Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip

Ravin Balakrishnan[1,2] , George Fitzmaurice[1], Gordon Kurtenbach[1], Karan Singh[1]

[1]Alias|wavefront
210 King Street East
Toronto, Ontario
Canada M5A 1J7
{ravin | gf | gordo | ksingh }@aw.sgi.com

[2]Department of Computer Science
University of Toronto
Toronto, Ontario
Canada M5S 3G4
ravin@dgp.toronto.edu

Abstract

We explore a new input device and a set of interaction techniques to facilitate direct manipulation of curves and surfaces. The input device, called ShapeTape™, is a continuous bend and twist sensitive strip that encourages manipulations that use both hands and, at times, all 10 fingers. We explore this input and interaction design space through a set of usage scenarios for creating and editing curves and surfaces as well as consider general interactions such as command access and camera controls. This investigation is carried out by extending Alias|wavefront's modeling and animation package, Maya.

CR Categories and Subject Descriptors: H.5.2 [Information Interfaces and Presentation (e.g., HCI)]: User Interfaces - Input devices and strategies, Haptic I/O, Interaction styles; I.3.3 [Computer Graphics]: Picture/Image Generation - Line and curve generation; I.3.6 [Computer Graphics]: Methodology and Techniques - Interaction techniques.

Additional Keywords: Input devices, bimanual input, ShapeTape, interaction techniques, gestures, curves, surfaces, 3D modeling.

1 INTRODUCTION

In 3D computer graphics modeling, curves are the quintessential primitive for constructing and manipulating surfaces. Successful 3D modelling is largely based on producing the right set of curves which ultimately give rise to the desired 3D shape. Thus, techniques for developing and controlling curve shapes are a critical issue.

Most current interactive curve manipulation techniques require that the user, to some extent, work with the mathematical definition of a curve to control its shape. For example, curves are created and controlled by virtual techniques such as control vertex positioning and adjusting curve continuity and tangency.

In the design industry, traditional physical techniques such as clay modeling and paper drawings are still very popular. In these techniques, the curve itself is manipulated directly by copying preshaped physical curves (e.g., french curve templates) or using physical tools which flex to produce curves (e.g., flexible steels).

Because virtual manipulation and physical manipulation of curves are so different, a designer's physical modelling skills do not wholly transfer to virtual modelling. For example, a designer can express a particular shape using a flexible french curve by simply bending the french curve. However, with a virtual curve it may not be clear how the control vertices need to be placed to attain this shape.

Certain physical objects can also quickly produce curves and surfaces that are hard to create using virtual techniques. For example, the affordances of spring steels are exploited by clay autobody sculptors who use large spring steel rulers, flexed into shape using both hands, to smoothly sweep out a curved surface on clay.

Obviously, both virtual and physical curve modelling have their own pros and cons. What we are interested in is exploring the idea of combining virtual and physical curve creation and control techniques. The key element in our ability to combine these two worlds is a unique input device called ShapeTape™ (Figure 1) [8], which allows users to directly manipulate a virtual curve as a physical object. Our combined interaction style is inspired by our previous example of clay autobody sculptors using steels to sweep out curved surfaces.

In this paper, we explore the use of ShapeTape for performing some basic curve and surface creation and manipulation operations. We present a prototype system we have built to serve as a framework for this exploration. This exploration differs from previous non-conventional modeling paradigms [7, 10, 12] in that we use ShapeTape to directly control modeling curve primitives. We describe the set of interactions that we implemented within this framework and our observations and issues with these interactions. We then discuss how these specific issues generalize to other domains and devices.

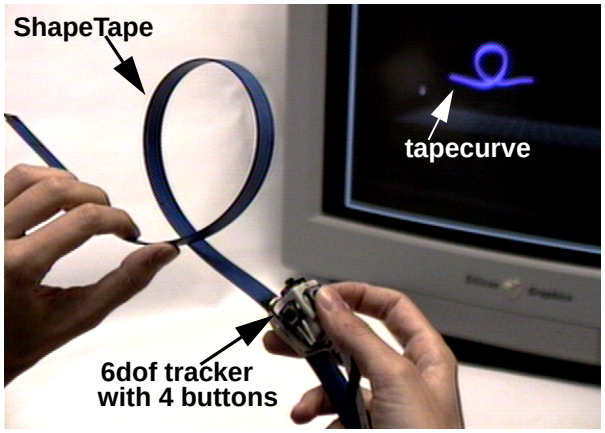

Figure 1: ShapeTape controlling a 3d virtual curve.

Published in Proceedings of 1999 ACM Symposium on Interactive 3D Graphics (I3DG'99), pp 111-118.

2 SHAPETAPE

ShapeTape is a 48 x 1 x 0.1 cm rubber tape that senses its bend and twist. Bend and twist are measured at 6 cm intervals by two fiber optic bend sensors. Resolution is limited by the spacing of these sensors. By summing the bends and twists of the sensors along the tape, the shape of the tape relative to the first sensor can be reconstructed. We sampled all 16 sensors along the tape at 30Hz.

3 APPLYING SHAPETAPE TO MODELING

Our prototype system is built within Alias|wavefront's 3D modeling and animation application, Maya. Maya ran on a Silicon Graphics Indigo2 workstation.

