CD-ROM Proc. of Nicograph International 2009, Kanazawa, Japan,
June 19-20 2009.

Camerawork Editor for Automatic Comic Generation from Game Log

Ruck Thawonmas, Ko Oda, and Tomonori Shuda
Intelligent Computer Entertainment Laboratory
Ritsumeikan University
1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan
ruck@ci.ritsumei.ac.jp

In this paper, we propose a module for editing the camerawork of comic generated automatically from online-game play log by a system recently developed by us. The proposed module enables the user to readily edit the camerawork of a comic frame. We discuss results of the questionnaire taken during the system demo conducted at the 71th National Convention of the Information Processing Society of Japan during March 10-12, 2009. These results confirm the effectiveness of this module.

1. Introduction

Comic has recently been used as a media for summarizing user experiences in an entertaining fashion. For example, it has been used for summarizing activities in a conference [1], daily activities [2], and game-play activities [3-7]. Comic-based summaries can not only augment users' personal memories but also promote communication among their communities.

Recently, we have developed a comic generation system that automatically generates comic from game log [4-6]. Our approach is similar to the one in [3] in that the game engine is used for rendering comic based on information in the game log. Another approach adopted in [7] is to compose comic from selected screen shots. The difference between our system [4-6] and the system in [3] is that our system can provide more varieties of frames and prevent having similar consecutive frames. However, the camerawork of our system is limited and needs improvement.

The organization of this paper is as follows. In the next section, we briefly describe our previous system, followed by a section on the new system. Then we give our descriptions on system implementation and evaluation, followed by our discussions.

2. Our Previous System

In our previous system, first, the acquired game log is temporally divided into multiple partitions, each called a scene. A scene consists of multiple game events and connects a story (if any) in the play. A scene is then further divided into multiple chunks,

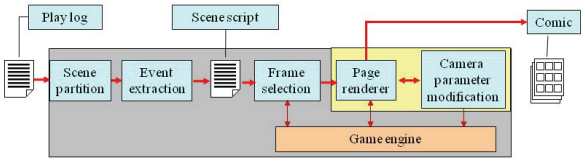

Fig. 1 Proposed comic generation system

each called an event. An event is composed of multiple related interactions such as *approach* and *fight*. Idioms, or rules, for deciding rendering parameters are determined during event extraction.

After scenes and events have been extracted, the game engine is used to render those events as candidate comic frames from which important ones will be selected for the comic. Here, to prevent extraction of similar consecutive frames, a method based on HSOM was proposed in [5]. This method removes frames whose content is similar to recently selected frames. After frames have been selected, the parameters for deciding frame size and shape are decided. These parameters are used for rendering each comic page.

3. Proposed System

Figure 1 shows our new comic generation system that consists of the previously proposed modules and the newly proposed module (in yellow). The new module contains the page renderer sub-module and the camera parameter modification sub-module. With the page renderer sub-module, the user can choose a frame that they want to edit, and edit that frame with the camera parameter modification sub-module. The new module allows direct edition of the camera parameters of those frames and thus enables representation of comic with a high variety of camerawork.

3.1 Page Renderer

Figure 2 shows an example of the output screen from the page renderer sub-module. All frames in a page of interest are rendered here based on the information in the frame information list (described later in Section 4.1). This screen accepts the input via mouse operations, allowing the user to choose any displayed frame by clicking the mouse on it. Once a click is performed, the corresponding frame number is acquired from the coordinate information in the screen. The system will then shift to the camera parameter modification sub-module, and this sub-module will display its output screen for the chosen frame.

After finishing the editing task, the user can click a specific button to end the system. Once this button is clicked, all frames including those with updated camera parameters will be rendered using the game engine. The resulting comic will be saved into a file.

3.2 Camera Parameter Modification

Figure 3 shows an example of the output screen from the camera parameter modification sub-module. Here the frame chosen at the page renderer sub-module is displayed, the right-bottom one in the example. Its camera parameters can be modified with specific keyboard inputs. An example of an edited frame of this example is shown in Fig. 4.

To be more precise, the user can readily edit three types of camerawork:

i) Edition of the camera angle in the following four orientations: Right, Left, Up or Down;

ii) Edition of the camera position, while fixing its angle, in the following four directions: Right, Left, Up or Down;

iii) Edition of the zoom position: In or Out.

In addition, the reset button is implemented for allowing the user to retrieve the original frame, the one whose camera parameters have not been modified. The corresponding keyboard keys adopted in our system are shown in Fig. 5.

