The University of Wisconsin
Computer Sciences Department
1210 West Dayton Street
Madison, Wisconsin 53706

Received: January 2, 1975

"RECOGNITION CONES" THAT PERCEIVE AND DESCRIBE SCENES THAT MOVE AND CHANGE OVER TIME

by

Leonard Uhr

Computer Sciences Technical Report #235

January 1975

"RECOGNITION CONES" THAT PERCEIVE AND DESCRIBE SCENES THAT MOVE AND CHANGE OVER TIME

Leonard Uhr
University of Wisconsin
Madison, Wisconsin

Abstract

This paper describes a "recognition cone" model of the perceptual system that can input and process a continuously changing image of a scene (e.g. the successive frames of a movie or television camera). It recognizes and describes the things (mixtures of words and objects) in the scene, including their parts, qualities, and interrelations. It uses a hierarchy of configurational characterizing transforms, where a transform can imply a) other transforms to apply, as well as b) possible names and other descriptive information to assign to the scene, and c) triggers to choose among possibilities in subregions of the scene. Choices of the components of the description are made from the set of implied possibilities, where each can be implied by a variety of contextually interrelated characteristics.

The system will output either a "complete" or a stylized description, or a description that is the dynamic consequence of a user's conversational sequence of interactive requests for more information. This system is actually the perceptual "front end" of a wholistic cognitive system that calls for and uses its internal descriptions to help choose and carry out sequences of actions, both internal (e.g. to deduce, remember) and external (e.g. to move, manipulate, glance about).

Introduction

This paper describes and discusses a computer-programmed model for the perception of scenes of mixed words and objects that change and move about over time. The model (coded in EASEy (Uhr, 1973f) an English-like variant of SNOBOL designed to enhance list-processing capabilities, and to be easy to read) is a hierarchically layered parallel-serial structure of probabilistic configurational transforms, and is designed to embody a number of the features of living visual systems. It is an extension of "recognition cone" systems that name objects and describe scenes (Uhr, 1972, 1973e) so that they now merge successive momentary scenes into short-term memories that allow them to recognize objects even though they come into view a little bit at a time, and to keep track of objects even though they move about.

The ability to handle continuously changing scenes allows the system to "glance about" and look for hypothesized objects as a function of expectations that are themselves implied by both external and internal presses, including previously recognized things, partially recognized things, and internal needs and goals. The system handles serial processes quite simply and naturally, by applying them at successive moments in time. It

thus combines many of the advantages of parallel and serial processes, and begins to coordinate external time-needed-for-environmental-changes with internal time-needed-to-recognize-and-"think".

Perceivers for Static and Changing Scenes

Description of Static Scenes

Systems for pattern recognition and scene analysis and description have with only a handful of exceptions been designed to handle a single input scene (e.g. Austerman et al, 1972; Firschein and Fischler, 1971), or a spoken sentence (e.g. Vicens, 1969; Tappert and Bixon, 1973). This is still an enormously complex problem. Only the simplest of scenes can be described at all well and even for them today's systems give only very stylized descriptions, since the repertoire of possible descriptive information is quite small and fixed even though real-world scenes are rich in information that may be relevant to a pertinent description. Three-dimensional scenes have been given a great amount of attention in recent years in the context of robot research where the robot must recognize solid objects in the room it moves through. But there problems of non-linear distortions have been avoided, since only perfect straight-edged flat-surfaced figures (polyhedra - chiefly cubes and wedges) are allowed (e.g. Falk, 1972; Waltz, 1972).

