PAPER *Special Issue on Human Communication*

Comic Image Decomposition for Reading Comics on Cellular Phones

Masashi YAMADA[†a)], **Rahmat BUDIARTO**[††b)], ***Nonmembers***, **Mamoru ENDO**[†c)], ***and*** **Shinya MIYAZAKI**[†d)], ***Members***

SUMMARY This paper presents a system for reading comics on cellular phones. It is necessary for comic images to be divided into frames and the contents such as speech text to be displayed at a comfortable reading size, since it is difficult to display high-resolution images in a low resolution cellular phone environment. We have developed a scheme how to decompose comic images into constituent elements frames, speech text and drawings. We implemented a system on the internet for a cellular phone company in our country, that provides downloadable comic data and a program for reading.

key words: *cellular phone, mobile application, image decomposition, comic*

1. Introduction

Comics are one of popular reading. People traditionally read comic books in Japan. Nowadays, there are many e-comics services available on the web. It would be even more convenient if we could read comics on mobile computers such as PDA or cellular phones, but almost all of them can only be accessed through desktop or laptop computers. We therefore tried to develop a system for reading comics on cellular phones. The problem which arose in the development of the system was how to display high-resolution images in a low-resolution cellular phone environment.

Comics are originally printed in high quality image on B6, A5 or B5 paper, but cellular phone screens' resolution is about 130×150 pixels. Therefore, it is difficult to reproduce high quality comic images on them.

We transmit comic image data from web servers to cellular phones through mobile phone links, which are a narrow bandwidth and the cost is expensive at present. Moreover, the memory capacity of cellular phones is limited. Therefore, cellular phones cannot handle heavy image data and needs to be as small as possible.

To overcome these problems, we propose a scheme how to decompose comic images into constituent elements, such as frames, speech text and drawing. We implemented a system for a company's cellular phone users who can access the web, that provides downloadable comic data and a program for reading.

1.1 Related Work

Recently, some e-comics services for PC or PDA are provided[1]–[3]. RoidTime2[1] enables broadcast for PDA by dividing page images by frames or smaller areas than frames. A part of a frame is magnified to make the speech content readable. The frame cutting process is simple, but it must be done manually and demands special skills for effective cutting. In reality, the number of contents is far fewer than that of 10daysbook[3].

Kurlander et al. [4] developed Comic Chat. This system generates a comic automatically from chat data. In order to generate a comic, the system selects various character patterns prepared in advance. Image synthesize techniques are only needed techniques.

There is some work about comic image processing. Yamada et al. [5] proposed a personnel recognition method. Syeda-Mahmood et al. [6] proposed a method to detect texture patterns in comic images. But, these techniques are difficult to be applied in general cases. Our approach needn't deal with object recognition.

Mobile phones are small in size, and have a relatively small memory and processing capacity. The wireless bandwidth is also rather limited compared to wireline networks. Ojanen et al. [7] proposed a compression method in Wireless Application Protocol environments, but it is only effective in a specific application. Chen et al. [8] explored the issues of compression methods to perform efficient data transfers from server to client. Instead of using a fixed method, they chose a combination of compression methods, that use statistical and semantic information of query results. Their techniques achieved significant performance improvements over standard compression tools like WinZip.

Manuscript received October 04, 2003.
Manuscript revised December 12, 2003.
Final manuscript received February 1, 2004.

[†]The authors are with the School of Computer and Cognitive Sciences, Chukyo Univ. Kaizu-cho, Tokodachi, Toyota, 101 Japan.
[††]The author is with the School of Computer Sciences, Universiti Sains Malaysia, 11800 Minden, Pulau Pinang, Malaysia.
a) E-mail: myamada@sccs.chukyo-u.ac.jp
b) E-mail: rahmat@cs.usm.my
c) E-mail: endoh@sccs.chukyo-u.ac.jp
d) E-mail: miyazaki@sccs.chukyo-u.ac.jp

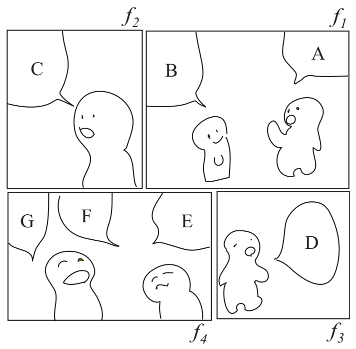


Fig. 1 Structure of a comic page

2. Overview

2.1 Comic Structure

Comics consist of pages. Each page has frames. Each frame has *speech text* and a *drawn image*. This structure is represented as follows.

