Alternate User Interfaces for the Manipulation of Curves in 3D Environments

by

Tovi Grossman

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Alternate User Interfaces for the Manipulation of

Curves in 3D Environments

Tovi Grossman

Master of Science

Graduate Department of Computer Science

University of Toronto

2004

In the design industry, profile drawings are typically sketched from various perspectives and combined in some sort of CAD system to build a digital representation of a 3D model. Limited virtual systems are available for this sketching process, as the current interfaces for manipulating curves typically use a standard point cursor to indirectly adjust curve parameters. Furthermore, limited functionality is available for allowing artists to integrate their profile sketches into a 3D environment, allowing them to survey the model as they design it from multiple viewpoints. In this thesis we will describe the design and implementation of interaction techniques for the creation, manipulation, and integration of curves in 3D environments. The systems which we present have been demonstrated to various users, from colleagues to professional artists, and user feedback is reported.

Acknowledgements

I feel incredibly lucky to have worked in the research department at Alias|Wavefront for two summers. For me it served as an introduction to human computer interaction and to research. However it was the people that worked there which made it such a valuable experience, and I would like to thank Ravin Balakrishnan, Gord Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton, for their guidance, encouragement, and conversations. I would also like to thank Karan Singh who provided valuable suggestions and ideas during my work with ShapeTape at U of T. Lastly, I am thankful to all of my family and friends who have supported me throughout my academic career.

ACM should also be acknowledged for allowing previously published material to appear in this thesis[1].

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Specifying 3D curves is one of the most important tasks that a 3D user interface must support [10]. The interface must not just provide a tool for the artist, but provide an appropriate interface for creating curves which fit the artist's tasks and skills. Whether it is an animator specifying the path of a camera angle in a three dimensional world, or a car designer detailing the side profile of a new car, it is of utmost importance that the right tools are available to the user.

In the physical world, this importance is well recognized and tended to. Different artistic methods are used based on the goal of the artist. For example, an artist who is creating a technical drawing matching specific engineering criteria may use a drafting table. This allows them to create a precise drawing with the availability of various tools such as rulers and French curves. On the other hand, an artist who wishes to create a sketch of a large model, such as an automobile, could use an upright wall-sized surface. Using techniques which have been developed for such large scale drawings the artist can create long smooth curves, and then examine the results at a scale which closely matches the final product.

This rich set of tools and interaction techniques simply isn't available in the virtual realm. Whether their task is to create conceptual sketches, precise technical drawings, or even three dimensional models, graphic artists typically use some sort of mouse or pen based interface on a standard desktop computer. In addition, the current interactive curve

manipulation techniques require that the user, to some extent, understand and work with the underlying mathematical representations of curves in order to control its shape and size. It is thus desirable to investigate alternate user interfaces for the creation and manipulation of curves. This is the topic explored in this thesis. We will look at a suite of interaction techniques for designing curves in three dimensional environments.

1.2 Basic Concepts

This thesis will present a number of novel interaction techniques, and the prototype systems which they were implemented in, for creating and manipulating virtual curves. The two main goals are to provide interfaces for shape based creation and editing of curves, and for integrating curves into 3D environments. These basic concepts are now described

1.2.1 Shape Based Curve Manipulations

Typical user interfaces for creating and editing curves are point based. The user first specifies control points through which the curve will pass, and then the system, based on some mathematical representation, creates a curve which passes through these points. The user can then change the location of a control point, and modify the direction and amplitude of the tangent at which the curve passes through a control point. The user iterates until the system creates the desired shape. This tool is well suited for engineers and industrial designers, as they can use this interface to quickly create a desired curve which follows specific geometric properties. However, they will only be able to do so if they have a good understanding of the underlying mathematics of the curve, and the system's internal representation of it. Many skilled artists and designers require no such knowledge. This makes the status-quo point based tool somewhat ineffective for them. The interaction techniques presented in this thesis are based on users specifying and editing the entire shape of a curve, rather than its underlying mathematics. This allows artists to utilize the skills which they would use in the physical world.

1.2.2 Integrating Curves Into 3D Environments

Another fundamental concept which will be presented is the ability for users to integrate the 2D and 3D curves which they create into a 3D environment, allowing users to build up 3D wireframe models. Currently, the most common method for creating 3D models is for artists to first create separate sketches of a model from various perspectives, and then after these sketches are completed combine them into a single model using some sort of 3D CAD package. Previous virtual systems for sketching 2D and 3D curves provide methods for creating these individual sketches, but do not provide techniques for integrating multiple curves or sketches into a single 3D model [2, 3, 5, 10, 35, 37]. The interfaces described in this thesis allow artists to create the 2D profiles while the underlying 3D model is being simultaneously updated, and easily transition between the 2D and 3D views.

1.3 Thesis overview

The following chapters will first present a background on the major aspects of the prototypes which were designed. The new interaction techniques and prototype systems which they are implemented in are then described in detail. We will conclude with a discussion of the major contributions of the work presented, and possible lines of future research. Much of the work presented in this thesis extends the work of two core publications [2, 3] which will be described in sections 2.5 and 2.6. The new work which is described in Chapters 3-5 is also based on published material [16-18], however this thesis expands on the background and discussion of these publications.

Chapter 2

Background

In this chapter we review literature which provides rationale for the choices we made in the design of our interaction techniques. This review has been categorized into the major aspects of our user interfaces. First, we will look at interfaces which benefit from two handed interaction. We then look at previous interfaces which utilize large displays followed by tangible user interfaces. Next, we do a survey on previous publications that present different forms of user interfaces for designing curves. Finally, we will review the work done in [2] and [3], the two major publications which this thesis expands on.

2.1 Two Handed Interaction

Over the past two decades several user interface researchers have recognized that people often use both hands to cooperatively perform many tasks, and have explored the idea of simultaneously using both hands in the computer interface. In an initial study, Buxton and Myers [9] showed that a status-quo one-handed interface was inferior to a two-handed interface in a compound task. The two-handed interface split the compound task into two subtasks that could be performed in parallel by both hands. Similar findings were found by Kabbash, Buxton, and Sellen [28]. However, they also showed that if an inappropriate interaction technique is employed two hands could be worse than one, particularly when cognitive load is increased. In the physical world, artists are trained and skilled with simultaneously working with both hands. In the virtual domain however, techniques that allow artists to drive continuous input signals with both hands are still not in common use. Presented in Chapters 3, 4 and 5 are techniques for curve and surface

creation and editing, camera navigation, and other general interactions which utilize both hands working in parallel. Previous navigation and drawing techniques have been developed for two handed interaction, and are now described.

2.1.1 T3

T3 [31] is an experimental GUI paradigm which allows the creation and editing of simple 2D graphics, such as circles and rectangles. The system was created with the goal of increasing the degrees of manipulation and comfort of input. To accomplish this, two Wacom Tablet pucks were used, allowing both hands to be used for input. Each puck sensed both x and y position and rotation. Each puck also had a single button for input. As in standard GUIs, the dominant hand is used to drive the cursor, drag objects and carry out the functions of the current tool. The non-dominant hand is used to position and orient a ToolGlass palette from which the dominant hand can select tools, and can also be used for camera controls, such as panning and rotating the artwork. The authors explain the benefits of this bimanual mode of interaction:

> Because humans are very skilled at having one hand follow or stay close by the other hand, this skill transfers very effectively in T3. It is very easy to keep the Tool-Glass always close to the cursor. An artist does not have to constantly "pick-up, move, and put-down", the tool palette as required by traditional floating tool palettes. [31]

This is analogous to the physical world, where an artist could create a pencil sketch with their dominant hand, and hold an eraser in their non-dominant hand, so the eraser can be quickly transferred to the dominant hand if corrections need to be made. The T3 interface also supports the notion of curve guides, which is a tool such as a ruler or a French curve. Much like they would in the physical world, the artist positions and orients the curve guide with their non-dominant hand, and sketches over it with their dominant hand. T3 is an interesting paradigm which increases the quality of performing functions, as it supports the use of both hands.

2.1.2 Two Handed Navigation Techniques

Along with the two-handed drawing techniques presented in [31], the system also allowed users to simultaneously translate, scale, and rotate objects using both hands. When both hands selected an object, the points where the object was selected become pinned to the corresponding cursor. Moving the hands apart would scale the image up, moving them together would scale down, and moving both hands along the same line would translate the object. Rotating the hands would rotate the object. A similar technique was implemented in by Hinckley et al. in [24] for map navigations. They state that "the two-handed interface allows users to think in terms of 'navigating the map,' rather than strictly in terms of the atomic actions of panning or zooming." They call their technique a "stretch and squeeze" metaphor. Similar to the technique in [31], moving the hands apart zooms in (stretches), and moving them together zooms out (squeezes). In their implementation, rotation was not allowed. They performed user testing, and found users initially only used the two handed zooming to navigate the map, but after a few trials, they gained sufficient skills to perform compound panning and zooming actions.

2.1.3 Two Pointer Input For 3D Interaction

The described interactions developed in [24] and [31] were extended to 3D environments in [38]. A system was created where two hands were used to control two independent cursors to perform various operations in 3D desktop applications. Their operations included transformations, navigations, camera controls, and basic shape editing. They found that the use of two hands allowed them to simultaneously perform two related operations, where as in an interface using a single pointer, the operations would have to be performed one after another. For example, objects could be rotated and translated at the same time in a single fluid interaction operation. Along with doing two operations simultaneously, using two cursors also allowed users to do a single operation with both hands, allowing for more control. Specifying the 3D orientation of an object, for example, requires three degrees-of-freedom, so with the standard two degrees-of-freedom input device, it would take multiple steps to accomplish. They describe a technique in which both hands work in parallel to accomplish the task in a single step.

2.1.4 Evaluation of Bimanual Camera Control

While the described two-handed interactions seemed to work well in practice, formal evaluations were not performed. In 1999 Balakrishnan and Kurtenbach carried out an empirical study of using the non-dominant hand to control a virtual camera while the dominant hand performs other tasks in a virtual 3D scene [4]. Their results showed that bimanual interactions perform better than the status-quo unimanual interaction style, and that it was preferred by users. In one experiment a target was placed on one of the faces of a cube. Users would have to tumble the scene so that the target was on a face of the cube which they could see, and then select it. In the one-handed scenario, clicking the target with the mouse button would select it, and clicking and dragging anywhere else would tumble the scene. In the two-handed technique, participants used their dominant hand to operate a mouse which controlled the selection cursor, while their non-dominant hand operated a second mouse which controlled the camera. It was found that the two-handed technique was 20% faster. In a second experiment, users were given a docking task, in which they had to place a sphere inside of a cube. The sphere could be dragged along the monitors XY plane, so to position the object in a 3D location, the scene would have to be tumbled. As in the selection experiment, there was a one-handed technique, where a single mouse either dragged the object or tumbled the scene, and a two-handed technique, where the dominant hand dragged the object and the non-dominant hand controlled the camera. In this experiment, the two handed technique was 10% faster, but only after practice. Finally, in an informal study, users showed preference of a two handed technique for a 3D painting application. Users painted on the surface of the sphere with their dominant hand, while rotating the sphere with their non-dominant hand.

2.2 Large Display Interfaces

In many work environments large physical displays are essential for visualization, collaboration, and communication. For example, in the design of automobiles, it is important for the artists to have a large scale representation of the model before it is

produced, as “the primary curves that define a car’s style may look perfectly fine at quarter scale but elicit a completely different emotional response at full scale.” [8]. Large scales are also important to keep the design process a collaborative one. Designers need to communicate their ideas and receive feedback from their colleagues and management. Digitizing large displays requires special consideration to their user interfaces, as the standard workstation graphical user interface will not suffice. For example, a toolbar menu along the top border of the screen would not be a good design decision, since the user may not be able to reach, or even see the top of the screen. Chapters 3 and 4 describe interaction techniques for designing virtual models on large surfaces. Previous works on interacting with large displays are now described.

2.2.1 VIDEOPLACE

Some of the earliest work in interactions with large display systems was done by Krueger. who created a system called VIDEOPLACE [29]. Instead of tailoring the system for a specific use, VIDEOPLACE was a deliberately informal and playful arena for the exploration of human interfaces with large displays. The user is in a darkened room with a real time image of their silhouette displayed on a large video projection screen. A camera aimed at the user sends information about what the user is doing. The system responds to the user’s image and motion with interactive graphics, video effects, and sound. Thirty interactions were developed. For example, users could draw on the video screen by holding out a single finger. Closing the hand allowed the hand to be moved without drawing, and holding up all five fingers would erase the drawing. A more playful interaction was implemented using a graphic creature, called CRITTER. CRITTER follows the user’s image, and responds to their actions. If the user holds out the palm of their hand, CRITTER would land on it and attempt to climb up their silhouette. If a finger is held out, CRITTER would dangle from the finger tip. While the VIDEOPLACE system was not made to implement a specific function, the exploration which was done was valuable in understanding how humans will effectively interact with large displays.

In [8], Buxton et al. explore more specific uses of digital large displays. They look at the traditional use of large displays in the automotive industry, and survey new applications that make innovative use of large format electronic displays. A number of current physical tools used by the automotive industry are investigated and the extensions to them in the virtual world which they proposed are now outlined.

2.2.2 ActiveDesk

The drafting table is a good example of a large display which artists work on. The ActiveDesk [7] is essentially a drafting table with a computer image projected from the rear onto the surface. Users work on the surface, which is a digitizing tablet, much like they would work with paper on a traditional drafting table. The large surface allows for large-scale gestures which artists would traditionally use when creating large drawings. Along with the artistic advantages, the large display provides social advantages as well:

> This respect for the traditional skills of designers could play a significant role in the acceptance of this technology. From a sociological perspective, despite the benefits, when automotive designers started using conventional workstation computers, they lost something valuable in the design studio. By its nature, the technology changed some aspects of the social interactions within the studio for the worse. For a variety of reasons, the social mores around a conventional workstation differ from those of a drafting table. [8]

2.2.3 ImmersaDesk

The ImmersaDesk [13] extends the ActiveDesk concept to display stereoscopic images rendered to the viewer. This allows the artist to visualize the 3D data to get a sense of the shape and form. This is similar to the goals of large scale clay models, which are often created before a car is produced.

2.2.4 Portfolio Wall

While the ImmersaDesk and ActiveDesk improve awareness of other's work in a studio, they are intended to be personal work surfaces. Their features cannot replace the traditional wall mounted corkboard. The Portfolio Wall [7] is a system that was designed to serve this purpose. It is an array of 20-30 images projected onto a studio wall where everyone can see them. Designers can drag and drop their drawings or animations from their desktop onto one of the tiles of the Portfolio Wall. The Portfolio wall allows artists to share ideas in a public domain. Gestures and Marking Menus [30] were used to allow users to expand images to full screen, play animations, organize tiles, and compare two versions of a model. A Marking Menu is a type of pop up menu, which allows the user to access the menu in any part of the screen. This is especially effective on a large display since users do not need to worry about getting to specific sections of a large display to evoke commands.

2.2.5 The Cave

The Cave [12] is a display which increases the level of immersion and presence, by surrounding the user with wall sized displays. Rooms with multiple projection surfaces (front, sides, top, and bottom of the room) collectively display a single image that surrounds the viewer. Caves can also provide stereoscopic viewing capabilities through the use of LCD shutter glasses. Viewers can be placed inside the scene that is displayed, such as the interior of the car, giving them a greater sense of presence.

2.2.6 LiveBoard

Large displays are also commonly used during a meetings or seminars. In these environments a central display or drawing surface is commonly used to present and capture ideas. The LiveBoard system [14] is an interactive, stylus-based, large-area display for use in computer-supported meetings. Input is accomplished through a cordless pressure sensitive pen with four buttons, corresponding to different states of

interaction. Supported interaction techniques include drawing, erasing, and changing brushes. Pop up menus were used to execute commands. The interface is directly transferred from physical whiteboards, but by digitizing the display it allows for functionality not possible with the standard whiteboard. For example, users can easily save the notes they create at a meeting or presentation, and subsequently access the records at a later time.

2.2.7 Interactive Wall

Guimbretiere also looked at large sized displays for use as a digital brainstorming tool, and developed an "interactive wall" interaction metaphor [20]. Hand-drawn marks, running applications, 3D visualizations, information structures, and images could be placed on the wall and manipulated with a pen. Their system was not only wall sized, but high resolution. Their implementation consisted of twelve 1028x768 digital projectors which were aligned to create a single image. Input was provided by a wireless digital pen. Like the LiveBoard, it was found that pop up menus were very useful in maintaining fluid interactions. They used FlowMenus [21], which are pop up menus which allow control over continuous parameters. They also combine handwriting with FlowMenus to produce a technique they call "typed Drag and Drop". The system applies handwriting recognition to all user strokes. After completing a word or number, the user can select a command from the FlowMenu to use the text that they just wrote. For example, a user can write "population 746" and then select "attribute" from the FlowMenu, and then drag the text onto a target that could be a city. The city would then have the attribute of having a population of 746. Another technique they introduced is called ZoomScapes. This is a location based scaling method which allows users to scale items by simply moving them into the appropriate area of the screen. Different region would have different scaling factors, and moving objects from one region to another would continuously change the objects' zoom factor according to its new region. This is especially interesting on a high resolution display since text or object details may still be completely visible even at a very small size. While the Interactive Wall techniques demonstrated were in the context

of a digital brainstorming tool, they can be applied to any application on a large, high resolution display.

2.3 Tangible User Interfaces

Traditional graphical user interfaces define a set of graphical interface elements that reside in a purely electronic or virtual form. Status quo input devices such as the mouse or keyboard are primarily used to manipulate these virtual interface elements. Graspable User Interfaces [15], have virtual user interface elements take on physical forms. The Graspable User Interfaces allow direct control of electronic or virtual objects through physical artifacts which act as handles for control. The use of Graspable User Interfaces has a number of advantages. For one, it allows for specialized, context sensitive input devices. Another benefit is that it allows users to apply their existing everyday or specialized skills of physical object manipulation. For example, a skilled sculptor would benefit from a virtual modeling interface where there was a physical clay input device. In Chapter 5 a tangible user interface is presented for manipulating the shape of curves. The following summarizes some previous tangible user interfaces.

2.3.1 GraspDraw

Fitzmaurice et al. [15] implement a graspable user interface, GraspDraw, where the physical artifacts are bricks, which are mounted on an ActiveDesk (see sec. 2.2.2). The bricks act as physical handles to electronic objects. Two bricks were used, one with a push button primarily used for creating new objects. Grasps are registered when a brick is near or on the desktop surface. To release a grasp, the user lifts the brick off of the desktop. A physical tray is used to select different tools or colors. The user “dunks” a brick into a section of the tray to select a particular tool. The application lets users create objects such as lines, circles, rectangles and triangles. The location, orientation, and size of the objects are directly controlled by one or both of the bricks. For example, when creating a rectangle, each brick is attached to diagonally adjacent corners of the rectangle.

By moving the bricks, the position of the corresponding corners are moved as well, which results in the new shape and position of the rectangle. Once created, the objects can be moved, rotated and scaled, again using the bricks.

Ishii describes interaction techniques which couple virtual objects (or “bits”) with graspable physical objects. Their concepts are illustrated with two prototype systems – the metaDESK and transBOARD.

2.3.2 metaDESK

The metaDESK [27] is a Tangible User Interface that creates physical instantiations of the common graphical user interface elements. For example, window handles are implemented as physical “phandles” similar to the bricks of GraspDraw, and menus are implemented as physical trays, similar to those is GraspDraw used to select tools. They created a prototype application of the metaDESK platform called GeoSpace. Physical bricks, which they called “phicons” (as they were a physical representation of icon), were used to represent landmarks of the MIT campus. When a phicon of a landmark was placed on the surface of the desk, a two-dimensional map of the MIT campus would appear on the desk surface beneath the object, so that the position of the phicon was directly on top of its corresponding landmark’s position on the map. By rotating or translating the phicon, the 2D desk view can be transformed in the corresponding manner. By placing another phicon on the desk, the map will rotate and scale such that both phicons are over their corresponding map locations. The user can then rotate, translate and scale simultaneously with either one or both of the phicons.

