International Journal of Computer and Information Sciences, Vol. 6, No. 4, 1977

An Application of Formal Linguistics to Scene Recognition

Flavio Roberto Dias Velasco[1] and Celso de Renna e Souza[1]

Received April 1976; revised November 1976

A model for scene description and syntactical recognition is proposed based on branch-and-node-labeled graphs. Generating grammars for scene descriptions are proposed, and the problem of parsing general descriptions is treated. An example of application is given for scenes from a popular Brazilian comic strip.

KEY WORDS: Scene recognition; syntactical recognition; graph grammars; artificial intelligence.

"A sentence should contain no unnecessary words
a paragraph no unnecessary sentences
for the same reason that a drawing
should have no unnecessary lines
and a machine no unnecessary parts."

William Strunk, Jr., 1918

1. INTRODUCTION

Though pattern "recognition" systems exist that are entirely based on feature extraction and pattern classification by statistical means,[1–3] they are clearly inadequate if the ultimate goal is to approach human recognition capabilities by automated methods.[4] It is more or less accepted now that a system that actually recognizes say, scenes, must have as its integral part a highly structured model of the universe, of which the scene being scanned is a sample.

[1] Computer and Information Sciences, Space Research Institute (INPE), São José dos Campos, S.P., Brasil.

So-called "structural methods" have appeared in which, in addition to the information contained in partitions of multidimensional feature spaces, relational information is also taken into account in the process of recognizing a scene or pattern. Generally speaking, the use of structural information also simplifies the costly and time-consuming feature-extraction step of the process.

2. SYNTACTICAL METHODS

The syntactical models see the patterns as formed by primitive elements and the relations among them. Generating grammars are used to provide finite models for countably infinite universes and to guide the parsing; the extension to two dimensions of the natural notion of concatenation for string grammars, though, is not obvious. Several models have been proposed, including those of Kirsch,[5] Dacey,[6] the array grammars of Milgram and Rosenfeld,[7] and others, such as that of Feder.[16]

Grammars that generate graphs have been studied by Pfaltz and Rosenfeld.[8] Another reasonable approach has been to propose grammars that generate not the scenes or patterns themselves but suitable descriptions of them; see, for instance, Menninga,[9] Shaw,[10] Pfaltz,[11] Fu and Bhargawa,[12] and Masumi.[13]

The present work is based on the notion of web grammars, as in ref. 8, generating not the patterns themselves but their descriptions, as in ref. 11.

3. GLOGS

Definition 1. An *oriented graph* is a pair $g = \langle N, A \rangle$ where N is a finite, nonempty set of nodes and A is a binary relation on N, $A \subset N \times N$.

Definition 2. A *vocabulary* is a triple $\mathscr{V} = \langle V, R, \sigma \rangle$ where V is a finite, nonempty set of node labels, R is a finite, nonempty set of branch labels, and σ is a bijection $\sigma: R \to R$, such that $\sigma = \sigma^{-1}$.

The branch-and-node-labeled graphs that will be used to describe scenes may now be defined.

Definition 3. A *general labeled oriented graph* (GLOG) over a vocabulary $\mathscr{V} = \langle V, R, \sigma \rangle$ is a quadruple $\omega = \langle N, A, F, E \rangle$ where $\langle N, A \rangle$ is an oriented graph (the *ground graph*), $F: N \to V$ is the node labeling function, and $E: A \to \mathscr{P}(R) - \{\phi\}$ is the branch labeling function.[2]

[2] We shall use $\mathscr{P}(A)$ for the power set of A.

Not all GLOGs are valid descriptions of scenes, however. It was found expedient to restrict somewhat the previous definition.

Definition 4. A GLOG as in Definition 3 is *descriptive* iff its ground graph is strongly connected, symmetric, antireflexive, and, moreover, for all $a, a' \in A$ such that a' is the symmetric branch of a, $r \in E(a)$ iff $\sigma(r) \in E(a')$.

Descriptive GLOGs, as defined above, are those that naturally arise in the description of real world scenes. The function σ in this case associates to each spatial relation (e.g., "is above") its semantic inverse (e.g., "is below"). The other requirements for a GLOG to be descriptive are also of the same practical nature.

Example 1. Let $V = \{r, w, h, i\}$ where $r \leftrightarrow$ roof, $w \leftrightarrow$ wall, $h \leftrightarrow$ house, $i \leftrightarrow$ window; let $R = \{a, b, i, s, t\}$ where $a \leftrightarrow$ above, $b \leftrightarrow$ below, $i \leftrightarrow$ inside, $s \leftrightarrow$ surrounds, $t \leftrightarrow$ touching; the GLOG on the right describes the scene on the left of Fig. 1. In this case, $F(1) = r$, $E((1, 2)) = \{t, a\}$, etc.

It is clear that the same GLOG describes infinitely many scenes that may, from this point of view, be considered "equivalent." Also, the natural concept of isomorphism applies to GLOGs, isomorphic GLOGs describing the same or equivalent scenes. Also naturally, subGLOGs may be defined.

