IMHAP – An Experimental Platform for Humanoid Procedural Animation

Chang-Hung Liang
Computer Science Dept., National Chengchi Univ., Taipei, Taiwan
s9204@cs.nccu.edu.tw

Pai-Cheng Tao
Computer Science Dept., National Chengchi Univ., Taipei, Taiwan
g9310@cs.nccu.edu.tw

Tsai-Yen Li
Computer Science Dept., National Chengchi Univ., Taipei, Taiwan
li@nccu.edu.tw

Abstract

Humanoid animation is one of the most common but complicated types of animation. Although more sophisticated animation packages have been developed in recent years, there is no extensible platform allowing researchers to test developed procedures for experimental purposes. In this paper, we preset an experimental platform called IMHAP (Intelligent Media Lab's Humanoid Animation Platform) *especially for humanoid animation generated with a procedural approach. IMHAP is constructed with a model-view-control (MVC) methodology and features flexibility and extensibility. In addition to present the architecture of the system, we also use a few lower-body motions generated with a procedural approach to demonstrate the use of the system as an experimental testbed of humanoid animation.*

Keywords: experimental animation platform, procedural animation, computer animation

1. Introduction

The generation of human motions has been one of the oldest research topics in computer animation that have attracted much attention in the past. However, human figure animation is also the most challenging one because we are most familiar with this form of animation. In the literature of computer animation, there have been many researches focusing on generating human motions automatically [1][2][3][6]. According to a survey in [5] on human walking animation by Multon et al., the study of generating human motions can be classified into three categories: *procedural methods*, *dynamics*, and *motion editing*. The most popular approach in recent years is the motion capture technique, which belongs to the last category. Most researchers in this category strive to develop methods to make the captured motions adaptable to the environments that are different from the one where the motions were captured.

Despite this popular approach can produce more natural motions, many humanoid animations are still created with the key-frame-based approach or the procedural approach. Procedural methods were the earliest approach developed in the 1980s. For example, Bruderlin and Calvert [1] designed a procedure embedded with empirical and physics knowledge to generate the animation of human running. Chen and Li [2] presented a high-level control system for human walking by the use of procedural animation and motion planning. Faloutsos and van de Panne [3] proposed a system that can be used to compose multiple dynamics-based animation controllers, which can be treated as animation procedures. In another work by van de Panne [6], an algorithm is designed to compute the animation for human walking by taking footprints and timing information as the inputs. He used a physics-based approach to optimize the trajectory of the figure body while using procedural methods to generate limb motions.

In order to develop good animation procedures, researchers often need to develop extensive programs to perform desired experiments. Although there are many commercial 3D software packages available in the market, none of them are suitable to be used as an experimental testbed for real-time animation. Some of these packages such as Quest3D [9] and Virtools [10] are real-time 3D packages that can produce excellent quality of real-time animation. However, in addition to affordability, most of them lack easy extensibility and are usually too heavy for the purpose of performing animation experiments. Many of the main-stream 3D software packages such as 3ds Max and Maya [7], only have limited import and export abilities and cannot be incorporated into an experimental platform for real-time animation. Consequently, most researchers still