2.3.3 transBOARD

The transBOARD [27] is a networked digitally enhanced physical whiteboard which can absorb information from the physical world. transBOARD is used just like a white board, as it monitors the activity of tagged pens and erasers with a scanning infrared laser. hyperCARDs, which are barcode-tagged paper cards, are used as physical containers of

strokes. With a hyperCARD attached to the surface of the transBOARD, all strokes are virtually stored within the card. This physical card, which is another example of a phicon, can then be moved to another transBOARD or desktop computer, to bring up the digital strokes which were stored on the card.

The work done in [3] is another example of a tangible user interface, and will be described in section 2.6. It is important to note that in previous tangible interfaces there has always been a direct mapping between the physical artifacts and their virtual representations, whether this be shape, orientation, or location. This thesis (specifically Chapter 5), will describe a tangible user interface where non-linear mappings between the physical and virtual objects are made possible.

2.4 Alternate Curve Design Interfaces

The work presented in this thesis is not the first which attempts to diverge from the status-quo point based curve manipulation techniques. In recognizing the limitations of the existing tools, researchers have been working on tools for sketching and refining curves in a more direct manner. In many of these tools, artists will use a stylus on a digitizing tablet to sketch curves, and the strokes which they make are interpreted by the system as geometric entities. Such tools have been designed for both 2D and 3D environments, and the prototype systems which they were demonstrated in are now discussed.

2.4.1 Free Hand Drawing

Baudel [5] presents an interface for creating and editing curves without the use of control points and tangencies. He notes:

> Although editing splines by specifying control points and tangents may be appropriate for engineers, graphic designers think more in terms of strokes, shapes, and gestures appropriate for editing drawings. [5]

The interface which is presented is based on patterns extracted from graphic artists' existing drawing and editing techniques rather than on a data representation focused on mathematical models and computer management of geometric data. To edit a curve, an artist sketches over it, and this stroke is interpreted as a shape modifier rather than a new curve. Artists can sketch a large curve to get the overall shape of their sketch and then make local changes to refine their work. This top down approach is more appealing to a graphic artist than a bottom up approach consisting of building up control points and tangencies at specific locations.

Cohen extends this free hand drawing interface to a sketching interface for 3D curves [10]. He explains that "the ability to specify non-planar 3D curves is of fundamental importance in 3D modeling and animation systems". While sketched curves may be imprecise by nature, it allows a user to quickly create a curve that is close to the desired result, without requiring any understanding of the underlying mathematics of the curve. This makes for an effective interface for trained artists, as they can apply their existing drawing skills to produce accurate curves. Cohen's work closely matches the work of Baudel, but the ideas are extended to 3D environments. The interface allows artists to draw new curves and overdraw curves to edit their shape by sketching over an existing curve. However the system also allows users to define the shadow of a curve, which defines a non planar surface for the curve to be projected onto. The user can also overdraw a shadow to change the three dimensional shape of its corresponding curve. With this simple sketching interface, artists are able to quickly create and edit three dimensional curves.

2.4.2 Digital French Curves

In the work by Baudel and Cohen, artists must rely solely on their freeform strokes to create sketches. In the real world however, designers often use French curves or sweeps to create or edit curves to bring out a personal style or reflect a corporate standard in all their designs. A French curve is a general term for a pre-defined curve template used to create high quality 2D drawings or sculpt 3D models. Singh [35] presents a user interface for a system in which digital French curves interactively create and sculpt curves and

surfaces that comprise the design of an object. It is a two-handed interface, where the non-dominant hand controls the position and orientation of a French curve with a puck, and the dominant hand creates and edits curves by sketching along the French curve with a stylus. This is an excellent example of a user interface which tailors to the needs and skills of the user:

> French curves emulate a traditional design paradigm, allowing many designers to utilize already acquired skills towards digital curve design. [35]

The system allows for the creation of curves, where a segment of the French curve is specified and gets converted into a design curve. It also allows for the editing of curves, where a French curve interacts with an already existing design curve to alter or extend it along another. In addition, users can edit the position of control points, giving extra control over the final shape. This provides an interface which allows for both quick conceptual sketching, but also for more precise design and editing.

2.4.3 SKETCH

While previously discussed free hand sketching interfaces are used to draw and edit curves, SKETCH [37] is an interface for rapidly conceptualizing and editing approximate 3D scenes. To achieve this, SKETCH uses a purely gestural interface based on simplified line drawings of primitives that allows all operations to be specified within the 3D world. SKETCH is especially useful for artists who would like to quickly build up a conceptual 3D scene. The interface allows users to construct, place, edit and copy 3D geometry with a gestural input interface. The gestures are specified with a three-button mouse, and effort has been made to make them as intuitive as possible. For example, to draw a cube, three perpindicular lines are draw which all meet at a single corner. When two non-axis aligned lines that meet at a point are drawn, a cone is created. While the authors admit that SKETCH is only a tool for initial design, the concepts that they employ could be extended for more precise editing as well.

2.4.4 Chateau

Chateau [25] is a suggestive interface which extends gestural interfaces such as SKETCH. The authors describe a major drawback of gestural interfaces:

> It is difficult for novice users to learn a set of gestures because the user must complete a gesture to see the result, and must start over if it fails. [25]

Chateau alleviates this problem by providing a suggestive user interface. The user gives the system hints through a set if gestures of what they would like to draw, and the system responds with suggestions of subsequent operations in an array of small thumbnails, derived from hints and the overall scene configuration. The user can then complete their drawing by either clicking on one of the thumbnails, or continuing with their own construction. This is very useful as it allows novice users to learn the different gestures by having the system predict and visually express what the remainder of a current gesture would accomplish. Also, because suggestions are merely offered, the suggestive user interface is useful because a set of strokes can either serve as a gesture, or as a subset of a more complex gesture. For example, a single stroke could be a gesture to define a new drawing plane, while extending the gesture by drawing a second perpendicular line would be a gesture to construct a rectangle. Chateau supports the creation of 3D scenes consisting of straight lines and flat polygons. The concepts that have been implemented however could be extended to creating more precise or complex scenes including curved shapes and surfaces.

2.4.5 Teddy

Teddy [26] is a sketching interface which combines freeform sketching work done by Baudel in [5], with the gestural interfaces described by Zeleznik in [37], for easily designing freeform models such as animals and other round objects. The main idea is to use 2D freeform strokes drawn by the user to define the silhouette of an object. The system automatically constructs a 3D polygon surface model based on the strokes. A nice

aspect of the Teddy interface is that while it uses a traditional 2 degree-of-freedom input device such as a mouse or stylus, it does not use the standard interface widgets such as buttons and menus for modeling operations. Users can specify their desired operations through freeform strokes, which allows for a continuous workflow. The interface is not meant to be used for the design of precise objects such as a mechanical piece in an engineering project. Teddy is designed for the rapid construction of approximate, yet expressive, models. It supports major operations such as bending, extruding, cutting, and painting.

2.4.6 3-Draw

Another system which is used to create 3D models is 3-Draw [33]. The 3-Draw computer aided design tool is a tangible user interface for creating systems of 3D curves and editing them using deformations such as stretching, cutting, bending, and erasing. It is a two handed user interface. The non-dominant hand holds a thin rectangular plate with a 6 degrees-of-freedom tracking device attached to it, and it is used to specify the viewing perspective. The dominant hand is used to point or draw 3D curves with a 6 degree-of-freedom stylus. A user can directly draw three dimensional curves of a model as if the model were sitting on the plate that they are holding and positioning with their non-dominant hand. The user can also draw two dimensional curves to specify the shape of their curve and then specify its endpoints on an existing model. This allows an artist to use their sketching skills to accurately draw the shape of a curve, without worrying about its scale, placement, and orientation. This is one example of an interface which allows multiple curves to be integrated into a 3D environment. The tool allows artists to build up three dimensional wireframe models, without worrying about the underlying mathematics of the curves which they are drawing.

2.4.7 Surface Drawing

Schkolne presents a more artistic system, Surface Drawing, for creating organic 3D shapes in a semi-immersive environment [34]. Hand gestures are used to mark 3D space in a virtual environment. The path of the hand is directly realized as geometry, just as the path of a pencil creates a curve on a page. This is another interface which is not focused on creating numerically precise models. Emphasis is placed on "expression and communication", which is desirable for rough conceptual models. The interface also allows for the use of tangible tools. Tongs are used to move and scale the model, and a magnet tool is used to edit a model, by deforming shapes. The interface provides a coupling between body and shape, which is something a sculptor may be used to. Surface Drawing differs from virtual sculpting though, as artists work with surfaces instead of volumes, and artists start with a blank space, rather than an existing mass.

2.5 Digital Tape Drawing

2.5.1 Physical Tape Drawing

We now review the work presented in [2], one of the two main publications which this thesis develops on. The authors present a digital version of the physical tape drawing tool. Tape drawing is the art of creating sketches on large scale upright surfaces using black photographic tape. It is typically used in the automotive industry to create full or near full scale sketches of automobiles. In this industry, concept sketches of cars are traditionally created on large scale upright surfaces to preserve a 1-1 scale factor between the sketch and car. This allows designers to evaluate the curves of the car at an early stage of the design process, as necessary refinements may not be spotted if the work were done at a reduced scale. Tape drawing is achieved by unrolling the tape with one hand and sliding the other hand along the tape while fastening it to the large upright surface (Figure 2.1a). While the concept is quite simple, the artwork which can be created with this tool by an experienced designer is equal to that of any other artistic medium, as

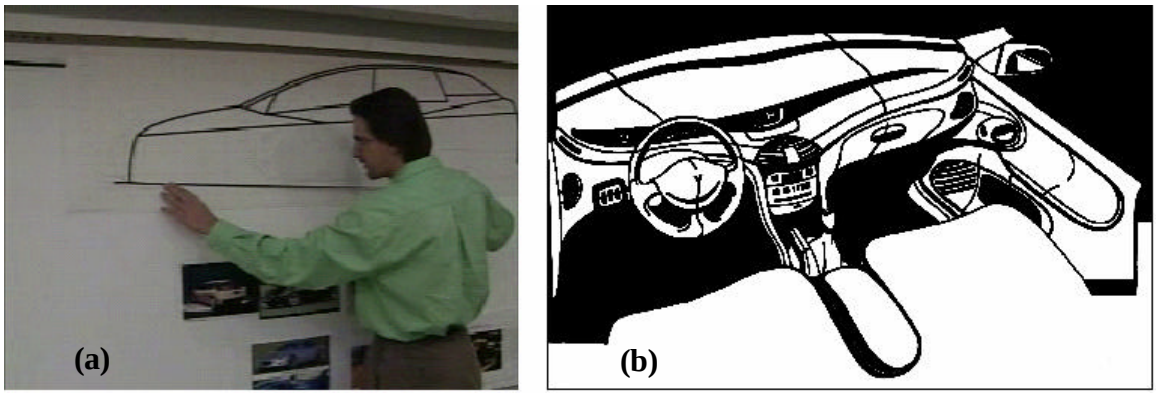

Figure 2.1: Physical tape drawing. (a) An artist using the traditional tape drawing technique. (b) A tape drawing of a car interior using the traditional tape drawing technique. Photos taken from [2].

Figure2.1b illustrates. The advantage of tape drawing is largely due to the large scale size of the sketches. It is very difficult to draw freehand straight lines and smooth curves at such a large scale, and physical aids such as rulers and French curves are not appropriate for a stroke that may span several feet. Drawing with tape facilitates the creation of both long straight lines and smooth curves. In addition, refinements are easily made with the tape drawing tool, as artists can simply lift the tape off the surface, and either relay it or tear it off and replace it with new tape as required.

The advantages of tape drawing have secured its place in automotive design, however there are several problems inherent with working with a physical medium. First of all, the results of a tape drawing artist must be transferred into a computer. Currently, this is done by laboriously digitizing the primary curves of the tape drawing using a hand-held position sensor and then recreating them in a CAD package. This will invariably introduce errors in the electronic version which will have to be identified and corrected. The second major problem is the difficulty of storing and retrieving old drawings. Over time both the purity and accuracy of the drawing will not be maintained, as both the surface and adhesiveness of the tape will deteriorate.

2.5.2 Prototype Implementation

These disadvantage led the authors in [2] to design a prototype implementation of a digital tape drawing system which would alleviate the outlined problems. The prototype

was implemented with a 1280x1024 image back projected onto an 8x6ft screen. This represents a minimal powerwall size which would be used in design studios, and was sufficient to investigate the interaction techniques involved with the tape drawing tool. Two six degree-of freedom electromagnetic trackers were used as input devices, one held in each hand. Each tracker was equipped with a single button.

2.5.3 Interaction Techniques

The actual interaction techniques which were implemented replicated the affordances of the physical tape drawing tool as closely as possible. The left hand controlled the position of a cursor which represented the roll of tape, while the right hand controlled the position of a second cursor which representing the end of the tape. The positions of both cursors were mapped directly from the position of the corresponding hand held trackers, so that they always appeared directly under the user's hands. An unfastened segment of tape is always drawn between the two hands. Moving the hands around effectively moves this unfastened tape segment on the screen, while the distance between the two hands controls the length of the segment.

Just like the physical tool, the digital tool supports two main operations: laying the tape down, and editing. In order to lay the tape down, the left hand presses the button on its tracker, which corresponds to the act of pressing down on the tape against the surface in the physical version. With the button pressed the left hand can then slide towards the right hand to create a curve. This mimics how in the physical version the left hand slides along the segment of tape to fasten it to the surface. When the button is released, the tape currently being laid is cut at the position of the left hand. Just as with the physical tool, straight lines can be drawn be keeping the right hand steady while the left hand moves towards it. Smooth curves can also be drawn by adjusting the position of the right hand while the left hand slides towards it. The right hand thus controls the tangency of curve at the current location of the left hand. Figure 2.2 illustrates these interaction techniques.

The digital tape tool can also edit shapes in the same two manners as the physical tool. While fastening a segment of the tape, the user can press the button on the right hand and then move the left hand backwards, to pull back the segment which they have

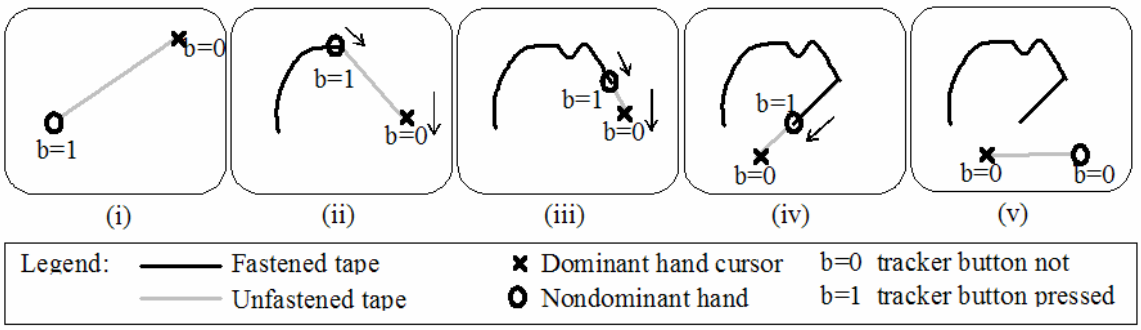

Figure 2.2: Tape drawing techniques. (i) To start taping, the nondominant hand tracker button is pressed. (ii) While moving the dominant hand, the nondominant hand lays down tape as it slides along the unfastened tape segment between the two cursors. Movement of the nondominant hand cursor is constrained to the unfastened tape segment in the direction towards the dominant hand cursor. A long unfastened tape segment results in smooth curves with a gradually changing tangent. (iii). Reducing the length of the unfastened tape segment permits the generation of higher variation curves with a more rapidly changing tangent. The length of the unfastened tape segment can be changed on-the-fly by simply moving the two cursors closer or farther apart. (iv) Switching from taping curves to taping straight lines is achieved by keeping the dominant hand cursor in a fixed position while taping with the nondominant hand. An explicit mode switch is not required. (v) releasing the nondominant hand tracker button cuts the tape.

fastened. They can then relay the segment if desired. Users can also cut a segment of tape, which is done by simply clicking on two points of a fastened segment, which specifies the cut points. The tape segment between these two points is removed.

By creating a digital tool which closely matches the physical tool which artists currently use, artists are allowed to utilize their skills, while the advantages of working on a digital medium can be taken advantage of. For example, drawings can easily be stored and retrieved without deterioration, and sent to different studio locations. Their prototype was demonstrated to a number of professional designers. They found that within minutes, the artists were creating drawings which were clearly superior to those created by the system developers, showing that the artists were indeed able to transfer their skills to the new system.

2.6 ShapeTape

The second core publication which this thesis develops on is an exploration of the use of a new type of input device to interactively manipulate curves and surfaces [3]. The input device used is called ShapeTape, a continuous bend and twist sensitive strip that is used by both hands. Like many sketching interfaces, the authors investigated ways to interact with virtual curves and surfaces without having to work with the underlying mathematical definitions which control their shape. However instead of using a sketching interface a tangible interface was used: users could directly control the shape of a curve by controlling the shape of the input device. With the use of ShapeTape as the main input device, they attempted to combine physical and virtual curve and surface modeling. The main inspiration again comes from the automotive industry, where flexible steels are sculpted into specific curves and then used to sweep out a curved surface of a clay model (Figure 2.3).

Figure 2.3: Physical curved tools. An artists sweeps out the surface of a clay model using a steel curve

2.6.1 Prototype Implementation

The paper presents a prototype system where ShapeTape is used to perform basic curve and surface creation and manipulation operations. Since users would have both hands on the input device when using the system, a status-quo mouse/keyboard interface for secondary functions, such as camera controls and command executions, would not be appropriate. To reduce the need of users removing their hands from the input device, supplemental input devices were used. The ShapeTape was augmented with a 6 degrees-of-freedom tracker so its location could be tracked along with its shape, and four buttons, which could be pressed while holding the physical curve input device. A 2 degrees-of-freedom puck on a Wacom digitizing tablet was placed on the floor, and used as a foot mouse by the right foot, to rotate the 3D virtual scene. Two foot pedals (a rocker pedal and a momentary pedal) were operated by the left foot for extra functionality.

2.6.2 Interaction Techniques

The system implemented a number of curve and surface manipulation functions using different tools. The keyboard was used to switch between tools. Instead of creating a seamless interface the authors concentrated on how each individual tool or function would be used by the ShapeTape, and left ideas for how to switch between these tools to the discussion.

By using a physical curve as an input device, virtual curves could easily be created. The shape of the virtual tapecurve was controlled by the ShapeTape, and its position and orientation controlled by the tracker. Pressing button 1 resulted in a snapshot copy of the tape curve being placed in the 3D environment. This was referred to as “baking” the tapecurve. To ensure the correct shape was created, users could first lock the shape of the virtual curve by clicking a foot pedal button.

The prototype also supported the construction of surfaces. The first method uses the lofting technique, which is the construction of a surface which passes over a series of profile curves. To use this lofting tool, the user bakes an initial curve from which the system creates a lofted surface. The user can then define subsequent profiles of the surface by baking addition curves. An interesting aspect of this technique is that the

lofted surface is dynamically updated in realtime as the user shapes each profile curve. This visual feedback is possible since the user has direct control over the entire shape of the curve, and not just a single control point or segment.

A revolve tool was also developed for creating surfaces. The user first specifies an initial profile curve and then presses one of three buttons, corresponding to the x, y and z axes. The profile curve is revolved about the selected axis. The resulting surface can subsequently be manipulated in a very interactive manner, as the initial profile curve is still controlled by the ShapeTape. Again the surface is dynamically updated as the user manipulates the shape of the profile curve.