A comic is a set of pages P . A page $p \in P$ is a set of frames F. The elements of F have an order and can be made into a sequence:

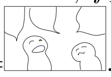
$$f_1 < f_2 < \cdots < f_n,$$

where $f_i \in F$. A frame $f \in F$ has some groups of speech text. A group of speech text usually corresponds to a speech bubble in the frame. The elements of speech text bubbles S have an order and can be made into a sequence:

$$s_1 < s_2 < \cdots < s_m,$$

where $s_i \in S$. A frame $f \in F$ also has a drawn image where speech text was removed from the original frame image. Consequently, a frame is represented as follow:

$$(d, S),$$

where d is a drawn image.
Example: Figure 1 illustrates a page of a comic. This page has 4 frames $f_1 < f_2 < f_3 < f_4$ in this order. Japanese comics are presented in this order from right to left. Each frame has groups of speech text. For example, frame f_4 has three groups of speech text, E, F and G in this order. Also, f_4 is represented as $(d_4, \{\mathrm{E}, \mathrm{F}, \mathrm{G}\})$, where $d_4 =$ .

2.2 Outline of the Reading Comic System

In this section, a system for reading comics on cellular phones is outlined. The scheme of the system is shown in Fig. 2. The input is an image of comic page. It is obtained from a scanner as a binary image with about 400dpi resolution, which is enough. First, the system detects the frames and establishes a sequence of them by analyzing their layout. Next, the system extracts speech text from each frame image, then speech text is made into groups, and establishes a sequence by analyzing their layout. Speech text is recognized by using OCR. The remaining drawn image is compressed and formatted for cellular phone display.

We developed a customized java program to handle frame data, that acts as an engine and enables cellular phones to reproduce frames from the data.

3. Frame Sorting

In this section, a frame sorting method is described. There are many layout analysis systems, which are limited. Their target is specific domain documents such as office forms, newspapers and so on. Unfortunately, there is no layout analysis system for comic frames. The shapes of comic frames are not just rectangular but are polygonal in general. A new layout analysis method had to be formed to make frames into a sequence. We assume the followings:

1. Frames are convex polygons.
2. Frames have closed outlines.
3. A comic page sequence starts from right to left.

Reference 1 and 2 are restrictions in our system. Reference 3 is supposed for easy explanation in the following sections.

3.1 Frame Sorting Rules

First, we define three rules, R1, R2, and R3, for sorting two neighboring frames (see Fig. 3). In general, comic pages are read from the upper right frame. These three rules decide which one of the two neighboring frames is relatively the upper right and then give higher priority to that frame. We created a line between the two neighboring frames and called it the *dividing line.* Let the direction of the dividing line have a positive y value or be the same of x-axis (i.e. the direction vector (x, y) satisfies $y > 0$ or (1,0)). An example of the dividing line is shown in Fig. 3. The dividing line is parallel to the neighboring border lines of the two neighboring frames. The frame on the right side of the dividing line is called the *right side frame.* The other is called the *left side frame.* The first rule uses angle α , to sort the right and left side frames. α is the angle formed where the x-axis and the dividing line cross each other. The rule is defined as follow:

(R1) $90° \leq \alpha < 180° \Rightarrow rf < lf.$

where rf and lf corresponds to right and left side frames respectively.
Example: Figure 4(a) shows an example sorting two neighboring frames. From rule R1, $rf < lf$ is obtained.