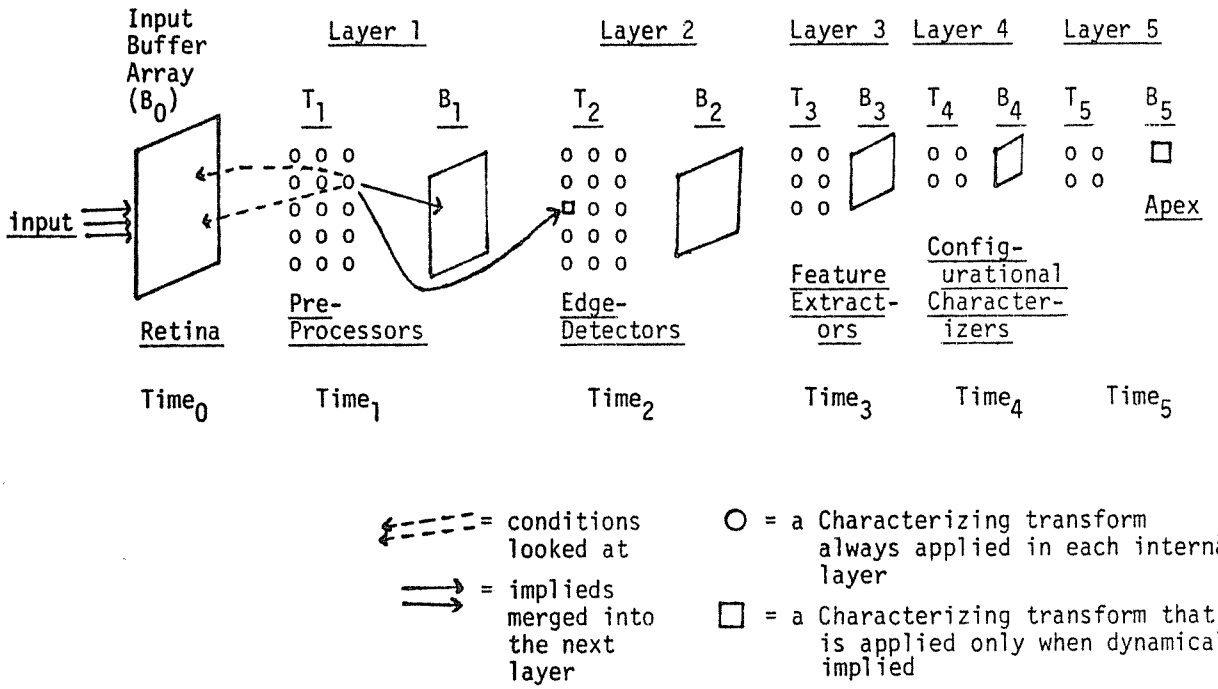
Motion in Time

For real-world scenes the non-spatial dimension of time has been examined by meteorologists interested in cloud cover pictures from satellites (e.g. Hall et al, 1972; Lillestrand, 1972; Smith et al, 1972). Potter (1974) has developed a sequence of algorithms for recognizing scenes of several simple objects moving in different directions in relation one to another, as sensed and input by 3 or 4 frames from a tv camera. Uhr (1973c) has presented a simple algorithm for a short-term memory that merges new scenes into what is remembered about the recent past. Here old things are gradually faded away and forgotten, while new and moving things are given higher weights, and thus made more salient.

The Overall Architecture and General Configurational Transform of Recognition Cones

The basic structure of a "recognition cone" is described in Uhr (1972,1973e) (see also Riseman and Hanson, 1973). I will give a brief overview here, and then describe the extensions that have been made so that it can sense, recognize and describe continuing scenes of things that move about and change over time.

Figure 1. The Overall Architecture of a Simple "Recognition Cone."



A scene is input into the original Buffer array (called B_0, or the Retina). The first Layer of perceptual Transforms (T_1) is applied to this input, and the implications of those that succeed are merged into the next Buffer array (B_1) or, if they are dynamically implied Transforms, the next layer of Transforms (T_2).

An implication can be any function on any set of conditions specified by the Transform. Each Layer can contain any desired mixture of these Transforms. But this figure specifies a simple configuration where pre-processing, edge-detection, feature-extraction, and configurational characterizing proceed in that order.

The output of any and all of these Transforms can be a mixture of any type of implication, and all outputs are merged into the next layer (either Buffer array or Transforms, as specified as the implication's type). Thus possible names are implied into the same cells of the cone as all other information, and passed along through the Layers until they finally reach the Apex cell.

A "recognition cone" describer is a parallel-serial system of layers of transformations (see Figure 1). A scene from the external world is input to its Retina, where a set of characterizers transforms it, merging their implieds into the next layer of the cone. Thus a layer consists of an array of cells, each containing information input to it, and a set of transforms.

The purpose of characterizing transforms is to abstract - to extract, coalesce, and generalize the vast array of detailed information in the retinal representation of the input scene, so that only the "useful" "meaningful" and "relevant" "descriptive" information is finally got. Therefore, each Layer of the system is smaller than the last, giving the overall cone structure, as it transforms from the Retina-base to the central Apex, where all the transformed information finally resides.

A transform specifies a set of tests and/or processes to be effected, at specified relative locations in the layer that it "looks at," along with a threshold for success and a set of things that are implied if the transform succeeds. These implied things include (see Figure 2) 1) possible names to be assigned to a) perceived objects (e.g. TABLE); b) parts of objects (e.g. LEG); c) characteristics of objects (e.g. PURPLE); 2) internal names that the program's transforms might look for in attempts to imply other things (e.g. VE, for a Vertical Edge segment); 3) internal values (e.g. the result of local averaging or differencing, giving smoothed or edge-enhanced figures); 4) names of a) additional things to look for and b) subsequent transforms to apply; and 5) triggers that imply a choice be made among alternate implications of a certain type (e.g. "choose an ANIMAL").

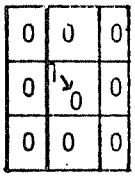
Thus the same general type of transform performs the variety of pre-processing (e.g. averaging, smoothing), feature-extraction, compounding, naming and describing functions that are typically performed by different subroutines of a scene describer (e.g. Falk, 1972; Reddy, et al, 1973).

Uhr (1971) has discussed the similarity between "syntactic" techniques for pattern recognition and this kind of configurational transform which Uhr (1974b) has shown can also be used to express standard rewrite rules for grammars, associations for memory search and productions for deductive problem-solving. They therefore allow us to embed the perceptual recognition cone system described here in a larger wholistic cognitive system (Uhr, 1974a) that also answers queries, deduces consequences, and finds and manipulates objects.

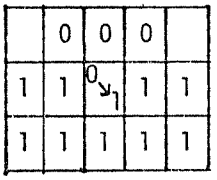
The transforms also imply further things to look for and transforms to apply, so that the system can dynamically look about, directing its search as a function of what it has tentatively found out about the scene so far. Finally, the transforms imply that, because enough has been found out, a decision be made for a specified sub-region of the scene among the possible alternatives that have been implied.