A final method for creating surfaces with was implemented was an extrude tool. An initial profile curve is again first baked. The ShapeTape is then used to define a curve which is the path for the profile curve to sweep out. As with the other surfacing tools, the path curve can be interactively manipulated, with the surface being dynamically updated.

Along with creating curves and surfaces, tools used to deform existing surfaces were implemented. ShapeTape was used to manipulated "wires", which is a geometric deformation technique based on space curves [36]. A wire is a curve whose manipulation deforms the surface of an associated object near the wire curve. The deformations are based on the relative deviation between the wire curve and its corresponding reference curve. The shape, position, orientation, and twist of the wire curve can be deformed, making it and appealing technique for use with the ShapeTape. Three styles of interaction were implemented for the "wire tool". In the first, the ShapeTape controlled the bend and orientation of the wire, while the position of the wire curve remained static. In the second technique, the wire curve could also be translated, allowing for what they call "traveling wrinkle deformations". In the third technique, the wire could be twisted as well, allowing surfaces to be deformed in a manner similar to pinching.

It was found that the one-to-one mappings between the ShapeTape and virtual curves allowed for ease of use and rapid learning. The manner which the physical tool controls virtual curves is immediately obvious. One observation was that the prototype allowed different shapes and effects to be quickly attained. This is a property especially suitable for conceptual modeling, where designers need to quickly explore form, shape, and size of potential models.

Chapter 3

Creating 3D Models by Integrating 2D Tape Drawings[1]

3.1 Introduction

In this chapter an alternate interface for 3D modeling for use on large scale displays will be presented. Section 2.5 described a system in which single 2D tape drawings sketches could be designed on large scale displays [2]. We extend this work by describing interaction techniques and a prototype system for creating and integrating multiple 2D sketches into a single 3D model. While the main goal of the system is to provide a user interface for creating and integrating curves of a 3D model, focus will be put on developing appropriate interaction techniques specifically for large displays. The tape drawing tool will be used for drawing curves, and so like the work in [2], the system will be tailored for automotive design. The system designed in [2], retained much of the fluidity and affordances of the physical technique, including working on a large scale display, while providing the advantages inherent in using electronic media. In discussions with designers at various auto design studios during the development and subsequent demonstration of that prototype system, a wealth of user feedback was obtained, and two main observations can be made from the feedback.

First, large scale (greater than 8x6 feet) electronic projection displays are being widely deployed in most design studios [8]. However, most of these displays are

[1] The material presented in this chapter is based on the following publication: Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton.. *Interaction techniques for 3D modeling on large displays*. in *ACM I3DG 1999 Symposium on Interactive 3D Graphics*. 2001. New York, NY p. 17-23.

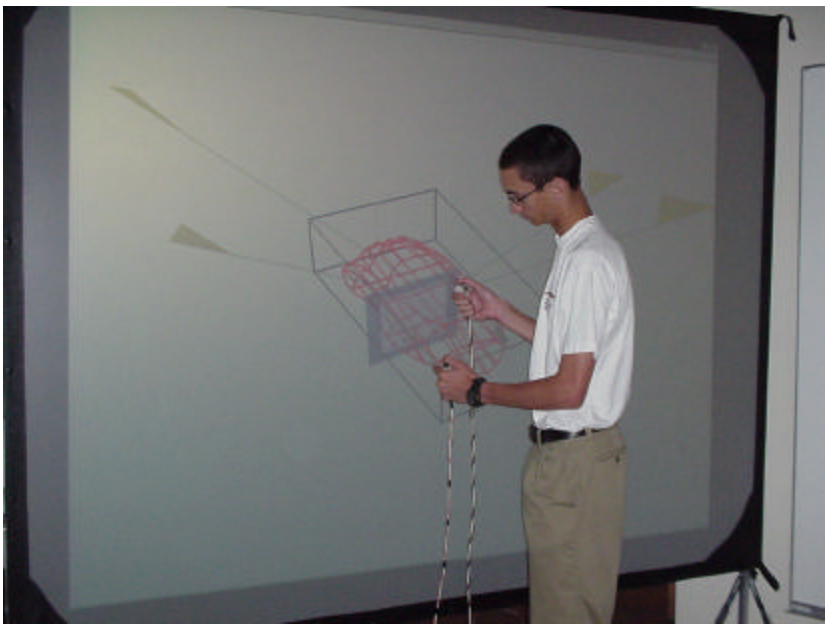

Figure 3.1: 3D modeling on a large display.

currently used as passive output devices for evaluating final renderings of designs at large scale and are not used in an interactive manner in the design creation process. One reason for this is that user interfaces for current sketching and modeling software are designed for desktop scale interaction and are often awkward when used interactively on a large display. The previous tape drawing system, designed from scratch to work on a large display, was seen by the designers as a start towards being able to work interactively at this scale in the early stages of the design process. Clearly there is a need for sketching and modeling applications, with capabilities beyond that of the initial tape drawing prototype, whose user interface is explicitly designed to work with large displays.

Second, traditional tape drawings, and the previous system, only allow for the creation of planar 2D drawings. Since these 2D drawings eventually form the basis of the final 3D model, the designers felt that it would be valuable if they could create the 2D profiles while the underlying 3D model was being simultaneously updated, and easily transition between the 2D and 3D views. Interestingly, the very designers who rejected the use of conventional software in favor of drawing with physical tape, when given an electronic analogue of tape drawing were asking for capabilities that the original physical technique could not provide.

Based on these observations, we have extended the initial prototype. The result, whose design and implementation is discussed in this chapter, is an interface for 3D modeling that integrates several interesting concepts, including: large scale interaction, 2D construction planes spatially integrated in a 3D volume, enhanced orthographic views, smooth transitions between 2D and 3D views, tape drawing as the primary curve and line creation technique, visual viewpoint markers, and continuous two-handed interaction (Figure 3.1). The primary goal for developing the current system was not so much to create a system for actual deployment, but rather a vehicle to explore and integrate various user interface techniques suitable for 3D modeling on large scale displays.

3.2 System Hardware

3.2.1 Display

The implementation uses a Hughes/JVC G1000 digital projector with a true 1280x1024 image back projected onto a collapsible 8x6ft screen. The size of the screen and projector ensures that the system is portable, which is important so that the system can be demonstrated at different auto design studios. The 8x6ft screen represents the minimum size for large displays used in the auto industry, and is sufficient for us to implement interaction techniques whose scale of interaction is vastly different from desktop scale interaction. Similar display sizes have also been used in research systems such as Krueger's VIDEOPLACE [29].

3.2.2 Input Devices

Since our system heavily utilizes two-handed interaction techniques, we need to be able to sense the position of both hands on the display surface. There are potentially several solutions to this sensing problem. These include optical tracking techniques [14], the use of a transparent digitizing tablet on the display surface, and electromagnetic/ultrasonic trackers. Our prototype uses an Ascension Flock-of-Birds six degree-of-freedom tracker

held in each hand. Each tracker is augmented with a single momentary switch. We only use two translational degrees-of-freedom (up/down and left/right) of the tracker in our prototype. Two cursors on the screen indicate each tracker's position.

3.3 Interaction Techniques

3.3.1 2D Construction Planes Spatially Integrated in a 3D Volume

The basic interaction model of our system is to allow for the creation of 3D models by drawing appropriate 2D profile curves. Status-quo modeling applications provide this functionality by having separate 2D orthographic views of the 3D model on which the profile curves are drawn. However, because these orthographic views are typically shown in their own windows separate from the underlying 3D model, it is not easy for a user to see the correspondence between these different views. While users ultimately appear to be able to integrate these views, this is achieved only after much experience with such views. In our system, in order to maintain correspondence with the underlying 3D model being created and to ease the learning process, the 2D profile curves are created on construction planes displayed as sides of a cuboid which acts as a bounding volume for the 3D model within it (Figure 3.2). Earlier work by Sachs [33] similarly use construction planes in a 3D working volume to create 3D models. Our prototype allows for three primary construction planes, representing the top, side, and front views of the 3D model being created (Figure 3.2).

Switching between construction planes is achieved by clicking, using the dominant hand tracker's cursor, on the coloured tab on the corner of the plane. When switching between planes, the previous plane moves away to the edge of the cuboid, while the newly active construction plane moves to the position it was in when it was last active. This reduces clutter in the cuboid since only the active plane is visible while the others are pushed off to the periphery. This “construction plane memory” feature allows the user to switch between different construction planes while retaining the precise location of each plane for later recall.

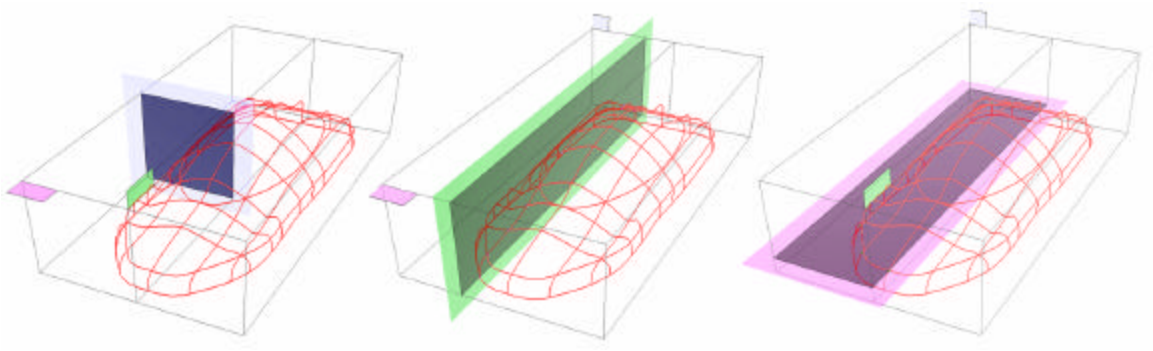

Figure 3.2: 2D Construction planes in 3D perspective view. The left image shows the front/back plane, the middle image shows the side plane, and the right image shows the top/bottom (horizontal) plane. Although not visible in this picture, the intersection points between the plane and the 3D model are highlighted as each plane moves through the cuboid space.

3.3.2 Symmetric Reflections

Depending on the type of 3D model being created, one or more of the construction planes may allow for automatic symmetric reflections about the medial axis. For example, when modeling a car, the top/bottom horizontal construction plane allows for symmetry about the medial axis so that whatever is drawn on the left side is mirrored onto the right.

3.3.3 Intersection Points on Construction Plane

The construction planes in our system are semi-transparent so that the 3D model is not obscured as the plane is moved back and forth through the cuboid. To highlight the intersection of the active construction plane with the 3D model and to reinforce the construction plane's position, we display the relevant intersection points on the plane. These points are dynamically updated as the plane is moved.

3.3.4 Enhanced Orthographic Views

While we can create 2D profile curves on the relevant construction planes while in a 3D perspective view, it is often more accurate to draw these curves in an orthographic view. In status-quo modeling applications, orthographic views are truly 2D views of the 3D scene. All curves in the 2D view have the same "weight" and are visually not disambiguated regardless of their distance away from the camera. To create and position a new curve in 3D space the user typically has to work in two or more orthographic views. This is the reason for the common configuration of having top, side, and front orthographic views simultaneously visible in most applications. While this configuration is adequate, it results in dividing the user's attention between several views. This problem may be exacerbated when working on a large screen since the user cannot easily view the entire display when up close.

To reduce this divided attention problem and to allow for a single orthographic view to be usable when displayed full size across the entire display, we developed some enhancements to the traditional orthographic view. First, the position of the appropriate construction plane within the cuboid determines the depth position at which new curves are drawn. When the model is viewed orthographically, we display the curves in that view at different grey levels depending on their distance from the current construction plane's position (Figure 3.3). This essentially provides another dimension of information when in orthographic view, albeit at a coarse granularity, that is not typically available in status-quo applications. Furthermore, we can move the position of the construction plane while still in orthographic view by clicking on any of the curves (at which point the

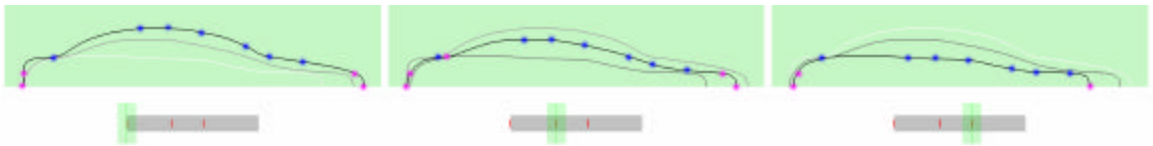

Figure 3.3: Enhanced orthographic views. The curves are drawn in different grey levels depending on how far they are in depth from the position of the construction plane. Users can navigate between curves either by selecting the curves themselves or using the slider below. Points of intersection between the curves and other construction planes are indicated as well (Note that these points have been enlarged in this image for clarity).

selected curve becomes black and the grey levels on the other curves are adjusted accordingly). Alternatively, a slider widget at the bottom of the screen (Figure 3.3) allows for positioning of the construction plane. This slider, which is operated by the dominant hand, can work in either continuous mode through the entire space of the cuboid, or in discrete mode where it snaps between the positions of the existing curves. On the slider, these curve positions are represented by tick marks. This discrete sliding mode provides an alternative way to easily move between curves for editing.

Secondly, as in the perspective view, we show the intersection points between the current construction plane with the 3D model. These intersection points are dynamically updated if we move the position of the construction plane. Further, the points are colour coded to indicate the orientation of the curves that are being intersected. This is useful when creating new curves that are to intersect two or more existing curves which may be located in different planes. We have also found these intersection points to be useful in orthographic view as they act as portholes into the perspective 3D model, providing a coupling between the 2D and 3D views even if only one view is visible at a time. We note that the advantages of our enhanced orthographic views are not restricted to large scale displays, but would likely be useful on standard desktop scale displays as well.

3.3.5 Animated Transitions Between Views

The two enhancements described in the previous subsection are examples of our efforts to provide a strong connection between 2D and 3D views in our system. In combination, these two enhancements allow for the user to work in a single orthographic view when needed, without requiring the other views to be present in order to maintain context with the 3D model. However, users will often still need to go back to the 3D view to evaluate the model in its entirety. To visually indicate the relationship between the 2D and 3D views as we transition between them, we smoothly animate the transition between orthographic and perspective views (Figure 3.4). We not only animate the window outline as is commonly done in modern 2D window managers, but we also animate the transition of the underlying data. This prevents the often jarring immediate switch between views that is present in status-quo modeling software. The smooth transition

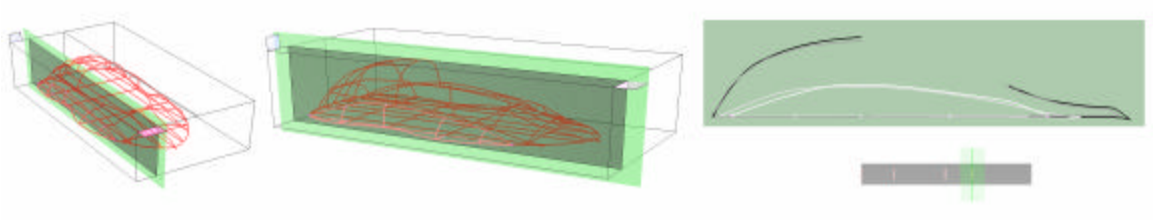

Figure 3.4: Animated transition between 3D perspective and orthographic views. The image on the left is the system in perspective view, the image in the middle was captured midway during the animated transition, and the image on the right is the final enhanced orthographic view. The data is animated along with the bounding cuboid volume during the transition. This helps the user maintain context when switching back and forth between views

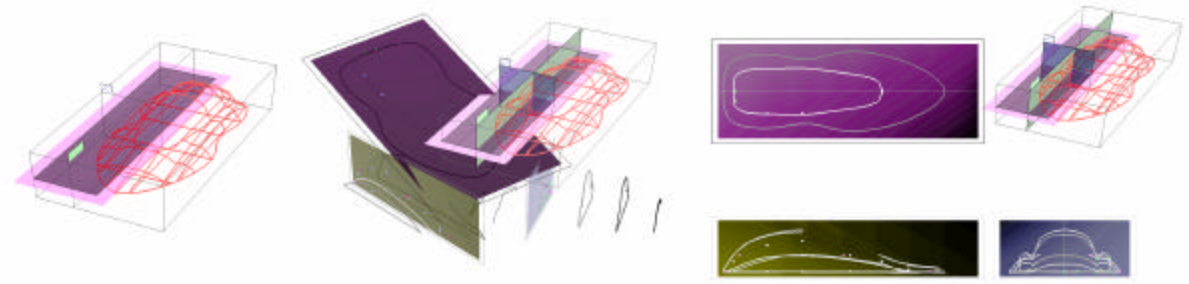

Figure 3.5: Animated transition between 3D perspective and multiple views (three orthographic and one perspective). The image on the left is the system in perspective view, the image in the middle was captured midway during the animated transtition, and the image on the right is the final multiple view. In the multiple view, moving the construction planes in the perpective view dynamically updates the intersection points and grey levels on the three enhanced orthographic views

provides yet another cue for the user to maintain the appropriate spatial correspondence between the two views.

3.3.6 Multiple Views

Apart from separate perspective and enhanced orthographic views, our system also supports a multiple view layout (Figure 3.5, on the right), similar to the four view layout of status-quo systems, where the perspective view is shown along with three orthographic views. Our system is unique in that the orthographic views are enhanced as described above. Sliding the construction planes in the perspective view, either continuously

through the cuboid space or discretely between the profile curves, updates the grey levels and intersection points in the appropriate enhanced orthographic view. This facilitates inspection of the model. Again, to maintain 2D/3D spatial integration, transitions between multiple views and single views are smoothly animated (Figure 3.5).

3.3.7 Marking Menus

Activating the transition between 2D and 3D views, switching between discrete and continuous sliding, and other command based actions are accomplished by making a selection on a marking menu [30]. On large displays, conventional menu bars at the edges of the screen are cumbersome to operate given the large movements required of the user. Marking menus, however, have the advantage of appearing at the location of the dominant hand's cursor, requiring very little movement from the user and are thus particularly suitable for use on large displays. Also, once users have learnt the location of the various menu items in the menu, they simply have to make a mark in the direction of that item in order to activate it and do not have to wait for the menu to be displayed. In our system, we activate the system wide marking menu by making a mark using the dominant hand tracker with its button pressed.

3.3.8 Tape Drawing

As earlier discussed, the motivation for the present work was an earlier digital tape drawing system which mimicked the functionality of traditional tape drawing techniques. We retain this digital tape drawing technique as the primary method for creating and editing curves in the present system. Section 2.5 gives a description of the functionality of the digital tape drawing tool. We note that there are some important differences between our electronic tape drawing tool and the traditional technique using real tape. First, in our current system the collapsible screen we use does not have a rigid surface. As such, unlike in traditional tape drawing, users cannot press against the drawing surface (i.e., the screen). While this is somewhat of a drawback, we have found that users are able to adapt by holding the input devices just above the surface of the screen. The use of

a rigid screen would solve this problem. However, this would be at the expense of system portability that is crucial at this stage of our exploration in order for us to be able to demonstrate our system at different design studios world wide. Another difference between the electronic and traditional media is that in the traditional technique the user gets kinesthetic feedback in the form of tension in the tape, whereas the electronic version provides only visual feedback. While we do not believe that the lack of physical tension in the electronic version is a serious handicap, we have experimented with providing physical tension using spring loaded cords (much like a spring loaded tape measure). Our initial efforts indicate that providing physical tension in this manner is rather different from the tension in real tape and therefore detracts from the drawing task. In some sense, it appears that it is better to not provide any kinesthetic feedback than to provide feedback that is inferior or vastly different to that of real tape.

