SLAC-PUB-429
August 1968
(MISC)

# LINGUISTIC METHODS IN PICTURE PROCESSING - A SURVEY*

by

W. F. Miller and A. C. Shaw**
Stanford Linear Accelerator Center, Stanford, California

(Presented at the 1968 Fall Joint Computer Conference, San Francisco, California, December 9, 10, 11, 1968.)

*Work supported by U. S. Atomic Energy Commission and National Science Foundation, Grant GP-7615.

**Present Address: Department of Computer Science, Cornell University, Ithaca, N.Y.

ABSTRACT

Linguistic Methods in Picture Processing – A Survey

by

W. F. Miller and A. C. Shaw
Stanford Linear Accelerator Center, Stanford, California

This paper surveys research in linguistic methods for describing and processing pictures. The rationale for a linguistic approach to picture processing is first reviewed. A general linguistic picture processing model is then presented as a basis for discussion in the survey; the central idea within the model is that of a formalism for picture description. A number of research efforts are described in terms of their accomplishments, limitations, and potential usefulness. While experimental in nature, the surveyed works provide evidence that complex richly-structured pictures can be successfully processed using linguistic methods. Several common characteristics and directions for future research are indicated in the concluding section.

Linguistic Methods in Picture Processing - A Survey

W. F. Miller and A. C. Shaw
Stanford Linear Accelerator Center, Stanford, California

## I. INTRODUCTION

By "picture processing" we mean the analysis and generation of pictures by computer, with or without human interaction; this definition includes both computer graphics and digital pattern recognition.

A number of people have advocated that picture processing problems be attacked with linguistic methods; perhaps the strongest early exponents were Narasimhan[1] and Kirsch.[2] The basic idea was to extend the notions of syntax and semantics to n-dimensional patterns ($n > 1$) and then apply some adaptation of the techniques of natural and artificial language processing. Several researchers have attempted to develop this concept during the last few years. While the work is still experimental, several practical uses have been demonstrated and ideas seem to be emerging that could form the basis of a picture theory.

This paper surveys research in linguistic methods for describing and processing pictures. The next section discusses the rationale and application area for a linguistic approach. We then present a general linguistic picture processing model as a basis for the survey discussion. The central idea within this model is that of a formal picture description. The survey itself is contained in section IV. In the concluding section we extract some common features and difficulties, and indicate directions for future research.

## II. MODELS FOR PICTURE PROCESSING

The term "model" denotes the general framework or "paradigm"[3] within which workers pose and solve problems. Until recently, most theoretical work in picture analysis has, either implicitly or explicitly, been based on the receptor/categorizer model (RCM) described in Marill and Green.[4]

The analysis of pictures (or pattern recognition) proceeds as follows within the RCM: A picture is first reduced to a "feature set" by the receptor; this is a set of quantities which may range from the raw digitized values at one extreme to the results of a complex feature extraction process on the other. The feature set is then assigned to one of a finite number of classes or patterns by the categorizer. The assignment is the recognized pattern class to which the picture supposedly belongs. Most of the theory has dealt with the problem of categorization or classification. The principal technique is one of treating the feature or measurement set as a point in a multidimensional space. The task of the categorizer then becomes one of partitioning the space so that measurements from pictures belonging to the same pattern class are "close" (according to some metric) and measurements from pictures of different classes are far apart. (Sebestyen[5] and Nilsson[6] are references for the RCM).

The RCM is the basis for a number of recognition systems, notably in character recognition.[7] The model fails to be useful for analyzing complex pictures where the structure and interrelationships among the picture components are the important factors. To illustrate this point in a simple setting, consider the one-dimensional pattern recognition task required of a programming language translator. One purpose of the syntax analysis phase of the compiler is to categorize an input program into one of two mutually exclusive classes – the class of syntactically correct programs and its complement. Theoretically, one can

envision a receptor which produces a feature vector from an input program; the categorizer then determines in which of the two possible subspaces the feature vector lies. While this can be done in principle, it is never considered seriously because of the complexities involved; for example, what is the feature set for a program? Even if this approach were practically feasible for program classification, it would not produce the most important byproduct of a successful analysis i.e., a description of the structure of the input program.

Richly-structured pictures that are difficult, if not impossible, to analyze within the RCM include those produced in particle detector chambers by high-energy particle physics reactions; text and standard two-dimensional mathematical notation (not isolated characters); line drawings, such as flow charts, circuits, and mechanical drawings; and complex biomedical pictures. What is required in these examples is a description of the pictures in which the meaningful relations among their subparts are apparent. The appropriate place to apply the RCM is for the recognition of the basic components of the pictures. In a series of papers, Narasimhan[1, 8, 9, 10] has forcefully stated this case:

> "Categorization, clearly, is only one aspect of the recognition problem; not the whole of it by any means. It is our contention that the aim of any recognition procedure should not be merely to arrive at a 'Yes', 'No', 'Don't know' decision but to produce a structured description of the input picture. Perhaps a good part of this confusion about aims might have been avoided if, historically, the problem had been posed as not one of pattern recognition but of pattern analysis and description."[1]

Much of the research in computer graphics* has been concerned primarily with data structures[11] and command and control languages. Picture descriptions are embedded in the data structures; in fact, the data structure is the description. This could be viewed as a linguistic specification of a picture since the structure (syntax) and values or interpretations of each structure (semantics) are explicitly contained in the data structure in most cases. However, the processing (analysis or synthesis) of the pictures is not directed by the data structure description but rather towards them through the command and control languages.

---

*"Computer graphics" has usually referred to that set of techniques for computer processing of pictures using on-line displays and plotting equipment.

---

In this survey we shall consider only those works where some attempt is made to describe pictures and classes of pictures, and use these descriptions to direct the processing. The analogy to linear language processing is evident and hence the term "linguistic model"** is employed.

---

**Narasimhan[1] first used this term as applied to picture processing.

---

## III. A GENERAL LINGUISTIC PICTURE PROCESSING MODEL

The linguistic model for picture processing[12] is comprised of two parts:

1. a general model within which pictures may be described (i.e., a meta-description formalism), and
2. an approach to the analysis and generation of pictures based directly on their descriptions.

The description, D, of a picture, $\alpha$, will consist of two parts – a primitive or terminal symbol description, T, and a hierarchic description H. T specifies the elementary patterns in the picture and their relationship to one another and H describes groupings of the elements into higher level structures. This can be written $D(\alpha) = (T(\alpha), H(\alpha))$. T and H, in turn, each have a syntactical (or structural) component $T_s$ and $H_s$, and a semantic (interpretation or value) component $T_v$ and $H_v$. That is,

$$T(\alpha) = (T_s(\alpha), T_v(\alpha))$$
$$H(\alpha) = (H_s(\alpha), H_v(\alpha)) \quad .$$

$T_s(\alpha)$ names the elementary component classes or primitives in $\alpha$ and their relationship to one another; $T_v(\alpha)$ gives the values or meaning of the primitive components of $\alpha$. The primitives in $T_s(\alpha)$ will denote classes; let $\mathscr{P}(T_s)$ be the set of all pictures with primitive structure $T_s$. We present a simple example of a primitive description T.

Example 1

Let $\ell$ name the set of all straight line segments and c name the set of all circles. $\ell$ and c are picture primitives. Let $\odot$ denote the geometric relationship of intersection. Then, if a picture $\alpha$ contains a line segment $\alpha_1$ intersecting a circle $\alpha_2$, its primitive description $T(\alpha)$ might be:

$$T_s(\alpha) = \ell \odot c \qquad T_v(\alpha) = (v_\ell(\alpha_1), v_c(\alpha_2)) \quad ,$$

where $v_\ell(x)$ is the pair of endpoint coordinates of the line x and $v_c(x)$ is the center coordinates and radius of the circle x. $\mathscr{P}(\ell \odot c)$ is the set of all pictures consisting of a line segment intersecting a circle.

