Tangible Manipulation of Temporal Variables

ABSTRACT
This paper presents a tangible interface for manipulating time-varying parameters. These parameters may, for example, represent properties of objects or characters in an animation. The primary challenge in this problem is providing an intuitive way to control both the independent variable (time) and the dependent variables. Our system gives the user control over the position, velocity, and scale of the time axis and provides methods for spatially manipulating dependent variables at each point along the time axis. For one-dimensional variables, we use a tangible spline editor. For higher dimensionalities, we use cameras to track the position of a spatial pointer.

Author Keywords
Tangible interfaces, temporal interfaces, time-varying parameters, variable resolution control

INTRODUCTION
This paper presents a tangible interface to time. Tangible interfaces are naturally suited to spatial manipulation, but using space to represent time is less straightforward. The manipulation of time brings forward issues that occur throughout interface design (even interfaces that are neither tangible nor temporal), such as resolution, precision, granularity, and dimensionality. We discuss and attempt to address these issues in the context of a tangible interface for the manipulation of time-varying parameters.

Many existing non-tangible interfaces provide spatial representations of time. For example, video editing programs use a horizontal axis to represent time, allowing the user to place video clips along the timeline. However, these systems lack the benefits of a tangible interface. A temporal interface presents issues (such as the trade-off between fast and fine motion) that can be more effectively solved by taking advantage of the user’s experience with tactile manipulation.

This paper explores the design space of systems for manipulating time-varying parameters and presents one design that addresses several of the corresponding issues (Figure 1). Our design generalizes to any domain with a temporal axis, such as music, video, transportation, and manufacturing.

One concrete example of an application for our system is animation. This domain is challenging because it involves a large number of parameters that must be precisely manipulated by the user, including both the pose and appearance of various animated objects. Animation is well-suited to a tangible interface because many of the parameters of an animation system are spatial and users of an animation system are often artists who excel at working in real-world space.

Figure 1. Our tangible interface for manipulating time-varying parameters provides a simple interface to the complex issue of temporal scale.

RELATED WORK
Several tangible systems record and play back time-varying configurations. Curlybot [1] is a wheeled tabletop device that captures and replays motions. Topobo [2] extends this to an actuated system of building blocks that can record and playback complex kinematic motions. Both systems feature coincident input/output and provide some ways to modify the recorded motion, but they are intended for education, not for detailed editing of general time-varying parameters.

Other tangible systems are used to guide the motion of virtual objects. The Sympathetic Interface [3] uses an instrumented plush toy as an input device for computer generated characters. Popovic et al. [4] provides a system that allows the user to move a physical object to act out a motion that is the input for a physics-based simulation. Like Curlybot and Topobo, these systems use the real-world passage of time to...
move along a temporal axis, which is quite intuitive, but not amenable to editing of temporal data.

Some interfaces aim to provide flexible movement through time. Dontcheva et al. [5] present a system that allows the user to edit a 3D computer animation using a visually tracked tangible device. In this system, GUI controls allow the user to select a time interval and time scale to obtain detailed manipulation. The AnimaatioKone system [6] allows users to create clay animations with functionality for adding and removing frames at various positions along the timeline. Our design has some elements in common with these systems, but provides a completely tangible interface to variable-resolution temporal movement and allows editing of generalized parameters.

DESIGN SPACE FOR TEMPORAL PARAMETER EDITING

Movement through time

Many consumer devices provide basic interfaces for moving through time. Music and video players provide control over temporal velocity in discrete modes: play, fast forward, rewind, possibly with multiple settings for fast and slow motion. For alarm clocks, the content is time itself and the interfaces are quite direct: move forward by a minute or hour.

One problem with devices that move through time at a limited set of velocities is that they do not adapt to the resolution of the task. The user may want to jump across large spans of time, then narrow in on a particular moment. Other systems are more flexible. The knob on a mechanical watch can be turned very quickly or very slowly. A jog-shuttle dial lets the user set the temporal velocity (e.g. for fast-forwarding video) over a continuous range of values.

Non-tangible software systems often use a timeline slider to represent time as a spatial continuum, which is sometimes awkward. To obtain fine-scale positioning, the user typically must zoom in on the temporal axis (or use limited discrete controls such as stepping forward one frame at a time). Zooming the temporal axis can be disorienting, even with clear conventions about the zoom centering (typically around the current time marker). Our system addresses these issues through a tangible representation of temporal scale described in later sections.

