"RECOGNITION CONES," AND SOME TEST RESULTS; THE IMMINENT ARRIVAL OF WELL-STRUCTURED PARALLEL-SERIAL COMPUTERS; POSITIONS, AND POSITIONS ON POSITIONS

by

Leonard Uhr

Computer Sciences Technical Report #310

December 1977

"RECOGNITION CONES," AND SOME TEST RESULTS; THE IMMINENT ARRIVAL OF WELL-STRUCTURED PARALLEL-SERIAL COMPUTERS; POSITIONS, AND POSITIONS ON POSITIONS*

by

Leonard Uhr

Computer Sciences Technical Report #310

* To be published in "Machine Vision," A Hanson and E. Riseman, Editors, Academic Press, in press.

The set of "position questions" posed the participants in this volume are presented in the Appendix. The discussion in the first section of this paper refers to them, by number.

"Recognition Cones," and Some Test Results; The Imminent Arrival of Well-Structured Parallel-Serial Computers; Positions, and Positions on Positions*

Leonard Uhr

Computer Sciences Department, University of Wisconsin

"Science must debate all admissible hypotheses in order to obtain a complete survey of all possible attempts at explanation....But it is unworthy of a thinker who claims to be scientific to forget the hypothetical origin of his propositions. The arrogance and vehemence with which such concealed hypotheses are defended are the usual result of the feeling of dissatisfaction which their champion harbors in the secret depths of his conscience about the justification of his cause." H. v. Helmholtz (the father of perception as a science).

Overview

1) This paper first presents a set of positions (on the issues posed for this volume) that argue for probabilistic parallel-serial variable resolution perceptual systems that can make use of large amounts of diverse contextually interrelated information, in a dynamic flow of mixed environment-driven (bottom-up) and internally driven (top-down) flow of processes. It further presents the position (on positions) that the open-minded application of scientific method by the whole community of scientists, attacking such a complex and difficult task as perception, is necessary, to insure as great as possible a diversity of considered approaches.

2) The imminent arrival of true hardware-embodied parallel-serial computers is briefly discussed, and the suggestion made that many of today's perceptual systems will gain orders of magnitude increases in speed and power when their highly iterated operations are coded for highly parallel computers built with arrays of 10^4 to 10^8 programmable processors.

3) Finally, the author's "recognition cone" systems are described, as an example of the parallel-serial probabilistic approach being suggested. A basic system for multi-layer recognition and description of a scene has been coded in EASEy (a variant of Snobol), Fortran, and Simula, and a few test results have recently been obtained. Extensions and variations have been coded in EASEy, to handle continuing scenes of objects that move about and change over time, binocular (and multi-modal) perception, learning-by-discovery, and perception as embedded in a larger cognitive system (called "See-err").

* This research has been supported by NSF Grant MCS76-07333 and by the University of Wisconsin Graduate School.

* To be published in "Machine Vision," A. Hanson and E. Riseman, Editors, Academic Press, in press.

Section 1. Positions, and Positions on Positions

This paper tries to do three different things: A) Positions are argued on some of the basic issues of perceptual models that have been posed this group. B) The final coming of very large arrays of (10^4 and, potentially, more) processors whose hardware is truely parallel, and the enormous potential power this promises for many perceptual processes, are discussed. C) The author's "recognition cones," for probabilistic parallel-serial perception, are described, as examples of the kind of architecture being proposed, and some test results are presented.

Introduction

All the participants in this volume were asked to state their positions on a number of key questions about how we should go about developing perceptual systems. It was suggested that I use my answers in this volume, I suspect partly because they express a rather extreme viewpoint, at least when compared with the mainstream approaches being taken today.

This field has seen several major swings, first to overly parallel systems (Rosenblatt's "perceptrons") and then to overly serial systems (Minsky and Papert's "serial algorithms"). I think the pendulum has begun to swing again. If I seem to be attacking a straw man, fine - that means things are becoming better faster than I had judged.

The Free Choice of Science

But far more important than any specific position that I (or anybody else) could possibly take is the following: All approaches that reasonable, thoughtful and knowledgeable people find promising should therefore be tried. No individual or grant-giving agency knows what the "truth" is, what the "correct" approach is - especially in such a complex and strange new field as this. People should listen to and try to build upon one another. But we should resist the impulse to propagandize and dogmatize mere opinions, or turn them into schools. We have already seen too many bandwagons, too many false hopes.

The scientific method means, above all, the free choice by each expert scientist of the problems, approaches and techniques that she or he has

judged to hold the most promise. Thus individuals should spread themselves over the promising alternatives (those that one individual finds underexplored becoming for him especially promising in their likely pay-offs; those that many individuals find promising being more vigorously pursued).

This is the way for the whole community of science to advance as fast as possible. Each individual should realize the strong likelihood that he has chosen the wrong approach. This the failures as well as the successes play their vital role, and all participate in any success. And far more success can be achieved than if people expend effort trying to make everything look like success.

A Discussion of the Position Questions

I. The most important answer to the questions posed for this volume is the following simple and rather obvious set of observations:

A. Nobody knows today what the right approach will be. We are engaged in the empirical enterprise of designing by far the most complex and highly structured system man has ever attempted. All promising approaches should be pursued as vigorously as possible.

B. Worse, nobody even knows how to evaluate what is done. We still don't have a clear idea of our goals - e.g. what is a "description," or what would be a good "model for perception." We don't know how to compare the different dimensions of complexity - e.g. linear vs. non-linear transforms, depth, color, texture, motion, or the relative difficulties of straight vs. curved lines, or letters vs. chairs vs. faces vs. trees.

C. Ours is a most difficult problem. It has been a central preoccupation of psychologists and philosophers for thousands of years. For the first time we have a powerful enough tool in the large computer to grapple with it effectively. We are making much faster progress; but we should make clear and attack the difficulties, rather than evade them. There is little value in handling a particular suitably simple problem unless it can then be generalized (or is of real practical importance).

D. Learning is probably the key to the flexibility and general-purpose adaptability that is needed for a system that is able to recognize and describe changing scenes of unanticipated objects. Without learning, ad hoc programming can (cumbersomely) handle a few specific problems. But no system can be large enough to handle "all possible objects" (indeed this is a potentially infinite set, with new objects created all the time). Rather, a system must be able to learn about that subset of objects with which it must interact.

E. We have had arguments and proofs, that have often been falsely overgeneralized and misinterpreted, about the limitations of either of the extremes of, for example: strictly parallel vs. strictly serial; non-deterministic vs. deterministic; perceptual characterizing vs. heuristic problem-solving; pre-programming vs. learning; multivariate vs. syntactic vs. framed; bottom-up vs. top-down. Often a straw-man extreme is set up; often the supposedly opposing approaches boil down to much the same thing; often a judicious golden mean can combine the several approaches to exploit the strengths and eliminate the weaknesses of each.

II. To answer or respond to some of the questions posed:

1. There can be no "primary basis for segmentation" (unless we restrict the set of scenes the program is asked to handle to scenes that always contain the information (e.g. perfect contours) with which the program has been designed to segment). One object will have clean and sharp edges, but internal regions that vary in color, texture, intensity, etc. in wild and arbitrary ways (e.g. Hawaiian Mu-Mus). Another object will have homogeneous regions but no edges; or dotted, fragmented or foggy - essentially vague - edges (e.g. Chinese landscapes). Therefore perception systems must look for both regions and edges--and expect not to find them. They must be able to make do with fragmentary information, combined from diverse sources. And higher-level features are probably crucial: many objects can be segmented only after they have been perceived. Perception infers and constructs the object; segmentation can then be used to re-cover non-existent boundaries.

2. The more information the easier the segmentation process - if the system is capable of combining information from many diverse and unanticipated sources. Color can be especially helpful in giving qualitatively different regions between which edges can relatively easily be drawn. The motion of an object against a background is probably even more helpful, potentially (but this poses very difficult problems), since the perceiver now has a sequence of very slowly changing images from which to construct the object. Striking textures are helpful, like color; but many textures are quite subtle and variable. Perception of objects that move seems to me the most important problem still to be attacked. Once we achieve such systems we will be able to make powerful use of the additional information they can gather.

4. Parallel-serial systems are obviously needed, rather than either parallel or serial systems. Minsky and Papert devastated strictly parallel systems (1-layer "perceptrons"), showing their explosive inefficiency at handling global properties like parity or connectivity. Long before that Selfridge had pointed out that strictly serial systems are only as good as their weakest links. And attempts to use serial techniques for real-world patterns seem to develop intolerably long chains of tests. Parallel takes space; serial takes time.

