Animation in a Demonstrational Interface Builder

Gene L. Fisher, Hsin-Hui Sung, Suzanne H. Nguyen, Thang D. Nguyen
Department of Computer Science
California Polytechnic State University
San Luis Obispo, CA 93407

ABSTRACT

This paper describes the integration of animation capabilities into a demonstrational interface builder. Basic interface behavior is defined using a technique called stimulus-response demonstration. To support animation, a number of new primitives have been added to specify paths, animated sprites, and timing control. These new capabilities have been smoothly integrated with the stimulus/response mechanism to provide high-level animation control. This integration permits animations to be manipulated in a number of useful and interesting ways.

KEYWORDS: Animation, Demonstrational Interfaces, Interface Builders, User Interface Management Systems

BACKGROUND

The DEMO interface builder [2,4] allows a developer to define interface behaviors using a techniqued called *stimulus-response demonstration*. The interface development begins by drawing a picture of an interface, using a basic drawing editor. Once the interface appearance is drawn, the developer demonstrates a stimulus that represents an action that an end user will perform. After the stimulus, the developer demonstrates the response(s) that should result from the given stimulus. DEMO makes generalizing inferences from the demonstrations and implements the demonstrated behavior.

Simple animations can be defined in the original DEMO system. In [4] for example, a "rabbit control" animation was illustrated (see Figure 1). Three animated behaviors are defined for the rabbit: Jump Forward, Jump Backward, and Play Dead. The basic animation scene contains animation control buttons and a picture of the animated rabbit. These are drawn with a graphic editor that is built-in to DEMO. To define an animated behavior, say 'Jump Forward',the following steps are performed: (1) Define a stimulus as the pressing of 'Jump Forward'. (2) Define the response as a sequence of rabbit move operations (see Figure 1).

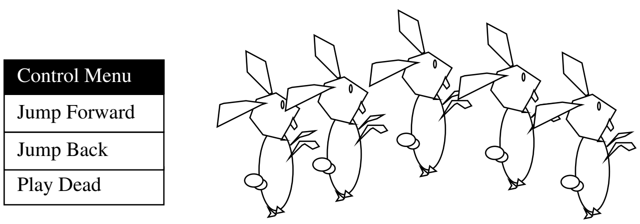

Figure 1: Rabbit Control Animation

When the interface is operated by an end-user, pressing the 'Jump Forawd' button replays the sequence of responses, thus producing the animation.

There are a number of significant weaknesses in defining animations using only the stimulus/response mechanism:

- *No animation speed control* -- The system simply replays the response as quickly as it can.
- *No interpolation between response frames* -- The system simply draws each frame at its new position, which leads to discontinuous movement in many cases.
- *No complex interactions* -- The responses of one animation cannot easily trigger other animated actions.

These points are addressed in the remainder of the paper.

ANIMATION CREATION AND CONTROL

The first step to improving DEMO's animation capabilities is the addition of basic operations to create and control animations. These operations are added at the level of the graphic drawing functions that are already part of DEMO. The animation operations are based on a simple path/sprite model. A sprite is a graphic object to be animated, and a path is the (B-spline) curve over which the sprite can travel. A number of animation control functions are available. These include starting, stopping, and resuming; speed controls, on full paths or individual path segments; keyframes, to specify sprite transformations at key points on a path.

Using the new animation operations, an enhanced 'Jump Forward' rabbit animation can be created as follows: (1) Draw the animated rabbit. (2) Draw the path on which the rabbit will jump. (3) Add any special effects, such as speed controls and/or rabbit transformations along the path.

Note that at this level, the stimulus/response mechanism is not yet needed. Rather, basic animation is now a primitive of the system. To run a new animation, the user selects the built-in start/stop animation controls.

ANIMATIONS AND DEMONSTRATION

Using the new animation functionality, users may draw scenes that contain both static graphic objects, as well as animated objects and their paths. All animation-related operations may be used as stimuli and/or responses. Four cases are of interest: animations as responses, animations as stimuli, cyclic animations, and animations as gestures.

