International Journal Information theory & applications, Volume 9, Number 2, (51-60)

Formal Languages for Image Primitives Description

Peter L. Stanchev, David Green Jr.
Kettering University, Flint, MI 48504, {pstanche, dgreen}@kettering.edu

How do people recognize things in the images? We do not know the answer of this question. But in this paper we try to present some of the methods used by the artificial intelligence to recognize primitives into an image. We will analyze methods where images could be presented, as primitive's strings. Such images are CAD/CAM engineering drawings, handwrite and typed characters, chromosome images, bubble chamber images, computer graphics images, etc. Syntactic description is most easily applied to images that have the following properties: (1) they can be decomposed into a set of primitives and (2) the primitives are recognizable by an automatic procedure.

Keywords: Formal Languages, Pattern Recognition

1. Formal Definitions

An ***alphabet*** *V*, is any finite set of symbols. A ***string*** over an alphabet *V* is any string of finite length composed of symbols from the alphabet. A string with length 0 is called the ***empty string***, which is usually denoted by λ. V* - the ***free monoid*** over the set V is the set of all strings composed of symbols from V, including the empty string. The symbol V^+ denotes the set of strings $V^* - \lambda$. A ***language*** is any set of strings over an alphabet. A ***string grammar*** is a four-tuple $G = (V_N, V_T, P, S)$, where V_N is a finite set of nonterminals (variables), V_T is a finite set of terminals (constants), $(V_T \cap V_N = \varnothing)$, *P* is a finite set of productions and *S* is the start (root) symbol $(S \in V_N)$. A ***language generated by G***, denoted by *L(G)*, is a set of strings that satisfy two conditions:

a) each string is composed only of terminals;
b) each string can be derived from *S* by suitable applications of productions from the set *P*.

A grammar, in which the productions have the general form $\alpha \to \beta$, is called a ***phase structure grammar***. According to the type of restrictions placed on the production rules the grammars may be categorized into four types:

a) a grammar in which the form of production rules $\alpha \to \beta$, where $\alpha \in (V_N \cup V_T)^+$, $\beta \in (V_N \cup V_T)^*$ is allowed, is called ***unrestricted grammar***;

b) a grammar in which the form of production rules is $\alpha_1\ A\ \alpha_2 \rightarrow \alpha_1\ \beta\ \alpha_2$, where α_1, $\alpha_2 \in (V_N \cup V_T)^*$, $\beta \in (V_N \cup V_T)^*$, $A \in V_N$, is called ***context-sensitive grammar***;
c) a grammar in which the form of production rules is $A \rightarrow \beta$, where $A \in V_N$, $\beta \in (V_N \cup V_T)^+$, is called ***context-free grammar***;
d) a grammar in which the form of production rules is $A \rightarrow a\ B$ or $A \rightarrow \alpha$, where $A, B \in V_N$, a $\in V_T$, is called ***regular grammar***.

It is easy to see that all regular grammars are context-free, all context-free grammars are context-sensitive, and all context-sensitive grammars are unrestricted. The most often used grammars for image element description, are context-free and regular grammars. We will describe some of the most popular grammars concerned with some particular applications.

2. A Grammar for Chromosome Description

This grammar is proposed by Ledley at al. ([Ledl65]) and it is used to characterize submedian and telocentric chromosomes. The grammar is context-free grammar and uses the following set of primitives (Figure 1):

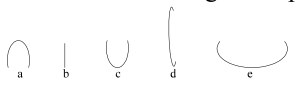


Figure 1. Set of chromosome primitives

The grammar G is given by $G = (V_N, V_T, P, S)$ where V_N = {<submedian chromosome>, <arm pair>, <left part>, <right part>, <arm>, <side>}, V_T= {a, b, c, d, e}, S={<submedian chromosome>}, and P:

```
<submedian chromosome> →     <arm pair> <arm pair>
<arm pair>             →     <side> <arm pair>
<arm pair>             →     <arm pair> <side>
<arm pair>             →     <arm> <right part>
<arm pair>             →     <left part> <arm>
<left part>            →     <arm> c
<right part>           →     c <arm>
<side>                 →     b <side>
<side>                 →     <side> b
<side>                 →     b
<side>                 →     d
<arm>                  →     b <arm>
<arm>                  →     <arm> b
<arm>                  →     a
```

An example of submedian chromosome is given in Figure 2. Their string representation is "babcbabdbabcbabd".

