded region is known to be a shadow. Then in any labelling

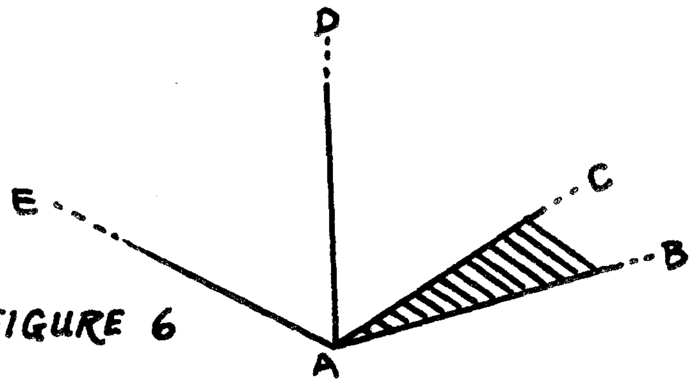


FIGURE 6

FIGURE 6A

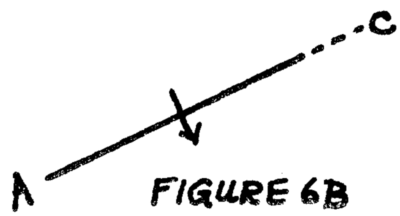


FIGURE 6B

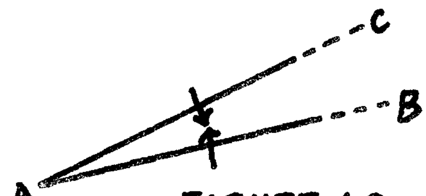


FIGURE 6C

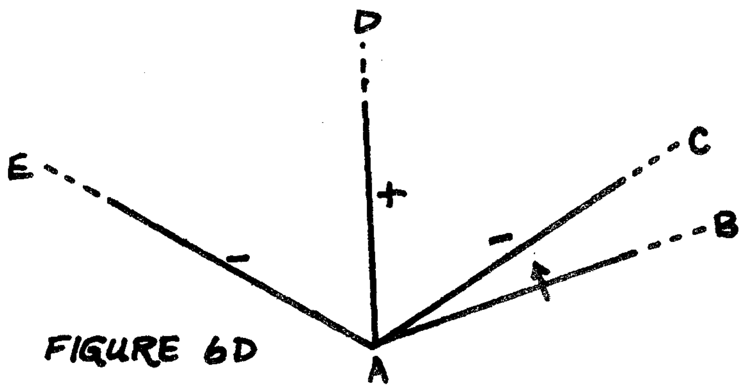


FIGURE 6D

either AB must be labelled as in figure 6A or AC must be labelled as in figure 6B or both AB and AC must be labelled as in figure 6C. However there is no peak which has two branches which are both labelled as shadow edges, which eliminates 6C. Similarly, there is no peak with a shadow edge as line AC and with the shadow in the region CAB, which eliminates 6B. Therefore the junction must be labeled as shown in 6D.

Another fact which is probably useful is that all junctions of type 1 except for one rather unlikely case can be labelled safely. The exception is shown in figure 7. This exception can also have a real physical interpretation as an L-shaped block with a shadow behind it. In any event, there are only two possible labellings.

Similarly, whenever we have a junction such as the one shown in figure 8, we can also label it with relative safety. Figure 9 shows the only possible exception; the lighting must be from a low enough angle to cause the shaded area to be called a shadow. This is unlikely, because the area behind the object would almost certainly be darker than the top of the object, in which case the top of the object would not be listed as a definite shadow region.

We suspect that there are a considerable number of such cases which would at the least produce an ordering on the likelihood of each possible labelling.

