Transmedia Machine

YUZURU TANAKA*, KINYA TAKAHASHI*, MOJTABA MOZAFFARI*

This paper proposes Transmedia Machine architecture for storage and processing of printed materials. Transmedia Machines integrate processing capability such as text editing in database management systems, and efficient input by image readers in electronic filing systems. All the information such as graphics, pictures, and even text may be entered into a Transmedia Machine by image readers. Thus we avoid the input bottle-neck for building large databases that type in their input data.

The users are provided with a single view of information. It is possible to enter a document by typing or reading it by image readers, or even use a mixture of these methods to enter the same document. However the users need not know the internal representation of information, and how it was entered into the system. Edited documents and original documents are managed in a unified way.

Individual characters are not recognized, instead the boundaries of character regions, lines, and paragraphs are recognized as rectangles. Then text editing is done by movement or copying of rectangular areas. The basic algorithms for extraction of these rectangles and text editing are presented in this paper.

1. Introduction

Science Citation Index provides references to more than 10 million articles published during 1955–1986, and it does not include many of the papers published in proceedings [1]. Probably no library includes all these papers, but a university library with more than one million books is not surprising. This huge amount of information suggests use of a computer to store and process them. A book that is entered into a computer, may be accessed by various readers at the same time, they can browse it, or search it for specific technical terms such as 'database' and 'knowledge base', and print pages of the book that include these terms. However the input of this increasingly huge amount of information is very hard and error-prone, if we use conventional database management systems [3, 4, 6] with their typed-in method to enter information. On the other hand, the electronic filing systems use optical disks and store information as images. All data including drawings, pictures, photographs, and even texts are stored as images in electronic filing systems. The rewritable image file devices such as optomagnetic disks, allow the users of recent electronic filing systems to input information directly into the computer by image readers (even texts need not be typed in). They make data entry more efficient and reliable than typed-in method used in database management systems. However the processing capability of electronic filing systems is more restricted than database management systems.

A database management system allows editing of retrieved text [10]. The edited text can be saved into the same database. An electronic filing system only offers restricted functions to edit retrieved information. These functions are mostly layout change functions implemented by cut-and-paste operations on images. A cut-and-paste operation cuts out an area in an original image and pastes this area in an arbitrary location of this image. The text editing functions are not provided by electronic filing systems.

Database management systems and electronic filing systems provide complementary functions. However their integration into one system is not straightforward. For example we can not input the information by electronic filing systems and process it by database management systems, since these systems use different data structures and different processing algorithms. The text editing of a retrieved document adds a new meaning to this document. For example a word in italics or bold type has a different meaning. The editing yields new meaningful information, it is a major part of our intellectual activities. Therefore, an information management system should be able to treat edited documents and original documents, in the same way. The same system should be capable of storing, editing, and retrieving both original and edited documents. Besides, it should be able to read an original document without retyping the document by the users. In building large databases, the manual input of documents through keyboards requires incredible manual labor, and it will inevitably become the most serious bottle-neck in these systems.

Transmedia Machine [8, 9] is a new document management architecture that has been studied in Hokkaido University since 1983 to satisfy the above requirements. In a Transmedia Machine, all the functions

*Hokkaido University Sapporo, 060 Japan.

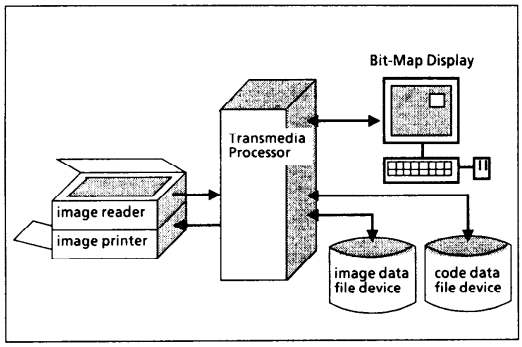


Fig. 1 Configuration of a Transmedia Machine.

