Fuzzy-Shape Grammars for Cursive Script Recognition*

Marc PARIZEAU†, Réjean PLAMONDON‡ and Guy LORETTE§

printed July 8, 1997

Abstract

This paper presents a new approach for cursive script segmentation and recognition, based on intrinsic models of cursive letters (allographs). The models are built using stratified context-free shape grammars that permit the definition of both syntactic and semantic attributes. These attributes synthetize pertinent morphological characteristics of allographs that are then used for recognition. The main topic of this paper concerns the parsing process developed for allograph segmentation, which uses fuzzy-logic to evaluate the likelihood of segmentation hypotheses. This process is the first step of the recognition method and leeds to the construction of a graph where nodes represent segmented allographs and arcs link adjacent nodes. The analysis of this segmentation graph can be carried out for submitting possible letter sequences to higher linguistic evaluation modules. Preliminary results are given for multi-writer isolated cursive letters. For a test database containing cursive samples of 10 different writers, an average recognition rate of 91.7% is obtained. Recognition is non personalyzed, that is, cursive samples of all writers are treated with the same algorithm parameters.

*This work was published in *Advances in Structural and Syntactic Pattern Recognition*, H. Bunke (editor), World Scientific Publishing, pp. 320-332, 1993. It was supported in part by NSERC Canada under grant OGP000915, and in part by FCAR Québec under grant CRP2667. Marc Parizeau received scholarships from NSERC and FCAR.

†Marc Parizeau was with the Département de génie électrique et de génie informatique, École Polytechnique de Montréal. He is now with the Département de génie électrique et de génie informatique, Université Laval, St-Foy (PQ), Canada, G1K 7P4. E-mail: *parizeau@gel.ulaval.ca*

‡Réjean Plamondon is with the Département de génie électrique et de génie informatique, École Polytechnique de Montréal, C.P. 6079 – Succ. "A", Montréal, Canada, H3C 3A7.

§Guy Lorette is with the Institut de Recherche en Informatique et Systèmes Aléatoires, Université de Rennes I, 35042 Rennes Cedex, France

I Introduction

Cursive script recognition is a difficult problem for essentially three reasons. First, because of the inherent variability of the handwriting process, which is emphasized by individual and regional variations in handwriting styles. Second, cursive letters within cursive words cannot be segmented unambiguously before recognition and, thus, letter segmentation must be integrated in the recognition process. Third, linguistic knowledge (higher than lexical) is often essential for total recognition of cursive script but automatic comprehension of natural langages is an other very difficult (unsolved) problem.

To tackle handwriting recognition, three types of kwowledge can be distinguished : *morphological*, *pragmatic* and *linguistic*. Morphological knowledge concerns everything that is known about the shapes of letters that are to be recognized. This type of knowledge can take two forms : letter prototypes or intrinsic letter models. *Letter prototypes* are samples of handwriting used as models either directly by template matching methods, or indirectly via feature extraction and comparison. The main advantage of this form of morphological knowledge lies in its ability to develop systems that can be adapted to writers individuality. Its disavantage is that common characteristics of writers are difficult to generalize. *Intrinsic letter models* are theoritical morphological knowledge in the sense that they are not constructed from the particular habits of writers. These a priori models exist — although they can vary from region to region — because they are taught to school children. The main advantage of intrinsic models is to provide basic morphological knowledge of handwriting that can be used as a starting point for a truly multi-writer recognition system. Its disavantage, obviously, is that some writers will deviate from these theoretic models and that adaptation is much more complicated. From a review of the literature [1, 2, 3], it appears that, except for [4] where the idea was first introduced but with a simplified model based on a four directions coding, no one has ever tried to build intrinsic letter models for cursive script recognition.

Pragmatic knowledge[1] corresponds to all information not related to the shape of the symbols but nevertheless usefull for their recognition. Generaly, it concerns information like spatial arrangement of *allographs*[2] (for roman alphabets : from left to right on a common baseline) or temporal order of handwriting *components*[3] (for dynamic systems). This type of knowledge is used by all existing systems but usualy not explicitly.

Linguistic knowledge can be divide into three categories : lexical, syntactical and semantics. However, only lexical knowledge is used commonly and, in fact, most current systems rely on a limited vocabulary and cannot function correctly without it.

The object of this paper is to present a method for building intrinsic models of *allographs* using *stratified context-free shape grammars* [5, 6] and to describe the parser for these grammars, that enables allograph segmentation within cursive words. This method is part of a recognition approach [3] designed to recognize cursive script using only morphological and pragmatic knowledge. It is also part of a larger project which aims toward the proposal of an intelligent electronic penpad [7].