Animations as Responses

Consider again the complete rabbit control interface shown in Figure 1. In addition to the animations, there are the three control buttons. Using the new animation primitives, specifying the behavior of the control buttons is a matter of defining a response to be the start of an animation. Specifically, the full rabbit control interface is now defined as follows: (1) Draw the complete scene, including rabbit animations and control buttons. (2) Define a stimulus to be pressing a button. (3) Define the response to be starting the appropriate animation.

Animations as Stimuli

Another useful way in which animations can integrate with the stimulus/response mechanism is to allow an animated sprite to gesture a stimulus. This can be accomplished by defining a particular point on an animation path to be a "stimulation point". When a moving sprite crosses such a point, it triggers a stimulus. The response to such a stimulus can be defined in the normal way within DEMO.

Suppose, for example, the user would like the jumping rabbit to frighten a bird in a nearby tree. Such an animation could be constructed as follows: (1) Add a flying bird animation to the existing scene. (2) Define a point on the rabbit animation, say the apex of its path, to be a stimulus point. (3) Define the response to be starting the flying bird animation.

Cyclic Animations

The combination of animations as stimuli and responses can be used to create cyclic (i.e., continuous) animations. Suppose, for example, the user wants the rabbit to run around a particular path indefinitely. To do so, the user can draw a closed animation path and define the end point of the path to be a stimulus to resume the same animation. Cyclic behavior can also be defined indirectly. That is, the end of one animation can stimulate another animation, which can in turn stimulate the original animation to restart at some point in the second animation.

Cyclic animations can also be "stationary". A stationary animation is one with a degenerate path drawn as a single point, but with a number of key frames that change the shape and/or rotation angle of the sprite. Points in a stationary animation are points in (real) time as opposed to points in space. Considering the rabbit once again, suppose the user would like it to stand still, but to move its ears at regular intervals. To do so, the user would perform the following steps: (1) Define a stationary animation, consisting of the rabbit with a number of different ear shapes. (2) Define a pausing time interval at the end of the stationary animation. (3) Define the end of the stationary animation as a stimulus to restart itself.

Animations as Gestures

An animation can be used to represent the movement that is gestured by a human user when the user performs some operation on the screen. For example, when the user performs a stretching operation on some graphic object, she selects the object, and then moves the mouse in a horizontal or vertical direction to perform the stretch. Using a gesturing animation, the motion required for the stretch can be that of the animation itself. Specifically, to define a gesturing animation, the user: (1) defines an animation; (2) specifies the animation itself to be a stimulus; (3) specifies the response to be some operation and the object to which the operation will apply.

Consider the rabbit once again. Suppose the user wants to animate the rabbit tugging on the stretching ears of a companion rabbit as it jumps. This animation can be defined by using the jumping animation to stimulate a stretching operation on the companion rabbit's ears.

Gesturing animations can be used for less frivolous tasks than jumping rabbits. A particularly useful application is tutorial animations that explain how to use an interface. In such applications, the developer defines a sprite in the shape of a normal mouse pointer, and animates the mouse to show how sample operations are performed.

RELATED WORK

Integration of animation and interface construction is discussed in [1,3,5]. In [1] and [6], tools are provided for animation effects that are more advanced than in DEMO. The advantage of the DEMO animation system is the smooth integration of the demonstrational capabilities with animation tools. In [5], an alternative approach to the integration of animation in DEMO is described. In their approach, the authors focus on inferring animated behavior, without providing more sophisticated built-in animation tools.

REFERENCES

1. C. Bay-Wei and D. Ungar, "Animation: From Cartoons to the User Interface", 6th UIST, Nov, 1993.

2. G. Fisher, et al., "Adding Rule-Based Reasoning to a Demonstrational Interface Builder", 5th UIST, Nov, 1992.

3. S. Hudson and J. Stasko, "Animation Support in a User Interface Toolkit: Flexible, Robust and Reusable Abstractions", 6th UIST, Nov, 1993.

4. D. Wolber, "Developing User Interfaces by Stimulus Response Demonstration", PhD Thesis, U.C. Davis, 1991.

5. D. Wolber, et al., Creating Real-Time Animated Interfaces with Stimulus-Response Demonstration", CHI Proceedings (Demonstrations), Apr, 1995.