Consider a set of rules or grammar $\mathcal{G}$ generating a language $\mathcal{L}(\mathcal{G})$ whose "sentences" are primitive structural descriptions. Then, $\mathcal{G}$ is said to describe the picture class $\mathcal{P}_{\mathcal{G}} = \bigcup_{T_s \in \mathcal{L}(\mathcal{G})} \mathcal{P}(T_s)$. For a given picture $\alpha \in \mathcal{P}_{\mathcal{G}}$, the hierarchic structural description $H_s(\alpha)$ is the ordered set of rules of $\mathcal{G}$ that were used to generate $T_s(\alpha)$; that is, $H_s(\alpha)$ is the "linguistic" structure or parse of $T_s(\alpha)$ according to $\mathcal{G}$. A one-to-one correspondence exists between the elements of a set $\mathcal{I}$ of semantic or interpretation rules and the elements of $\mathcal{G}$. $H_v(\alpha)$ is defined as the result of obeying the corresponding semantic rule for each rule of $\mathcal{G}$ used in $H_s(\alpha)$.

Example 2

Let $\mathcal{G}$ be the phrase structure grammar[13]:

$\mathcal{G} = \{ LC \rightarrow L,\ LC \rightarrow C,\ LC \rightarrow L \odot C,\ L \rightarrow \ell,\ C \rightarrow c \}$. Then $\mathcal{L}(\mathcal{G}) = \{ \ell, c, \ell \odot c \}$ and $\mathcal{P}_{\mathcal{G}} = \mathcal{P}(\ell) \cup \mathcal{P}(c) \cup \mathcal{P}(\ell \odot c)$. We interpret the terminal symbols $\ell$, c, and $\odot$ as in Example 1 and let

$$\mathcal{I} = \{ v_{LC} := v_L,\ v_{LC} := v_C,\ v_{LC} := \text{xsect}(v_L, v_C),\ v_L := v_\ell,\ v_C := v_c \}.$$

The $k^{th}$ rule of $\mathcal{I}$ corresponds to the $k^{th}$ rule of $\mathcal{G}$ for $k = 1, \ldots, 5$. Within a rule, $v_i$ designates the value associated with the syntactic unit i in the corresponding grammar rule; xsect is a function that computes the intersection(s) of a line with a circle, and $v_\ell$ and $v_c$ are defined in Example 1. If $T_s(\alpha) = \ell \odot c$ for a given $\alpha \in \mathcal{P}_{\mathcal{G}}$, $H(\alpha)$ could be represented by the simple tree of Fig. 1, where $\alpha = \alpha_1 \cup \alpha_2$, $\alpha_1 \in \mathcal{P}(\ell)$, $\alpha_2 \in \mathcal{P}(c)$, $v_\ell = v_\ell(\alpha_1)$, and $v_c = v_c(\alpha_2)$.

It is important to emphasize that the "meaning" of a picture will be expressed in both its primitive and hierarchic descriptions. Thus, several grammars may

be used to generate the same class of primitive descriptions, but the hierarchic descriptions, and hence the meaning, may be different for different grammars. Even more generally, the same picture class may be described by totally different primitive and hierarchic descriptions; the intended interpretation of the picture dictates its description.

With the description model, our approach to picture processing can now be formulated:

1. The elementary components or primitives which may appear in a class of pictures are named and defined.
2. The picture class is described by a generative grammar $\mathcal{G}$ and associated semantics $\mathcal{I}$ .
3. A given picture $\alpha$ is then analyzed by parsing it according to $\mathcal{G}$ and $\mathcal{I}$ to obtain its description $D(\alpha)$; that is, $\mathcal{G}$ and $\mathcal{I}$ are used explicitly to direct the analysis.

   Conversely, a picture $\alpha$ is generated by executing its description $D(\alpha)$.

Descriptions are then not only the results of an analysis or the input to a generation, but they also define the algorithms that guide the processing. This approach provides a framework in which picture processing systems may be implemented and theoretically examined. The arguments for treating analysis and synthesis problems together, i.e., using a common description scheme, are generality, simplicity, and the universal use of common description languages in science. We also note that most picture analysis applications have (and need) an associated generative system and vice versa; there are also many situations where both a synthesis and an analysis capability are equally important, for example, in computer-aided design.

Syntax-directed translation of programming languages[14, 15] can be interpreted within our model as the analysis of patterns of linear strings. In this case, the primitive description is obtained immediately – the input program corresponds to $T_S$ and the meaning of the basic symbols of the language to $T_V$. The grammar $\mathcal{G}$ is generally a BNF grammar plus some constraints on the use of identifiers; the semantics $\mathcal{I}$ is most often a set of code-generating rules. The analysis of a well-formed program yields the syntactic structure of the program and an equivalent program in some other language.

We find it most illuminating to evaluate picture processing research within the framework of the above model. In each case, the various components of the particular descriptive scheme – $T_S$, $T_V$, $H_S$, and $H_V$ – are extracted and discussed in terms of their power and limitations. We are interested in the description mechanism both as a language of discourse about pictures and as a driver for analysis or generation systems.

## IV. THE SURVEY

The literature survey of Feder[16] covers the few basic developments up to and including 1965; since then, there has been a relatively large surge of activity.

### Early Developments

There are several early works that explicitly utilized primitive descriptions. Grimsdale et al.,[17] produced geometric descriptions of hand-drawn line figures, such as alphabetic characters; the description consisted of an encoded list of the picture curves, their connectivity, and geometric properties. Sherman[18] reduced a hand-printed letter to a graph, and then built a character description out of the topological and geometric features of the abstracted picture. Neither $T_S$ nor $T_V$ is defined formally in the above examples; picture analysis (recognition) occurs by comparing or matching picture descriptions with descriptions of standard patterns.

Eden[19, 20] presented a formal system for describing handwriting. His primitive elements are a set of basic "strokes" or curves; the value of each stroke is a point pair (the endpoints) and a direction. Eden gives a set of rules $\mathcal{G}$ for concatenating or collating strokes to form letters and words. The description $T_s$ of a word of handwriting is then a sequence of n-tuples of strokes, each n-tuple representing a letter. This is one of the first works where the author recognizes the benefits of a generative description:

> "Identification by a generative procedure leads to a clear definition of the set of permissible patterns. The class of accepted patterns is simply the set which can be generated by the rules operating on the primitive symbols of the theory."[20]

Eden did not report any attempts at using his scheme for recognition purposes; however, his descriptions were used for generation.

In Minsky,[21] we find one of the earliest arguments for the use of "articular" or structured picture descriptions in pattern recognition. Minsky suggests a description language consisting of expressions of the form (R, L), where L is an ordered list of subpictures or figures related to one another by the relation R. For example, (→, (x, y)) might indicate that the figure y is to the right of x. Expression composition within the elements of the list L permits the description of complicated structures; using the above notation, (→, ((→, (a, b)), c)) means that b is to the right of a, and c is to the right of the subpicture containing a and b. Although it is not explicitly linguistic, this work has influenced several later efforts (see discussion under Evans).