But these are both straw-man extremes. By combining serial and parallel processes we can minimize time and space and gain the virtues of both. For example, to compute parity (that is, whether there is an odd or even number of 1s) over an array of 0s and 1s, a completely parallel "1-layer perceptron" takes only one moment of time, but needs 2^N elements, each with N connections (N = the Number of cells in the array) (Minsky and Papert, 1969). They propose instead a very simple serial algorithm, that looks at and counts (modulo 2) each cell in the array, and therefore takes N (times a small constant) moments of time. But a parallel-serial set of simple binary exclusive-or operators, where the first layer looked at the array, a second layer looked at the output array of the first layer, and so on, needs only N - 1 elements, each with only 2 connections, and takes only $\log_2 N$ moments of time! Thus both space and time requirements are brought down to acceptable

small numbers. Input to a visual system, or to any sensor that is gathering information about an ed-vironment, in highly parallel, and it seems foolish to insist upon handling it in a serial manner. A serial approach usually has a problem-solving flavor; parallel-serial approaches force us to a more probabilistic combining of diverse sources of information - I think giving more the flavor of living perceptual systems and of semantic contextual interactions. They also give great efficiency, robustness over unanticipated distortions, and economies in speed.

5.-6. The human eye - like the computer - can get a global property only by successively compounding the local intensities and gradients sensed by its retinal receptors. It may well be useful to "glance" first at the higher-level things found, and then dip down to details, as needed. But if this "glance" entails computing lower-level partial functions in order to achieve these higher-level things it seems wasteful not to use them, to have them interact with "top-down" semantic guidance. Again, all levels are obviously needed. For example, grey-level gradients sometimes can be resolved into contours only after the highest semantic levels have resolved vague objects so that the level and type of fogginess can be assessed, and used to change the parameters for low-level edging and contour detection.

7. We can always construct counter-example-like problems that cannot be segmented at the retinal level. Therefore there must be an inner-directed top-down component to perception, one that allows higher-level characteristics, and "what the system expects and is looking for" to influence segmentation. This entails a judicious combining of both inner- and outer-directed processes. For example two eyes, one nostril and one lip can be enough to imply a face, which then segments cheeks, chin, etc.

We do not know what "primitive features" are necessary for vision. Attneave and Arnoult, Hubel and Wiesel, and others have suggested primitives that seem buildable into larger wholes. But the only way we can tell whether they work (and, more important, how efficiently, generally and powerfully) is to build a computer program around them, and test it.

Polyhedra recognition reduces the problem enormously, to straight-edge and regular-region detection. And many other scene-analysis systems similarly simplify to look for features that seem appropriate for recognition of a very small set of objects of special interest. This is fine if future expansion, to a wider variety of objects, is kept in mind, and effected. But too often ad hoc techniques are particularized to the simple problem, only to make it even more difficult to generalize.

For example, real-world objects can have almost any kind of complex (broken and fragmented) curve for their contours. They hardly ever have flat surfaces and straight edges.

11. Multiple sources of knowledge can be accessed and used when and if a serial program calls for them. Or they can be got by more or less independent processors or demons, and put in a common working memory for one another's use. But there are problems with such structures, and it seems best to impose a structure over the system that lets the flow of processes trigger needed processes, as appropriate. This needs techniques to merge and combine, and to choose among, what is found. The parallel-serial pyramid/cone systems attempt to do this, and the living visual system does this with celebrated success.

13. Objects like furniture, faces, or letters have a 2- or 3-dimensional structure that is far more complex than is the "syntactic" structure of words in a sentence. The basic problem of perception is to characterize that structure. Almost any system for reasonably difficult real-world problems does just that, whether by compounding and interrelating primitive features, or by looking for features that are themselves structural.

14.-16. Much current work seems to attack a specific domain in order to chop the problem down to size, and develop ad hoc partial solutions, without any thought, or hope, of future relaxation and generalization. This is fine when there is an important practical problem to be solved, and enough general understanding of our techniques so that a particular solution can be achieved with a reasonable amount of effort. But too often we are jollied into thinking that a problem (e.g. polyhedra) is practical, or generalizable, and solvable. Then, when it proves to be more difficult than originally claimed, ad hoc devices are scotch-taped on to give a few striking demonstration results, but also eliminating any possibility of generalizing.

A "general vision system" must be our long-term goal, since it is simply a part of science's age-old goal of building a theory of the intelligent mind/brain. This is not a 5 or 10 year goal; it will probably only be set back by crash-project approaches. Applications can help move toward that goal, but only if they are chosen to give us a variety of experiences, so that we can examine, compare, generalize, and improve upon the specific systems we develop to handle each application. An application of overriding practical importance (that cannot be handled well enough, or cheaply enough, by humans) should certainly be attacked for its own sake. But progress will be faster, and results will be more informative, if we develop a general body of knowledge that can be applied to each application.

15. In my opinion, parallel-serial systems that integrate characterizing transforms for a variety of kinds of information in a combined outer- and inner-directed flow of processing are the most attractive. The transforms must be capable of assessing configurations of more or less loosely interrelated characteristics (which are themselves configurations of characteristics, etc.). Weights and thresholds should be used, to give efficiency, and to allow for the many non-linear variations and defective information that will always be present in any but artifical toy scenes.

More important, no system could handle "all possible objects." Rather, we must develop techniques for discovery-learning of sets of features and higher-level characteristics sufficient for the environmental scenes that confront such a learning system.

Most important, as wide as possible a variety of approaches should be explored, compared and evaluated.

17.-18. One of our biggest problems is just the development of criteria for evaluating and comparing systems.. We don't even know what "des-

cription" means, or what are the basic dimensions of complexity. There are a potentially infinite number of possible objects, and of each object's parts, and momentary representations (think of all the faces, and noses, and expressions of Liv Ullman's face). How do we compare a system that can handle 26 printed vs. handwritten letters (by the same, vs. by different writer(s)) vs. 4 different polyhedra vs. chair-or-table, vs. chair-or-couch, vs. John-or-Teddy vs. John-or-Jackie? How compare scenes with only one object, vs. two or four, when they are separated, vs. touching or overlapping?

We have a rough idea of the dimensions, (e.g. 2-d, 3-d, time, intensity, color, texture, motion, linear transformation, noises, non-linear transformations). But it is too easy to find a problem that looks hard but turns out to be handleable - where we don't know whether that shows we have achieved an interesting system, or have shown the problem to be simpler than it appears.

But some steps can be taken immediately: Programs should be tested, and their performance compared to one another. Standard sets of test scene-instances would therefore be very helpful. They should fairly mirror the range of underlying problems, and should not be biased toward any particular approach. Tests should be made on scene-instances that the designer of the system has not seen himself, since that is the only way to guarantee that he has not built in overly specific ad hoc knowledge. (It is fine to build in knowledge. But the designer must be given an understanding of the whole set of scenes his system will be asked to handle, either by being given a good set of sample instances or, if possible, in some other way. But if somebody designs a program to handle one particular picture of a red house with evergreens, and tests his program on just that one picture, how can we judge the results?)

18. We don't know how "hard" vision is, or how long we must work. A good vision system must use contextual semantic knowledge of the most central sort, as and when it judges these appropriate. It must manipulate its world, construct its environment, and discover good concepts. So it needs a solution to the problems of "cognition" and "intelligence" as well. But we can certainly simplify. A full-blown vision system might need a $10{,}000^2$ retina, and the ability to describe scenes with hundreds of objects and thousands of qualities, features and peculiarities, from among millions of possible object-classes, each with billions of possible instances (I have just been describing a human being). But we can develop our systems using a far smaller retina (500^2? 100^2?) and far simpler scenes (of 5 or 10 objects) and possible object classes (1000? 100?) each with far fewer possible instances (10^6? 10^4?). So today's (or next year's) computers should be sufficient. But we must try to develop systems that can handle more objects once given either more built-in knowledge or learning experiences.

III. To recapitulate some of the issues:

I think we need non-deterministic parallel-serial systems that intimately integrate large amounts of contextually relevant information in a mixed inner- and outer-directed way, combining weights and making choices, trying to do as well as possible and expecting very defective and unanticipated scenes. They must learn, and be as flexible, general and adaptive as possible.

I know we need to encourage all promising approaches, develop good criteria for evaluating and comparing systems, achieve a much better conception of our intermediate and longer-term goals, and be more realistic in our expectations and promises.

Several things seem compelling about perceptual systems. Their purpose is to gather useful information about the external environment. They must be efficient, robust over wildly fragmented and distorted scenes and fast enough to respond in time. The cognitive system lies within a two-dimensional "skin" that can most efficiently be used by a set of parallel input sensors, whose information can best be combined and analyzed by sets of relatively local parallel transforms. This means the overwhelmingly complex functions computed by perceptual systems are best decomposed into successively simpler sets of functions, and all applied in a parallel-serial flow of processes. Some of these functions should be built in, but some of them must be learned.

