Journal of Korea Multimedia Society Vol. 12, No. 12, DECEMBER 2009(pp. 1736-1744)

A Two Layered Approach for Animation Sketching

Eisung Sohn[†], Jaewoong Jeon[††], Tae-Jin Park[†††], Won-Sung Sohn[††††], Soon-Bum Lim[†††††], Yoon-Chul Choy[††††††]

ABSTRACT

In this paper, we present an animation sketching system using a two layered approach. Animation sketching is a popular technique to create informal animations but it is often suffered by the low-quality output due to a trade-off between convenience and complexity. Our aim is to support sketching practical animation scenes easily and fast while not complicating the simple sketching interface. The key idea is to combine two conceptual stop motion layers, a whiteboard and cutout animation layer, in a seamless interface. As a background, the whiteboard animation layer handles stroke-oriented objects, while the cutout animation layer takes charge of transform-oriented objects. We found that this approach enables users to express more complicated animation fast while still maintaining a concise sketching interface. We demonstrate the usability and flexibility through resulting animations from user experiments.

Key words: Animation Sketching, Sketch-based Interface, Content Authoring

1. INTRODUCTION

Animation sketching is a popular technique to create animations easily for general users. Creating animations using sketches or gestures can be useful in many situations. Unlike complex traditional animation tools, it is easy, intuitive, and rapid in creating informal animations. When someone is explaining, teaching, or presenting something, he or she can deliver it more efficiently using some animations together because moving pictures can attract attention more than static ones.

The animation tools for this purpose have a different aim than others. It has to be easy to learn and quick to produce results. This is more important than producing a polished result. Previous animation sketching systems can produce a simple animation easily, but it is often not practical to make various animations due to lack of expression ability. A practical animation tool has to have enough flexibility to express various ideas while maintaining an easy and concise user interface. The problem is that, because a sketching interface basically has a low degree of freedom (DOF), its results tend to be of low quality. To produce a wide range of animation expression, a way overcoming these issues is needed.

In this paper, we present a concept of creating animations through manipulating animation objects like paper cutouts. Our method applies the concept of cutout animation and whiteboard animation for creating and displaying animation [1].

※ Corresponding Author : Yoon-Chul Choy, Address : (120-749) Room C512, 3rd Engineering Hall, Yonsei University 262 Seongsanno, Seodaemun-Gu, Seoul, Korea, TEL : +82-2-2123-2712, FAX : +82-2-365-2579, +82-2-393-7663, E-mail : ycchoy@mglab.yonsei.ac.kr
Receipt date : Nov. 30, 2009, Revision date : Dec. 18, 2009
Approval date : Dec. 29, 2009
[†] Dept. of Computer Science, Yonsei University, Korea (E-mail : essohn@gmail.com)
[††] Dept. of Computer Science, Yonsei University, Korea (E-mail : demiblue@gmail.com)
[†††] Dept. of Computer Science, Yonsei University, Korea (E-mail : parktj2003@gmail.com)
[††††] Dept. of Computer Education, Gyeongin National University of Education, Korea (E-mail : sohnws@ginue.ac.kr)
[†††††] Dept. of Multimedia Science, Sookmyung Women's University, Korea (E-mail : sblim@sookmyung.ac.kr)
[††††††] Dept. of Computer Science, Yonsei University, Korea
※ This work is supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2008-313-D01012).

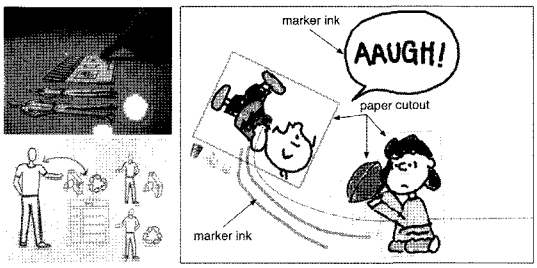
Our approach basically is an animation sketching system built upon using a motion-by-example technique that animates according to demonstrations made by a user. We also present a timeline control that is capable of coordinating complex motion, along with an editing method to apply multiple motions to an object.