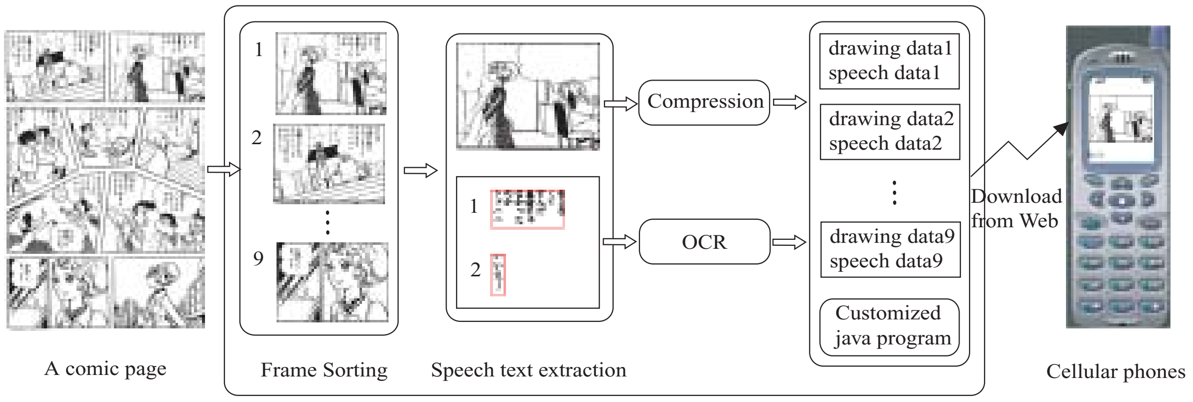


Fig. 2 System overview for reading comics on cellular phones

In case $0° \leq \alpha < 90°$ it is difficult to judge which one of the two neighboring frames is relatively the upper right. R2 and R3 are rules for only two neighboring frames that satisfy $0° \leq \alpha < 90°$.

Both R2 and R3 use the *base line* defined as follows: the base line is a common line which includes the border line of both neighboring frames respectively (see Fig. 3). Suppose the frames are convex polygons, there are two common lines at most. To determine the base line, first, the border line of the frames are given a direction counterclockwise and the common lines are given the same direction as the border line. After that, if a common line crosses the dividing line of the two neighboring frames from the right side to the left side, let the common line be the base line. R2 and R3 use two angles, α and β. β is the angle formed where x-axis and the base line cross each other. R2 and R3 give priority to one of two neighboring frames as follows:

(R2) $0° \leq \alpha < 90° \wedge \alpha \leq \beta \leq 135° \Rightarrow lf < rf$.

(R3) $0° \leq \alpha < 90° \wedge 135° < \beta < \alpha + 180° \Rightarrow rf < lf$.

R2 gives higher priority to the upper frame in two neighboring frames. R3 gives higher priority to the right frame in two neighboring frames.

Example: The order of the two frames in Fig. 4(b) is $lf < rf$ from R2. The order of the two frames in Fig. 4(c) is $rf < lf$ from R3.

Next, we defined a constraint rule for sorting three frames that form a 'T' pattern shown in Fig. 5. Frames B and C have a common base line. Frame A is neighboring both frames B and C. We call this case *T- type neighboring* and represent it as $t(\mathrm{A,B,C})$ or $t(\mathrm{A,C,B})$. The constraint rule is defined as follow:

(R4) $t(f_1,f_2,f_3) \Rightarrow (f_1 < f_2 \wedge f_1 < f_3) \vee (f_1 > f_2 \wedge f_1 > f_3)$.

Example: Figure 6 shows an example of a comic page, which has five frames. We obtained the order of the frames as follows: From rule R1, C < E, C < D, B < D, E < D were obtained. From rule R2, A < C was obtained. From rule R3, C < B was obtained. There are three T-type neighboring cases which are indicated by circles. From the constraint rule R4, we have (A < C $\wedge$ A < B) or (A > C $\wedge$ A > B) for $t(\mathrm{A,C,B})$. A < C has already held, then A < B.