Definition 5. Given two GLOGs $\alpha = \langle N_\alpha, A_\alpha, F_\alpha, E_\alpha \rangle$ and $\beta = \langle N_\beta, A_\beta, F_\beta, E_\beta \rangle$, over the same vocabulary $\mathscr{V}$, then β is a "subGLOG" of α iff: 1) $N_\beta \subset N_\alpha$; 2) $A_\beta \subset A_\alpha$; 3) $F_\beta = F_\alpha / N_\beta$; and 4) $E_\beta = E_\alpha / A_\beta$. β will be a complete subGLOG of α iff conditions 1 through 4 hold and $A_\beta = A_\alpha \cap N_\beta^2$.

4. GLOG GRAMMARS

The definition of generating grammars for GLOGs follows closely that for phase-structure grammars,[13] with the additional difficulty brought

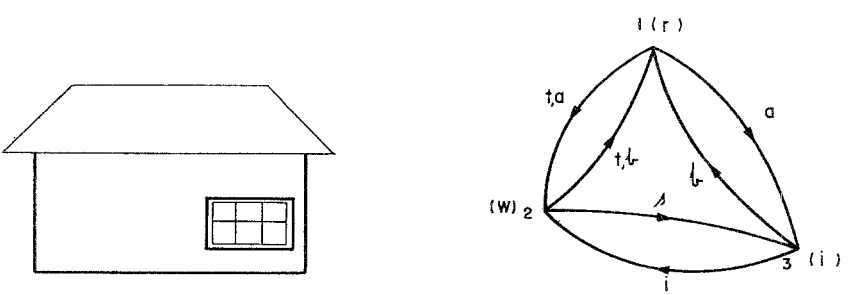


Fig. 1. A simple scene and its descriptive GLOG.

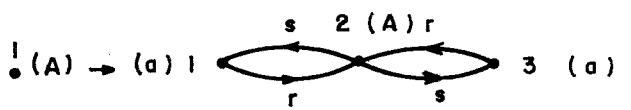
about by the handling of graphs. Thus, whenever the left side of a production is substituted by its right side, the new connections with the remaining nodes must be unambiguously specified. Also, there must be some assurance that only descriptive GLOGs, as in Definition 4, are generated.

Definition 6. A *GLOG Grammar*, GLOGG, is a triple $G = \langle \mathscr{V}, I, P \rangle$ where $\mathscr{V}$ is a vocabulary as in Definition 2, where $V = V_T \cup V_N$, $V_T \cap V_N = \phi$, I is a set of single node (trivial) initial GLOGs whose labels are in V_N and P is a set of triples (productions)(α, β, ψ) where α and β are GLOGs over $\mathscr{V}$, and ψ is the *embedding function*, ψ: $N_\alpha \times R \times \{-1, +1\} \to N_\beta \times R$. Moreover, in α there should be at least one node labeled with a nonterminal symbol.

As in phrase-structure grammars, V_N is the set of "nonterminals" and V_T the set of "terminals." The triples of P are rewriting (here one could say "redrawing") rules and can also be written $\alpha \to \beta(\psi)$ or $\alpha \to^{\psi} \beta$.

The embedding function ψ introduced above describes how the new GLOG being inserted, β, should be connected to the host GLOG nodes, say $N_{\rho-\alpha}$. If $\psi(t, r, x) = (n, s)$, then every node m of $N_{\rho-\alpha}$ linked to t by a branch (t, m)—in this case $x = +1$—or(m, t), for $x = -1$, labeled r should be linked in the resulting graph with n by a branch (n, m) if $x = +1$ or (m, n) if $x = -1$, labeled s.

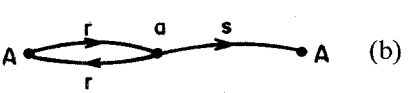
Example 2. Let $V_N = \{A\}$ and $V_T = \{a, b\}$, $R = \{r, s\}$, $\sigma = \{(r, s), (s, r)\}$, and ω the initial GLOG $\omega = \{\{1\}, \phi, \{(1, A)\}, \phi\}$. We can represent ω as $\underset{\cdot}{1}(A)$; let P be given by

1. 

with $\psi_1(1, r, +1) = (2, r)$, $\psi_1(1, s, +1) = (2, s)$, $\psi_1(1, r, -1) = (2, r)$, $\psi_1(1, s, -1) = (2, s)$ and

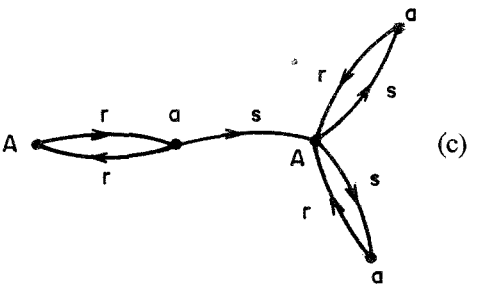
2. $\underset{\cdot}{1}(A) \to \underset{\cdot}{1}(b)$

with $\psi_2(1, r, +1) = (1, r)$, $\psi_2(1, s, +1) = (1, s)$, $\psi_2(1, r, -1) = (1, r)$, $\psi_2(1, a, -1) = (1, s)$.
For instance, from the GLOG



(b)

we could generate



It is easy to see that there is a condition of "coherence" for embedding functions in that $\psi(n_1, r_1, x) = (n_2, r_2)$ iff $\psi(n_1, \sigma(r_1), -x) = (n_2, \sigma(r_2))$. Also, the notion of "derives directly" carries over from the phrase-structure grammars and will be indicated by $\Rightarrow$.

