THE SYMBOLIC NATURE OF VISUAL IMAGERY

Thomas P. Moran

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania

Abstract

The issue is whether human visual imagery can be represented by symbolic structures and processes. Protocols of the task of imagining a path in space were analyzed using a production system interpreter. A detailed simulation of the subject's behavior within the confines of a symbolic short-term memory (STM) demonstrates that symbol structures are sufficient to explain imaging behavior. The experience of programming with productions and a homogeneous STM has brought out the importance of a control language when using an unstructured rule system.

Descriptors: cognitive psychology, cognitive representation, visual imagery, visual encoding, protocol analysis, simulation, production systems.

The Issue

The nature of the human cognitive representation of the spatial world is a topic of current interest to both the cognitive psychology and the artificial intelligence communities. But in general the two communities seem to have different conventional views of the mode of cognitive operation.

The psychologist tends to work from the sensory channels inward and hence thinks of cognitive structures in terms of modality-specific representational systems--plus an abstract "conceptual" system. For example, the bulk of Neisser's book *[8]* is split between visual and auditory cognition, plus a chapter at the end on "the higher mental processes". Brook's experiments [3] seem to lend support to this multiple-system view. He showed that a visual task (scanning an imagined figure) will be hindered much more by a visual response (pointing) than by a verbal response, but that a verbal task (scanning an imagined sentence) will be hindered much more by a verbal than by a visual response. That is, there seems to be interference only within each modal system.

Some psychologists claim that the conceptual system is really a verbal mode, but probably most believe that the visual system is distinct. This is reinforced by Sperling's demonstration [12] of the existence of a remarkably iconic short-term memory (although the exact nature of the representation in this memory is not understood).

In a series of elegant experiments on the mental rotation of figures, Roger Shepard is trying to establish whether internal visual representation is "analog" in nature. His notion of an analog mental process involves a mapping between the intermediate mental states and the intermediate states of the corresponding process In the real world. In one experiment [11] it was shown that the time to mentally rotate a complex three-dimensional figure is a linear function of the angle of rotation and that this rotation rate is the same whether or not the rotation is in the plane of the drawing presenting the figure. This suggests that mental rotation may be a continuous process. (Shepard specifically does not require his analog process to be continuous, but it Is hard for me to understand how his mapping can be carried out if it is not.)

The a. i. researcher, on the other hand, works on programming a spectrum of intellectual functions using abstract symbol manipulation, which he considers to be the mode of thought processes. This is true of both those concerned primarily with artificial [7] and with human [10] intelligence. Thus in a. i. even vision is cast as a problem of "scene analysis"—the encoding of visual input into a symbolic description which is able to interact with other encoded modes of information.

This division between a. i. and psychology is not so sharp, of course. Clark & Chase *[4]* have nicely demonstrated the necessity of an abstract level of representation in several experiments using the simple task of comparing sentences and pictures. The most thorough information processing investigation of a visualization task (which involves the painting, cutting, and counting of imagined blocks) was done by Baylor [1,2]. His analysis, however, postulated two separate "problem spaces" for his subject, an "image space" and a "symbol space". Information is divided between these two spaces roughly along the generic/ specific dimension; and each space has its own operators.

My interest is the nature of the cognitive representation of synthetic visual imagery. What I mean by 'synthetic" imagery is the mental construction of new spatial patterns not previously visually perceived, that is, not simply the recall of stored visual perceptions. The relation between synthetic images and perceptual images is moot (but see [9J for a theoretical stab at the perceptual encoding process). In this paper all remarks about imagery should be read to mean synthetic imagery.

By concentrating on internally generated visual images and not using visual input we can focus on the cognitive "deep structures", which lend themselves to internal manipulation, rather than on the sensory buffer images. (But whether the Sperling memory [12] is involved in imagery is not really known.) I am not interested in subjective issues, like the "vividness" of imagery, which are hard to operationalize and which do not seem to have any substantive effects [5].