3.3.9 Camera Control

When in 3D perspective view, we support the usual camera control operations of tumble, pan, and zoom. Based on the results of earlier research which showed benefits in operating the camera with the non-dominant hand [4], and in line with theoretical models of bimanual interaction [19], we assign the non-dominant hand to control tumbling of the camera, freeing the dominant hand for other operations. Pressing the non-dominant hand tracker button invokes camera tumbling which is controlled by movement of the non-dominant hand. Panning and zooming are two handed operations. With both tracker buttons pressed, moving the non-dominant hand pans the camera, while moving the dominant hand zooms the camera. Other two handed camera controls have been previously explored in 2D by Kurtenbach et. al. [31] and in 3D by Zeleznik et. al. [38, 39].

3.3.10 Viewpoint Markers

When creating 3D models, designers typically go back and forth between a few particular views of the scene. In status-quo systems, users are given the ability to define viewpoints

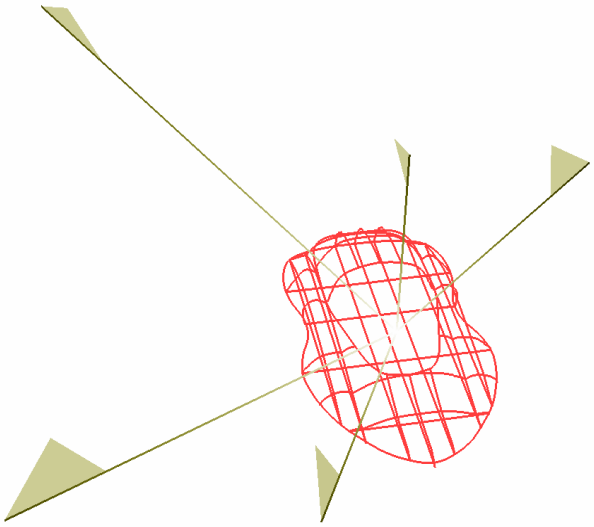

Figure 3.6: Viewpoint markers. Viewpoints can be saved using a marking menu, at which point a viewpoint marker flag is added to the scene to visually indicate the direction of that camera view. Clicking on any of the viewpoint marker flags transitions to that view with a smooth animation.

and switch between them via a menu or hotkeys. Each preset view is usually abstractly named and no visual indication is provided as to what each view is until the user actually selects that view. In our system, we increase the directness in which these special viewpoints can be set and selected using a new widget we call “viewpoint markers”. Essentially, when the user wants to save a particular view for later recall, they make a selection on the system's marking menu. A visual marker flag is then drawn in the scene, pointing in the direction of that camera view (Figure 3.6). The directionality of each viewpoint marker provides an immediate visual indication to the user as to the approximate view each marker corresponds to, without having to actually select that view. The user can return to any of these views simply by clicking on the appropriate viewpoint marker. Transitions between views are always smoothly animated to maintain visual continuity. Other techniques for saving and restoring camera viewpoints were previously explored by Zeleznik and Forsberg [39].

3.4 Discussion

While no formal user tested was done on this system, it was shown to several designers who visited our lab. From their reactions and our own experiences in using the system while under development, we feel that some of the techniques are “clear winners” while other areas could use improvement.

Firstly, we believe that the use of tape drawing as the curve and line creation tool significantly reduces the complexity of creating curves that is present in status quo applications. Very smooth curves, with the desired curvature, are easily created with this technique. Also, users only have to use a single tool to create both lines and curves.

Another aspect of our system that works well is the use of animated transitions between views, and our enhanced orthographic views. Both of these increase the coupling between 2D and 3D views of the underlying 3D model and allows users to easily work in 2D without overly sacrificing their ability to comprehend the 3D scene.

The use of two handed input throughout the system builds upon previous research [4, 19, 24] which has shown the benefits of bimanual interaction when designed properly. Our non-dominant hand camera controls in particular are based on earlier empirical work [4].

Working on a large display poses the problem of not being able to easily operate a keyboard. As such, we are restricted to a few buttons on whatever hand held trackers we use. In our case we designed the entire system to work with just two buttons, one on each tracker, and used marking menus to access commands. In order to increase the number of states that can be triggered by just two buttons, we have considered the idea of using the sequence in which the two buttons are pressed as an additional state.

The prototype was implemented on a display that is approximately 1/3 the size of a real car. We feel that working with truly 1-1 scale displays and multiple views will result in yet another set of interaction issues that will have to be addressed. For example, imagine having the multiple view layout in our system where each view is full 1-1 size. This would require the user to possibly walk around the display in order to interact with different parts of it. The challenges posed by this setup are yet to be investigated.

3.5 Summary

The system outlined in this chapter achieves the two main goals of this thesis. First, the tape drawing tool allows artists to create curves based on their shape, and not on their underlying mathematics. This allows artists to use their existing skills when working with the system, and does not require any understanding of the underlying mathematics of the curves. Secondly it provides an interface for artists to integrate the curves which they draw into a single 3D environment, allowing them to survey the final model as they build it up. Along with achieving these main goals, the design rationale throughout this work was to maximize the potential of the large display and promote integration between 2D and 3D views of the data. At the same time we sought to minimize the level of intrusiveness of user interface components. As such, only user interface widgets that are absolutely necessary are displayed at any one time, resulting in a system that emphasizes the artwork over user interface components. In designing the interaction techniques, rather than inventing everything from scratch, we chose to borrow from previous work and modify or enhance the techniques as required. Some techniques like marking menus and camera controls we use as is, others like orthographic views and animated transitions are enhanced, while some techniques like viewpoint markers are new. Thus, the contribution is not so much in the individual techniques themselves, but in the combination of these techniques into a fluid system for 3D modeling.

The main limitation of this system is that only planar curves, i.e. those that exist on a flat surface can be created. However, the principle curves of an automobile, or any other 3D model, are generally non-planar. The following chapter will present an extension of this system to allow for construction planes of arbitrary orientation. Further, the ability to create 3D, non-planar curves will be supported.

Chapter 4

Extending Tape Drawing to Non-Planar 3D Curve Creation[1]

4.1 Introduction

The previous chapter presents a system for 3D modeling on large scale displays. While the system provides an interface for creating and integrating curves into a 3D environment, focus is put on developing effective user interaction techniques specifically for large scale displays. While this work allowed us to explore the challenges of combining 2D tape drawing with the simultaneous creation and management of a 3D model on a large display, the system was fundamentally limited in that the 3D model was made up of 2D planar curves. Real car models require non-planar 3D curves in order to fully define their primary shape. In this chapter, we extend the system with the high-level goal of the creation of the largely non-planar 3D primary curves of a real car design. The digital tape drawing tool will still be used for creating both planar and non-planar curves. In effect, we now have the capability to completely eliminate the traditionally separate steps of interpreting and combining various 2D tape drawings to form a 3D car model. We also present results of a study observing users performing a representative task of creating the primary curves of a car using the system (Figure 4.1).

[1] The material presented in this chapter is based on the following publication: Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton. *Creating principal 3D curves with digital tape drawing*. in *ACM CHI 2002 Conference on Human Factors in Computing Systems*. 2002 p. 121-128.

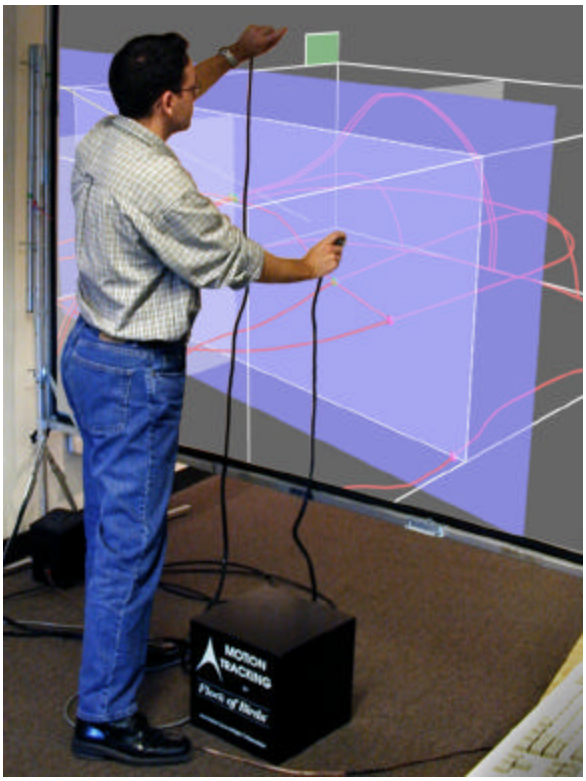

Figure 4.1: 3D digital tape drawing system.

4.2 System Hardware

4.2.1 Display

As described in Chapter 3, our implementation uses a Hughes/JVC G1000 digital projector with a true 1280x1024 image back projected onto a collapsible and portable 8x6ft screen. While the portability of this 8x6 ft screen allows us to easily take the system to various studios for demonstration purposes, it only allows for midsize cars to be displayed at about half scale. As such, in addition to this default display configuration, we have also explored using a 16x6 ft screen with imagery created by tiling two projectors. This size allows for a 1-1 scale display of a midsize car, which is representative of the scale at which traditional tape drawings are created.

4.2.2 Input Devices

The system still utilizes two-handed interaction techniques, so the position of both hands on the display surface needs to be sensed. The use of limited range magnetic trackers were sufficient for the 8x6 ft screen, however, to accommodate our larger screen we now use extended range Ascension Bird trackers held in each hand.

4.3 Interaction Techniques

Before delving into the details of the system which supports true 3D curve creation, we review the core characteristics of the system described in Chapter 3 upon which the present system extends. These characteristics are:

- We retain the interaction style of the traditional tape drawing technique for creating curves. This allows for smoothly varying, continuous 2D curves to be created easily at this scale, and within a single tool [2].
- The 2D tape drawings can only be performed from the three canonical orthogonal viewpoints – side, top, and front. However, rather than creating these in separate 2D views as is traditionally done, the 2D drawing planes are spatially integrated into a 3D working volume (see Figure 4.2). This provides the user with information on the correspondence between different viewpoints within a single integrated display.
- The drawing depth within this integrated view can be dynamically changed. Effectively, we draw on the sides of a cube, but since we can vary the depth of these drawing planes, the 2D curves are thus created within the 3D volume. This results in a 3D model, albeit one made up only of planar 2D curves.
- We only have one view of the model at a time. This maximizes the amount of screen real estate available for viewing the model, and doesn't divide attention between separate views.
- Smooth animated transitions when moving between perspective and orthographic viewpoints of the 3D model allow for the user to visually retain and understand the relationship between these views.

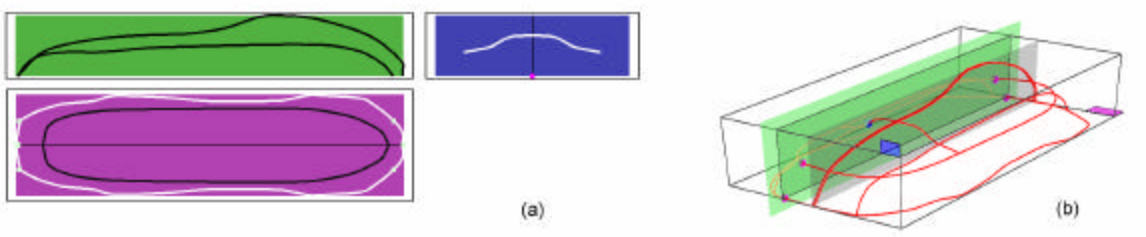

Figure 4.2: (a) Separate orthographic views (b) Views integrated into cubic volume.

• Conventional user interface widgets such as menu bars at the edges of the screen are largely untenable when working close-up on large displays. As a result all our interface widgets either appear on parts of the model itself, or can be popped up where the user's hands are. We support fast popup menu and command access using Marking Menus [30].

These basic concepts as demonstrated in the previous chapter were sufficient to allow us to explore how the 2D tape drawing technique could be extended to create simple 3D models. However, it was fundamentally limited in that it could only create 2D planar curves. Now, we discuss how we have extended and added to these concepts to enable both the creation of a wider array of models with non-planar 3D primary curves and additional support for the workflow required to support this process.

4.3.1 3D Curve Creation

Our first major extension to the system is a mechanism for creating non-planar 3D curves. Other systems have allowed users to articulate 3D curves directly by inputting a 3D hand gesture [33, 34]. In practice, expressing, visualizing and positioning 3D curves using these freehand methods with precision is difficult. We take an alternative approach that is based on our user group being skilled at drawing precise 2D curves using tape drawing and use repeated applications of this technique to generate precise 3D curves.

We draw on previous work by Cohen et. al. [10] who describe a system for creating 3D curves by first drawing a new curve in a plane, and then drawing the curve's "shadow" [22]. This shadow or depth curve essentially defines the shape of the curve in the third dimension. This technique is nice in that it leverages off artists' drawing skills

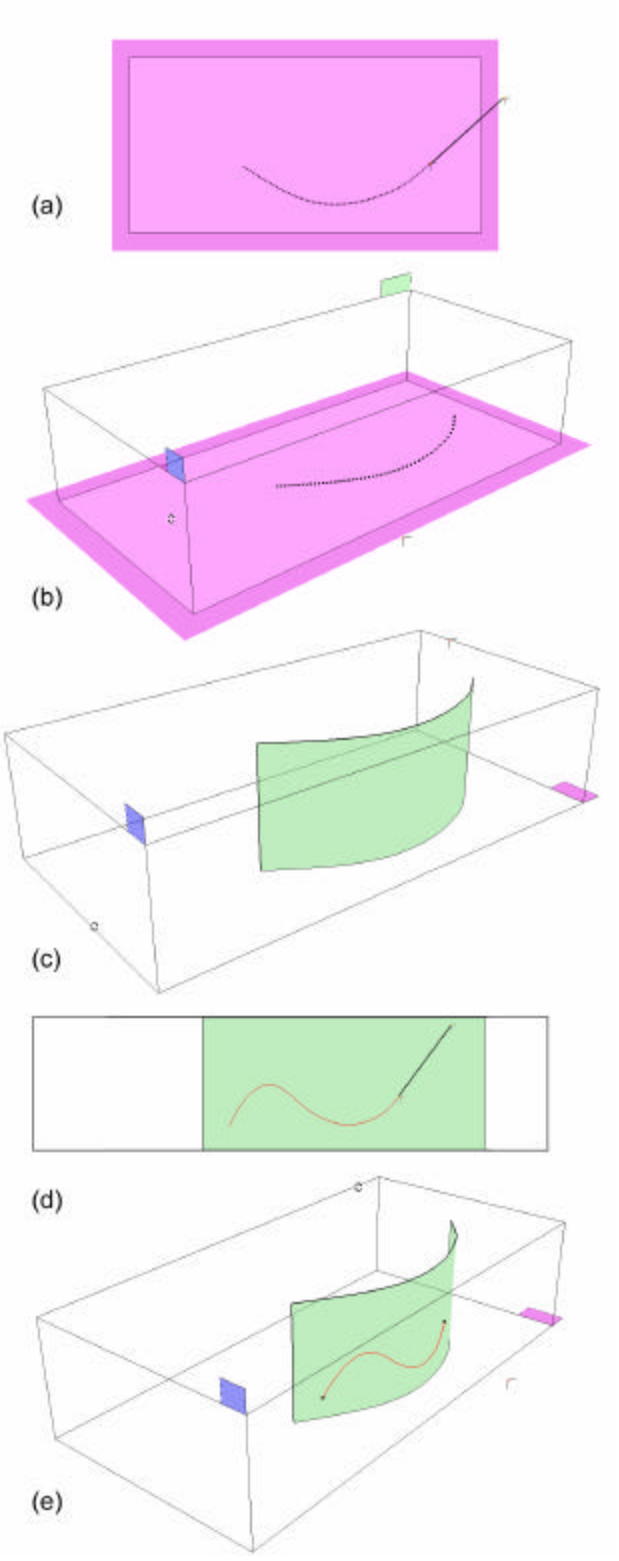

Figure 4.3: Creating a 3D curve: (a) create depth curve; (b, c) show the depth plane defined by depth curve; (d) from orthographic view, drawing and projecting a curve onto the depth plane; (e) resulting 3D curve.

rather than requiring them to learn and operate a more abstract technique for curve creation. In our system, we flip the order of these operations such that the depth curve is created first. This has two advantages over the technique in [10]. First, while we can simply create a new depth curve (Figure 4.3a, b), we can also select a preexisting curve in the scene to serve as a depth curve. This is important since when building up a car model, curves are typically drawn relative to those already in place. Second, after the depth curve is created or selected, we can then pick the drawing plane on which we will create the final curve, and then display the 3D surface onto which this final curve will be projected (Figure 4.3c). This provides extra visual feedback to the user as to where their final curve will lie in depth. The final curve may now be drawn in the appropriate orthographic view drawing plane (Figure 4.3d). This curve is then automatically projected onto the depth surface resulting in a 3D curve (Figure 4.3e). All curve creation is done using the tape drawing tool [2].

4.3.2 Orthographic To Perspective Tumbling

One of the challenges in integrating the tape drawing technique with 3D model creation is that tape drawing is inherently a 2D technique, and all curves are initially drawn on 2D drawing planes. However, the model being created is in 3D. Hence there is the need to switch back and forth between orthographic views of the 2D drawing plane(s) and the perspective view that is necessary for inspecting the 3D model. Commercial 3D modeling packages typically display three orthographic views and one perspective view in four separate windows on the display. Given that one of our system's goals is to maintain the large scale that is critical in car design, dividing up the display in this manner would undesirably reduce the viewing area of the model. Further, users' attention would be divided between the multiple views. As described in Chapter 3, we attempted to solve this problem by switching between fullscreen orthographic and perspective views via a marking menu issued command, while animating the transition between views in order to maintain visual correspondence between the data in the different views. While this animation eliminated a visually jarring context switch, there still remained a visually discrete step when the orthographic view was replaced by a perspective view.

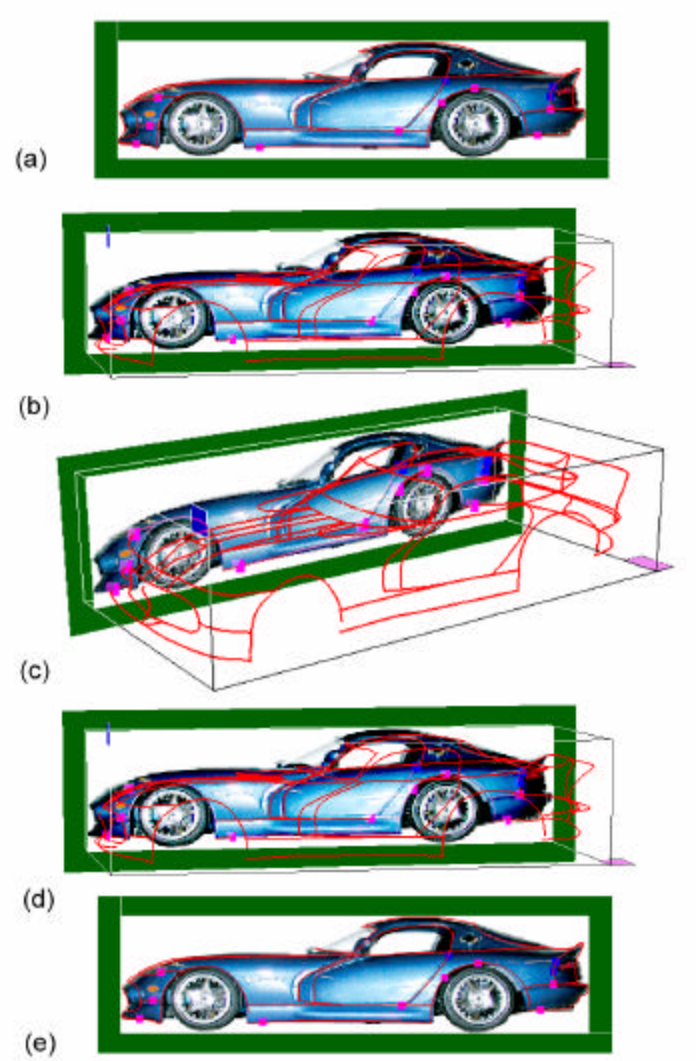

Figure 4.4: OrthoTumble

We improve on this in two ways. First, we smoothly interpolate the viewing matrix from the orthographic view to the perspective view over a 50 frame duration. Thus, the transition truly is gradual, rather than animated but distinctly switched as in the previous implementation.