Definition 7. A given GLOGG, $G = \langle \mathscr{V}, I, P \rangle$ is *coherent* iff every production (α, β, ψ) of P is coherent.

Theorem 1. Given a coherent GLOGG G, if $\rho \Rightarrow \mu$ by G, then, if ρ is a descriptive GLOG, then μ will be descriptive.

Proof. The proof is straightforward. It can be shown that A is anti-reflexive and symmetric, that (N_μ, A_μ) is strongly connected, and that $r \in E_\mu(n_1, n_2)$ iff $\sigma(r) \in E_\mu(n_2, n_1)$. □

Definition 8. The *language* $L(G)$ generated by a given GLOGG G is the set $L(G) = \{\omega / \omega$ is a terminal GLOG and $I \overset{*}{\Rightarrow}_G \omega\}$.

Theorem 2. If a given GLOGG G is coherent and $\omega \in L(G)$, then ω is descriptive.

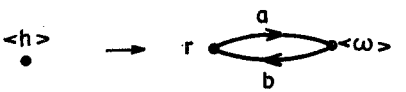
Proof. This follows immediately from Theorem 1 and the fact that any GLOG of I is descriptive. □

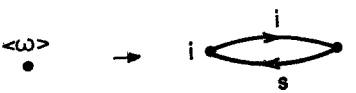
The well-known Chomsky hierarchy for phrase-structure grammars carries over nicely for GLOGGs, with somewhat unexpected results. Though it is simple to define context-sensitive GLOGGs, we will concentrate on context-free and regular GLOGGs.

Definition 9. A GLOGG $G = \langle \mathscr{V}, I, P \rangle$ is *context-free* iff in every production $(\alpha, \beta, \psi) \in P$, α is a trivial one-node GLOG with a nonterminal label. The language $L(G)$ will also be said to be context-free.

For a context-free GLOGG the embedding function ψ can be simplified to ψ: $R \times \{-1, +1\} \to N_\beta \times R$.

Example 3. Using the vocabulary $\mathscr{V}$ of Example 1, let $V_N = \{\langle\omega\rangle, \langle h\rangle\}$, $V_T = \{r, \omega, i\}$, $I = \langle h\rangle$, and let the productions be given by

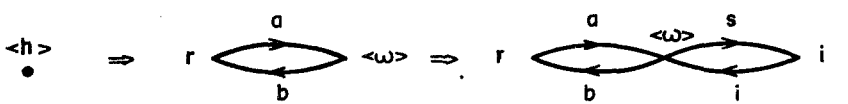
1. 

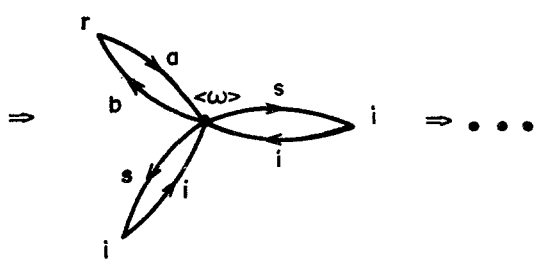
2. 

$\psi(a, +1) = (\langle\omega\rangle, a)$,
$\psi(i, -1) = (\langle\omega\rangle, i)$

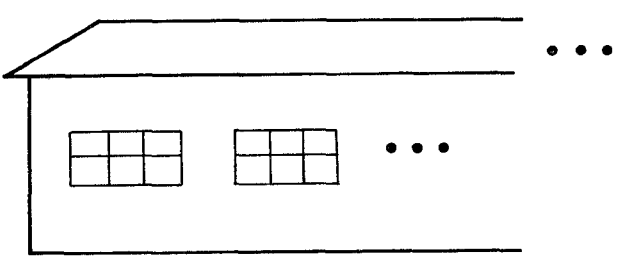
3. $\langle\omega\rangle \to \omega$ $\quad \psi(a, +1) = (\omega, a), \quad \psi(i, -1) = (\omega, i)$

For instance, the derivations





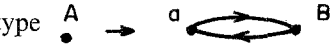
would lead to descriptions of scenes such as



the situation being identical to that of context-free grammars with self-embedding rules.

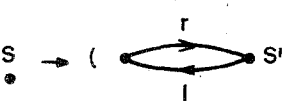
Theorem 3. It is decidable if a given GLOG ω over $\mathscr{V}$ belongs to a given context-free language $L(G)$; it is also decidable if a given such G generates the empty language. It is also decidable if a given context-free GLOGG generates a finite language.

