MVA'90 IAPR Workshop on Machine Vision Applications Nov. 28-30,1990, Tokyo

Knowledge-Based System for Structured Document Recognition

A. Belaïd*, J.J. Brault** and Y. Chenevoy*

* CRIN - INRIA Lorraine, Campus Scientifique - B.P. 239,
54506 Vandœuvre Cedex, France. **Email** : abelaid@ loria.crin.fr

** Ecole Polytechnique de Montréal, Dépt. Gén. Elect., C.P. 6079,
Succursale A, Montréal, Canada. **Email** : brault@vlsi.polymt1.cat

Abstract

This paper discribes a document analysis system broadly consisting of a knowledge base, a blackboard and a set of tasks having their own set of spacialists for segmentation, recognition and for inheritance. The knowledge base contains a generic hierarchical description of the document structure in terms of layout objects labeled logically. This allows the generation of hypothetic networks of linked objects in the blackboard. The specialists cooperate indirectly through the blackboard by updating the layout object descriptors. A blackboard modification causes an "event" to propagate up to some specific tasks. A task could then choose another subset of specialists to carry on with the process. Finally, a synthesized blackboard summary allows a task selector to focus efficiently on the most useful layout object to process.

1 Introduction

A document is a complex pattern made up of different kinds of graphical symbols and organized with an underlying structure. Document recognition is thus a task that requires the cooperation of various types of processes such as segmentation, analysis, learning and recognition. These processes have to take into account a great number of different pertinent and inherent document information which are often heterogeneous, incomplete, inaccurate and uncertain. Optical character recognition only makes sense when applied to a real document with a minimal structure providing a context for the recognition. Each character cannot always be treated separately and entities such as words, lines and paragraphs composing the document give particular and helpful contexts thereby reducing ambiguities. It is obvious that to correctly exploit this context, it is necessary to have properly defined the document structure, including the logical chaining of different entities and their layout presentation on the document physical supports.

This growing interest for document structuring [2,7] is now emphasized by international standards like ODA [6] and SGML [5]. A lot of work has been done in this field during the past decade [3,12]. Nowadays in optical reading, several systems try to take into account the document structure as a fundamental context for the recognition. Among the earliest works in this field, we can find general systems such as in [10,11,14] and more specific applications like postal address recognition [13], reading chess [1], business letter interpretation [8] and recognition of legal documents [4].

GRAPHEIN is a general-purpose system that could deal effectively with a variety of document classes. It is able to organize and control the diverse document recognition processes in a flexible and efficient manner. Section 2 presents the classes of document structure adopted and the knowledge sources taken into account in the GRAPHEIN project. The system architecture and the control structure will be detailed respectively in sections 3 and 4. Finally, we conclude with a discussion on the opportunity of such an architecture and propose further improvements.

2 Document structures

GRAPHEIN wants to process the class of documents that could be described with ODA-like specification. In the ODA standard, two complementary structures are defined : the layout and the logical structures. Each one is a set of generic classes representing objects and their components. The logical structure precises only three classes which are : "document", "composite objects" (corresponding to recursive objects like chapters and sections) and "basic objects" : like paragraphs. The layout structure is essentially composed by "document", "set-of-pages", "page", "frame" (corresponding to a rectangular region within a page that can be broken down into other regions or blocks) and "block". Blocks and basic objects represent leaves in the corresponding hierarchies and cannot be broken down.

The two structures are linked together by a common entity called "content architecture" which represents the effective contents of both hierarchies. They correspond essentially to characters (attributes giving font description, line progression, alignment, line justification, etc.), to raster graphics (attributes necessary for the encoding of pel array), and to geometric graphics (attributes controlling the rendition of primitives like markers, lines, filled areas and text). The class description of each non-elementary object in the structures is given by a constructor which describes the nature of the composition of its subordinate objects. Possible constructors are "sequence", "aggregate" and "choice". Furthermore, each object can be "optional", "required", "repetitive" or "optional-repetitive" [6].