Narasimhan

The pioneering work in suggesting and applying a linguistic model for the solution of non-trivial problems in picture processing was done by

Narasimhan.[1,8,9,10,22,23] He first proposed a general linguistic approach in 1962, calling it a "linguistic model for patterns"; he has since experimented with it in the analysis of bubble chamber photographs using a parallel computer,[1,9,10,22] and in the generation of "handprinted" English characters.[10,23] Narasimhan restricts his model to the class of pictures containing only thin line-like elements.

We first discuss the analysis model in Narasimhan's 1962 paper.[1] Here, $T_s$ is a list of the "basic sets" and their connectivity. Basic sets refer to neighborhoods on the picture having specified topological properties, for example, the neighborhood about the junction of two lines or the neighborhood about an endpoint of a line. Two sets are said to be connected if there exists a "road" or line-like element between them. $T_v$ is the value of the sets (their topological meaning) and the geometry of the connecting roads. An informal set of rules $\mathcal{G}$ then describes how strings of connected sets may be combined into other strings and phrases; phrases are of the form: <name> ( <vertex list> ), for example, ST(1, 2, 3), where the <vertex list> labels those points that may be linked to other phrases. Finally, there are additional rules of $\mathcal{G}$ for combining phrases into sentences. The hierarchic description $H_s$ of a picture is a list of sentences and phrases. Analysis proceeds from the "bottom up", first labeling all points as basic sets or roads, then forming phrases and, last of all, sentences.

Narasimhan does not define a general form for either $\mathcal{G}$ or the description D. In the bubble chamber application, the hierarchic system of labeling imposed by $\mathcal{G}$ is slightly different than above, starting with points at the most primitive level; $\mathcal{G}$ is implicitly defined by the computer program itself. On the other hand, the generation of English "hand-printed" characters is explicitly directed by a finite-state generative grammar $\mathcal{G}$ and an attribute list $\mathcal{I}$, the latter specifying some geometric properties of the characters, for example, position, length, and

thickness. The primitives are simple geometric forms, such as straight lines or arcs; the definition of each primitive includes a set of labeled vertices to which other primitives may be attached. Productions or rewriting rules in $\mathcal{G}$ are of the form:

$$S(n_S) \rightarrow S_1 \cdot S_2(n_{S_1S_2}; n_{S_1S}; n_{S_2S}) \quad ,$$

where $S_1$ is a terminal symbol (primitive name) or non-terminal symbol (phrase name), $S_2$ is a terminal symbol, S is a non-terminal symbol – the defined phrase – , $n_{S_1S_2}$ is a list of the nodes of concatenation between $S_1$ and $S_2$, $n_{S_1S}$ and $n_{S_2S}$ define the correspondence between the nodes of $S_1$ and $S_2$ and those of S, and $n_S$ is a node list labeling the nodes of S. Figure 2 illustrates Narasimhan's rewriting rules for generating the letter "P", the primitives required, and the generated letters. All nodes of possible concatenation must appear in the description; this is cumbersome for simple pictures such as the English alphabet, and might be unmanageable for more complex pictures. The system can only describe connected pictures and some other mechanism is required when dealing with pictures whose subparts are not connected. This scheme has been used successfully as part of an experimental system for the computer generation of posters.[23] To our knowledge, it has not been applied to other picture classes.

Kirsch

Kirsch,[2] in a stimulating article, argues that the proper way to view picture analysis is within a linguistic framework. Following this line of thought, he poses several problems: How does one

1. express picture syntax or structure,
2. generalize the idea of concatenation to several dimensions,

3. describe geometric relations among picture components,
4. do syntax analysis of pictures, and
5. define picture primitives?

Kirsch gives a two-dimensional context-dependent grammar for $45^{o}$ right triangles generated in a plane divided into unit squares; this is suggested as an illustration of the possible form of picture grammars. Figure 3 contains a sample production and a derived triangle. Here, $T_S$ is a two-dimensional $45^{o}$ right triangle with labeled unit squares (the primitives); $T_V$ is the meaning of the labels. There is no semantic portion corresponding to the grammar. As Kirsch admits, it is not evident how this approach may be generalized for other pictures; it is also a debatable point whether context-sensitive grammars are desirable since the analysis would be extremely complex. More recently, Lipkin, Watt, and Kirsch[24] have argued persuasively for an "iconic" (image-like or picture) grammar to be used for the analysis and synthesis of biological images with a large interactive computer system; however, the search for suitable iconic grammars continues. The work of Kirsch and his colleagues is notable for their clear and early recognition of the importance of a linguistic approach to picture processing problems and for their detailed enumeration of some of the difficulties.

Ledley

Ledley[25] and Ledley et al.,[26] employed a standard BNF grammar to define picture classes. Their published method for the analysis of chromosomes[26, 27] illustrates this approach. Here, Ledley's "syntax-directed pattern recognition" is embedded in a large picture processing system that searches a digitized picture for objects, recognizes the primitives of an object, performs a syntax analysis of the object description, and finally computes further classifications

and some statistics on all the chromosomes found. The object primitives consist of five types of curves from which chromosome boundaries can be generated. An edge-following program traces the boundary of an object in the picture and classifies each boundary segment into one of the primitive classes; since the boundary is a closed curve, a linear string or ordered list of its segment types is sufficient for the description $T_s$. If $T_s$ represents a chromosome, the parse $H_s$ will contain a categorization of it as, for example, submedian or telocentric in type; otherwise the parse fails, indicating the original object was not a chromosome. Figure 4 contains samples from the chromosome syntax, examples of the basic curve types, and some chromosome descriptions. Ledley and Ruddle[27] state that the human complement of 46 chromosomes can be processed in about 20 seconds (on an IBM 7094) using this system – a factor of 500 as compared to manual methods –, but no data is given on the quantity of pictures examined and error rates, or how their methods compare with others, for example, chromosome classification by moment invariants.[28] Ledley's work is an example of a direct application of artificial language analysis methods to picture classification. It is difficult to generalize this approach to figures other than closed curves unless relational operators are included as part of $T_s$; in the latter case, the most difficult task is obtaining $T_s$, not parsing the resulting string.

Guzmán

Guzmán[29, 30] describes pictures consisting of sets of isolated points and concatenated straight line segments using a figure description language (FDL). The primitive syntax $T_s$ is given in FDL by listing every node in the figure and its immediate neighbors, and adjoining to this an arbitrary property list; $T_v$ is a list of the actual coordinates of each node. Figure 5 contains two

possible descriptions of an isosceles triangle and one of a quadrangle and a rectangle. Hierarchic descriptions and the equivalent of a grammar may be specified in FDL by assigning names to both primitive descriptions, and sets of names and descriptions. This is illustrated at the bottom of Fig. 5, where a POLY is defined as either a RECT or an ISOS1. Several figures may be concatenated to form new ones by listing in a =TIE= statement the nodes of concatenation. The FDL language is used to drive some general scene analysis programs. A given scene is first preprocessed to produce a symbolic description in terms of points forming line segments and isolated points. A scene analysis program then accepts a series of "models" described in FDL and searches the scene for all or some instances of the models. Experiments with the system have served to pinpoint a number of extremely difficult problems associated with the analysis of two-dimensional projections of three-dimensional objects. While restricted to concatenated straight line segments and isolated points, the FDL language has some very desirable features. Chief among these is the ability to define "open" or bound variables (X, Y, and A1 in Fig. 5) in the property list; this allows an elegant description of the relations among picture components.