Proof. The proof parallels the corresponding proofs for phase-structure grammars. □

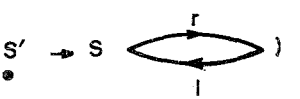
Definition 10. A context-free GLOGG G is *linear* iff in every production $(\alpha, \beta, \psi) \in P$, at most one node of β is labeled with an element of V_N. It is *regular* iff every production $(\alpha, \beta, \psi) \in P$ is such that $\#N_\beta \leqslant 2$ and the replacements are of the type  or $A \to a$ where $A, B \in V_N$, $a \in V_T$, $x, y \in R$. The corresponding languages will also be called linear or regular.

Theorem 4. Every regular GLOG language is linear; every linear GLOG language is context-free.

Example 4. Let $V_N = \{S, S'\}$, $V_T = \{(), ()\}$, $R = \{\text{right } (r), \text{ left } (l)\}$ $I = S$, and let the productions of P be

1.  $\psi_1(r, +1) = (S', r)\ \psi_1(r, -1) = ((, r)$ $\psi_1(l, +1) = (), l)\ \psi_1(l, -1) = (S', l)$

2.  $\psi_2(r, +1) = (), r)\ \psi_2(r, -1) = ((, r)$ $\psi_2(l, +1) = ((, l)\ \psi_2(l, -1) = (), l)$

3.  $\psi_3(r, +1) = (), r)\ \psi_3(r, -1) = (S, f)$ $\psi_3(l, +1) = (S, l)\ \psi_3(l, -1) = (), l)$

The corresponding language is that of nested parentheses, the same generated by the phrase-structure grammar $S \to (S)/(\ )$; note that this language is known not be "regular," yet it can be generated by a regular GLOGG.

5. PARSING ALGORITHM

A considerable portion of the total computer time in automatic scene analysis is devoted to preprocessing (noise cleaining, deblurring, filtering);

segmentation (region growing, partitioning, region grouping); property measurements (local properties, texture); and shape analysis.[14,15] These steps are outside the scope of this work and will be considered as having been previously performed, while we focus on the structural analysis. It must be said, however, that an interplay between the structural analyzer and the previous programs, especially the segmentation and feature extraction steps, may simplify them considerably.

The system considered in this paper has the overall organization shown in Fig. 2; the preprocessing subsystem delivers to the lexical analyzer the scene already partitioned. The lexical analyzer preclassifies the regions and extracts a complete GLOG, with the branches properly labeled by the relations in R. Though all parts of the system are operational, we shall concentrate on the syntactical analyzer.

As for languages generated by phrase-structure grammars, our syntactical analyzer for context-free GLOG languages can proceed top-down or bottom-up; there is no direct analog to the left-to-right parsin, however, since the nodes of a GLOG are not naturally totally ordered. Algorithm 2, is a bottom-up parser for linear context-free GLOG languages.

Definition 11. Given a context-free GLOGG $G = \langle \mathscr{V}, I, P \rangle$, and two GLOGs ρ and μ over $\mathscr{V}$, ρ *directly reduces to* μ, $\rho \vdash \mu$, iff there exists a ρ' such that $\mu \Rightarrow_G \rho'$, ρ' is a subGLOG of ρ, and $N_{\rho'} = N_\rho$.

Definition 11, at first glance, should read: ρ *directly reduces to* μ, $\rho \vdash \mu$, iff $\mu \Rightarrow_G \rho$. However, it was found that, in practical applications, it was expedient to allow the "erasure" of redundant or unnecessary spatial relations during the parsing process.

Clearly then, if $\alpha \Rightarrow_G \beta$, then $\beta \vdash \alpha$, but the converse is not true. Our procedure will be, given a GLOG ω, to try to find a chain of reductions $\omega = \omega_0 \vdash \omega_1 \vdash \cdots \vdash \omega_n$, until a ω_n is found that contains a GLOG ω belonging to I (set of initial GLOGs).

One needs, then, an algorithm that will tell whether some β from a production $(\alpha, \beta, \psi) \in P$ is a subGLOG of a given ρ and another that will substitute such β by α, producing a μ such that $\rho \vdash \mu$.

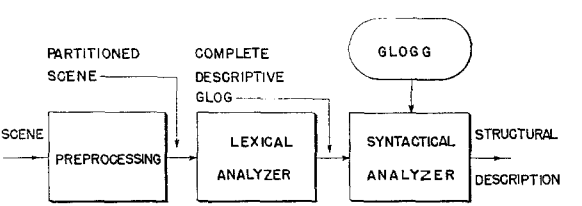


Fig. 2. The syntactical scene recognition scheme.

Definition 12. Given a GLOG ω and a node $n \in N_\omega$, the *neighborhood of* n, nbh(n), is the set of nodes connected by branches of ω to n; that is,

$$\text{nbh}(n) = \{m/(n, m) \quad \text{or} \quad (m, n) \in A_\omega\}$$

Algorithm 1. Given two GLOGs β and ρ over $\mathscr{V}$, there is an algorithm for telling whether β is a subGLOG of ρ, generating an injection Ω: $N_\beta \to N_\rho$ such that $(n_1, n_2) \in A_\beta$ iff $(\Omega(n_1), \Omega(n_2)) \in A_\rho$, $F_\beta(n_1) = F_\rho(\Omega(n_1))$, and $E_\beta(n_1, n_2) \subset E_\rho(\Omega(n_1), \Omega(n_2))$.

