Braille-Embedded Tactile Graphics Editor with Infty System

Toshihiro Kanahori[1], Masayuki Naka[1], and Masakazu Suzuki[2]

[1] Research and Support Center on Higher Education
for the Hearing and Visually Impaired, Tsukuba University of Technology,
4-12-7 Kasuga, Tsukuba, Ibaraki 305-8521, Japan
{kanahori,naka}@k.tsukuba-tech.ac.jp
[2] Faculty of Mathematics, Kyushu University,
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan
suzuki@math.kyushu-u.ac.jp

Abstract. We are developing a graphics editor to easily draw Braille-embedded tactile graphics for scientific documents including mathematical expressions with Infty specialized system. Each graphics embedded in a document is cut out and characters on the graphics are recognized by InftyReader. The recognized graphics can be edited with this graphics editor, which is based on scalable vector graphics. Since, in the editor, ordinary texts and mathematical expressions on the recognized graphics are presented by InftyEditor, users can easily edit them with its input interface. After completing the edit process, all texts and mathematical expressions are automatically translated into Braille by our original Braille translator, InftyBraille. Then, the edited graphics is output as an EMF file. Embossing out the EMF file, users even who do not know Braille codes can get a Braille-embedded tactile graphics.

1 Introduction

We have been developing an integrated suite, named "Infty", specialized for scientific documents including mathematical expressions. Infty mainly consists of a document reader "InftyReader" and an authoring tool "InftyEditor" [1, 2, 3]. InftyReader recognizes scientific documents including mathematical expressions. Its recognition results can be output as files in various accessible formats such as MathML, LaTeX and so on. They can be directly imported into InftyEditor and can be edited with it. InftyEditor has two intuitive input interfaces for mathematical expressions; a handwriting interface, and a keyboard-based one. They provide seamless environment for input between ordinary text parts and mathematical expressions. Edited documents with InftyEditor can be output as in the same various formats as InftyReader. In addition they can be converted to Japanese Braille codes. "ChattyInfty" is an accessible math-document editor derived from InftyEditor, which provides aloud-reading output (Japanese/English) interfaces for not only ordinary text parts but also mathematical expressions [1]. Visually impaired people even who do not know Braille notation for mathematical expressions can read mathematical documents written by Infty, and can

K. Miesenberger et al. (Eds.): ICCHP 2008, LNCS 5105, pp. 919–925, 2008.

write mathematical documents in those various formats with its speech output. Although Braille codes that Infty can output is only Japanese in this moment, Infty documents can be converted into UEBC, American Braille (Nemeth codes), British Braille, French Braille, German Braille and so on by UMCL (Universal Maths Conversion Library [4]), DBT (Duxbury Braille Translator [5]) or other Braille translation system via LaTeX or MathML.

As a next phase of our research and development for authoring accessible scientific materials for visually impaired people, we are developing a Braille-embedded tactile graphics editor for scientific graphics including mathematical expressions, by applying Infty system. The process of producing tactile graphics advances in the following steps. First, an extended version of InftyReader detecting figure area of page image, cuts out texts and mathematical expressions embedded in the graphics and outputs the recognition results in an integrated data format of graphics editor including image data and characters. The recognition results of characters and mathematical expressions can be embedded in the graphics in either InftyEditor's data format or directly in Braille codes by the choice of users. If necessary, the recognized graphics can be edited by our graphics editor to correct recognition errors and adjust the sizes of characters and figures. Then, the characters and mathematical expressions are translated into Braille code. In this paper, we introduce our new graphics editor and show process for producing those graphics by using it.

2 Related Work

There are many related works about making tactile graphics (e.g. [6,7,8]). Some of them use learning algorithms to detect and recognize characters in a figure. Therefore they need a lot of learning data before their practical use and their systems are often-complicated. Most of them cannot recognize mathematical expressions.

Developing our software, we focus in easy operation for users. It based on WYSWYG, so users can edit tactile graphics seeing what they get by OCR. Utilizing InftyReader, it can recognize also mathematical expressions. Other several systems are using InftyReader to recognize them. In our system, it is easy to correct and edit text parts including mathematical expressions, because of InftyEditor components. IVEO [7] also provides WYSWYG editor (IVEO Creator/IVEO Creator Pro [9]) for making tactile graphics. Our system is simpler than the editor, so information which can be embedded in tactile graphics is less than IVEO can (hidden text, links, etc.).