Such a picture fits well with the structure of living systems, with their highly parallel sensory organs, and successive layers of parallel and relatively local synapses/transformations in the retina, lateral geniculate and cortical projection areas. But nature is not peculiar; rather, such systems evolved because they are eminently sensible. I know of no good arguments why any unnatural "artificial" device gives advantages. We can exploit the computer's "brute force" only if we cut our problems down to toy size - just as a British Museum algorithm (randomly examine every move!) can win at tic-tac-toe but never at chess.

Could a nature capable of evolving protein molecules, DNA and neurons be unable to develop magnetic memories? Did nature build the awesome complexity of a cell just to slow neurons down to millisecond speeds, rather than the nanosecond speeds already available in inorganic metals? This is not to say that we shouldn't try to develop non-natural techniques, but to remind how intelligent nature is.

Section 2: Parallel-Serial Hardware Structured for Perception Systems

1976 saw the coming of two new computer systems that connect relatively large parallel arrays of programmable processors (100^2 in Duff's CLIP 4, 1976, and 512^2 in Kruse's PICAP, 1976) to a small serial computer system. Each processor would be cumbersome as a general-purpose CPU, but is quite appropriate for a wide variety of perceptual operations. Kruse's PICAP (which is serial at the hardware level, but fast enough because of its built-in array processor to handle TV images in real time) has already been used for very interesting fingerprint and microscope slide recognition. And Cordella, Duff and Levialdi, 1976, have shown that for thinning, contour extraction, and several other basic picture processes Duff's CLIP 4 is many orders of magnitude faster than a serial computer. And such arrays are now cheap. The parallel array increases the cost of Duff's total mini-computer-based system less than 20%.

But these systems are only the beginning. For the technologies are rapidly improving, so that more can be packed on a single chip, and the cost

of chips continues to go down (and is even cheaper in larger quantities). Possibly even more important, there is a whole range of architectures for parallel-serial computers of this sort - ones with very large arrays, rather than the few dozens we see on systems like ILLIAC-4 - that are almost entirely unexplored.

Toward Parallel-Serial Architectures For Very Large Arrays

For example, a system with 5 or 10 arrays arranged in layers like the layers of pyramids and cones (see below) would literally be able to handle almost all of their processes in one fell sweep. New scenes could be input by a tv camera every 30 msec, or faster, so that real-time motions could be handled without any strain. Even better would be a large array (say 1200^2) that could be re-configured under program control, e.g. into one deep cone, or a binocular pair of slightly smaller cones.

A variety of configurations for networks of processors, and of arrays of processors, are now possible - if we knew what to do with them and how to program them. But for scene description we do know what to do with the large parallel arrays. And, as in this case, when parallel structures are appropriate, and are known, the savings can be enormously greater than what has become known as "Minsky's conjecture" - that concurrent processors speed things up only by $\log_2 N$ (N = Number of processors). On the contrary, for cone/pyramid structures they would speed things up by a factor of LN_r (where N_r = Number of processors in the largest (retinal) layer and L = the number of Layers). So a 10-Layer pyramid with a 1000^2 retina would be processed 10^7 times faster!

Systems that Would Benefit Greatly from Parallel-Serial Hardware

A number of researchers have been exploring layered parallel-serial variable-resolution structures for scene description, including Hanson and Riseman, 1974, 1976; Tanimoto, 1975, 1976; Levine, 1976; Douglass, 1977; Klinger and Dyer, 1974; Uhr, 1971, 1974; Uhr and Douglass, 1977. A variety of different strategies are being pursued, but all have in common a relatively well structured set of large numbers of parallel-serial operations and a regular flow of processes through this pyramid/cone structure. So these systems could literally gain 10^5 to 10^9 - fold in speed if they could be programmed for and run on a computer with hardware parallel arrays, one for each layer.

An increasing number of today's systems for scene description also seem to be moving in the direction of using successive stages of processes that are already conceived of as highly parallel, or would benefit greatly if re-structured into parallel form (as they almost certainly would be if the appropriate parallel-serial hardware were available). For example, the recent development of "spring-loaded" "relaxation" techniques, by Zucker, 1976; Davis and Rosenfeld, 1976; Tenenbaum and Barrow, 1976; and Fischler and Elschlager, 1973 use parallel and in some cases probabilistic operations that appear to speed-up and generalize earlier serial systems that resolve incompatibilities (e.g. Waltz, 1975; Guzman, 1968).

Still others are consciously exploring the values of parallel-serial processes (e.g. Waltz, 1977; Williams, 1976; Hannah, 1974; Zobrist, 1971; Zobrist and Thompson, 1975.)

Highly Parallel Aspects that are Present in Many Other Perceptual Systems

Finally, a number of systems that have not emphasized these aspects of their structure would appear to have important parallel and probabilistic components. These include, I think, some of the most interesting and powerful of today's perceptual systems, e.g. Ejiri, 1972; Sakai, Kanade and Ohta, 1976; Nagel, 1976; Mckee and Aggerwal, 1976; Moayer and Fu, 1976; Ohlander, 1975; Reddy, 1973; Lowerre, 1976. Such systems are usually described in terms of successive serial stages, or of a small set of demon-like processors that act concurrently. Superficially these may appear to demand a traditional serial computer, or, possibly, a multi-processor with 10 or 20 independent CPUs (and all the attendant headaches) interacting through one common working memory.

But usually when one looks at the details of the processes performed within each stage, or each demon, one finds very large amounts of parallel processing. For any scene description system must first process the very large array of parallel information input to the "retina." And almost always local averaging, differencing and edge detecting - all essentially parallel processes - go on at the early stages.

But even at the highest levels, the fact that the same objects might be found almost anywhere in the scene often entails large iterating loops that usually have an equivalent parallel algorithm. And the similarity-match of each of the alternative possible "models" or "descriptions" of the possible objects is often far more efficiently done by a parallel-serial sorting network of their partial features, much like a complex discrimination net, rather than by the serial matching of each possible model in turn (which becomes exponentially slow, and extremely complex e.g. in backtracking possibilities when "similarity" is assessed, as must be done for natural real-world objects) rather than the far simpler exact match that can be used for straight-edged polyhedra or similarly simplified sets of objects).

Section 3: "Recognition Cones" for Perception, and Some Results

The following describes the basic structure of the parallel-serial probabilistic "recognition cone" perceptual systems I have been developing. Extensions are briefly described that begin to handle motion and change, binocular and multi-modal perception, learning-by-discovery, and perception as embedded in and aided by the larger set of processes of a wholistic cognitive system. A series of layers of transforms, applied in parallel at each layer, successively extracts, compounds, coalesces and abstracts information. Each transform acts much like a simplified synapse that fires when enough of its specified pattern of inputs is present to exceed its threshold. A large number of transforms must be programmed into or learned by the system. Only a few preliminary sets of transforms have been used so far.

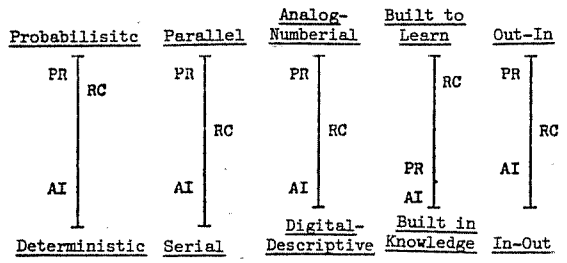


Figure 1. Recognition Cones (RC) compared with the (mythical) typical Artificial Intelligence (AI) systems for vision, and with typical Pattern Recognition (PR), on some of the basic dimensions and positions discussed above.

The Basic Perceptual System for a Single Static Scene

The basic model (Uhr, 1971; 1974 see Figure 2) inputs a sensed scene onto its "Retina" (an array of any size specified). A first internal

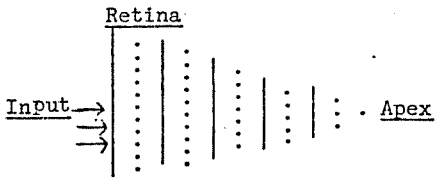


Figure 2. Recognition Cone Structure. Layers of transforms (dotted lines) look at and imply things into buffer stores, transforming from retina to apex.

layer of "transforms" looks at the cells of the retina. Each transform is located at a particular cell, and "looks at" any number of (usually neighboring) cells for specified information. When and if any transform succeeds (because the combined weights of the found parts exceeds its threshold) it merges those things that it implies into the corresponding cell of the internal buffer into which it fires. A single transform is thus, roughly like a synapse, with a number of neurons firing into it from the cells it "looks at" and a number of neurons firing out of it, sending the messages as to what it implies (when it fires) into the next layer.

Layering

Thus the first layer of transforms looks at the retinal input and fires into the first internal buffer, an array that now stores information and is treated exactly like the retina.

A second layer of transforms looks at this first internal buffer and merges its successfully implied things into the next internal buffer. This process continues, with a third, a fourth, and whatever number of layers of transforms desired (that is, specified for a particular run).