Evans

The earlier work of Evans[31, 32] on solving geometric-analogy intelligence test problems employed picture description methods similar to those suggested by Minsky.[21] Recently, Evans[33] has developed a linguistic formalism for picture description and an associated pattern analyzer that is driven by a "grammar" $\mathcal{G}$ written in the formalism. The syntax of a class of pictures is given by a set of rules, each of which has four components: (L R P I) (our notation). An example of a rule that we will use in the discussion below is:

(TRIANGLE (X Y Z) ( (VERTEX X) (VERTEX Y) (VERTEX Z) (ELS X Y)
(ELS Y Z) (ELS X Z) (NONCOLL X Y Z) ) ( (VERTICES (LIST X Y Z) ) ) )

The first component, L, names the construct or pattern whose components are defined by R and P; in the example, the pattern TRIANGLE is named. R is a list of "dummy" variables, one of which is associated with each constituent of the defined pattern. P is a list of predicates which names the pattern type represented by each dummy variable, and describes the relationships that must exist among these patterns. X, Y, and Z are named as type VERTEX; ELS is a predicate which tests for the existence of a line segment between two points, and NONCOLL tests for noncollinearity among 3 points. The last part I of the syntax rule can specify any computation over the properties of the pattern components; during analysis, it assigns the result to the new construct defined by the rule. After a successful analysis TRIANGLE will have attached to it the name VERTICES followed by a list of the values of X, Y, and Z. These attached properties can then be used by predicates in subsequent syntax rules. In terms of our model, the I component can be viewed as part of the syntax in some instances or as an interpretation or semantic rule of $\mathscr{I}$ in others.

Evan's pattern analyzer assumes that a picture is first preprocessed to produce a list of its primitive elements and their properties; this is the primitive description T. The pattern analyzer (a LISP[34] program) accepts a preprocessed picture and a grammar, and parses the picture to produce hierarchic descriptions of all patterns satisfying the grammar; the library of predicates may first have to be extended if new relational predicates appear in the grammar. While the description and analysis systems are very general, they have only been tested on simple examples and it is too early to predict how useful they will be.

Shaw, Miller, and George

In the Shaw papers[12, 35] a picture description language (PDL) is presented and applied. PDL is a language for expressing the primitive structural description $T_s$ of a picture. The basic components or primitives may be any pattern having two distinguished points, a tail and a head; primitives can be concatenated together only at these points. The PDL language can describe the concatenations among any connected set of primitives. By allowing the definition of blank (invisible) and "don't care" primitives, a large class of pictures may be described in terms of concatenations and simple relations among their primitive elements; these include photographs produced in high energy particle physics experiments, characters, text, flow charts, and line drawings of all varieties.

Figure 6 illustrates the use of PDL to describe a simple "A" and an "F". For each primitive, the figure contains its class name, a typical member, and an arrow pointing from its tail to head; for example, h denotes the set of all horizontal line segments of a restricted length, with tail at the left endpoint and head at the right endpoint. h can be defined more precisely either theoretically or pragmatically by an equation, an attribute list, a recognition program, or a generation program. The tree beneath the "A" indicates how the letter is generated from its description. The operators +, x, and * describe particular combinations of tail/head concatenations of their operands. Each PDL expression, and the pictures they describe, has a tail and head defined respectively as the tail of the first element and head of the last element in the expression. Thus $(S_1 + S_2)$ has a tail equal to the tail of $S_1$ and a head equal to the head of $S_2$; the "+" describes the concatenation of the head of $S_1$ to the tail of $S_2$. One more binary operator (-), a unary tail/head reversal operator (~), and a "rewriting" convention complete the description scheme. PDL has a number of useful formal properties

that permit descriptions to be transformed into more convenient forms for processing, and forms the basis of a picture calculus discussed in Miller and Shaw.[36] The primitive semantic description $T_V$ consists of a list of the primitives and their attributes.

A hierarchic structure is imposed on a class of pictures by means of a restricted form of context-free grammar $\mathcal{G}$ generating sentences in PDL. Figure 7 contains several productions from a flow chart grammar for a small ALGOL-like language. The tail and head of each primitive are labelled t and h respectively. The line segments with arrow heads leading from enter, fn, and cond may be any sequence of concatenated segments thus allowing the head of these primitives to be placed anywhere in a picture relative to the tail. The box in fn is a function box and pred represents a predicate or test. cond may be either the true or false branch of the predicate; the initial blank (dotted) part at its tail carries it to one of the vertices of the diamond. In the syntax, the / and superscript labels indicate "rewriting" so that both appearances of TEST in the STEPUNTIL rule refer to exactly the same entity. The hierarchic structural description $H_S$ is defined as the parse of $T_S$ according to $\mathcal{G}$ ; no mechanism for attaching arbitrary semantics to $\mathcal{G}$ has been developed yet.

A goal-oriented picture parser (analyzer) (Shaw[12]) accepts a pattern recognition routine for each primitive class and a grammar, and uses the latter to direct the recognizers over pictures and produce their primitive and hierarchic descriptions; tail and head pointers are moved over the two or three-dimensional picture space in a manner analogous to the movement of a string pointer in linear language analysis. An implemented system has been applied to the analysis of some digitized spark chamber film. Each picture consisted of a data box with 22 identification digits; 4 fiducial markers ("X"s); and 2 views of 6 spark

chambers containing sets of isolated and collinear sparks. 39 syntax rules were used to describe the possible contents of all pictures. The description $D(\alpha)$ of each picture $\alpha$ was produced in approximately 7 seconds on an IBM 360/50. With the picture parser available, it took less than 2 man months to put together the spark chamber system. The spark chamber application, even though experimental, has demonstrated certain pragmatically useful advantages of the above methods. These may be summarized as follows: There can be significant simplifications in implementing and modifying picture analysis systems and one need not pay an exorbitant price in computer processing time when compared with the more ad hoc systems in use in various physics laboratories.

George[37] and George and Miller[38] employ PDL as the basis of an interactive graphics system. Pictures are generated and modified on-line by manipulating PDL descriptions. Pictures can be stored and retrieved by assigning names to their descriptions; the picture data structure is the PDL description itself so that the machine always contains a structured representation. Any changes to a named subpicture are immediately reflected in all pictures that refer to it as a component. The chief limitations of the descriptive scheme are the restricted set of relations that may be expressed, the practical constraints resulting from only two points of concatenation for a primitive, and the absence of a general mechanism for hierarchic semantics.

Anderson

Anderson[39, 40] syntactically analyzes standard two-dimensional mathematical notation after the primitive elements or characters have been classified by conventional pattern recognition techniques. The value $T_v$ of a primitive is its name and 6 positional coordinates: $X_{min}$, $X_{center}$, $X_{max}$, $Y_{min}$, $Y_{center}$, $Y_{max}$, where ($X_{min}$, $X_{max}$, $Y_{min}$, $Y_{max}$) define the smallest enclosing

rectangle of the character and the point ($X_{center}$, $Y_{center}$) is its typographic center. Each syntax rule consists of four structural parts (elements of $\mathcal{G}$) and one semantic part (element of $\mathcal{I}$). Figure 8 contains a typical syntax rule. The meaning of the notation is as follows:

Si: the $i^{th}$ syntactic unit of the right part of the rule.

Pi: a partitioning predicate that Si must satisfy. $c_{ij}$ is the $j^{th}$ positional coordinate of Si; the positional coordinates above are numbered from 1 to 6 so that $c_{13}$ represents the 3rd coordinate ($X_{max}$) of syntactic unit S1. $c_{oj}$ refers to the $j^{th}$ coordinate of an arbitrary character in the syntactic unit.

R: a predicate testing the spatial relationship among successfully parsed elements of the right part of the syntax rule.