This system does not recognize non-text parts (lines, curves, etc.), so only images of characters are replaced with recognized results. For non-text parts, the original image is used as a background image of the texts. Hence, it is not necessary to correct graphic recognition results. However, edited graphics of the non-text parts are not necessarily smooth.

Our recognition method for characters in a figure, which is based on our character recognition method of InftyReader, does not use learning algorithms (in

Section 4). Any preparations are not required to recognize figures. The method cannot recognize rotated characters at this moment, but the several systems can recognize them [6].

3 Outline of the Braille-Embedded Tactile Graphics Editor

Our graphics editor is a system based on scalable vector graphics. Its functions are very simple as follows:

- Putting 4 types of objects:
  1. drawing primitive graphic objects; line segments, curves, polygonal lines, polygons, rectangles, circles, ellipses, sector forms and arcs,
  2. putting text objects, which can contain both of ordinary texts and mathematical expressions implemented with InftyEditor components.
  3. pasting picture files (TIFF or BMP) as picture objects,
  4. drawing function graphs (ex. $y = \sin x$) as graph objects,

  where styles and colors of drawing lines and painting colors of those objects can be chosen with color palettes.
- Importing picture files (TIFF or BMP, must be black and white, binary images scanned in either 600dpi or 400dpi) of diagrams to be recognized by InftyReader (an outline of the recognition process in Section 4).
- Saving edited graphics into original XML files (.ipl).
- Outputting edited graphics into EMF (Enhanced Metafile).
- Directly printing out graphics to a printer.
- Primitive edit functions; cutting, copying and pasting objects, and undoing and redoing operations.
- Selecting objects.
- Grouping objects.
- Relocating objects.
- Scaling up and down sizes of objects.
- Editing picture objects; painting, erasing, drawing lines on the picture.
- Setting paper parameters; a size, an orientation (portrait or landscape) and color.
- Setting font parameters; a font style and a size.
- Setting grid parameters; drawing on a paper or not, distance between lines, and color.
- Zooming up and out.

In order to write ordinary texts and mathematical expressions on graphics, InftyEditor component is incorporated in the graphics editor. Graphic format of the editor has a layer structure, so that it is possible to draw primitive graphics on the first layer and to put texts over a pasted picture (on the second layer). With InftyEditor components, texts written on the second layer can be easily translated into Braille codes as was mentioned in the previous section by combining UMCL or DBT. After the translation into Braille codes completes, each of the

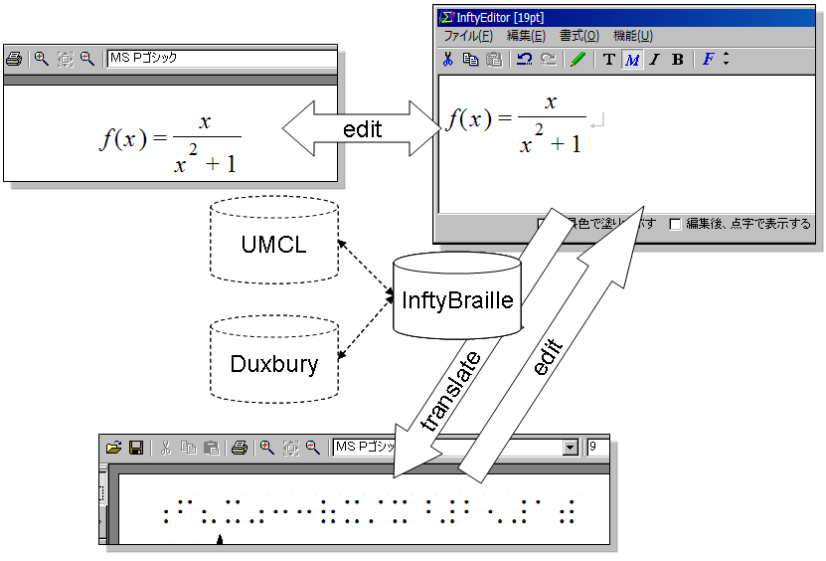

Fig. 1. After Braille translation, the Braille codes for $f(x) = \frac{x}{x^2+1}$ can be edited in their printed characters

original texts is replaced with Braille characters. The Braille codes can be edited with InftyEditor component in printed characters before converted into Braille (fig. 1). Users do not need to edit Braille codes directly.

