Procedural animation for glove puppet show

Chih-Chung Lin and Tsai-Yen Li*

Department of Computer Science,
National Chengchi University,
64, Sec. 2, Zhi-Nan Road,
Taipei, Taiwan
E-mail: s9421@cs.nccu.edu.tw
E-mail: li@nccu.edu.tw
*Corresponding author

Gee-Chin Hou

Department of TV and Radio,
National Chengchi University,
64, Sec. 2, Zhi-Nan Road,
Taipei, Taiwan
E-mail: phou@nccu.edu.tw

Abstract: Traditional Taiwanese glove puppet show is a unique form of performing art. We aim to use the aid of computer animation technologies to preserve this cultural heritage and create innovative ways of performance. By observing the demonstration of a puppet master, we analyse the characteristics of how a hand puppet is manipulated and design animation procedures mimicking the motion of a glove puppet. These procedures are implemented on a real-time animation platform for procedural animation. Through the system, we allow a user to perform glove puppet animation with high-level inputs. We hope that, with the help of this system, not only the art of manipulating a glove puppet can be systematically documented, but also the entry barrier for learning it can also be greatly reduced.

Keyword: character animation; glove puppet show; procedural animation; traditional art.

Reference to this paper should be made as follows: Lin, C-C., Li, T-Y. and Hou, G-C. (2011) 'Procedural animation for glove puppet show', *Int. J. Arts and Technology*, Vol. 4, No. 2, pp.168–180.

Biographical notes: Chih-Chung Lin is currently a Graduate Student at the Department of Computer Science, National Chengchi University (NCCU). He received a bachelor's degree in 2009 from the same department. He has been a Visiting Researcher at the Intelligent Systems Centre (IntelliSys) of Nanyan Technological University, Singapore, from June to September 2009 for the project titled 'Replication and processing of human body motion for participatory interaction in cospace'. His research work focuses on 3D procedural animation tools and puppet animation. He is also extending his research interests to digital art, human–computer interface and machine learning.

Tsai-Yen Li is a Professor and the Head of the Computer Science Department at the National Chengchi University in Taiwan. He received his MS and PhD from Stanford University. He is a Member of ACM, IEEE, TAAI of Taiwan, IICM of Taiwan. His research interests include computer animation, digital character, crowd simulation, robot motion planning, virtual environment, intelligent user interface and intelligent tutoring system.

Gee-Chin Hou is a Senior Lecturer at the Department of Radio and TV at National Chengchi University. His research interests are in electronic media production and programming, instructional technology, musicology and musical creativity.

1 Introduction

Glove puppet show is one of the most popular folk arts in Taiwan and is also an indispensable entertainment in the daily life of Taiwanese. It combines elements of traditional drama, music, delicate costumes and props to perform a show. Being an exquisite art of action, the manipulation of glove puppet requires precise eye–hand coordination, and the glove puppet show also requires seamless team cooperation. Due to the nature of these psychomotor skills, repeated practice is must in order to achieve a mastery level of performance. Young puppeteers usually are trained by masters with oral instructions. Without written materials, a senior puppeteer let apprentices know a simplified storyline and have them observe closely when he/she performs. It usually took apprentices about four years to be competent as the assistant of principal puppeteer.

In recent years, the applications of computer animation are becoming prevailing due to the fast development of hardware and software for computer graphics. These applications can be found in entertainment, education and commerce. In addition to commercial films and TV, computer animation also serves as an effective way to preserve traditional performing arts. For example, Li and Hsu (2007) have designed an authoring system that can create the animations of traditional shadow play by taking high-level inputs from a user. The goal of designing this type of animation system is not only to capture the art in a digital form but also to lower the entry barrier for learning and producing this kind of show. Another example of such animation systems for traditional folk arts is the animation system for traditional Turkish shadow theatre, Karagoz, which is also a type of puppet show (Güdükbay et al., 2000a,b). It uses techniques, such as hierarchical modelling and texture mapping, together with keyframing as a tool for producing animations.