Ci: each higher level structure (syntactic unit) is given 6 positional coordinates similar to those of a primitive. Ci, i = 1, . . . 6, defines the 6 coordinates assigned to the left part of the syntax rule in a successful parse.

M: the semantic rule indicating an action to be taken or the meaning to be given to the rule.

The mathematical expression $\frac{a^2 + b}{c}$ satisfies the syntax of "term" in the figure; the typographic center is (C2, C5) which is defined in the replacement rule as ($c_{22}$, $c_{25}$), the center of the primitive "horizline". Anderson has described several non-trivial classes of pictures in this notation, including two-dimensional arithmetic expressions, matrices, directed-graphs, and a proposed form for a two-dimensional programming language.

A top-down goal-directed method is used for analysis; the basic idea is to use the syntax directly to partition the picture space into syntactical units such

that the predicates Pi and R are satisfied. The analysis algorithm has been implemented in several experimental systems and tested with hand-printed arithmetic expressions in an interactive mode. He assumes that the primitive characters are correctly classified by recognition routines and that the expressions satisfy some reasonable constraints on their form, for example, the limits above and below an integral sign must not extend further to the left than the leftmost edge of the integral sign. Simple expressions can then be parsed successfully in a reasonable amount of computer time. The expression $\int_1^N |x|\,dx$ takes approximately 5 seconds to analyze on an IBM 360/50 with an unoptimized PL/I version of the general system; a program optimized especially for mathematical notation and running on the Digital Equipment Corporation PDP-1 takes less than half a second to recognize the expression $\frac{1 + Z - Z^2}{1 - \frac{1}{Z}}$.

One of the virtues of Anderson's model is the provision for arbitrary predicates to test spatial relationships as part of the syntax. In order to handle this generality, the analysis algorithm must test a large number of possible partitionings of the picture space before rejecting an inapplicable syntax rule. However, in the case of mathematical notation, increased efficiency can be obtained by taking advantage of its normal left-to-right flow. While adequate for driving a picture analyzer for a restricted class of pictures, the descriptive scheme does not appear suitable for synthesis problems. (Anderson argues that these two aspects of picture processing are fundamentally different and should be treated by entirely different methods.) Anderson has demonstrated the feasibility of interactive mathematics using the above concepts; he concludes, and the authors concur, that future efforts could be directed towards engineering such systems.

Other Works

We conclude the survey by noting several other works which employ either linguistic methods or closely related techniques.

Clark and Miller[41] use the language of graph theory to describe spark linkages and the topology of physics "events" appearing in spark chamber film. These descriptions are embodied in computer programs that apply elementary graph theory to assist in the decision-making process and perform the film analysis. The primitive elements of the pictures are sparks; a multi-list structure provides the description $T_S$ and $T_V$ of the spark connectivities. Hierarchic descriptions result from combining sparks according to their geometric and graph properties to form tracks and events. While an explicit linguistic approach is not employed, the underlying graph model acts as a formal description language, much as in the work of Sherman and Narasimhan. The above program formed the basis for a practical production system that was used for several physics experiments.

Clowes[42] employs a set $\mathcal{G}$ of Boolean functions on pictures to define the syntactic classes for hand-written numerals; the successive execution of these functions from the bottom up serves to analyze and describe the pictures. More recently, he[43] has been working on a scheme based on transformational grammars and Chomsky's model for natural language syntax. Other efforts which are explicitly linguistic in nature include Feder,[44] Watt,[45,46] Inselberg and Kline,[47] Inselberg,[48] Breeding,[49] Knoke and Wiley,[50] and Nir.[51]

A related area of research has been pursued by Kirsch,[2] and, more recently, by Coles[52] and others. Natural language statements about pictures are translated into some formal notation, usually the predicate calculus; the predicate calculus statement then describes a set of pictures – those for which

the statement has the truth value of true. The natural language statement "Each polygon smaller than a black triangle is a square," could be translated into the predicate calculus as "$(\forall x)\,(P(x) \wedge (\exists y)\,(B(y) \wedge T(y) \wedge Sm(x, y)) \supset Sq(x))$" which may be read as "for all x, if x is a polygon and if there exists a y such that y is black and y is a triangle and x is smaller than y, then x is a square." The predicate calculus expression directs a picture analyzer to determine the truth value of the statement with respect to a given picture. By these methods, Coles[52] is able to recognize some fairly complicated electric circuits and chemical molecules drawn on a computer-controlled display. One of the aims of this research is to provide interactive question-answering systems with a pictorial data base. A principal, and extremely difficult, problem is that of translating natural language input to the formal notation. In terms of our description model, the predicate calculus statement is the primitive structural description $T_S$; hierarchic descriptions do not exist in these schemes.

## V. CONCLUSIONS

As this survey indicates, there has been a great deal of research in picture description methods and associated processing systems; most of this is recent and is still in progress. A principal reason for this research has been the lack of adequate techniques for dealing with complex richly-structured pictures; we feel that relevant techniques are now emerging. All of the reported work is experimental in the sense that, to our knowledge, there do not exist any "production systems" that employ linguistic methods to any large extent. However, in several instances, notably in the work of Anderson[40] and the authors,[12,36] the benefits and practicality of these methods have been demonstrated.

With the exception of Kirsch's triangle example,[2] all of the descriptive schemes are basically linear. One suspects that the development of explicit

two- and three-dimensional picture languages would lead to much greater insight into picture processing problems (and, quite possibly, human perception); we are still waiting for a breakthrough in this direction. Regardless of the detailed forms of the descriptive notation and grammars in the various systems, each syntax rule essentially specifies a list of patterns and a set of relations satisfied by them. Practical analysis systems will clearly have to restrict the class of pictures and the types of relations that may exist among the elements of a picture. This is entirely analogous to the linear language situation where extremely efficient parsers exist when the grammar form and class of languages are restricted, for example, in simple precedence grammars.[53] One of the most difficult problems in pattern analysis is the classification of primitive patterns; in many situations, ambiguities and recognition failures can be resolved by examining the picture field surrounding the pattern in question, i.e., by using contextual information. Most of the surveyed works assume that the primitive elements have been classified before entering the analysis; in Shaw,[12] the grammar $\mathcal{G}$ directs the primitive recognizers about the picture and assists the classification process by using the contextual information embedded in $\mathcal{G}$. Work in this direction should be pursued further. Finally, we note that, with the exception of Eden,[19, 20] Narasimhan,[10, 23] and Shaw, Miller, and George,[12, 35, 36, 37, 38] the research has been concerned only with the analysis of pictures. As we argued in section III, there are advantages in treating both analysis and synthesis problems within the same formalism. However, picture generation using formal description schemes has not yet been examined in depth and remains a fruitful area for future work.

## VI. REFERENCES

[1]Narasimhan, R., "A linguistic approach to pattern recognition," Report No. 21, Digital Computer Laboratory, University of Illinois (July 1962).

[2]Kirsch, R. A., "Computer interpretation of English text and picture patterns," IEEE Transactions on Electronic Computers EC-13, 4 (August), 363-376 (1964).

[3]Kuhn, T. S., The Structure of Scientific Revolutions (The University of Chicago Press, Chicago, 1962).

[4]Marill, T., and Green, D. M., "Statistical recognition functions and the design of pattern recognizers," IRE Transactions on Electronic Computers EC-9, 4 (December), 472-477 (1960).

[5]Sebestyen, G. S., Decision-Making Processes in Pattern Recognition (The Macmillan Company, New York, 1962).

[6]Nilsson, N. J., Learning Machines (McGraw-Hill, New York, 1965).