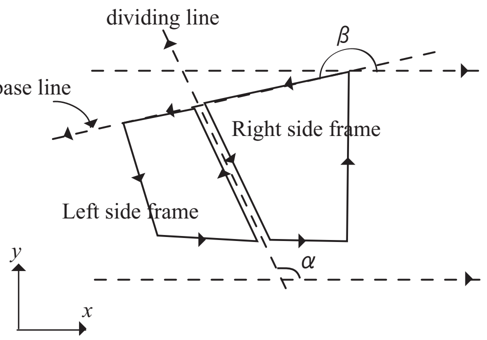


Fig. 3 The base line and the dividing line

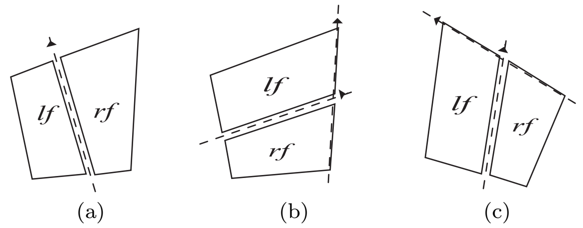


Fig. 4 Examples of sorting two neighboring frames

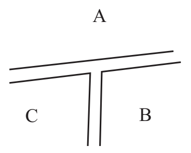


Fig. 5 T-type neighboring $t(\mathrm{A,B,C})$

3.2 A Frame Sequence Search

Next, a depth first search is used. This search finds a unique sequence of frames that satisfies frame sorting rules R1, R2, R3 and R4. This search regards each

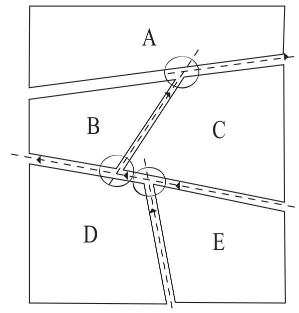


Fig. 6 An example of T-type neighboring and dividing lines

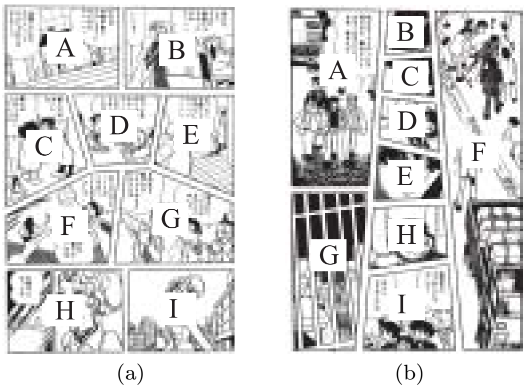


Fig. 7 Labeling frames for sorting

frame as a node and finds a path by following the neighboring frames. If there are plural neighboring frames, the search selects a frame on the basis of priority order among the neighboring frames.

We explain the search process of the following example in detail in the Appendix .

Example: Figure 7 shows two comic pages. The proposed method gives the frame sequence B < A < E < D < C < G < F < I < H in Fig. 7(a) and the frame sequence F < B < C < D < E < H < I < A < G in Fig. 7(b).

4. Speech Text Sorting

In this section, we propose a speech text sorting method. Before the sorting process, speech text is detected in frames by the following processes. First, square areas are detected by scanning a frame. The areas which contain black pixels are not connected to any black pixels outside of the square. In this process, the size of square is adjusted to the size of the letters (see Fig. 8(a)). Then, the detected areas are gathered on the basis of distance between areas. In this process, some isolated areas are disregarded. We assume rectangles which enclose all of the gathered areas and consider the rectangle areas as speech text bubbles (see Fig. 8(b)).