The most parsimonious hypothesis, It seems to me, is that synthetic imagery is simply a symbolic process--there is no need for a distinct "image space" with its own special (non-symbolic) data structures and operators and, perhaps, its own working memory. If we are to take this as a serious psychological hypothesis, then we must show that symbol structures and processes do adequately characterize images by using symbol manipulation to explain behavior in visualization tasks (e.g., Brooks', Shepard's, Baylor's). The constructive way to do this is to discover the coding techniques by which complex visual/spatial relations are represented symbolically. We would like to be able to precisely specify the information content of visual images (i.e., what information is explicit and what is implicit) and the allowable operations on these image structures.

The Task

As a first step I ran some exploratory experiments using a simple spatial memory task. A blindfolded subject was asked to Imagine a blank, two-dimensional plane in front of himself and to locate himself at some point in the plane. Then the experimenter verbally gave him a series of directions (North, South, East, West). For each direction the subject imagined a line of unit length being drawn in that direction on the plane from his current location. The subject tried to understand (visualize) the path thus far drawn by organizing it In some way. He then described the path and repeated the direction sequence of the path to the experimenter. The subject was allowed as much time as he wanted at each move, and he was free to determine his own cognitive organization of the path. The following are the direction sequences for two of the experiments;

```
                    Problem 1

     N E S E S E S E N E S S E E N E N W W N

                    Problem 3

 N E N E N W N E E E S W S E S S W S W N W S W N
```

(To encourage you to try the task yoursulf, I have not included drawings of the paths in this paper. However, you will find it helpful for the following discussion to have made the path drawings.)

Verbal protocols were taken of these experimental sessions. The subject's (verbal) behavior was analyzed by creating a simulation program. The program takes the directions as input and produces a stylized verbal output. Figure 2 presents a short segment of the subject's protocol *lor Problem* 1 along with the corresponding program output.

More important than the verbalizations themselves is what we can infer from them about the subject's internal representations and processes. The program was designed to satisfy not only the external constraints of matching the protocol, but also some of the known internal memory limitations of the human information processor; and so the program is written in a system based on some specific hypotheses about the structure of the human information processor.

The System

My programming system (called VIS) is a production system interpreter of the type advocated by Newell & Simon [10]. Its focus is a small short-term memory (STM) of symbolic expressions which represent the system's (currently) immediately accessible knowledge. The system is driven by a potentially unlimited long-term memory (LTM) of production (condition-action) rules, whose conditions are patterns of expressions in SIM.

VIS's STM is an ordered list of about 10-20 expressions. STM is constantly changing, both in order and content. The STM expressions which are matched by the condition patterns of a rule are brought to the front of STM (attention, rehearsal). Newly created expressions are pushed into the front of STM, forcing out old expressions at the back (forgetting), thus keeping the length of STM constant.

But the expressions in STM may be arbitrarily complex. An expression represents an aggregated "chunk" 16] of information which is accessed in an all-or-none fashion. Formally, it is just a linear list of symbols. An expression can be implicitly embedded in another expression by including the name symbol *of* the former in the latter. This facilitates the creation of hierarchic structures (the chunking of information). Embedded expressions are stored in an intermediate-term memory (ITM), where they may be retrieved by name in case they should be forced out of STM.

The LTM holds the system's permanent and unchanging knowledge (VIS has no operations for adding new information to LTM). The only form of knowledge in LTM is the production rule. The collection of rules completely determines the system's behavior, that is, they constitute the system's program of action. Any rule will fire its action component whenever its condition component matches some current expressions in STM. Rule actions are simple symbolic transformations (e.g., adding or deleting symbols in an expression, creating a new symbol or expression, etc.) which change the state of STM, thus causing other rules to fire.

VIS is a serial system. Only one rule fires at a time. VIS has an ad hoc mechanism for efficiently selecting which rule to fire next. The rules are bunched into groups (called procedures), within which the rules are totally ordered. Only one procedure is active at a time; and this is controlled by a stack of procedure names (which is considered to be part of SIM).

The Program

A set of rules was designed to model the subject's behavior in the path tasks. Figure 1 shows the procedures into which the rules were partitioned and their interactions. Procedure PLAY controls the basic task cycle of (A) getting a direction from the experimenter and (B) thinking about *It. The latter* (B) consists of (1) creating new knowledge structures about the path, (2) consolidating the newly created structures with the existing knowledge of the path, and (3) describing the path to the experimenter.