The organization of the paper is as follows. Section II first describes the grammatical formalism adapted from [5]. Then, Section III elaborates the allograph modelling methodology and illustrates the formalism with a model for lower-case letter *a*. The parsing mechanism for allograph segmentation is developed in Section IV. Section V includes experimental recognition results obtained from a multi-writer database of isolated lower-case cursive letters. Finally, Section VI contains conclusions.

II Fuzzy-Shape Grammars

Attributed languages have been proposed as a way to unify syntactic and statistical pattern recognition [8]. With attributed (shape) grammars, it is possible to define patterns with attributes that both describe syntactic and semantic information[4]. The syntactic attributes consist of various attachment points used for spatial arrangement of subpatterns into a more complex pattern, and the semantic attributes describe various geometric or other types of properties of the pattern. The advantage of these grammars over conventional string grammars is that relations between primitives are no longer restricted to concatenation. Any morphologically significant relation can be used to define patterns.

A programming language named $\mathcal{HAD}$ ($\mathcal{H}$ierarchical $\mathcal{A}$llograph $\mathcal{D}$escription language) has been developed for creating intrinsic allograph models [3]. The compiler for this langage is written in **C** and has been tested on both MS-DOS and UNIX operating systems.

II.1 Grammatical Formalism

Although the stratified context-free shape grammar formalism described in this section is derived from [5], it is rewritten here because, first, several minor changes were made to it and, second, to establish the mathematical notations relevant to the parsing mechanism developed in Section IV.

Let $\mathcal{F} = \{S_1, \ldots, S_i, \ldots, S_n\}$ be the set of n pertinent symbols associated to a given segmentation problem (i.e. allographs, sub-allograph patterns and eventually pattern primitives). Then, to all S_i symbols (except for a few primitive symbols) is associated a grammar $G_i(T_i, N_i, P_i, S_i)$ where T_i is the set of terminal symbols, N_i is the set of nonterminals, P_i is the set of production rules and $S_i \in N_i$ is the start symbol (i.e. the symbol representing the pattern that is modelled by the grammar). Let $V_i = N_i \cup T_i$ denote the set of all vocabulary symbols for grammar G_i ; and associate $\forall v \in V_i$ a level number $\nu(v) \in \{0, 1, \ldots, k_i\}$ with $\nu(S_i) = k_i$ and $\forall v \in T_i$, $\nu(v) = 0$.

The set T_i of terminal symbols contains already segmented symbols, that is, either pattern primitives that are obtained by an external segmentation process, or start symbols of other grammars. The possibility of using start symbols as terminal symbols of other grammars justify the hierarchical epithet of the $\mathcal{HAD}$ langage. However, to avoid circular definitions, if $S_j \in T_i$ then the constraint $j < i$ is imposed to

[1] Although linguists might consider this type of knowledge as linguistic, we make this distinction for *pragmatic* reasons!

[2] The term *allograph* is used to designate a particular style for writing a given letter.

[3] The term *component* is used to designate the portion of the written trace between a pendown and a penlift (while the pen is in contact with paper).

[4] N.B. The words *syntactic* and *semantic* used in this context have the same meaning than in the introduction, but do not refer to the same languages ! In the introduction, we were referring to syntax and semantics of natural languages like english or french...

make the grammars *stratified*.

The nonterminal symbols of N_i are intermediate patterns that permit the passage from the start symbol to terminal symbols. Unlike string grammars, all symbols $v \in V_i$ possess a non trivial structure consisting in a name, a syntactic part and a semantic part :

$$\text{v} = \textit{name}[\textit{syntactic part}][\textit{semantic part}]$$

where *name* is a token that identifies uniquely the symbol, *syntactic part* is a set of attachment points used to specify spatial arrangement of the symbol and *semantic part* is a set of properties used to characterize the symbol.

The set P_i contains all production rules for symbols of N_i. Every production rule has the form $(v := \lambda, C, G)$ which indicates that symbol $v \in N_i$ can be rewritten as the group of constituent symbols $\lambda = \{c_1, \ldots, c_j, \ldots, c_m\}$ if condition C is verified with $c_j \in V_i$ and $\nu(c_j) \leq \nu(v)$, $\forall j \in \{1, \ldots, m\}$. The case where $\nu(c_j) = \nu(v)$ is not, however, permitted for the first production rule of v. The applicability condition C can mix both syntactic and semantic parts (attributes) of any symbol $c_j \in \lambda$ to test for acceptable spatial arrangement and coherent properties. G describes the set of generation rules associated to the syntactic and semantic parts of v. Again, these rules are functions of both syntactic and semantic parts of any symbol $c_j \in \lambda$. No production rules are associated to terminal symbols.

An illustration of this formalism, expressed in the $\mathcal{HAD}$ language, will be presented in Section III.