Kinds of parameters

Parameter dimensionality influences the mechanisms that can be used in a time-varying parameter editor. One dimensional variables can be spatially manipulated along a single axis. Two dimensional variables, such as spatial position or velocity, are best manipulated in a two-dimensional space, rather than as two separate one-dimensional parameters.

Some types of parameters are suited to specialized input mechanisms. An orientation parameter should have a cyclic interface. Color manipulation may be easier using interfaces other than three scalar (RGB or HSV) values. Also, the system may need to handle discrete variables (binary, integer, or text) that change over time, but do not allow continuous interpolation.

Our design, described in the next section, provides methods of manipulating a variety of different parameter modalities and dimensionailities. This design currently focuses on continuous numeric or spatial values, not discrete variables.

OUR DESIGN

This section describes our proposed system design. The implementation section describes how our prototype differs from this proposed design.

We aim to achieve a variety design principles. The interface should make use of existing behaviors and conventions in order to reduce training. The system should allow two-handed interaction for increased efficiency. Input and output should be concurrent and the physical state of the interface should maintain consistency with the virtual state, regardless of how the user interacts with the system.

Figure 2. This figure gives an overhead view of our tabletop interface. The system uses two tangible tokens on a single slider to select a temporal interval (a). This interval is mapped onto another physical slider that is used to select a particular time (red line) (b). A 1D parameter is edited using a set of vertical sliders (c). A 2D parameter is edited by placing keyframes using a physical pointer (d). Nearby keyframes are also displayed (yellow squares). The color elements in this figure are projected onto the table surface.

Timeline slider

The first part of our timeline slider is a temporal interval selector. As shown in Figure 2(a), this device consists of two physical tokens moving along a single physical axis, which is marked with the timing of the current dataset. If the user touches both tokens, they can be moved independently. If the user slides just one token, the other follows, such that the temporal interval maintains the same length.
The temporal interval provides the time range for the second part of the timeline slider, which consists of a single physical token, moving along another physical axis, as shown in Figure 2(b). This token is used to select a particular time within the temporal interval, which becomes the current time for the rest of the system. The token is actuated such that it moves along the temporal interval on playback. Also, when the temporal interval is changed, the token is automatically moved to maintain the same current time (so long as the current time remains within the interval).

This two-part timeline slider provides an intuitive way to control temporal movement. Each component separately follows straightforward conventions for selecting intervals and points along an axis. Working together, the two components provide a simple way to change the temporal resolution: the interval selector provides a direct correspondence between temporal scale and physical scale. This allows both coarse and fine control over the time axis.

The timeline slider can optionally be augmented with a jog-shuttle dial. The dial’s default state corresponds to zero velocity (playback is stopped). Rotating the dial clockwise increases the velocity in a continuum from very slow playback to very fast playback. Rotating the dial counter-clockwise results in reverse playback on the same continuum of speeds. These temporal velocities result in physical motion of the current-time selector.

**Scalar parameter editor**

The scalar parameter editor is used to view and manipulate one-dimensional time-varying values. The device is a tangible instantiation of a graph of the parameter, where the horizontal axis represents time and the vertical axis represents the parameter value.

The parameter graph is actuated and manipulated using a set of vertical sliders (Figure 2(c)). Each slider has a movable token that gives the value of the parameter at a particular point in time. Intermediate values are displayed (via a projector) between the tokens. When the user moves one of the tokens, the intermediate values change. Unlike a GUI parameter editor, this device allows complex physical actuations, including two-handed manipulation and pushing many sliders at once (the “bulldozer” action noted by Fitzmaurice et al. [7]).

The horizontal axis of the parameter editor corresponds physically and virtually with the timeline slider. The user can change the scale of the parameter editor by manipulating the temporal interval of the timeline slider. The system automatically updates the vertical sliders to reflect the current timeline interval.

The vertical axis of the parameter editor is augmented with a similar interval selector to choose the range of values to be represented on the vertical sliders. This interval selector defaults to an application-specific standard range, but can be manipulated to edit parameters that have an unusually large or small range.

