A Generic Method for Eager Interpretation of On-Line Handwritten Structured Documents

Sébastien Macé                Éric Anquetil
IRISA - INSA
Campus Universitaire de Beaulieu
35042, Rennes Cedex, France

Abstract

In this paper, a new approach for on-line handwritten structured document interpretation is presented. It aims at interpreting the strokes progressively. The major component of our approach is a flexible formalism for the recognition of the document elements. Its originality is the modeling of the document global structure. The system then drives dedicated recognizers and looks only for the likely symbols depending on the document structural context in which that element is located. Moreover, the formalism defines a “canonical on-line signal form” of the recognizer entries to facilitate the interpretation process which is then more robust and more efficient. To highlight its genericity, this approach has been used to design two pen-based prototypes, for musical score editing and for graph editing.

1. Introduction

The interpretation of strokes in the context of handwritten structured document composition, which we call on-line recognition, is a complex problem, because such documents are constituted of elements of various natures. Moreover, a stroke can have different interpretations depending on its drawing context which must so be taken into account [1][3].

Lazy recognition, which consists in analyzing handwritten documents once they are achieved [6], is a promising approach to offer unconstraint understanding of ink. But the difficulties to design automatic parsing coupled with robust handwritten shape recognizers show that it remains an open problem. Eager recognition, which consists in interpreting the strokes “on the fly”, directly while the user is drawing [2], constitutes another way to consider on-line structured document interpretation. The main difficulties are then to interpret incomplete documents and to make a decision as quickly as possible to avoid disturbing the user. Moreover, it is possible to exploit the interaction with the user, who is aware of the system answers. The main contribution of this paper is a new generic approach to design pen-based systems with an eager interpretation process. In order to interpret complex documents, we believe that it is necessary to have a global vision of their structure: we need to know what can be drawn and where it can be. So far, approaches were mostly based on local representation of the shape to recognize thanks to a description language [1][4]. But they do not enable the representation of too complex elements such as symbols or text, which can be recognized by powerful handwritten shape recognizers. The originality of our approach to recognize handwritten shapes in context of document is to use a coupling of a rule-based description of the document and its elements with the exploitation of handwritten shape recognizers. This approach is generic: it can be applied to various structured document natures, such as musical scores and graphs, which are the domains we present in this paper, but also electronic figures, plans, etc.

The following section presents the architecture of the system we developed. Section 3 focuses on its main component, a flexible formalism to model structured document interpretation, and the way it is exploited. Then section 4 describes two existing pen-based systems based on the presented approach. Finally, we summarize our future work.

2. Architecture of the system

The developed system, presented in figure 1, is based on a framework constituted of three main components. The first one is a set of graphical functions (display, zoom, etc.) and editing functions (selection, deletion, etc.) for pen-based systems (1). We do not present its complete possibilities because it is outside the scope of this paper. The second component is a formalism to model on-line document interpretation which can be adapted to various domains (2). Once compiled, this knowledge is directly used by the third
component, a generic analyzer which interprets the strokes eagerly by evaluating their document contexts and driving the handwritten shape recognizers (3).

Next sections focus on the formalism and its exploitation.

3. A formalism for eager interpretation

The basic concepts associated to our approach to model on-line structured document composition and eager interpretation can be summarized as:

- **the management of the chronological information**: as the interpretation process is eager, the system has to deal with incomplete documents; a way to do so is to model which element can be drawn after which other;
- **the representation of the document spatial structure**: it is necessary to exploit the relative positioning of the document elements to model on the one hand in which document structural contexts an element can be identified, and on the other hand which of these contexts are generated due to the creation of an element;
- **the driving of the recognition process by the document context analysis**: it is not possible to have a unique recognizer for all the symbols that a document can contain, because the more symbols a recognizer must interpret, the less efficient and the less robust it is. By exploiting the document context of an element, the system must reduce the likely symbols and choose which dedicated handwritten shape recognizers to use;
- **the pen-based human-computer interaction**: as the user is aware of the system answers, he can implicitly validate or explicitly reject them.

Our approach is based on a formalism which takes all these concepts into account. It is composed of interpretation rules which define how to generate the element which name they bear. Several rules can have the same name, which enables to model different ways to compose a same element.

A rule takes a set of elements as parameters and returns a new one that can replace them; a parameter is a component of the new element and can be either a stroke or an already interpreted element, which enables a hierarchical shape description. A rule can only be executed if the document contains all its parameters. The structure of a rule is composed of a document context verification (DCV) block, a shape context verification (SCV) block, a shape recognition (SR) block and a document context creation (DCC) block, as follows:

\[ \text{SymbolName(Parameter 1, \ldots, Parameter n)} \]

\[ \text{document context verification (DCV) block,} \]

\[ \text{shape context verification (SCV) block,} \]

\[ \text{shape recognition (SR) block,} \]

\[ \text{document context creation (DCC) block.} \]

On the one hand, the DCV and DCC blocks enable a global vision of the document to define in which document context an element must be. Such a context is both a positioning and the name of the rule of the expected element. On the other hand, the SCV and SR blocks enable, given this document context, a local description of the element to recognize.