Second, we provide the user with a more direct method, which we call OrthoTumble, to invoke this transition and to subsequently control the camera in the perspective view. Most 3D modeling applications have a "tumble" camera control that gives the user the sense of controlling a two degree-of-freedom turntable on which the 3D model sits. Previous research [4] has shown that performing the tumble operation in the non-dominant hand while the dominant hand manipulates the model can result in more facile

interaction with significant performance advantages. Building on this prior work, our system's OrthoTumble tool is invoked by using the non-dominant hand to click and drag outside the 3D model space (Figure 4.4a). While the user drags, or OrthoTumbles, the view transitions from orthographic to perspective (Figure 4.4b). If the drag is continued, the tool acts like the typical tumble tool, allowing for the inspection of the 3D model in the perspective view (Figure 4.4c). When in the perspective view, the user can change the active drawing plane using their dominant hand. Finally, when the drag operation is terminated, the view transitions back to the closest orthographic view – either the front or back – of the active drawing plane (Figure 4.4d, e). Transitioning back to the closest view, rather than a default front view, allows the user to flip back and forth between the front and back of a model using a simple "spin" gesture with the non-dominant hand.

Effectively, OrthoTumble allows for the user to quickly inspect the model in a 3D perspective view and then automatically return to an appropriate orthographic view to continue drawing curves. All this is done smoothly without any discontinuous view switches, thus helping the user to maintain and understand how the data corresponds between multiple views. This draw-inspect-draw-inspect workflow is one which we feel critically contributes to an overall feel of fluid and seamless interaction in the system. Comments by users during our study (to be described later in the chapter) further reinforces this belief. Other research [32] has explored similar transient view change approaches and found similar effectiveness.

4.3.3 Culling planes

As the number of curves making up the 3D model increases, it becomes difficult to discriminate between them when in the orthographic views. In order to effectively manage which curves get displayed, we have introduced the notion of culling planes to our system. Essentially these are two planes that are dragged out from the active drawing plane in the perspective view. Only those curves between the two culling planes will be subsequently visible when in the relevant orthographic view. If a planar or non-planar curve intersects a culling plane then the section outside of the planes will be clipped.

4.3.4 Tape Drawing Extensions

The present system extends the digital tape drawing technique of the previous work [2] in several ways. A description of the original technique is given in section 2.6. The reader is referred to [2] for a more detailed discussion. From an interaction standpoint this two-handed digital tape drawing technique is interesting in that both straight lines and curves can be drawn without an explicit mode switch. Implicitly, the dominant hand determines whether curves or straight lines are drawn by either moving or not moving respectively.

4.3.5 One-handed Tape Drawing

There are situations, however, where the two handed technique falls short. For example, it is extremely difficult to draw circles or curves that loop around (which are needed in car design) using the technique because of both the biomechanical limitations on the movements of the two hands, and the fact that the two hands will begin to collide in physical space. In order to address this limitation we developed an extension to the tape drawing technique that uses only one hand.

In our special one-handed tape drawing mode, illustrated in Figure 4.5f-j, only the dominant hand cursor is used which by default represents the roll of tape. Moving this cursor with the tracker button pressed allows the unfastened tape segment to be lengthened or shortened (Figure 4.5f-g). Moving without the button pressed simply drags the unfastened tape segment around. As the unfastened tape segment is moved, however, it leaves behind a trail of fastened tape (Figure 4.5h). Terminating the tape involves pressing the button (Figure 4.5i) and reducing the unfastened tape segment length to zero (Figure 4.5j). This returns the user again to the state of adjusting the unfastened tape segment (Figure 4.5f).

Essentially, this one-handed version behaves like the two-handed technique would if the non-dominant hand button was always pressed, and the distance between the two cursors was fixed once taping began. Given the fixed distance between the end of the tape and the roll of tape during dragging, perfectly straight lines are difficult to achieve in this

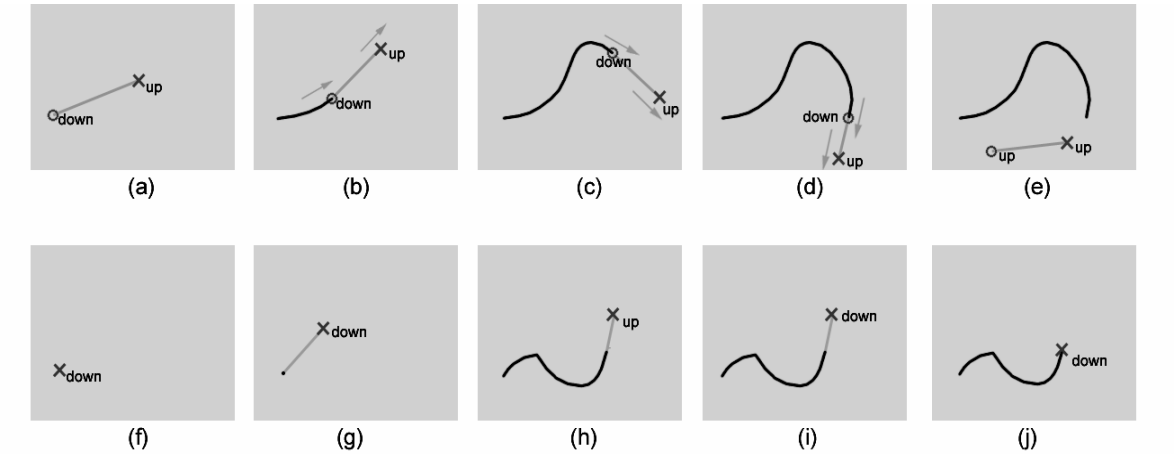

Figure 4.5: (top row) Two-handed tape drawing technique. (bottom row) One-handed tape drawing.

one-handed technique. The benefit, however, is that looped curves can now be created easily.

Given these tradeoffs, and in the interest of maintaining compatibility with the traditional technique, our system continues to use the two-handed technique as the default line and curve creation tool with the one-handed technique available as an alternate when the need to create looped curves and circles arises.

4.3.6 Curve Guides

When designing 3D models consisting of many curves it is often necessary for one curve to intersect other curves in order to form appropriate skeletons for surfaces. While the tape drawing technique allows for high quality curves to be generated, it is not easy to plan ahead and be able to draw the curve such that it will definitely intersect some predetermined points. To assist the tape artist in this regard, we developed a mechanism for guiding the curve towards these intersection points.

As Figure 4.6 illustrates, we first select the intersection points of interest, and then specify the desired tangents at these points (Figure 4.6a). Then, using the tape drawing technique we can begin to draw a new curve that is intended to intersect these points. When the new curve is at a certain distance from the first intersection point, a guide curve begins to fade in (Figure 4.6b-c). Using Bezier interpolation with two slopes (the current

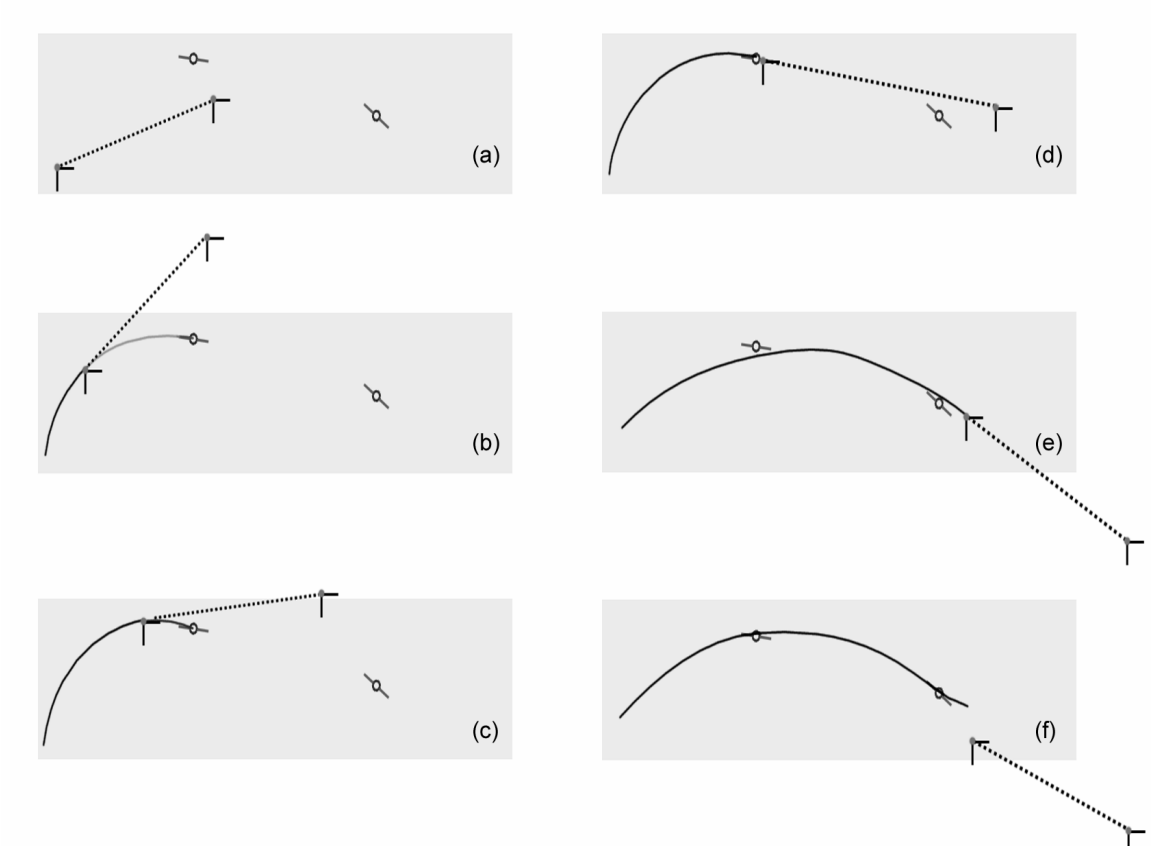

Figure 4.6: Curve Guides

slope of the new tape curve, and the slope of the tangent that was specified at the intersection point) and two points (the tape fastening point, and the intersection point) as inputs, the guide curve gives the user a best guess preview of what the new tape curve should look like in order to smoothly pass through that intersection point. The user can choose to continue taping along this guide curve, or simply choose to accept the guide curve as the desired curve. There are of course situations where the user may choose to deviate significantly from the guide curve. In this case, the system attempts to dynamically keep updating the guide curve with successive best guesses, adjusting the tangent at the intersection point in the process. Once the new curve passes the intersection point, the guide curve and tangent indicator for that point disappears (Figure 4.6d). In addition, if a new curve is within 20 pixels from an intersection point (Figure 4.6e), the system assumes that the curve should intersect that point and makes the required small interpolation adjustments to the curve such that it passes smoothly through that point (Figure 4.6f).

4.3.7 Tangent, Perpendicular Snapping, and Rail Guides

In order to make our system capable of performing our goal task of designing the principal curves of a real car, we included several functions which are not new to 3D modeling but have never been used in conjunction with digital tape drawing. To facilitate accurate joining of curves, we implemented a set of curve snapping techniques. These techniques are similar to the snapping approaches originally proposed by Bier et. al. [6] and found in many CAD packages. We have the ability to snap the start of tape curves to other curves to then begin drawing tangentially or perpendicular to the existing curve. Additionally, the snapped-to curve can act as a rail guide (i.e., before tape is fastened for a new curve, the snapped starting point can be moved along the snapped-to curve).

4.3.8 Editing

We also allow the cutting out segments of curves by specifying two points and applying a simple up-down gesture over the segment to be removed.

4.3.9 Loading Engineering Criteria

In auto design studios, tape drawings are commonly created on top of underlying engineering criteria which specify the car's underlying mechanical elements such as engine block position, transmission, and wheel wells, etc. The stylistic body designs have to be built around this engineering criteria. As such, our system provides for engineering criteria to be loaded as the background in our drawing planes. These engineering criteria are only visible when in orthographic view, since that is where the tape drawing is done. Another alternative to engineering criteria is to use a reference image to guide the creation of design curves. For example, tape drawing could be done against images of a previous year's model loaded into the background plane (see Figure 4.4).

4.3.10 Two-handed Pan and Zoom

As in a variety of previous systems [23, 31], we support camera control interactions of pan, tumble and zoom using a bimanual interaction technique. However, in the system described in Chapter 3, these operations were only available while in perspective viewing mode. In our current system, we have made the techniques available in both perspective and orthographic views. We found this to strongly reflect users' expectations. For example, just as one expects a tumble operation in an orthographic view to automatically and smoothly transition into a perspective view (and therefore the emergence of our OrthoTumble feature), users also expected zooming and panning within an orthographic view so they could place their work zone in a more comfortable position.

To support this, we adapted and refined the basic bimanual technique proposed in [31]. With this previous technique, when both tracker buttons are pressed, the cursors become attached to the drawing surface and this acts like manipulating the surface like a rubber sheet. Moving both hands together or apart zooms in or out. Keeping both hands at a constant distance apart and moving them in the same direction pans the surface.

One problem with the previous technique is that panning and zooming cannot be easily performed independently. To allow a user to only pan or only zoom, our technique allows a pan or zoom operation to only occur after a certain threshold of movement is met. To accomplish this we track the vectors of motion for each cursor. The threshold for panning is an angle between the two cursor movement vectors of below 45 degrees (indicating they are moving in the same direction), and a vector length for both cursors of 5 pixels (to filter tracker noise). For zooming, the cursor movement vector angle must be more than 135 degrees (indicating they are moving in opposite directions), with the same length requirement. Once the pan threshold is met, the cube will be panned in the vector of the average of the two cursor movement vectors. When the zoom threshold is met, the zoom is increased proportionally to the increase in distance between the two cursors.

4.4 User Study

User testing has been performed on previous tape drawing systems. In [2], the reactions of automotive designers using their tape drawing systems for short periods of time were reported. With our current system we have similar short-term user testing experiences. Specifically, we demonstrated our system at the SIGGRAPH 2001 tradeshow floor and allowed individuals to try our system hands-on. Users only spent enough time with the system to get a little experience in creating 2D curves and navigating around the model. We observed that users had little difficulty learning the basic user model (e.g., tape-style drawing on the sides of a 3D cube). In addition, our new navigation techniques appeared easy to understand and operate.

This type of user testing was meaningful to gauge users' initial reaction to the basic design approach and functions. However, for our current system, the goal was to provide enough functionality that a user could reasonably attempt to specify the primary curves of real car. Since this is a time consuming task we decided to study an artist attempting to use our system for an extended period of time (several hours) as opposed to testing many users over shorter periods of time.

As a test task we chose the creation of the primary curves of a real car, specifically, the Dodge Viper model. The Viper's body shape is typical of automotive body design which has sophisticated curved surfaces resulting from long flowing primary curves. We also had a scale metal die-cast model of the Viper at 1/16 scale, which could serve as a reference when trying to create the primary curves. To assist in the test task, we provided reference images of the Dodge Viper against which the user could tape draw (Figure 4.4). As a test subject we employed a professional 3D modeler and artist for approximately 3 one hour sessions to initially familiarize himself with the system, then attempt to construct the primary curves of the Viper. Our system is by no means "walk and use" and therefore we spent time training the user on the basic capabilities of the system, explaining minor interface bugs, and helping them out of error states. Our testing was not particularly concerned with the small details and pitfalls of interaction but more concerned with identifying successes and failures in the fundamentals of the user model and system capabilities.

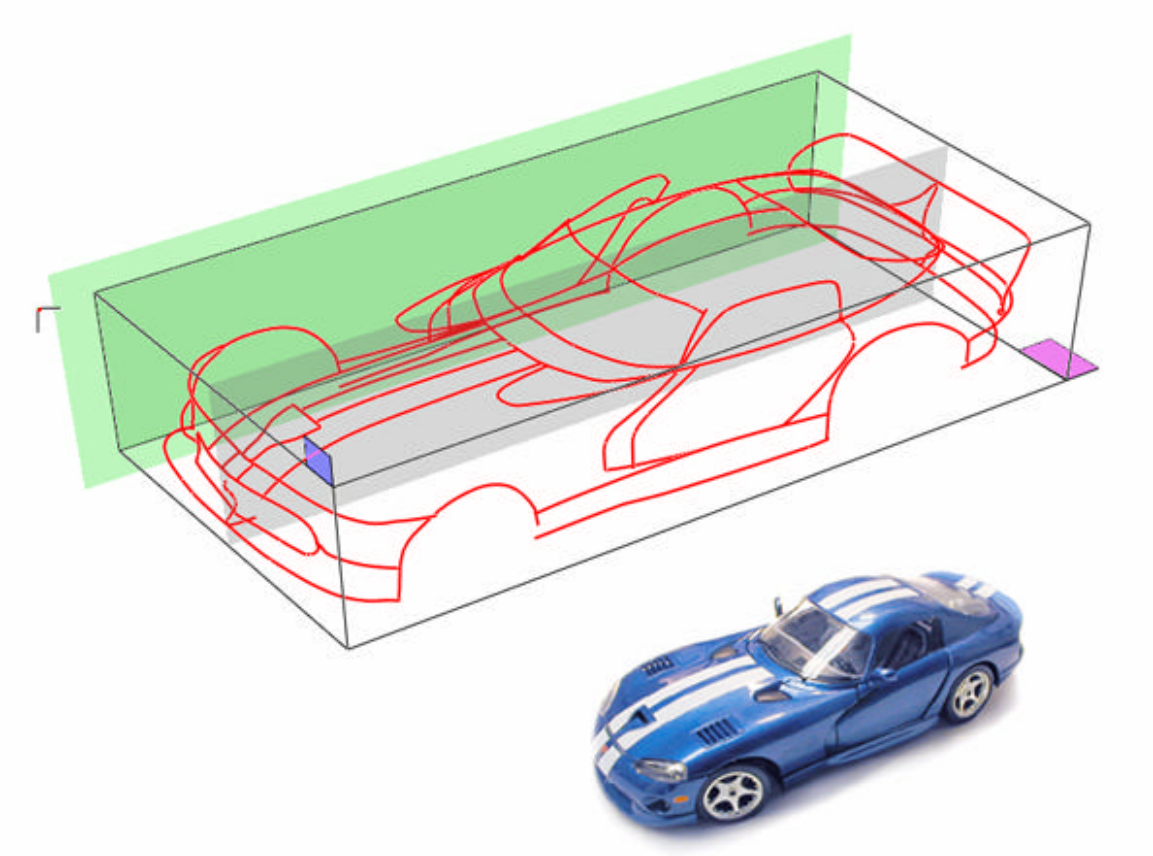

Figure 4.7: Principal curves of test task car model.

Despite efforts to assist the user in quickly learning how to use the system, the user could only construct a very crude model of the Viper after three hours of experience. We feel this is due to a series of issues which never allowed the user to graduate from the learning phase and be proficient with the system. While we originally thought that the user would refine a model over the three hours of work, we found the user simply discarded the model at the end of each session and restarted from scratch each time. Thus the user rather than trying to reach the goal of creating the model was actually creating initial crude models to explore and learn the system.