New structures are created by RECOGNIZE by interpreting each input direction as a line segment. New expressions are combined with existing expressions to form other new figures. For example, given the two expressions

(LI2 LINF. VEKT SOUTH PI NORTH P2 MOVE NORTH) and
(NEW L21 LINE HORIZ WEST P2 EAST P3 MOVE EAST)

in STM, the corner recognition rule would create a new expression something like

(NEW C123 CORNER P2 WEST L12 NORTH L23 ...)

in STMI. (Note that C123 has, in effect, chunked L12 and L23, which are copied into ITM. Since expression C123 holds the names of the lines of which it is composed, it may be used to retrieve these line expressions from ITM.) C123 might then be used to recognize a box or a step pattern, etc. The other figural concepts used by the subject and the program Include S-shapes, T-shapes, partial boxes, long lines, and crenelations. The hierarchic structure of some of the figures recognized In Problem 1 is shown in Figure 3.

Often, when a partial or extensible figure is recognized, it is expected to be completed or continued. To take a case In point, each step of a step pattern (see, e.g., STEPS in Figure 3} is expected to be completed. When a line is interpreted as part of a step (e.g., LINE 8), the program builds a structure representing the complete step (i.e., CORNER) and marks it incomplete. It is the job of ASSIMILATE to check all new input directions against any expectations. If an expectation is fulfilled, the new direction is quickly processed. (The timings in the subject's

This diagram sketches the simulation program. The boxes represent procedures, which are sub-production systems. The arrows indicate transfers of control between the procedures. The downward pointing arrows are subroutine-type calls, and the horizontal arrows are coroutine-type transfers.

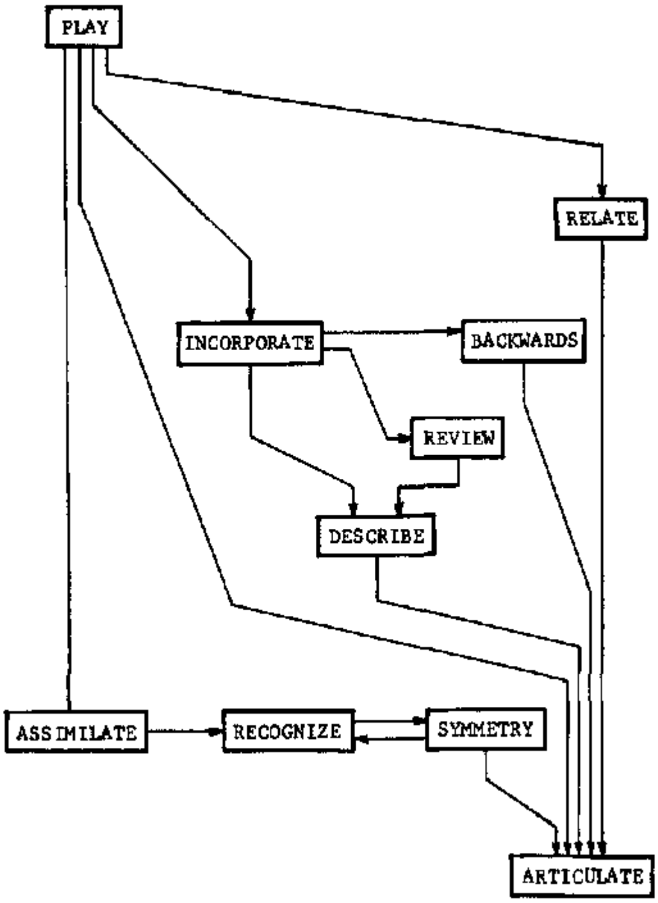

Figure 1. Procedure Structure of the Simulation Program.

protocol support this analysis.) But if an expectation fails, then the program has to reorganize; and, depending on how much commitment there was to the expectation, the program remembers this event as part of the path's structure. For example, ss 154-157 in Figure 2 shows where the subject recalls a previously thwarted expectation; the "LINE missing" in Figure 3 was the expected line, but LINE 9 came instead.

All the recognized figures must be organized into a consistent description of the entire path, and this is done by INCORPORATE. It meshes each new figure into the PATH expression and eliminates redundant and conflicting expressions. All newly incorporated figures are then DESCRIBEd.