We use ShapeTape to control NURBS curves within Maya. A one to one mapping was used between the Shapetape and a NURBS curve – changing the shape of the ShapeTape resulted in an identical change to the NURBS curve. This was implemented by mapping the shape segments along the ShapeTape to a subset of the control polygon of a NURBS curve. The rotation samples simply map to the control vertex sequence such that: $P_{i+1} = P_i + L*R_i$, where P_i is the position vector of the i^{th} control point, R_i the i^{th} rotation matrix and L a vector representing segment length between samples. P_0, R_0 is given by the position and orientation of the first sensor on the ShapeTape in 3D space (we describe how this is obtained in the next section). For most applications we would like the mapped curve to be planar. R_i is constructed from the bend samples in this case and is simply the rotation matrix for the bend corresponding to the sum of all bends from 0..i. Incorporating the twist samples into the calculation of R_i is straightforward.

3.1 Augmenting ShapeTape

To create and manipulate curves in a 3D scene we need more than the ability to simply input the shape of a curve. We need to support operations like command execution, camera controls, and positioning/orienting the entire curve in 3D space. Since ShapeTape requires and benefits from using both hands and all fingers to operate it, we felt that it would be unwieldy to rely on the status-quo mouse/keyboard for these secondary functions since this would require that the user release their hold on the tape. We therefore augmented ShapeTape so that secondary functions could be performed while both hands manipulated the tape. Another approach would be to design the interactions such that the ShapeTape could be picked up and put down. However, we were interested in the more extreme design of trying to accomplish everything while holding the shape tape. Alternative designs are discussed later in the paper.

To position and orient the curve in 3D space, we attached a 6 degrees-of-freedom (dof) tracker (an Ascension Flock of Birds) to the starting point of the tape (Figure 1). The tape and the virtual curve it controls (we call this the "tapecurve") then operates relative to this starting position.

All our interactions were designed to operate in a perspective view and, therefore, users need to at least be able to tumble the virtual camera to get both depth perception and different views of the curves/surfaces they were working on. We provided camera controls by using a 2-dof custom designed puck that was operated by the user's right foot on a Wacom digitizing tablet (Figure 2). This "footmouse" had a single button on it that allowed the user to switch to camera tumble mode and tumble the scene by stepping

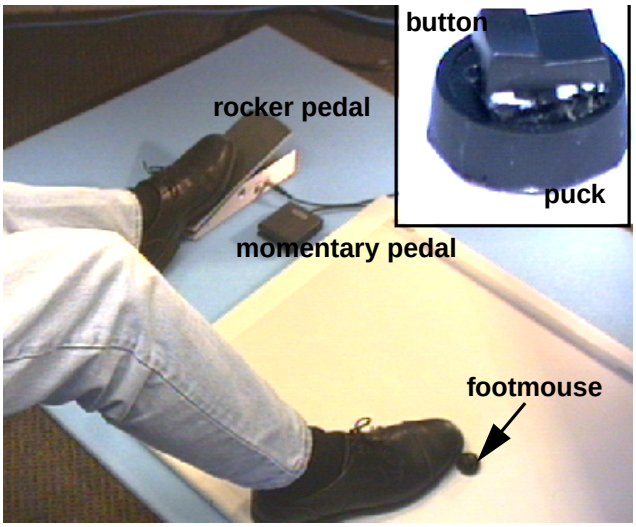

Figure 2. Foot pedals and footmouse. Inset picture is a closeup of the custom designed footmouse puck.

on the footmouse and moving it around on the tablet. Since the scenes we were working with were not very complicated, we felt that tumbling was a sufficient camera control. Other camera operations such as pan, dolly, and zoom were not implemented in our prototype.

We added four pushbuttons to the 6-dof tracker to provide for command execution and clutching of the tracker (Figure 1). The buttons were chosen and arranged on the tracker such that accidental triggering was minimized and more than one button could be pressed at the same time.

Using the tracker buttons requires one hand to be at the end of the ShapeTape which reduces the user's ability to manipulate the shape of the ShapeTape itself. To somewhat alleviate this problem, we used two footpedals (a rocker pedal and a momentary pedal) operated by the left foot for additional button input that could be operated while the user used both hands to shape the curve (Figure 2).

We now discuss several interaction techniques we have implemented based on this input configuration to explore the creation and modification of curves and surfaces.

3.2 Interaction Techniques using ShapeTape.

In a manner similar to most 3D modeling packages we implemented various curve and surface manipulation functions as temporal modes (commonly called "tools"). We did not implement a technique for switching between the different tools. As a stop-gap measure, we rely on the keyboard to do this. Ideas for supporting tool switching seamlessly in our system are discussed in a later section.

In each of our tools, the following footpedal and button assignments were used. Tables 1 and 2 summarize these assignments.

When the rocker pedal was up, the tracker was operational and the tapecurve could be positioned and oriented in 3D space. We call this "position/orient tapecurve mode". In this mode, buttons 1, 2, and 3 engage and clutch movement along the x, y, and z axes respectively. Chording buttons 1, 2, and 3 allowed movement in multiple axes simultaneously (e.g., pressing both buttons 1 and 2, engaged movement in the plane defined by the x and y axes). Button 4 was used as a toggle to enable and disable all three rotational degrees-of-freedom of the tracker.