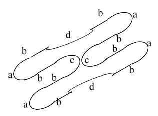


Figure 2. Submedian chromosomes

3. The Picture Description Language (PDL)

Another interesting example for image element description is the picture description language (PDL), proposed by Shaw ([Shaw69]). In PDL each terminal symbol is labeled at two distinguished points, a tail and a head. Four binary and a unary operations between primitives are defined. The corresponding operation descriptions and interpretations are given in the Figure 3.

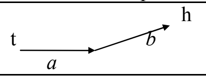
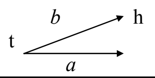
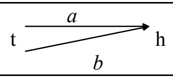
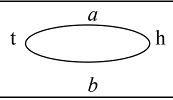
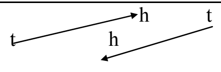
Primitive Structure	Interpretation	Graph
$a + b$	head(a) CAT tail(b)	
$a \times b$	tail(a) CAT tail(b)	
a - b	head(a) CAT head(b)	
a * b	(tail(a) CAT tail(b)) AND (head(a) CAT head (b))	
$\bar{a}$	-	

Figure 3. PDL description

Let consider the following example concerning a house drawing description. The set of the primitives is given in the Figure 4.

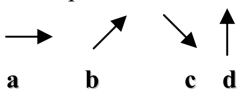


Figure 4. Set of a house drawing primitives

The grammar $G = (V_N, V_T, P, S)$ is given by: V_N = {house, roof, wall}, V_T = { a, b, c, d, +, x , - , *, ~ , “,” },

P: <roof>	$\rightarrow$	*((b + c)*a)*
<wall>	$\rightarrow$	*(((~d)+a)+d)*
<house>	$\rightarrow$	*(roof * wall)*

The PDL structural description $(((b+c)+a)^{*}(((\sim d)+a)+d))$ corresponds to the house drawing in Figure 5.

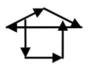


Figure 5. A house drawing

PDL is often used to express characters, for the purpose of their recognition.

4. Grammars for Two or Three Dimensional Pattern Description

In our considerations up till now the only relation between primitives was the concatenation. A natural generalization is to use some other useful relations. This is a more effective in describing two or three-dimensional pattern.

For example in the grammar proposed by [Ledl62] for house description, the following relations are included as terminal symbols:
$\rightarrow(X, Y)$ means that X is to the right of Y;
$O(X, Y)$ means that X is inside of Y;
$\Theta(X, Y)$ means that X is inside on the bottom of Y;
$\uparrow(X, Y)$ means that X rests on tops of Y;
$\rightarrow(X, Y)$ means that X rests to the right of Y.

The grammar is defined as $G = (V_N, V_T, P, S)$. An example of V_T is given in Figure 6, where V_N= {<door>, <windows>, <chimney>, <wall>, <gable>, <roof>, <front view>, <side view>, <house> } , S={<house>}

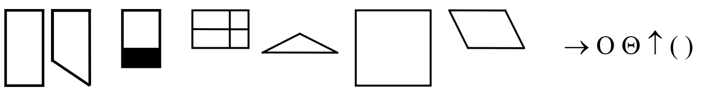


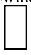
Figure 6. Definition of V_T in example for house description

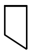
The production set P can be defined as:

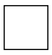
<door> →

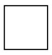
<windows> →

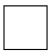
<windows> → (<windows>,)

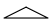
<chimney> → 

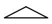
<chimney> → 

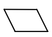
<wall> → 

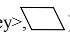
<wall> → Θ (<door>, )

<wall> → O(<window>, )

<gable> → 

<gable> → $\uparrow$ (<chimney>, )

<roof> → 

<roof> → $\uparrow$ (<chimney>, )

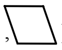
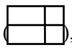
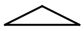
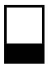
<front view> → $\uparrow$ (<gable>, <wall>)

<side view> → $\uparrow$ (<roof>, <wall>)

<house> → <front view>

<house> → (<house>, <side view>)

The following string corresponds to the house drawing shown in Figure 7:

→($\uparrow$($\uparrow$(, ),O (,)) $\uparrow$( , Θ ( ,))).