In GRAPHEIN, we considered that we only need a layout structure labeled by logical tags but reinforced by additional topographic relationships and overflow considerations. We give in the following the list of the constructors and qualifiers derived from ODA and adopted in GRAPHEIN :

Constructors :

- SEQ-TB($\mathbf{o}_1\ \mathbf{s}_{1,2}\ \mathbf{o}_2\ \mathbf{s}_{2,3}\ \ldots \mathbf{s}_{n-1,n}\ \mathbf{o}_n$) : describes an ordered sequence from top to bottom of n objects, where each object $\mathbf{o}_i$ is separated from its successor $\mathbf{o}_{i+1}$ by a specific separator $\mathbf{s}_{i,i+1}$.
- SEQ-LR($\mathbf{o}_1\ \mathbf{s}_{1,2}\ \mathbf{o}_2\ \mathbf{s}_{2,3}\ \ldots \mathbf{s}_{n-1,n}\ \mathbf{o}_n$) : the sequence is ordered from left to right.
- AGG-HB($\{\mathbf{o}_1\ \mathbf{o}_2\ \ldots \mathbf{o}_n\}\ \{\mathbf{s}_1\ \mathbf{s}_2\ \ldots \mathbf{s}_m\}$) : describes an aggregate of n unordered objects separated by one of the "m" separators.
- AGG-GD($\{\mathbf{o}_1\ \mathbf{o}_2\ \ldots \mathbf{o}_n\}\ \{\mathbf{s}_1\ \mathbf{s}_2\ \ldots \mathbf{s}_m\}$) : the aggregate is from left to right.
- CHO[$o_1\ o_2\ \ldots o_n$] : describes the choice of an object $\mathbf{o}_i$ from n objects.
- MOS($\{\mathbf{o}_1\ \mathbf{o}_2\ \ldots \mathbf{o}_n\}\ \{\{\mathbf{r}_{i,j}\},\ i \neq j\}\ \{\mathbf{s}_{i,j}\},\ i \neq j\}$) : describes a mosaic of n objects completely connected by topographic relationships $\mathbf{r}_{i,j}$ and separated by $\mathbf{s}_{i,j}$.

Qualifiers : The qualifiers concerning the occurence of a subordinate object (i.e. optional (OPT), required (REQ), repetitive (REP) or optional and repetitive (OPT-REP)) are kept just as ODA defines them. On the other hand, ODA provides no qualifier to express optional objects under certain conditions. It seemed important to us to add another qualifier optional conditional (OPTC <condition>) that can manage several problems such as the overflowing of a logical object onto several layout objects (to be seen in section 3).

Figure 1 shows the different elements of this physical-logical structure representing the regions found in the first page ("page_title") of a class of scientific papers. This page is made up of a sequence, from top to bottom (SEQ-TB) of two frames : "Frame_title" and "Frame_sections", separated by a horizontal space : "H-sep". "Frame_title" is composed of a sequence, from top to bottom, of three compulsory (REQ) blocks : ("Block_title", "Block_author" and "Block_address") separated by a horizontal spacing block. "Frame_sections" is composed of a sequence, from left to right, of two compulsory frames (columns) : "Frame_c1" and "Frame_c2", separated with a vertical spacing block : "V-sep". The abstract and the following paragraphs of the article (represented as blocks : "A", "B" and "C") are separated by horizontal separators and can overflow from "Frame_c1" to "Frame_c2". These blocks are qualified by the overflow conditions described in the "optc" attribute. The separators are considered as blocks and described directly by their content architecture.

3 System configuration

GRAPHEIN is a system for document understanding based on the exploitation of the structural context inherent to the document under study. It produces as output a processible file containing the text embedded with layout and logical tags (ODL-like encoding) describing the structure of the document. We will briefly describe in the following subsections the principal knowledge sources and an example of a hypothesis generation strategy. The next section describes the knowledge organization and control.