Convergence

Convergence occurs at each layer, as desired, by having a Row and Column* STEP-size specified, to indicate the amount of shrinkage from each layer to the next. E.g. the Retina might be set as a 16 by 36 array, and shrinkage factors used to divide this for convergence into the next layer occur as follows: 2 by 2 gives 8 by 18; 2 by 3 gives 4 by 6; 2 by 3 gives 2 by 2; 2 by 2 gives 1 by 1.

*Caps are used to refer to actual names used in the computer program.

Cone Structure of Layers of Parallel-Serial Transforms

Thus the system forms a parallel-serial structure of many layers of transforms sandwiched between buffer arrays that store information. The overall structure is that of a cone whose base is the retinal input buffer and whose apex is the final buffer that contains a single cell. Each layer of transforms operates in parallel (that is, in simulated parallel time, as programmed on the serial digital computer).

The Structure of Transforms

Each transform has the following form: A DESCRiption specifies the parts of the configuration that the transform is to look for. A "part" can be a specific thing, or a class, with an associated minimum acceptable value, a weight and relative location. (It can also be a procedure that computes any desired function that has been coded as a subroutine; but such procedures seem undesirable when modelling living visual systems, since they do not appear to be found in networks of neurons.) The transform looks for each part (as though neuronal processes were sending information about these parts to the transform-synapse) and combines the weights (which can be positive or negative) of those parts that are found (at the moment the combining function is simple addition or multiplication of weights, but any desired combining function can easily be inserted). If the combined weight exceeds the transform's THRESHold, the transform "fires," by MERGEing the things that it IMPLIES into the corresponding cell of the next layer internal buffer or one of the lists described below.

The Variety of Things, Names, and Structures that Transforms can Imply

This structure allows for a great enough variety of "things" that might be implied into the next layer so that the various processes that are usually handled by separate subroutines in most models of visual systems can be handled in a homogeneous way by this single general mechanism. Thus a transform can serve to "pre-process" in the sense of averaging out noise or differencing to enhance gradients and edges. Or it can imply internal things like "local horizontal edge" and then a next-layer transform can look for several of these things and build up bigger structures, like "long horizontal edge". Configurations can be got, like "horizontal connected to vertical" or "table-top." External names, like "A" or "FACE" or "TABLE" can also be implied. All this information is implied and merged back toward the apex of the cone.

A simple example of the structure we can give a cone follows: Transform layer 1 averages the intensity of light in each local region, or combines the primary colors. Layer 2 differences locally (e.g. looking at a 3 by 3 region, or if desired, a larger region), giving the center cell a high positive weight and the 8

surrounding cells low negative weights). At layer 3 around 5 or 10 transforms reside at each cell, each transform looking for a relatively simple local feature (e.g. Hubel-Wiesel-like local oriented edges; or local sets of cell values that the system has generated through prior learning). At layer 4 transforms look for spatially related configurations of several of these local edges (e.g. to build up longer edges, contour segments, and angles). At layer 5 transforms look for higher-level configurations of these configurations.

In addition to implying transformed values and internal names, transforms at any layer can also imply external names, ones that will be chosen among when the output - the assigned names or description - is chosen. For example, a long vertical can imply the letter "I" with a high weight, and the letter "E" and "TABLE" each with a lower weight. Then a transform that compounds the long vertical with a horizontal at the bottom going to the right can imply "E" with a high weight and negatively imply, to inhibit, "I" and "TABLE". A more peripheral transform that implies a short local vertical edge or gradient (which will be one of the input parts that allow a higher-level transform to successfully imply a long vertical edge) can also imply "I" or "E" - albeit with only a low weight. Thus each transform can imply a variety of things, including internal names and external names, of parts, wholes, wholes made up of wholes, and also of qualities (e.g. "STRAIGHT" or "DARK") and internal transformations (e.g. the cleaned-up or enhanced gradient).

Dynamically Implied Transforms and Things to Look For

A successful transform can imply new transforms to apply, and/or new things to search for (which in turn point to transforms that imply them), along with relative projected positions. This means that not all transforms need to be assigned to cell locations within the recognition cone, and applied to all input scenes. Rather, particular transforms can be called and used only when information gathered so far suggests that they might be useful. It seems hard to give a physiological interpretation to such dynamically implied transforms. But to the extent that a very large number of transforms is needed to handle all the different things that a general-purpose eye must perceive they can effect an enormous saving, since only the small sub-set needed for each particular scene need be used, rather than using all transforms on every scene. (Note that it is not clear how many transforms will be needed. But the parallel-serial structure that allows transforms to build up larger wholes at successive layers is designed for economy.)

Such dynamic transforms serve three important purposes:

1) They allow the system to "glance about" (at least with the "mind's eye") to look for additional characteristics that would confirm or deny tentative hunches. E.g. a vertical edge might imply "look for a horizontal edge near the top" or "look for an 'F'" which further implies "look for the transforms that imply 'F'" which will include this particular transform. Or "D" might imply "look for 'DOG'" which implies "look for 'O' a bit to the right".

2) They serve to give feedback loops to earlier layers of processing. E.g. "D" implies "look for 'DOG'" which implies "look for 'G' to the right" which implies "look for 'Vertical-curve' and 'Horizontal-edge'" (which will be found at earlier, more peripheral, layers).

3) They allow for quick and convenient shifts of attention, for the system can imply and roll in the transforms related to the particular type of things the system now infers it is viewing. E.g. when "chair" is implied it can imply "furniture," which then implies characterizers that imply other kinds of furniture. Thus a bottom-up flow triggers a top-down flow that merges in with it.

Dynamically Implied Triggers to Decide

A transform can also imply a trigger that a decision be made. The trigger can specify the particular class of objects among which the single most highly weighted is to be chosen. The cell where the transform trigger fires is then looked at and the things stored in it chosen among, the choice being transferred to a FOUND list. The cell at which the choice is made serves as the apex of a sub-cone whose base is a sub-region of the retina. This helps the system handle scenes of several objects, by assigning descriptive names to sub-regions.

A Summary of the Overall Structure of "Recognition Cones"

The overall structure, then, is a parallel-serial "cone" that uses successive parallel layers of probabilistic configurational transforms to process information input to a sensory retina. Convergence and merging of information implied by successfully applied transforms from one layer to the next gives the cone structure, from retinal base to a final layer with only one cell, the apex.

Each transform looks at a whole set of interrelated parts (things or processes), combines the weights of those that it finds, and fires if this combined weight exceeds its threshold, by merging the things it implies into the corresponding cell of the next internal buffer (or into the list of dynamically implied transforms or things to look for, or triggering a choice in that cell, putting that choice into the FOUND list).

The number of layers, steps of convergence, and particular transforms must be specified for a particular run (or, if learning is being investigated, some or all may be the result of prior learning experiences).

Handling Continuing Scenes of Moving and Changing Objects

The program for single static input scenes reinitializes itself for each new scene by erasing all of its temporary lists, including the retina, the apex, and the other internal buffers of the cone, The program for scenes of moving and changing objects (Uhr, 1976b) simply "fades" rather than erasing these lists. Fading means lowering slightly the weights of the things stored in them, and erasing a thing whose weight has been lowered below a specified minimum.

Now an implied thing is merged into a cell that may already contain that thing, because it was merged into it at some recent prior moment in time (but not at the same moment of time, since there

has not yet been time to place it there). The first time something is merged into a cell it is given an initial high weight. This serves to make new things salient, and also moving things, since their motion means that they will be merged for the first time into some local cell for which they are new things. Thus salience - in the sense of the size of the weight associated with a thing.- is a function of its newness (change) and motion. It is also, of course, a function of the transforms that imply it and, since transforms can imply other transforms to apply, and also things to look for which in turn imply transforms to apply that would imply them, of a potentially very rich set of contextual information that implies it.

Depending upon the fade factor, the initial and continuing merge weights, and the relative weights associated with implied things, different things will become salient, and different times will be needed for them to fade out of memory. Much playing around needs to be done to get good weights (hopefully this will be done automatically, by learning routines). But this appears to be a simple and attractive way of giving continuity over time, with a "short-term memory" that is dispersed throughout all the layers of the perceptual system, from retina to apex.

Handling Two or More Eyes

Extending the recognition cone to handle two eyes (Uhr, 1977b) turned out to be surprisingly easy. Since this was basically a matter of putting the single-eye system into a higher-level loop that iterates the same processes over the second eye, it was just about as simple to have the system handle any number of eyes, as designated to it at the start of a run.