The algorithm is given in Appendix A. Once it is found that β is a subGLOG of ρ, μ will be such that

1. $N_\mu = (N_\rho - N_\beta) \cup N_\alpha$.
2. $A_\mu = (A_\rho \cap (N_\rho - N_\beta)^2) \cup A_{\rho,\alpha}$.

where $A_{\rho,\alpha}$ is the set of branches that reconstitute the connections between $\rho - \beta$ and α (its formal description will not be given for the sake of clarity).

3. $F_\mu/(N_\rho - N_\beta) = F_\rho/(N_\rho - N_\beta)$
   $F_\mu/N_\alpha = F_\alpha$.
4. $E_\mu/(A_\rho \cap (N_\rho - N_\beta)^2) = E_\rho/(A_\rho \cap (N_\rho - N_\beta)^2)$.

and $E_\mu/A_{\rho,\alpha}$ is a suitable labeling of the branches linking $\rho - \beta$ to α (again its formal description will not be given).

Unfortunately, it may happen that the above conditions are satisfied, that $\rho \vdash \mu$, but that μ is not strongly connected (hence not descriptive).

Definition 12. Given a GLOGG $G = \langle \mathscr{V}, I, P \rangle$, a given production (α, β, ψ) will be said to be *final*, iff $F_\beta(n) \in V_t$ for all $n \in N_\beta$. Such labels for $n \in N_\beta$ will also be said to be *final.*

If $\omega \in L(G)$, then, final productions surely were used, hence there must be final labels in ω. This fact and Algorithm 1 are exploited in the bottom-up parsing algorithm.

Algorithm 2. Given a GLOG ω over $\mathscr{V}$ and a linear context-free GLOGG $G = \langle \mathscr{V}, I, P \rangle$ there is an algorithm that parses ω and decides whether $\omega \in L(G)$ (see Definition 8).

The complete algorithm is given in Appendix B. There is no implication that the structural description obtained from a successful application of Algorithm 2 is unique; instead of generating all possible descriptions, we postulated that the more complex productions be tried first, so that a richer description could be obtained.

6. AN APPLICATION

To test the applicability of the model proposed, a particular universe of scenes was chosen; since for "real-life" scenes the preprocessing required would be considerable and since the emphasis of the model was not on preprocessing techniques, it was decided that a simplified universe should be chosen. The final decision was to utilize a familiar Brazilian comic strip, "Monica," as a source of simplified scenes. In these scenes, the regions are clearly defined and uniformly colored, and the renditions of objects (cars, houses, trees, fences, etc.) are highly simplified. Yet, it represents a very rich, highly structured universe, very much related to that of our daily lives and perfectly recognizable to a human being.

A GLOGG G was written by (human) inference, from a sample of the comic strips, with 40-odd productions, easily expandable to allow for new characters, new objects, etc. In the implementation, the scene was represented by a 128×128 matrix, and INPE's image processing system (GE's IMAGE 100) was used for the preprocessing stages.

Figure 3 shows a typical scene after preprocessing, with levels of gray representing different colors. Note the effects of both quantizing and random noise.

The overall recognition algorithm was implemented as show in Fig. 4.

The region identifier utilizes a region-growing algorithm using the four-neighbor concept.[14] Some attributes of the regions are also obtained. It was found that, for the universe under consideration, the color of the region and its position in the frame were sufficient for the subsequent analysis, thus simplifying the feature-extraction step. The lexical analyzer I provides the syntactical analyzer with tentative multiple classifications, while the computing of the various relations among the regions is done at the lexical analyzer II.

The complete system was programmed in the B-6700's Extended ALGOL. The source deck has approximately 1100 cards. The complete processing of the scene shown in Fig. 3 took nearly 21 sec: 15 sec for region identification and 1 sec for the lexical analyzer I. These times were found to be typical. Figure 5 shows the output corresponding to Fig. 3.

Fig. 3. Typical 128 × 128 matrix scene after preprocessing.