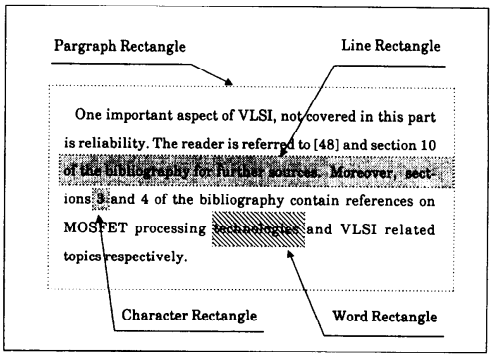


Fig. 2 Various rectangles recognized by a Transmedia Machine.

of database management systems and electronic filing systems are integrated into one system. The configuration of a Transmedia Machine is shown in Fig. 1. It consists of a transmedia processor, a bit-map display with a keyboard, code file storage devices (e.g. magnetic disk storage devices), image file storage devices (e.g. laser disk or optomagnetic disk storage devices), an image reader, and an image printer. The functions of a Transmedia Machine cover most of the office works, i.e., storage, retrieval, editing, word processing, and communication. A document may be entered into a Transmedia Machine by typing it or reading it as an image, or by a mixture of these two methods. The users are provided with a single view upon document representation, the method used to enter a document or its actual representation in the Transmedia Machine are not visible to the users of the system.

The functions of a Transmedia Machine other than editing and word processing are more or less provided by current electronic filing systems, which also provide basic layout editing functions implemented by cut-and-paste operations. Therefore text editing and word processing are the most fundamental functions a Transmedia Machine must implement. The implementation of these functions is difficult, it requires algorithms for text processing on image documents. This paper discusses these algorithms that has been implemented on Canon EZPS, a desktop publishing workstation with an image reader and an image printer. This prototype implementation shows the feasibility of Transmedia Machines.

2. Text Editing by a Transmedia Machine

A plausible method for editing of an image text is to recognize the characters in the text, and translate the text into a sequence of character codes, then the editing of the translated text is an easy task. However, pattern recognition of characters in an image text is diffucult, especially if multiple fonts are used for character, or different character sets are used (e.g. different books are written in different languages). We adopt a different approach in Transmedia Machines. Instead of character recognition, we recognize text line boundaries, character boundaries, word boundaries, and paragraph boundaries. These boundaries are recognized as rectangles with different lengths and widths as in Fig. 2. After the extraction of these rectangles, each rectangular area of an image can be easily moved to a desired location in the image using a cut-and-paste operation. With repetitive applications of cut-and-paste operations, you can manually move all the word rectangles in an image text to change the line length of the image text without changing its sentences. You can also align the left end and the right end of the text to the respective margins as if you had typed this text with a word processor. This operation can be performed by equally distributing the remaining space in each line to all gaps between the words in this line. With a font table that includes all the characters with the same font used in the object image texts, you can also rewrite the original texts. To delete a character, you can cut off the character rectangle enclosing this character from its enclosing word rectangle. Then the word rectangle including this character is cut into two parts that will be sticked together. To insert a character, you can split the enclosing word rectangle at the insertion point to make room, copy a character rectangle corresponding to this character in the font table, insert this copied rectangle into the split, and stick these three rectangles together to form a single word rectangle. Therefore, most of the text editing functions can be performed, even on image text manually with cut-and-paste and copy-and-paste operations. However, the manual execution of these procedures are too time consuming, error-prone, monotonous and tiresome even for a persistent person.

This monotonousness and elaborateness well encourage computer execution of these procedures. Transmedia Machines electronically perform the above operations in the same way as the above described manual procedures. For each original image text, the Transmedia Machine stores the size and the location of each character rectangle in this text. Image texts are stored in image file storage devices, while the sizes and