The one-eye system must be given the specific layers at the start of a run. The multi-input system must also be given, for each layer, the "from-eye" and "to-eye" for each eye at that layer. Two or more eyes therefore converge if they all have the same "to-eye." This allows us to specify a convergence layer wherever we wish. We can even start with many eyes (and/or "ears," "fingers" or other input organs) and have several groups converge into several different eyes at the same layer, and these, and others, converge at subsequent layers. That is, we can set up a tree whose buds are the separate retinal images of the several eyes, whose root is the layer at which all inputs have been converged together, with any number of intermediate nodes of convergence into which several converging nodes link.

Learning by Discovery and Induction of Transforms and Layers

Extensions were coded to the binocular vision system that allow the program to generate and discover new transforms, both as a function of external feedback and of the internal feedback got from the convergence of the transformed image from the two eyes into a single internal array (Uhr, 1977b). This was designed to explore issues of competition between the two eyes, and the anomalies that occur when one eye is sutured, so that its cone is not given the experience needed to develop properly. For the two eyes appear to be in a competitive-cooperative situation, as determined by very interesting recent experiments by Guillery, 1974; and others.

A second extension that learns much more extensively has been formulated, and is now being coded (Uhr, 1977a). This system attempts to generate a new transform that is as different as possible from already-existing transforms (either already-generated, or learned). It iterates the transform out through the entire layer, collects inductive evidence about this whole set of similar transforms, feeding it back to each individual transform in the set, and, after enough evidence has been accumulated, tries to decide on the appropriate (sub-) array within which that transform should reside, and be considered to have been "discovered as worth using." It also will generate a whole tree of transforms, when needed, sprouting back from the layer in which the root transform has been indicated by feedback.

Perception Embedded in a Larger Cognitive System

The perceptual recognition cone has been embedded in a larger "SEER"* system (Uhr, 1975a, 1975b, 1976a) that also begins to cycle through the other major cognitive processes - remembering, simple problem-solving, language understanding, and motor action. This gives a stronger inner-directed component to the perceptual (sub-) system, and allows perception to call on and interact with memory searches, deductions, and external motor actions (e.g moving the eye, searching for and prodding the object).

It also raises a number of very interesting problems that must be attacked if we want to put the separate sub-systems that are typically embodied in separate programs back together into well-integrated wholistic systems.

Tests of the Static One-Eye Recognition Cone

The EASEy-Snobol programs are too slow and take up too much core memory to be tested in any reasonably economical way. So a Fortran and, more recently, a Simula version of the basic recognition cone have been coded, and a few tests have been made made (see Uhr and Douglass, 1977, for a first report). These include tests of:

a) one letter or symbol at a time, but varying over many linear and non-linear distortions (e.g. rubber-sheet stretchings, introduction of many gaps and extraneous crossings, turning into dotted lines);

b) "place-settings" that are scenes of knives, forks, spoons and plates arranged in various traditional patterns;

c) natural "real-world" scenes, of houses, trees, cars, sky, grass, etc.

Figure 3 shows approximate drawings of three place-settings that the Simula program recognized and described (the individual pieces, and also the whole place-setting were correctly named). Table 1 summarizes the specifications of the actual layers and sets of transforms used.

*Semantic Sensed Environment Encoder and Responder; See-Errr).

Table 1. Specifications of the Cones Used for the Test Run

A) For simple letters and symbols:

Size of Array	Type of Transform Applied	Shrinkage
20 by 20	1. Local edge detectors	1/2
10 by 10	2. Feature detectors	1/2
5 by 5	3. Compound characterizers	1/5

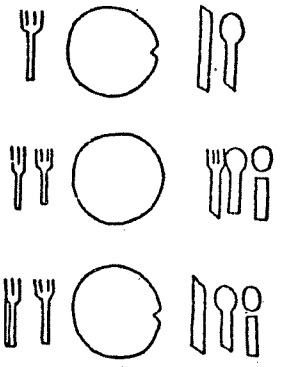
B) For place-settings:

20 by 48	1. Local edge detectors	-
20 by 48	2. Feature detectors	1/2
10 by 24	3. Compound characterizers	1/2
5 by 12	4. Compound characterizers	1/5, 1/12

C) For outdoor scene:

Fortran	800 by 600	1. Average	1/4, 1/3
	200 by 200	2. Hue, saturation, intensity	cropped
Simula	120 by 120	3. Gradients	1/2
	60 by 60	4. Short edges, texture	1/2
	30 by 30	5. Long edges, compounds, textures of short edges	-
	30 by 30	6. Higher-level compounds	-
	30 by 30	7. Higher-level compounds	-
	15 by 15	8. Average	1/2
	8 by 8	9. Average	1/2
	4 by 4	10. Average	1/2

Figure 3. Place-settings were input into a 20 by 48 array. About 30 transforms were used, in a 4-layer cone. It took roughly two hours for a human to formulate and code the transforms, and 8 seconds of 1110 CPU time to describe each scene.



Ohlander's (1975) color pictures were used for the natural "real-world" scenes. Figures 4-9 show how a 10-layer cone with approximately 70 transforms successively transformed a 600 by 800 array containing a house scene (also used by Hanson and Riseman, 1974, 1976), and, in the later layers, assigned names to regions of the scene. (It took roughly 120 hours to formulate and 40 hours to code the transforms, and 90 seconds of 1110 CPU time to process one scene.) The transforms were chosen to be useful and general over a wide variety of scenes. But additional tests will be needed to determine whether they are sufficient, or whether additional transforms are needed.

The following transforms were used, each transform iterated and applied everywhere to its input buffer (see Table 1c): (In general, weights of implications go up moving deeper into the cone.)

Layer 1: Averaging is effected by looking at each 4 by 3 local array of cells, and outputting the sum of the intensities for each of the 3 primary colors into the cell in the output buffer layer corresponding to the center of the 4 by 3, thus converging from an 800 by 600 to a 200 by 200 array.

This was done chiefly to reduce the very large amount of data in the 800 by 600 tv image. Averaging is probably usually a reasonable thing to do on most scenes. But if the scene might contain any tiny details of importance, then the system cannot take the chance of averaging such information out of existence. Rather, it should start with something like local differencing, to get gradients and edges.

Layer 2. The primary colors are combined, giving a) hue (the single combined color), b) saturation of that color, and c) intensity of that color. This combining is effected by a transform with 3 parts to its Conditions, where all 3 parts look at the same cell, giving a 200 by 200 output array.

The first two layers of transforms are effected by a Fortran program. The 200 by 200 image is now cropped to 120 by 120 and the Simula program takes over. (The Simula program is now being modified so that it will handle arrays larger than 120 by 120, so that the Fortran program will no longer be needed.)

Layer 3: Local gradients are computed, using a transform that looks at the 4 parts in a 2 by 2 array, and sums the absolute values of the difference between the Northwest and Southeast pair of cells, plus the difference between the Northeast and Southwest pair of cells.

Layer 4: Short local edges are searched for in a 4 by 4 array, using 4 edge detectors (one for each of the slopes 45°, 90°, 135°, 180°). A texture detector fires if more than 6 simple gradient points above a threshold of 16 (from layer 3) are found in a 4 by 4 local array.

Layer 5: Long edges, angles, curves and textures are compounded together, using transforms that typically fire if about 3 out of 5 parts are found.

Two additional textures are got, by counting the number of edges in a 4 by 4 window, and by getting the principal orientation of edges in a 4 by 4 window.

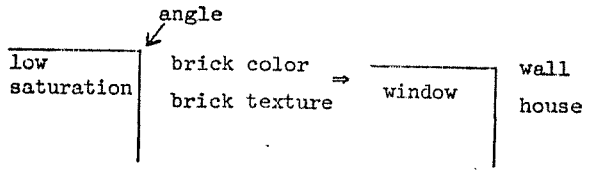
Layer 6: A number of different compounding transforms look for configurations of edges (e.g. vertical edge, slope) and region elements (e.g. wall, sky) and already-implied objects (e.g. roof, window, house). Three examples follow:

a) Blue above a long horizontal edge above the previously implied object roof implies sky (above) and house (below):

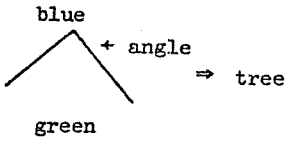
$$\overset{\text{horizontal}}{\underset{\text{Roof}}{\text{Blue}}} \Rightarrow \begin{matrix}\text{Sky}\\ \text{House}\end{matrix}$$

b) A low saturation region, an angle of long edges, and brick color and brick texture on the

other side of the edges implies house, with window in the low saturation region and wall in the brick region:



c) Blue above two long sloped edges giving an upward pointing angle with green below implies trees below:



Layer 7: Still more higher-level compounds that are combinations of previously-implied names are looked for.

Layer 8: Each implied name is averaged over a 2 by 2 array.

Layer 9: Each implied name is averaged over a 2 by 2 array.

Layer 10: Each implied name is averaged over a 2 by 2 array.

This is but a first approximation to a good set of transforms, and a good overall architecture.