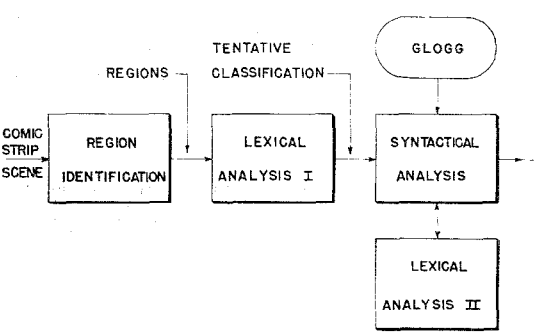


Fig. 4. Implementation of the recognition algorithm.

```
*** FOUND SKY      COMPOSED BY:
   1,

*** FOUND CLOUD    COMPOSED BY:
   2,

*** FOUND SKY      COMPOSED BY:
   3,

*** FOUND TREE     COMPOSED BY:
   8,   4,

STRUCTURAL DESCRIPTION:
LEAVES  REGION= 4   ABOVE, TOUCHING  ,  TRUNK   REGION=  8

*** FOUND SKY      COMPOSED BY:
   7,

*** FOUND EYE2     COMPOSED BY:
  10,  11,   9,

STRUCTURAL DESCRIPTION:
EYE      REGION=  9   SURROUNDS            IRIS    REGION= 11
EYE      REGION=  9   SURROUNDS            IRIS    REGION= 10
IRIS     REGION= 11   BY THE SIDE OF       IRIS    REGION= 10

*** FOUND EYE      COMPOSED BY:
   6,  10,  11,   9,

STRUCTURAL DESCRIPTION:
FACE     REGION=  6   SURROUNDS            EYE     REGION=  0

*** FOUND HEAD     COMPOSED BY:
  14,  13,   6,  10,  11,   9,

STRUCTURAL DESCRIPTION:
HEAD     REGION=  0   SURROUNDS            MOUTH   REGION= 13
HEAD     REGION=  0   SURROUNDS            TONGUE  REGION= 14
MOUTH    REGION= 13   TOUCHING             TONGUE  REGION= 14

*** FOUND HEAD     COMPOSED BY:
  14,  13,   6,  10,  11,   9,

STRUCTURAL DESCRIPTION:
HEAD     REGION=  0   ABOVE, TOUCHING      SHIRT   REGION= 15
SHIRT    REGION= 15   ABOVE, TOUCHING      SHORTS  REGION= 19

*** FOUND CEB2     COMPOSED BY:
  16,  24,  15,  19,  14,  13,   6,  10,  11,   9,

STRUCTURAL DESCRIPTION:
CEB1     REGION=  0   TOUCHING             ARM     REGION= 24
CEB1     REGION=  0   TOUCHING             ARM     REGION= 16
ARM      REGION= 24   BY THE SIDE OF       ARM     REGION= 16

*** FOUND CEB2     COMPOSED BY:
  16,  15,  19,  14,  13,   6,  10,  11,   9,

STRUCTURAL DESCRIPTION:
CEB1     REGION=  0   TOUCHING             ARM     REGION= 16

*** FOUND CEBOLI   COMPOSED BY:
  24,  23,  22,  26,  27,  25,  16,  15,  19,  14,  13,   6,
  10,  11,   9,

STRUCTURAL DESCRIPTION:
CEB2     REGION=  0   ABOVE, TOUCHING      LEG     REGION= 25
CEB2     REGION=  0   ABOVE, TOUCHING      LEG     REGION= 24
LEG      REGION= 25   TOUCHING             SOCKS   REGION= 27
LEG      REGION= 25   BY THE SIDE OF       LEG     REGION= 24
SOCKS    REGION= 27   ABOVE, TOUCHING      SHOE    REGION= 26
SOCKS    REGION= 27   BY THE SIDE OF       SOCKS   REGION= 23
SHOE     REGION= 26   BY THE SIDE OF       SHOE    REGION= 22
LEG      REGION= 24   TOUCHING             SOCKS   REGION= 23
SOCKS    REGION= 23   ABOVE, TOUCHING      SHOE    REGION= 22

*** FOUND HOUSE    COMPOSED BY:
  18,  20,  21,  12,

STRUCTURAL DESCRIPTION:
ROOF     REGION= 12   ABOVE, TOUCHING      WALL    REGION= 18
WALL     REGION= 18   SURROUNDS            WINDOW  REGION= 21
WALL     REGION= 18   SURROUNDS            WINDOW  REGION= 20
WINDOW   REGION= 21   BY THE SIDE OF       WINDOW  REGION= 20

*** FOUND GRASS    COMPOSED BY:
  17,

**** NON-CLASSIFIED REGIONS:
   5,  28,  29,
```

Fig. 5. Output corresponding to the scene of Fig. 3.

7. CONCLUSIONS

Through the use of branch-and-node-labeled graphs (GLOGs) as structural descriptions of scenes, and their generating grammars, it was possible to describe a chosen restricted universe and implement a complete syntactical recognition scheme, the performance of which was clearly surprising. It is obvious that the universe can be enriched, that more attributes of the regions can be used, and that the parsing algorithm can be somewhat improved in terms of its running time. However, these improvements in the particular implementation chosen to illustrate the method would not add to the method itself or to the basic ideas involved.

One interesting topic for further study is the problem posed by partially hidden objects. The two-dimensional frame being a projection of a three-dimensional scene, its information content is certainly smaller, and a good portion of it must be supplied by the viewer when recognizing the original (three-dimensional) scene. One could increase the grammars in order to describe partial objects, but only with the danger of inserting ambiguity.

Fuzzy models[15] also show promise in scene recognition and should be fully exploited.