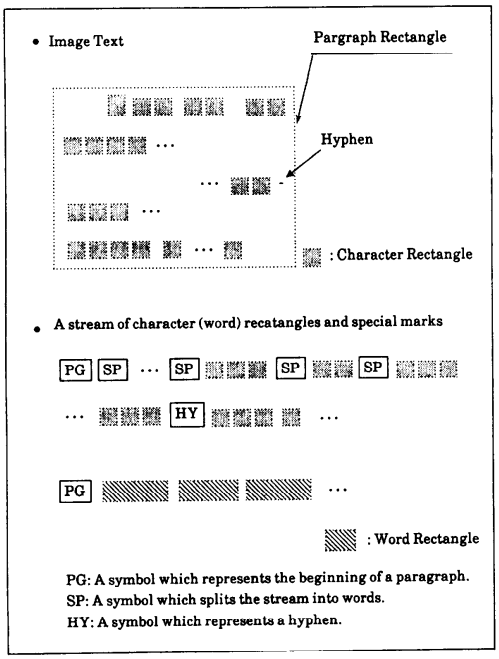


Fig. 3 Preprocessing of an image text.

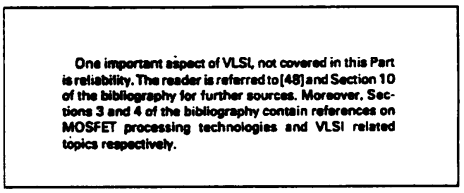


Fig. 4 An original image text.

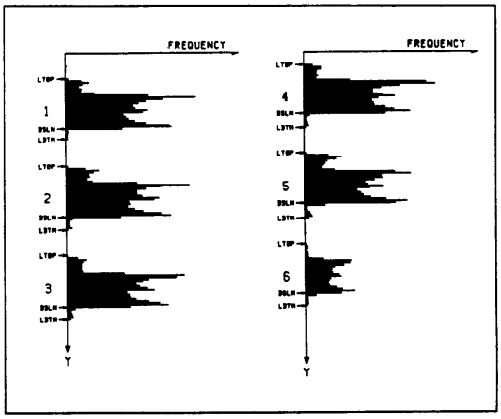


Fig. 5 Histogram of the frequency of black cells in each mesh row for the image text in Fig. 3.1.

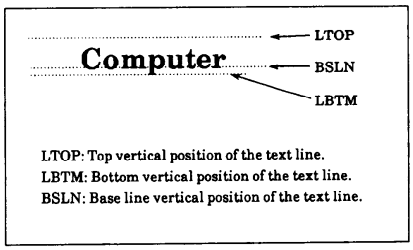


Fig. 6 Three important vertical positions.

the locations of character rectangles are stored in code file storage devices such as magnetic disk units. The latter kind of information is obtained by the preprocessing of each image text. It is represented as a stream of character rectangle characteristics with several marks inserted among them as in Fig. 3. The order of character rectangles in this stream is the same as in the original image text. The inserted marks indicate either the beginning of a paragraph, a word gap space, existence of a hyphen or an inserted character.

Transmedia Machines will provide screen editors, in which a single page of an image text will be displayed on the screen, and the character rectangle at the curso position repetitively reverses its contrast. The cursor is moved by moving the flickering from one character rectangle to another. The cursor movements are associated with the cursor keys on the keyboard or a mouse, thus the cursor moves as if we are using a text editor for code texts. From a users' point of view, the text editing of an image text by a Transmedia Machine works in the same way as editing of a code text by an ordinary word processor.

3. Basic Algorithms

The preprocessing of an image text to obtain line rectangles, character rectangles, word rectangles, and paragraph rectangles can be performed by a combination of known basic image processing techniques [2, 5, 7]. It does not require innovative algorithms. After the extraction of rectangles, each rectangular area can be easily moved to another location by a simple cut- and-paste operation, which does not require an innovative algorithm either. This section roughly describes algorithms for the steps of the preprocessing.

