AN HYPOTHE$!S-DR!VEN VISION SYSTEM

Michael R Dunlavey
MIT Artificial Intelligence Laboratory
Cambridge. Massachusetts

Abstract

This paper presents a small vision system created to study Minsky's theory of Frame Systems It is organized around a detailed semantic hypothetical model of the scene, which is capable of being structurally altered and adjusted to fit the visual data. Visual data is gathered through a window, whose position is controlled by suggestions which arise from uncertain parameters of the hypothetical model

Introduction

Minsky's theory of Frame Systems (3) offers strong advantages for a theory of perception, and this system attempts to study it. It does not need to see the whole scene, but only fragments of it, and occlusion is no particular problem. At all times there is a detailed semantic model of the scene which, although it may be different from the final model, is at least consistent The rapport between the high-level semantic model and the low-level visual data is very close, without intervening layers of computation

When a hypothesis is wrong, it is not discarded but is transformed in an orderly way into a more correct hypothesis, avoiding wasted effort. One tenet of the frame theory is that one may use dozens of different models linked in this way. The space of models that our system can propose is limited to wedges, blocks, and pairs of aligned blocks because there is no intermediate level of representation between solids and vertices that the system can sense The more shapes that get added to the repertoire, the more difficult it is to find ways to distinguish among them on the basis of vertex appearance So a lesson for the future would be to use descriptions organized around the notions of axis and section, as originated in the work of Agin and Binford. See Holterbach (2) for a thorough discussion

For a detailed account of the structure and operation of the system, see the author's forthcoming dissertation.

System Outline

Input data, to keep the system simple, was restricted to the isolated vertices of a hand-coded line drawing. The data is examined by a program called the fovea, which reports only vertices contained within a movable diamond-shaped window. Each reported vertex consists of a point and the directions of its rays

The basic operation of the system is to take vertices, one by one, and note their correspondence with points in an internal model of the scene This process is punctuated by occasional modifications to the model to accomodate the data. Blocks appear out of nowhere, shrink and grow, sometimes split in two or turn into wedges When the system can see no more vertices, it moves its window, looking for the far corners of objects whose dimensions are yet unverified.

The internal model for each block or wedge is a large actor (that is, a data object together with a procedure for accessing it) containing parameters for all of its corner points, principal directions, and dimensions It thinks of itself as a complete 3-D model of the object except that the numbers stored in its points, directions, and dimensions are those of its 2-D isometric projection Each parameter has not only its value but an authority-flag, to record whether the value is known with any certainty or is merely default Although the block contains over a dozen different parameters, it can be adjusted as a unit due to a complex system of constraint equations for adjusting default parameters

The process of matching vertices to points of each object is organized around two additional kinds of actors, vertex-matching-objects and vertex-matches Each block or wedge model contains a set of vertex-matching-objects, one for each point, which can compare a vertex to that point. If it finds a comparison, it constructs a vertex-match containing a comment about the quality or meaning of the comparison, such as "agree", "stretch" meaning the object should be stretched, "wedge" meaning it should be changed into a wedge, or "cut-after* meaning it should be divided in two. Since each vertex will usually be comparable to more than one point in the scene, the best vertex-match is chosen by sorting on the comment. The vertex-match also contains machinery to adjust or modify the model once it has been chosen If it is to make a large change in the hypothesis, it uses tabular transfer rules to guide it in setting up the parameters in the new hypothesis.

It does not insist that hidden vertices or rays be invisible, and sometimes they're not, if the block is really going to be a wedge.