APPENDIX A. ALGORITHM 1

The following algorithm will tell, given GLOGs β and ρ over $\mathscr{V}$ if β is a subGLOG of ρ:

procedure subGLOG (β, ρ);
glog β, ρ; ($\beta = \langle N_\beta, A_\beta, F_\beta, E_\beta \rangle$, $\rho = \langle N_\rho, A_\rho, F_\rho, E_\rho \rangle$)
begin
  set D, K; ; *set* R;
  node n, m; ; *function* ζ;
  comment subGLOG will try to find a function
    ζ: $N_\beta \to N_\rho$, one-to-one, such that:
    1) $\forall x \in N_\beta$, $F_\beta(x) = F_\rho(\zeta(x))$,
    2) $\forall (x, y) \in A_\beta$, $(\zeta(x), \zeta(y)) \in A_\rho$ and
      $E_\beta(x, y) \subset E_\rho(\zeta(x), \zeta(y))$
      D and K are sets of nodes, ($D \subset N_\beta$ and $K \subset N_\rho$),
      such that:
      1) $x \in D \Leftrightarrow \zeta(x)$ is defined
      2) $y \in K \Leftrightarrow y = \zeta(x)$, $x \in D$
      3) if $y \in K$ and $y = \zeta(x_1)$ and $y = \zeta(x_2)$ then $x_1 = x_2$
      The algorithm will stop when D is equal to N_β ;
let $n \in N$; $D := \{n\}$;
$R := \{t \in N_\rho \mid F_\beta(n) = F_\rho(t)\}$;

comment R will count the nodes of N_β that have the same label as that is, likely candidates for $\zeta(n)$;
while $R \neq \phi$
do begin
  let $m \in R$;
  $K := \{m\}$;
  $\zeta := \{(n, m)\}$;
  if equivalent (n, m, β, ρ)
  then
    begin
      output ("β is subGLOG of ρ,
      with isomorphism ζ");
      goto finis;
    end;
  $R := R - \{m\}$;
  end;
output ("β is not a subGLOG of ρ");
finis; *end* subGLOG
boolean procedure equivalent (n, m, β, ρ);
node n, m; GLOG β, ρ;
begin
  set H;
  function σ_n ;
  comment given $n \in N_\beta$ and $m \in N_\rho$, such that
    $F_\beta(n) = F_\rho(m)$, then equivalent (n, m, β, ρ) is true iff
    $\exists \sigma_n$: nbh$(n) \rightarrow$ nbh(n), an injection such that:
    a) if $x \in$ nbh$(n) \cap D$ then $\sigma_n(x) = \zeta(x)$
    b) $\forall x \in$ nbh(n):
      1. $F_\beta = F_\rho(\sigma_n(x))$,
      2. $E_\beta((n, x)) \subset E_\rho((\sigma_n(x)))$ and
         $E_\beta((x, n)) \subset E_\rho((\sigma_n(x), m)))$
      3. equivalent $(x, \sigma_n(x), \beta - n, \rho - m)$ is true, where $\beta - n$ and $\rho - m$ are GLOGs such that
         $N_{\beta-n} = N_\beta - \{n\}$, $N_{\rho-m} = N_\rho - \{m\}$; $A_{\beta-n} = A_\beta \cap N_{\beta-n} \times N_{\beta-n}$,...
$\forall x \in$ nbh$(n) \cap D$ *do* $\sigma_n(x) := \zeta(x)$;
$H :=$ nbh$(x) - D$;
comment in H are the nodes of nbh(n) not yet assigned to D (set of nodes to which ζ is defined presently),
let J be the set of injections σ: $H \rightarrow$ nbh$(m) - K$
such that $F_\beta(x) = F_\rho(\sigma(x))$;
while $J \neq \phi$

```
do begin
  let σ ∈ J; J := J − {σ};
  σ_n := σ_n ∪ σ;
  if ∀x ∈ nbh(n), F_β((n, x)) ⊂ E_ρ((m, σ_n(x)) and E_β((x, n)) ⊂ E_ρ((σ_n(x), m))
  then begin
            D := D ∪ H;
            K := K ∪ σ_n*(H);
            ζ := ζ ∪ σ_n ;
            if ∀x ∈ H equivalent (x, ζ(x), β − n, ρ − m)
            then begin equivalent := true; goto finis end;
            else begin (Reconstruct ζ, D, K)
              ζ := ζ − H × nbh(m);
              D := D − H;
              K := K − σ*(H);
              end
    end
  end;
  equivalent := false;
finis: end.
```

COMMENTS ON THE ALGORITHM

As can be seen, the algorithm used does not exploit heuristic methods to accelerate the processing time. A complete search is done among all possible alternatives, and the algorithm stops as soon as one solution is found. The backtracking, if a wrong path is searched, is done by the usual means.

However, the "combinational explosion" that would make the algorithm impractical is avoided because of two facts. First, the GLOGs on the right-hand side of the productions are very limited in their number of nodes. In our case, the largest number was 7, while the average was around 3. Second, the fact that both the nodes and branches are labeled drastically reduces the number of possibilities to be searched.