3.1 Line Rectangle Extraction

An image text that is input from an image reader is first transformed into a mesh of binary valued cells. Black cells are represented by '1', and white cells are represented by '0'. The mesh is assumed to cover an image text so that the mesh rows are parallel to the text lines. At the next step, line rectangles are extracted from the image text. This is performed by analyzing the histogram of the frequency of black cells for each mesh row. As an example the histogram for the image text in Fig. 4, is given in Fig. 5, where LTOP, LBTM and BSLN correspond to the positions defined in Fig. 6. The positions LTOP, and LBTM for all text lines are ob-

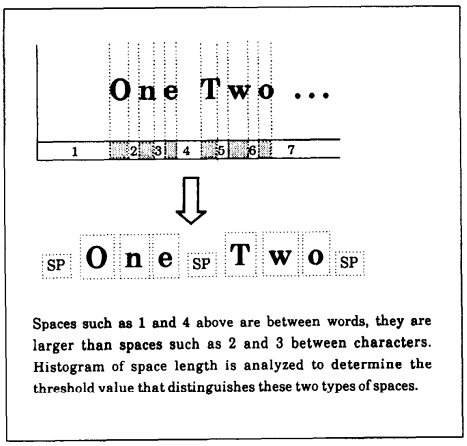


Fig. 7 Distinction of 'word gap spaces' from 'character gap spaces'.

tained by algorithm A1 of the appendix. This algorithm automatically discards phantom text lines that may sometimes result from noises between consecutive text lines. The base line position BSLN in each line rectangle is a characteristic point of the histogram, it is computed by algorithm A2 of the appendix.

3.2 Character Rectangle Extraction

After the line rectangles are extracted, the next step is to extract character rectangles in each line rectangle. The vertical projection of black cells in each line rectangle to a horizontal line yields a sequence of black and white runs of mesh cells as in Fig. 7. Usually a black run corresponds to one character. However, in proportionally spaced printing, consecutive characters sometimes loose the gap between them. Such a pair of consecutive characters may be considered as a single character. This causes no serious problems, since we need not recognize the characters. Extraction of character rectangles uses algorithm A3 of the appendix. White runs correspond to two kinds of spaces, i.e. character gap spaces and word gap spaces. They can be classified by cluster analysis.

After the extraction of character rectangles and the separation of word gap spaces from character gap spaces in an image text, the system generates a code file that is called the accompanying code file of the binary image text. This file stores a sequence of character rectangle characteristics and special marks that indicate either a new paragraph, a word gap, or a hyphen as in Fig. 7. It stores, for each character rectangle, only its location and size. Character images are not stored in the code file. We also note that paragraphs are easily extracted by detecting indentations, while a hyphen is detected by a simple template matching technique.

4. Line Length Change and Rewriting

In a Transmedia Machine, the accompanying code file for a given image text plays the same role as the text file produced by a conventional word processor. The ordinary text file stores a sequence of character codes, the order of character codes defines the order of characters in its printed image, and each character code is used to get its font image from character generator ROMs. In the accompanying code file for an image text, the order of character rectangles defines the order of characters in the printed image of this text, and the location and size of each character rectangle is used to get its font image by copying this rectangle area of the image text. A word gap space mark in an accompanying code file corresponds to a space code in an ordinary text file. The width of each character rectangle corresponds to the character width in proportionally spaced printing. It is used to determine the change of line and to align the left and right end of text to the respective margins.

Deleted characters lose their corresponding rectangles in the accompanying code file. Original image texts are not modified. In accompanying code files, each inserted character is represented by its character code preceded by an insertion mark. This pair of a mark and a character code is inserted at an insertion position in the accompanying code file. If such a character code is encountered in the printing of this file, the system gets the corresponding font image from a preprocessed font

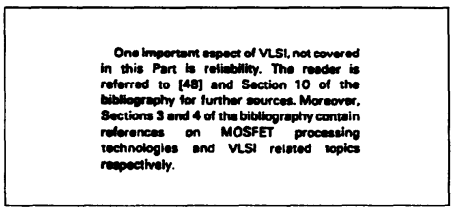


Fig. 8 Change of text line length of the image text in Fig. 4.

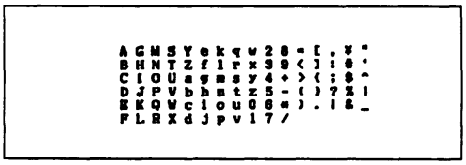


Fig. 9 A font table that is read in as an image text.