Before presenting each of the rule blocks, the next subsection describes the analysis process.

3.1. Presentation of the analysis process

When an element is created, it defines new structural contexts in the document, specifying which elements can be situated in a given positioning, *i.e.* which rules must be activated on a stroke drawn there. This process is formalized in the DCC block. Thus, the first step of the analysis of an element consists in determining in which of these document contexts it has been drawn. Contexts are defined gradually: the analyzer calculates a membership degree of an element to a context and then sorts out contexts to which it belongs. The rules corresponding to these contexts are then called. As the parameters of a rule indicate the components of the element, these ones can exist in this context: so the rules corresponding to these parameters are activated recursively. As a consequence, the analyzer only tests relevant rules.

Each rule specifies the expected document contexts for its parameters in the DCV block; it can be empty if only one context, the one that activated the rule, is enough to try it. If the DCV block succeeds, local constraints, defined in the SCV block, are tested: they correspond to relative spatial arrangement of the parameters of the rule. If these constraints are satisfied, a dedicated shape recognizer can be called on the parameters; this is modelled in the SR block. Finally, if the shape is recognized, the rule can be applied and new document contexts are created in the DCC block to update the document spatial structure representation.

If no interpretation rule can be applied to the user stroke, this stroke is rejected and disappears from the interface. On the contrary, if a rule is activated, the new element is created and the current iteration of the interpretation process is over; a new iteration starts on the same scheme, trying to apply rules on this new element, and so on until stabilization. Convergence is ensured by coherent rules. If there is an ambiguity between several interpretations, *i.e.* if more than...
one rule can be applied, we select the one with the highest document context membership degree. If the difference between degrees is not significant, the stroke is rejected. Therefore, our approach takes fully advantage of recognizers with reject options since they filter possible answers [5]. We exploit the human-computer interaction to integrate the user in the recognition process: he can implicitly validate an answer by going on with the composition of the document, or explicitly reject it by drawing a deletion gesture. As a consequence, the stroke analyzer does not need to question a decision made beforehand, because it has been validated. The recognition process is thus more robust.

Next subsections present the modelling of the knowledge exploited by the analyzer. We illustrate the notations with music editing rules.

3.2. The document context verification and document context creation blocks

The purpose of the coupling of the DCV and the DCC blocks is the modelling of the document spatial structure.

• The DCV block specifies the document structural contexts which have to be verified by the rule to try it. It refers document contexts created in a DCC block. Its syntax is:

\[ DCV: A_1[\text{position}_1, \text{part}_1]B_1, \ldots, A_N[\text{position}_N, \text{part}_N]B_N. \]

This expression means that at the relative position \( \text{position}_i \) (e.g. on the left, above, etc.) of an element \( A_i \), the part \( \text{part}_i \) (e.g. one, all, the first, the highest points, etc.) of an element \( B_i \) exists. \( A_i \) and \( B_i \) do not have to be part of the parameters. For example, an accidental (i.e. a flat, a sharp or a natural) must be drawn on the left of an head; a context block for such a rule taking a stroke \( s \) as parameter can be written as:

\[ DCV: \text{Head}[\text{left}, \text{all}]s. \]

• The DCC block specifies the document contexts that are created due to the recognition of an element. Its syntax is:

\[ DCC: A_1[\text{position}_1, \text{part}_1]B_1^N, \ldots, A_N[\text{position}_N, \text{part}_N]B_N^N. \]

This means that at the relative position \( \text{position}_i \) of an element \( A_i \), the part \( \text{part}_i \) of an element \( B_i \) can exist. The current element is referenced as \( \text{this} \). The number \( n_i \) indicates how many \( B_i \) can exist in this context and is \( \ast \) if there is no limit. For example, the creation of a note head allows to draw one accidental on its left, one dot on its right and one stem, either uprising or descending; a DCC block and the corresponding document contexts created can be:

\[ DCC: \text{this}[\text{left}, \text{all}]\text{Accidental}, \]

\[ \text{this}[\text{right}, \text{all}]\text{Dot}, \]

\[ \text{this}[\text{rightUprising}, \text{all}]\text{Stem}, \]

\[ \text{this}[\text{leftDescending}, \text{all}]\text{Stem}. \]

Note that disjunction between contexts are possible and that aliases are defined to specify that a Head can be a WholeNote, a HalfNote or a QuarterNote, and that an Accidental can be a Flat, a Sharp or a Natural.

3.3. The shape context verification and shape recognition blocks

The purpose of the coupling of the SCV and the SR blocks is the recognition of an element: it distributes the recognition process among local constraints, formalized in the SCV block, and recognizers, formalized in the SR block.