[7]Character Recognition (British Computer Society, London, 1967).

[8]Narasimhan, R., "Syntactic descriptions of pictures and gestalt phenomena of visual perception," Report No. 142, Digital Computer Laboratory, University of Illinois (July, 1963).

[9]Narasimhan, R., "Labeling schemata and syntactic description of pictures," Information and Control 7, 151-179 (1964).

[10]Narasimhan, R., "Syntax-directed interpretation of classes of pictures," Comm. ACM 9,.3 (March), 166-173 (1966).

[11]Gray, J. C., "Compound data structure for computer aided design; a survey," Proc. of 22nd National Conference of ACM (Thompson Book Co., Washington, 1967), 355-365.

[12]Shaw, A. C., "The formal description and parsing of pictures," Ph. D. Thesis, Computer Science Department, Stanford University, Stanford, California. Also published as CS 94, Computer Science Department, Stanford University and SLAC Report No. 84, Stanford Linear Accelerator Center, Stanford, California (1968).

[13]Chomsky, N., Syntactic Structures (Mouton and Co., London, 1957).

[14]Feldman, J., and Gries, D., "Translator writing systems," Comm. ACM 11, 2 (February), 77-113 (1968).

[15]Shaw, A. C., "Lectures notes on a course in systems programming," Report No. CS 52, Computer Science Department, Stanford University (December 1966).

[16]Feder, J., "The linguistic approach to pattern analysis – a literature survey," Technical Report 400-133, Department of Electrical Engineering, New York University (February 1966).

[17]Grimsdale, R. L., Sumner, F. H., Tunis, C. J., and Kilburn, T., "A system for the automatic recognition of patterns," Paper No. 2792 M, The Institution of Electrical Engineering (December), 210-221 (1958).

[18]Sherman, H., "A quasi-topological method for machine recognition of line patterns," Proceedings of the International Conference on Information Processing, (UNESCO, Paris, 1959), 232-238.

[19]Eden, M., "On the formalization of handwriting," Proceedings of Symposia in Applied Mathematics, American Mathematical Society 12, 83-88 (1961).

[20]Eden, M., "Handwriting and pattern recognition," IRE Transactions on Information Theory IT-8, 2, 160-166 (1962).

[21]Minsky, M., "Steps toward artificial intelligence," Proceedings of the IRE 49, 1 (January), 8-30 (1961).

[22]Narasimhan, R., "A programming system for scanning digitized bubble-chamber negatives," Report No. 139, Digital Computer Laboratory, University of Illinois (June 1963).

[23]Narasimhan, R., and Reddy, V. S. N., "A generative model for hand-printed English letters and its computer implementation," Technical Report No. 12, Computer Group, Tata Institute of Fundamental Research, Bombay, India (June 1966).

[24]Lipkin, L. E., Watt, W. C., and Kirsch, R. A., "The analysis, synthesis, and description of biological images," Annals of the New York Academy of Sciences 128, 3 (January), 984-1012 (1966).

[25]Ledley, R. S., Programming and Utilizing Digital Computers (McGraw-Hill, New York, 1962), Chapter 8.

[26]Ledley, R. S., Rotolo, L. S., Golab, T. J., Jacobsen, J. D., Ginsberg, M. D., and Wilson, J. B., "FIDAC: film input to digital automatic computer and associated syntax-directed pattern recognition programming system," Optical and Electro-Optical Information Processing, Tippet, J., Berkowitz, D., Clapp, L., Koester, C., and Vanderburgh, Jr., A. (Eds). (MIT Press, Cambridge, Massachusetts, 1965), Chapter 33.

[27]Ledley, R. S., and Ruddle, F. H., "Chromosome analysis by computer," Scientific American, 40-46 (April 1966).

[28]Butler, J. W., Butler, M. K., and Stroud, A., "Automatic classification of chromosomes," Proceedings of the Conference on Data Acquisition and Processing in Biology and Medicine (Pergamon Press, New York, 1963).

[29]Guzmán, A., "Scene analysis using the concept of a model," AFCRL-67-0133, Computer Corporation of America, Cambridge, Massachusetts (1967).

[30]Guzmán, A., "Some aspects of pattern recognition by computer," MAC-TR-37, Project MAC, Massachusetts Institute of Technology (February 1967). (M.S. thesis).

[31]Evans, T. G., "A program for the solution of a class of geometric-analogy intelligence-test questions," Ph.D. Thesis, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts (1963). (Available as Physical and Mathematical Sciences Research Paper No. 64, Air Force Cambridge Research Laboratories, L. G. Hanscom Field, Massachusetts.)

[32]Evans, T. G., "A heuristic program to solve geometric-analogy problems," Proceedings of the AFIPS Spring Joint Computer Conference (Spartan Books, Inc., Washington, D. C., 1964), 327-338.

[33]Evans, T. G., "A description-controlled pattern analyzer," Proceedings of the IFIP Congress (Edinburgh, 1968).

[34]McCarthy, J., Abrahams, P. W., Edwards, D. J., Hart, T. P., and Levin, M. I., LISP 1.5 programmers manual (The MIT Press, Cambridge, Massachusetts, 1962).

[35]Shaw, A. C., "A proposed language for the formal description of pictures," GSG Memo 28, Computation Group, Stanford Linear Accelerator Center, Stanford, California (February 1967) (internal report).

[36]Miller, W. F., and Shaw, A. C., "A picture calculus," Emerging Concepts in Graphics, University of Illinois (November 1967) (in press).

[37]George, J. E., "Picture generation based on the picture calculus," GSG Memo 50, Computation Group, Stanford Linear Accelerator Center, Stanford, California (December 1967) (internal report).

[38]George, J. E., and Miller, W. F., "String descriptions of data for display," SLAC-PUB-383, Stanford Linear Accelerator Center, Stanford, California. Presented at 9th Annual Symposium of the Society for Information Display (1968).

[39]Anderson, R. H., "Syntax-directed recognition of hand-printed two-dimensional mathematics," Proceedings of the ACM Symposium on Interactive Systems for Experimental Applied Mathematics (to be published) (1967).

[40]Anderson, R. H., "Syntax-directed recognition of hand-printed two-dimensional mathematics," Ph. D. Thesis, Applied Mathematics, Harvard University, Cambridge, Massachusetts (1968).

[41]Clark, R. and Miller, W. F., "Computer-based data analysis systems at Argonne," Methods in Computational Physics, Adler, B., Fernbach, S., and Rotenberg, M. (Eds.) (Volume 5, Academic Press, New York, 1966), 47-98.

[42]Clowes, M. B., "Preception, picture processing and computers," Machine Intelligence 1, Collins, N., and Michie, D. (Eds.), (Oliver and Boyd, London, 1967), 181-197.

[43]Clowes, M. B., "A generative picture grammar," Seminar paper No. 6, Computing Research Section, Commonwealth Scientific and Industrial Research Organization, Canberra, Australia (April 1967).

[44]Feder, J., "Linguistic specification and analysis of classes of patterns," Technical Report 400-147, Department of Electrical Engineering, New York University (October 1966).

[45]Watt, W. C., "Morphology of the Nevada cattlebrands and their blazons – Part One," National Bureau of Standards Report No. 9050, U. S. Department of Commerce (1966).

[46]Watt, W. C., "Morphology of the Nevada cattlebrands and their blazons – Part Two," Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania (1967).

[47]Inselberg, A. and Kline, R., "A syntactic and contextual pattern recognizer; a preliminary study," Technical Memo 45, Computer Systems Laboratory, Washington University, St. Louis, Missouri (October 1967).