Since the last three layers of the cone simply average and converge, they serve only to merge implied names and labels into larger and larger regions. They thus indicate what higher-level identifications predominate; but they do not actually add further information to the description process. Probably five or ten more layers of transforms would be needed in a full-blown system.

Much more work is also needed in choosing the individual transforms, and the weights of their various implications. The present set of transforms was not chosen with the specific scene used for this first test of the system in mind. So they will probably work reasonably well on a variety of outdoor scenes of this sort. But we can expect that experience with new scenes will indicate difficulties, and that a good bit of work will be needed to add, modify, and reweight transforms.

As the system is asked to describe a greater variety of scenes and objects, it will need to be given a larget set of transforms. There are good reasons to think that this system's successive compounding of the rather general probabilistic configurational transforms that it uses means that the total number of transforms needed will level off at a manageable size. But only experimental tests with a wide range of scenes will be able to decide this issue.

The hope is to put as much as possible of the burden of generating a good set of transforms onto learning routines. The transform structure was designed with learning in mind, and first versions of learning routines are being coded. But the amount of computer time needed for the program to generate and adjust the weights of a good set of transforms may be quite large. It seems most likely that a mixture of some learning by the program and a good bit of human effort spent in modifying and fine-tuning the set of transforms will be needed to get as good performance as possible. But learning becomes crucial when the system must recognize and describe new and unanticipated objects.

The test results that follow should be examined keeping in mind these possibilities for extending and improving the system's overall architecture and individual transforms.

Figure 4 shows the results of three transformations layers through the cone (only the results of local gradient detectors are indicated by the asterisks; many other things have been implied by this time).

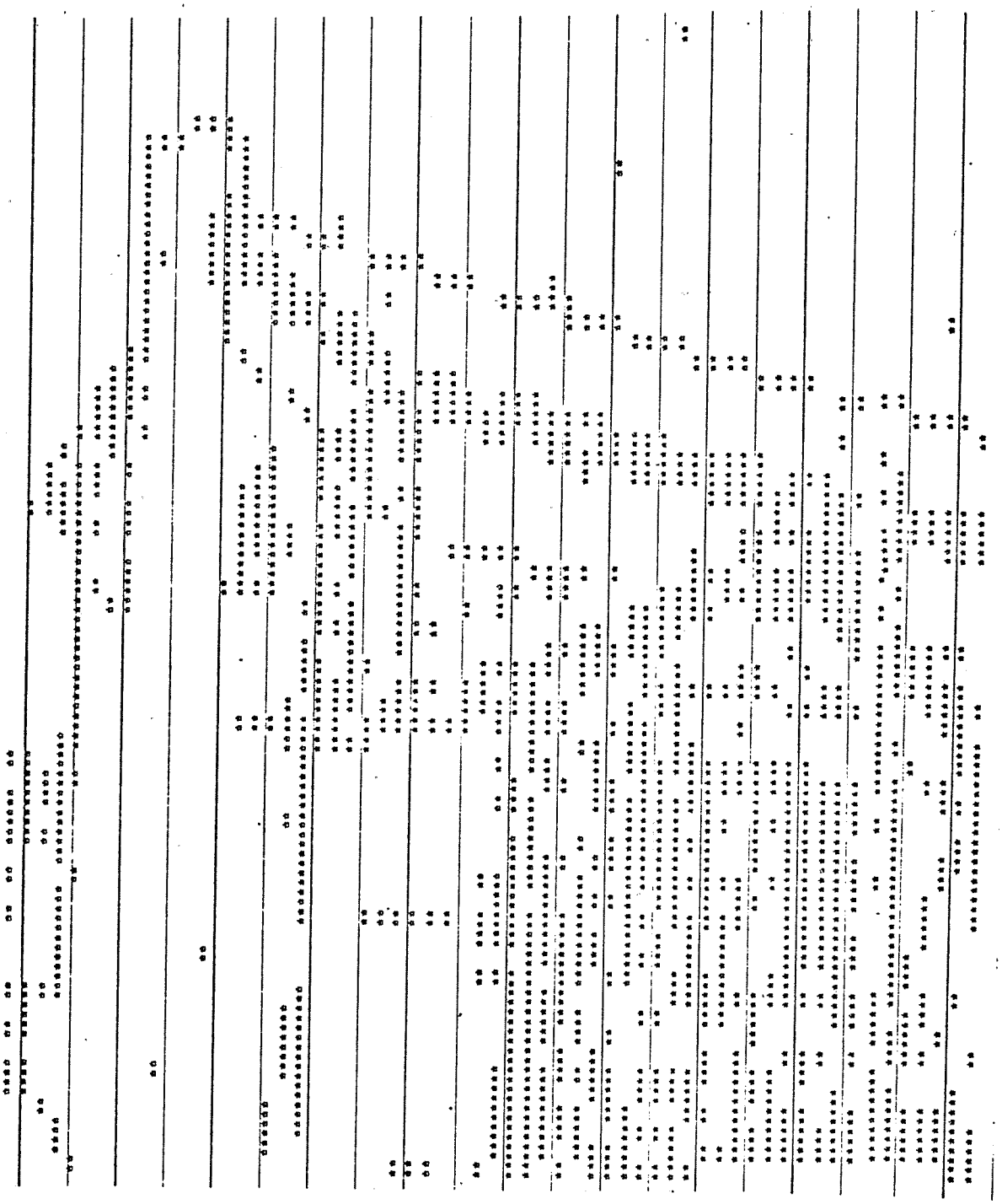


Figure 5 shows the single most highly implied thing output to each cell by Layer 6 (F = Window, G = Grass, H = House, S = Sky, T = Tree, W = Wall).

```
S S S S S S S S S S S S S S S S S S S S S S S S S S
S S S S S S S S S S S S S S S S S S S S S S S S S S
S S S S S S S S S S S S S S S S S S S S S S S S S S
S S S S T T W T S S S S S S S S S S S S S S S S S S
S S S S T W W T S S S S S S S S S S S S S S S S S S
S S S S T W H H H H H S S S S S S S S S S S S S S S
S S S S H W H W H H T H H H S S S S S S S S S S S S
S S S S T W W T H H H G W G S H H H S S S S S S S S
S S S S T W W W W H H W W W T S G T T H H H S S S S
S S S H H W W W W W H H H T H H T T G G T F H H H H
S S H H W W W W W W W T T H H H H H T H H S T S H S
S H H H W W W W H H R H H W W W H T H H H H H H H T S S T
S H W W W W W H W H H H H W W W G W W W W H H H H H H H T T
S T H W W W W H H H H W H W W W W W W W W H H H H H H F F
S S H H W W W F H H H H W H W W H W W W G H T R H H S H H G T
S S W H W W W W W H H W H W W H H H H R H H H H H H H H W W
S S T W W W W W W W H W H W H W H H H H H W W H H H F W W
S S H W W W W W W W W T H H H H R H H H H W H W H H H F W
S S H W W W W W W H H H T H H H H H H H H F F H H T F
F H W W W W W W W H H W W W W H H H H H H F F F F C H H W T
F H W W W W W W W W G W W W W H H W H T T H H H H H H H W T
T H W W W W W W W W W W W W W H H W H T T T S H H S F H W W
S H W W W W W W W W W W W W W H H H H T T T H S H S F H W T
S H W W W W W W W W W W W W H T H T H T T H H S H S H H W T
S H W W W W W W W W W W W W W H H H T H T H H S H H H H T T
T T W W W W W W W W W W W W W H T T G T H H H H H H H H T T
W W W W W W W W W W W W W W W W H T T W H H H H H H H H T G
W W W W W W W W W W W W W W W H W H H T T T T T T H H T G G
W W W W W W W W W W W W W W T W H T T T T H T T T T T T T
G T W W W W W W W W W W W W W W T W T T T T W T T T T T T T
```

Figure 6 shows only those cells in which House (H) was most highly implied (this differs slightly from Figure 5, since ties are shown here but may not appear there).