Although space does not permit greater detail, here is an outline of the contents of each type of actor:

```
Hypothesis
    Parameters
        Value
        Authority-Flag
        Immediate Hard Link
    Parameter Names - Fixed
    Parameter Names - Variable
    Rotation Rules
    Current Transformation
    Constraint Equations
    Adjustment Mechanism
    Vertex-Matching-Objects
    Suggestion Generator

Vertex-Matching-Object
    Insides of Hypothesis
    Mapping From Point and Ray Names of the Vertex
        to Point and Direction Names in the Hypothesis
    Procedure for Comparing Against Visual Vertex
    Procedure for Generating the Comment
    Prototypical Vertex-Match

Vertex-Match
    Insides of Vertex-Matching-Object
    Comment
    Procedure to Carry Out Comment
        Insides of Second Hypothesis, if any
        Transfer Rules
```

Example - Perception of a Block-Pair

Operation of the system in perceiving an aligned pair of blocks is portrayed in Figure I. The initial fovea placement is provided by the user Vertices are processed singly in any order. The first vertex, since there is no hypothesis to accept it, causes a

new block hypothesis to be created, complete with default points, lengths, and directions The second vertex establishes its x length. At this point, the parameters for the block hypothesis are the following:

```
p    = point (3.00 2.00) nil
px   = point (3.50 2.50) nil
pxy  = point (2.25 2.50) nil
py   = point (1.75 2.00) nil
pz   = point (3.00 3.00) known
pxz  = point (3.50 3.50) known
pxyz = point (2.25 3.50) nil
pyz  = point (1.75 3.00) nil
pax  = direction 0.79 known
pax- = direction 3.93 known
pay  = direction 3.14 known
pay- = direction 0.00 known
paz  = direction 1.57 known
paz- = direction 4.71 known
plx  = length 0.71 known
ply  = length 1.25 nil
plz  = length 1.00 nil
```

The first vertex established point *pz* and all the directions, and caused all the remaining points to be adjusted consistent with the first point, the directions, and the default lengths The second vertex established point pxz, causing three other points and the x length to be adjusted. The authority-flag of the x length became "known" because the two end points were known

The third vertex matches against the same point as the second vertex, but with the comment "row-after**. A second block is hypothesized, al:gned with the first and butting up against it The fourth vertex establishes us x length and point pz. By proposing such pairs of blocks, we avoid the problem of hypothesized blocks intersecting in space. Of course we can't completely prevent it, but neither can people.

Having exhausted the visible vertices, the system asks all block or wedge hypotheses to generate suggestions. In the first block, since the y length is unknown, suggestions are generated to look for points pxyz. pyz, pxy, and py, which would establish that length. Suggested locations to look for them are based on the default value of the y length and perturbations of that value. A suggestion is taken, the fovea moved, and more vertices reported. The fifth vertex establishes point pxyz. The hypothesis is adjusted and the y length and point pyz become known. Due to global soft equality links placed between the respective y and z lengths of the two blocks, the second block is also stretched and adjusted. At this point, the parameters for the two blocks are as follows:

	First Block	Second Block
p	= point (3.00 2.00) nil	(3.75 2.75) nil
px	= point (3.50 2.50) nil	(4.25 3.25) nil
pxy	= point (1.50 2.50) nil	(2.25 3.25) nil
py	= point (1.00 2.00) nil	(1.75 2.75) nil
pz	= point (3.00 3.00) known	(3.75 3.75) known
pxz	= point (3.50 3.50) known	(4.25 4.25) known
pxyz	= point (1.50 3.50) known	(2.25 4.25) nil
pyz	= point (1.00 3.00) known	(1.75 3.75) nil
pax	= direction 0.79 known	0.79 known
pax-	= direction 3.93 known	3.93 known
pay	= direction 3.14 known	3.14 known
pay-	= direction 0.00 known	0.00 known
paz	= direction 1.57 known	1.57 known
paz-	= direction 4.71 known	4.71 known
plx	= length 0.71 known	0.71 known
ply	= length 2.00 known	<-> 2.00 nil
plz	= length 1.00 nil	<-> 1.00 nil

After processing the remaining vertices, all of which agree (except for the T vertex, which is ignored), the uncertain z length generates another suggestion, which is taken, and the scene is finished.

Related Work

This system is most similar in spirit to those of Roberts (4) and Crape (1), in that the problem is to find a model that can be verified using fragmentary data.