This sub-module renders the corresponding frame each time it receives a keyboard input from the user. After finishing editing of the current frame, the user presses a specific button. This will make the system go back to the page renderer sub-module whose output screen will updated taking into account the recently edited frame (Fig. 6).

4. System Implementation and Evaluation

4.1 Implementation

We implemented the proposed system for The ICE, the online game under development at our laboratory. The game engine of The ICE was therefore used in our system. However, we note here that the system is applicable to any game if its game engine is accessible for rendering comic.

As shown in Fig. 7, the map name and the character data, necessary for rendering comic, were added into each element, corresponding to a frame, of the frame information list. In the previous system, the frame information list contains less information, based on which rendering of the output screen for the new module was not possible. Addition of the aforementioned pieces of information solves this technical issue.

In addition, the frame layout module residing in the previous system is not used.

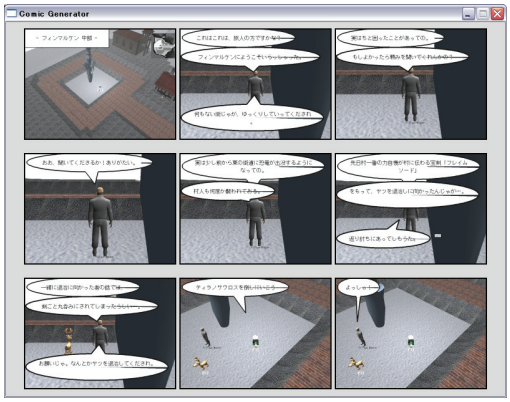

Fig. 2 Output screen from the page renderer sub-module

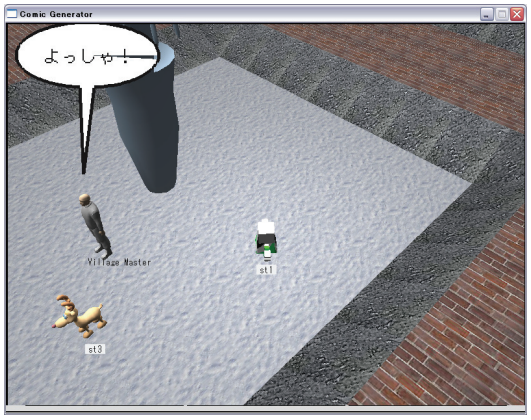

Fig. 3 Output screen from the camera parameter modification sub-module

Fig. 4 Result after editing the frame in Fig. 3

This module automatically decides both frame size and shape. Because the aim of this research is to study the effect of camerawork,

not frame layout, this module was excluded from the present system.

4.2 Evaluation

The objective here is to evaluate from the user perspective whether the comic generation system with the camerawork editing module is effective in generating comic that represents play memories and to investigate the user impression on the generated comic before and after camerawork edition. We first asked each evaluator to play The ICE. Comic was then automatically generated from the play log. We then asked the evaluator to edit the camerawork of their comic, compare the two versions of comic (before and after camerawork edition), and answer the questions in the questionnaire described in Section 4.3.

This evaluation was conducted during the demo of the system at the 71th National Convention of the Information Processing Society of Japan, March 10-12, 2009. We were able to collect questionnaire data from seven demo participants who volunteered to spend their time evaluating the system. All of them are male with the age around 20.

4.3 Questionnaire Content

The questions used in the questionnaire are as follows:

Q1 Does the system suitably represent memories of your play?

Q2 Comparing the versions before and after camerawork edition, which one would you like to show to other persons?

Q3 Is the function for editing the camera position useful?

Q4 Is the function for editing the camera angle useful?