Device	Limb	Function
rocker pedal	left foot	up: position/orient tapecurve mode down: command mode
momentary pedal	left foot	toggle between freezing and unfreezing shape of tapecurve
footmouse	right foot	tumble camera
ShapeTape	both hands	control shape of tapecurve
tracker	right hand	control position and orientation of tapecurve
tracker buttons	right hand	command access and tracker constraints (see Table 2)

Table 1: Functionality of devices

tracker button	position/orient tapecurve mode	command mode
button 1	constrain to x axis	next step in tool
button 2	constrain to y axis	end tool
button 3	constrain to z axis	
button 4	rotation on/off	

Table 2: Tracker button assignment

When the rocker pedal was down, the tracker was disengaged and the tracker buttons could be used to execute commands. We call this "command mode". Button 1 was always used to activate the next step in the tool currently being used. Button 2 signals completion of a tool's operation and resets the tool to its initial state (this allows a tool's operation to be repeated without having to re-invoke the tool). Buttons 3 and 4 were used for commands specific to particular tools, which we describe later.

The footmouse and momentary pedal were independent of modes and thus could be used at any time.

3.2.1 Curve Creation

The first tool we explored allows the creation of curves in 3D space. As described earlier, the shape of the tapecurve was controlled by the ShapeTape and its position and orientation controlled by the tracker.

At any time, the momentary pedal could be depressed to freeze the shape of the tapecurve. Depressing the momentary pedal a second time unfreezes the shape of the tapecurve. This concept of freezing/unfreezing the tapecurve shape using the momentary pedal is used throughout our different interaction techniques. Note that the tapecurve can still be positioned and oriented in 3D space when its shape is frozen.

When in command mode, pressing button 1 resulted in a snapshot copy of the tapecurve being placed in its current location and orientation. We refer to this as "baking" the tapecurve into the 3D scene. Note that we can bake the tapecurve either when it is live or frozen.

We found this technique to be intuitive for creating free-form 3D curves and it allowed for quick exploration and specification of curve shapes, position, and orientation.

While the position and orientation of the tapecurve can be controlled fairly precisely using our methods for constraining movement to particular axes, it was difficult to precisely control the shape of the tapecurve. Borrowing from the physical tools used by designers, we investigated using physical constraints to improve control over the shape of the tapecurve.

One form of physical constraint is to attach spring steels to ShapeTape. Using steels of different thicknesses and degree of flexibility (Figure 3a), we can vary the deformability of ShapeTape and, in a sense, physically control the smoothness and curvature of the tapecurve. Using small strips of velcro on the ShapeTape and the steels, we are able to switch between different steels easily. One characteristic of spring steels is that they have to be continually held in the desired shape and do not retain the deformed shape when released. While this can be a desirable feature when exploring shape, it can be a shortcoming when trying to maintain a particular shape for a period of time. To address this shortcoming, we devised a jig (Figure 3b) that allowed us to mechanically hold the spring steel in a deformed shape. Once the desired shape is obtained, the wingnuts on the jig are tightened and the entire jig (and resulting tapecurve) can be positioned and oriented as

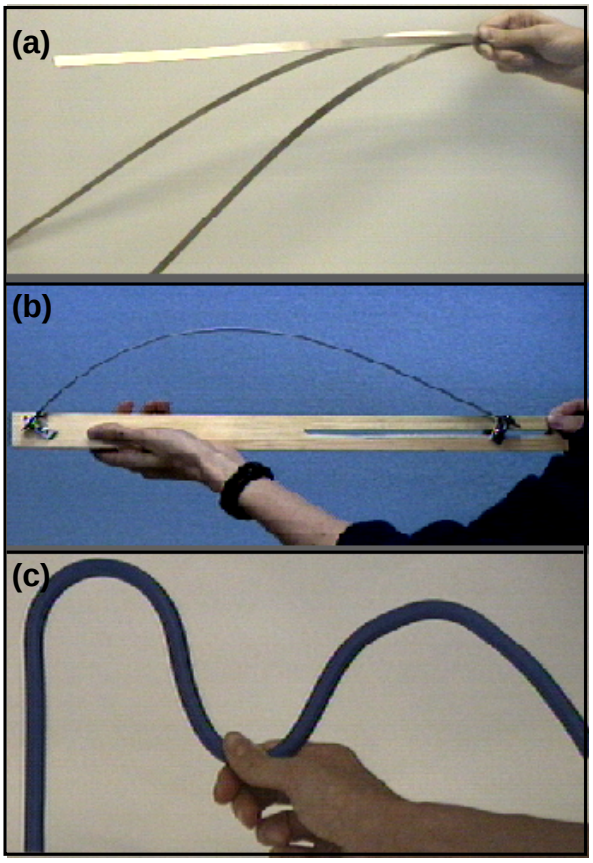

Figure 3. (a) Spring steels of different thicknesses and flexibility. (b) Jig for constraining spring steel. (c) Flexible curve that retains its deformed shape.