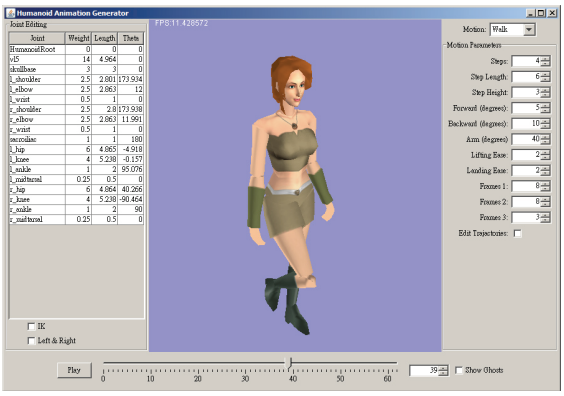

Figure 1. The snapshot of an example application based on IMHAP

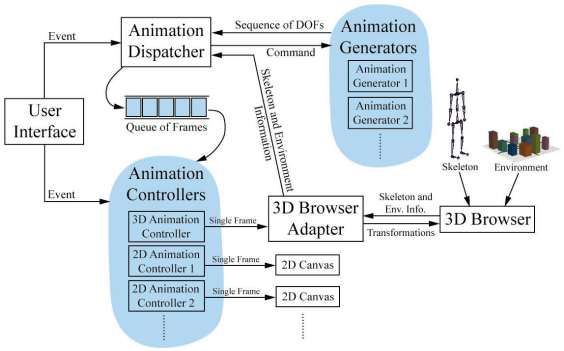

Figure 2. The system architecture of IMHAP

need to perform their experiments on a platform created and maintained by themselves.

A typical system for computer animation research usually consists of various modules such as 3D or 2D displays, animation control, parameter inputs, and animation generation. Routines for these modules usually are very similar and desirable to be reused. However, our experiences show that the goal of reusability cannot be achieved without an extensible design and good software engineering practice.

In this paper, we propose to design an experimental platform called IMHAP (Intelligent Media Lab's Humanoid Animation Platform) that can be used to perform real-time procedural animation for human characters. A snapshot of the system used to test a human walking procedure is shown in Figure 1. The proposal system is designed to be self-contained and with reusability and extensibility in mind as described in the next section.

2. System Design

The architecture of IMHAP, as shown in Figure 2, is based on the MVC pattern. In MVC, a system is divided into three components: *Model*, *View*, and *Controller*. The model encapsulates the raw data, the view interacts with the user, and the controller responds to the user's actions and invokes changes on the model. The modules of animation dispatcher and animation generator are the main components in IMHAP, and play the role of "model" in MVC. The module of animation controller plays the role of controller, receives events from the user, and controls how the animation is played. Finally, the modules of user interface, 3D browser, and 2D canvas play the roles of "view" in MVC. They are responsible for interacting with the user and drawing the scene on the screen. The function of each module will be described in more details below.

The 3D browser is treated as a plug-in in IMHAP. We make no assumption on what kind of 3D browser being used, except for that it needs to be able to read 3D models from files and provide External Authoring Interface (EAI) to other modules. EAI is a programming interface defined in most VRML browsers [11] that allows a programmer to retrieve and set the attributes of an arbitrary object or skeleton joint. Although EAI is a common standard for external interface of VRML browsers, not all 3D browsers support exactly this interface. For those browsers that support an interface with different naming or argument conventions, we supply another module called "browser adapter" to act as a wrapper of the 3D browser. It encapsulates 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.

The modules of animation dispatcher and animation generator are the core of our system. They are defined with abstract interfaces and implemented by the designer to perform experiments on animation procedures. The animation dispatcher can be used as a global planner which directs one or more animation generators. The animation dispatcher receives high-level commands, such as a path along which the human character should follow, from the graphical user interface provided by the system. Then, the animation dispatcher maps the high-level commands into a sequence of various types of locomotion according to the kinematics and geometric model of the human character as well as the environment. The animation dispatcher defines a local motion generation problem for each clip of locomotion and then dispatches them to the appropriate animation generators. An animation generator could be a local planner in charge of generat-

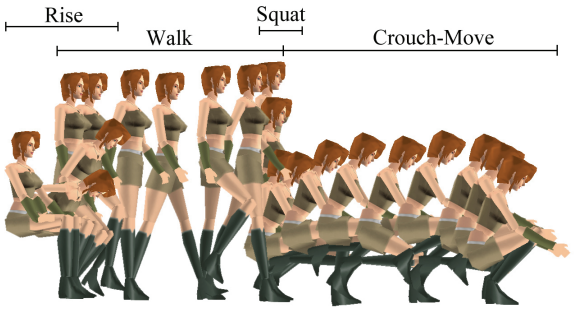

Figure 3. A sequence of rising, walking, squatting, and crouch-moving

ing collision-free motions with smaller number of degrees of freedom (DOFs) or a motion editor acting on motion capture data. The generated motions are returned to the animation dispatcher for composing a sequence of animation frames that will be appended to the animation queue of the animation controller.