Ultimately, we still believed that our system was capable of creating reasonable primary curve model of the Viper. Thus we challenged the implementer of the system, who was the most experienced with the system and had become quite skilled at the tape drawing technique, to attempt to model the Viper. Figure 4.7 shows the results of 2 hours of work by the system implementer and shows a fairly sophisticated set of 3D curves that are representative of the Viper Model.

From all of these experiences of our user testing we have the following observations:

- Accurate, stable tracking is critical. Although our system used 3D input sensors with a larger range, the accuracy and stability of these trackers was still a major stumbling block in learning and operating the system. While our implementer had learned to understand and deal with these problems, our artist-user experienced numerous problems. The source of these problems, however, is not a fundamental limitation but a technological one posed by the tracking hardware we used. This is easily solved by simply purchasing better tracking technology. The artist-user also reported that the tracker cables somewhat restricted their movement. Not only did this hinder working directly on the display but it also restricted his freedom in stepping back from the screen to examine the model.
- Despite the problems with input, our artist-user had some very positive reactions with regard to navigation. Both the OrthoTumble and two-handed zooming and panning features were learned quickly then used frequently and effectively.
- In terms of 3D curve creation we observed that planning which curves need to be created to specify the primary characteristics of a shape is a difficult problem, independent of the tool. In general, it requires skill to decompose a shape into its 3D principal curves. For example, our implementer-user claims that majority of time spent in producing the Viper model was not spent actually drawing the curves but spent considering what sort of curve or curves should be drawn to capture the primary characteristics of the model.
- The most encouraging reports from our artist-user concerned the basic user model. Our artist-user reported that he found the metaphor of tape drawing and the notion of building up a 3D model using tape very easy to understand, specifically the concept of creating complicated curve sets through the combination of simpler 2D curves. Finally, our artist-user reported that while tape drawing is used in automotive design, it is also a general technique used in many other types of art work and therefore has utility to a broader audience than automotive designers.

4.5 Discussion

The system discussed in this chapter allows artists to create the primary curves of a model from various perspectives, and at the same visualize how these curves integrate into the final model. However an important issue with 3D line drawing is the limitation of using lines to visualize what will eventually be surfaces. For example, one can draw a circular profile of a sphere from the side, front and top and project each of these curves into a 3D volume by giving them depth. From a perspective view though, the combination of these orthographic circles does not produce the perfect circular profile of the sphere because there is no surface running between the orthographic circles. In general, this lack of surface information "between the curves" can be combated somewhat by adding in more curves that will run along the surface to help a viewer perceive a surface from a series of curves. Future research could be done on how to devise a way to integrate profile curves drawn from a perspective view into the 3D model or alternatively allow a user to create and manipulate surfaces between the profile curves.

The present work gives rise to the major design issue of how far to continue to develop the system. Traditionally, the automotive design workflow involves the production of concept sketches that are turned into 2D tape drawings. These drawings are subsequently digitized and turned into 3D surface models. Based on feedback from automotive designers, we have built a system that allows the 2D tape drawing phase to replace some of the early phases of the construction of the 3D surface modeling. The question is where to stop. Future work would be to study how far 2D tape drawing can be extended into 3D surface modeling. There are several issues to consider. On the one hand, our system could benefit from adding functionality to help edit and refine existing curves similar to those found in 3D modeling programs. On the other hand, there is a question of how much functionality can we introduce into our digital tape drawing system before tape artists reject it because it is perceived to be as complicated as typical 3D CAD surface modeling applications. Another issue is the possible importance of having the tape drawing phase clearly separate from the surface design phase. For example, it may be considered “too much design detail” to mix principal curve creation with surface specification.

4.6 Summary

In this chapter we have identified and described the implementation of the majority of functionality needed to create the principal curves of an automotive design using tape drawing as the basic curve creation technique on a large scale display surface. We extended the interactions described in Chapter 3, providing an interface for integrating these principles curves into a 3D environment to build up a single wireframe model. New interaction techniques, such as curve guides, one-handed tape drawing, and 3D curve creation, were developed to allow users to create these principle curves.

Many of the ideas we have explored have general applications in both CAD and large display systems. In terms of CAD systems, digital tape could be used as an additional curve generation technique for traditional desktop-based CAD packages especially if two-handed input is available. If not, our one-handed tape drawing technique could be substituted. Additionally, we have already seen our techniques for animating between orthographic and perspective views be adopted in several commercial software packages. In terms of large display systems, various techniques employed in our tape drawing system could be used in any application where one is working at arm's length on large display systems, as is done for example by Guimbretiere et. al. [20]. Examples of these techniques include pop-up marking menus, one-button per input tracker, and other techniques for avoiding the traditional widgets around the periphery of the display such as menu bars and tool pallets. The extension of our two-handed panning and zooming navigation techniques to be operable from both perspective and orthographic views is particularly useful. We believe that, collectively, these techniques could serve as an effective basic interaction model for any application that involves working at arm's length on large displays.

Because the tape drawing tool was used for curve creation, users had more direct control over curve parameters than provided in the point based status-quo interfaces. However, as with any sketching interface, users are still not directly manipulating entire curves, as their manipulations are limited to the current point of interaction. In the following chapter we will decrease this level of indirection by exploring the use of a high degree-of-freedom input device to create and manipulate curves.

Chapter 5

High Degree-of-Freedom Curve Creation and Manipulation [1]

5.1 Introduction

The previous two chapters described a prototype system for designing curves on large displays. Such a system would be beneficial to the artists who are skilled in creating large scale physical sketches. Thus, the work presented provides a significant contribution, as limited research has previously been done on how these interactions could be integrated into virtual systems.

Another technique common to artists which has limited virtual representation is the use physical tools and objects. Many artists are trained in sculpting and have specialized skills in manipulating physical forms with their hands or hand held tools. However the previous research which attempts to improve on the status-quo virtual systems for working with curves is generally focused on sketching interfaces [2, 5, 10, 16, 17, 26, 33, 35, 37]. While these interfaces do increase the degree of direct interaction with curve parameters, there still exists a level of indirection. By using physical tools as input devices, it may be possible for a virtual system to further decrease the level of indirection. It is therefore desirable to investigate virtual systems for curve design which provide a more hands-on experience.

[1] The material presented in this chapter is based on the following publication: Tovi Grossman, Ravin Balakrishnan, and Karan Singh. *An Interface for Creating and Manipulating Curves using a High Degree-of-Freedom Curve Input Device*. in *ACM CHI 2002 Conference on Human Factors in Computing Systems*. 2003 p. 185-192.

In this chapter we will explore such a system, in which a high degree-of-freedom curve input device is used to directly manipulate virtual curves. The motivation comes from the design industry, where traditional high degree-of-freedom physical techniques for manipulating curves in clay modeling and paper drawings are very popular. Here, curves are created directly by copying segments from physical templates (e.g., French curve templates) or using physical tools, which flex to produce curves (e.g., spring steels). Many of these physical techniques allow for very fast and accurate specification of curves, while current virtual techniques are typically more cumbersome. Given the success of these techniques in the real world, it is reasonable to expect that virtual interaction techniques could benefit from the use of physical artifacts more closely matched to the task [15, 27, 35]. In [3], based on physical techniques used in the design industry, Balakrishnan et. al. explored using a high degree-of-freedom curve input device to directly create curves and surfaces (see section 2.6 for a review). While theirs was the first such system that exploited the affordances of physical tools for manipulating virtual curves, their interaction techniques were limited by simple absolute mappings between physical tool and virtual curves/surfaces. Their system also provided little precision control over virtual curve parameters.

This chapter presents a system that significantly extends this previous research, demonstrating the use of a high degree-of-freedom curve input device for quick but precise curve creation and manipulation in both 2D and 3D space. The system achieves this via a suite of new interaction techniques for relative mapping of the parameters of the physical device to the virtual world.

5.2 System Hardware

The primary input device that forms the core of our interface is the ShapeTape (www.measurand.com): a 96 x 1 x 0.1 cm rubber tape with a flexible spring steel core that has 32 fiber optic sensors distributed in pairs uniformly, 6 cm apart, along its length. Each sensor pair provides bend and twist information at its location, and by summing the bends and twists of the sensors along the tape, the shape of the tape can be reconstructed

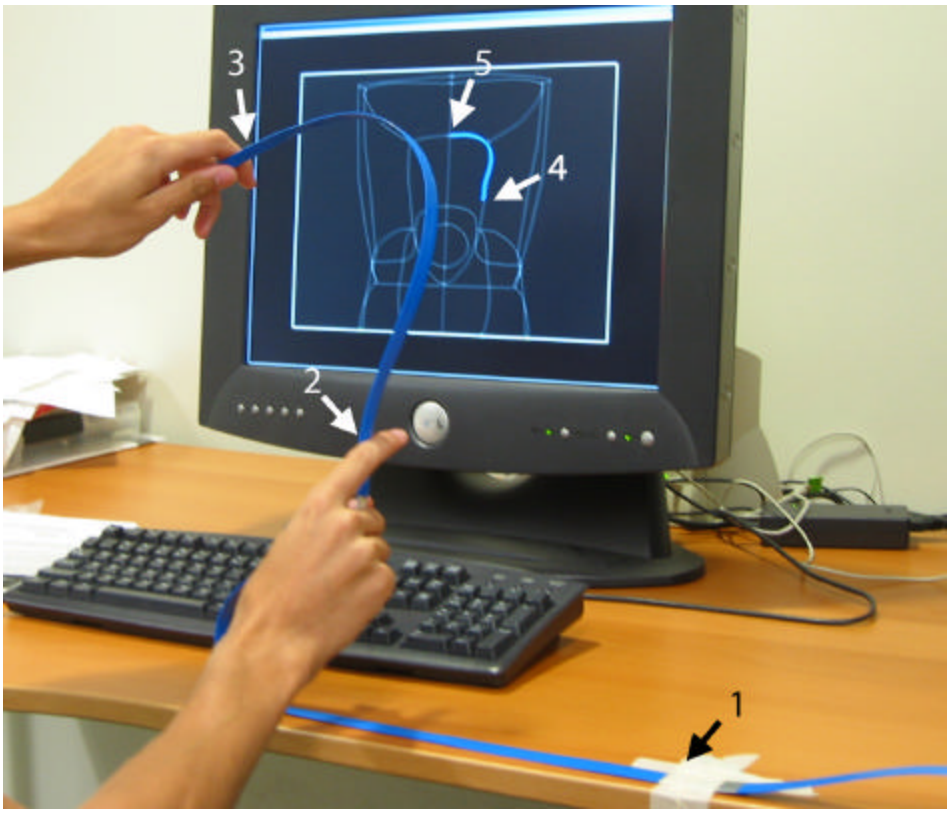

Figure 5.1: System setup. The tape is secured at point #1. The first half of the tape (segment 1-2) is used to position and orient the starting point (#2) of the second half of the tape (segment 2-3). The position and shape of the second half of the tape (segment 2-3) is mapped to the virtual TapeWidget (segment 4-5).

in three dimensions. This shape reconstruction is relative to the location of the first sensor pair. The tape was secured to a desk at the location of this first sensor pair. In [3], a separate position and orientation tracker was used to determine the location of the starting point of the tape in 3D space. Rather than adding another piece of hardware to our system, we instead used the first half of the tape to position and orient in physical 3D space the starting point of the second half of the tape. The second half of the tape was then used to input shape information. Figure 5.1 illustrates this hardware setup.

The only other input device used was a footpedal hinged in the middle, with two momentary buttons: one at the front of the pedal (henceforth referred to as the FrontButton) and the other at the back (BackButton).

It is important to note that we have deliberately chosen this minimal hardware setup, in order see how far we could go with using only a curve input device and two buttons. As will be evident as we progress through this chapter, we were able to develop a

significant repertoire of gestural interaction techniques using only this minimalist configuration. We readily admit, however, that while this was an excellent setup for pushing the boundaries of our research, any commercially viable system for curve manipulation using curve input devices would likely require additional input modalities.

5.3 Gestures

In building a usable system for curve manipulation using such a minimal hardware configuration, we are faced with the challenge of providing a mechanism for command input. We use the footpedal's two buttons for the most frequently used commands, and to kinesthetically maintain a few modes. Additional commands are specified via a set of gestures performed using the physical tape (henceforth referred to simply as tape). (Figure 5.2) illustrates this gesture set. Six of the gestures (Figure 5.2a-f) are recognized by tracking the velocity vectors of the centre and two endpoint sensors of the tape. The last two gestures (Figure 5.2g, h) are recognized by measuring the amount, direction, and timing of twisting of the tape. These eight gestures are used throughout our system. The commands associated with these gestures will be described as we progress through the paper explaining the various interaction techniques.

5.4 Physical To Virtual Interface

In the standard mouse/keyboard GUI interface, the system cursor serves as an abstract representation of mouse movements. This cursor is then used to manipulate various parts of the interface. In a sense, the cursor serves as an intermediary between the mouse and the rest of the interface. Unlike the standard point cursor that has only two changeable parameters (X-Y position in 2D space), analogous intermediaries for high degree-of-freedom devices would likely have more parameters that can be manipulated by the device. Based on previous experience [3], and building on foundational work on three dimensional widgets by Conner et. al. [11], we have designed such an intermediary for our interface, called the TapeWidget. Manipulations on the tape are used to control

parameters of the TapeWidget, including position in space, size, and shape. The TapeWidget is then used to create, edit, and manipulate other virtual curves in the graphical scene. This TapeWidget is one major difference between this present system and the exploratory work done in [3].

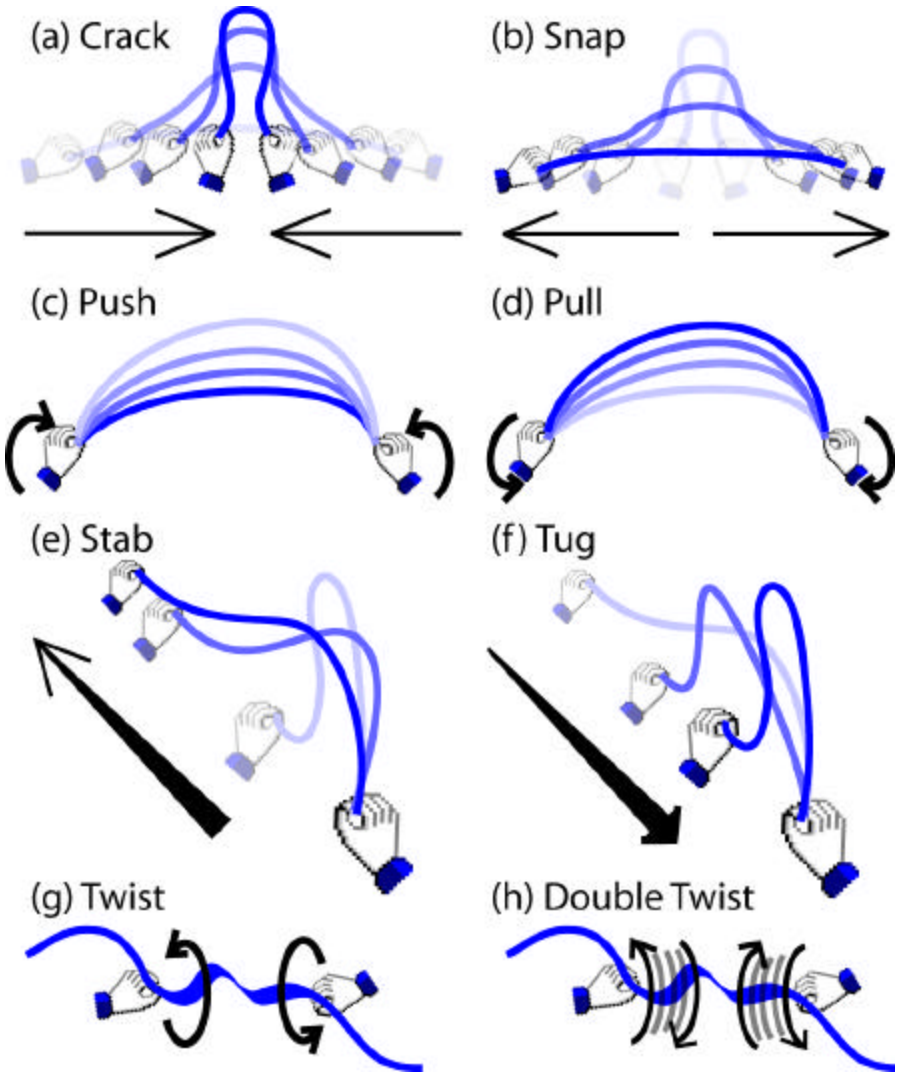

Figure 5.2: Gesture set. (a) Crack – quickly move the tape endpoints close together. (b) Snap – quickly move the tape endpoints apart. (c) Push – quickly move the centre of the tape towards the screen by "flicking" the wrists forward. (d) Pull – quickly move the centre of the tape away form the screen by "flicking" the wrists backward. (e) Stab – quickly move the free endpoint of the tape towards the screen. (f) Tug – quickly pull the free endpoint of the tape away from the screen. (g) Twist – twist the tape with each hand moving in opposite directions. (h) Double Twist – twist the tape in one direction and then in the opposite direction in quick succession.

The following subsections describe the various techniques we have developed that use manipulations of the tape to change the TapeWidget's parameters.

5.4.1 Tape to TapeWidget Mapping

Two points on the physical tape are mapped to the TapeWidget's endpoints (Figure 5.1). Thus, the TapeWidget takes on the shape of this section of the physical tape.

5.4.2 TapeWidget Positioning

The first half of the tape, which is used to track the location in space of the second half of the tape, does not allow for enough freedom to move the TapeWidget around the entire screen. As such, we need an interaction technique for gross scale position of the TapeWidget in space. This is accomplished as follows using a "flying" metaphor: when the FrontButton is pushed and held, moving the tape moves the TapeWidget in the same direction with a velocity relative to the tape's distance from its starting point.

5.4.3 TapeWidget Scaling

Scaling the TapeWidget is accomplished by using a twist gesture on the tape. When the tape is twisted in one direction, the size of the TapeWidget is increased. A twist in the opposite direction scales down the TapeWidget. The twist gesture can be made anywhere along the tape.

5.4.4 Endpoint Mapping

The two endpoints that are set by default to map a section of the tape to the TapeWidget can be changed at any time using a double twist gesture. Double twisting at any point of the tape will make that the endpoint (Figure 5.3) – analogous to the real-world action of twisting a piece of wire back and forth to break it. Together with scaling, this enables

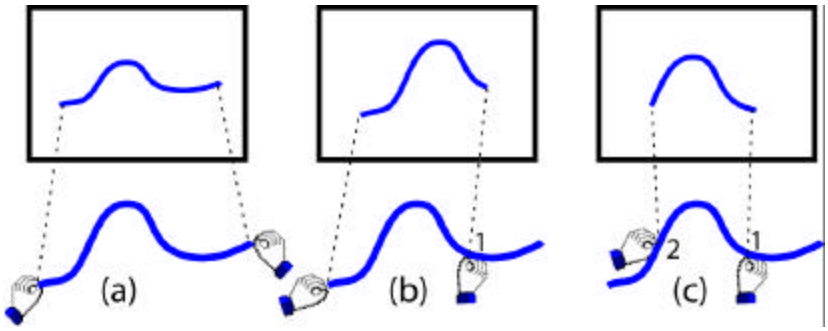

Figure 5.3: Endpoint mapping. (a) shows the default mapping where the whole tape is mapped to the whole TapeWidget. (b, c) a double twist gesture at points 1 and 2 respectively sets those points as the new endpoints.

subsections of the tape to be mapped to the TapeWidget, resulting in changing the gain between the tape and the TapeWidget. If a small section of the tape is mapped to the entire TapeWidget, the resulting high gain mapping is good for changing the shape of the entire TapeWidget with just a small change in the shape of the tape. Conversely, mapping a large section of the tape to the TapeWidget results in a low gain mapping that is better for precise tweaking of portions of the TapeWidget. A snap gesture restores the default endpoint mapping.