Generally, adjacent figures in the path spatially intersect in rather complex ways, but (surprisingly) these intersections do not have to be explicitly described by the program. Instead, each routine which operates on the path description sorts out the figural interactions for itself (by keeping track of common Bub-expressions), Nor does the subject have a complete understanding of all the figural interactions in the path; e.g., the overlap of the STEPS and the BOXes of the S-FIG in Problem 1 (see Figure 3).

After incorporating new figures into the path description, the subject sometimes REVIEWS this description, either forward or BACKWARDS. REVIEW operates at two different levels of detail. The sample protocol in Figure 2 shows one of the program's more detailed review sessions (slightly more verbose than the subject's review, yet quite consistent with the subject). REVIEW simply controls the successive descriptions of the parts of the path. DESCRIBE takes the internal representation of a figure and outputs a stylized "verbal" description. This involves selecting which features to mention and which to ignore in the context of each description.

The internal representation of the path is also used for spatial processing. RELATE tries to find a simple spatial relation (that is, horizontal or vertical alignment) between the current last part of the path and some earlier part of the path. It does this by "spatially" scanning backwards over the path keeping track of where it is relative to the last point.

SYMMETRY is used only in Problem 3 where the subject discovered bilateral symmetry over a large part of that path configuration. It is the most complex routine in the model. It begins with a small mapping between a couple of line segments in the path and then keeps trying to expand this mapping to cover more and more of the path until it exhausts the symmetry. SYMMETRY maps not only line segments of the path, but also its higher level figures. It climbs up and down the hierarchic structure of the path description in its quest to generalize the symmetry. It also drives RECOGNIZE to look for other figures it would like to have.

Many other routines also make heavy use of the hierarchic structure of the internal path description. This can be seen by noting in Figure 1 all the procedures which use ARTICULATE. The hierarchy is a result of the chunking strategy, which is the way the program (and the subject) copes with the limited size of STM. Chunking is a coding operation, and ARTICULATE is the complementary decoding operation. It takes a figure expression and retrieves its sub-figure expressions from I1M and puts them in STM; and it continues recursively with these sub-figures until it reaches unit line segments of the path. ARTICULATE usually recalls figures in their temporal sequence, but it can be used selectively to recall only certain sub-figures. The program spends a great deal of its time in ARTICULATE.

The Representation

The expressions in STM represent what's "on the subject's mind" at any moment in time. The most striking feature about STM is that It contains a mixed bag of information in a homogeneous representation. There are not only figure expressions (the data for this task), but also expressions for verbal input/ output and expressions for control (e.g., goal and context expressions and expressions for place-keeping). Most of the expressions in STM ere obsolete and are just waiting to be pushed out (forgotten).

We began with the issue of the nature of imagery. Does this system have visual imagery? The intermixture of information makes it hard to separate out "visual images", but one interpretation is that an image is represented by the collection of figure expressions in SIM at any one time. Given the dynamic character of STM, the system's "images" are fleeting, changing structures which exist in STM in various degrees of completeness and detail. While "images" are only small pieces of the path description, they may range anywhere over the structure of the path description; and hence they may be either "vague" (high-level) or "vivid" (low-level). At least intuitively, this interpretation has the right flavor.

The internal representation of (the subject's cognitive structure of) the path is an abstract symbolic structure. It is neither "visual" nor "verbal", but both "visual" and "verbal" processes operate on it. The representation is not "visual" in the sense that not all the information that would be immediately available with a visually present drawing of the path is in the representation. It is not "verbal" because a process is needed to transform it into a verbalizable output form.

The path description contains more *:han just spatial information. The description is predominantly hierarchic, as is emphasized in Figure 3. Almost as impoitant as the hierarchic relations are the temporal relations among the parts of the path, which are carefully encoded into all the figure expressions. This extensive involvement with temporal relations is a result, of course, of the nature and demands of the path task. Explicit spatial relations are hardly used at all In the program, and extensive use of the semantics of the figure symbols (e.g., CORNER, BOX, STEPS) is confined to the RECOGNIZE and DESCRIBE routines. ARTICULATE uses only the hierarchic and temporal relatione.