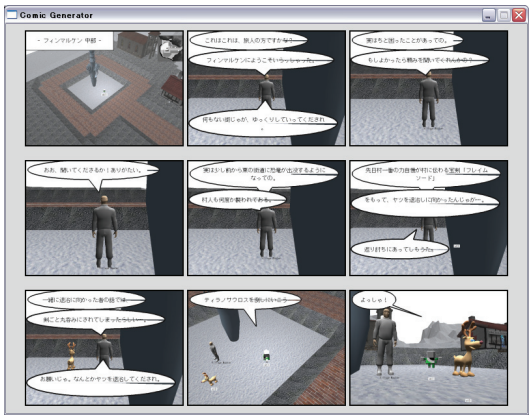

Fig. 5 Updated output screen from the page renderer sub-module

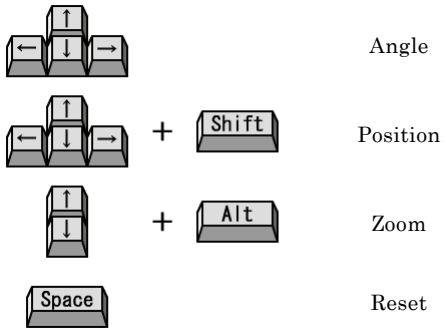

Fig. 6 Keyboard interface for camerawork edition

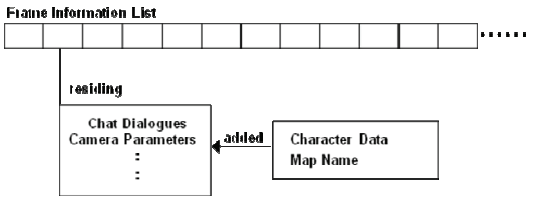

Fig. 7 Extended frame information list

Q5 Please let us know other comments you might have.

5. Results and Discussions

The results for Q1 and Q2 are shown in Tables 1 and 2, respectively. Those for Q3 and Q4 are summarized in Table 3. Our discussions on these results are as follows.

1) Representation of Play Memories

From Table 1, almost all of the evaluators considered the generated comic suitably represent their play. This result hence confirms the effectiveness of the proposed system in generating comic that represents play memories with comic.

2) Comparison between the Two Versions

From Table 2, more than half of the evaluators selected the edited version. In other word, we can state that they had better impression on the edited one than the non-edited version. This might be because the camerawork-editing module enabled them to well reflect their feeling about their play memories into comic.

3) Usability

From Table 3, almost all of the evaluators considered editing of the camera angle and position is useful. This result might arise from the fact that edition of these parameters well assisted the evaluators to achieve the camerawork they wanted and that the implemented keyboard interface was easy to use.

6. Conclusions and Future Work

To improve the camerawork of our previous comic generation system, we proposed a new module for camerawork edition. We then performed system evaluation. Evaluation results confirmed that comic generated from our new system represents well user memories and the implemented editing functions are useful.

As our future work, we will be studying on automatic camerawork planning.

Table 1 Representation

	Persons
Very Good	2
Good	4
Satisfactory	1
Bad	0
Very Bad	0

Table 2 Comparison

	Persons
Before	0
After	4
Both	2
Not Sure	1

Table 3 Usability

	Position	Angle
Yes	6	6
No	0	0
Not Sure	1	1

References

[1] Sumi, Y., Sakamoto, R., Nakao, K., Mase, K.: ComicDiary: Representing individual experiences in a comic style. In: Borriello, G., Holmquist, L.E. (eds.) UbiComp 2002. LNCS, vol. 2498, pp. 16–32, Springer, Heidelberg (2002)

[2] Cho, S.B., Kim, K.J., Hwang, K.S.: Generating Cartoon-Style Summary of Daily Life with Multimedia Mobile Devices. In: Okuno, H.G., Ali, M. (eds.) IEA/AIE 2007. LNCS (LNAI), vol. 4570, pp. 135–144. Springer, Heidelberg (2007)

[3] Shamir, A., Rubinstein, M., Levinboim, T.: Generating Comics from 3D Interactive Computer Graphics. IEEE Computer Graphics and Applications 26(3), 53–61 (2006)

[4] Ruck Thawonmas and Tomonori Shuda, "Comic Layout for Automatic Comic Generation from Game Log," IFIP International Federation for Information Processing, vol. 279/2008 (ECS-2008), pp. 105-115 (2008)

[5] Tomonori Shuda and Ruck Thawonmas, "Frame Selection for Automatic Comic Generation from Game Log," Lecture Notes in Computer Science, Scott M. Stevens and Shirley J. Saldamarc (Eds.), vol. 5309 (ICEC 2008), pp. 179-184 (2008)

[6] Tomonori Shuda and Ruck Thawonmas, "Automatic Comic Generation Using Online-game Play Log," Journal of Game Amusement Society, vol. 3 no. 1, pp. 40-45 (2009) (in Japanese)

[7] Chia-Jung Chan, Ruck Thawonmas, and Kuan-Ta Chen, "Automatic Storytelling in Comics: A Case Study on World of Warcraft," CHI Extended Abstracts 2009: 3589- 3594 (2009)