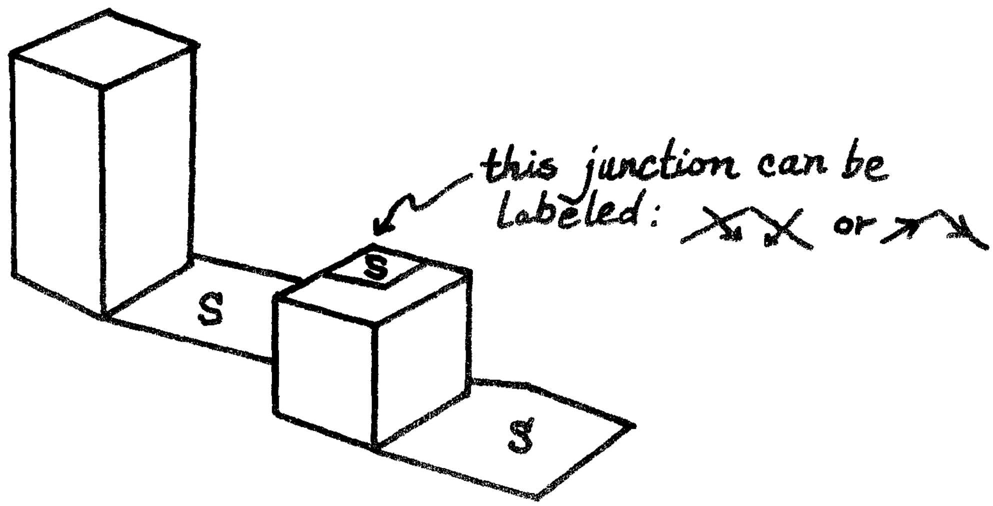


FIGURE 7

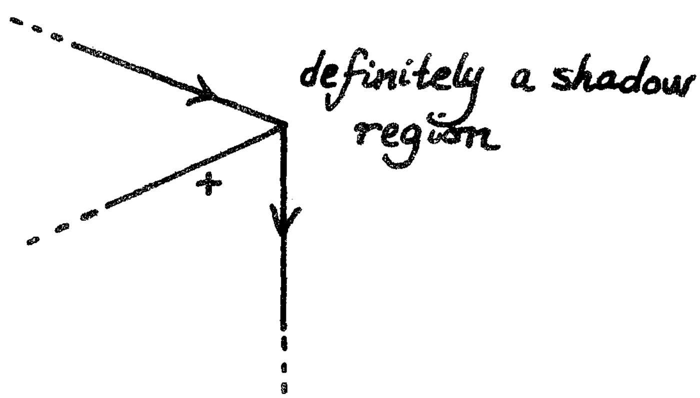


FIGURE 8

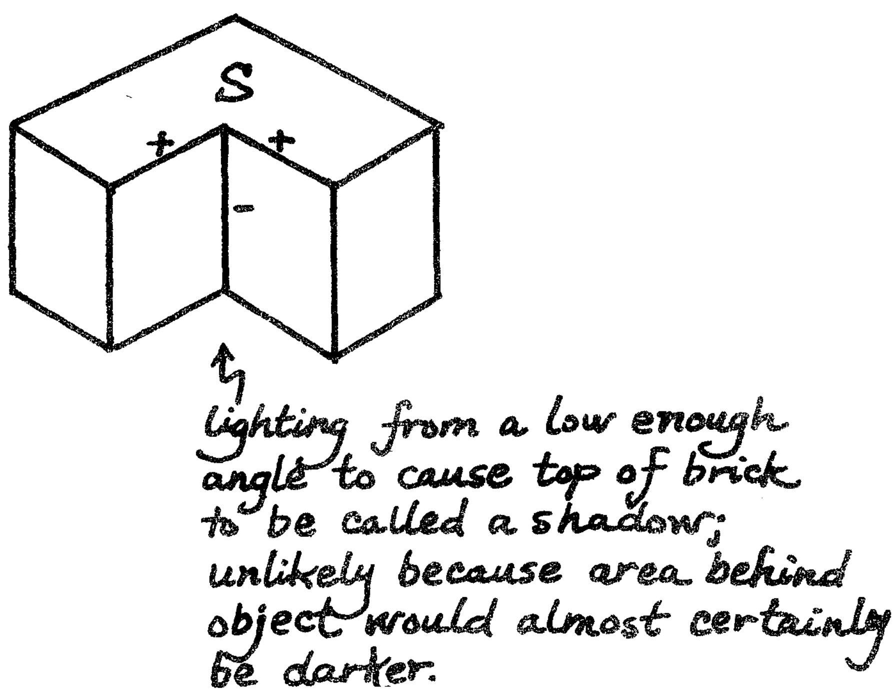
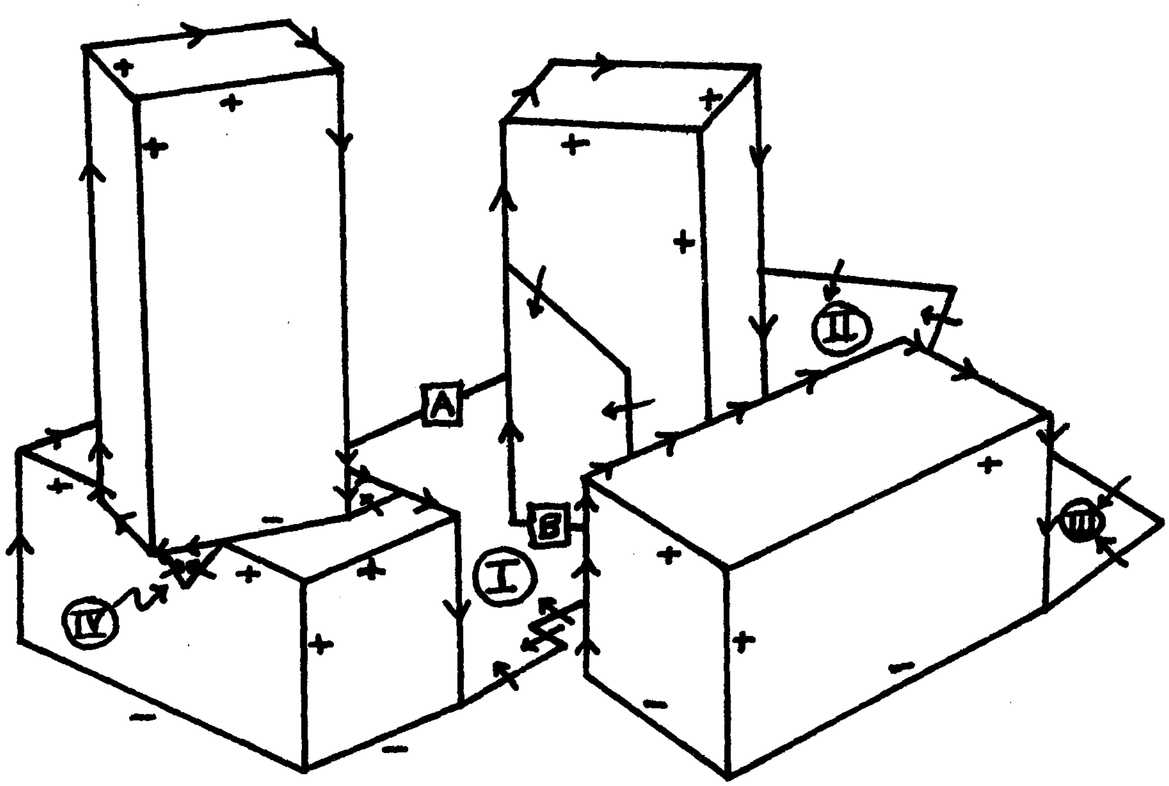


FIGURE 9

It has already been suggested that the number of possible parsings of a scene will be reduced if shadows are present. Basically, without shadows there is always an ambiguity between the cases where an object is actually supported by or part of another object or surface, and cases where an object is suspended above the other object or surface. This means that there are usually 4 possible parsings for each object or 4 exponent n parsings for a scene with n objects. This is a serious problem, since we must resort to identifying "bottom" edges or use other tricky means to eliminate these parsings. When shadows are present, most of these spurious parsings are impossible. (See figures 5A and B)

Figure 10 shows the 10 possible parsings for a rather complicated scene, with no assumptions made about lighting direction. (These assumptions merely prune the search tree, except in rare cases.) Regions marked I, II, III, and IV were assumed to be shadow regions, as their intensity levels were lower than all surrounding regions. This eliminated the possible interpretation of region II as a hidden object under the assumption that the case shown in figure 7 is very unlikely. Notice that a heuristic to label line [A] properly would reduce the number of parsings to two. (Such an heuristic might be "If a line can take any of the five possible labelings, label it a shadow with the arrow pointing to the darker side.")