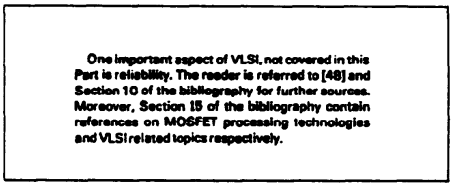


Fig. 10 An image text obtained by rewriting of the text in Fig. 3.1.

table to print this character. An accompanying code file that only consists of insertion marks and character codes, corresponds to a text that is fully typed into the Transmedia Machine.

We have not implemented a Transmedia Machine yet, but we have written a partial emulation program in C. This program performs image reading, preprocessing, extraction of various rectangles, generation of an accompanying code file, line length change, left and right alignment to the margins, text rewriting, and printing by a laser printer. An example of text line length change of the text in Fig. 4 by this program is shown in Fig. 8, where a hyphen at the end of the third line in the original image text is deleted. Another example that rewrites the original text using a font table in Fig. 9 is shown in Fig. 10. These examples show the feasibility of Transmedia Machines.

5. Conclusion

This paper has proposed Transmedia Machine architecture that integrates functions of two complementary information management systems, i.e. database management systems for coded information and electronic filing systems for image information. A Transmedia Machine can read printed documents through an image reader. Printed documents need not be manually typed into the system, thus the input bottle-neck of information management is removed. This machine provides a screen editor that allows its user to rewrite image texts as if they are typed-in texts. Rewritten texts are also managed by the system. This machine processes image texts, rewritten image text, and typed-in coded texts in a unified way. Rewritten image texts are internally represented as a mixture of read-in image texts and coded texts, but the difference of these texts or their internal representations are hidden from the users.

The algorithms used by the machine to preprocess original image texts are a combination of known image processing algorithms. The machine does not recognize each character in image texts. Instead, it recognizes the boundary of each character. The fundamental algorithms necessary to implement a screen editor of a Transmedia Machine have been given in this paper. An emulator C program based on these algorithms has shown the feasibility of Transmedia Machines.

Our current algorithms, however, still have some limitations. They cannot be directly applied to a skewed text image, i.e., an input image text whose text lines are not parallel to the horizontal line of the image reader. We will require some preprocessing to straighten the skewed text. Image reading may introduce noise and voids. These, however, do not cause a serious problems as in the case of recognizing characters. Another problem is the difficulty of separating some characters appearing in a certain order such as two 'f's followed by an 'i' in the word 'difficult'. Our current algorithms treat 'ffi' as a single character, but this is not a serious problem for the screen editor. The user may delete 'ffi' as a single character, and then add other characters in place of 'ffi'.

Acknowledgement

We wish to thank Hideyuki Torii for discussions on this paper.