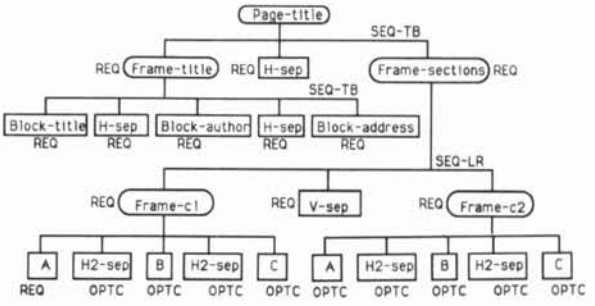

Figure 1: *Example of document structures in* GRAPHEIN.

3.1 The principal knowledge sources

The knowledge sources used in the system could be partionned into character recognition, document segmentation and structure analysis. Also an *a priori* model hypothesizes on the particular structure of the document and guides the system processes. The segmentation operates on the scanned image and tries, from the hypotheses, to find the real frontiers of the regions to recognize. The character recognition is operated from a multifont learning base and completed by a lexical analysis.

3.2 Hypothesis generation strategy

As the model is generic, many hypotheses may be given at any moment on one region. The hypothesis generation in GRAPHEIN tries to reduce the number of hypotheses by using many strategies. We resume one of them in two cases :

3.2.1 Limited hypotheses

When the number of hypotheses is limited, such as the example in the fig. 1, the system could select the "best" one by computing which region belongs to the fewer hypotheses. This information is given by the following formula :

$$I(o_i/R_j) \;=\; -\log_2(P(o_i/R_j)/P(o_i))$$

where $P(o_i/R_j)$ is the probability of observing the object "o_i" in the region of interest R_j, and $P(o_i)$ is an a priori probability of the observed object o_i. $P(o_i/R_j)$ is calculated from a "hypothesis table" shown in fig. 2 representing a case of overflow of three paragraphs A, B and C on to two regions Frame_c1 (R_1) and Frame_c2 (R_2). This table is automatically constructed from the model for every situation.

Hypotheses	Frame_c1			Frame_c2		
	o_1	o_2	o_3	o_1	o_2	o_3
h_1	A			A	B	C
h_2	A				B	C
h_3	A	B			B	C
h_4	A	B				C
h_5	A	B	C			C
$P(o_i/R_j)$	1.	.6	.2	.2	.6	1.
$I(o_i/R_j)$	0.	.74	2.3	2.3	.74	0.

Figure 2: *Example of hypothesis generation.*

It is interesting to focus the attention of the optical reader on the block with the maximum $I(o_i/R_j)$. The hypotheses can be taken as a function of the magnitude, in decreasing order, of the value of the $I(o_i/R_j)$ of their constituting elements. This hypothesis selection is represented by a decision tree.

When there is equality between two information measurements, we could add other tests using, for example, a complexity score as follows :

if $\forall j \neq i,\ I(o_i/R_1) > I(o_j/R_2)$ then
    test first o_i
else if $\forall j \neq i\ /\ I(o_i/R_1) < I(o_j/R_2)$ then
        test first o_j
    else if $S_c(o_i) \geq S_c(o_j)$ *then*
            test first o_i
        else test first o_j
        endif
    endif
endif

where $S_c(o_i)$ is a score of the extraction complexity in the document of the region o_i. We determine this quantity as follows :

$$S_c(o_i) = \prod_{k=1}^{j} C(e_k)$$

where e_k is the k^{th} subordinate element of the region o_i given by the model, and $C(e_k)$ is a complexity measurement of the sub-object e_k given directly by the model or estimated from its attributes. An example of $C(e_k)$ can be the average of attribute scores describing the content architecture of the block e_k in a region.

This algorithm favors the plausibility to the complexity to reduce rapidly the number of hypotheses given by the model.

3.2.2 Combinatorial Explosion

In this case, a primary selection is necessary to reduce the large number of hypotheses. The hypothesis graph obtained could be reduced again by the previous algorithm.

Two strategies are adopted for the primary selection. The first one consists of focussing on the easiest regions to extract. A score of complexity S_c is calculated on regions from their composition and the content architecture of their blocks. The second one is performed when the regions given by the model are not precise. In this case, a series of measurement algorithms are applied to the document parts and an aggregate of homogeneous regions is obtained. This aggregate is then matched against the model to reduce the hypothesis graph.