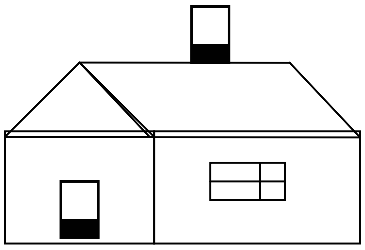


Figure 7. Drawing of a house

5. Tree Grammar

A ***regular tree grammar*** is a five-tuple $G = (V_N, V_T, r, P, S)$, where V_N is a finite set of nonterminals, V_T is a finite set of terminals $(V_T \cup$

$V_N = \varnothing$), S is the start symbol ($S \in V_N$), P is a finite set of productions of the form $\Omega \rightarrow \Psi$, where Ω and Ψ are trees, r is a ranking function, which denotes the number of direct descendants of a node whose label is a terminal in the grammar ([Moay76]). A tree grammar is an ***expansive tree grammar*** iff each production in P is of the form

A → a

A_1 A_2 ....A_N or $A \rightarrow a$, where $a \in V_T$, $A, A_1, A_2, ..., A_N \in V_N$. A house drawing and the corresponding tree representation is given in the Figure 8.

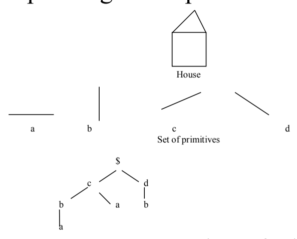


Figure 8. Tree representation of a house drawing

6. Fuzzy Grammar

Let X be a classical set of objects, called the ***universe***, whose generic elements are denoted as x. ***Membership*** in a classical subset A of X is often viewed as a ***characteristic function*** from X to {0, 1}, such as $\mu_A(x) = 0$ iff $x \notin A$; and $\mu_A(x) = 1$ iff $x \in A$. {0, 1} is called a ***valuation set*** . If the valuation set is measured on the real interval [0, 1], A is called a ***fuzzy set*** [Zade65]. $\mu_A(x)$ is the ***grade of membership*** of x in A.

Informally, a fuzzy grammar may be viewed as a set of rules for generating the elements of a fuzzy set [Lee69]. More precisely, a ***fuzzy grammar*** is a quadruple $G=(V_N, V_T, P, S)$, where V_T is a set of terminals, V_N is a set of non-terminals ($V_T \cap V_N = \varnothing$), P is a finite set of productions, $S \in V_N$ is the initial symbol. The elements of P define fuzzy sets in $(V_N \cup V_T)^*$ (referred as f***uzzy productions***) and for the case of context-free fuzzy grammars are expressions of the form: $A \overset{\rho}{\rightarrow} \beta$, $A \in V_N$, $\beta \in (V_N \cup V_T)^*$, where $0 \leq \rho \leq 1$ is the grade of membership of β given A. In case of general grammars, let $\alpha_1, \alpha_2, ..., \alpha_m$ be strings in $(V_N \cup V_T)^*$ and

$\alpha_1 \overset{\rho 2}{\rightarrow} \alpha_2$, $\alpha_2 \overset{\rho 3}{\rightarrow} \alpha_3$, ..., $\alpha_{m-1} \overset{\rho m}{\rightarrow} \alpha_m$, where $0 \leq \rho_i \leq 1$ be productions. The expression

$\alpha_1 \overset{\rho 2}{\rightarrow} \alpha_2$, $\alpha_2 \overset{\rho 3}{\rightarrow} \alpha_3$, ..., $\alpha_{m-1} \overset{\rho m}{\rightarrow} \alpha_m$, will be referred to as a ***derivation chain*** from α_1 to α_m. A string x of V_T^* is said to be in the ***fuzzy***

language L(G) iff x is derivable from S. The grade of membership $\mu_G(x)$ of x in $L(G)$ is given by:

$$\mu_G(x) = \sup \min (\mu (S \xrightarrow{\rho_1} \alpha_1), \mu (\alpha_1 \xrightarrow{\rho_2} \alpha_2), \ldots, \mu (\alpha_m \xrightarrow{\rho_{m+1}} x)),$$

where $\mu (\alpha_I \xrightarrow{\rho_{i+1}} \alpha_{i+1})$ is the non-null ρ_{i+1} such as $(\alpha_I \xrightarrow{\rho_{i+1}} \alpha_{i+1}) \in P$, for $i = 0, 1, \ldots , m$, if $\alpha_0 = S$ and $\alpha_{m+1} = x$. The supremum is taken over all derivation chains from S to x. In any derivation chain, the minimum ρ_i is taken.

A fuzzy grammar for a house drawing used in [Rabi87] is presented below. Let the set of primitives are described in the Figure 9.