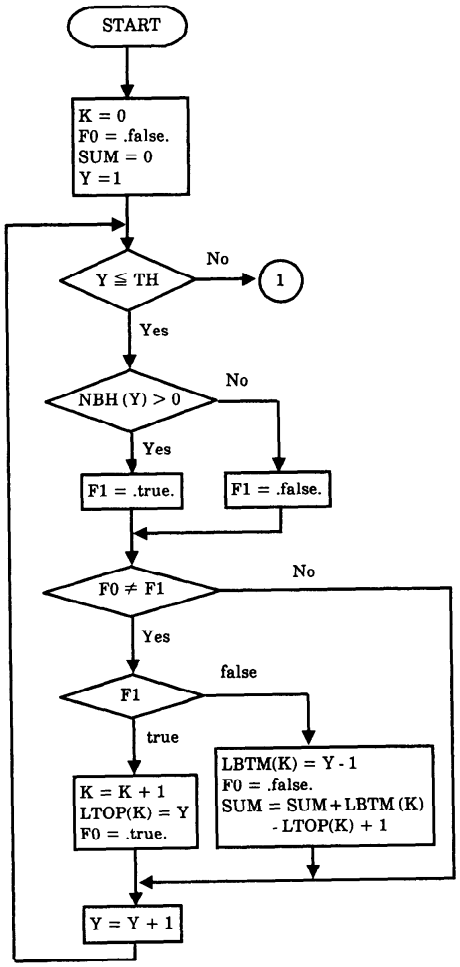
References

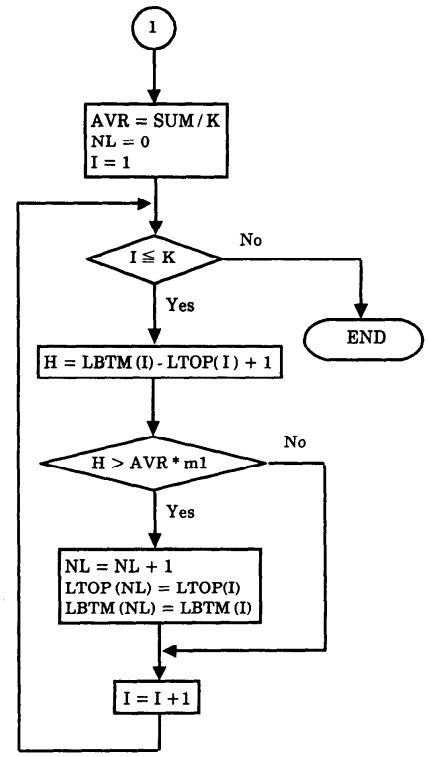
1. Science Citation Index 1986, Guide and List of Source Publications, p. 7, Institute for Scientific Information, 3501 Market Street, Philadelphia, PA 19104.
2. AKIYAMA, T. and MASUDA, I. A Segmentation Method for Document Images without the Knowledge of Document Formats, *IECE* J66-D, 1 1983, 111-118.
3. CHANG, S. K. ed., Management and Office Information Systems, *Plenum Press*, New York, 1984.
4. DATE, C. J. An Introduction to Database Systems 1, Fourth edition, Addison-Wesley Publishing Co. 1986.
5. INAGAKI, K. et al., MACSYM: A Hierarchical Parallel Image Processing System for Event-Driven Pattern Understanding of Documents, *Pattern Recogn.* 17, 1 (1984), 85-108.
6. MARTIN, J., Computer Database Organization, Englewood Cliffs, N.J., Prentice-Hall (1975).
7. NAKAMURA, O. et al., A Symbol Segmentation Algorithm from Free Format Document, *IECE* J66-D, 1 (1983), 437-444.
8. TANAKA, Y. and TAKAHASHI, K., Transmedia Machine, in NATO ASI Series, F24, Database Machines, Springer-Verlag Berlin Heidelberg (1986), 459-471.
9. TANAKA, Y. and TORII, H., Transmedia Machine and its Keyword Search over Image Texts, Proc. of RIAO 88 Conference MIT, Cambridge MA, (March 1988), 248-258.
10. ZLOOF, M. M., Office-By-Example: A Business Language that Unifies Data and Word Processing and Electronic Mail, *IBM Syst. J.*, 21, 3 (1982).

(Received February 16, 1988; revised January 5, 1989)