Even the "visualization" routines (RELATE and SYMMETRY) use only the hierarchic and temporal relations to traverse the path. Visual imagery obviously depends on more than Just spatial data. (The subject's behavior in reviewing the path BACKWARDS, for example, clearly reveals his dependence on a hierarchical representation.) What is interesting is how much "imaging" behavior can be exhibited without the explicit use of spatial data. The subject does not exhibit (in the protocol) any kind of knowledge about the path that cannot be explained by this kind of symbolic encoding and the rules to act on it.

This view of the nature and place of imagery in cognitive functioning is a consequence of the "melting pot" view of STM. This should be seen in contrast to Baylor's analysis using an "image space". From my point of view, such a separate "image space" is redundant. There are no pure "image operators"; but rather imagery depends strongly on structural (mostly hierarchic) information. This seems quite natural when one simulates all aspects of the subject's behavior in a task situation. (Baylor only programmed his image operators.) But Brooks' results still present a problem for which I don't yet have an answer. My program does not show any intramodel interference, suggesting that my representation is not yet correct.

To be honest, I must admit that the current program does miss some subtleties of the subject's behavior. In Problem 1 the subject often complains that the path "keeps dragging out" to the east; and by the end of Problem 3 he knows that he is near the beginning of the path, but not exactly where. However, it appears (from an analysis by hand) that these can be explained by simply expanding the internal representation to include some crude spatial relations between the sub-figures of the path description. (This is a job for INCORPORATE.) In any case, all of the subject's knowledge of the path seems to fit comfortably into a symbolic representation.

The Analysis

There are some features of this task and its analysis which distinguish it from previous protocol analyses. The task was rather loose; the subject did not have any specific goals to reach. Hence, the task cannot be characterized as searching in a problem space. (As a result, the subject's high-level goals are not understood or covered by this analysis.) The subject's behavior was not simulated by hand, but rather the analysis was carried out using the production system interpreter. The task was analyzed at a very fine level of detail, using only simple symbolic operations as primitives; no high-level operators are assumed. The simulation operates within the framework of a model of the human memory structure.

The decision to analyze these protocols was opportunistic. It meant confronting the Issues on which the protocols yielded good data and putting a^ide issues which they did not help with. For example, a very interesting issue is whether the articulation operation is a reconstruction based on an encoding plus generic knowledge about figures or whether it is a recall operation from ITM (as in my present program). But the subject makes no regeneration/recall errors and gives no clues as to the exact nature of this process. Either technique, in effect, comes out looking the same. A specific case is the nature of the encoding for an iterative configuration such as a step pattern. Is each step of the pattern encoded individually or is a "typical step" encoded (cf, [13])? The subject's use of the STEPS concept does not give a way to decide this question.

Deferring to the data in this case was worthwhile because these protocols did have a lot to tell us. Our understanding of human cognitive representation is so scanty that this kind of exploratory experiment is still productive. The program is merely a demonstration that symbolic representation can explain some imaging behavior. But now that an operating context for representations has been set up, we are ready to design and use experiments which address some of the specific issues raised. For example, tasks requiring more specific kinds of spatial manipulation (such as Shepard's mental rotation tasks) would help to focus on some of the image mechanisms. This poses some experimental issues on how to collect data rich enough and appropriate for this kind of detailed operational analysis.

The Productions

My simulation program is rather long (more than 170 rules) for a production system. Although the computations it performs are somewhat trivial by a. i. standards, it was a lot of work to create and debug. Since production programming is a new and little understood style of programming, a few words on my experience are in order.