The decision is made among the implications that have been merged into a single cell, which forms the sub-apex of a cone whose base in the Retina delimits the sub-region of the scene to which the chosen name refers. Further descriptive information about that choice can now be got by examining the succession of transforms that implied that choice, including all the things they looked for and found, and also all the other things that have

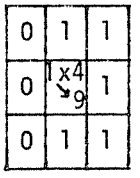
Figure 2. Some Examples of Specific Transforms the Configurational Transforms Used by Recognition Cones Can Handle.



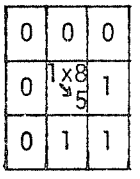
i) eliminates local noise



ii) fills local gap

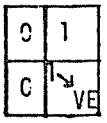


iii) averages to smooth (center cell x 4, other cells x 1)

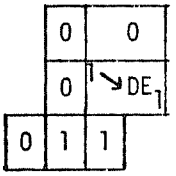


iv) differences, to enhance contour (center cell x 8, other cells x -1)

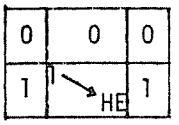
a) Examples of Transforms used for preprocessing



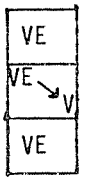
i) Vertical Edge



ii) Diagonal Edge

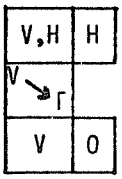


iii) Horizontal Edge

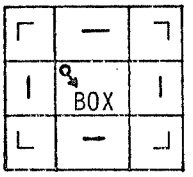


iv) VEs imply Vertical

b) Examples of Simple Edge- and Stroke-Detectors



i) Extracts an angle (or gamma)



ii) BOX, an enclosure

c) Examples of Feature-Extractors

V,H	H	H	V,H
V	0,BOX	0,BOX TABLE	V
V,H	H	H	V,H
V,LEG	0	0	V,LEG

i) TABLE (compounded from alternate possibilities)

TAIL			EAR	
TAIL	TRUNK	TRUNK	HEAD	SNOUT
TAIL		TRUNK DOG		
	LEGS	LEGS		

ii) DOG or other animal (more features would be specified to differentiate)

d) Examples of Characterizers that look at Compounds

In Figure 2 on the previous page note that the diagonal arrow shows what the Transform implies and merges into the corresponding cell of the next Layer of the cone. Only one implied is shown, but there can be any number of implieds. Weights and thresholds are not shown. The actual transform specifies the relative locations of all cells to be looked at, and their weights if found, and an overall threshold that must be achieved for success, and the weights and relative locations of all things implied if successful.

In addition to implying new internal values, internal names, and external names, as shown, transforms can also imply new transforms to apply and triggers that choose among the things implied into the designated cell.

been implied within the same sub-region. Thus the rather subtle issues of deciding how many things there might be in a scene, which things are where, and how they might best be described, are all handled in what appears to be a rather straightforward and elegant way.

Each cell is capable of containing a variety of different kinds of information, including names of wholes, parts and characteristics, and also fragments and qualities without external names. These can include such things as shades of grey, textures, colors, or even moods, since transforming characterizers are capable of looking for and implying any type of thing (e.g. RED and BLUE imply PURPLE; a rough texture above a face implies HAIR; eyebrows arched above eyes implies ANGER).

Each cell can have any number of Transforms (but since this models a nerve network where physical space is needed for connections 10 or 20 seems a reasonable maximum). Each layer can have any desired mixture of any kinds of transforms.

Recognition Cones for Scenes that Change Over Time

The system described in this paper embeds a modified version of Uhr's (1973c) short-term memory algorithm into the recognition cone system for scene description. Essentially, each cell of the recognition cone serves as a separate short-term memory.

The recognition cone that handles static scenes merges the implications of all successful transforms into the appropriate cells, until they are merged all the way to the apex cell, or a trigger sets off a choice among alternate possible implications. At the end of processing of each static scene, after the description has been output, all cells are erased, as part of the initialization process for the next scene.

The Dispersed Short-Term Memory

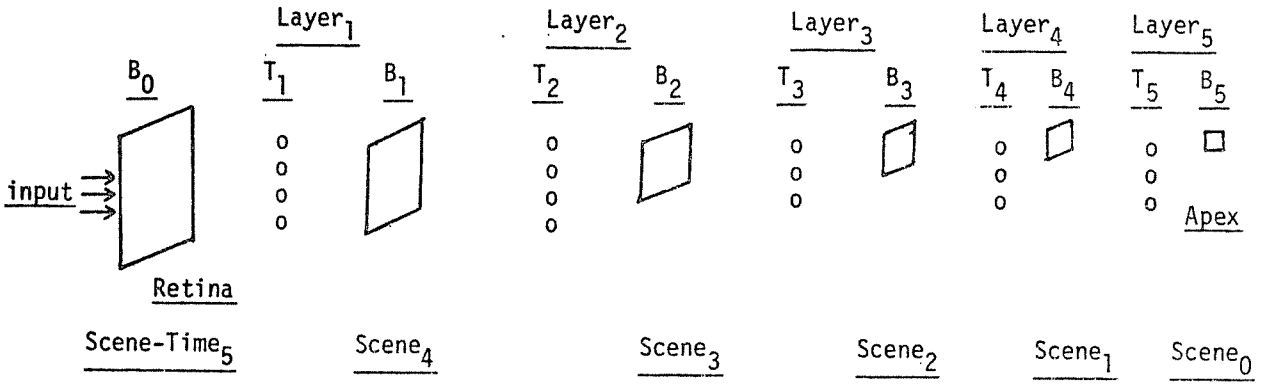
In contrast, to handle changes over time the present system fades rather than erases each cell, by lowering the weight of each thing that has been implied into it, using a