Roberts' system was the pioneering work in machine perception of solids, and our system studies only a subset of his domain. His work was not extendible because his description of each object consisted of a homogeneous transform of a prototype solid. It could only make analog, not structural adjustments to its hypotheses, nor could it represent qualitative constraints, nor could it represent imagined intermediate results or use them to guide its processing.

Grape's is a more serious system than ours, and he has solved technical problems associated with using real image data. However, we are both addressing the same organizational issues and our system embodies significantly different answers. His system relies on the notion of standard views of objects rather than on whole objects. A standard view is recognized on the basis of a signature (my word) of connected line segments and associated vertices In other words, a signature must be a large enough subset of a standard view to distinguish it from other standard views, because a mistaken choice cannot be changed. This is not a picayune objection, but is characteristic of the content-free syntactic approach to perception. To make such a system see more types of objects, one must assemble a catalogue of subsets of standard views which grows much faster than the catalogue of object types

Our system differs from Grape's in that the model is much closer to being an actual 3-D model of the scene, stating what kinds of objects are present, simple relations among them, and their precise dimensions in the image (despite occlusion). It can be more bold about hypothesizing the identity of objects on vertex evidence alone because there is no penalty for guessing wrong, as the hypothesis can be changed.

All three systems suffer from a poorly chosen vocabulary of basic models, namely blocks and wedges. Without going into detail, the problem is that these objects are effectively "defined" in terms of their lines and vertices, and these definitions are too complex The addition of more basic shapes so defined would make it more difficult for a perception system to discriminate among them, even if it is able to change its mind. A solution for this, as previously mentioned, is to use descriptions organized around the notions of axis and section (2).

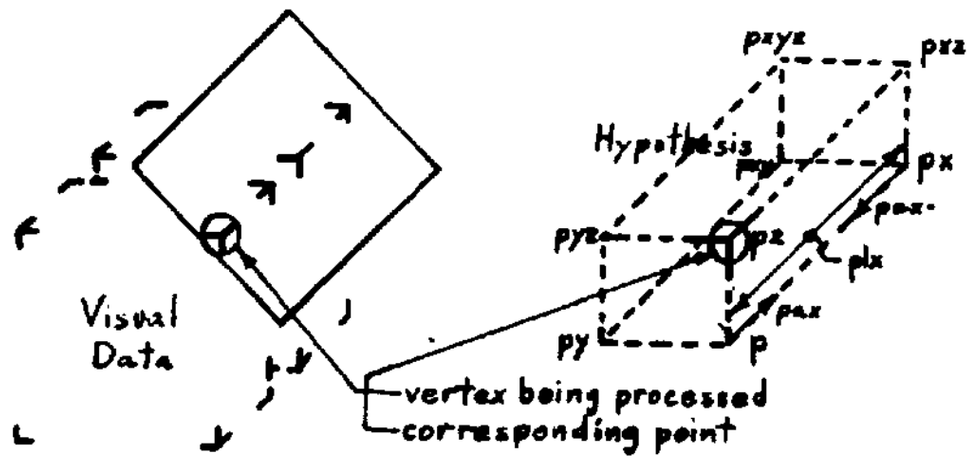
References

1. Grape, G. R., Model-Based (Intermediate-Level) Computer Vision., Stanford Artificial Intelligence Lab., Memo AIM-201, May 1973.