II.2 Fuzzy-Grammars and Fuzzy-Logic

With the stratified context-free grammar formalism, a symbol is thus transformed in other (simpler) symbols under a certain syntactic and semantic applicability condition. Consider, for example, a proximity condition between two symbols. Obviously, it is possible to arrange these symbols in such a way that this proximity condition is absolutly satisfied. Reciprocally, the two symbols can be moved away from each other in such manner that this condition becomes absolutly inacceptable. But in between these two arrangements, there exists a number of other arrangements where it would be dubious to take an unambiguous decision.

Fuzzy-logic is thus used to managed this ambiguity [9, 10]. Let X represent a non-fuzzy universe and A a fuzzy set on X, that is, $A \subset X$. Then, the characteristic function $\alpha_A(x)$ defines the grade of membership of element $x \in X$ with respect to fuzzy set A :

$$A = \{(x, \alpha_A(x)) \,|\, x \in X\} \ , \quad 0 \leq \alpha_A(x) \leq 1 \qquad (1)$$

Fuzzy-shape grammars can be defined simply by evaluating with fuzzy-logic the applicability conditions of production rules that become characteristic functions of fuzzy-sets associated to each symbols. In practice, interesting elements of set A are those for which $\alpha_A(x)$ is non zero.

III Allograph Modelling

The type of shape grammars described in the previous section has the advantage of allowing any relevant relation between pattern primitives. However, it not clear how such grammars can be inferred automatically and, hence, all allograph models were hand-generated. This process is very time-consuming but need to be done only once.

Although, grammatical inference is not conceived at the moment, this does not mean that automatic learning can't be done. Indeed, it is possible to define for each allograph model a set of morphological characteristics (properties) imbedded into the grammars but on which statistical learning could be applied.

This section presents the general methodology followed for intrinsic allograph model building and gives an example for one allograph of letter *a*.

III.1 Attributed Handwriting Primitive

But first, the handwriting primitive on which the models are built is presented briefly. For more details concerning this primitive, the reader is referred to [11].

The $\mathcal{A}$ttributed $\mathcal{H}$andwriting $\mathcal{P}$rimitive ($\mathcal{AHP}$) used as a starting point for modelling allographs is based on a pragmatic model of handwriting which represents handwriting components as a sequence of *characteristic points* linked together by *constant curvature* segments (i.e. circular arcs) [3, 11]. A primitive is associated with every characteristic point and represents a portion of the component that stretches from the previous characteristic point to the following one.

Each primitive, is extracted — by an external segmentation process not described here — with three attachment points : a starting point (ps), a characteristic point (pc), and an ending point (pe) ; and seven

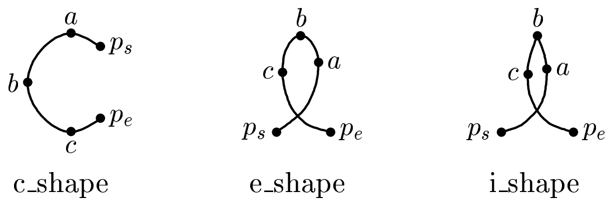

Figure 1: *Examples of basic shapes.*

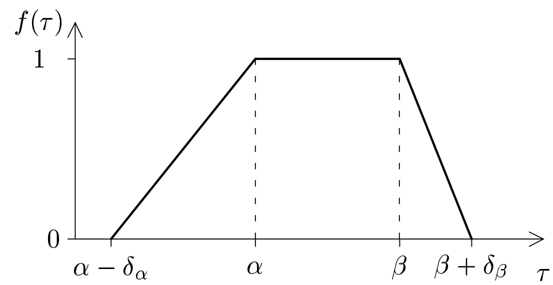

Figure 2: *Characteristic function for fuzzy-thresholds.*

properties : an index number unique to the primitive (n), a discontinuity measure at the characteristic point (dc), slant measures at the starting point (ss), characteristic point (sc) and ending point (se), and curvature measures at the start (cs) and end (ce) of the primitive.

III.2 Modelling Methodology

The methodology followed for the modelling of allographs can be divided in three phases :

1. The creation of different classes of primitives : "continuous", "discontinuous", "horizontal" and "vertical". The "continuous" class contains primitives that have a *low* discontinuity measure (dc). Reciprocally, the "discontinuous" class contains primitives with *high* discontinuity measure. The "horizontal" class contains primitives that globally correspond to *horizontal* displacements (sc). Likewise, the "vertical" class contains primitives corresponding to *vertical* displacements. These primitive classes and their combinations are defined by grammars that apply fuzzy-thresholds on the properties of the $\mathcal{AHP}$. It should be noted that these primitive classes are fuzzy-sets and that a particular $\mathcal{AHP}$ can belong to several of them, for example to the "continuous" and "discontinuous" classes if its discontinuity measure is *medium*, although not necessarily with the same grade of membership.