fade value (which is itself a parameter). In addition, the system checks whether an implied is already stored in the cell into which it is to be merged (because it had recently been implied, but not yet faded away) and, if it is not, it is merged with an extra initial weight, to make it more salient. (Note that this is a kind of differencing operation, over time.) This serves to handle both newly perceived things and also things that move, since the latter by moving become newly-perceived by specific particular cells of the cone. This happens in every cell at every layer of the cone. Therefore a "new" or "moving" thing can be of any level of complexity, generality or size - e.g. a primitive edge, or a curve, stroke, part, whole, or collection, or a quality.

Processes Involved in Merging Over Time

This means that at each moment of time only one Layer of Transforms is applied to the Retinal image, and to each of the internal Buffer representations of the image as transformed by prior Layers of the cone. For example (Figure 3) if the cone is 5 Layers deep it will take 5 moments of time before a new scene-frame input to the Retina has its effect on the implied things merged into the Apex 5 layers back. During each of those 5 moments a new scene-frame is input to the first Retinal Layer, and each Layer Transforms whichever of these scenes it is "looking at."

Figure 3. The Merging of Changing Scenes from Successive Moments of Time Into the Continuing Recognition Cone.



At the original moment of Time, $Scene_{Time_0}$ was input to the Retinal Buffer array. At $Time_1$ the first Layer of Transforms merged its implications into the next Layer ($Buffer_1$ and $Transforms_1$, i.e. B_1 and T_1), and the next moment's Scene, $Scene_{Time_1}$, was merged into B_0. And so on, until at $Time_5$, the implications of successively Transformed original $Scene_0$ are merged into the Apex B_5 (and $Scene_5$ is input into B_0).

Thus at each moment of Time a new Scene is input and merged into the Retina, B_0, and the transforms from each Buffer ($B_0, B_1, \ldots, B_{n-1}$) are applied and their implications

merged into the next Layer ($L_1, L_2, \ldots, L_n$). At $Time_5$ implications from the original $Scene_{Time_0}$ are finally merged into $Layer_5$ (the Apex B_5), from $Scene_1$ into $Layer_4$, from $Scene_2$ into $Layer_3$, from $Scene_3$ into $Layer_2$, from $Scene_4$ into $Layer_1$, and from $Scene_5$ into $Layer_0$ (the Retina).

Each Buffer array is a function (because of the short-term-memory algorithm) not only of the implications merged into it at the present moment of Time, but also of implications merged into it during the recent past that have not yet completely faded away. E.g., at moment 21 the 21st Scene-frame, S_{21} is input to the retina, the transforms from S_{20} are merged into the second layer's input buffer, S_{19} into the third layer's input buffer, S_{18} into the fourth, S_{17} into the fifth, and S_{16} into the apex, which is the fifth layer's output buffer.

Some Examples of Scenes that Change Over Time

What kind of scenes of what kinds of objects should such a system handle?

Once time is introduced, it seems reasonable to think of the time the system needs to decide upon, compose, and effect its own behavior, as a function of perceived stimuli. So we really need to think in terms of something I will call "event-sequences," where stimuli that themselves unroll over time elicit behavior that acts upon these or further stimuli, in turn leading to further consequences that are more or less anticipated and serve usually as feedback or reinforcement. The simplest prototypical example is the food-object that comes into view, leading to its capture and ingestion, leading to pleasure (see Uhr, 1975).

But for the moment, since we are concentrating on the perceptual aspects of such event-sequences, we can examine the recognition and description of the scene of moving and changing objects. For example, the prey - the mouse - comes slowly into view. Or a sequence of cues and tip-offs are perceived - e.g. bent grass, droppings, a tail, a squeak.

It is probably simpler to consider several different interpretations when we are concentrating on the perceptual system alone. Consider a succession of moving picture frames, where the first contains the mouse's whiskers, the second whiskers and snout, the third the head, the fourth the back of the head and neck, etc., as though the mouse passing by is viewed through a narrow slit. In general, we are talking about events or things that are not perceived at one moment. These are uncommon for physical objects like mice, tables, or letters, since the perceptual apparatus has evolved to be large enough to subtend a whole scene of such objects at the same glance. But there are a number of common perceptual experiences where this occurs. A good model of perception must handle such situations, as does the perceptual system of man.