[48]Inselberg, A., "An approach to the syntax-directed analysis of graphic data," Technical Memo 52, Computer Systems Laboratory, Washington University, St. Louis, Missouri, (January 1968).

[49]Breeding, K., "Grammar for a pattern description language," Report No. 177, Department of Computer Science, University of Illinois (May 1965) (M.S. Thesis).

[50]Knoke, P. J. and Wiley, R. G., "A linguistic approach to mechanical pattern recognition," Digest of the First Annual IEEE Computer Conference, Chicago, Illinois (September) 142-144 (1967).

[51]Nir, M., "Recognition of general line patterns with application to bubble chamber photographs and handprinted characters," Ph.D. Thesis, Electrical Engineering, University of Pennsylvania, Philadelphia, Pennsylvania (1967).

[52]Coles, S., "Syntax directed interpretation of natural language," Ph.D. Thesis, Carnegie Institute of Technology, Pittsburgh, Pennsylvania (1967).

[53]Wirth, N. and Weber, H., "Euler – a generalization of Algol and its formal definition: Part I," Comm. ACM 9, 1 (January), 13-25 (1966).

List of Figures

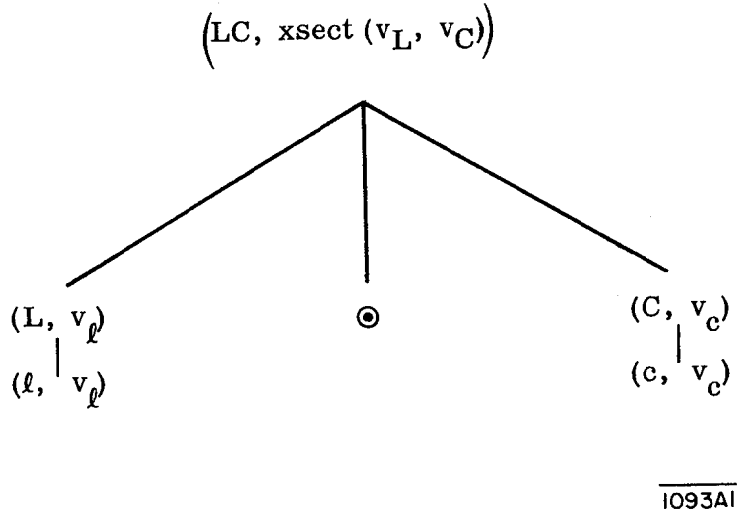
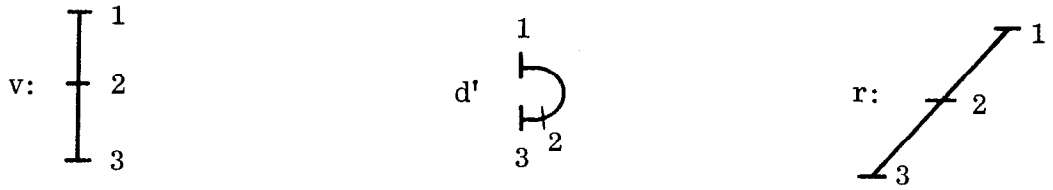



Fig. 1

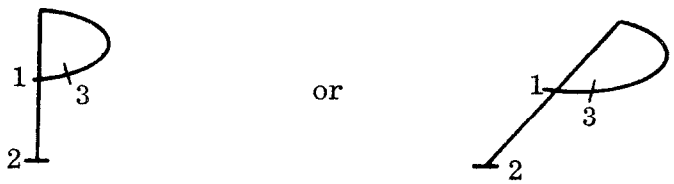
$$PE(1,\ 2,\ 3) \rightarrow v \cdot d'(11,\ 23;\ 2,\ 3;\ 2) \mid$$
$$r \cdot d'(11,\ 23;\ 2,\ 3;\ 2)$$

$$P \rightarrow PE$$

Rewriting Rules



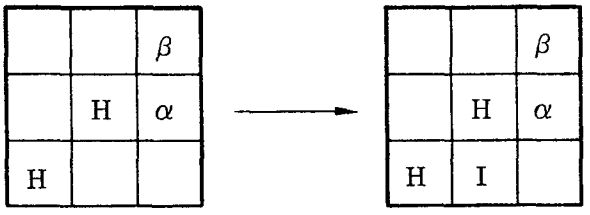

Primitives




P and PE

1093A2

Fig. 2




Sample Production: $\alpha \in \{L, I\}$, $\beta \in \{H, W\}$

| | | | | |
|---|---|---|---|---|
| | | | | W |
| | | | H | L |
| | | H | I | L |
| | H | I | I | L |
| V | B | B | B | R |

A Derived Triangle

1093A3

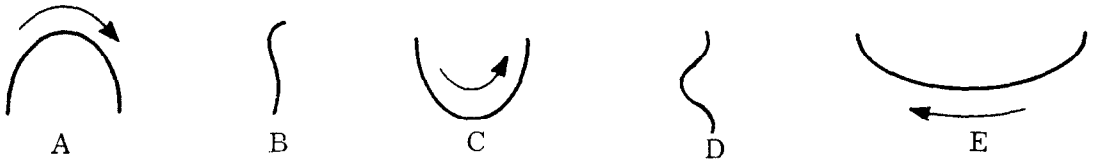
Fig. 3

⟨arm⟩ : := B⟨arm⟩|⟨arm⟩ B|A

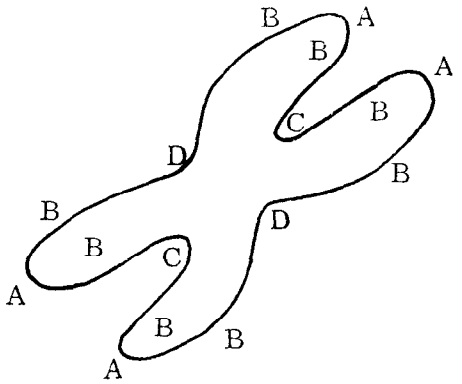
⟨side⟩ : := B⟨side⟩|⟨side⟩ B|B|D

⟨submedian chromosome⟩ : := ⟨arm pair⟩⟨arm pair⟩

Sample Productions

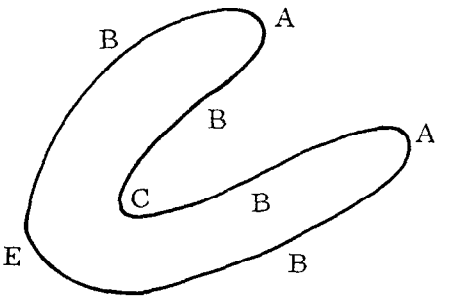



Basic Curve Types




BCBABDBABCBABDBA

Submedian



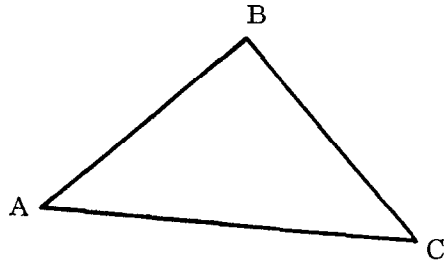

BCBABEBA

Telocentric

Chromosome Examples

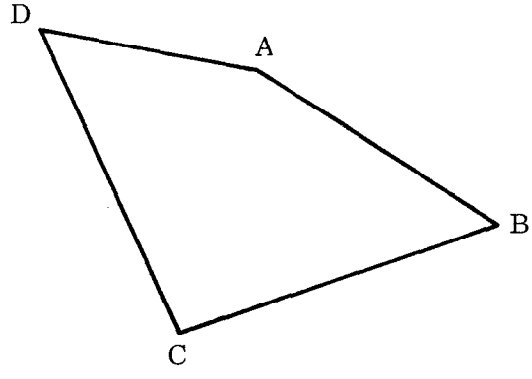
1093A4

Fig. 4




(=DEF= ISOS1 (( A (B C) C (B A) B (A C)) where ((LENG A B X)
(LENG B C X) (VARIABLES X))))