4 Knowledge organisation and control

We have adopted a blackboard architecture which is a form of knowledge-based problem-solving model capable of dealing with multiple cooperating knowledge sources. GRAPHEIN is partially implemented on top of a blackboard-building environment called ATOME [9]. ATOME organizes the domain knowledge into a set of individual computation modules known as *specialists* that are kept separate and independent. The blackboard is divided into levels which can be instantiated into nodes representing at each stage the current solution of the problem. It constitutes a shared working memory which the specialists use for operations such as viewing, creating, modifying or deleting *nodes*. In order to enable the various specialists to cooperate to find solutions, a control mechanism is needed. It is carried out by a set of knowledge-source controllers that is organized into two levels : *tasks* and *selector*.

The *tasks* coordinate the activities of the specialists. Each of them has a local control structure called *event-list* (Evt in fig.3) that contains specific events that occurred in the blackboard : creation, modification, substitution or deletion of nodes. For efficiency sake, each task declares the type of events that will be received in its local event-list. Once activated, it selects a subset of specialists and prepares them for possible activation.

The *selector* coordinates the task's activities. It accesses a data structure called *blackboard summary* which contains only information capable of influencing the remainder of the problem-solving.

Figure 3 gives a schema of the GRAPHEIN architecture. All knowledge-source controllers are made up of local production systems, while the specialists can be either local production systems or programs.

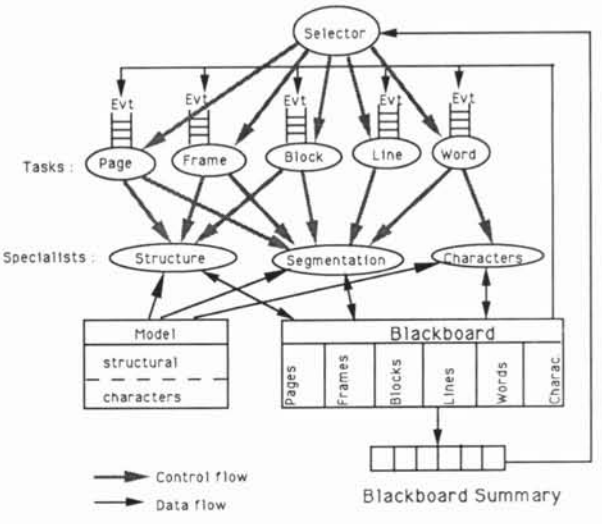

Figure 3: *The* GRAPHEIN *architecture.*

4.1 The blackboard

The blackboard is divided into 6 levels of abstraction, each corresponding to physical components. The defined levels are *page*, *frame*, *block*, *block-line*, *block-word* and *block-character*. Each level is represented by a set of attributes and links. Attributes of one level contain typographical and geometrical properties that characterize the type of the component it represents. They are organized according to the nature of the level and information they contain like font des-

cription, lines and character spacing, location of the components with respect to their neighbors. The attribute values which are weighted by a certainty coefficient, are computed by specialists based on the blackboard's current state. The links define relations between components of the blackboard. Each link has a reverse link that is automatically updated by ATOME so that all related components are easily accessible.

This graph of nodes reflects at all times the nature of the structural context available on the document.

4.2 The specialists

Specialists are composed of low-level programs for measurement and preprocessing, segmentation, character and word recognition, structure identification, and knowledge learning about the document under consideration. All specialists are composed of a precondition part and a body. The precondition is a state-based predicate required to activate the body. The latter is defined in GRAPHEIN as a program or as a set of rules when the specialist checks contextual methods.

Preprocessing and recognition specialists are composed of :

Measurement algorithms : to determine respectively the physical component's inclination, to redress the components and to extract particular features like italic, ligatured and underlined characters, etc.

Segmentation algorithms : to segment a component into subordinate objects according to the spacing and size information. We associate to them *merging algorithms* to reconstruct the hierarchy of the physical-logical structure.

Character and Word recognition algorithms : to perform feature extraction, identification of fonts and character recognition using Markov models and lexical analysis.