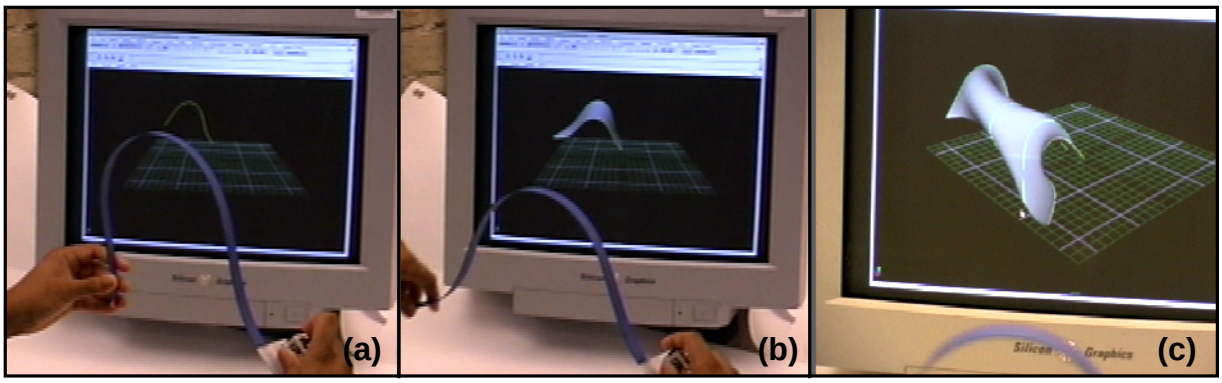

Figure 4. Loft. (a) Placement of initial profile curve. (b) Dragging out first section of the lofted surface. (c) The final surface lofted over five interactively placed profile curves.

required. Position and orientation of the jig can also be physically constrained in a variety of ways. Examples include simply dragging the jig on a tabletop to constrain movement to a plane, or mounting the jig within a larger jig that imposes some other positional or rotational constraints.

The last form of physical constraint we explored was the use of flexible curves (Figure 3c). These curves, used in the design industry, do not provide the high level of smoothness that spring steels offer but retain their deformed shape when released. They are a good compromise when smoothness is not an important factor.

The use of steels, jigs, and flexible curves have the advantage that the user can easily switch between these different constraints and leverage off their existing knowledge of the physical world when learning to use these tools. These advantages have been expounded by Fitzmaurice et. al. [3] in their Graspable UI paradigm, by Ishii et. al. [4] in their Tangible UI research, and by Hinckley et. al [5]. However, one disadvantage is that we also inherit all the limitations of the physical world. Since we haven't yet implemented virtual solutions to address these limitations, we defer the discussion of these solutions to a later section of this paper.

Given the ability to interactively create 3D curves using ShapeTape, we now describe three techniques for creating surfaces interactively from these curves.

3.2.2 Loft

"Loft" refers to the construction of a surface that passes through a series of profile curves. The status-quo interaction technique requires that at least two profile curves be predefined before a surface can be lofted over them. Additional curves can then be added to extend the lofted surface.

Using ShapeTape, our "loft tool" creates surfaces as follows: first, we use ShapeTape to bake the initial profile curve (Figure 4a). Then, we press button 1 in command mode to create a lofted surface from the initial profile curve (c1) to the tapecurve. Since the tapecurve is still "live", the user can dynamically change the shape of the lofted surface segment in real time (Figure 4b). Pressing button 1 in command mode again bakes the tapecurve, resulting in baked curve c2 and a baked surface from curves c1 to c2. A new live surface is then lofted from curve c2 to the tapecurve. This process can be continued to successively extend the lofted surface. Once the final surface is obtained, button 2 is pressed and the tapecurve is detached from the final lofted surface (Figure 4c).

Thus, this technique allows users to "drag out" a surface starting from the initial profile curve, baking sections of the surface as desired. The ability to manipulate the current surface segment in a live manner allows the user to preview and explore variations of the forthcoming surface before baking it. In contrast, the status quo interaction technique requires the user to lay down a series of curves and then loft a surface across those curves. No preview of the resulting surface is given, and any changes have to be made in a post-creation process.

The physical constraints we explored in the previous section can also be used here to constrain the tapecurve and thus create the smooth controlled surfaces that are typically used in non-organic technical modeling.

3.2.3 Revolve

"Revolve" refers to construction of a surface by revolving a profile curve about a given axis.

In our "revolve tool", we first specify the profile curve using Sha-

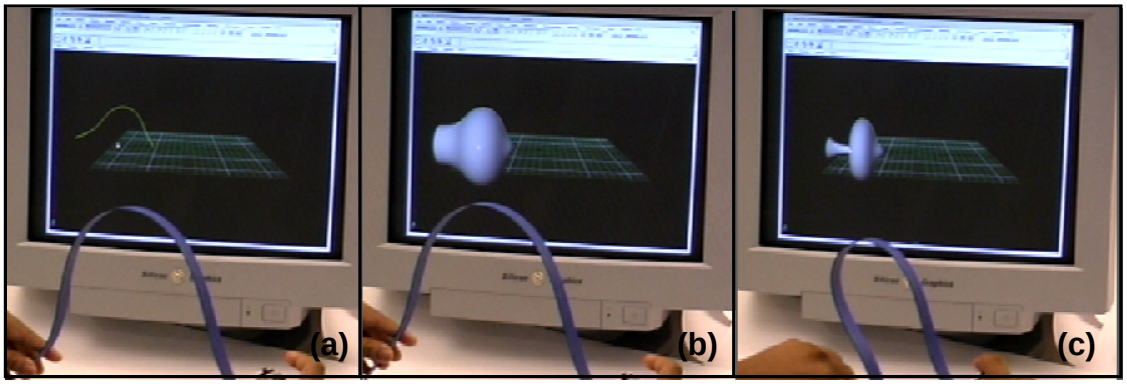

Figure 5. Revolve. (a) Placement of initial profile curve. (b) Revolving the profile curve about the x-axis. (c) The revolved surface can be interactively manipulated to explore different shapes, positions, and orientations.