1.1 Basic Concept

Anyone can utilize a whiteboard to aid his ideas for drawing various figures, diagrams, and text. A whiteboard has strong advantage over static drawing, but it is fairly tedious to express an animation with it. To create 'whiteboard animation', since it is stop-motion animation, every frame should be captured while erasing and redrawing the changed region. It may be better than flip book animation, in which everything has to be redrawn, but it is still a tedious job.

There is a simple method that makes whiteboard animation a lot easier - make an animated object into a movable paper cutout and then make an animation by manipulating it with the hands. This method might make the motion task easy and at the same time allow the whiteboard to be used as the background. This concept can be thought of as a hybrid form of whiteboard animation and cutout animation. In whiteboard animation, expressing text, line, background, and some drawing effects can be handled easily. In cutout animation, object motion can be handled easily because all the objects are paper cutouts. We have explored how to exploit these two animation methods in computer-aided animation tools for improving output quality, reducing user interaction, and addressing some issues in animation sketching.

Our method was inspired by movie clips of Common Craft Show (http://commoncraft.com). Their service is explaining various topic by using animated paper cutouts and marker ink on whiteboard as shown in Figure 1(a).



(a) (b)

Fig. 1. (a) A concept of whiteboard and cutout used in Common Craft Show (b) A simple animation created by using our system

1.2 Contributions

The primary benefits of our approach are as follows: a fast animation creation process due to small delays in object selection; the ability to express detailed objects or complex motion; the ability to use image objects; and the ability to apply multiple motions by editing a moving object.

Our method has a short selection time. The delay in object selection is a typical problem of previous stroke-oriented animation systems [2-5]. It not only causes time delays but also might result in a considerable decrease in usability as the number of objects in an animation increases. Since every stroke is considered a movable object, almost every instance of applying motion requires a lassoing selection. In our approach, instead of selecting individual strokes, a user creates and selects a paper cutout object that contains drawing. It is already logically distinct by a user as a movable object since the beginning. This helps users to manipulate and edit complex motions, and this cutout concept makes support for image object and detailed drawing more natural.

Our system also introduces a novel method for applying multiple motions. One of the difficulties of sketching animation is making compound motions such as changing an object's size or orientation during its movement. They are often required in some kinds of animations, but they could not be freely handled using previous approaches.

2. RELATED WORK

Sketching interfaces offer a natural method of interaction because they mimic the traditional pencil-and-paper method. Recently, many sketch-based approach have been presented in computer animations.

There have been many sketching systems aimed for specific purposes include math equation [6], rigid-body simulation [7], physics-based mechanical simulation [8], and character animation [9]. However, here we focus animation sketching interfaces that for creating general animation.

Kato et al. proposed a prototype system called KOKA [10,11]. In their system, line gestures, similar to the motion lines used in cartoons, are interpreted as motion commands like rotation, translation, and oscillatory motion. It aims to communicate well with real-time observers and the animation system itself. In KOKA, sketching is used only to specify animation, not to handle object creation. A similar approach was taken by Rogers in the system called Living Ink [5]. Rogers formed an optimistic view of the prospects for creating real-time animation in the system. Its basic interaction paradigm is stack based. Compared to KOKA, it opted for a more conservative solution such that movements could be specified with visible sketched strokes. These two approaches are appropriate to communicate with real-time observers, but gesture-oriented systems can only handle relatively simple animation and are not yet capable of dealing with various types of animation due to ambiguity problem.

Moscovich et al. proposed the idea of animating 2D sketched shapes using motion-by-example in their prototype system RaceSketch [4]. In their system, the animator simply grabs an object, and moves it about as he likes, and then the position and timing information is recorded and the motion can then be played back. Also it addresses problems such as time warping for timing control, a multi-layering method for composed motion, and freeform skeletal control. The K-Sketch is another approach based on motion-by-example by Davis et al. [2,3] They conducted field studies to find out how an informal animation tool might be used. Their goal was not to design novel interaction techniques but rather to focus on high-level choices regarding tool features. These motion recording approaches are intuitive for handling low-level motions.

3. ANIMATION SKETCHING SYSTEM

3.1 Creating Animation Objects

Animation objects can be created in two ways in our system. As shown in Figure 1(b), one is a paper cutout object, which can be manipulated like a paper cutout, to express moving objects. It is displayed as a transparent sheet on which pen strokes can be drawn. The other one is a marker ink object, which is usually used for drawing text, lines, arrows, background drawings, and various effects like smoke, wind, water, etc. If a user selects the pen mode and draws, a paper cutout object is created, and if a user selects the marker mode and draws, marker ink is then drawn in background.