Appendix

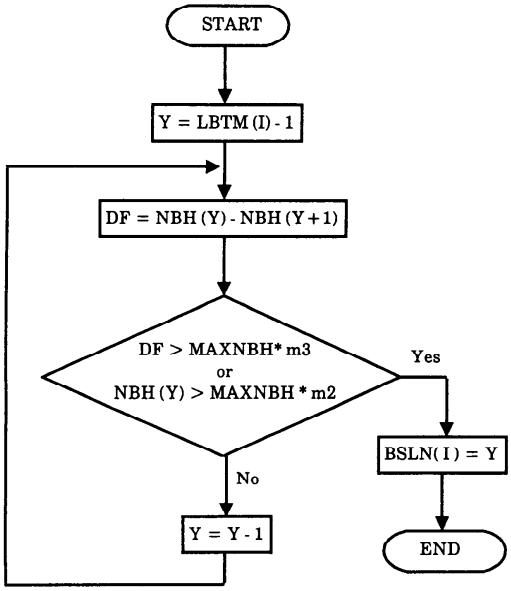
A1. Text Line Extraction Algorithm





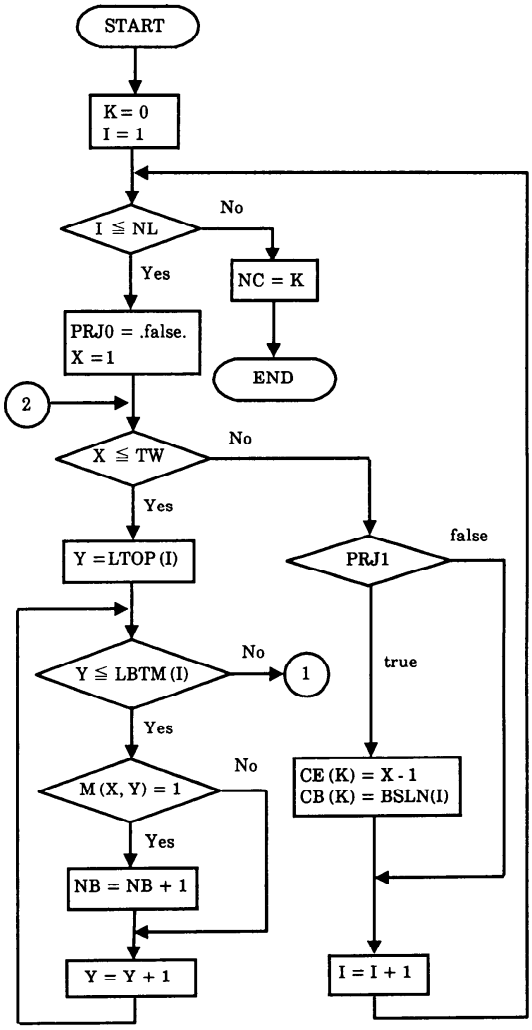
X: A Horizontal position in the text.
Y: A vertical position in the text.
TW: The width of the text.
TH: The height of the text.
NBH(Y): The number of vertical black mesh cells in the horizontal mesh at vertical positon Y.
LTOP(I): The top vertical position in line I of the text.
LBTM(I): The bottom vertical position in line I of the text.
NL: The number of lines of the text.
AVR: The Average height of all lines of the text.
m1: Heuristically defined parameter (We set m1=0.5), it is used to remove phantom text lines.

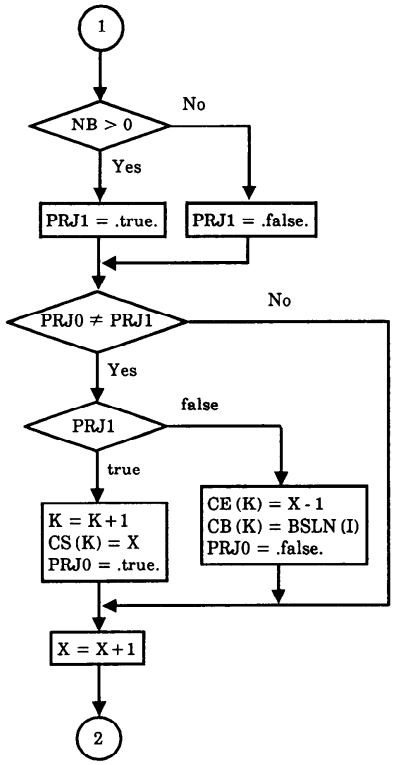
A2. Base Line Extraction Algorithm



BSLN(I): Base Line vertical position of the text line I.
MAXBNH: Maximum number of black mesh cells in a line, for all mesh lines in the text.
m2, m3: Heuristically defined parameters (We set m2=0.3=m3).

A3. Character Extraction Algorithm





M(x, y): Binary Value of the mesh cell at location (X, Y), a black cell is represented by 1.
NB: Number of black mesh cells, x-coordinate—X,y-coordinate—from LTOP to LBTM.
NC: Number of all characters in the image text.
CS(K): Horizontal start position of the character K.
CE(K): Horizontal end position of the character K.
CB(K): Base line vertical position of the character K.