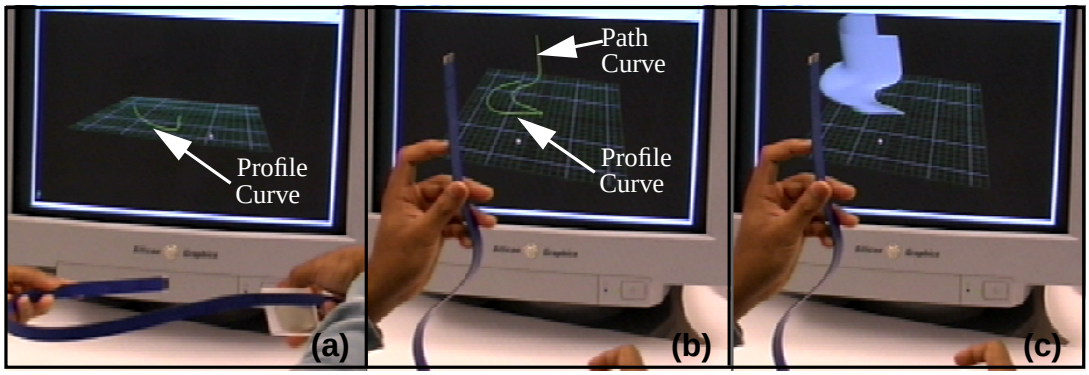

Figure 6. Extrude. (a) Placement of profile curve. (b) Placement of initial path curve. (c) The extruded surface can be interactively manipulated to explore different shapes, positions, and orientations.

peTape (Figure 5a). This curve can either be frozen or live. Then, we press button 1, 3, or 4 in command mode to revolve the profile curve about the x, y, or z axis respectively (Figure 5b). Since the profile curve is still controlled by ShapeTape, the resulting surface can therefore be manipulated in a very interactive manner to explore different shapes, positions, and orientations (Figure 5c). Button 2 can be pressed at any time to complete the revolve operation which bakes the revolved surface.

In status-quo revolve techniques, the resulting revolved surface can only be manipulated by moving control vertices of the profile curve or by translating, orienting, or scaling the entire curve. While this is fine for small modifications, it does not provide the sense of engagement or expressiveness of the ShapeTape technique. On the contrary, ShapeTape in its current configuration does not easily support precision adjustments to the surface.

3.2.4 Extrude

"Extrude" refers to constructing a surface by sweeping a cross sectional profile curve along a path.

In our "extrude tool", we first specify and bake the profile curve (Figure 6a) by pressing button 1 in CommandMode. Then, the tapecurve is used to specify the path curve (Figure 6b). This curve can either be frozen or live. Pressing button 1 again creates an extruded surface by sweeping the profile curve along the path curve (Figure 6c). Since the path curve is still controlled by ShapeTape, the extruded surface can now be manipulated interactively. Button 2 can be pressed at any time to bake the extruded surface and detach the tapecurve from it.

As with the Revolve example, the ShapeTape extrude technique allows for more expressive manipulations of the surface than the status-quo interaction technique. However, our technique currently allows interactive manipulation of the surface only by controlling the path curve, not the profile curve. We plan to develop techniques to dynamically select which curve ShapeTape controls.

3.2.5 Surface Deformations

The previous tools permit the creation of surfaces. We now discuss techniques for deforming existing surfaces of arbitrary shape. We use ShapeTape to manipulate "wires" – a geometric deformation technique based on space curves [11]. This application also highlights the use of the ShapeTape's twist capability.

A wire is a curve whose manipulation deforms the surface of an associated object near the wire curve. The deformations to objects are based on the relative deviation between the wire curve and its corresponding reference curve (Figure 7a). The reference curve is a congruent copy of the wire curve made when objects are associated with it. An appealing attribute of wires is that not only do they

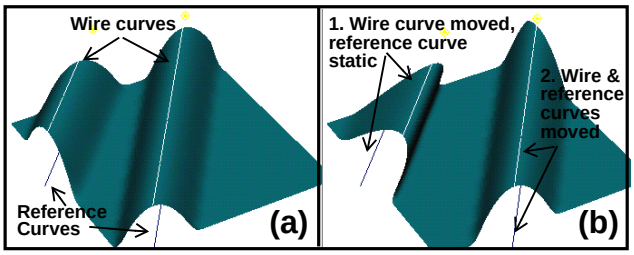

Figure 7. Wrinkles and creases using wires. (a) Shows two wire curves and associated reference curves deforming a surface. (b) 1. If a wire curve is moved while its reference curve is static, the wrinkling effect is increased. 2. If a wire curve is moved along with its reference curve, the wrinkle travels along the surface.

utilize the bend of the curve, but they also embody the notion of twist around the wire curve and impart it to the surfaces they deform. We thus are able to use the twist of the ShapeTape to directly control the twist along a wire curve. The effect of twisting the ShapeTape is thus manifested as a surface deformation even though it is not visually represented on the wire curve.