Structure identification algorithms : to confirm the document structure generally described by the model and partially given by the segmentation algorithms.

Hypotheses control specialists are composed of :

1- *model-inheritance* specialist which provides information on the model content in order to describe objects for the segmentation and recognition. This information is arranged in the blackboard in the form of a hypothesis graph with the different choices and options permitted by the model.

2- *compute-utility* specialist which compiles the hypothesis graph and determines the nodes that can rapidly lead to the solution.

3- *follow-hypothesis* specialist which assures the update of hypotheses such as the elimination of paths that have become less probable from the fact that their scores have been reduced by other specialists in the system.

4.3 The tasks

Tasks are meta-knowledge sources that manage a subset of specialists based on its local *event-list* state. GRAPHEIN integrates segmentation and recognition in tasks for each principal type of layout component such as *page, frame, block, line* and *word* in order to organize in an effective manner the specific treatment they need. We summarize as an example the function of two different tasks, *page* and *word* :

The page task :

This task has to organize the *page* processing. It has under its responsibility measurement, segmentation, recognition and *page* hypotheses specialists. It needs to be informed only of changes occurring at the *page* level of the blackboard, thus its local *event-list* will contain *pages* to be considered.

The idea is to search for information which can be inherited for each new page, and then to interpret the hypothesis graph with the specialist "compute-utility". Three cases can arise :

- There is no ambiguity on the page description. In this case, we apply a top-down segmentation of the page into frames. The specialist "seg-fr" propagates the events corresponding to the creation of new frames towards the task "frame". The specialist "rds-page" analyzes the results and either confirms them and reinforces the hypotheses scores or invalidates them and reduces the scores. In the last case, another strategy, for example bottom-up is chosen.

- There is no valid hypothesis given by the specialist "compute-utility". The task adopts a bottom-up strategy by merging the connected components into lines and blocks and matching them against the vague hypotheses within the model. This matching is realized by the specialist "rds-page".

- The hypothesis graph gives some interesting top priority points corresponding to some kinds of frames or blocks to locate (given by the specialist "compute-utility"). In this case, the task "page" interrupts its activities and transfers control to other tasks. Upon return, the task takes up its activities again from the same point but with more precise information.

The word task :

At this stage, there is practically no distinction between the physical and logical entities. This task facilitates the use of specific methods for segmentation and recognition of words by examining the type of problems that might have been signaled in the blackboard. Information like font-type, baseline, inclination etc. is either inherited directly from the line or block containing the word, or calculated if absent.

As a function of the information available, we either segment directly into characters or solve a problem first. For example, in the case of an underlined word, we first localize and erase the underline before segmenting. This group of physical components is then passed for recognition to a specialist that uses the Viterbi algorithm based on Markov models. In ambiguous cases, n-grams are used or verification with a lexicon is performed. When there are only non viable results (low score, too many alternatives etc.), the system furnishes indications on the type of anomalies encountered. The appropriate specialists are then alerted to solve the problems. For example, while removing the underline, ligatures might appear. Once these anomalies have been corrected, the word is sent back to the specialists in charge of segmentation and recognition.

4.4 The selector

Given a description of the "importance" of the nodes situated in a blackboard summary, the selector determines how and where to pursue the document processing. At each cycle of the problem-solving process, it selects the appropriate task in relation to the blackboard level of the node under consideration.

The importance of nodes is derived from segmentation and recognition scores obtained on the blackboard nodes. These scores reflect at all times the processing state of each node. For example, at node creation when no information is known, the system assigns a low score to this node to make it important and propagates this information into the summary blackboard. From this summary a focussing strategy can be operated by the selector.

In a more precise manner, when there are many important hypotheses in the blackboard, the selector has to use either the complexity score or the plausibility within the blackboard to select the best node to reduce the number of hypotheses. Once this node is selected, the corresponding task is alerted focussing attention to the related region.

4.5 Experiments and discussion

GRAPHEIN is developped on a SUN workstation in LISP, using ATOME, while the functions needing iconical representation of the image are written in C to improve the speed of resolution. The document acquisition is performed with a 300 dpi scanner attached to an IBM AT.