5.4.5 Sharp Corners

Since the tape cannot be physically bent into sharp corners (the fiber sensors would crack if bent too sharply), the TapeWidget's shape by default also cannot have sharp corners. However, in many curve editing tasks, it is desirable to be able to create sharp changes in a curve's shape. To support this, we use a crack gesture to "crack" the continuity of the TapeWidget's shape, resulting in a "corner" that consists of two straight lines joined at a vertex (Figure 5.4). The location of the endpoints of this corner TapeWidget are controlled by the endpoints of the tape, and the angle between the lines is relative to the distance between the tape's endpoints. A snap gesture restores the regular curved TapeWidget.

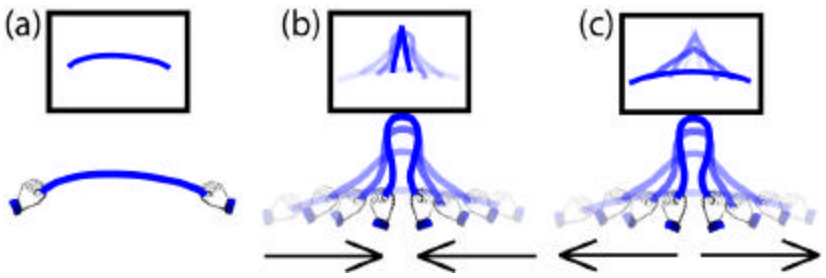

Figure 5.4: Sharp corners. (a) default mapping of tape to TapeWidget. (b) a crack gesture changes the TapeWidget into a sharp corner. (c) snap gesture restores TapeWidget to default mapping in (a).

5.4.6 TapeWidget Locking

We have found many situations (to be described shortly) where it is desirable to temporarily lock the parameters of the TapeWidget. We use the BackButton as a toggle to lock and unlock the TapeWidget's position, shape, and size.

5.4.7 Relative Tape to TapeWidget Orientation Mapping

So far, apart from the technique for creating sharp corners, all the manipulations we have described result in the TapeWidget taking on the exact orientation of the tape. However, it is sometimes desirable to have a more relative mapping between the orientation of the tape and TapeWidget, particularly in situations where the desired orientation of the TapeWidget would otherwise necessitate holding the tape in an awkward position. To support relative orientation mapping, we first click the BackButton to lock the TapeWidget. The tape can now be reoriented in a comfortable pose by the user, without affecting the TapeWidget. When the BackButton is clicked again the TapeWidget is unlocked and its shape, position, and size responds to new manipulations of the tape, but with a transformed orientation (Figure 5.5). Again, a snap gesture restores the default absolute mapping.

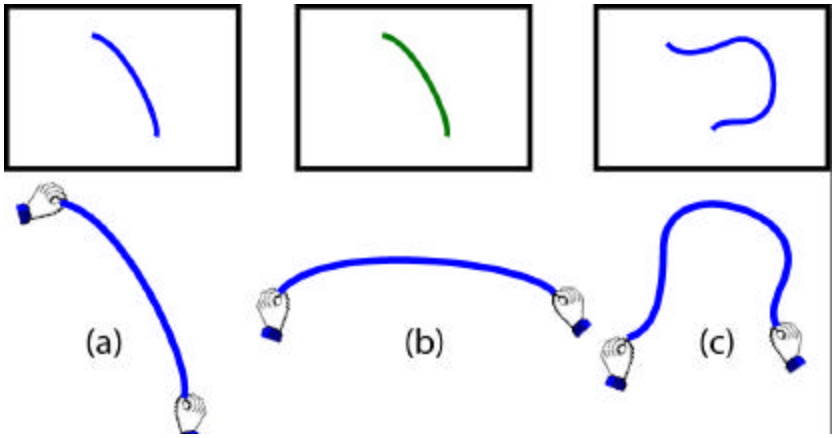

Figure 5.5: Relative orientation mapping. (a) normal mapping. The tape is held in a fairly awkward position. (b). BackButton is clicked to lock the TapeWidget. The tape now be repositioned without affecting the TapeWidget. (c) BackButton is clicked again to unlock the TapeWidget. Tape manipulations are now mapped with a relative orientation to the TapeWidget.

These techniques result in a sophisticated interface between the physical tape and its virtual instantiation – the TapeWidget. The interface supports both simple and complex (but precise) control of the TapeWidget's parameters. Using this highly maneuverable TapeWidget, coupled with a few more gestures, we have developed a set of interaction techniques for creating and manipulating other virtual curves in a graphical scene. We describe these interaction techniques in the following sections, beginning with 2D techniques, and then moving to 3D.

5.5 2D Curve Creation

Creating a new curve in the scene is accomplished by using a push gesture (Figure 5.6). The metaphor here is that of "pushing forward to drop a curve onto the scene", echoing the paradigm shift from control point based creation to a faster, more direct approach. Invoking this gesture creates a new curve at the position of the TapeWidget that replicates its current shape and size. We lock the TapeWidget (using the BackButton) before invoking the push gesture. This prevents any movements in the tape caused by the gesture itself from accidentally changing TapeWidget parameters.

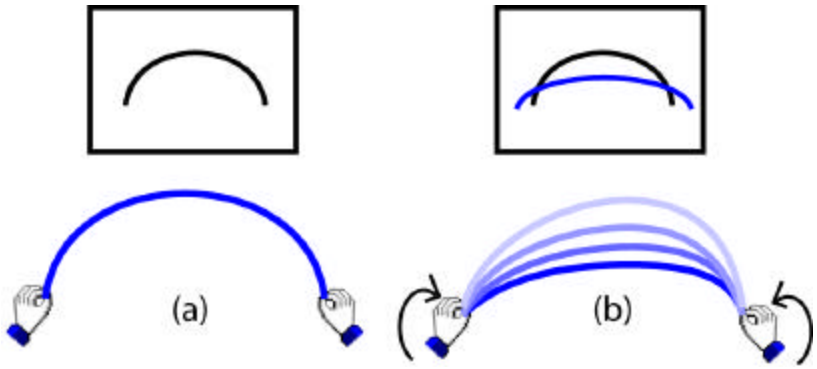

Figure 5.6: Curve creation. (a) tape shapes and positions the TapeWidget. BackButton click locks the TapeWidget. (b) push gesture drops a new curve with TapeWidget's shape into scene.

5.6 2D Curve Editing

Our curve editing design philosophy is based on a sculpting metaphor with the TapeWidget as the analogue of a sculpting tool. Just as a sculpting tool's behaviour changes depending on how it is used, we implicitly use the TapeWidget's proximity to, and intersection with, curves in the scene to determine the type of editing to be performed. We have developed four example editing techniques that as a whole can be viewed as a single editing mechanism that changes its behaviour depending on the proximity and intersection of the TapeWidget to the curve. The next few subsections describe these techniques.

5.6.1 Curve Selection

To edit a curve, the user must first select it. Our system considers the curve closest to the endpoint of the TapeWidget to be the "current curve". To distinguish the current curve from others in the scene, we render it as a thicker curve. Clicking the FrontButton toggles selection and deselection of the current curve. We can select multiple curves by moving the TapeWidget around the scene.

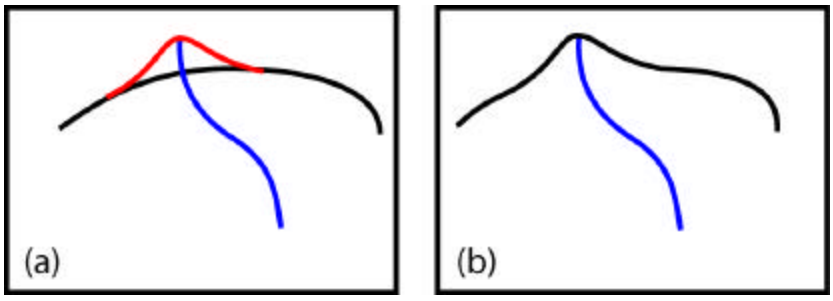

Figure 5.7: Reshaping with a single intersection point. (a) the TapeWidget intersects a curve in the scene at one point, and the interpolated red preview curve is shown. (b) FrontButton is clicked and the curve takes on the shape of the preview curve.

5.6.2 Reshaping with a Single Intersection Point

The simplest method for reshaping a curve is by intersecting one endpoint of the TapeWidget with a selected curve. Two interpolation endpoints are placed on either side of the point of intersection, at a default distance. A preview curve that is a Bezier interpolation joining these interpolation points and the endpoint of the TapeWidget is displayed in red, indicating to the user what the resulting change would look like (Figure 5.7). Increasing or decreasing the interpolation interval is achieved by twisting the tape. Clicking the FrontButton results in the curve being reshaped as indicated by the red preview curve.

5.6.3 Reshaping with Two Intersection Points

If the TapeWidget intersects a selected curve at two points, the red preview curves appears, taking on the shape of the TapeWidget between these two intersection points. The red preview curve is finely interpolated at the two points of intersection to maintain smoothness (Figure 5.8). As with the previous technique, clicking the FrontButton results in the curve taking on the shape of the preview curve.

Both these methods of editing can be used while there is a relative orientation mapping between the tape and TapeWidget, and/or when the TapeWidget is shaped as a sharp corner as previously described.

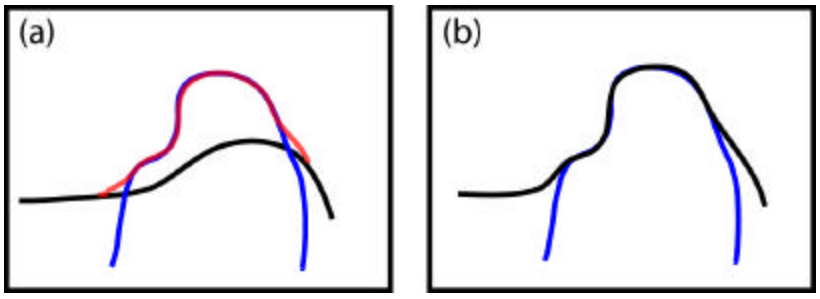

Figure 5.8: Reshaping with two intersection points. (a) TapeWidget (blue) intersects a curve (black) at two points, and the resulting interpolated red preview curve is shown. (b) FrontButton is clicked and the curve takes on the shape of the preview curve.

5.6.4 Compound Reshaping

A third method, useful for precise but compound relative reshaping of curves, is now described with Figure 5.9 illustrating. First, the TapeWidget glues to an existing curve when its shape closely matches the shape of a curve near to it (Figure 5.9a). This is the analogue of point snapping to parts of a scene when using a standard point cursor for editing. The endpoint extents are placed on the existing curve closest to where the TapeWidget endpoints were before the TapeWidget was glued (Figure 5.9b). Once glued, the TapeWidget is controlled by both the endpoints and subsequent manipulation of the physical tape. When the TapeWidget's shape is changed, we keep the end points constrained to their original glued position by displacing every point on the TapeWidget by an offset vector as follows: The difference vectors between the original end points e1, e2, and the end points after the TapeWidget's shape is changed, are shown in (Figure 5.9c) as d1,d2. The offset vector at any point along the TapeWidget is an interpolation of the vectors d1 and d2, varying from d1 at point e1 to d2 at point e2. These offset vectors are added to the TapeWidget, shown in (Figure 5.9d). Clicking the FrontButton reglues the TapeWidget in its current state, while preserving the same end points.

The other set of extents seen in (Figure 5.9b), called interpolation extents, are used to blend the results of the curve generated using the algorithm described above with the unedited segments of the curves. A Bezier curve smoothly joins the unedited curve segments in the regions between the endpoints and the interpolation extents (see Figure 5.9e). The end points and interpolation extents can also be edited from their default

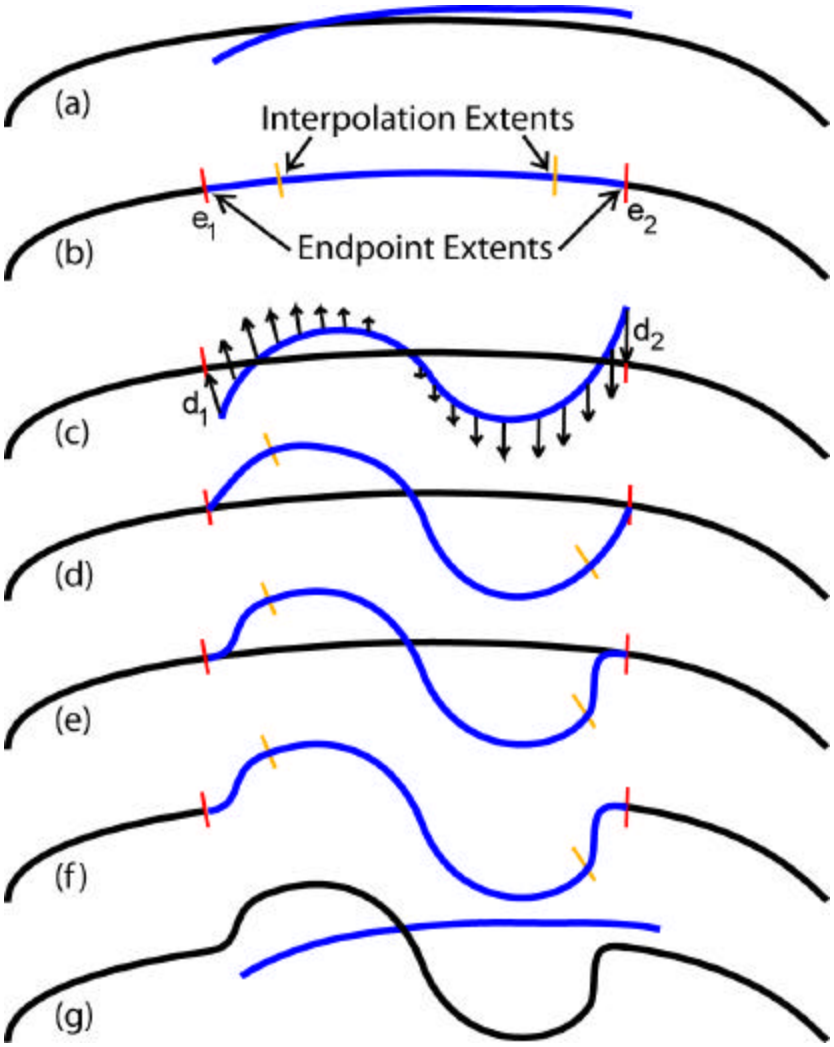

Figure 5.9: Compound reshaping. (a) the TapeWidget (blue) moves close to an existing curve (black), causing it to "glue" onto the curve. (b) endpoint and interpolation extents are displayed. (c) the computed vector offsets (the arrows are for illustration only and do not appear when the system is in use). (d) shows result of adding the vector offsets. (e) curve is interpolated between the endpoint and interpolation extents. (f) a push gesture reshapes the curve. (g) a tug gesture unglues the TapeWidget from the curve.

locations. Either set of extents are active at any given time. When the tape is twisted close to the center, the active set of extents move closer together or further apart, depending on the twist direction. If either endpoint of the tape is twisted, then only its corresponding extent will move. A double twist toggles the active set of extents.

A push gesture makes the existing curve permanently take on the shape of the preview curve in between the endpoint extents (Figure 5.9f). The TapeWidget remains

glued, so the process can be repeated. As always, clicking the BackButton locks/unlocks the TapeWidget, allowing for the shape of the curve to be "cranked" in a relative manner (Figure 5.10). This technique allows for precise, compound, relative reshaping of curves to be performed, which would be quite difficult to achieve using existing curve editing techniques. The TapeWidget unglues with a tug gesture.

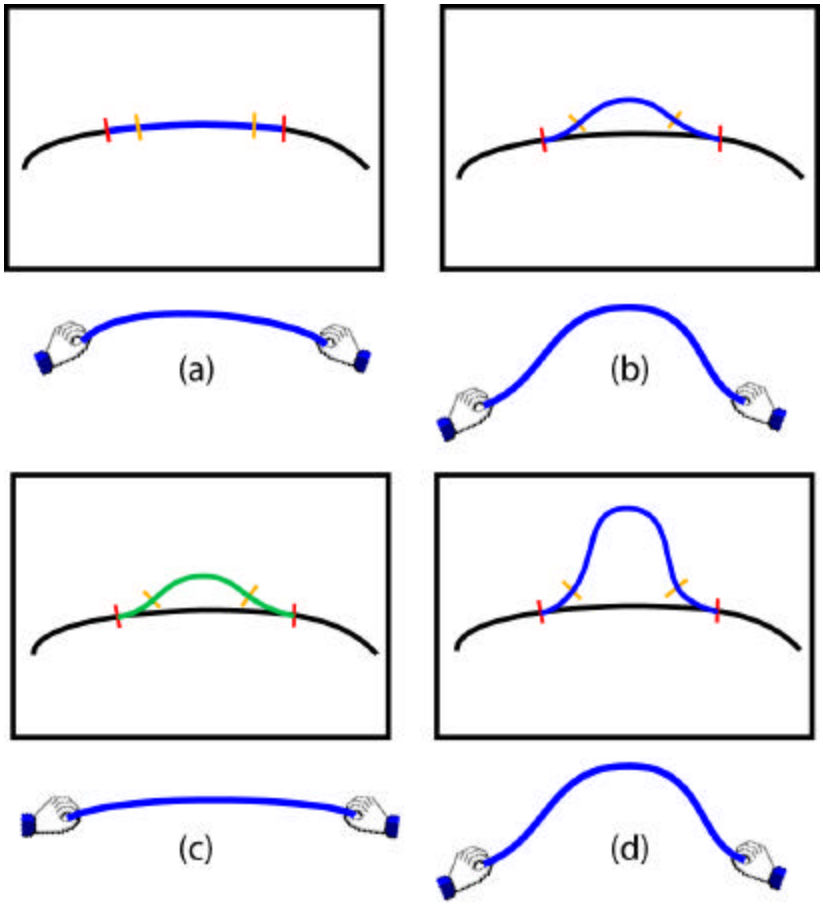

Figure 5.10: Cranking during compound reshaping. (a) TapeWidget is glued onto the curve. (b) manipulating the TapeWidget shapes the curve. (c) BackButton is clicked to lock the TapeWidget, allowing the tape to be repositioned without affecting the TapeWidget. (d) BackButton is clicked again to unlock the TapeWidget. Manipulating the TapeWidget again results in the curve being further reshaped in a relative manner.

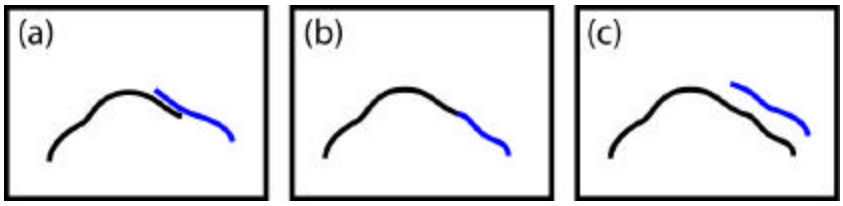

Figure 5.11: Extending curves. (a) TapeWidget moves close to the endpoint of an existing curve. (b) TapeWidget glues to the endpoint of the curve. (c) after BackButton is clicked to lock the TapeWidget, a push gesture extends the curve.

5.6.5 Extending Curves

If the TapeWidget is close to an endpoint of a curve, it glues to that endpoint. As usual, the TapeWidget can be locked and unlocked by clicking the BackButton. Twisting the tape determines how much the curve's endpoint will be extended along the curved path provided by the TapeWidget's shape (Figure 5.11). A push gesture makes the curve extension permanent.