3.2 Creating Paper Cutouts

When a user creates a paper cutout object, its size and position should be specified. We thought that this could be somewhat of a burden to a user. Therefore, we decided to make this step automatically so as not to sacrifice the benefit of the simple object creation process compared to previous systems. If a user draws on a canvas, a small paper cutout object is created automatically, and its size is expanded as the drawing gets bigger.

When a new stroke is being applied, based on whether the user's drawing is inside of existing paper cutout objects or not, the decision to append strokes on it or to create a new paper cutout object is made. If a user keeps drawing on an existing

paper cutout object, all the strokes are appended to the object without increasing the number of animation objects. If a user keeps drawing outside of the objects, in the background, new paper cutout objects are created continuously.

One benefit of this data structure is that an image can be used as a background instead of a transparent background without any further changes. Our system supports a scissors mode for image cutout. In addition, a user can modify the clipped image by drawing on it.

3.3 Drawing Marker Ink

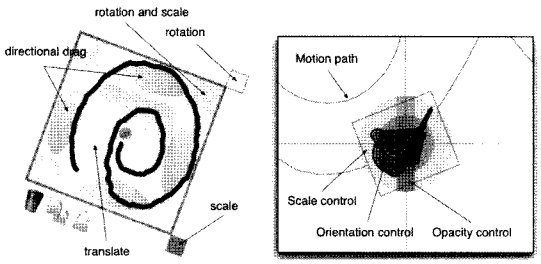
A marker ink object is the one used to draw on the background of an animation, behind the paper cutout objects. By drawing on a background after selecting the marker, a marker ink object is created, and the system remembers every drawing point. Therefore, when the animation is playing, a marker ink object drawn in pause status appears instantly, while the others drawn in play status appear gradually as they are recorded.

3.4 Animation Object Manipulation

A sketching interface has little degree of freedom. Therefore manipulating objects is not easy. Therefore, solutions to address this problem have been studied by many researchers. One method is using a menu interface by which a user can perform a specific movement or transform an object. This is an easy and intuitive method for novice users. Our approach is similar to this method, but the difference is that instead of popping up the menu on the events, each paper cutout object has manipulation areas on itself. This approach helps in manipulating animation objects like real paper cutouts. Figure 2 shows the manipulation areas of a paper cutout object.

3.5 Multiple Motion Control

Our system supports multiple motions of paper



(a) (b)

Fig. 2. (a) Manipulation areas of cutout object (b) Camera view and a pop up menu interface for multiple motion control

cutout objects. A user can change an object's size, orientation, or opacity value while it is slowly moving. In sketch environments, a user can manifest only one point at a time using a pen or a mouse. It is a difficult problem to perform both movement and rotation at the same time. Indeed, previous animation sketching systems proposed a technique for controlling multiple motions.

In K-Sketch, movement and rotation operations are recorded separately for making an object rotate while it is moving. For example, if we want to create an animation of a rolling stone falling down a slope, First, a user draws a stone and rotates it, and then he selects both the object and the rotating motion path and moves them together down a slope. This approach is appropriate to create rough motion, but it is not capable of making motions that change orientation freely along the motion path.

In our approach, we first records the movement and then applies orientation along the motion path. After the object's motion path is established, a method to assign orientation value gradually is needed. To assign an orientation value to a moving object by sketching input, two methods are considered. One is displaying a menu at the static position on the background while the object moves on, letting the user handle orientation, scale, or opacity value through that menu. The other method is to fix the object at the center of the animation canvas and pop up a menu upon it so that a user can control the object's attributes.

We compared these two methods in our initial experiments, and we found that the latter method offers better usability. Therefore, to fix an object at the center of an animation canvas, the concept of a camera mode is introduced. If the camera mode is selected, the target object is fixed at the center of an animation canvas. As the object is moving, the camera also moves along with the object; thus, the target object is always fixed at the center of an animation canvas. At the same time, as shown in figure 2(b), a menu for editing attributes pops up on it so that a user can manipulate the object by clicking and handling it. In camera mode, animation time slowly advances so that the user can manipulate it easily.