Puppet animation is a special type of character animation. The mainstream methods for generating character animations can be divided into three categories. The first type of methods generates character animation by sampled data. For example, as low-level signals, sampled data can be obtained with the motion capture technologies when the motions are performed by a real actor (Kovar et al., 2002; Witkin and Popovic, 1995). Animations made with this kind of technologies are more plausible. However, without knowing the meaning and structure of the motions, it is more difficult to modify a captured motion to fit the constraints of a new environment. Much research in recent years has been devoted to solve this kind of problem (Gleicher, 1997, 1998; Safonova and Hodgins, 2007). The second way is about using commercial 3D animation software

to set the key frames of a character and interpolate between key frames to generate the final animations. However, even for good animators, producing an animation in this way is usually labour intensive and time consuming. The third way is about generating animations by simulation or planning procedures (Bruderlin and Calvert, 1996; Chen and Li, 2002; Perlin, 1995). It is also called procedural animation or knowledge-based animation since it usually requires a dynamics model of the objects under simulation or motion-specific knowledge to plan and generate the motions. Due to the complexity of the underlying computational model, this type of animations usually is generated in an off-line manner and displayed in real time. Nevertheless, due to the increasing computing power of computers, it is becoming more feasible to generate this kind of animations in real time with appropriate model simplification.

We use the approach of procedural animation to generate the puppet motions in a Taiwanese glove puppet show. We attempt to capture the knowledge of how to manipulate a glove puppet to design animation procedures. In such a procedure, a motion is usually divided into multiple phases delimited by keyframes with distinct spatial constraints in each phase. Then, the motion between keyframes is computed by appropriate interpolation functions capturing temporal or spatial constraints. Animations produced with this approach have the advantages of being flexible and computationally efficient for real-time environments. Nevertheless, it is a great challenge to design a procedure with appropriate parameters that can produce plausible motions.

In the rest of this paper, we will first review the research related to our work in Section 2. In Section 3, we will introduce the animation system that we have designed by describing the system architecture and the puppet models. In Section 4, we will demonstrate the animation system by giving a few examples of puppet motions generated by the system. Finally, we will conclude this paper with possible extension of the work.

2 Related work

Some previous work on animation systems for traditional folk arts was done by build a user interface for editor to set the key frames of a character and interpolate between key frames to generate the animations (Güdükbay et al., 2000b). And most of previous work on traditional Taiwanese glove puppet show was conducted via the motion capture approach. For example, recently the X-Lab at National Chao-Tung University (2008) captured the motion of the hand playing a glove puppet with a data glove (XLab). They also have implemented a system with a user interface allowing user to create a glove puppet show by putting small pieces of motions together. In addition to display the motions with 3D graphics, they have also designed a puppet robot of the same size as the hand puppet that can mimic the captured motion. Shin et al. (2001) adopted an importance-based approach to retarget captured motions to whole-body articulated puppets in real time. In our work, the design goal is different from capturing and reproducing the hand motion of a puppet show or build an animation generate system to generate animation via setting keyframe by editor. Instead, we hope to realise the animation by modelling the way that a hand puppet is manipulated in a procedural manner. By adjusting the parameters of the procedures, we can compose a good variety of flexible puppet animations.

Puppet animations usually consist of two types of motions: intended primary motions and passive secondary motions. For example, the motions for the body, head and hands

are primary motions manipulated by the animator while the motions of the legs, clothes and hair belong to the secondary motions that move passively according to the primary motion, the gravity and the environment. Previous work divided the problem into layers of abstraction and treated the secondary motions at the lower layers. For example, in Dontcheva et al. (2003), the main actions of a character were acquired by the motion capture technique and then the associated secondary motions were computed. Perlin also proposed to add noise to the primary motion as part of the secondary motion to make the generated motion look more realistic (Perlin, 1995).