Our "wire tool" provides three styles of interaction to deform surfaces with wires. In all three styles, we attach a wire curve to a surface to be deformed by pressing Button 1 in CommandMode. Pressing Button 2 in CommandMode detaches the wire from the surface. Button 3 is used to change between the three styles of interaction.

In the first style, ShapeTape controls the bend, position, and orientation of the wire curve while the reference curve remains static. This allows for creasing deformations to be created as illustrated in Figures 7b(1) and 8a,b.

The second style operates in the same manner as the first style except that the reference curve is translated along with the wire curve. This allows for "travelling" wrinkle deformations as illustrated in Figure 7b(2).

The third style uses twist in addition to bend, position, and orientation to control the wire curve. Adding twist further deforms the crease and wrinkle deformations in a manner similar to pinching (Figure 8c).

Wires are a deformation technique originally designed to create organic surfaces like cloth and skin. We found that using ShapeTape with wires allowed for deformations of surfaces that would be very difficult to specify with traditional tools for manipulating wires. Like our surface creation tools, the ability to quickly explore different deformations effects allowed for more expressive manipulation than the control vertex positioning status-quo techniques.

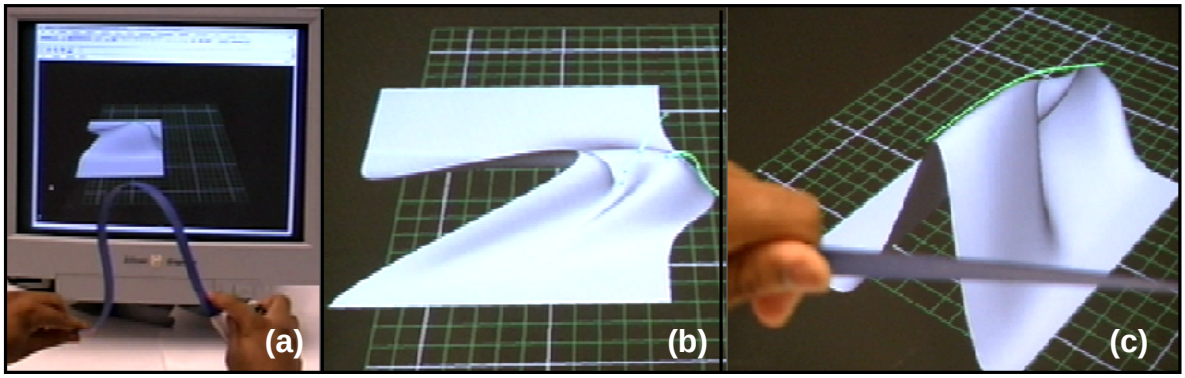

Figure 8. Surface deformations using ShapeTape. (a) Bend of wire curve deforming a surface. (b) Bend and position of wire curve deforming a surface. The reference curve is static. (c) Twist of wire curve deforming a surface.

4 FURTHER ENHANCEMENTS

There are several ideas which, although we have not implemented, we feel are important in continuing to develop our ShapeTape prototype.

ShapeTape subsection specification – The ability to specify subsections of the ShapeTape would be useful. For example, suppose a user is happy with the shape of one half of the tapecurve but wishes to modify the other half. Sensors along the length of the ShapeTape could be used to specify which subsections are active, thus limiting changes to the corresponding parts of the tapecurve. Possible sensing technologies include binary microswitches and pressure sensitive strips.

ShapeTape to tapecurve mappings – An important issue is the control mapping between the ShapeTape and the tapecurve. In our prototype a one-to-one mapping was used where the unit length of the ShapeTape mapped to the unit length of the tapecurve with a constant gain. The ability to modify this mapping would be valuable. For example, the entire ShapeTape could be mapped to a subsection of the tapecurve, allowing finer control over that portion of the tapecurve. Subsections of the ShapeTape could also be mapped to subsections of the tapecurve in a non one-to-one manner. Editing of existing curves in a scene could be achieved by selecting a subsection of a curve and mapping it to a subsection of the tapecurve. This section of the virtual curve could then be edited by the ShapeTape.

Increasing/decreasing control gain – The control gain of the ShapeTape could also be modified. For example, by increasing the control gain ratio, small ShapeTape bends could translate into larger bends in the tapecurve. This could be used as a convenience mechanism to reduce physical movement. In contrast, the gain ratio could be decreased and this would result in more precision control over the bends of the tapecurve.

Non-uniform control gain – Varying the gain ratio over the unit length of the ShapeTape may also be a useful mechanism. Mappings could be devised where the ShapeTape is much more sensitive (or insensitive) over certain sections of the shape. This could be used to create curves which when bent have a pre-bias towards a certain shape.

Frame of reference – As the scene rotates (i.e., when the camera is manipulated) should the tapecurve remain stationary in user space (ego-centric) or turn with the scene (scene-centric)? If the tapecurve follows a scene-centric model, this will sometimes produce a stimulus response mismatch between movement of the ShapeTape and movement of the tapecurve. However, if the tapecurve follows an ego-centric model, this too can lead to problems since moving the scene then in effect moves the tapecurve relative to the scene. For example, if the tapecurve was being used as a deformer, unwanted deformations would occur when the scene was rotated. While we have some ideas for solutions to these problems, they have not been sufficiently explored.