2. The modelling of basic shapes like the *c_shape, e_shape* and *i_shape* of Fig. 1. These shapes are modelled by grammars that assemble the different possible combinations of primitive classes that make up the shape. For example, the *c_shape* is constituted by at least one "continuous vertical" primitive possibly preceded and followed by one "continuous horizontal" primitive. Of course, not all combinations are allowed. Slant and curvature Constraints are imposed. All basic shapes are defined with a starting point (p_s), an ending point (p_e) and several other attachment points that generally correspond to the characteristic points of the constituent primitives (points a, b and c of Fig. 1). However, all attachment points are not necessarily disjoint. For example in the *c_shape*, points p_s and a may be confounded if no preceding "continuous horizontal" primitive exist. Also, all basic shapes are defined with properties representing their starting and ending index number (n_s and n_e), their starting and ending slant (s_s and s_e) and their starting and ending curvature (c_s and c_e) so that further constraints may be imposed if needed for a particular allograph model.

   Again, it should be noted that, contrary to most other structural cursive script recognition method that have used these types of basic shapes, the existence of an *e_shape* (a loop), for example, does not override the existence of the *c_shape* that is part of it. Both shapes can coexist and be used in different allograph models.

3. The final phase is the modelling of the allograph themselves. Each of them is assembled from basic shapes and morphological characteristics are defined and tested to take the final decision. Section III.3 gives an example of an allograph model.

Fuzzy-thresholds in the $\mathcal{HAD}$ language are defined by a characteristic function $f(\tau)$ illustrated by Fig. 2. The syntax is : $\tau[\alpha, \beta, \delta_\alpha, \delta_\beta]$, where τ is an integer expression, α and β are respectivly the lower and upper bound for a *true* result ($f(\tau) = 1$), and δ_α and δ_β are the lower and upper range for a *fuzzy* result ($0 < f(t) < 1$). Other types of characteristic functions can be found in the literature but, however, we have not tried them so far. This one was chosen for its simplicity.

Slant and Curvature Constraints can be specified in the $\mathcal{HAD}$ language by using macro-definitions that

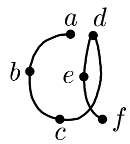

Figure 3: *An allograph of letter* a.

define fuzzy-thresholds. For slant constraints, the macro *slant*(θ_1, θ_2) can be used to limit slant from θ_1 to θ_2 with two equal fuzzy-ranges fixed by a constant.

For curvature constraints, three macros were defined for three types of curvature corresponding to three different fuzzy-thresholds : *cpos* for positive curvature[5], *cneg* for negative curvature and *crect* for rectilinear curvature (null curvature). No additional precision was deemed necessary.

Morphological Characteristics are relative measures of size, that characterize each allograph model. Consider, for example, the principal allograph of lower case letter *a* illustrated in Fig. 3. If one would move point f (and point e with it) up to approximatly point d, then one would get an allograph of letter *o*. Likewise, if point d is moved up sufficiently relative to point a, then we get an allograph of letter *d* or, if point f is moved down sufficiently we get an allograph of letter *q*. Hence, for a same combination of basic shapes, it is possible to form different allographs of different letters. Morphological characteristics are thus defined to discriminate between them.

III.3 An Allograph of Letter a

The allograph of Fig. 3 is simply defined as a *c_shape* followed by either an *i_shape* or an *e_shape*. The listing in $\mathcal{HAD}$ of the grammar associated to this allograph follows :

```
symbol a[p1,p2][m] {
  symbaux ca1[a,b,c,d,e,f][] :=
    (c_shape,i_shape)
       if(sequ(@1.ne,@2.ns) & (crect(@2.ce) |
          cpos(@2.ce)) & slant(@2.se,-135,45))
      {a = xypnt(@1.ps,@1.a); b = @1.b; c = @1.c;
       d = @2.b; e = @2.c; f = @2.pe;},
    (c_shape,e_shape)
       if(sequ(@1.ne,@2.ns) & (crect(@2.ce) |
          cpos(@2.ce)) & slant(@2.se,-135,45))
      {a = xypnt(@1.ps,@1.a); b = @1.b; c = @1.c;
       d = @2.b; e = @2.c; f = @2.pe;};
} := (ca1) if(yratio(@1.a,@1.d,@1.a,@1.c)[-30,30,20] &
              yratio(@1.c,@1.f,@1.d,@1.c)[-30,30,20] &
              xratio(@1.e,@1.a,@1.e,@1.b)[-50,30,20])
       {p1 = xypnt(@1.b,@1.c); p2 = xypnt(@1.d,@1.a);
        m = 1;};
```

The reserved token *symbol* starts a grammar definition for the symbol that follows. Likewise, the token *symbaux* starts the definition of an auxiliary (non terminal) symbol. All auxiliary symbols are defined within the scope of two brackets before the production rules of the symbol. Here, symbol a is defined with two attachment points p_1 and p_2 which are used to identify the bounding box of the main body of the letter (letters with ascenders or descenders have an additional attachment point) and are employed for constructing the allograph segmentation graph (not described in this paper) that represents possible allograph adjacency ; and one property m used to identify the allograph model (in this case $m = 1$).