(=DEF= ISOS2 (( A (B C) C (B A) B (A C)) where ((ANGLE B A C A1)
(ANGLE B C A A1) (VARIABLES A1))))



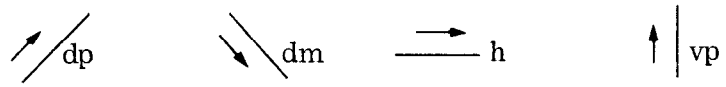

(=DEF= QUADR ( A (B D) B (C A) C (D B) D (A C)))

(=DEF= RECT (QUADR where ((LENG A B X) (LENG D C X)
(LENG A D Y) (LENG C B Y) (ANGLE D A B $90^{O}$)
(VARIABLES X Y))))

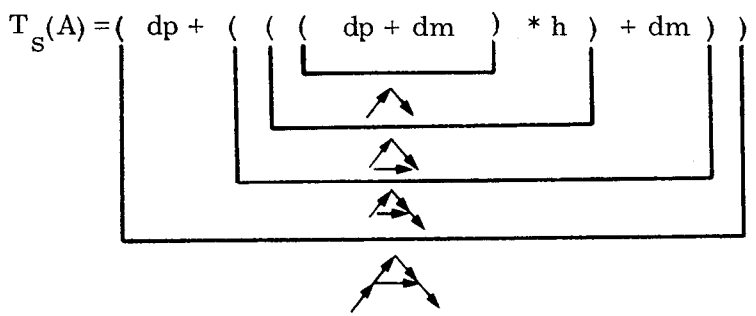
(=DEF= POLY (=OR= (RECT ISOS1)))

1093A5

Fig. 5



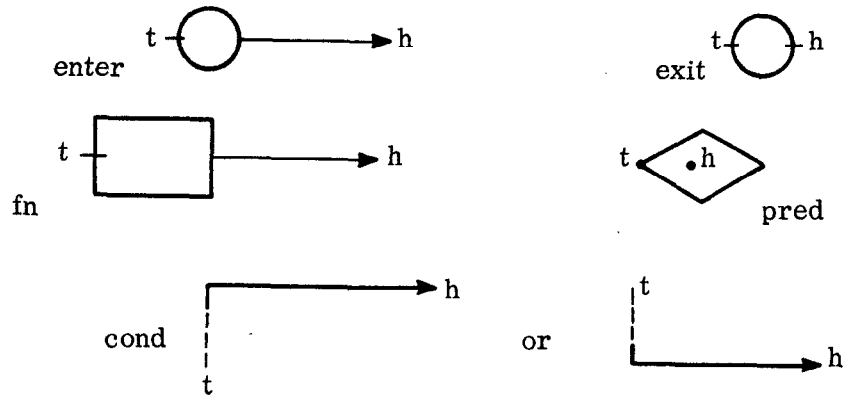

Primitive Classes




$T_s(F) = (\ vp + (h \times (vp + h))\ )$

1093A6

Fig. 6




Primitives

STMNT ⟶ BASIC | CNDTNL

BASIC ⟶ ASSIGN | FOR | $BLOCK^{b}$

FOR ⟶ STEPUNTIL | WHILE

STEPUNTIL ⟶ (INIT + ( ( ( ($TEST^{su}$ + cond) + $STMNT^{su}$)

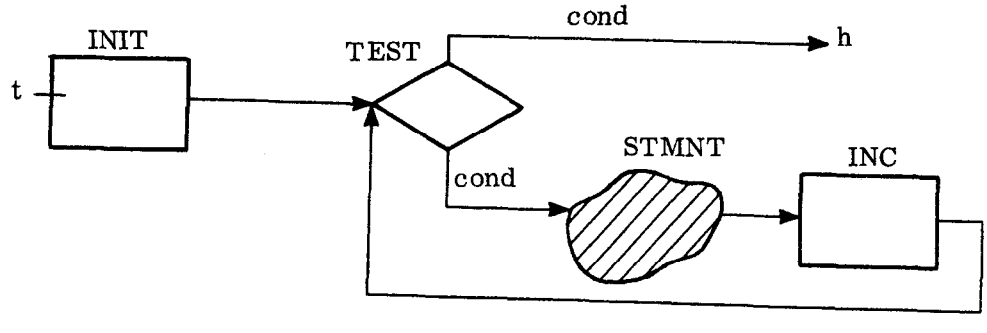
*(~ INC) ) × ( ($/TEST^{su}$) + cond) ) )

INIT ⟶ fn

INC ⟶ fn

TEST ⟶ pred

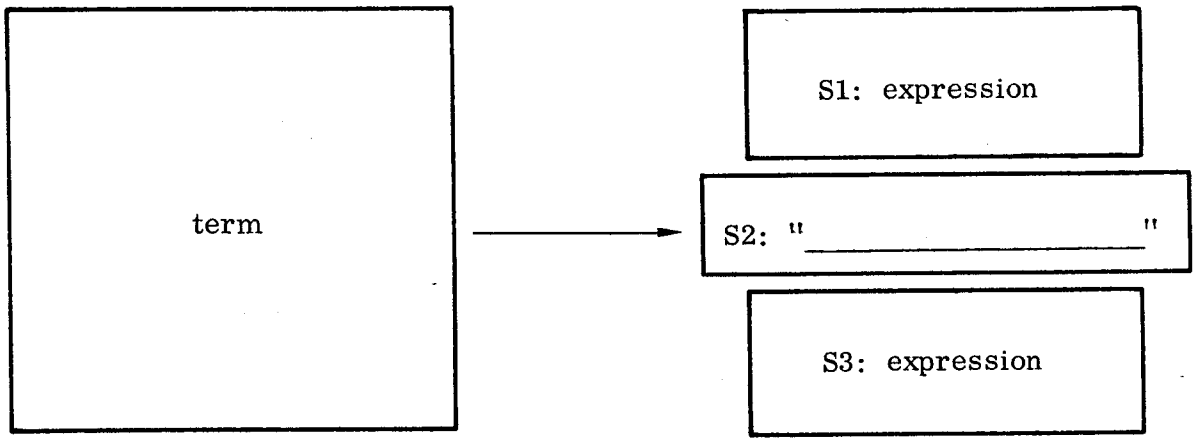
Partial Flow Chart Syntax




Stepuntil Element

1093A7

Fig. 7




Graphical Form of Replacement Rule

term ⟶

| | | |
|---|---|---|
| S1: expression | P1: $c_{01} > c_{21}$ and $c_{03} < c_{23}$ and $c_{04} > c_{26}$ | C1: $c_{21}$ |
| | | C2: $c_{22}$ |
| S2: horizline | P2: $\phi$ | C3: $c_{23}$ |
| S3: expression | P3: $c_{01} > c_{21}$ and $c_{03} < c_{23}$ and $c_{06} < c_{24}$ | C4: $c_{34}$ |
| R: $\phi$ | | C5: $c_{25}$ |
| M: $(s_1)/(s_3)$ | | C6: $c_{16}$ |

Tabular Form of Replacement Rule

1093A8

Fig. 8