Additional command access – While working with ShapeTape, we found it necessary to provide a way to switch between tools. There are many possible solutions to explore here. First, we could add additional push buttons to the tracker or introduce more foot pedals. This solution is not very attractive as the tracker is already crowded with buttons. Introducing more foot pedals may be problematic as the user must search for the proper fool pedal, diverting their attention from the 3D scene. Second, we could use speech and voice recognition to specify commands. Third, we could create a set of ShapeTape gestures that would map to commands. Here the challenging issues are being able to define meaningful shapes that match their assigned command and finding good gesture and shape recognition algorithms. Also, we'll have to toggle the ShapeTape between specifying command gestures and controlling the tapecurve. Last, we could add a series of pressure sensors along the length of the tape. These pressure sensors could be used as a button strip for command control buttons. One limitation of this idea is that these buttons cannot be used while simultaneously specifying a shape since pressing will deform the tape (for example, the "freeze" command would be a poor choice). While some of these ideas may result in a good solution, the problem of providing additional command access remains an open issue.

GUI access – Beyond command access, the ShapeTape device could work in conjunction with standard GUI elements by driving the cursor. This would allow us to use standard GUI widgets like graphical buttons, sliders, and menus for operations such as tool switching without having to put down the ShapeTape. This could be done by tracking the location of the end of the ShapeTape relative to the screen and mapping this to a cursor location. The foot pedals could be used for simulating mouse buttons. Alternatively, button presses could be simulated when the tape endpoint is moved in or out a fixed distance from the screen.

5 GENERAL OBSERVATIONS

Our experiences so far have lead us to some general observations about this style of input and these types of interaction techniques. Below we outline our findings and how they are relevant to other application domains.

High dimension input – We consider ShapeTape to be in a class of input devices we call HiD (High dimension input). Roughly speak-

ing, HiD devices are devices or arrangement of devices which allow simultaneous input of more than 3 degrees of freedom. Systems like the monkey armature [6], dataglove [2] and haptic lens [9] are examples of HiD devices. In many ways, this paper explores issues in harnessing HiD input. We believe that there are issues common to most HiD input configurations. We now discuss what we believe to be the major issues.

Regulating Input – HiD devices require the ability to regulate the input. Mechanisms are needed for easily engaging and ignoring sets of input dimensions. For example, in our prototype, we found the need to freeze the 3D position, 3D orientation, and shape of the input curve. These mechanisms could be provided in either or both the virtual or physical mediums.

Need for other independent devices – In our prototype, we made use of auxiliary devices to assist in regulating the input from our HiD device (e.g., using a footpedal to freeze the tapecurve) or for interface control (e.g., a footmouse to tumble the camera view). In general, auxiliary devices are needed if a regulating or interface action interferes with control of the HiD device. For example, there was a need to be able to hold the shape of the tapecurve and at the same time trigger a "freeze" action. Note that employing the use of other limbs is a common practice in other HiD domains. For example, guitar and piano players use footpedals to select playing modes and effects while playing.

Input retention – Rather than requiring a user to "hold" a particular setting of a HiD device, a device could be built such that it retains its settings. For example, by attaching ShapeTape to jigs or flexible french curves (Figure 3), we created the ability for the ShapeTape to retain its settings. Removing the requirement of constantly holding the device frees the hands to operate other devices such as the mouse and keyboard. This may allow more standard UI techniques to be used to support regulating and auxiliary functions.

Interdependence and quality of input dimensions – A simplifying but sometimes confounding factor to consider in HiD input is the interdependence of input dimensions. Consider ShapeTape – while there are a total of 16 sensors, and thus 16 degrees of freedom, it is difficult to actuate one sensor in isolation. In fact, the user perceives the ShapeTape as a single malleable input strip. They judge the quality of the input based on how quickly and accurately the virtual shape matches the physical input shape. This directly corresponds to the quantity and quality of the sensors as well as the physical material properties of the ShapeTape.

Sense of engagement – HiD input can offer a greater sense of engagement and expression compared to traditional lowD input (e.g., mouse) which often emphasize specification and precision. With HiD input, precision can be temporarily attained by reducing the input dimensions being sensed, by using physical/virtual constraints, and by varying the control gain. In contrast, there is no easy way to improve the sense of engagement with lowD input. 3D graphical manipulators [1] are one technique for providing a greater sense of engagement but this still offers limited expressibility.

Control skill demands – HiD input may place a higher demand on the user's motor and cognitive processes. Users are required to attend and monitor many streams of simultaneous input. This is especially true for precision work. We believe that cognitive and motor demands may be reduced when: (1) the physical device closely matches the virtual representation, (2) the input device allows the high dimensions to be coordinated in a familiar metaphor (e.g., the ShapeTape bend and twist sensors are aggregated in a single strip), and (3) the interaction techniques allow for constraining input through other input streams (e.g., tracker buttons constrain movement along the x, y, and z axes).