3.6 Recording Motions

To record the motion of animation objects, our system provides two methods. One is real-time recording such that a user's motion is recorded in real-time while the system's time is going on. The other is procedural recording that records only while a user's manipulation is being applied.

When a user presses the play button and then manipulates an object, the motion is recorded in real time. However, unwanted blank spaces may occur because time keeps going even while the user is interacting with the menu. Through our experiments, we found that this real-time recording technique is generally difficult for coordinating motions because it is a quite burden to a user to make the timing right while time goes by. Therefore, we think that this method might be more appropriate for drawing a marker ink object, which is more natural for a real-time recording environment and requires little timing sense on the part of the user.

The procedural recording is not a real-time technique. It offers a big advantage in controlling motion not in a hurry. Therefore, it typically produces motion of better quality. To perform procedural recording, a user clicks a button at the center

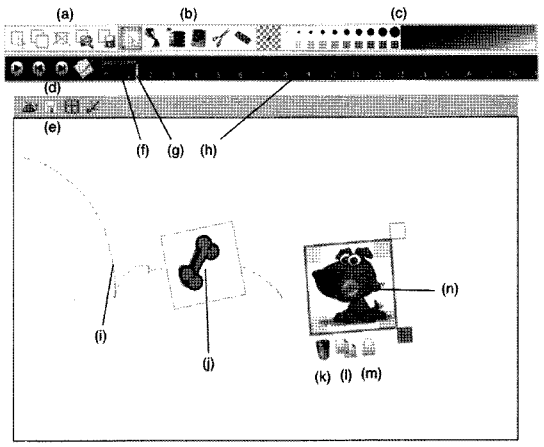


Fig. 3. User interfaces of our system. (a) Canvas tools: Select, Create, Delete, Load, Save, (b) Drawing tools: Select, Pen, Marker, Image, Scissor, Eraser, (c) Paint tools: Preview, Stroke size / Opacity, Color, (d) Playback tools: Play, Backward, Forward, Edit mode, (f) Bar indicator, (g) Slider knob, (h) Timeline Control, (i) Motion path, (j) Paper cutout object, Selected object menu: (k) Erase, (l) Duplicate, (m) Lock, (n) A procedural recording button.

of a paper cutout object as shown in figure 3(n). When the button turns red, it is in the procedural recording state. The user grabs the object by an area of preferred operation and moves it, and then every motion is recorded only while a motion is being applied.

3.7 Editing Motions

Users can edit a recorded motion by going back in time and recreating the motion all over again. Modification can be done from that position by moving the object with procedural or real-time recording. The motion of each paper cutout object is displayed as a path of points over time, and a user can modify the motion by erasing the path directly.

To do this, users simply click the eraser button and then erases the desired region of the motion path. The erased path disappears. If the motion path is divided in two by erasing in the middle of the path, the object might stop at the end of the

first path for a while and then move on to the beginning of the next path.

3.8 Timing Control

The motion-by-example method is intuitive and fast to record motions but not precise. Timeline control is one of the refinement solution of our system. After a motion is recorded, a time slider appears in the timeline control as shown in Figure 3(h). The timeline interface is a global control for displaying and controlling the time value of a current selected animation canvas. It displays information about the current canvas with a ruler, a slider, and bars. A user can designate the current time value by clicking or dragging at a certain point of a timeline interface.

In the timeline interface, each paper cutout object is displayed as a semi-transparent bar. Each line at the end of a bar indicates start time and end time. This simple concept helps a user to understand and to control timing intuitively, especially in case of coordinating complex motions.

4. IMPLEMENTATION

A brief overview of our system is shown in Figure 4. Our system is implemented in JavaScript for web environment. We intended this to allow users to create and share animations directly on a web page. A single line is to be inserted to the source web page to include our module written in JavaScript.

We use the <canvas> tag to use 2D vector graphics in Web browser. As shown in Figure 5, our system creates transparent layers and displays animations on it so that no changes to be made to a layout of the existing web page. In addition, instead of covering the whole page with an animation layer, a user can create multiple animation layers at the desired position and size.