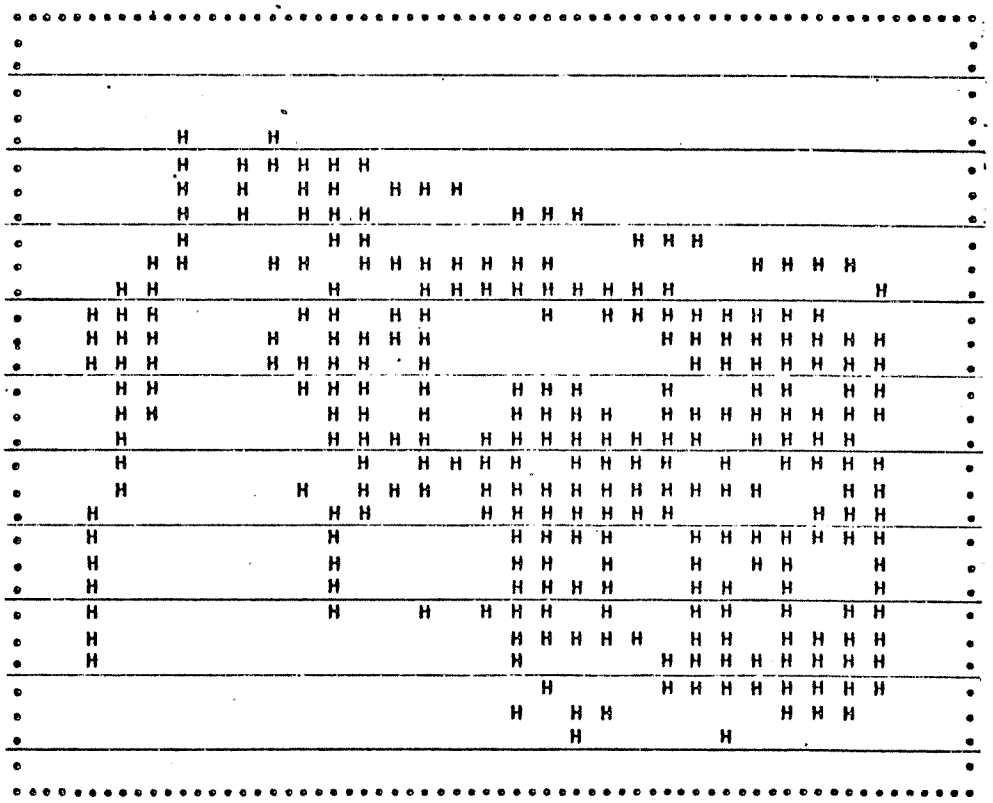


Figure 7 shows the most highly implied things in Layer 7, the next layer.

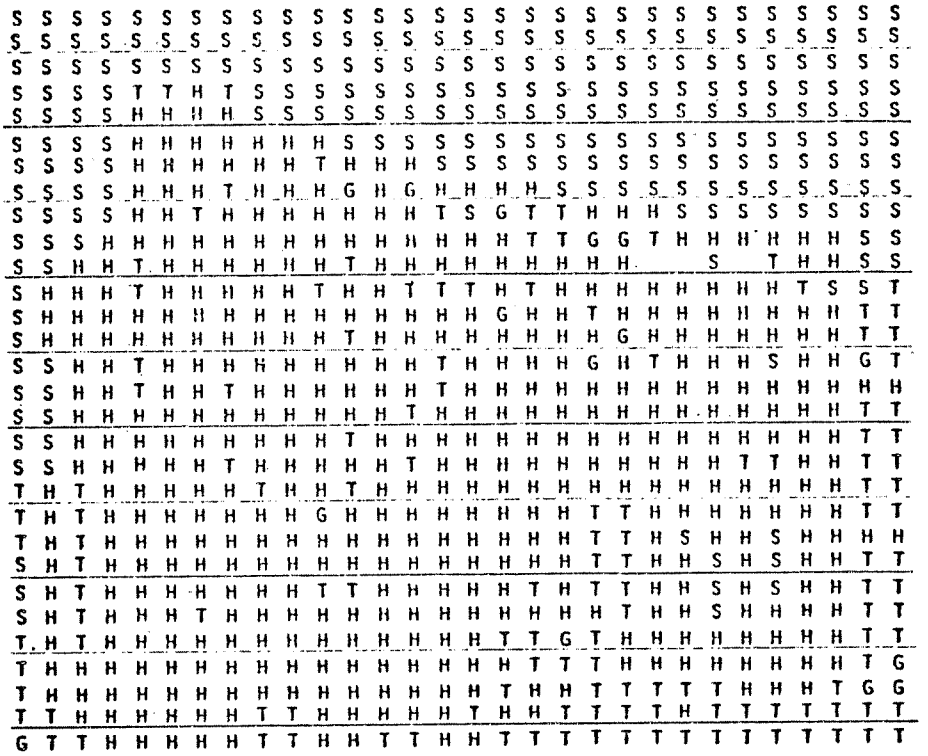
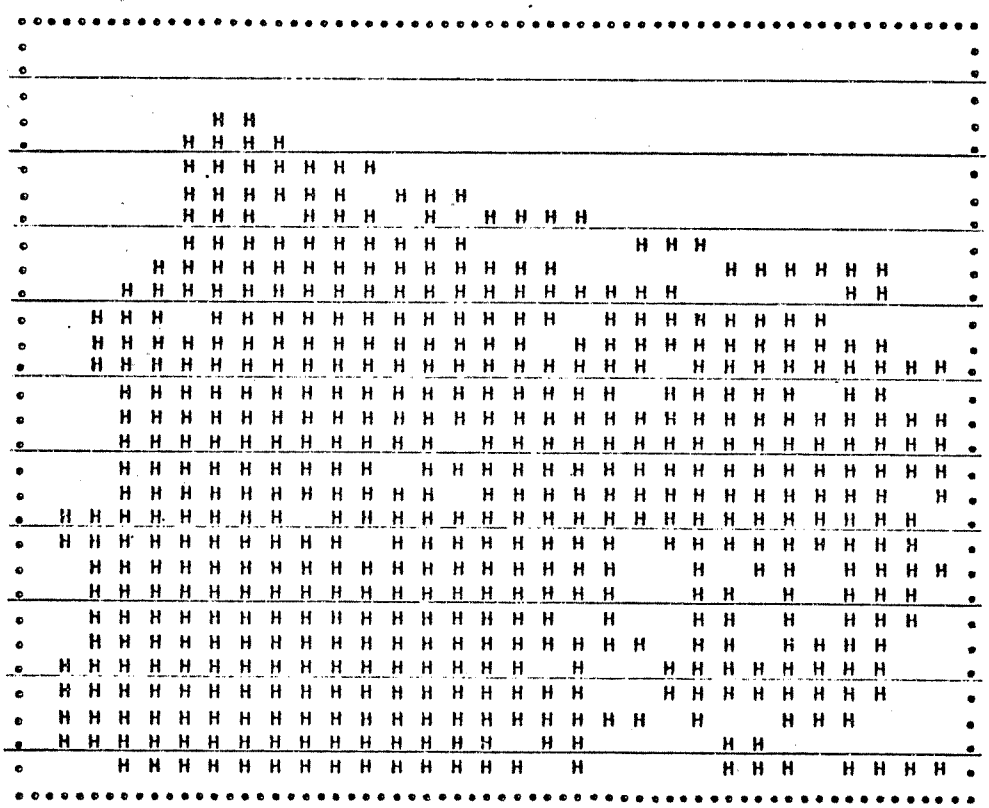


Figure 8 shows only those cells into which House (H) is most highly implied. Note how the higher-level compound for house has turned many of the local "Wall" areas into "House."



Figures 9a, 9b, and 9c show the outputs from Layers 8, 9, and 10.

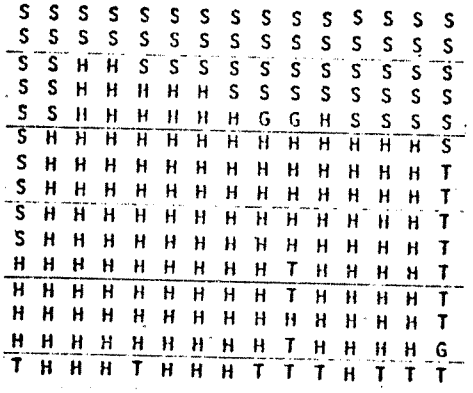


Figure 9a

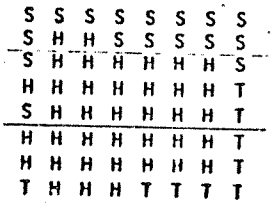


Figure 9b

Figure 9c

Conclusion

These are but first results, yet they seem quite comparable to those achieved to date by other perceptual systems on these very difficult scene description problems. The hope is that a wide variety of scenes of natural objects will be handled by this system, after it has been given, or has learned, a sufficient (yet economical, efficient and general) set of transforms. More transforms are needed to handle a wider range of scenes. But the present transforms were chosen to be general, not ad hoc to the particular test scenes. And when appropriately structured parallel-serial hardware becomes available this kind of system will certainly be extremely fast, and will hopefully be powerful.

References

1 Cordella, L., Duff, M.J.B. and Levialdi, S. Comparing sequential and parallel processing of pictures, Proc. IJCPR-3, 1976, 4, 703-707.

2 Davis, L. S. and Rosenfeld, A. Applications of relaxation labelling 2: spring-loaded template matching, Proc. IJCPR-3, 1976, 4, 591-597.

3 Douglass, R., Recognition and spatial organization of objects in natural scenss, submitted to IJCAI-5, Cambridge, 1977.