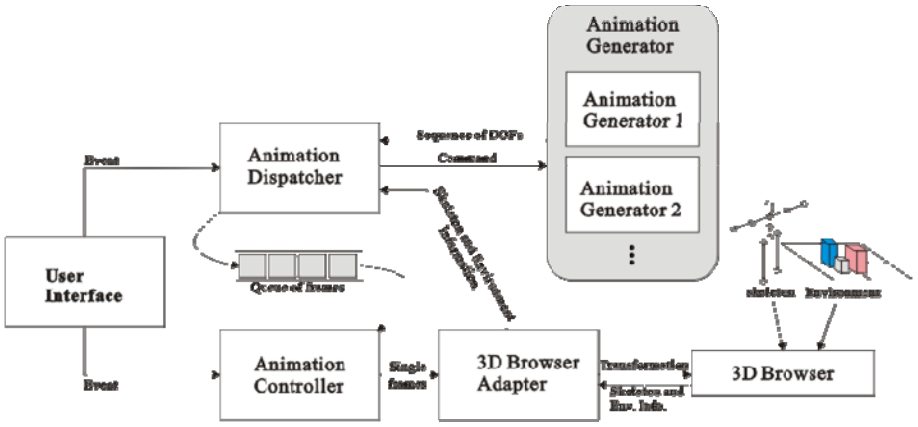
Li and Hsu (2007) also distinguished these two types of motions for producing 2D traditional shadow play animation semi-automatically. The authoring tool allows a user to specify few key postures of the animated character in a 2D environment and then uses a motion planning algorithm to compute the path of primary motions. Secondary motions are then computed according to the obstacle constraints and the gravity force. For glove puppet shows, although the performance is typically viewed from a fixed direction, puppet motions need to be modelled in the 3D space of the performance platform.

3 Design of animation system

3.1 Animation platform

We aim to design a 3D animation system customised for real-time display of glove puppet animation. The system is constructed based on an experimental testbed for procedural animation, proposed in our previous work (blind). The architecture of the system is shown in Figure 1. The system was designed with the model, view and controller (MVC) design pattern. Model is used for encapsulating data, view is used for interaction with the user and controller is used for communication between model and view. In Figure 1, every block represents a module, and the arrows between the modules stands for data flow. In these modules, the animation dispatcher and the animation generator are the kernel of this animation system defining how the animation generation problem is decomposed into elementary procedures implemented in the animation generators. These two components play the role of 'model' in MVC while the animation controller plays the role of 'controller' that allows playback and speed control of the animation. The last two components, 3D browser and user interface, play the role of 'view' that enables interactive display. The 3D browser is treated as a plug-in in this system. The browser adapter module acts as a wrapper of the 3D browser. It encapsulated the 3D browser and provides a uniform interface to other modules for the rest of the system. As a result, the 3D browser can be replaced easily without modifying the rest of the system as long as appropriate browser adapter has been implemented. In the current system, we have adopted the open-source JMonkey package as our 3D browser (jMonkey Engine, 2003).

Figure 1 System architecture of the puppet animation platform (see online version for colours)



3.2 Kinematics model

The kinematics structure of a puppet is similar to a human figure with minor differences as shown in Figure 2. For example, the major active joints of a puppet are at the neck, shoulder, thigh and pelvis. The joints of elbows and knees are usually ignored. There are 17 joints and 54 degrees of freedom (DOF) in total for this model. In a real puppet show, the motions of the legs are triggered either by another hand or by the gravity during swinging. In our current system, the leg motion is part of the animation procedures designed for a specific type of motions. The clothes are the passive part that moves with the legs as they swing. As shown in Figure 2, we attach additional links and joints to the clothes at the front and back of the legs such that we can move the clothes when the legs are lifted. Since the clothes are modelled as a skinning deformable mesh, it deform as it is lifted by the legs.

3.3 Motion model

As mentioned earlier, in procedural animation, different motions require different animation generators. We divide the process of designing an animation procedure into three main steps. In the first step, we observe the motions performed by a master puppeteer and define appropriate motion parameters that can be used to describe the characteristics of the motion. For example, for the walk motion, the adjustable motion parameters include step length, body swing, etc. In the second step, we decompose the motion into several phases by defining key frames separating these phases according to the motion parameters described above. In the last step, we define the procedure for interpolation between the key frames. Two types of interpolations are needed to produce plausible motions. One is to interpolate along a curve for trajectory following while the other defines how a motion is timed. In the current system, linear interpolation is used for simple motions while Bezier curves are used for more sophisticated motions.

To illustrate how the motions are generated in a procedural manner, we use four different types of motions e.g.: walk, run, roll-over and punch, as described in Sections 3.3.1–3.3.4.

3.3.1 Walk

There exist various ways of walking with different meanings under different scenarios. Advised by our master puppeteer, we have divided a normal walk into four phases separated by four key frames. In the walk motion, the adjustable spatial parameters include step length, step height and body swing angle. As shown in Figure 3, we define four key frames according to the given motion parameters. The relationship between the adjustable spatial parameters and each key frame is shown in Figure 4 (Bruderlin and Calvert, 1989). The vertical axis represents the main moveable joints as puppet is manipulated while the horizontal axis represents the time. A joint is denoted with a blue box if the joint has a significant motion in that phase. In this case, the motion of the joint in the phase is generated according to the driving spatial parameter denoted in the blue box. When no parameter is indicated in the blue box, the motion will be only influenced by the timing parameters defined for interpolation.