We considered that our system would be used mainly in a Tablet PC environment. However, a desktop PC with a mouse also works well. Users can save animation results in a remote server for showing to every visitor of the web page.

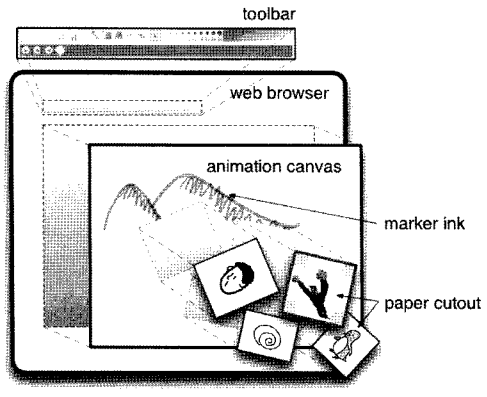


Fig. 4. A brief overview of our animation system

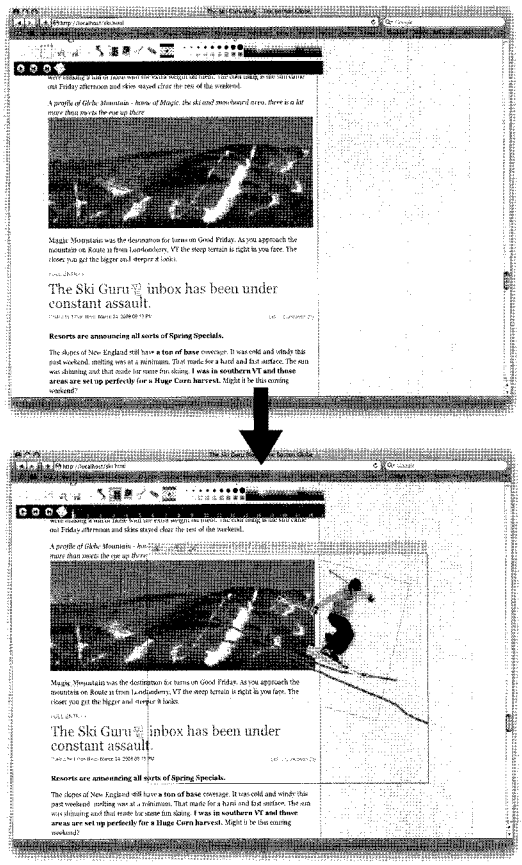


Fig. 5. A transparent animation layer overlaid on a web page.

5. EXAMPLE SCENARIOS

We asked 13 novice participants to test our system: 8 graduate students and 5 undergraduate students. After 30 minutes of training, we asked them to create a simple animation among some topics of their choice. The following animation examples are some selections from these user experiments. Each example took about 10-20 minutes to animate.

5.1 2 on 2 Basketball

As shown in Figure 6, the animation displays a short basketball game. Its running time is about 39 seconds, and it took about 6 minutes to create. The animation was overlaid on a page in a basketball Web site.

First, the hoop was drawn with a marker as the background. Then a ball and a player were drawn with a pen, and then other players were duplicated. A generated motion has many complicated and overlapped motions.

5.2 Intersection

Figure 7 shows an intersection where many cars passing by. The running time of this animation is

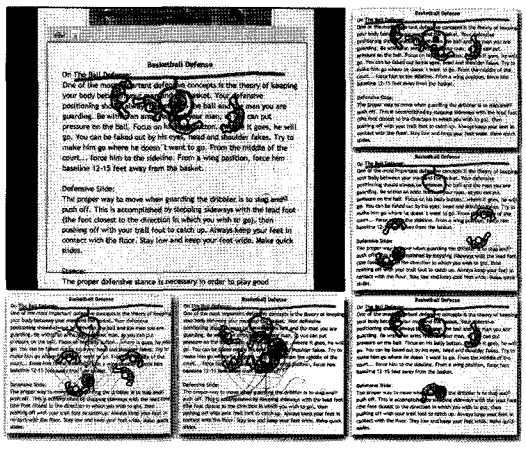


Fig. 6. "2 on 2 basketball" animation. Many complicated and overlapped motions are included