Auxiliary symbol *ca1* defines candidates for allograph model $m = 1$ (see Fig. 3). Two production rules are specified for *ca1* : one for the combination *c_shape* with *i_shape* and one for *c_shape* with *e_shape*. Predicate *sequ* tests for two shapes in sequence relative to their ending and starting indexes. In an expression, to specify an attribute of a constituent, the @ operator is used. The notation *@1.ne* mean "attribute *ne* of the first constituent". The applicability condition of the rule follows the token *if* and is delimited by parentheses. Operators & and | correspond respectivelly to fuzzy *and* and *or* operators implemented with the usual *min* and *max* functions. Generation rules for the attributes are enumerated after the applicability condition and are delimited by two brackets. Macro-definitions *crect*, *cpos* and *slant* have already been discussed. As for macro *xypnt*, it is used to extract the x and y coordinates from its first and second arguments to form a new point.

Finally, macro-definitions *xratio* and *yratio* compute in percent, the ratio of the difference of the first two arguments over the difference of the last two in order to extract the morphological characteristics for the model, namelly : $\frac{a_y - d_y}{a_y - c_y}$, $\frac{c_y - f_y}{d_y - c_y}$, $\frac{e_x - a_x}{e_x - b_x}$; where ρ_x and ρ_y denote respectivelly the x and y coordinates of point ρ. When the fourth parameter of a fuzzy-threshold is not specified, it takes automaticaly the value of the third.

This example illustrates one particular allograph of the letter *a*. In general, several different models exist for each letter but nevertheless their number is limited. In practice, their is usualy two main models : the cursive type that is taught in primary schools

[5]Positive curvature correspond, by definition, to counter-clockwise curves.

(for which variants can be modeled) and the printed character type that is frequently used to write more clearly.

IV Allograph Segmentation

Allograph segmentation is conducted by a parsing process described in this section. The objective is to determine if, from an initial set of primitive objets, there exist subsets of these that could be generated by any of the defined grammars. First, a few definitions are made and then the parsing mechanism is developed.

IV.1 Definitions

We use the term *object* to designate an instance of a particular *symbol* which itself can represent either an allograph (i.e. a particular model of a letter), any pertinent sub-allograph (i.e. a part of an allograph) or a primitive (in this later case we might use the expression *primitive object*).

The only role of the parser process is to assemble objects, initially starting with primitives objects, according to the set of production rules P_i of a grammar G_i. An object o is a structure that contains several informations :

1. a set of constituent objects $\mathcal{C}(o) = \{c_1, \ldots, c_m\}$
2. a set of attachment points $\mathcal{A}(o) = \{a_1, \ldots, a_u\}$
3. a set of properties $\mathcal{P}(o) = \{p_1, \ldots, p_v\}$
4. and a grade of membership to its class $\alpha(o)$

The constituents are the objects that were assembled to form the new object. The attachment points and properties are computed from the attributes of the constituent objects using the generation rules of the production rule that created the object.

Each object is associated to a particular symbol, that is, belongs to a particular class of objects which corresponds to a fuzzy-set. Let C denote the applicability condition of the rule that created an object o. Then, the *grade of membership* α of that object is defined by :

$$\alpha(o) = \min\left[C(o), \min_i \alpha(c_i)\right] \ , \quad 1 \leq i \leq m \qquad (2)$$

where $\alpha(c_i)$ is the grade of membership (to its own class) of the i^{th} constituent of o.

The *domain* $\mathcal{D}$ of an object o is defined by the set of primitive objects that are either direct constituents of o or, recursively, constituents of its constituents :

$$\mathcal{D}(o) = \sum_{i=1}^{m} \mathcal{D}(c_i) \qquad (3)$$

An object o is said to be *redundant* relative to an other object o' of the same class if and only if its domain is completly included in the domain of o' and its grade of membership is lower or equal :

$$\exists\, o' \neq o \ : \left\{ \begin{array}{c} \mathcal{D}(o) \subseteq \mathcal{D}(o') \\ \text{and} \\ \alpha(o') \geq \alpha(o) \end{array} \right. \iff \text{redundant}(o) \quad (4)$$

Two objects o and o' are said to be *consistent* if and only if the intersection of their domains is empty :

$$\mathcal{D}(o) \cap \mathcal{D}(o') = \emptyset \iff \text{consistent}(o, o') \qquad (5)$$

Two objects o and o' are said to be *adjacent* if and only if the distance between their respective bounding box along the X axis[6] is smaller than a certain threshold. The bounding box of an object corresponds to the smallest imaginary rectangle that completly encloses its attachment points.