A production system is the most unstructured (anarchic) type of programming language. Any rule can (potentially) fire at any time (and often doesl). Hence, the programmer cannot predict the system's behavior as well as with a structured language. For me this was a problem because I was trying to simulate a known behavior. Thus several ad hoc rules are in the program Just to keep the simulation "on the track". (However, when the program went off and did its own

This segment of the protocol is taken from move 15 of Problem 1 where the subject reviews the whole path. (The numbers at the left are speech segment numbers.)

ss	subject's verbalization	program output
144	Uh, I can review what I have,	(REVIEW)
145	just so I don't forget it.	
146	Uh, you began with a box,	(BOX
	<pause>	
147	uh, going upside-down,	OPEN SOUTH)
	<pause>	
148	uh, up, over, and down.	(MOVE NORTH
		EAST SOUTH)
		(THREE STEPS)
149	And then you did a step,	(STEP TWO)
		(MOVE EAST
		SOUTH)
159	a step,	(STEP THREE)
	<pause>	
151	a step means over, uh, to the east	(MOVE EAST
152	and then down.	SOUTH)
153	[Uh huh.]	
	<pause>	
154	Uh, then you did, uh,	
	<pause>	
155	you began to go on the fourth step	(BEGIN STEP
		FOUR
156	across and/	(MOVE EAST)
157	but you came up.	(EXPECT FAIL
		INSTEAD
		MOVE NORTH)
		(S-FIG)
	<pause>	(BOX OPEN
		NORTH
		(BOX OPEN
		SOUTH)
158	At which point you went over	(MOVE EAST
159	and down, and	SOUTH)
	<pause>	
160	made that S-shape.	
161	and then you dropped down,	(MOVE SOUTH)
		(TWO BY TWO BOX
		OPEN NORTH)
162	went over two,	(MOVE TWO EAST)
163	and up one,	(LINE NORTH)
164	and that's where we are right now.	

Figure 2. Sample Protocol Segment.

This diagram shows some of the program's "cognitive" structure for Problem 1. The program's representation is actually much more complex, but this diagram covers those aspects which are reviewed in Figure 1. The *solid lines indicate hierarchic part-whole relations* between figures; sequential relations are implied by the vertical arrangement.

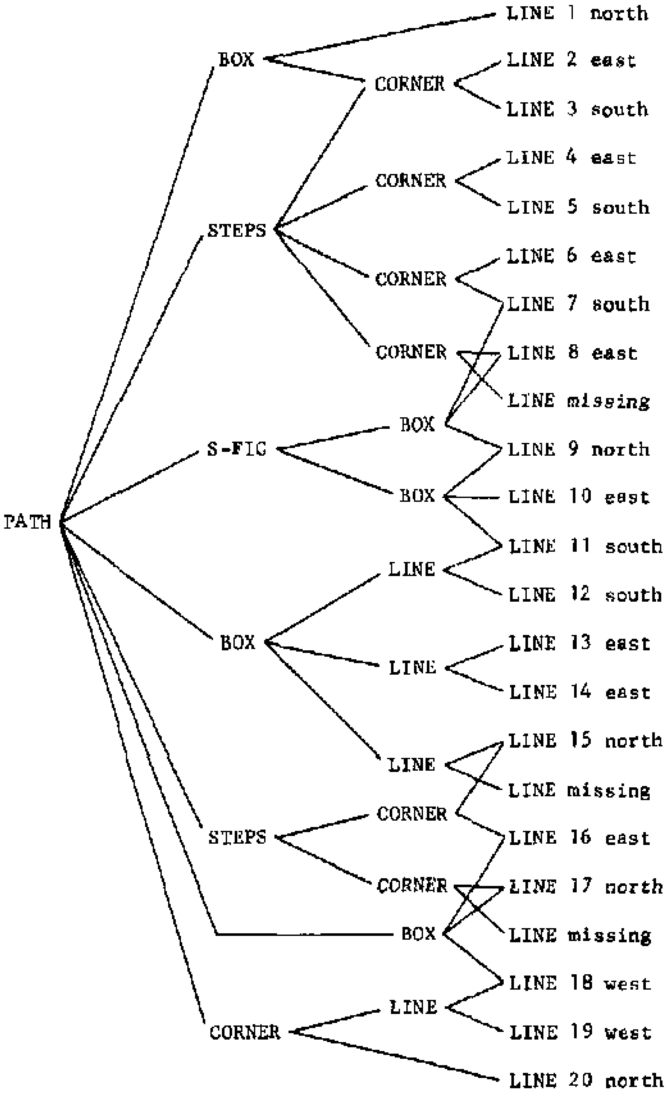

Figure 3. The Program's Path Representation.

thing, it usually behaved quite plausibly.)