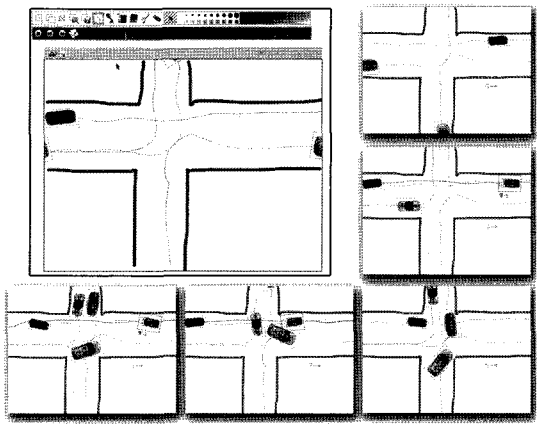


Fig. 7. "Intersection" animation. 5 cars pass by without a crash

only 6 seconds, but it took about 8 minutes to create.

The animated objects are all images of different cars. 5 cars drive through the intersection almost at the same time without a crash. The participant generated this animation by adjusting each car's timing by dragging each bar in the timeline control.

5.3 Escape from a Rocket

Figure 8 shows an animation about an escape from a rocket crash. The rocket is getting smaller as it falls, and an explosion grows bigger at a fixed point. The pilot's form is changed on some occasions. This was done by replacing it with a

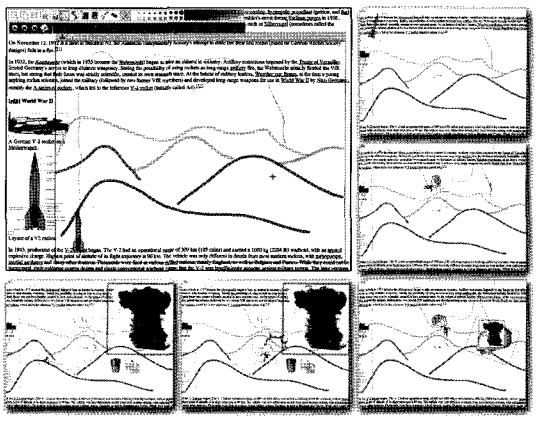


Fig. 8. "Escape from a rocket" animation. Multiple motions are included

new paper cutout object. The running time of this animation is about 18 seconds, and it took about 15 minutes to create.

6. CONCLUSIONS AND FUTURE WORK

The animation sketching methods, including our system presented here, are not substitutes for the existing professional animation tools and techniques. These are highly accessible ways for users to create various animations in informal use. However, the previous methods are still not capable of expressing various kinds of appealing animations. We explored new techniques to tackle this problem.

Our system provides greater usability and flexibility in creating and manipulating animation objects. The experimental results show that our method makes it easy to create appealing animations, and it is more capable of expressing complex motion and detailed drawing than previous methods while still requiring minimal user interaction.

Although we have overcome some limitations of previous methods, animation sketching techniques still need to be improved in many ways. In the near future, we will continue to investigate methods to increase usability and animation quality. We plan to implement a template interface to make an animation process easy for novice users who need more artistic assistance. Also, we are interested in extending the method for a multi-touch interface. Finally, we plan to run an evaluation in educational environments to fully demonstrate the usability of our tool.

REFERENCES

[1] E. Sohn, W. Sohn and Y. Choy. "Paper Cut-out Style Animation Sketching," Proceedings of the KMMS, pp. 303, 2009.

[2] R. Davis, B. Colwell, and J. Landay. "K-sketch: A "kinetic" sketch pad for novice animators," Conference on Human Factors in Computing Systems - Proceedings, pp. 413-422, 2008.

[3] R. Davis and J. Landay. "Informal animation sketching: Requirements and design," Proceedings of 2004 AAAI Fall Symposium on Making Pen-Based Interaction Intelligent and Natural, pp. 21-24, 2004.

[4] T. Moscovich and J. Hughes. "Animation sketching: An approach to accessible animation," Unpublished Master's Thesis, C. S. Department, Brown University, 2001.

[5] B. Rogers. "Living ink: Implementation of a Prototype Sketching Language for Real Time Authoring of Animated Line Drawings," EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling, pp. 115-122, 2006.