Global segmentation refines itself, according to the increase in knowledge about the document extracted and stored during the process. Since the spacing thresholds between the different physical components are more reliable, a top-down segmentation, focused on the specific component to analyze and its subset of knowledge (thresholds, dimensions...) gives better results than a bottom-up segmentation needing the fusioning of the components.

However, a great number of problems will not be resolved without a close collaboration with a recognition module which is realized separately. Thus, for example, the case of the local segmentation presents many difficulties unresolvable without dealing with the local linguistical context surrounding the block to segment. The local segmentation specialists already developped handle the case of overlapping characters (search for a passage through the block) and bounded characters. In order to improve the rate of correct segmentation of bounded blocks (around 75%), as well as to be able to reconnect the cut blocks, and to refine the evaluation of suspicious blocks, we aim at present, to attain a closer cooperation with the recognition stage. Using linguistical models such as trigrams of consecutive characters, a lexicon of current words with their declensions, grammar rules..., fairly good hypotheses can be made to help both the segmentation and recognition stages, avoiding a waste of time in the analysis of the graphical representation of each character.

5 Conclusion

We have presented in this paper a knowledge-based architecture designed for document recognition. This implementation has painted the type of knowledge to take into account and its benefits for a more effective recognition. We plan to refine the recognition process by taking more advantage of information related to the document, by finding the best strategy for the sequence of processing and by extending text segmentation strategies to consider other types of documents. Further developments concern also the integration of the figures and arrays recognition process and their interaction with the segmentation and the structure recognition process.

References

[1] H. Baird and K. Thompson. *Reading Chess.* Proceedings of the Workshop on Computer Vision, Miami Beach, FL, Nov. 30 - Dec. 2, 1987.

[2] B. Brown. *Standards for Structured Documents.* The computer Journal, n. 6, vol. 32, 1989.

[3] R. Furuta, J. Scotfield and A. Shaw. *Document Formatting Systems. Survey, Concepts and Issues.* ACM Computing Surveys, vol. 14, no. 3, pp. 417-472, 1982.

[4] R. Ingold. *Structures de documents et lecture optique : une nouvelle approche.* Thèse de l'Ecole Polytechnique Fédérale de Lausanne, 1989.

[5] ISO 8879 Information Processing, Text and Office Systems, Standard Generalized Markup Language (SGML), 1986.

[6] ISO 8613 Information Processing, Text and Office Systems, Office Document Architecture (ODA) and Interchange Format, Parts 1,2,4-8, 1989.

[7] V. Joloboff, *Trends and Standards in Documents Representation*, In Van Vliet book, 1986.

[8] J. Kreich, A. Luhn and G. Maderlechner. *Knowledge-Based Interpretation of Scanned Business Letters.* IAPR Workshop on CV-Special Hardware and Industrial Application, Tokyo, Oct. 12-14, 1988.

[9] H. Lâasri et B. Maître. *Coopération dans un univers multi-agents basée sur le modèle du blackboard : Etudes et réalisations.* Thèse de l'Université de Nancy 1, Février 1989.

[10] G. Nagy and S. Seth. *Hierarchical Representation of Optical Scanned Documents.* In Proceedings of the 7th International Conference on Pattern-Recognition, Montreal, pages 347-349, 1984.

[11] D. Niyogi and S. N. Srihari. *A Rule-based System for Document Understanding.* Proc. of American Assoc. on AI, Philadelphia, pp. 789–793, 1986.

[12] V. Quint and I. Vatton. *An interactive system for structured document manipulation.* In text Processing and Document Manipulation, edit. J.C. Van Vliet, pp. 200-213, Cambridge University Press.

[13] S. N. Srihari, C. H. Wang, P. W. Palumbo and J. J. Hall. *Recognizing Address Blocks on Mail Pieces : Specialized Tools.* AI magazine,n. pp. 25-40, 1987.

[14] K. Y. Wong, R. G. Casey and F. M. Wahl. *Document Analysis system.* IBM Journal Research Developement 26, pp. 647-655, 1982.