FIGURE 10



[A] can be labeled with any of the edge types.

[B] can be $\overline{}$ or ←.

If no shadows were in the scene, there would be 48 allowable parsings instead of 10.

Other possibilities

In all the preceeding sections scenes have been treated as if the only way to identify shadows and shadow lines was to pick a region darker than all those surrounding it. If there were a reliable method of identifying shadow lines directly, then these could be labelled directly as well. It is quite likely that shadow lines can be directly identified. (They are "wider" and more jagged than edge lines, for example.)

It is also likely that particular junctions can be ruled out on the basis of relative intensities around the junction, even if shadows are not present at the junction. We have not yet considered this possibility systematically.

Ramifications

On the basis of the results so far, We can make some suggestions as to how these results can be integrated into a vision system. If there are junctions which can be labelled with certainty, then it would be desirable to label all such junctions prior to making arbitrary assignments of labellings, since the number of arbitrary decisions and consequently the number of back-ups will thus be considerably lessened. Therefore we suspect that a good stategy would involve looking at shadow regions first.

Cracks

Extending the algorithm to deal with cracks provides fewer explicit heuristics than the shadow case dealt with above. Some, at least tentative, identification of lines as representing cracks may be made at preprocessor level. Coplanar surfaces in a scene tend to reflect similar amounts of light particularly if they are adjacent; thus a line dividing two regions of similar brightness is a good candidate for a crack. Other very local optical features of cracks dividing bodies may help to identify them.

Some junction types (PSIs and EXEs for example) are almost unique to scenes involving cracks. The parsing tree can be minimised by starting the parsing process at these kinds of junction. ('Start with the junction types which have the least number of interpretations' is a good general heuristic, of course.)

Within a complete vision system higher level information may point to the existence - or nonexistence - of cracks. A scene where the number of bodies is known in advance to be equal to the number of isolated subpictures it yields can have no cracks; cracks may be discovered when an attempt is made to pick up something previously identified as a single body.

Implementation

The VIRGIN program, as noted above, can be extended to include the junction labelings of figures 3 and 4 merely by adding additional, appropriate, junction theorems. It will, without modification, accept assertions about the interpretation of specific lines. These assertions can be made tentative by arranging that they are erased if no parsing is possible with them or if failure backs up to them for any other reason. They could be ordered in decreasing order of confidence so that the most certain was the last to be erased or interdependencies of any complexity set up.

Trivial modifications to the basic program would ensure that the parsing of the whole picture and of each region started at the most 'favourable' junctions (i.e. those with fewest interpretations; see above)

Incorporating information about lighting direction will be particularly interesting as it requires mechanisms which will be generally applicable to using higher level information to direct the parsing process. It can be accomplished by extensive use of recommendations in goal statements so that the choice of junction theorems will be restricted by which theorems have been applied previously.

Once these mechanisms have been set up they will allow a priori assertions of global properties of the scene as in the case of specific assertions about labels.

Some of the other heuristics like 'two vertices cannot cast the same shadow edge' and 'no shadow arrows both into and out of the same region' will be easy to incorporate in the program and we hope to do this quite soon. Others will be more difficult and may have to wait until we understand them better.

Future Work

We hope to have an extended version of VIRGIN, incorporating a subset of the 'crack' and 'shadow' labels together with some of the heuristics described above running in the near future. This will gradually be extended to include the full set of junction labels given in figures 3 and 4 and more sophisticated heuristics so long as preliminary results seem to justify continuing the approach.

Meanwhile we will go on looking at the theoretical basis of the work described above in the continuing hope that a better theoretical understanding of what things look like will lead to better vision programs.

References

Clowes M.B. On Seeing Things
A.I. Journal 1971

Huffman D.A. Impossible Objects as Nonsense Sentences
M.I.6 1970

Dowson M. What Corners Look Like
Vision Flash 13