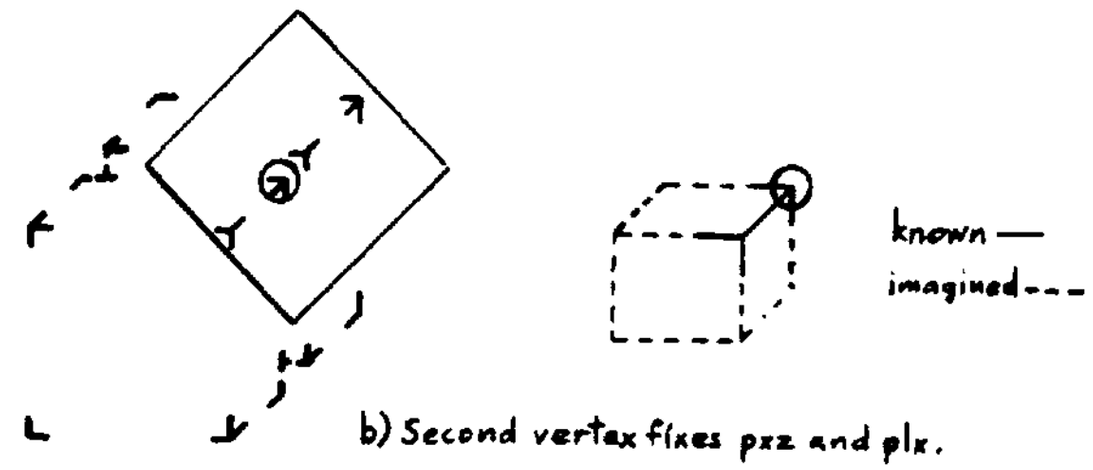
2. Hollerbach, J. M., Hierarchical Shape Description of Objects by Selection and Modification of Prototypes., MS Thesis, Electrical Engineering Department, Massachusetts Institute of Technology, June 1974.

3. Minsky, M. L., A Framework For Representing Knowledge., MIT Artificial Intelligence Lab. AI Memo 306, June 1974.

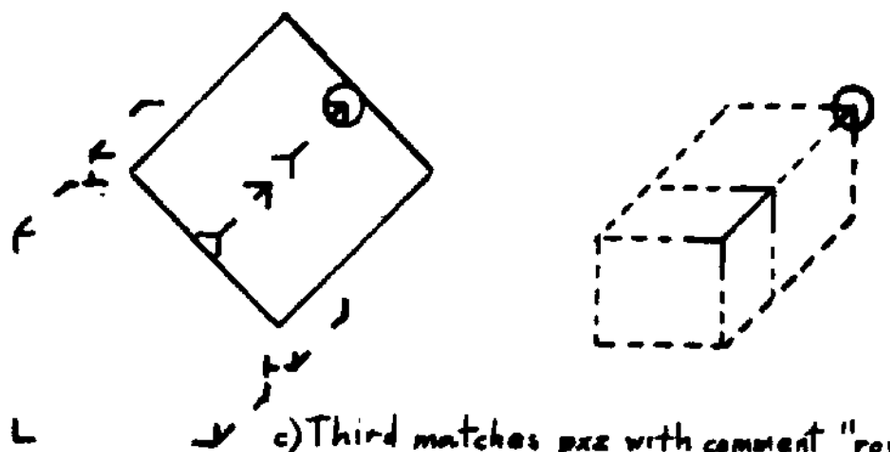
4. Roberts, L. G., Machine Perception of Three-dimensional Solids., in Optical and Electrooptical Information Processing, Tippett, J. T., et al (ed.). MIT Press, 1965.



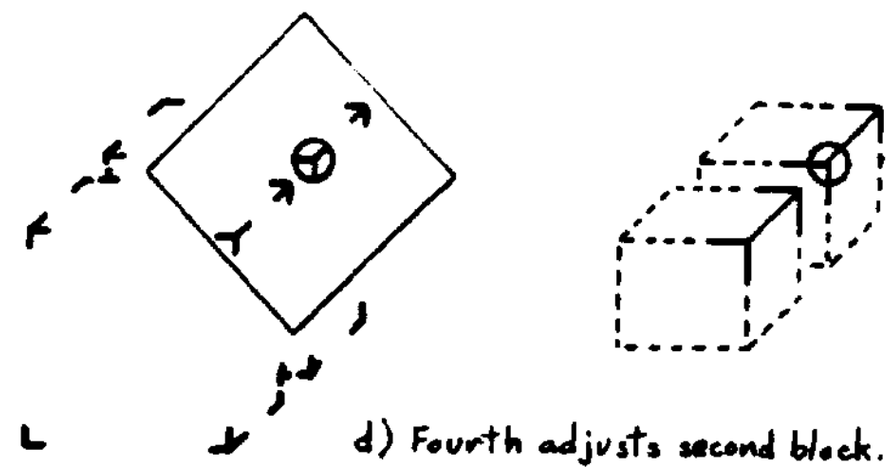
a) First vertex triggers a block hypothesis.