Disadvantages of physical representations – While the ShapeTape offers physical manipulation of an input strip this approach is susceptible to the constraints of the physical world. For example, any given ShapeTape has certain bend properties which are invariant. In addition, having customized input devices attached to a given system makes it difficult to move to another workstation. This is in contrast to virtual tools being available on any system. Physical tools are also subject to the "nulling problem." This problem occurs when the physical state of the device starts out matching the virtual representation but becomes stale as the virtual state changes without keeping the physical device consistent. This nulling problem can often be alleviated by operating the physical device in relative mode instead of absolute mode.

"Iron horse effect"– In general, a major design issue for HiD input is the danger of mimicing properties of the analogous physical tools too closely. That is, replicating not only the advantages of a physical tool but also its disadvantages (the iron horse effect – some of the first automobiles were not only controlled like a horse but also shaped like one). Avoiding the iron horse effect requires carefully determining exactly what ability a physical tool offers versus what is merely an artifact of physicality.

6 FUTURE RESEARCH

There are a number of issues relating to ShapeTape that need to be further explored:

- Our current prototype paradigm has ShapeTape as the primary input device, always in hand, but alternative input configurations with different costs and benefits are possible. For example, one alternative has the ShapeTape operating on a 2D surface where the contour of the tape is sensed as an input curve but the location and orientation of the curve is managed through more traditional interaction techniques (i.e., manipulators) with the mouse. The benefits of this configuration is that the tape does not need to be continuously held and a 6dof tracker isn't required.
- We would also like to consider the use of two or more ShapeTape devices to form a shape sheet. This would allow one to directly manipulate surfaces.
- While we were happy with the performance of the footmouse and foot pedals, we believe that additional design can be done to improve their usage.
- In addition to using ShapeTape for modeling, we would like to explore other application domains such as animation. Here the ShapeTape could be used to specify motion paths, adjust timing curves, motion capture, or for quickly editing and posing characters and deformable objects like cloth.
- Finally, we would like to consider if any of the interaction techniques will transfer to other two handed input configurations. For example, one could imagine a "poor man's" ShapeTape. Rather than using ShapeTape, two devices such as two pucks on a digitizing surface or two 6dof trackers could be used. A virtual curve between the two devices could be inferred given their positions and orientations.

7 CONCLUSIONS

The one-to-one mapping between Shapetape and a NURBS curve allows for great ease of use and learning. For example, the manner in which the shapetape controls the NURBS curve is immediately obvious. The fact that the underlying curve being controlled is a NURBS curve is completely transparent.

One dominating observation in our prototype was that ShapeTape imparts an expressive and live feeling to operations. Specifically it allows different shapes and effects to be quickly attained. This property is especially suitable for conceptual modeling – modeling done to allow a designer to quickly explore form, shape, and size.

ShapeTape at this point appears less suitable for technical modeling, which focuses on constructing precise curves and surfaces. To make ShapeTape more suitable, first, the precision of the shapetape itself would have to improve. Second, both physical and virtual ShapeTape specific modeling constraints and constructs would have to be invented and developed.

We believe we have discovered some fundamentals of the basic interaction framework and input configuration which is effective for managing the HiD input of ShapeTape.

ACKNOWLEDGMENTS

We thank Russell Owen, Eugene Fiume, and Bill Buxton for valuable discussions and assistance during the course of this work. We also thank Lee Danisch of Measurand Inc. for advice and technical assistance with regards to the ShapeTape device.

REFERENCES

[1] Conner, B.D., Snibbe, S.S., Herndon, K.P., Robbins, D.C., Zeleznik,R.C. & van Dam, A. (1992) Three-dimensional widgets. *Proceedings of Symposium on Interactive 3D graphics '92*, 183-188.

[2] CyberGlove. Virtual Technologies. *(www.virtex.com)*

[3] Fitzmaurice, G. W., Ishii, H., & Buxton, W. (1995). Bricks: Laying the foundations for graspable user interfaces. *Proceedings of CHI'95 Conference on Human Factors in Computing Systems*, 442-449.

[4] Ishii, H., & Ullmer, B. (1997). Tangible Bits: Towards seamless interfaces between people, bits and atoms. *Proceedings of CHI'95 Conference on Human Factors in Computing Systems*, 234-241.

[5] Hinckley, K., Pausch, R., Goble, J.C., & Kassell, N.F. (1994). Passive real-world interface props for neurosurgical visualization. *Proceedings of CHI'94 Conference on Human Factors in Computing Systems*, 452-458.

[6] Monkey. Digital Image Design, Inc. *(www.didi.com)*

[7] Sachs. E., Roberts, A., & Stoops, D. (1990). A tool for designing 3D shapes. *IEEE Computer Graphics, 17(3)*, 253-261.

[8] ShapeTape. Measurand Inc. *(www.measurand.com)*

[9] Sinclair, M. (1997). The haptic lens. *Visual Proceedings of SIGGRAPH'97 Conference*, 179.

[10] Shaw, C. & Green, M. (1994). Two-handed Polygonal Surface Design. *Proceedings of UIST'94 ACM Symposium on User Interface Software and Technology*, 205-212.

[11] Singh, K., & Fiume, E. (1998). Wires: A geometric deformation technique. *Proceedings of SIGGRAPH'98 Conference*, 405-414.

[12] Zeleznik, R.C., Herndon, K.P., & Hughes, J.F. (1996). SKETCH: An interface for sketching 3D scenes. *Proceedings of SIGGRAPH '96 Conference*, 163-170.