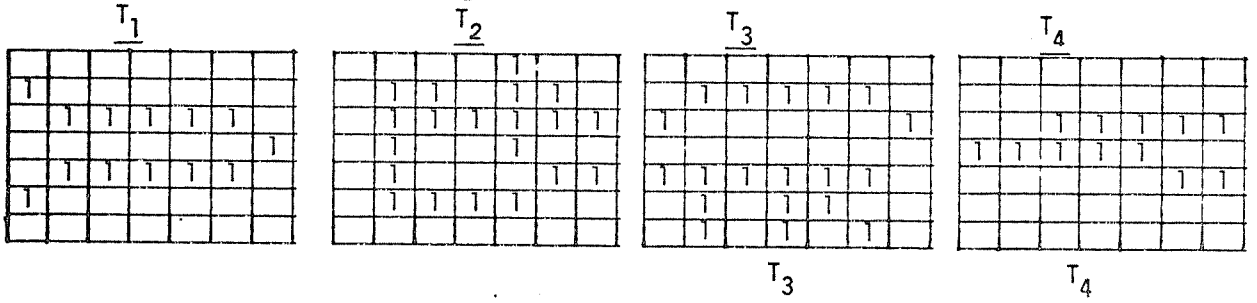
Consider spoken speech, where an utterance of words (composed of phonemes or other components) unrolls over time, only a small slit of the utterance being present at any moment. A visual analog would be a sentence flashed on a screen one word at a time, or one syllable or letter at a time. People are probably not nearly so good at handling such visual as spoken utterances over time, but we can do it, and quickly learn to do it well.

Reading morse code or a ticker tape is another example.

Demonstration Examples of Recognition and Description Over Time

This system is presently coded in EASEy-2 (Uhr, 1973f) an English-like variant of SNOBOL designed to enhance its list-processing capabilities and make for programs that are easier to read. Only a very simple example of the system's behavior is given, to demonstrate that it can indeed, when given a small set of characterizing transforms, handle scenes of moving objects.

Inputs at Successive Times (T_1,T_2,T_3,T_4):



Output = "STICKDOG"

Inputs of Simpler higher-level features of successive Times:

T_1	T_2	T_3	T_4
SNOUT	HEAD	TRUNK	TAIL
	EARS	BACK	REAR
	EYE	LEG	
		LEG	

Output = "STICKDOG"

T_1	T_2	T_3	T_4	T_5
HEAD	NECK	ARM	LEGS	FEET
EARS		TRUNK		
MOUTH				

Output = "STICKPERSON"

The examples use only a few simple transforms, make little use of weights and thresholds, and assume carefully drawn input scenes. A much larger memory of transforms is needed to handle messier real-world scenes. (The transforms are equivalent to the rewrite rules that specify a particular grammar for a parser.) But sets of probabilistic parallel-serial (or even strictly parallel) transforms appear to have given the best results in pattern

recognizers for real-world messy pattern sets whose members vary over unknown and non-linear transformations (e.g. Doyle, 1960; Uhr and Vossler, 1961; Chow, 1963; Andrews et al., 1968; Zobrist, 1971; Uhr, 1973a).

More extensive tests with larger sets of transforms will wait on the addition of routines that generate and discover these transforms, and a faster and more efficient Fortran system. See Uhr, 1975 and 1973e for the program.

Using the Description

All implied descriptive information about objects, parts, and qualities is finally merged and passed back into the Apex of the cone. There is now a potentially very difficult problem, of choosing the most relevant information. This can be handled in several ways: 1) All descriptive information can be output. 2) The 1, 2, or N (a pre-set number) of most highly implied things can be output, each, optionally, with 1,2, or N descriptive things about it. 3) The description can be output in response to interactive requests that allow the user to ask for the kinds of information he thinks will be relevant (see Uhr, 1973e for such a system). 4) The larger cognitive system within which the perceptual system is embedded can make use of this information, as appropriate (see Uhr, 1973b,d, 1974a,b, 1975).

This means that further associations and deductions can be made about perceived information, as continuing cognitive transformations move into the cognitive memory to search for relevant information.

Parallel vs. Serial vs. Parallel-Serial Recognizers

A wide variety of different structures have been given to programs that attempt to perceive complex patterns, some of which are indicated in broad outline in Figure 4.

Strictly Parallel Systems

By far the most common structure (Figure 4a) for programs for "pattern recognition" is a single layer of transforms (usually called "feature-extractors," "templates," or "demons") that look in parallel at the Retinal representation of the input (e.g. Selfridge, 1959; Doyle, 1960; Rosenblatt, 1958). Each transform specifies one or more implied names (usually with weights). These are merged into a list of possibilities, from which the single most highly implied name is chosen.

Strictly Serial Systems

In sharp contrast (Figure 4b), a few pattern recognition systems have used "strictly serial" structures of transforms (e.g. Unger, 1959; Naylor, 1971a,b). Here a single 1st-level transform is applied to the retinal input. Its outcome implies which transform to apply next, and so on, until an external name is implied, and this is output as the

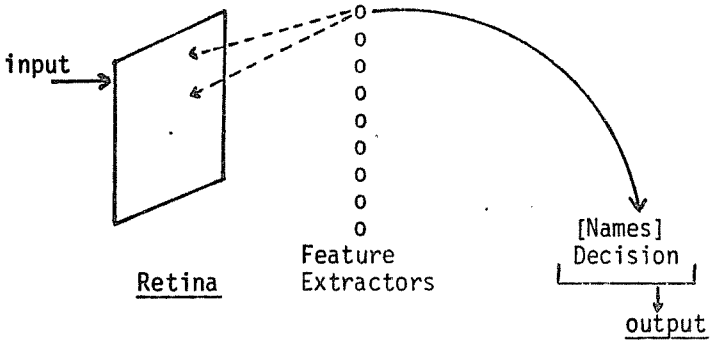
name the system has chosen to assign. Thus a serial recognizer is a tree of transforms whose root transform is applied first, and whose buds are names to be output.