Figure 2 Kinematics model of the hand puppet including the additional linkages for clothes (see online version for colours)

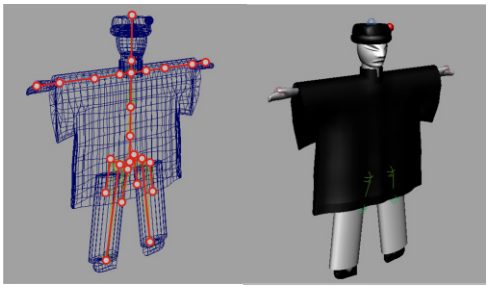
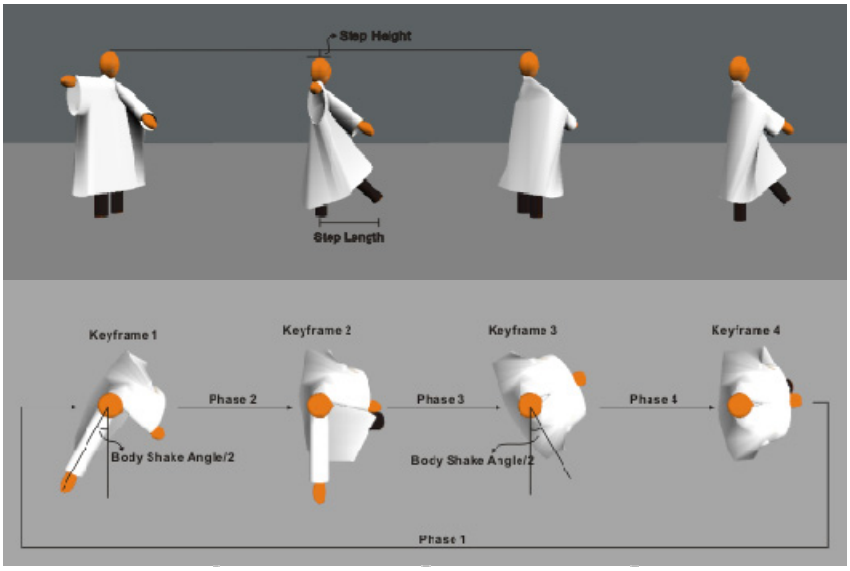


Figure 3 The keyframes and the procedure parameters for the walking motion (see online version for colours)



3.3.2 Run

The key of creating this type of motions by a puppeteer is on a sudden swing of one leg upward and then moving the body up and down to swing both legs further to create the running motion as shown in Figure 5. The swinging leg motions belong to the type of secondary motions since the legs move in accordance with the primary motion of the body. The speed and frequency of the body moving up and down determine the rotational speed of the legs swinging back and forth.

Figure 4 Illustration of how the spatial parameters (the white and the red words) affect the definitions of key frames for each major joint in the walk motion (see online version for colours)

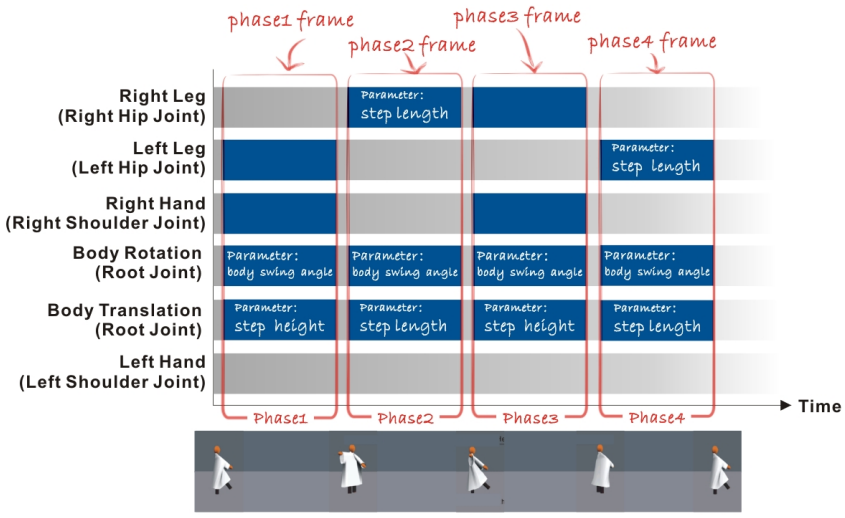
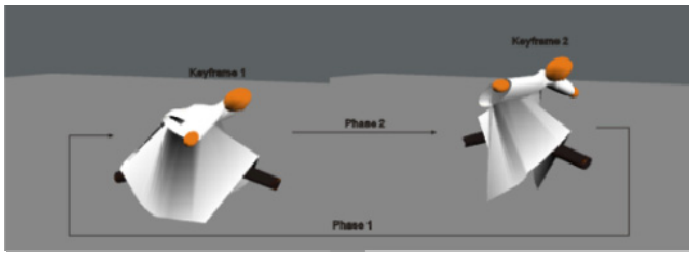