The module of animation controller reads the queue of frames generated by the animation dispatcher and controls the playing of animation. The user controls the animation by accessing the given user interface, and the user interface then calls the animation controller to invoke proper actions. For example, in Figure 1, a slider bar at the bottom of the screenshot is used to control the playing of animation in a frame-by-frame or real-time mode. The animation controller then calls the 2D canvas and the browser adapter simultaneously to display images in 2D or 3D, respectively. There could be more than one controller used in an application. For example, a separated 2D animation controller can be implemented to provide orthogonal views such as the top view, side view, and even the chart graph while another 3D animation controller may be used to create perspective view at a frame different from the 2D view.

3. Experiment

The IMHAP system has been implemented with the Java Language and a screenshot is shown in Figure 1. To test IMHAP, we have also implemented a sample application system that can generate four kinds of locomotion: rising, squatting, walking, and crouch-moving. In this application, we can experiment the implemented procedures by selecting the desired motion and adjusting its parameters to control the style of the motion. For example, in Figure 1, we select "walk" and set its parameters such as step length and step height in the right panel, and edit the kinematics

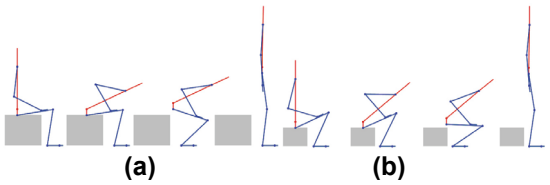

Figure 4. Two different rising motions from different heights of seats

(skeleton) settings in the left panel and then the walking motion can be generated and displayed on the 3D browser.

We evaluated the performance with on a Pentium 2.4GHz PC with 1GB RAM. All the motions can be generated in 100 ms. Figure 3 shows a sequence of the four available motions (rising, squatting, walking, and crouch-moving) generated by the system. The sequence of motions is specified by the user via the graphical user interface, and the animation dispatcher was invoked to direct the corresponding animation generator for each motion to generate the desired animations. Taking the geometric and kinematics model of the human figure as an input, an animation generator may reduce the model into a simpler one with partial degrees of freedom (DOF) only. For example, some motions can be produced by projecting the human model into a 2D plane and then only generate the motion on the main DOF's. The remaining DOF's can then be recovered in a post-processing step. The animation dispatcher then collects the sequences of motion clips generated by the animation generators and merges them into one single queue of frames for the animation controller to display.

All motions shown in Figure 3 were generated in a procedural manner. The skeleton we used follows the standard of H-Anim [8]. To simplify the problem, we defined the motions on the sagittal plane of the humanoid model such that each joint has only one degree of freedom. The procedures were composed of a few parameterized keyframes and an interpolation method for each motion phase between two keyframes. For example, we defined two parameters: chair height and sitting depth for the sitting or the reversed rising motion. Then, the keyframes are computed with a simple inverse kinematics routine according to these two parameters. An ease-in and ease-out interpolation method is also chosen to generate the frames between two keyframes. Figure 4 shows the keyframes of rising motions generated with two different sets of parameters. Similarly, in Figure 5, we show two different crouch-move motions with different heights for the center of mass of the body.