If a frame has more than one speech text bubble, they have to be sorted. Here, a sorting rule is defined for two speech text bubbles. This rule uses the rectangles of speech text bubbles, obtained in the above process. The top right corner position of rectangles are used for sorting (see Fig. 9). s and s' represent speech text bubbles, $P(s)$ and $P(s')$ denote the top right corner positions of s and s' respectively. $dist(p)$ denotes the distance between position p and the top right corner of the frame. The sorting rule for s and s' is defined as follows:

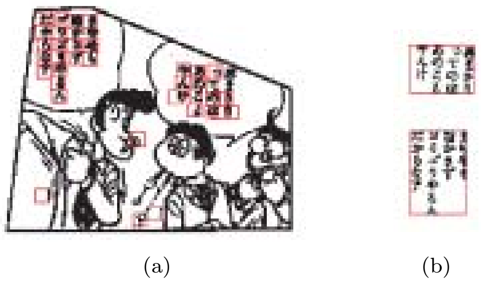


Fig. 8 Detected letters and groups of speech text

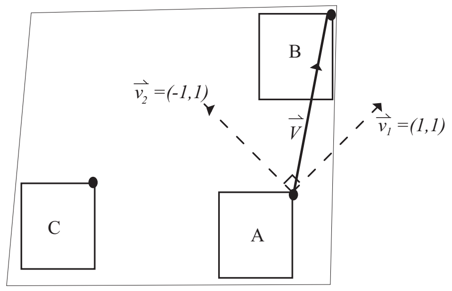


Fig. 9 Sorting speech text

(R5) if $\vec{V}$ is between $\vec{v_1}$ and $\vec{v_2}$, then $s' < s$,
else if $dist(P(s')) < dist(P(s))$, then $s' < s$,
else $s < s'$.

Here $\vec{V} = P(s') - P(s)$, $\vec{v_1} = (1, 1)$ and $\vec{v_2} = (-1, 1)$. The condition of $\vec{V}$ being between $\vec{v_1}$ and $\vec{v_2}$ is that both $\vec{V} \cdot \vec{v_1} \geq 0$ and $\vec{V} \cdot \vec{v_2} \geq 0$ are satisfied.

The order of speech text bubbles is determined by rule R5.

Example: In Fig. 9, speech text bubbles have orders B < A and A < C. Figure 8(a) has two speech text bubbles. The order after sorting is shown in Fig. 8(b).

5. Data Storing

5.1 Drawn Image Data

We investigated in depth representing data of drawn images. We found many restrictions in present cellular phones, such as screen dimension capacity, screen resolution capacity, memory capacity, processing capacity, downloadable data capacity and so on. In order to enable comics to be read on the cellular phones comfortablely, these restrictions need to be overcome.

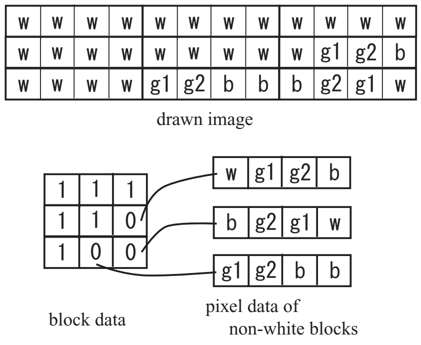


Fig. 10 An example of drawn image data

First, drawn images are obtained from frame images by removing speech text. Then, resolution is reduced to fit cellular phone screen. During this process, drawn images are converted from binary images to gray-scale images by bi-cubic convolution. Each pixel of the gray-scale image has a density value of 8bit. In the next step, the gray-scale images are compressed.

Many kinds of compression methods for digital images have been developed for general use. To overcome the limitation of downloadable data size on cellular phones, we developed an original compression method. It is based on limitation of gray-scale level and grouping pixels in white color areas.

First, a whole image area is divided into blocks which are $M \times N$ pixel rectangle areas. If all pixels in the block are white (density value for each pixel is 255), it is called *white block* and compression is applied. Otherwise, the pixel values in the block are represented by n gray-scale level. This method made a good compression performance while saving the processing power for image reproduction.

Example: Figure 10 shows an example of drawn image data. Here, the level of gray-scale is 4 and the size of the blocks is 4×1 pixels. White blocks are stored as 1 in the block data and non-white blocks are stored as 0. Pixels values for non-white blocks have to be stored additionally.

In our experiments, we adopted 8×1 pixel block size and 4 level gray-scale for downloading comic data page by page.