Figure 5 The key frames of the running motion (see online version for colours)



3.3.3 Roll-over

The roll-over motion is specified by two parameters: *rolling height* and *rolling length*. Two key frames are needed to define this motion. A Bezier curve defined with two parameters is used as the roll-over trajectory of the puppet. For a successful roll-over, the puppet must make a complete revolution on the orientation along the trajectory. A snapshot of the generated animation denoted with the procedural parameters for the rollover motion is shown in Figure 6.

3.3.4 Punch

We also use two parameters to define the punch motion: *hold back angle* and *step length*. The hold back angle parameter defines how much the puppet pulls back its hand when preparing to make a punch. The larger the value, the harder the puppet punches. The step length parameter specifies the degree of moving forward after the puppet makes the punch. A snapshot of the generated animation denoted with the procedural parameters for this motion is shown in Figure 7.

Figure 6 A snapshot of the roll-over motion (see online version for colours)

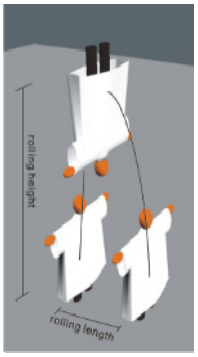
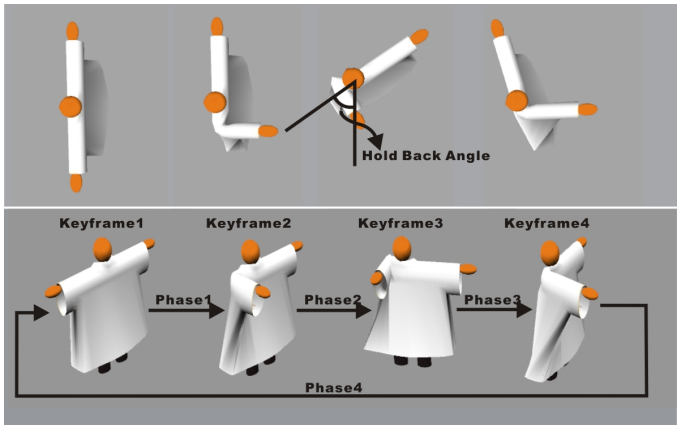


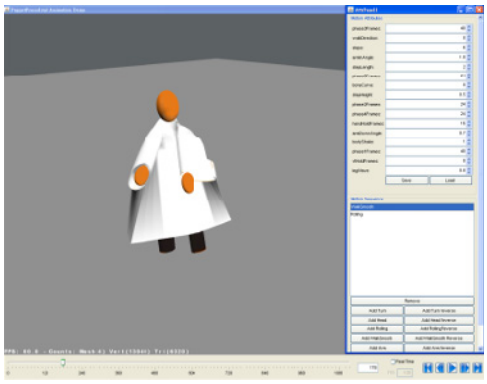
Figure 7 The keyframes and the procedure parameters for the punching motion (see online version for colours)