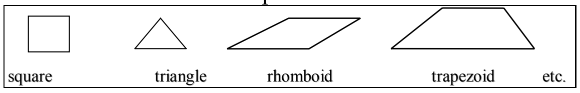


Figure 9. Set of primitives

The fuzzy grammar G is given by $G = (V_N, V_T, P, S)$, where V_N = {<house>, <facade>, <front>, <sideview>, <wall>, <window>, <door>, <chimney>, <roof>}; V_T = {in, inW, inN, inS, inE, inNW, inSW, inNE, inSE, outN, outS, outE, outNE, outSE, intersect, “,” , square, triangle, rhomboid, trapezoid}.

The position symbols are presented in Figure 10, where I, W, N, S, denote the four cardinal points and "in" and "out" denote that an object is inside or outside another object. Other position terminal symbols are "intersect" for intersection, "(...)" for enclosure, etc.

	outN		OutNE
inNW	inN	inWE	
inW	in	inE	outE
inSW	inS	inSE	
	outS		outSE

Figure 10. Relation symbols

S = <house> and P:

<house> $^{(1.0)}\rightarrow$ <sideview> outE <house> ;
<house> $^{(1.0)}\rightarrow$ <facade> ;
<door> $^{(1.0)}\rightarrow$ <square> ;
<window> $^{(0.8)}\rightarrow$ <square> ;
<window> $^{(1.0)}\rightarrow$ square outE square ;
<chimney> $^{(0.8)}\rightarrow$ trapezoid ;
<wall> $^{(1.0)}\rightarrow$ square ;
<wall> $^{(1.0)}\rightarrow$ (<window> in square) ;
<wall> $^{(0.8)}\rightarrow$ (<window> <door> in square) ;
<front> $^{(1.0)}\rightarrow$ triangle ;
<front> $^{(1.0)}\rightarrow$ <chimney> outS triangle ;
<roof> $^{(1.0)}\rightarrow$ rhomboid ;
<roof> $^{(1.0)}\rightarrow$ <chimney> outS rhomboid ;
<facade> $^{(0.95)}\rightarrow$ <front> outS <wall> ;

```
<sideview> (1.0)→ <roof> outS <wall> ;
<chimney> (0.9)→ square ;
<wall>(0.95)→ (<door> inS square).
```

For the house drawing in Figure 11 the following string corresponds: "(square outS rhomboid outS (square in square)) outE (triangle outS (square inS square))". After processing by the given grammar the object "house" is recognized with grade of membership 0.8.

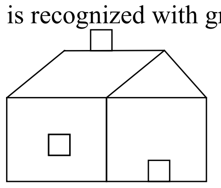


Figure 11. A house drawing

7. Attribute Relational Graph

The ***Attributed Relational Graph*** (ARG) is proposed in [Eshe84]. It is a relational structure, which consists of a set of nodes and a set of branches representing the relations between the nodes. Both nodes and branches may have some attributes assigned to them.

Formally an ARG is defined as $G = (N, B, A, E, G_N, G_B)$, where N $(N = \{n_1, n_2, ..., n_{|N|}\})$ - a finite set of nodes; $|N|$ is the number of nodes in N; B $(B = \{ b_1, b_2,...,b_{|N|}\})$ - a set of ordered node pairs (or directed branches), i.e., $b=(n_i, n_j)$ for some $1 \leq i, j \leq |N|$ denotes the branch emanating from node n_i to node n_j and $|B|$ is the number of branches in B; A - an alphabet of node attributes; E - an alphabet of branch attributes; G_N - a function (or a set of functions) for generating the node attributes; G_B - a function (or a set of functions) for generating the branch attributes.

Nodes are used to represent the basic elements in the image, while their properties are assigned as attributes to the respective nodes. Attributed branches between the corresponding nodes represent the relations between two basic elements. As an example we choose the area of House-furnishing design [Rabi89]. We limited the example to the basic elements shown in Figure 12, with the associated attributes, and the relations shown in Figure 13.