We design the graphics editor to emboss out an edited graphics utilizing *ViewPlus Tiger Embosser* [9] via EMF at this moment. By this graphics editor, teachers even who does not know Braille notation can make original tactile graphics including mathematical expressions for visually impaired students using various Braille notations.

4 Recognition Method for Characters in Figures

Our recognition method for characters in figures is done by taking the following steps:

1. Before area segmentation, character recognition is done in order to estimate global character features, and the features are used to segment areas. Then, characters which are touching line segments are separated from them.
2. According to estimated sizes of gaps, areas are segmented in Manhattan layout.
3. An area is determined as a figure area if one of these 2 conditions is satisfied;
   (a) the number of small connected components which are not characters included in the area is greater than a certain threshold,
   (b) a big connected component included in the area is neither an integral symbol, a root symbol, a parenthesis nor a horizontal line.

4. A connected component is determined as a character if both of these 2 conditions are satisfied;
   (a) its size is close to the estimated character size,
   (b) the score of character recognition of the component is greater than a certain threshold.
5. Standard character sizes (sizes of baseline characters [10]) are estimated by using the determined characters .
6. A connected component is determined as a character if both of these 2 conditions are satisfied;
   (a) the normalized size of a recognized character of the component is very close to the estimated standard character size,
   (b) the score of the character recognition is greater than a certain lower threshold.
7. A connected component is determined as a subscript character if these 3 conditions are all satisfied;
   (a) it is on a subscript position of baseline characters,
   (b) the sizes of its recognized character are also close to subscript character sizes,
   (c) the recognized character is one of alphanumeric characters or Greek characters.
8. A connected component is deleted if both of these 2 conditions are satisfied;
   (a) its recognition result is an operator character,
   (b) no characters (alphanumeric or Greek) are beside the component.
9. A small component is deleted if both of these 2 conditions are satisfied;
   (a) its recognition result is a point character (comma, prime, period, etc.),
   (b) there is no character whose character type and position are proper to the point character.

After this area segmentation, character recognition is performed again for each text area.

5 Outline of Process for Producing Tactile Graphics from Printed Graphics in a Document

A printed material including texts, mathematical expressions and graphics is treated in the following steps (fig. 2).

1. Scanned images of the document are recognized by the extended version of InftyReader. InftyReader automatically detects and cut out graphics from the documents. Embedded texts in graphics are recognized automatically. For each detected graphics, InftyReader outputs both of recognition results of embedded texts and an image file of the graphics. It outputs the results together into our graphics editor data format. If the document has complicated layouts, InftyReader sometimes fails extracting graphics. At that time the graphics are supposed to be cut out by hand as image files after the recognition process.

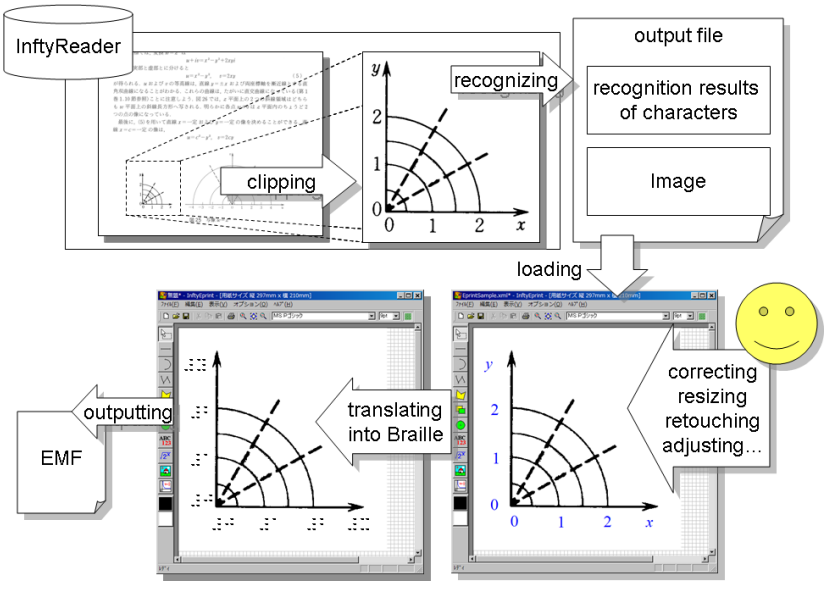

Fig. 2. Outline of making tactile graphics. Clipping embedded graphics and recognizing them are done by InftyReader. Editing each of them is done with the graphics editor.