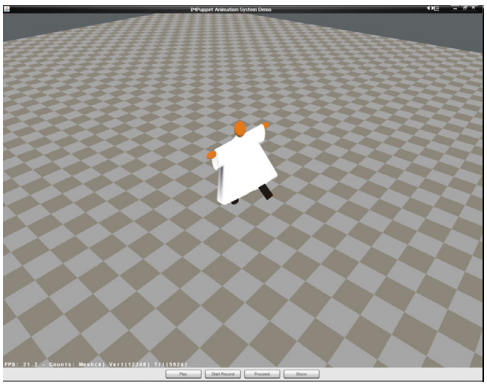
4 Experimental results

We have implemented a java-based animation system that allows a user to design the motions in a glove puppet show with an interactive graphical interface. Currently, there are two different modes in our system: the editor mode and the director mode. The graphical user interfaces for both modes are shown in Figure 8. In the editor mode (Figure 8(a)), at the upper left corner, the system hosts a 3D browser to display the animations generated by the user-designed procedures. The user can compile a sequence of motions (such as walk, run, roll-over and punch) by selecting them and setting their parameters from the graphical user interface on the right. The play control that allows a user to view the animation continuously or frame-by-frame is at the lower portion of the interface. In the editor mode, we can edit the result produced in the director mode described below. The user can select a specific motion and tune the motion parameters one by one in order to make the style of the motion fit the character.

Figure 8 Graphical user interface for interactive animation generation of puppet show (see online version for colours)



(a)



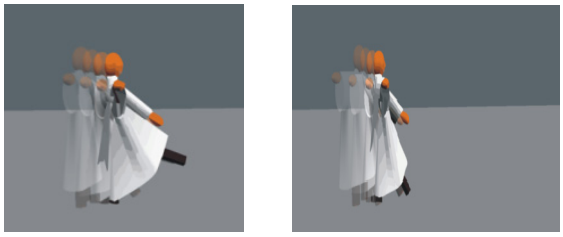
(b)

Note: Editor mode in the left and right is the director mode.

The graphical user interface for the director mode is depicted in Figure 8(b). In this mode, user just act as a director who can command how an actor behaves. The lower part is the control buttons that allow user to decide when to start or stop recording. In this mode, user needs to assign keys on the keyboard that correspond to the motion parameters for a given motion before the system starts up. Then, the user can use the keyboard to control the motion of the puppet at run time. When the start recording button is pressed, the system records all the motions that puppet perform under user direction until the user call a stop. Then, the user can use the editor mode to load and play the motions for further editing.

To illustrate that the motions generated by our animation procedures are rather flexible, we have used different sets of parameters to generate animations. In Figure 9(a), we show the walk motion generated with parameters of (swing_angle, step_length) = (0.3, 1.5) while the parameters are (1, 0.8) for Figure 9(b). In Figure 9(c), we show a run motion generated with the parameters of (step_length, step_height) = (0.5, 2) while the parameters are (1.5, 3) for Figure 9(d). In Figure 9(e), we show a roll-over motion generated with the parameters of (rolling_height, rolling_length) = (10, 5) while the parameters are (20, 10) for Figure 9(f). In Figure 10, we show a composite example generated by composing multiple motions into a final show. The puppet ran and then rolled over to make a complete revolution. The motions generated on the platform can be further exported into a MEL script for further rendering in animation packages such as Maya. An example of the motion rendered in Maya is shown in Figure 11.

Figure 9 Using different sets of motion parameters for walking (a and b), running (c and d) and rolling-over (e and f) (see online version for colours)



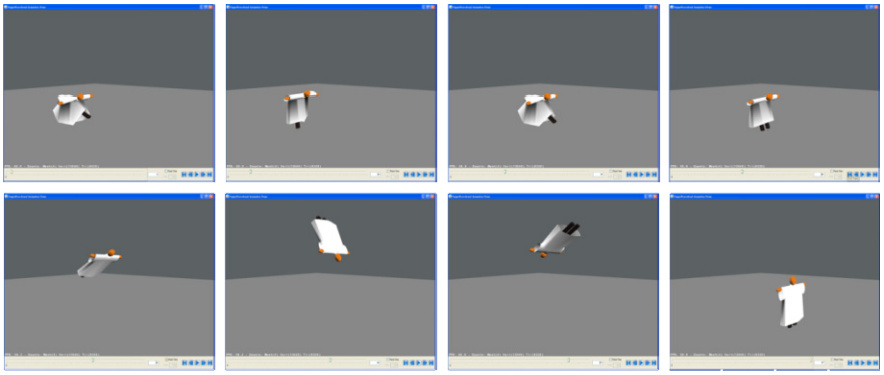
(a) (b)

(c) (d)

Figure 9 Using different sets of motion parameters for walking (a and b), running (c and d) and rolling-over (e and f) (see online version for colours) (continued)

(e) (f)

Figure 10 An example of compiling different motions together (see online version for colours)



Note: The puppet entered with a running motion and then did a roll-over motion at an end.