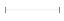
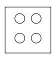
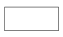
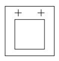
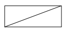
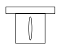
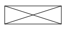
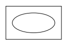
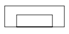
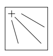
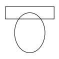
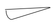
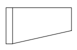
line (coordinate$_1$, coordinate$_2$)		gas_stove (center_coordinate)	
rectangle (coordinate$_1$, coordinate$_2$, coordinate$_3$, coordinate$_4$)		sink (center_coordinate)	
bed (center_coordinate)		W.C. (center_coordinate)	
wardrobe (center_coordinate, number_of_doors, type)		wash_tub (center_coordinate)	
sofa (center_coordinate)		shower_bath (center_coordinate)	
chair (center_coordinate)		bidet (center_coordinate)	
door (coordinate$_1$, coordinate$_2$)		piano (center_coordinate, type)	

Figure12. Basic elements

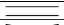
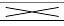
Parallel (no parameters)	
Intersection (no parameters)	

Figure 13. Relations

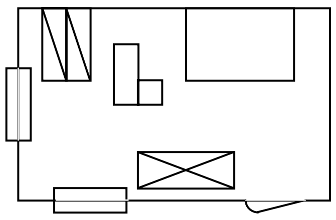


Figure 14. House furnishing design plan

The corresponding ARG representation of the house furnishing plan from Figure 14 is: N (set of nodes)

n_1 → *line* ([0.0, 0.0], [0.7, 0.0])
n_2 → *line* ([0.0, 0.0], [0.0, 1.2])
n_3 → *line* ([2.3, 0.0], [4.8, 0.0])
n_4 → *line* ([6.1, 0.0], [6.6, 0.0])
n_5 → *line* ([0.0, 2.7], [0.0, 4.0])
n_6 → *line* ([6.6, 0.0], [6.6, 4.0])
n_7 → *line* ([0.0, 4.0], [6.6, 4.0])
n_8 → *rectangle*([0.7,-0.25],[2.3,-0.25], [0.7,0.25],[2.3, 0.25])

n_9 → *rectangle*([-0.25,1.2],[0.25, 1.2],[-0.25,2.7],[0.25, 2.7])
n_{10} → *rectangle*([3.5, 2.5], [5.8, 2.5], [3.5, 4.0], [5.8, 4.0])
n_{11} → *door* (1, [4.8, 0.0], [6.1, 0.0])
n_{12} → *bed* (1, [0.75, 3.25])
n_{13} → *bed* (1, [1.25, 3.25])
n_{14} → *wardrobe* (1, [3.7, 0.7], 2, wood)
n_{15} → *chair* (1, [2.5, 2.5])

Some of the G_B elements are:

b_1 → *parallel* (n_{12}, n_{13}),
b_2 → *parallel* (n_{12}, n_3),
b_3 → *parallel* (n_{13}, n_3).

8. Conclusions

This paper presents some methods on understanding and interpretation of primitives in automated image analysis and processing. A new strategy using extended linear combination and fast two pass parallel pattern matching techniques is proposed in [Wang98]. This new method is able to distinguish objects with very similar patterns which is a limitation in the described methods.

References

[Eshe84] Eshera M., Fu K., "A Graph Distance Measure for Image Analysis" , IEEE Trans. Syst., Man, Cyber., Vol. SMC.-14, No.3, pp.398-408 (May/June 1984)

[Ledl62] Ledley R., "Programming and Utilising Digital Computers", DigitalComputers, McGraw-Hill, N.Y. (1962)

[Ledl65] Ledley R., Rotolo L., Golab T., Jacobsen J., Ginsberg M., Wilson J., "FIDAC: Film Input to Digital Automatic Computer and Associated Syntax-Directed Pattern-Recognition Programming System" , in "Optical and Electro-Optical Information Processing" , Tippet J. et al., (edts.), MIT Press, pp.591-614 (1965)

[Lee69] Lee E., Zadeh L., "Note on fuzzy languages", Information Sciences, Vol.1, pp.421-434 (1969).

[Moay76] Moayer B., Fu K., "A Tree System Approach for Fingerprint Pattern Recognition" , IEEE Trans. Comput., C-25, pp.262-274 (Mar. 1976)

[Rabi87] Rabitti F., Stanchev P., "An Approach to Image Retrieval from Large Image Databases", Proc. ACM-SIGIR Conf., New Orleans, pp.284-295 (1987)

[Rabi89] Rabitti F., Stanchev P., "GRIM_DBMS - a GRaphical IMage DataBase System", in "Visual Database Systems", T. Kunii (edt.) North-Holland 1989 (415-430).

[Shaw69] Shaw A., "A Formal Picture Description Scheme as a Basis for Picture Processing Systems" , Inf. Control, No. 14, pp.9-52 (1969)

[Wang98] Wang P., Analysis, Learning, and Recognition of Articulated Line Drawing Images, Int. J. CSIM, Special Issue on Image Recognition, Vol 1-2, (1998).

[Zade65] Zadeh L., "Fuzzy Sets", Inf. Control, No. 8, pp.338-353 (1965)