• The SCV block explicits local constraints about the rule parameters to recognize an element constituted of several sub-elements: it enables to identify which part of the new element corresponds to each parameter. The syntax is the same as in the DCV block, but this time \( A_i \) and \( B_i \) must be parameters. There is no constraint on the drawing order of the elements; if this is necessary, a chronological operator can be used. For example, a sharp is constituted of two horizontal segments \( h_1 \) and \( h_2 \), and two vertical segments \( v_1 \) and \( v_2 \), which can be drawn in any order. A SCV block (in which the alias \( \text{intersects} \) is used for the position operator \( \text{in, one} \)) and its corresponding sharp figure are:

\[ SCV: hs_1[\text{intersects}]v_1, \]

\[ hs_1[\text{intersects}]v_2, \]

\[ hs_2[\text{intersects}]v_1, \]

\[ hs_2[\text{intersects}]v_2, \]

\[ hs_1[\text{below, all}]h_2, \]

\[ v_1[\text{right, all}]v_2. \]

These local constraints are not necessarily enough to ensure the four segments represent a sharp. So we may want to exploit a classical handwritten shape recognizer: this can be done with the SR block.

• The SR block corresponds to the call to a recognizer and makes it possible, depending on the context of an element, to call only the relevant recognizers; this is essential to increase the interpretation process’ robustness, since the less symbols a recognizer must interpret, the more efficient and the more reliable it is. The SR block syntax is:

\[ SR: \text{SymbolFamily}, \{\text{AcceptedAnswers}\}. \]

This expression means that the recognizer of SymbolFamily is called on the parameters of the rule; if its answer is included in AcceptedAnswers the recognition process is a success. The order of the strokes presented to the recognizer is the order of the parameters in the declaration of the rule; so, it is always the same. Actually, we also deduce from the SCV block an order for the points of each element (e.g. from the most left one to the most right one); thus, we model a “canonical on-line signal form” for the recognizer entries. As a result, the recognizer work is relieved, since it always has to interpret strokes in a same order.

For example, a SR block for the Sharp rule can be:

\[ SR: \text{Accidental}, \{\text{Sharp}\}. \]

Finally, the complete QuarterNote and Sharp interpretation rules can be as follows (we use a saving operator \( \rightarrow \) to associate an element to an identifier and reuse it afterwards):
4. Generic properties of the formalism

In order to highlight the generic properties of our approach, we present two systems which have been developed on tablet PC with this methodology: one for musical score editing and one for graph editing. For each system, we just needed to write the interpretation rules (about 40 for the musical score editor, about 10 for the graph editor), design the necessary recognizers based on Radial Basis Function Network (RBFN) classifiers and specify the graphical aspects of the document elements.

Figure 2 presents a sequence of screenshots of a pen-based musical score editor prototype. It enables the drawing of the main musical symbols, such as clefs (1), accidentals (2), notes (3,4), stems (5) beams (6) and line bars (7).

Figure 2. Musical score editor.

Figure 3 presents two screenshots of the graph editor prototype, on the left an interpreted document and on the right the corresponding user strokes. It enables the drawing of nodes, represented by geometrical shapes such as rectangles (2,8), triangles (6) and circles (4), and edges, represented by segments (1,3,7) or arcs (5). All these components can be drawn with one (1,2,3,4,5,6) or several strokes (7,8).

Figure 3. Graph editor.

For example, rectangles can be drawn with one or four strokes. The corresponding interpretation rules can be written as (empty blocks are not shown):

- **QuarterNote(Stroke s)**
  - DCV : Staff[in, all].
  - SCV : none.
  - SR : Notes.
  - DCC : cf. §3.2.

- **Sharp(HSeg h1, HSeg h2, VSeg s1, VSeg s2)**

- **DCV : Head → h1[left, all|hs1], h1[left, all|vs1], h1[left, all|vs2], SCV & SR : cf. §3.3.**

- **DCC : none.**

Note that in the left rule, all the recognition process is done by the RBFN recognizer of Geometrical Shapes, whereas in the right one, it is done by the local structural shape context: the **SCV** block is constraining enough to ensure the four segments constitute a rectangle.

5. Conclusion and future work

This paper has presented a new approach for on-line structured document interpretation, based on the driving of the shape recognizers by a structural analysis. We have highlighted the genericity and the flexibility of the formalism we introduced to model the interpretation of documents. The use of dedicated recognizers instead of a unique one increases the robustness of our system. It is operational and has been used to design pen-based interfaces for musical score editing and graph editing. Future work will aim at improving the decision-making procedure in case of ambiguity in order to design an even more effective approach.

6. Acknowledgements

Authors would like to thank Guy Lorette from the IMADOC team and Bruno Bossis from the MIAC team for their contribution to this work. This project benefits from the financial support of the Brittany Region.

References