This structure has often been used for "concept formation" (e.g. Hunt, 1962; Towster, 1969) and remembering language (e.g. Feigenbaum, 1961) and is, essentially, what is often called a "discrimination net."

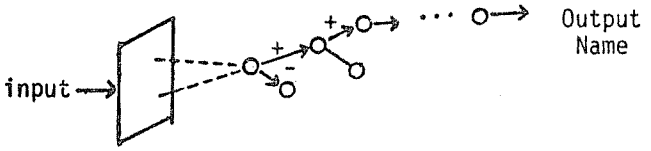
Criticisms of Strictly Serial Systems (e.g. by Selfridge)

Serial systems have been criticized by Selfridge (1959) and others as being only as strong as the weakest link in each path. To the extent that inputs are messy and their basic features unknown (which is the essential problem with real-world patterns) the recognizer's individual transforms will be weak and fallible, and this flaw will loom larger. So most people have built parallel structures of weighted transforms, expecting to gain power by combining them into a more reliable and valid total set,

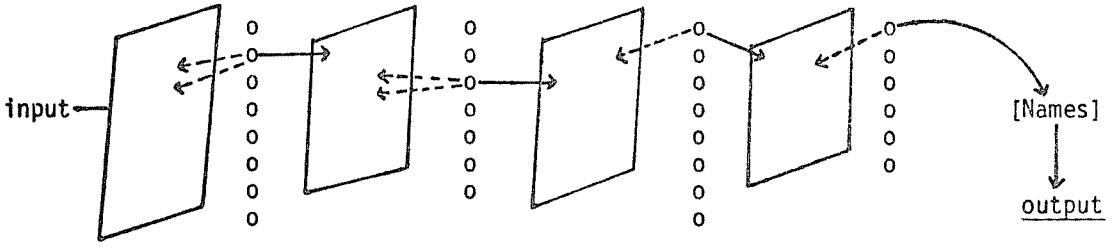
Figure 4. A Survey of Some Configurations Used in Programs for Perceptual Systems



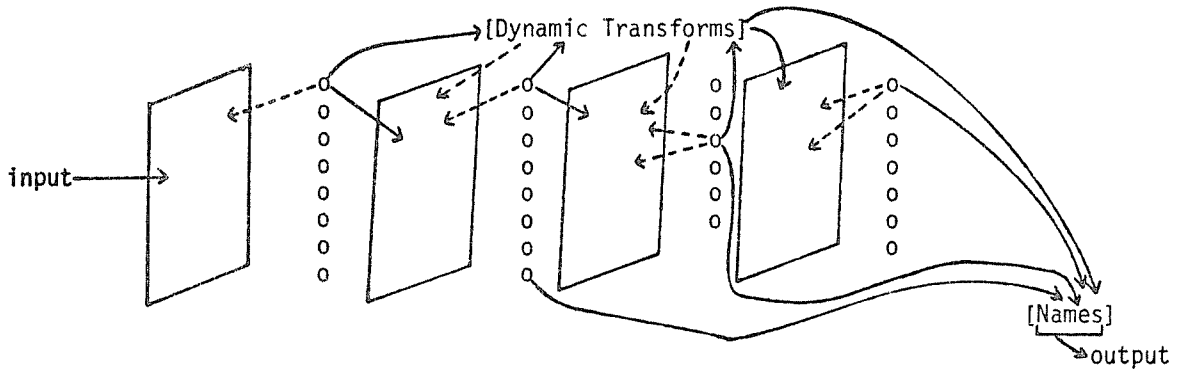
a) 1-layered strictly parallel recognizers (e.g. Rosenblatt, 1958; Selfridge, 1959).



b) Strictly serial recognizers, and concept formers, serial algorithms and Turing Machines.



c) Simple parallel-serial recognizers



d) Dynamic, flexible parallel-serial recognizers. Possible external names can be implied from any Transform Layer. Transforms can dynamically imply new Transforms to apply.

just as large numbers of individually weak measures are often combined in agricultural and psychological experiments and in achievement tests.

Criticisms of Strictly Parallel Systems (e.g. by Minsky and Papert)

Minsky and Papert (1969) and others have criticized strictly parallel recognizers, showing the very simple "perceptron" (Rosenblatt, 1958) type to be enormously inefficient to the extent it is confronted with global characteristics (e.g. parity, continuity). They propose "serial algorithms" as the alternative (but it is not clear whether by this they mean strictly serial discrimination nets). They present elegant Turing Machine programs to show how efficiently parity and continuity can be computed, and Turing Machine programs are indeed good examples of serial algorithms; but they are also notoriously slow and inefficient for any but especially well-chosen problems. They also seem to suggest that Guzman's (1968) system (which does not actually work with input scenes, but rather builds much simpler graphs of straight edges joined at vertices into sets of overlapping objects) is a "serial algorithm." But that is hard to understand, since it contains a mixture of parallel processes organized by stages.