But I now think that my system is too structured. The grouping of rules into procedures (though a simple way to get an efficient implementation) violates the spirit of the production iden (and the spirit is what is important here because rules cannot fire anytime. For example, RECOGNIZE and INCORPORATE should be intermingled in some places in the simulation; but they were each programmed independently, and it would be quite tedious to bring them together now.

Production rules are supposed to he independent, but only sytnactically; semantically they are interrelated. Rules can communicate with each other only through STM by using some sort of control language. Thus the condition part of a rule is what expresses its semantic ties to other rules in the system. This is desirable, but there are factors which tend to dilute this explicit semantic expression.

Any structure which is added to the system diminishes the explicitness of rule conditions. This is true not only of the grouping of rules, but also of the ordering of rules. Each rule in an ordered system implicitly assumes that the rules preceding it must fail before it can fire. Thus rules acquire implicit conditions. This makes them (superficially) more concise, but at the price of clarity and precision. Some other method, such as a sorting network, is needed to select rules for firing. (A natural criterion for breaking ties when more than one rule can fire at a time is to use the ordering of expressions in STM to decide which rules have priority.)

Another questionable device in most present production systems (including mine) is the use of tags, markers, and other cute conventions for communicating between rules. Again, this makes for conciseness, but it obscures the meaning of what is intended. The consequence of this in my program is that it is very delicate: one little slip with a tag and it goes off the track. Also it is very difficult to alter the program; it takes a lot of time to readjust all the signals.

The lesson 1 learned from this programming effort is the importance of a clear, explicit language of control in this kind of programming system. The study of a control language is what is needed. The goal of this study would be to develop precise statements (but as general as possible) of the relations between rules so that any rule can fire sensibly in a variety of contexts. Hopefully, this control language would be based on some new insights about procedural interactions and would not be merely a re-expression of the usual control regimes of structured programs. Another way of stating this is in terms of form and function. Whereas a structured programming language expresses the relations between its primitive actions formally (via the syntax of the language), a production system expresses them functionally (via the conditions for the actions).

Acknowledgment

This report is based on my thesis research at Carnegie-Mellon University. 1 would like to acknowledge the help and encouragement of my thesis advisor, Dr. Allen Newell. This research is supported by the Advanced Projects Research Agency of the Office of the Secretary of Defense, contract F44620-70-C0107, and is monitored by the Air Force Office of Scientific Research.

References

[1] George W. Baylor, A Treatise on the Mind's Eye, unpublished Ph.D. Thesis, Carnegie-Mellon University, 1971.

[2] George W. Baylor, "Program and protocol analysis on a mental imagery task", 1JCA1, 1971.

[3] Lee R. Brooks, "Spatial and verbal components of the act of recall", Canadian Journal of Psychology, 1968, vol. 22, pp. 349-368.

[4'] Herbert H. Clark and William G. Chase, "On the process of comparing sentences against pictures", Cognitive Psychology, 1972, vol. 3, pp. 472-517.

[5] Ralph Norman Haber, "How we remember what we see", Scientific American, May 1970, vol. 222, no. 5, pp. 104-112.

[6] George A. Miller, "The magical number seven ...", Psychological Review, 1956, vol. 63, pp. 81-97.

[7] Marvin Minsky and Seymour Papert, "Artificial intelligence progress report", AI Laboratory, MIT, Memo no. 252, 1972.

[8] Ulric Neisser, Cognitive Psychology. Appleton-Century-Crofts, 1972.

[9] Allen Newell, "A theoretical exploration of mechanisms for coding the stimulus", in Coding Processes in Human Memory (Arthur Melton and Edwin Martin, editors), Winston/Wiley, 1973.

[10] Allen Newell and Herbert A. Simon, Human Problem Solving. Prentice-Hall, 1971.

[Il] Roger N. Shepard and Jacqueline Metzler, "Mental rotation of three-dimensional objects", Science, 1971, vol. 171, pp. 701-703.

[12] George Sperling, "The information available in brief visual presentations", Psychological Monograph, 1960, vol. 74, whole no. 498.

f13] Patrick H. Winston, Learning Structural Descriptions from Examples, Ph.D. Thesis, Project MAC Report TR-76, 1970.