IV.2 Parsing Process

We now consider the problem of parsing grammar $G_i(T_i, N_i, P_i, S_i)$ associated to symbol S_i. Let $T_i = \{T_i^1, \ldots, T_i^j, \ldots, T_i^p\}$ be the set of p terminal symbols. For each symbol T_i^j, a set of objects D^j is knowned and constitutes the starting data :

$$D^j = \{d_1^j, d_2^j, \ldots, d_{p_j}^j\}, \qquad 1 \leq j \leq p \qquad (6)$$

It is subsets of these objects that can, depending on the set of production rules P_i, correspond to constituent objects of symbol S_i. In fact, the objective of the parsing process is to identify these subsets that, possibly, at step $i+1$ will serve to construct objects of S_{i+1}.

Let $N_i = \{N_i^1, \ldots, N_i^j, \ldots, N_i^q\}$ be the set of q non-terminals. Without any loss of generality — because of the stratified formalism — we can consider the problem of parsing symbol N_i^j. Then we seek to find

[6] This definition is used to limit combinatory explosion. Obviously, it is relevant to the parsing of the roman alphabet for which letters are aligned on a common baseline. For other types of script, it might not be justified.

the set O^j of objects that respect the production rules of N_i^j :

$$O^j = \{o_1^j, o_2^j, \ldots, o_{q_j}^j\}, \qquad 1 \leq j \leq q \tag{7}$$

Ultimatly, it is the set O^q — the objects associated to $N_i^q = S_i$ — that we seek.

Let $\lambda = \{c_i^{j.1}, \ldots, c_i^{j.l}, \ldots, c_i^{j.m}\}$ be the set of m constituents of N_i^j for a particular production rule, where $c_i^{j.l}$ represent the symbol of the l^{th} constituent of symbol N_i^j for which its associated object set $O^{j.l}$ is knowned (i.e. either $c_i^{j.l}$ corresponds to a terminal symbol, say T_i^k, and thus $O^{j.l} = D^k$, or $c_i^{j.l}$ is a nonterminal that can and must be parsed before N_i^j). Then the set $\mathcal{O}$ of possible assemblies for objects of N_i^j is constructed in the following manner :

$$\mathcal{O} = O^{j.1} \otimes O^{j.2} \otimes \cdots \otimes O^{j.m} \tag{8}$$

with the product $A \otimes B$ defined as follows :

$$A \otimes B = \{x \cup y |\; x \in A,\, y \in B \wedge x \bowtie y\} \tag{9}$$

and where the notation $x \bowtie y$ should be interpreted as x is *consistent* (§IV.1) with and *adjacent* (§IV.1) to y.

Then, the set O^j is given by :

$$O^j = \left\{ o \in \mathcal{O} \;\middle|\; C\left(\sum\{\mathcal{A}(c) \cup \mathcal{P}(c) |\; c \in \mathcal{C}(o)\}\right) > 0 \right\} \tag{10}$$

where C is the applicability condition of the production rule which is a function of the sum of the attributes ($\mathcal{A}$ and $\mathcal{P}$) of the set of constituent objects $\mathcal{C}(o)$ of o.

The attributes of each object o_w^j of set O^j are then computed using the set of generation rules G of the production rule :

$$\begin{aligned} &\forall a_k \in \mathcal{A}(o_w^j), \quad 1 \leq w \leq q_j, \\ &\quad a_k = G_{a_k}\left(\sum\{\mathcal{A}(c) \cup \mathcal{P}(c) |\; c \in \mathcal{C}(o_w^j)\}\right) \end{aligned} \tag{11}$$

$$\begin{aligned} &\forall p_k \in \mathcal{P}(o_w^j), \quad 1 \leq w \leq q_j, \\ &\quad p_k = G_{p_k}\left(\sum\{\mathcal{A}(c) \cup \mathcal{P}(c) |\; c \in \mathcal{C}(o_w^j)\}\right), \end{aligned} \tag{12}$$

where G_{a_k} and G_{p_k} represent respectively the generation rules for attachment point a_k and property p_k of the production rule.

When symbol N_i^j possesses multiple production rules, they are applied sequentially. Non-redundant (§IV.1) objects are appended to set O^j :

$$O^j = \left\{ o \in \sum_r O_r^j \;\middle|\; \neg\text{redondant}(o) \right\} \tag{13}$$

where O_r^j denotes the parse result for the r^{th} production rule.