[6] J. J. LaViola, Jr. and R. C. Zeleznik. "Mathpad2: a system for the creation and exploration of mathematical sketches," ACM Trans. Graph., 23(3):432-440, 2004.

[7] J. Popovic, S. M. Seitz, and M. Erdmann. "Motion sketching for control of rigid-body simulations," ACM Trans. Graph., 22(4):1034-1054, 2003.

[8] C. Alvarado and R. Davis, "Resolving ambiguities to create a natural computer-based sketching environment", In In Proc. of IJCAI, pp. 1365-1371, 2001.

[9] M. Thorne, D. Burke, and M. van de Panne. "Motion doodles: an interface for sketching character motion," In SIGGRAPH '04: ACM SIGGRAPH 2004 Papers, pp. 424-31, New York, NY, USA, 2004.

[10] Y. Kato, E. Shibayama, and S. Takahashi. "Effect lines for specifying animation effects," Proceedings - 2004 IEEE Symposium on Visual Languages and Human Centric Computing, pp. 27-34, 2004.

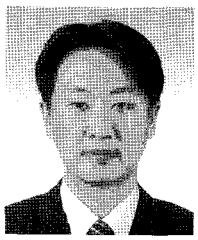
[11] S. Takahashi, Y. Kato, and E. Shibayama. "A new static depiction and input technique for 2d animation," Visual Languages and Human-Centric Computing, pp. 296-298, 2005.

Eisung Sohn

received the BS degree in Mechanical Engineering from Hongik University, Seoul, Korea in 2004. He is currently a Ph.D. candidate in Computer Science from Yonsei University, Seoul, Korea. His research interests include HCI, Computer Animation and sketch-based interface.

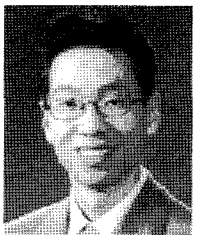
Jaewoong Jeon

received the M.S. in Computer Science from Yonsei University, Seoul, Korea in 2006. He is currently a Ph.D. candidate in Computer Science from Yonsei University, Seoul, Korea. His research interests include Sketch-based interface, 3D Animations, Non-photo realistic rendering and Mobile interface.



Tae-Jin Park

received the BS and MS degrees in computer science from Yonsei University, Seoul, Korea, in 1993 and 1995, respectively. Previously, he worked as a research engineer at LG Digital Media Research Laboratory and LG DTV Laboratory in 2000 and 2005. Currently, he is working towards the PhD degrees in the Department of Computer Science at Yonsei University. His research interests include DTV system middle-ware, DTV data broadcasting, multimedia information interface, multimedia data presentation and DMB.

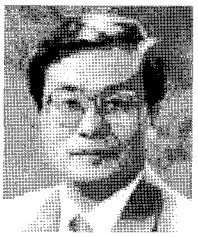


Won-Sung Sohn

received the M.S. in Computer Engineering from Dongguk University, Seoul, Korea in 2002, and the Ph.D. in Computer Science from Yonsei University, Seoul, Korea. Dr. Sohn is currently working at the Department of Computer Education, Gyeongin National University of Education, Inchon, Korea. His research interests include HCI, multimedia document and computer education.

Soon-Bum Lim

received his MS and PhD degrees in Computer Science from Korea Advanced Institute of Science and Technology, Daejun, Korea, in 1983 and 1992, respectively. He was an Assistant Professor at Kunkook University from 1992 to 1997. He has been with the Department of Multimedia Science at Sookmyung Women's University, Seoul, Korea, where he is currently a Professor. His research interests include Computer Graphics, multimedia application, electronic publication, DMB data broadcasting and mobile interface.



Yoon-Chul Choy

received his MS degree from the University of Pittsburgh in 1975 and the MS and PhD degrees in IE & OR from the University of California, Berkeley, U.S.A, in 1976 and 1979, respectively. He was at Lockheed and Rockwell as an International Researcher in 1979 and 1982. He was a Visiting Professor at the University of Massachusetts, U.S.A in 1990 and 1991 and Keio University, Japan, from 2002 and 2003. He has been with the Department of Computer Science at Yonsei University, Seoul, Korea, where he is currently a Professor since 1984. His research interests include computer graphics, multimedia, non-photo realistic rendering and sketch-based interface.