Figure 11 An example of exported animation rendered in Maya (see online version for colours)

5 Conclusion and future work

With the help of computer animation techniques, we hope to preserve and promote the art of glove puppet show, a precious culture heritage in Taiwan. We have made a first attempt to model the basic motions of a glove puppet by observing the performance of an expert puppeteer and implementing the motions with procedural animation. We also provide two modes and their interactive graphical user interfaces for users to author the animations. We believe that the system provides an easy way for one to create animations for a puppet show.

These animation procedures designed in this system will work as reusable units for composing a more complete play in the future. In the current implementation, the way we link different motions together is by making the standing pose as the transition pose after each motion is performed. We are currently implementing motion blending procedures to provide a smooth transition between any two motions. Furthermore, we hope to design an animation scripting language that can be used to document the motions of glove puppets as well as the screenplay. In this way, it will become possible for us to generate a puppet play automatically according to a given story plot. We also hope that the music behind the script can be generated automatically according to the tempo of the show and synchronised with the cues from the puppeteer.

References

Bruderlin, A. and Calvert, T.W. (1989) 'Goal-directed dynamic animation of human walking', *ACM Computer Graphics (Proceedings of SIGGRAPH'89)*, Vol. 23, No. 3, pp.233–242.

Bruderlin, A. and Calvert, T.W. (1996) 'Knowledge-driven, interactive animation of human running', *Proceedings of the Conference on Graphics Interface '96*, pp.213–221.

Chen, P.F. and Li, T.Y. (2002) 'Generating humanoid lower-body motions with real-time planning', *Proceedings of 2002 Computer Graphics Workshop*, Taiwan.

Dontcheva, M., Yngve, G. and Popović, Z. (2003) 'Layered acting for character animation', *ACM Transactions on Graphics*, Vol. 22, No. 3, pp.409–416.

Gleicher, M. (1997) 'Motion editing with spacetime constraints', *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, pp.139–148.

Gleicher, M. (1998) 'Retargeting motion to new characters', *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pp.33–42.

Güdükbay, U., Erol, F. and Erdogan, N. (2000a) 'Beyond tradition and modernity: digital shadow theater', *Leonardo*, Vol. 33, No. 4, pp.264–265.

Güdükbay, U., Erol, F. and Erdogan, N. (2000b) 'Tradition offers artistic possibilities for new media technologies: an animation system for shadow theatre', *Catalogue of the Tenth International Symposium on Electronic Art-ISEA'2000*, Forum des Images, Paris, France, p.56.

jMonkey Engine (2003) Available at: http://www.jmonkeyengine.com/.

Kovar, L., Gleicher, M. and Pighin, F. (2002) 'Motion graphs', *ACM Transactions on Graphics*, Vol. 21, pp.473–482.

Li, T.Y. and Hsu, S.W. (2007) 'An authoring tool for generating shadow play animations with motion planning techniques', *Int. J. Innovative Computing, Information, and Control*, Vol. 3, pp.1601–1612.

Perlin, K. (1995) 'Real time responsive animation with personality', *IEEE Transactions on Visualization and Computer Graphics*, Vol. 1, pp.5–15.

Safonova, A. and Hodgins, J. (2007) 'Construction and optimal search of interpolated motion graphs', *ACM Transactions on Graphics (TOG)*, Vol. 26, No. 3, pp.106–116.

Shin, H.J., Lee, J., Shin, S.Y. and Gleicher, M. (2001) 'Computer puppetry: an importance-based approach', *ACM Transactions on Graphics (TOG)*, Vol. 20, No. 2, pp.67–94.

Witkin, A. and Popovic, Z. (1995) 'Motion warping', *Proceedings of ACM SIGGRAPH '95*, pp.105–108.

X-Lab of National Chao-Tung University (2008) Available at: http://xlab.cn.nctu.edu.tw/modules/x_movie/x_movie_view.php?cid=1&lid=35.