5.2 Speech Text Data

In Sect. 4, we described a method for detecting speech text. The detected speech text is extracted from frames as a partial image and then it is translated into ASCII data by OCR (Media Drive e.Typist entry R2). Speech text of frames is recorded in a text file (see Fig. 11). In this file, each row corresponds to a speech text bubble. Furthermore, the position (coordinate values) of speech text bubbles is put at the head of each row.

```
(110,10) まったく しょうの…
(10,20) おかげで かれと…
```

Fig. 11 An example of speech text data file

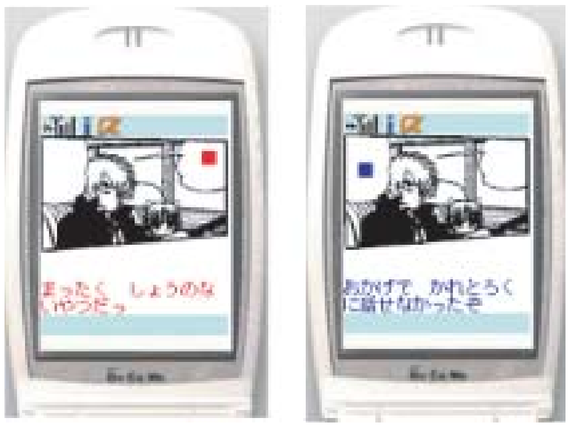

まったく しょうのな いやつだっ
おかげで かれとろく に話せなかったぞ

Fig. 12 Examples of reproduced comic on a cellular phone

5.3 Implementation

We implemented a cellular phone program in Java. We call it *comic engine*. This engine reproduces drawn images and speech text for cellular phone screens from downloaded data. The engine has the following functions:

User interface: Users can switch frames by pressing a button. If there are plural speech text areas in a frame, users can switch them by pressing another button. Thus, all the sequence in each comic page is navigated through.

Drawn image reproduction: The engine reproduces the drawn image of the current frame from drawn image data. This process is done every time the frame is redrawn in order to reduce the occupied memory.

Speech text reproduction: The engine reproduces speech text from the speech text data. The engine uses cellular phone fonts in order to make speech text easily read. To indicate the corresponding bubble, the engine puts a small square mark the same color as the speech text inside the bubble. We should not disregard the bubble because the bubble is one of the artistic effects, though white bubble space wastes display area. Figure 12 shows examples of execution on the emulator.

We have implemented our real system and generated downloadable comic data of a full comic story consisting of 23 pages[9]("Black Jack", Vol.1, No.6). Figure 13 shows a screen shot in our experiments.

6. Evaluation

First, we describe the results of frame sorting. In this experiment, we used 175 pages of 4 real comic stories

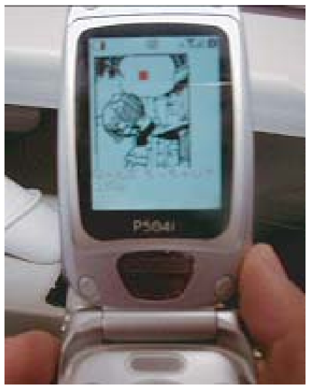


Fig. 13 A snapshot of the experiments

written by different artists. Our system was not successful on 10 pages, because those pages included some frames which boundaries were not obvious. The remaining 165 pages was processed to be decomposed into frames automatically. The frame sequence success rate was 94%.

Next, we describe the results of generating speech text data. In this experiment, we used 123 frames of a real comic. First, speech text detection was executed. The speech text of 115 frames was detected correctly. The success rate was 93%.

The cause of failure on 8 frames was as follows: A part of drawn image was detected as speech text. Two speech text were detected as one group of speech text. The OCR recognition rate of extracted speech text was 81%.

Next, we evaluate the performance of our data format for drawn images. Figure 14 shows two examples in the upper row and lower row respectively. The left side image in each row is a pre-compressed image to fit cellular phone screens. The right side image in each row is a reproduced image by a cellular phone. The sizes of the pre-compressed images are 12Kbyte and 15Kbyte respectively, whereas the sizes of the compressed data are 2Kbyte and 3Kbyte respectively.