Parallel-Serial Systems

Minsky and Papert have been wrongly misunderstood to dismiss parallel-serial systems (e.g. Hunt (1973, p. 370) says, "Minsky and Papert (1969) have shown that there are definite limitations to the capabilities of interesting classes of simple parallel processors when the task requires analysis of geometric properties of stimuli." Anderson and Bower (1973, p. 213) say, "for example, it is known (see Minsky and Papert, 1969) that a Pandemonium cannot distinguish odd vs. even numbers of elements in its viewing field, nor distinguish connected vs. unconnected line drawings." But they have actually merely questioned whether an extension of Rosenblatt's extremely over-weak and over-simple perceptron transforms to structures of 2 or 3 rather than a single layer will make them

appreciably more efficient. And they actually present as an example of the alternative approach that should be taken not the Turing Machine that they used to make explicit what they meant by "strictly serial" algorithms but rather Guzman's program, which is a mixture of parallel and serial processes.

A number of variations on parallel-serial structures are possible, two of which are shown in Figure 4c and 4d. (See Uhr and Vossler, 1961, for an example of a structure without well-delineated layers, and Reddy et al, 1973 for a structure that treats the different layers as co-routines, rather than ordering them successively.)

Parallel processes can save time, but sometimes, as Minsky and Papert have shown, with horrendous costs in space. Serial processes can save space, but at the expense of time and errors. But strictly parallel and strictly serial processes only define the endpoint extremes, and waiting to be explored is a rich middle ground of systems that mix parallel and serial processes. Our problem is one of efficiency, not general power (the "general-purpose" character of the Turing Machine is given it by its "potentially infinite" tape, which, as Minsky and Papert themselves mention (p. 231), can equally well be given to a parallel system (or a mixed system).

Considerations of efficiency would seem to suggest parallel-serial configurations that try to maximize the virtues and minimize the vices of each process taken separately. All nervous systems and most programs for perception, including, I think, the most powerful, are parallel-serial (occasionally, as with the "Robot Vision" systems influenced by Minsky and Papert's work, ignoring their parallel aspects).

The power of the transforms used is also crucial.

Discussion

Time and Mixed Top-Down Bottom-Up Processes

The external scene can imply possible things to look for, which in turn imply specific characterizing transforms to apply. This gives a bottom-up flow to the things, then a top-down flow from things to characterizers.

Characterizers can also imply lower-level transforms to apply. For example, several muddy strokes got at the third or fourth layers might imply the need for further sharpening of edges, at the first two layers that smooth and difference. The new transform will be applied at a moment several time-units later. The need to backtrack and reprocess is completely replaced by just another variant on the standard "glancing-about" capability.

Implicit vs. Explicit Storage of Relative Times

This system handles the merging, short-term memory, and forgetting of things that unroll over time, but without any explicit noticing or referencing of time. It is capable of characterizing events that are sequences of things that occur at different times.

For example, a transform might specify "If A,B,C,D occur then E,F,G are implied," and it will succeed even though A,B,C and D occur successively rather than simultaneously.

It seems best to examine how well such a minimal approach to time works, before adding additional mechanisms more specifically designed to note and handle the specific times at which each thing occurred. But an explicit clock could easily be given the system, and (relative) locations in time as well as in space used to characterize.

Conclusion

This paper has described a parallel-serial "recognition cone" system that uses probabilistic contextually interrelated configurational transforms for the perception of scenes of mixed words and objects that change and move about over time. This perceptual system is actually embedded in a wholistic cognitive system called SEER-T1 (see Uhr, 1975) which integrates perception with memory search, deductive problem-solving, language handling, and acting, and thus is a beginning attempt to handle the variety of cognitive processes that humans handle.

References

Anderson, J. R. and Bower, G. H., Human Associative Memory, Washington: Winston, 1973.

Andrews, D. R., Atrubin, A. J., Hu, K., The IBM 1975, Optical Page Reader: Part III: Recognition and Logic Development, IBM J. Res. and Devel., 1968, 12, 364-372.

Ausherman, D. A., Dwyer, S. J., and Lodwick, G. S., Extraction of connected edges from knee radiographs. IEEE Trans. Comp. 1972, 21, 753-758.

Chow, C. K., A recognition method using neighbor dependence, IRE Trans. Electronic Computers, 1962, 11, 683-690.

Doyle, W., Recognition of sloppy, hand-printed characters, Proc. WJCC, 1960, 17, 133-142.

Falk, G., Interpretation of imperfect line data as a three-dimensional scene. Artificial Intelligence, 1972, 3, 101-144.

Feigenbaum, E., The simulation of verbal learning behavior, Proc. WJCC, 1961, 19, 121-132.

Firschein, O. and Fischler, M. A., Describing and abstacting pictorial structures. Pattern Recognition, 1971, 3, 421-443.