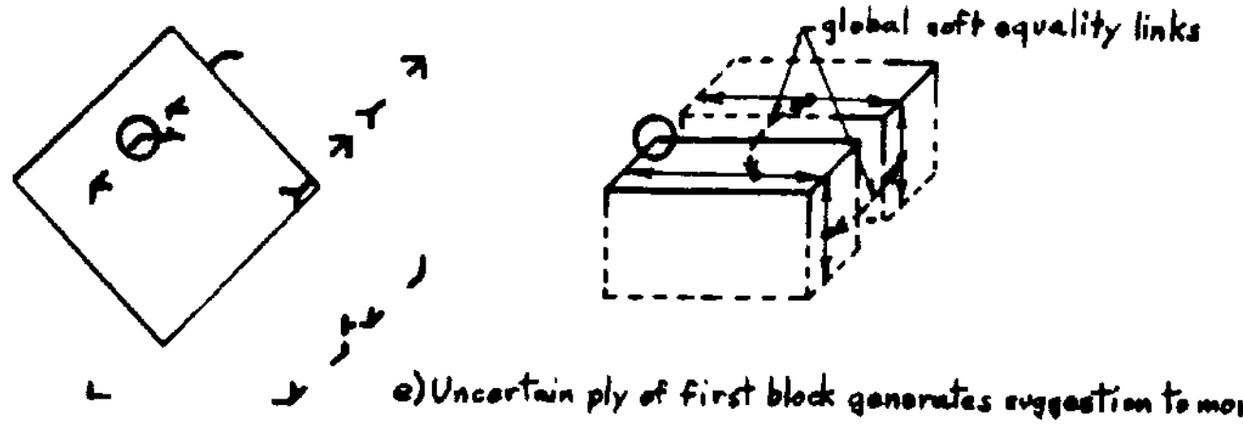
b) Second vertex fixes pxz and plx.



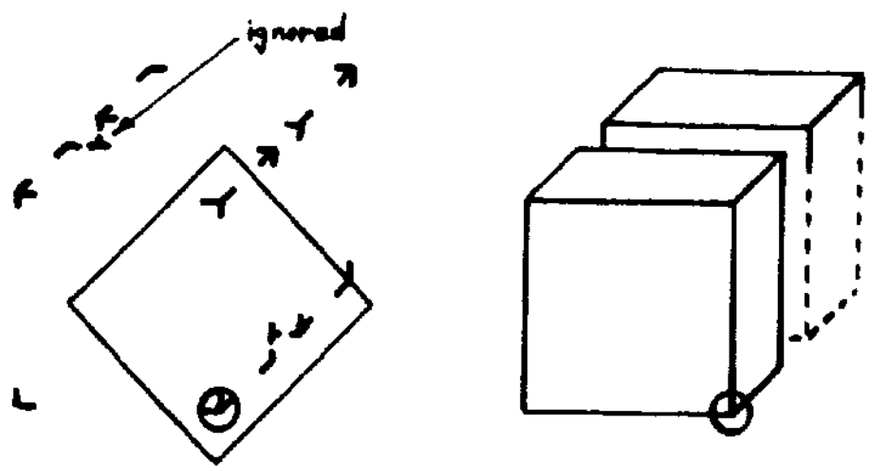
c) Third matches pxz with comment "row-after", triggering second block aligned with first.



d) Fourth adjusts second block.



e) Uncertain ply of first block generates suggestion to move fovea. Fifth vertex adjusts both blocks.



f) After three more vertices, (⊥ is ignored) fovea moves again, finding plz, and so on.

Figure 1. Visual data and hypotheses during perception of a block-pair.