Downloadable data size was limited within 30Kbyte in implementation. The total sizes of data for each comic page were between 8Kbyte to 19Kbyte in the example used in Sect. 5.3.

7. Conclusion

We implemented a system for reading comics on cellular phones (and other mobile computers). We proposed a decomposition method to overcome two main problems in the implementation of digital mobile comics: screen dimension and data size. In addition, we also developed a comic engine for reproducing comics on mobile computer screens. Through various experiments, the system and the comic engine exhibited good performance. To apply our system to general cases, the followings are important future work.

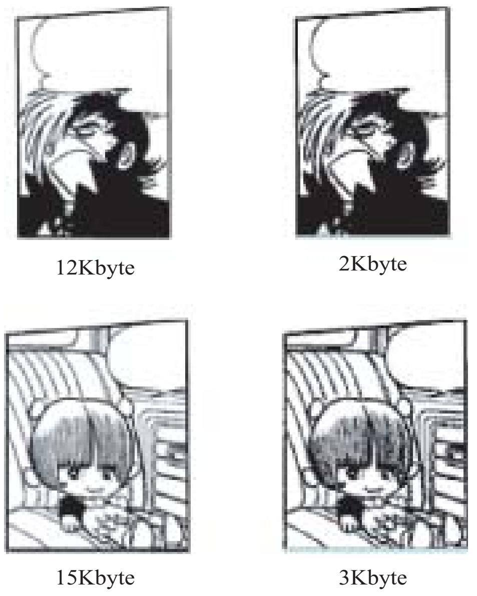


Fig. 14 Pre-compressed images (left) and reproduced images by our engine (right)

Drawn image data and speech text data are combined with comic engine and translated into an application unit for iAppli. Therefore, the engine program is downloaded each time for the time being. This problem will be solved in the future by using another function provided by a mobile phone company.

Drawn image sometimes results in poor quality if the pixel size of the frame is too large. These frames need some transformation to be comfortablely read, such as dividing frames into smaller pieces.

In general, screen tone patterns are used well in comics. Four level gray-scale images used in the current system are insufficient to display screen tone patterns. Reproduction techniques of screen tone patterns are also important for improving poor image quality.

Acknowledgements

We wish to thank Tezuka Production Co., Ltd. for their willing consent on the use and reproduction of Tezuka's works. This research was supported in part by the grant-in-aid for Private University High-Tech Research Center from the Ministry of Education, Japan.

References

[1] Roid Inc., Roid Time Net, http://www.comictime.net/index.html.
[2] MBEAT.COM, Inc., mangalian, http://www.mangalian.com/pc/.
[3] eBOOK Initiative Japan, Co., Ltd., 10daysbook, http://www.10daysbook.com/.
[4] D. Kurlander, T. Skelly and D. Salesin, "Comic Chat", Computer Graphics Proceedings, SIGGRAPH, pp.225–236, 1996.

[5] T. Yamada and T. Watanabe, "Identification of Person Objects in Four-Scenes Comics of Japanese Newspaper", Proc. International Workshop on Graphics Recognition, pp.152–159, 1999.

[6] T.F. Syeda-Mahmood, "Detecting Perceptually Salient Texture Regions in Images", Computer Vision and Image Understanding, vol.76, pp.93–108, 1999.

[7] E. Ojanen and J. Veijalainen, "Compressibility of WML and WMLScript byte code: initial results [Wireless Markup Language]", Proc. 10th International Workshop on Research Issues in Data Engineering, pp.55–62, 2000.

[8] D. Chen, P. Seshadri, "An Algebraic Compression Framework for Query Results", Proc. International Conf. on Data Engineering, pp.177–188, 2000.

[9] O. Tezuka, Black Jack, Akita Shoten, vol.1, 1993.

[10] M. Yamada, K. Amano and S. Miyazaki, "An Image Transformation Method for Reading Comics on Cellular Phones", Proc. 6th IASTED International Conf. on Internet and Multimedia Systems and Applications, pp.230–235, 2002.