2. The texts and mathematical expressions in the output graphics are corrected on InftyEditor component of the graphics editor, and then converted into Braille codes by the graphics editor.
3. Each extracted image data is pasted on the first layer of graphics editor. Paint tool dialog is launched by double clicking on the image. On the dialog, the image is corrected by deleting noises or smoothing lines, etc.
4. For each text on the image, the corresponding Braille string is pasted at the appropriate position of the second layer. Its size and position can be adjusted easily by hand. On the other hand, the text image part on the first layer is deleted by using Paint tool dialog also by hand.
5. Sizes of the image and the Braille texts are adjusted so as to be fit for output size as a tactile graphics. Positions of the Braille texts are fixed by hand.
6. The edited graphics is saved as EMF to be embossed out. The text part of the document is output in LaTeX or MathML for Braille translation.

6 Conclusion

Applying Infty system, we are developing a Braille-embedded tactile graphics editor. On this editor, mathematical expressions can be put on a graphics and translated into Braille codes by combining with UMCL or DBT, which are powerful Braille translators. At this moment, this editor uses Tiger Embosser in order to emboss out edited tactile graphics via EMF. Using this system, teachers can easily make tactile graphics for visually impaired students. Because not

specific Braille notation but InftyEditor components are used for putting texts including mathematical expressions, it is possible to translate texts on a graphics into various Braille codes via LaTeX or MathML.

At this moment, InftyReader can not recognize graphic components in figures (lines, curves, etc.) and rotated characters. Improve for those points and evaluation of the recognition method for characters in figures are parts of our future works.

References

1. Suzuki, M., Kanahori, T., Ohtake, N., Yamaguchi, K.: An Integrated OCR Software for mathematical Documents and Its Output with Accessibility. In: Miesenberger, K., Klaus, J., Zagler, W., Burger, D. (eds.) ICCHP 2004. LNCS, vol. 3118, pp. 648–655. Springer, Heidelberg (2004)
2. Suzuki, M., Tamari, F., Fukuda, R., Uchida, S., Kanahori, T.: Infty - an integrated OCR system for mathematical documents. In: Grenoble, Vanoirbeek, C., Roisin, C., Munson, E. (eds.) Proceedings of ACM Symposium on Document Engineering 2003, pp. 95–104 (2003)
3. Kanahori, T., Fujimoto, M., Suzuki, M.: Authoring Tool for Mathematical Documents - Infty -. In: 3rd International Conference MKM 2004, Bialowieja, Poland (September 2004), `http://www.activemath.org/~paul/MathUI`
4. Universal Maths Conversion Library, `http://inova.snv.jussieu.fr/umcl-demo/`
5. Duxbury Systems, `http://www.duxburysystems.com/`
6. Jayant, C., Renzelmann, M., Wen, D., Krisnandi, S., Ladner, R.E., Comden, D.: Automated Tactile Graphics Translation: In the Field. In: Proceedings of the 9th international ACM SIGACCESS conference on Computers and Accessibility, Tempe, AZ, October 2007, pp. 75–82. Association for Computing Machinery (ACM) (2007)
7. Gardner, J.A., Bulatov, V.: Scientific Diagrams Made Easy with IVEO^{TM}. In: Miesenberger, K., Klaus, J., Zagler, W., Karshmer, A.I. (eds.) ICCHP 2006. LNCS, vol. 4061, pp. 1243–1250. Springer, Heidelberg (2006)
8. Crombie, D., Lenoir, R., McKenzie, N., Ioannidis, G.: The Bigger Picture: Automated Production Tools for Tactile Graphics. In: Miesenberger, K., Klaus, J., Zagler, W., Burger, D. (eds.) ICCHP 2004. LNCS, vol. 3118, pp. 713–720. Springer, Heidelberg (2004)
9. ViewPlus Technologies, `http://www.viewplus.com/`
10. Eto, Y., Suzuki, M.: Mathematical formula recognition using virtual link network. In: Proc. ICDAR, pp. 762–767 (2001)