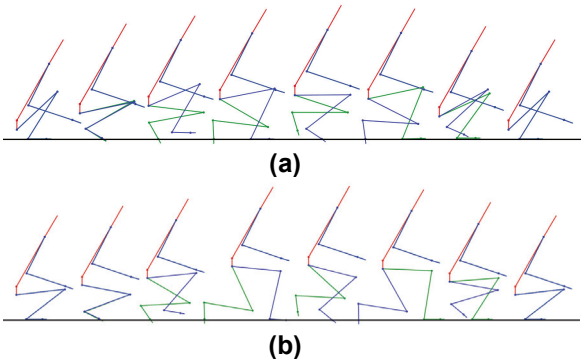

Figure 5. Two different crouch-move motions with different heights

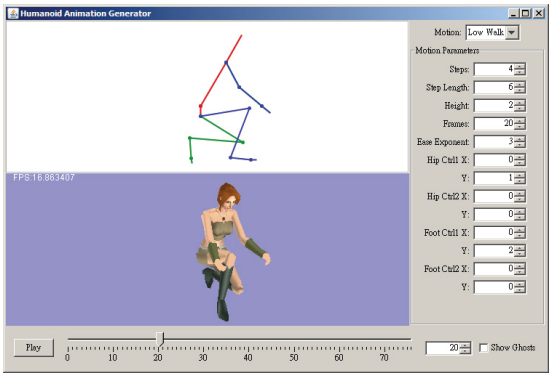

Figure 6. An example application layout containing a 2D and a 3D view

On IMHAP, we are allowed to perform experiments with the implemented animation procedures and show the animation results in real time by displaying perspective and orthogonal views at the same time. An example application layout consisting of two views are shown in Figure 6. The orthogonal side view at the top was implemented with the Java2D API. The perspective view at the bottom is an extensible 3D browser, called IMBrowser [4], designed and implemented in our previous work based on the Java 3D API. In our experiments, IMBrowser reads a humanoid model in the VRML format, and provides an EAI for other modules in IMHAP to access the attributes of each animation nodes.

4. Conclusion and Discussion

In this paper, we have presented an animation platform for experimental purpose. As illustrated in Figure 2, IMHAP is flexible and extensible in the several aspects. First, because of the design of browser adapter, the 3D browser is somewhat independent of IMHAP and thus can be changed to other browsers as long as they provide an EAI and a customized adapter. Second, the modules of animation dispatcher and animation generator usually play the roles of generating the desired animations with the procedures under experiments. Developers can customize their behaviors by implementing the required abstract interfaces to be integrated with other modules. Third, the animation controller is also extensible. We can add and connect as many animation generators and animation controllers as needed in an application. Multiple animation controllers allow us to have multiple views of the animations at different pace in the user interface. Fourth, multiple views of different types are supported on the same graphical user interface. As long as the display canvases register themselves as a listener to the animation controllers, the animations on different types of canvases can be synchronized. Thus, various visualization tools such as 3D display, 2D views, statistic graph can all be incorporated into the same interface.

IMHAP is not a standalone application but an animation platform allowing us to experiment developed animation procedures on it. More improvements such as configurable visual components and connections to commercial 3D packages are under development. We believe that the flexibility and extensibility of IMHAP will help us develop procedural animation for human characters in a more efficient and effective way.

References

[1] A. Bruderlin and T. W. Calvert. Knowedge-Driven, Interactive Animation of Human Running. *Graphics Interface 1996*, pp. 213-221, 1996.

[2] P. F. Chen and T. Y. Li. Generating Humanoid Lower-Body Motions with Real-time Planning. *Proceedings of 2002 Computer Graphics Workshop*, Taiwan, 2002.

[3] P. Faloutsos, M. van de Panne, and D. Terzopoulos. Composable Controllers for Physics-Based Character Animation. *Proceedings of ACM SIGGRAPH*, 2001.

[4] T.Y. Li, M.Y. Liao, and P.C. Tao, "IMNET: An Experimental Testbed for Extensible Multi-user Virtual Environment Systems," ICCSA 2005, LNCS 3480, O. Gervasi et al. (Eds.), pp. 957–966, 2005.

[5] F. Multon, L. France, M-P. Cani-Gascuel, and G. Debunne. Computer Animation of Human Walking: a Survey. *Journal of Visualization and Computer Animation*, 10:39-54, 1999.

[6] M. van de Panne, From Footprints to Animation. *Computer Graphics Forum*, 16(4): 211-223, Oct. 1997.

[7] Autodesk, http://www.autodesk.com

[8] Humanoid Animation Working Group (H-Anim), http://www.h-anim.org

[9] Quest3D, http://www.quest3d.com

[10] Virtools, http://www.virtools.com

[11] VRML, http://www.web3d.org/x3d/specifications/vrml/