[11] M. Yamada, R. Budiarto, M. Endoh and S. Miyazaki, "A System for Reading Comics on Cellular Phones and Comic Image Decomposition", Proc. 8th International Workshop on Mobile Multimedia Communications, MoMuC2003, pp.17–22, 2003.

Appendix: A Frame Sequence Search

We use a depth first search to form frames into unique sequences. The search starts at a frame and traces its neighboring frame recursively. If a frame f_c has plural neighbors, the neighbors are given a priority order as the following manner. The neighbors can be corresponded to each of the border lines of f_c. Let the border lines be denoted by $l_1, l_2, \cdots, l_m$ counterclockwise. Each border line is given a direction counterclockwise. We give the highest priority to border line l_i such that the angle between x-axis and l_i is more than 255°, and the nearest one to 255° among the border lines. The other border lines are given priority from l_i counterclockwise. As the result, the priority order of border lines, $l_i, l_{i+1}, \cdots, l_m, l_1, l_2, \cdots, l_{i-1}$ are obtained. The corresponded neighboring frame has the same order of priority. Let N be a set of neighboring frames of f_c such that $f \in N$ has not been included yet in the current sequence. If $f \in N$ has the highest priority in N, let f be denoted by $hn(N, f_c)$. The search also uses a minimum distance between a frame and an edge of the top and right side of the page. Let U be a set of frames that have not been included yet in the current sequence. If $f \in U$ has the minimal distance among U, let f be denoted by $md(U)$. If there is no frame neighboring a current frame, the search continues from $md(U)$. The search procedure always checks whether the sequence satisfies the order of two neighboring frames, where the order is obtained from rules R1, R2, R3 and R4. If it has found that the current sequence does not satisfy the correct order, the search looks for another sequence. The search is complete when all frames have been included into a sequence.

Figure A·1 shows the search, $FSS(U, L, f_c)$. Frame sequence L is obtained by calling $FSS(F - \{f\}, [f], f)$, where F equals all frames in a page and f is $md(F)$. The function $satisfy(L)$ returns true if L satisfies the order obtained by the rules. $[L \mid f_n]$ denotes the sequence that is obtained from appending f_n to L from the rear.

```
procedure $FSS(U, L, f_c)$:
    $U$: a given set of frames.
    $L$: a list of frames (frame sequence).
    $f_c$: a given frame.
begin
    if($U = \phi$) return $true$;
    else{
        $N$: a set of frames neighboring $f_c$.
        $N = N \cap U$;
        while($N \neq \phi$){
            $f_n = hn(N, f_c)$;
            $N = N - \{f_c\}$;
            if($satisfy([L \mid f_n])$ and $FSS(U - \{f_n\}, [L \mid f_n], f_n)$)
                return $true$;
        }
        $f_n = md(U)$;
        if($satisfy([L \mid f_n])$ and $FSS(U - \{f_n\}, [L \mid f_n], f_n)$)
            return $true$;
        else
            return $false$;
    }
end
```

Fig. A·1 Frame sequence search

Masashi Yamada is a lecturer in the School of Computer and Cognitive Sciences at Chukyo University, Japan. His research interests include artificial intelligence and multimedia applications. He received a B.S. and an M.S. degrees in Electrical and Computer Engineering from Nagoya Institute of Technology in 1992 and 1994. He received a PhD in Engineering from Nagoya Institute of Technology.

Rahmat Budiarto is a lecturer in the School of Computer Sciences at Universiti Sains Malaysia (USM) in Penang, Malaysia. His research interests include intelligent systems and network management/security systems. He received a PhD in Engineering from Nagoya Institute of Technology in 1998.

Mamoru Endo is a lecturer the School of Computer and Cognitive Sciences at Chukyo University, Japan. His research interests include network systems, computer graphics, virtual reality and their application in real society. He received a B.S. degree in Information Engineering from Shinshu University, an M.S degree and a PhD in Human Informatics from Nagoya University.

Shinya Miyazaki is an associate professor in the School of Computer and Cognitive Sciences at Chukyo University, Japan. His research interests include computer graphics and virtual reality. He received an M.S. degree and PhD in Information Engineering from Nagoya University.