**Spatial editor**

The spatial editor provides a way of editing higher-dimensional parameters. The basic idea is similar to many previous tangible interfaces: the user specifies positions using a physical object that is tracked by cameras.

In the case of a 2D parameter, the user places a pointer on a table-top surface that represents the range of the parameter. For 2D animation applications, the natural range is image space (the user places the pointer on a projected image of the current frame). The pointer has two primary buttons: one for adding a keyframe and one for deleting a keyframe. The pointer also has secondary buttons for related tasks such as jumping to the next or previous keyframe (which changes the current-time slider) and changing interpolation types.

For higher-dimensional parameters, a similar scheme is used. The user manipulates keyframes of a 3D parameter by placing the pointer in 3D space (tracked by multiple cameras). A four-dimensional parameter can be manipulated using two 2D pointers or a 3D pointer with another physical degree of freedom (such as a knob). A 6D pose (consisting of 3D position and 3D orientation) can naturally be manipulated using a 3D object with orientation sensing, as demonstrated by Dontcheva et al. [5].

What distinguishes this from traditional tangible interfaces is the need to represent how the parameters vary with time. Unlike the 1D parameters, we do not use a spatial axis to represent time. The temporal trajectory of the parameter is displayed (via a projector) in the same space as the current value of the parameter. To avoid visual clutter, we only project values for the current temporal interval and have the values fade away based on distance from the current time. These projected temporal trajectories also indicate the position of keyframes, allowing the user to see how the current frame relates to other keyframes, as shown in Figure 2(d). This display is easy in 2D case. In the 3D case, we propose showing the temporal trajectories and keyframes on 2D displays corresponding to each camera viewpoint.
IMPLEMENTATION
We have implemented a partial working prototype of the design described in the previous section. The prototype includes a physical implementation of the timeline slider, scalar editor parameter, spatial editor, plus a substantial software application that manages the system and associated data.

The current-time selector is actuated using a stepper motor driven by a microcontroller. The interval slider will be actuated in later versions of the prototype.

The scalar parameter editor is built using a disassembled Behringer BCF2000 fader (a device with multiple physical sliders used for mixing music). The current implementation has 8 vertical sliders that provide position feedback and position control. In the future, we will add more vertical sliders and a vertical interval selector.

We use a projector to display virtual state information onto the tangible components. This includes time markings on the timeline slider, parameter value markings on the scalar parameter editor, and the entire spatial parameter editor (color portions of Figure 2).

A software system runs on a desktop PC to control all of the components and manage the currently loaded dataset. Our dataset is structured as a set of layers, each of which has application-specific time-varying properties.

APPLICATIONS
Our system can be used for any domain in which temporal parameters are edited over a finite time domain. The system could also be used for applications that have a single spatial axis. For example, the time axis could represent position along a railroad track, allowing the editing of parameters that vary along the track. In our experiments, we focus on applications related to animation, as motivated in the introduction. Currently, these applications are only partially implemented in the prototype.

Motion Graphics
Motion graphics (e.g. the moving text and graphics for television commercials and new shows) is an ideal match for our system, because it involves many spatial parameters that must be carefully manipulated. A tangible interface to motion graphics is essentially a time-varying version of the GraspDraw system [7], a tangible tabletop drawing program.

In this domain, each layer represents a different graphical element: an image, shape, text string, or video (which could be rendered from 3D graphics). Each layer has a number of time-varying properties, including position, size, orientation, opacity, and matte feathering. The layers can be further modified with filters (e.g. blurring and color correction) that also have time-varying parameters.

3D Animation
Our system is also suitable for 3D computer animation. In this domain, the layers represent individual objects or characters in a 3D world and the properties for each layer include position, size, and articulation. This application requires multiple cameras for 3D sensing. The 3D coordinate system can be specified using a additional tracked object. Related applications include computer-aided stop-motion animation and manipulating the properties of computer generated objects that are added to video.

CONCLUSION
This paper presents the problem of developing a tangible interface for manipulating time-varying parameters. We present a solution that provides intuitive control over temporal motion and temporal scale.

In our current prototype, we have only begun to explore the issues related to the manipulation of time-varying parameters. Limitations of our current design include difficulty visualizing 3D parameters and issues of how to overlay multiple parameters so that the user can see the relationships between the related parameters. In the future, we plan to address these limitations and other remaining issues.

REFERENCES