V Preliminary Results

Intrinsic letter models for all 26 lower case letters of the alphabet have been developed and optimized with a multi-writer isolated cursive letter database containing 10 samples of each letter written by 13 different writers (total : 3380 cursive letters). These writers were instructed to reflect on their own writing styles or on any other style that they knew of, and write 10 samples of each letter representing possible variants, including ligatures that can precede or follow the letter. A similar test database has also been built but with 10 different writers.

Two informations are available for recognition : the grade of membership of parsed allograph objects and the proportion of $\mathcal{AHP}$ primitives contained in their respective domains, relative to the total number of primitives in the cursive trace. Let $O_{i,j}^{\nu}$ be the set of objects parsed in sample i of letter ν for writer j of the database. Then $\forall o_\sigma \in O_{i,j}^{\nu}$ parsed in a particular letter sample of the database, the letter recognized is the one associated with the object which maximises the following ranking coefficient :

$$\Gamma(o_\sigma) = \alpha(o_\sigma)\frac{\text{card}[\mathcal{D}(o_\sigma)]}{\eta} \tag{14}$$

where card$[\mathcal{D}(o_\sigma)]$ is the number of $\mathcal{AHP}$ in o_σ's domain and η is the total number of $\mathcal{AHP}$ in the letter sample.

Then, the recognition rates $\mathcal{R}(\mu, \nu)$ for model μ and letter ν, given in table 1 for the test database, are computed by :

$$\mathcal{R}(\mu,\nu) = \frac{\text{card}\left\{\max\{\Gamma(o_\sigma) |\; o_\sigma \in O_{i,j}^{\nu}\} \;\middle|\; \begin{array}{l} i \in \{1,\ldots,n_w\} \\ j \in \{1,\ldots,n_s\} \end{array}, \sigma = \mu \right\}}{n_w n_s} \tag{15}$$

where n_w is the number of writers in the database (13), n_s is the number of samples per writer (10) and where, as before, card$\{\,\}$ is the number of elements in set $\{\,\}$.

Table 1 shows average rates for correct recognition, that is when $\mu = \nu$ (the diagonal), but also confusion rates, that is when $\mu \neq \nu$. The average correct recognition rate over all letters is 91.7% with the lowest being 76% (for letter v) and the highest 100% (for letter n). Their are two explanations for confusion errors. Either the right model was not segmented in the cursive trace, or another model was segmented with a higher Γ. In the former case, either the cursive trace was too degenerated to be segmented by model ν (and thus the model is not to blame), or model ν is not adequate and needs to be tuned. Then again, in the latter

Table 1: *Recognition rates for models (columns) and letter samples (rows), in percent.*

letter samples	allograph models by letter																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	96	–	4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
b	–	90	–	–	4	1	–	–	–	–	–	1	–	–	1	–	–	–	1	1	1	–	–	–	–	–
c	–	–	95	–	2	–	–	–	2	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
d	2	–	–	97	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
e	–	–	1	–	99	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f	–	4	–	–	2	84	–	–	1	–	–	4	–	–	–	–	–	–	–	5	–	–	–	–	–	–
g	–	–	–	–	–	–	95	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	5	–
h	–	–	–	–	–	1	–	88	–	–	1	8	–	1	–	–	–	–	–	–	–	–	1	–	–	–
i	–	–	–	–	–	–	–	–	94	4	–	1	–	–	–	–	–	1	–	–	–	–	–	–	–	–
j	–	–	–	–	–	–	–	–	2	98	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
k	–	–	1	–	–	–	–	1	1	1	88	4	–	–	–	–	–	3	1	–	–	–	–	–	–	–
l	–	–	–	–	1	–	–	–	1	–	–	97	–	–	1	–	–	–	–	–	–	–	–	–	–	–
m	–	–	–	–	–	–	–	–	–	–	–	–	94	6	–	–	–	–	–	–	–	–	–	–	–	–
n	–	–	–	–	–	–	–	–	–	–	–	–	–	100	–	–	–	–	–	–	–	–	–	–	–	–
o	–	–	1	–	1	–	–	–	–	–	–	–	–	–	92	–	–	–	–	–	–	6	–	–	–	–
p	–	–	–	–	–	–	–	–	–	–	–	2	–	–	1	97	–	–	–	–	–	–	–	–	–	–
q	–	–	2	–	2	–	–	–	–	–	–	–	–	–	1	–	95	–	–	–	–	–	–	–	–	–
r	–	–	2	–	–	–	–	–	3	–	–	–	–	–	–	–	–	87	2	3	–	1	–	1	–	1
s	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	97	2	–	–	–	–	–	–
t	–	–	–	–	–	–	–	–	3	–	–	4	–	–	–	–	–	3	–	89	–	–	–	–	–	1
u	1	–	–	–	2	–	–	2	4	–	–	1	–	–	–	–	–	3	–	–	87	–	–	–	–	–
v	–	–	1	–	–	–	–	–	2	1	–	9	–	–	1	–	–	4	–	–	6	76	–	–	–	–
w	–	–	–	–	–	–	–	–	1	–	–	1	–	1	–	–	–	–	–	–	11	–	86	–	–	–
x	–	–	–	–	–	–	–	–	1	–	–	–	–	1	–	–	–	3	–	–	1	–	–	93	1	–
y	–	–	–	–	–	3	2	–	–	9	–	–	–	–	–	–	–	–	–	–	–	–	–	–	86	–
z	–	–	3	–	–	–	–	–	–	1	–	–	–	–	–	–	–	7	6	–	–	–	–	–	–	83