Guzman, A., Decomposition of a visual scene into three-dimensional bodies. Proc. Fall Joint Comput. Conf., 1968, 33, 291-304.

Hall, D. J., Endlich, R. M., Wolf, D. E., Brian, A. E., Objective Methods for Registering Landmarks and Determining Cloud Motions from Satellite Data, IEEE Trans. Comput. 1972, 21, 768-776.

Hunt, E. B., Concept Formation: An Information Processing Approach, New York: Wiley, 1962.

Hunt, E. B., The memory we must have. In: Computer Models of Thought and Language (R. C. Schank and K. M. Colby, Eds.), San Francisco: Freeman, 1973.

Lillestrand, R. L., Techniques for Change Detection, IEEE Trans. Comput., 1972, 21, 654-659.

Minsky, M. L. and Papert, S., Perceptrons: an Introduction to Computational Geometry. Cambridge: MIT Press, 1969.

Naylor, W. C., Machine Imitation, Unpubl. Ph.D. Diss., Univ. of Wisconsin, Madison, 1971.

Naylor, W. C., Some studies in the interactive design of character recognition systems, IEEE Trans. Computers, 1971, 20, 1075-1086.

Potter, J. L., The Extraction and Utilization of Motion in Scene Description, Unpubl. Ph. D. Diss., Univ. of Wisconsin, Madison, 1974.

Reddy, D. R., Erman, L. D., Fennell, R. D., and Neely, R. B., The hearsay speech understanding system. Proc. 3d Int. Conf. on Artificial Intell., 1973, pp. 185-193.

Riseman, E. M. and Hanson, A. R., Design of a semantically directed vision processor, COINS Tech Rept. 74-C1, Univ. of Mass., 1974.

Rosenblatt, F., The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev., 1958, 65, 386-408.

Selfridge, O. G., Pandemonium, a paradigm for learning, In: Mechanization of Thought Processes, 511-535, London: HMSO, 1959.

Smith, E. A., Phillips, D. R., Automated Cloud Tracking Using Precisely Aligned Digital ATS Pictures, IEEE Trans. Comput., 1972, 21, 715-730.

Tappert, C. C. and Dixon, N. R., A procedure for adaptive control of the interaction between acoustic classification and linguistic decoding in automatic recognition of continuous speech. Proc. 3d Int. Conf. on Artificial Intell., 1973, pp. 173-184.

Towster, E., Studies in Concept Formation, Unpubl. Ph.D. Diss., Univ. of Wisconsin, 1969.

Uhr, L. and Vossler, C., A pattern recognition program that generates, evaluates and adjusts its own operators, Proc. AFIPS WJCC, 1961, 19, 555-570. (Reprinted, with additional results, in E. Feigenbaum and J. Feldman, Eds., Computers and Thought, New York: McGraw-Hill, 1963.)

Uhr, L., Flexible linguistic pattern recognition, Pattern Recognition, 1971, 3, 363-384.

Uhr, L., Layered "recognition cone" networks that pre-process, classify and describe, IEEE Trans. Computers, 1972, 21, 758-768.

Uhr, L., Pattern Recognition Learning and Thought, Englewood-Cliffs: Prentice-Hall, 1973(a).

Uhr, L., Recognizing, "understanding," deciding whether to obey, and executing commands, Computer Sci. Dept. Tech. Rept. 173, Univ. of Wisconsin, 1973(b).

Uhr, L., The description of scenes over time and space, Proc. AFIPS NCC, 1973, 42, 509-517(c).

Uhr, L., DECIDER-1: a system that chooses among different types of acts, Proc. 3d Int. Joint Conf. on Artificial Intell., 1973, 396-401(d).

Uhr, L., Describing, using "recognition cones," Proc. 1st Int. Joint Conf. on Pattern Recognition, 1973(e). (Also Computer Sci. Dept. Tech. Rept. 176, Univ. of Wisconsin, 1973.

Uhr, L., EASEy-2: An English-like program language, Computer Sciences Dept. Tech. Rept. 178, Univ. of Wisconsin, 1973(f).

Uhr, L., A wholistic cognitive system (SEER-2) for integrated perception, action and thought. Computer Science Dept. Tech. Rept. 224, Univ. of Wisconsin, 1974(a).

Uhr, L., Toward integrated cognitive systems, which must make fuzzy decisions about fuzzy problems. Computer Sci. Dept. Tech. Rept. 222, Univ. of Wisconsin 1974(b). (Also in L. Zadeh et al., Eds.: Fuzzy Sets, New York: Academic Press, 1975.)

Uhr, L., A wholistic integrated cognitive system (SEER-T1) that Interacts with its environment over time. Computer Sci. Dept. Tech. Rept., Univ. of Wisconsin, 1975.

Unger, S. H., Pattern recognition and detection, Proc. IRE, 1959, 47, 1737-1752.

Vicons, P., Aspects of Speech Recognition by Computer, Unpubl. Ph.D. Diss., Stanford University, 1969.

Waltz, D. L., Generating Semantic Descriptions from Drawings of Scenes with Shadows, Unpubl. Ph.D. Diss., MIT, 1972.

Zobrist, A. L., The Organization of Extracted Features for Pattern Recognition, Pattern Recognition, 1971, 3, 23-30.