Even though this algorithm does not constitute a novelty, it was included to illustrate some of the difficulties that appeared during the computer implementation. Also, a clear need was felt for better tools to handle graph structures, other than the usual high-level languages (ALGOL, FORTRAN, PL-1, etc). Another valid observation is that the representation of a graph by adjacency matrices, with the purpose of simplifying the use of existing isomorphism algorithms, is not very practical in our case due to the constant structural changes of the graphs involved.

APPENDIX B. ALGORITHM 2

procedure syntactical analysis (ω, G);
GLOG ω; GLOGG G;
begin
node n;
set F, T; *production* p;
$F := \{n \in N_\omega \mid F_\omega(n)$ is final$\}$;
comment
F is the set of all nodes such that $F_\omega(n)$ is the final terminal; the algorithm will search all these nodes, trying to start the parsing. If this is not possible, the algorithm will stop with the exit "fail";
while $F \neq \phi$
do begin
let $n \in F$;
$T := \{q \in \rho \mid q = (\alpha, \beta, \psi)$ is final and $F(n) \in F^*(N_\beta)\}$;
while $T \neq \phi$
do begin
let $p = (\alpha, \beta, \psi) \in T$;
if subGLOG (β, ω) *then*
begin
reduce (using ρ) $\omega \vdash\!\!- \mu$;
if test $(F_\omega(n), \omega)$ *then*
begin
output ("ω contains a GLOG from $L(G)$");
goto finis
end
end;
$T := T - \{p\}$
end;
end;
output ("ω does not contain a GLOG from $L(G)$")
finis: *end*
boolean procedure test (α, ω);
GLOG α, ω;
begin
set T; *production* p;
comment test (α, ω) is true iff either $\alpha \in I$ (initial GLOG) or $\exists_p \in P$, $p = (\gamma, \delta, \psi)$, where $A \in F^*(N_\delta)$ (A is the name of α)
such that:

```
        1) δ subGLOG of ω, ω ⊢ μ using p and
        2) test (γ, ω) is true;
        if α ∈ I then test := true
        else begin
          T := {q ∈ P, q = (γ, δ, ψ)/A_δ ∈ F_δ*(N_δ),
          where A is the name of α};
          while T ≠ φ
          do begin
              let p = (γ, δ; ψ) ∈ T;
              if subGLOG(δ, ω) then
              begin
                reduce (using p) ω ⊢ μ;
                if test (γ, μ) then
                begin
                  test := true;
                  goto finis
                end
              end;
              T := T − { p}
          end;
        test := false
      end;
finis: end test
```

REFERENCES

1. M. A. Aiserman, "Remarks on two problems connected with pattern recognition," in *Methodologies of Pattern Recognition*, M. S. Watanabe, Ed. (Academic Press, New York, 1969).
2. L. Kanal, "Patterns in pattern recognition: 1968–1974," *IEEE Trans. Inf. Theory* **20**(6) (November 1974).
3. L. Kanal and B. Chandrasekaran, "On Linguistic, Statistical and Mixed Models for Pattern Recognition," University of Maryland Computer Science Technical Report No. 152 (1971).
4. F. D. Preparata and S. R. Ray, "An approach to artificial non-symbolic cognition," *Inf. Sci.* **4**:65–86 (January 1972).
5. R. A. Kirsch, "Computer interpretation of english text and picture patterns," *IEEE Trans. Electron. Comput.* **13**(4):363–376 (1964).
6. M. F. Dacey, "The syntax of a triangle and some other figures," *Pattern Recognition* **2**:1–31 (January 1970).
7. D. L. Milgram and A. Rosenfeld, "Array Automata and Array Grammars," Proceedings of the IFIP Congress, 1971 (North-Holland, Amsterdam, 1972), Booklet TA-2, pp. 166–173.
8. J. L. Pfaltz and A. Rosenfeld, "Web Grammars," University of Maryland Computer Science Technical Report No. 69-84 (January 1969).

9. L. D. Menninga, "A Syntax-Directed Approach to Pattern Recognition and Description," *AFIPS Fall Joint Computer Conference Proceedings*, Las Vegas (November 1971).
10. A. C. Shaw, "Formal picture description scheme as a basis for picture processing systems," *Inf. Control* **14**:9–52 (1969).
11. J. L. Pfaltz, "Web grammars and picture description," *Comput. Graphics Image Process.* **1**:193–220 (1972).
12. K. S. Fu and P. H. Bhargawa, "Tree systems for syntactic pattern recognition," *IEEE Trans. Comput.* **22**:1087–1098 (December 1973).
13. A. Masumi, "Picture Analysis by Graph Transformation," Ph.D. Thesis, University of Illinois at Urbana–Champaign (1973).
14. A. Rosenfeld and J. S. Weszka, "Picture Recognition," University of Maryland Computer Science Technical Report No. 344 (1974).
15. W. J. M. Kickert and H. Koppelaar, "Application of fuzzy set theory to syntactic pattern recognition of handwritten capitals," *IEEE Trans. Syst. Man. Cybern.* **SMC-6** (2) (February 1976).
16. J. Feder, "Plex languages," *Inf. Sci.* **3** (July 1971).