case, either model ν is not adequate (presumably for not generating a high enough grade of membership), or model μ is too permissive and should be tuned. Obviously, this "tuning" is time consuming and must be carried out very carefully.

Table 2 gives average recognition results for each individual writer. These results show that the segmentation method is quite robust over different handwriting styles. Especially considering the fact that writers #4, 6, 8 and 10 did not participate in the construction of the learning database.

VI Conclusion

This paper has presented a new approach for segmenting intrinsic allograph models in samples of cursive script. Although results are given only on isolated cursive letters, the presence of other letters in no way affects the segmentation process onless the shape of the letters is altered. Moreover, many different styles of handwriting have been considered in building the models, but without trying, at all cost, to fit the models to the data when this would leed to a non general solution.

This new approach uses a stratified shape grammar formalism for which a parsing algorithm has been described. The usual combinatory explosion associated with this type of parser was adressed by imposing consistency and adjacency constraints which proved to be efficient for the considered application.

Intrinsic letter models were built for all 26 lowercase letters of the roman alphabet. Average results over all allograph models yield a recognition rate of 91.7% which is very encouraging considering the difficulty of the database and the fact that the segmentation method is truly multi-writer (parameters are the same for all writers).

The method presented in this paper for modelling and segmenting allographs in cursive script, is part of a recognition method based on morphological and pragmatic knowledge only. The advantage of not using linguistic knowledge is to be able to recognize

Table 2: *Recognition rates for each writer, in percent.*

Writer										average rate
#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	
89.2	96.2	89.6	91.9	92.3	95.4	95.0	92.3	84.2	90.4	91.7

cleanly written script (as humans can do), without being dependant on linguistic context. However, this is not to say that the proposed method can work well in all situations and, in fact, the authors believe that the ultimate cursive script recognizer will make use of many different approaches.

References

[1] Tappert C.C., Suen C.Y., Wakahara T., "The State of the Art in On-Line Handwriting Recognition", *IEEE trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-12, no. 8, p. 787-808, August 1990.

[2] Suen C.Y., Berthod M., Mori S., "Automatic Recognition of Handprinted Characters — The State of the Art", *Proceedings of the IEEE*, vol. 68, no. 4, p. 469-487, April 1980.

[3] Parizeau M., "Reconnaissance d'écriture cursive par grammaires floues avec attributs : étape vers la conception d'un bloc-notes électronique", Ph.D. Thesis, École Polytechnique de Montréal, to be published summer 1992.

[4] Oulhadj H., Petit E., Lemoine J., Gaudaire M. and Lorette G., "A Prediction-Verification Strategy for Automatic Recognition of Cursive Handwriting", *Computer Processing of Handwriting*, Plamondon R. and Leedham C.G. (editors), World Scientific, 1990.

[5] Davis L.S., Henderson T.C., "Hierarchical Constraint Processes for Shape Analysis", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, no. 3, p. 265-277, 1981.

[6] Henderson T.C., Samal A., "Shape Grammar Compilers", *Pattern Recognition (GB)*, vol. 19, no. 4, p. 279-288, 1986.

[7] Plamondon R., "Steps Towards the Production of an Electronic Pen Pad", *Proc. International Conference on Document Analysis and Recognition*, held at St-Malo, September 30 to October 2, 1991, pp. 361-371.

[8] Fu K.S., "A Step Towards Unification of Syntactic and Statistical Pattern Recognition", *IEEE trans on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 2, pp. 200-205, 1983.

[9] Zadeh L.A., Fu K.S., Tanaka K., Shimura M., (editors) : *Fuzzy Sets and their Applications to Cognitive and Decision Processes*, Academic Press, 1975.

[10] Dubois D., Prade H., *Théorie des possibilités — Applications à la représentation des connaissances en informatique*, Masson, Paris, 1985.

[11] Parizeau M., Plamondon R., "A Handwriting Model for Syntactic Recognition of Cursive Script", accepted for publication at the 11th International Conference on Pattern Recognition, to be held at The Hague, August 31 to September 3, 1992.