4 Duff, M.J.B., CLIP 4: a large scale integrated circuit array parallel processor, Proc. IJCPR-3, 1976, _, 728-733.

5 Ejiri, M., A prototype intelligent robot that assembles objects from plan drawings, IEEE Trans. Comp., 1972, 21, 161-170.

6 Guillery, R. W., Casagrande, V. A. and Oberdorfer, M.D., Congenitally abnormal vision in siamese cats, Nature, 1974, 252, 195-199.

7 Guzman, A., Decomposition of a visual scene into three-dimensional bodies, Proc. FJCC, 1968, 33, 291-304.

8 Hannah, M. J., Computer matching of areas in stereo vision, Unpubl. Ph.D. Diss., Stanford Univ., 1974.

9 Hanson, A. and Riseman, E., Pre-processing cones: a computational structure for scene analysis, Tech Rept. 74c-7, Univ. of Mass., 1974.

10 Hanson, A., Riseman, E., and Williams, T., Constructing semantic models in the visual analysis of scenes, Proc. Milw. Symp. Auto. Comp. & Contr., 1976, 4, 97-102.

11 Klinger, A. and Dyer, C., Experiments on picture representation using regular decomposition, TR Eng. 7497, UCLA, 1974.

12 Kruse, B., The PICAP picture processing laboratory, Proc. IJCPR-3, 1976, 4, 875-881.

13 Levine, M.D. and Leemet, J. A method for nonpurposive picture segmentation, Proc. IJCPR-3, 1976, 4, 494-498.

14 Lowerre, B. T. The Harpy speech recognition system, Unpubl. Ph.D. Diss., Carnegie-Mellon Univ.; 1976.

15 McKee, J. W. and Aggerwal, J. K. Computer recognition of partial views of three dimensional curved objects, Proc. IJCPR-3, 1976, 4, 499-503.

16 Minsky, M. and Papert, S. Perceptrons, Cambridge: MIT Press, 1969.

17 Moayer, B. and Fu, F. S., Application of stochastic languages to fingerprint pattern recognition, Pattern Recog., 1976, 8, 173-179.

18 Nagel, H., Experiences with Yakimovsky's algorithm for boundary and object detection in real world images, Proc. IJCPR-3, 1976, 4, 753-758.

19 Ohlander, R., Analysis of natural scenes, Unpubl. Ph.D. Diss., Carnegie-Mellon Univ., 1975.

20 Reddy, D. R., et al, The HEARSAY-1 speech understanding system, Proc. IJCAI-3, 1973, 185-193.

21 Rosenblatt, F. Principles of Neurodynamics, Washington, Spartan 1962.

22 Sakai, T., Kanade, T. and Ohta, Y., Model-based interpretation of ourdoor scenes, Proc. IJCPR-3, 1976, 4, 581-585.

23 Tanimoto, S. L., Pictorial feature distortion in a pyramid, Comp. Graphics Image Proc., 1976, 5, 333-352.

24 Tanimoto, S. L., and Pavlidis, T., A hierarchical data structure for picture processing, Comp. Graphics Image Proc., 1975, 4, 104-119.

25 Tenenbaum, J. M. and Barrow, H. G., IGS: a paradigm for integrating image segmentation and interpretation, Proc. IJCPR-3, 1976, 4, 504-513.

26 Uhr, L. Layered "recognition cone" networks that preprocess, classify and describe, Proc. Conf. on Two-Dimensional Image Processing, 1971. (reprinted in IEEE Trans. Comp., 1972, 758-768).

27 Uhr, L. A model of form perception and scene description. Computer Sciences Dept. Tech. Rept. 231, Univ. of Wisconsin, 1974.

28 Uhr, L. Toward integrated cognitive systems, which must make fuzzy decisions about fuzzy problems. (in L. Zadeh, et al., Eds. Fuzzy Sets, New York: Academic Press, 1975, pp. 353-393). (a)

29 Uhr, L. A wholistic cognitive system (SEER-2) for integrated perception, action and thought. MSACC-75 Proceedings, Milwaukee, 1975, pp. 25-52. (b)

30 Uhr, L. A wholistic integrated cognitive system (SEER-T1) that interacts with its environment over time, MSACC-76 Proceedings, Milwaukee, 1976, pp. 75-80. (a)

31 Uhr, L. "Recognition cones" that perceive and describe scenes that move and change over time, Proc. IJCPR-3, 1976, 4, 287-293. (b)

32 Uhr, L. Learning by discovery in parallel-serial "recognition cones," Computer Sciences Dept. Tech. Rept., Univ. of Wisconsin, 1977.(a)

33 Uhr, L. Toward a model of binocular vision, in preparation, 1977. (b)

34 Uhr, L. and Douglass, R. A parallel-serial "recognition cone" system for perception: some test results, Computer Sciences Dept. Tech. Rept., Univ. of Wisconsin, 1977 (submitted for publication).

35 Waltz, D. Understanding line drawings of scenes with shadows, In P.H. Winston, Ed., The Psychology of Computer Vision, New York: McGraw-Hill, 1975.

36 Waltz, D. Position paper, Workshop on Computer Vision, Computer Sci. Dept. Univ. of Mass., Amherst, 1977.

37 Williams, H. A net-structure learning system for pattern recognition, Pattern Recog., 1976, 8, 261-271.

38 Zobrist, A. L., The organization of extracted features for pattern recognition, Pattern Recog., 1971, 3, 23-30.

39 Zobrist, A. L. and Thompson, W. B. Building a distance function for gestalt grouping, IEEE Trans. Comp., 1975, 24, 718-728.

40 Zucker, S. W. Relaxation labeling and the reduction of local ambiguities, Proc. IJCPR-3, 1976, 4, 852-861.

APPENDIX

Position Questions

Segmentation

1. What should be the primary basis of segmentation: regions, edges, clusters in feature space? Are these equivalent? (How) should they be combined?

2. How can multiple features and sensory modalities, including color, texture, depth, motion, be used to simplify segmentation? What are the current limitations in reliably processing each of those features? Given these difficulties, what is the relative utility of each to the segmentation process.

3. Which techniques will be powerful enough to deal with the textural variations of natural scenes. What techniques (e.g., histograms) are necessary (or desirable) to extract global feature activity? How can different levels of texture (micro-macro) be detected and utilized?

4. Do parallel vs. sequential approaches lead to different methodologies? To what extent should we be concerned with the development of parallelism at this point?

5. How does the resolution of the sensory data affect the segmentation process and ultimate performance of the system? Can crude segmentation be obtained from coarsely sampled data? Should we be considering the equivalent of eye movements and/or foveation? Is the processing of a hierarchy of resolution levels helpful? How can a scene be 'glanced at' to quickly determine and locate one or two objects or relationships of immediate interest?

6. How can (or should) semantic knowledge be used to guide low level segmentation? What level(s) of knowledge is needed e.g., domain independent interpretation of grey levels as surfaces; domain specific interpretation of regions as objects?

7. How well can (must) segmentation be performed at the retinal level without explicit knowledge? With only domain independent knowledge?

Representation

8. As logical 'units' for higher processing, what primitive visual features are necessary (or sufficient) to adequately characterize the nature of a region? A line? A surface? An object? What are the deficiencies of our current representations of these features?

9. Given that there is a transformation from the sensory data to the symbols representing the 'meaning' of the sensory data, where is the point in the process that the numbers and symbols meet? Are there any well-defined intermediate levels between sensory data and symbols? If so, what form do these intermediate representations take?

Systems

10. How should the overall systems be organized? How should semantic knowledge be represented and applied? Should there be a hierarchically structured knowledge base? Frame-like structures? Relaxation or constraint satisfaction methods?

11. How can multiple sources of knowledge be integrated?

12. How should goal-oriented vision systems be developed? Can general systems be provided goals after their design is fixed? Or should task-specific systems be developed out of standardized components? How can the goals be integrated into the control strategy to affect the focus of its processing and the depth of its analysis?

13. Does vision have anything to learn from speech understanding systems? Is there any 'visual linguistics'? Are syntactic methods viable?

Research Directions

14. Are our current techniques representative of the full range of possibilities? What are their fundamental and practical limitations? Can they be extended to much larger domains (to handle a much larger number of objects using less domain specific knowledge)? Are distinctly different approaches (e.g., from brain studies) required or potentially useful?

15. What are important and promising directions for further research and what problems can be expected?

16. What are the potential applications of these systems, special purpose (near-term) or general? Is a general vision system a viable goal for research or should research be applications driven?

Evaluation

17. What is the criteria for judging the performance of vision systems, both applications and general systems?

18. What are reasonable expectations for computer vision systems? All agree that vision is hard, but how hard is it? What is the level of complexity, amount of knowledge, computational capacity needed? Can we expect general vision systems in five years? Ten years? Fifty years?