5.7 Tools

When editing curves, it is often desirable to be able to reuse a previously defined TapeWidget shape. We support this by creating, saving, and recalling a set of user defined tools. To create a tool a crack gesture is made while the TapeWidget is locked. This closes the shape of the TapeWidget and locks it. The endpoint of the tape can then be used to control the position and rotation angle of the tool. The tool can be scaled and moved around the screen just like the regular TapeWidget. Similar to the TapeWidget, a tool can be used to drop new curves of the same shape as the tool into the scene, or to reshape existing curves. We also implemented a menuing system to provide the user with access to tools that have been previously created. A stab gesture pops up the menu directly above the position of the TapeWidget. The menu contains iconic representations of the tools arranged in an arc (Figure 5.12). A maximum of six tools are displayed at a time. If the menu has more than six tools, the edges of the menu fade out to indicate that the menu can be scrolled. Twisting the tape rotates the menu left or right, scrolling

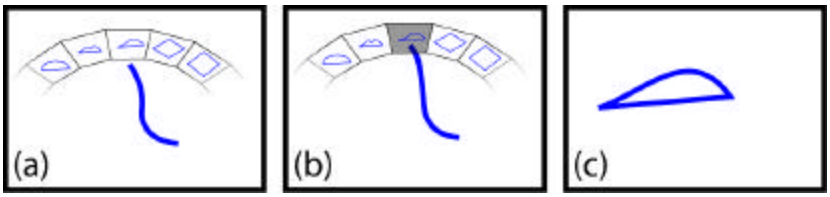

Figure 5.12: Tool menu. (a) stab gesture pops up the menu. (b) moving TapeWidget highlights desired menu item. (c) clicking FrontButton selects highlighted tool.

through all available tools. By moving the TapeWidget over a tool and clicking the FrontButton, the TapeWidget will take on the shape of that tool. If a tool is locked and a snap gesture is made, then the tool can be edited as if it were a curve in the scene. After modifications to the tool have been made, a tug gesture allows the user to return to using that tool. This new tool shape is also added to the tool menu.

5.8 3D Operations

We now extend our interactions into 3D space. The previous work [16, 17] described in Chapters 3 and 4 on the design of 3D curve editing tools indicated that to ensure accuracy it is preferable for 3D curves to be created and manipulated in 2D orthographic views of the 3D scene. Given that all the 2D techniques described in the previous sections will seamlessly work on 2D orthographic views of 3D space, we already have a suite of tools for 3D curve manipulation. What remains to be developed are techniques for use in the 3D perspective view, including: camera controls for maneuvering around the 3D view, selecting curves and construction planes, creating new 3D construction planes, and transitioning between 3D and 2D views.

5.8.1 2D to 3D Transitions

A tug gesture is used to seamlessly transition between 2D orthographic and 3D perspective views. Similar to the techniques described in Chapters 3 and 4 [16, 17], this transition is smoothly animated to allow the user to understand the correspondence between the 2D and 3D visuals.

5.8.2 Camera Controls

As in previous systems [16, 17, 37], we support standard camera controls of tumble, pan, and zoom. However, we have adapted these techniques to work with the tape. If the user points the tape towards the screen while pressing and holding down the FrontButton, movement of the tape's endpoint rotates the camera around the 3D scene. If the tape is parallel to the screen while the FrontButton is pressed and held, a pan-zoom mode is entered. The two endpoints of the tape control the panning and zooming. Based on the technique used in [31], moving the two endpoints closer together or further apart zooms in or out respectively. Keeping the tape's endpoints at a constant distance apart and moving them together in the same direction pans the camera.

5.8.3 Construction Planes

In order to create curves in 3D space, we project 2D curves onto 3D construction planes. By default, three base construction planes (x-y, x-z, y-z), are drawn in the 3D scene. These not only provide a base for drawing curves onto, but also serve as a basis for specifying other construction planes.

Selecting Construction Planes

When in the 3D perspective view, a cursor is drawn and is controlled by the endpoint of the tape. Using this cursor to point at any of the construction planes and clicking with the BackButton selects that plane. Only one plane can be selected at a time.

Creating Curves on Construction Planes

Once a construction plane has been selected, a tug gesture smoothly transitions from the 3D perspective view to a 2D orthographic view perpendicular to that construction plane. In this 2D view, curves can be created and edited using all the techniques previously

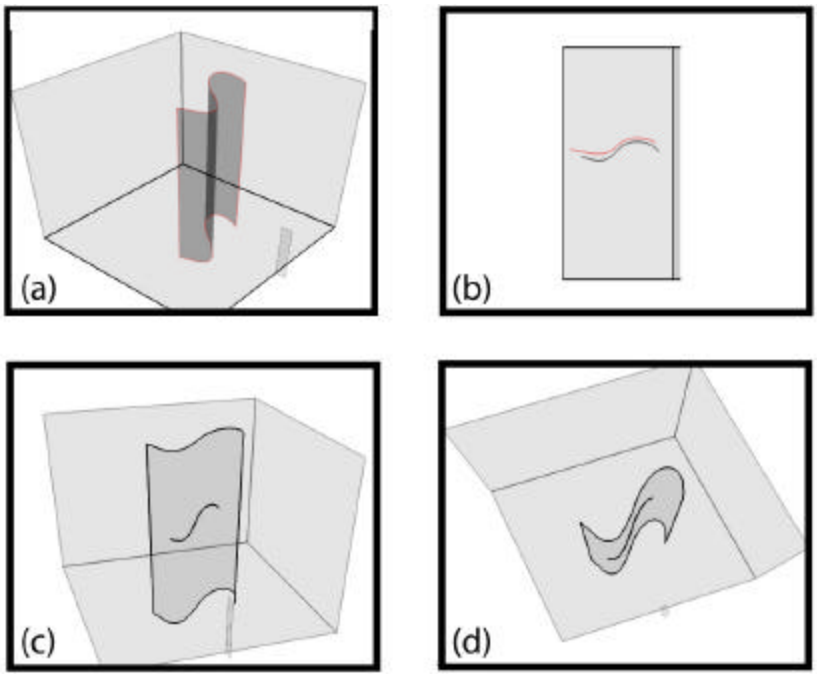

Figure 5.13 Drawing on construction planes. (a) construction plane is selected. (b) tug gesture transitions to 2D orthographic view of that construction plane, and a new curve is created on it. (c, d) another tug gesture transitions back to the 3D perspective view, where the curve can be inspected in 3D space.

described. Any new curves created are projected onto the surface of the construction plane. Figure 5.13 illustrates. Another tug gesture returns to the 3D view.

Intersection Points

In both 2D and 3D views, points where curves intersect planes are marked by a large green dot. This serves as a useful aid for the user to align curves.

Creating New Construction Planes

When an existing plane has been selected in the perspective view, a push gesture creates a new plane that's positioned perpendicular to this selected plane. The new plane is either flat or curved, depending on the type of plane last created. A snap gesture changes the plane from flat to curved, and a crack gesture does the opposite.

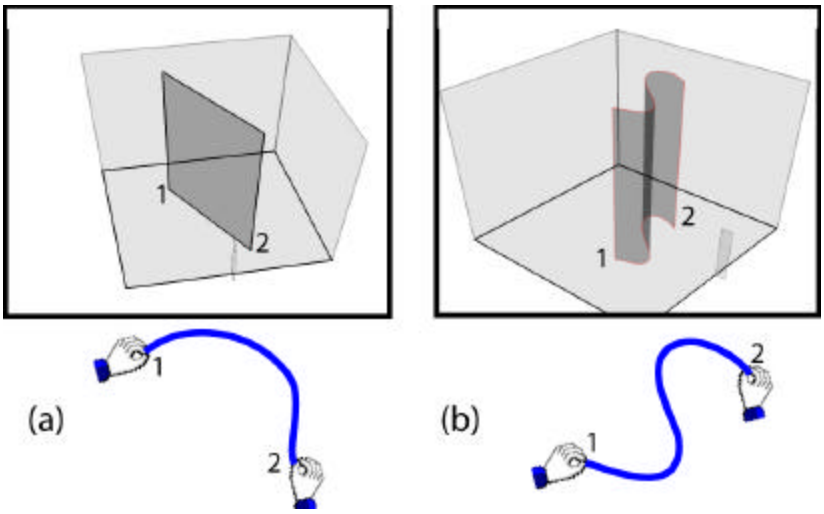

Figure 5.14: Creating new construction planes. (a) in 3D view, a push gesture creates a flat plane. The endpoints 1 and 2 of the plane correspond to the endpoints of the tape that are similarly labeled. (b) a snap gesture converts to a curved plane whose shape is controlled by the tape.

As Figure 5.14a shows, the location of a flat plane is controlled by the two endpoints of the tape. When the new flat plane's location is within a small delta of either major axis of the base selected plane, the flat plane will snap to that axis. Figure 5.14b illustrates how the shape of a curved plane is determined by the shape of the tape.

A new plane can be moved around the 3D scene by moving the tape, using the same technique for moving the TapeWidget around when in the 2D orthographic view. The shape, position, and orientation of the plane can also be locked and unlocked by clicking the BackButton (i.e., using the same method for locking/unlocking the TapeWidget). When the new plane is locked, another push gesture confirms the addition of this plane, with its locked shape, into the scene at its current position and orientation.

The technique for plane creation described above works fine for planes of approximate shape. If more precise shape and location is required of a plane, its curvature can be defined in the orthographic view. First, a curve is drawn using the previously described techniques. With a pull gesture this curve will be used to define the curvature of a new surface, extruded along the normal of the currently selected plane.

5.9 User Feedback

While the developed system is only a research prototype, we thought that it was important to get some early feedback from potential users. We felt that it would be more valuable to obtain feedback from expert users of other curve manipulation systems, rather than rely on novices who would be unlikely to understand all the subtleties involved in complex curve manipulation tasks. We asked two subjects very experienced in using various commercial 2D and 3D graphics software – one an academy-award nominated 3D modeler, and the other an industrial designer – to try out the system for a two hour session each. The first hour was used by the subject to learn the various gestures and interaction techniques. During the second hour, the subject was asked to freely use the system to create and manipulate curves of their own choosing.

Neither user chose to build complex 3D models with the system. One user did, however, build up a relatively simple 3D model of a table consisting of a circular body and four legs. Both users did become familiar enough with the system to get the overall feel of the various techniques, and were able to give us valuable feedback, leading to the following observations:

- Both users liked using the tape to directly manipulate curves without the abstractions found in current interfaces. However, they both felt that the tape would be more useful if it were complementing other tools and input devices, rather than being the only tool available. One user said he would like to be able to put the tape down and sketch part of a curve with a pen – in other words, using the best tool for the job as needed. Both users liked the fact that the tape could be manipulated using both hands simultaneously, and even suggested potentially new ways in which two-handed manipulation of the tape could be used.
- We found that the users were able to learn and perform our gesture set, and easily understood the underlying metaphors. In particular, the push gesture resonated with both users. This is likely because it only required a simple flick of the wrists and also because the metaphor was very obvious - that of pushing a curve or plane into the scene.
- One of the users commented on the amount of physical work that could be required to manipulate the tape. He said he much prefers a small tablet that can be used without

lifting his wrist. We note that while there will always be some physical effort required when using tangible devices, more extensive use of the widget flying technique and relative mappings we provide could have significantly reduced the effort required. Almost all tangible user interfaces [15, 27] face this challenge of providing simple, easily understood physical artifacts to control virtual elements without increasing the work required of the user. Indeed, one of the reasons why the mouse is such a popular device is that it can be operated in a "lazy" fashion [1].

- One user mentioned that the paradigm which we used would be very useful for organic modeling, where curves are drawn and then tweaked to the designer's preferences. However, he said he sometimes felt uncomfortable creating and editing curves without direct control over the underlying mathematics of the curve. This complaint could be due to the fact that this user is very highly skilled in the use of current interfaces that demand that the user understand the underlying math. Indeed, one of the goals of our system was to insulate the user from the math! So, in a sense this user's comment could be viewed as a measure of how well we had managed to achieve this separation between the underlying math and the interaction techniques. On the other hand, this could be an argument for providing other tools to complement the tape so that the user would have the choice of either very direct manipulation with the tape or more abstract manipulations of the foundational curve parameters.

5.10 Discussion

One of the first challenges we faced was finding a way to use the tape not only for the curve manipulation tasks that it was very well suited for, but also for command input. Our solution was to use the tape to capture user gestures, and we defined an initial set of eight distinct gestures. Note that each of these gestures was assigned a consistent meaning in our system. For example, the push gesture was consistently used to add elements to the scene: curves when in a 2D view, planes when in a 3D view. Similarly, the twist gesture was used to increase or decrease a particular variable: interpolation interval, widget size, and menu item position. Overall, we found that this small but well

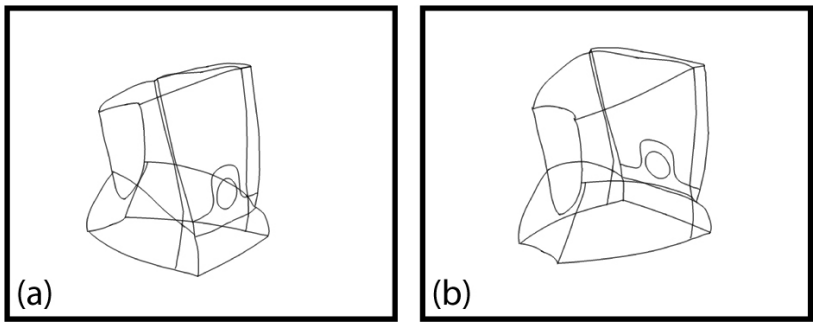

Figure 5.15: Two views of an example wireframe model created entirely with our system.

defined gesture set enabled us to support the fairly large set of interaction techniques used in our system.

While our minimalist hardware configuration was useful in forcing us to push the envelope on developing novel interaction techniques, including gestural command input, user feedback indicated that our system would benefit from integrating other techniques, such as sketching [5, 10, 25, 26, 37]. We also believe that it would be beneficial to have a suite of different physical tapes, each with unique physical characteristics such as bending tension, thickness, length, and precision. As discussed in [3], it could also be useful to have tapes that can maintain their physical shape over time.

5.11 Summary

In Chapters 3 and 4 [16, 17], the digital tape drawing tool, a sketching like interface, was used to achieve the goal of giving artists more direct control over the shape of virtual curves. In this chapter we diverged from sketching interfaces, and presented a tangible user interface to further increase the level of direct control. The other main goal of the systems described in this thesis, integrating multiple virtual curves into a 3D scene, was accomplished through the use of the TapeWidget. By controlling the parameters of this widget, the user can specify the shape, size, orientation, and location of virtual curves. By adding construction planes and camera controls, a useful set of tools for integrating these curves into a 3D scene is provided (Figure 5.15). Overall, our system has demonstrated how a high degree-of-freedom curve input device can be used for complex curve

manipulations. It is also an example of a graspable [15] or tangible [27] interface that goes beyond the simple one-to-one mappings between physical and virtual artifacts that have typically been demonstrated by previous research.

Chapter 6

Conclusions and Future Directions

6.1 Conclusion

In this thesis we have described the design and implementation of prototype systems for the creation and manipulation of curves in 3D virtual environments. The tools provided by the systems give artists direct control over the shape of curves, without requiring them to work with or understand the underlying mathematics. This allows artists to rely on their existing skills which they utilize in the physical world. The systems also allow users to integrate the curves which they create into a single 3D model, allowing the artists to survey the progress of the final model as it is built up from different profiles. By using smooth animated transitions, and the OrthoTumble technique [17], users can quickly switch between the 3D perspective and 2D orthographic views, while they maintain and understand how the data corresponds between the different views.

6.2 Contributions

In the implementation of the systems a number of different elements of human computer interaction were utilized: two handed interactions, large display interfaces, and tangible user interfaces. In some cases, interactions had been previously developed and the contribution in this thesis comes from the manner in which they were integrated into a

single seamless interface. In other cases however, existing techniques have been enhanced, and will now be summarized.

6.2.1 Two Handed Interactions

In Chapter 4 [17] we describe extensions to the bimanual digital tape drawing tool, which was first presented in [2]. Curves guides were used to provide visual cues, while rail guides and tangent and perpendicular snapping were implemented to provide physical constraints. Another two-handed technique that has been enhanced is the two-handed pan-zoom tool, previously described in [23, 31]. In our implementation in Chapter 4 [17], we provide threshold values of movements before panning or zooming occurs, so that users can easily pan and zoom independently, or pan and zoom in parallel. In Chapter 5 [18], we describe a suite of novel two-handed gestures used with a high degree-of-freedom curve input device.

6.2.2 Large Display Interfaces

We described interaction techniques for large scale displays in Chapters 3 and 4 [16, 17]. Two design principles were followed when working with such displays. Firstly, multiple windows should be avoided, as the user can only easily focus on the information directly in front of them. Secondly, standard widgets which appear along the border of the display, such as menus and scroll bars, should be avoided, as users may not be able to reach them. Many of the new interaction techniques which we describe are tailored specifically to these design principles. Viewpoint markers, for example, allow users to quickly change between preselected viewpoints, without the need of extra windows or bordering widgets. Similarly, the animated transitions allow users to seamlessly switch between views, eliminating the need for multiple windows.

6.2.3 Tangible User Interfaces

High Degree-of-Freedom Interactions

The system described in Chapter 5 [18] implements a number of interaction techniques for a high degree-of-freedom tangible user interface. Such a device allowed us to explore novel interactions techniques, which utilize the many degrees-of freedom. For example, the multiple parameters of the TapeWidget could all be controlled by the single input device, by positioning, orienting, shaping, and twisting the ShapeTape.

Relative Mappings Between Physical Devices and Virtual Objects

The main extension over previous tangible user interfaces which we provide is that non-direct relative mappings between the physical object, in our case the ShapeTape, and its virtual representation are used. In some cases this provides an easier mode of interaction. For example a relative mapping can be used so the user does not need to hold the physical tool in an awkward position. In other cases however, it allows users to perform operations that would be extremely difficult or even impossible to accomplish with the tool in the physical realm. For example, compound editing lets users repeatedly "crank up" the shape of a curve, into shapes that could be difficult to create with the ShapeTape. The corner tool is another example, as it allows users to create and edit with sharp corners, even though the physical tool will always be a smooth curve.

6.3 Integration of Interaction Techniques

The contributions outlined in section 6.2 were presented in separate chapters of this thesis, as the corresponding interaction techniques were implemented in separate prototype systems. However, there is no reason why these interactions techniques could not be utilized in a single fluid interface. It was clear from our user feedback that the ShapeTape interface would work best if complemented by other techniques for sketching curves.

One option would be to scale down the tape drawing interaction techniques for desktop use, using a tablet and two pucks. This would allow artists to quickly sketch accurate curves using the tape drawing tool, and also to create and edit curves by directly manipulating them with the curve input device. Similarly, an artist could use a large curve input device while creating tape drawings on a wall sized display, using techniques which were described for high degree-of-freedom input devices in Chapter 5. The device could be used to shape curves, specify the depth of non-planar curves, or even sweep out surfaces.

6.4 Future Directions

At the conclusion of Chapters 3-5, we discuss possible lines of future work specific to the interaction techniques which the chapter described. Here we will discuss more general lines of future work which can be applied throughout.

The prototypes which were designed and implemented allow for the construction and integration of planar and non-planar curves into 3D Wireframe models. The techniques which have been developed could be extended to allow for the creation and manipulation of surfaces, which overlay the profile curves of the model. While this process could be automated, as the system could interpolate surfaces over the principle curves, it would be interesting to explore tools which would enable the user to have an interactive role in the surfacing process. Tools that allow for sculpting and painting directly on the surface, such as those found in 3D software packages, would then need to be explored for use in this context.

Another aspect of the systems which have been developed is that they provide effective tools for organic or conceptual drawings, providing a freedom of artistic expression. Fewer tools are provided for the creation of curves and models which follow specific engineering criteria. A possible line of future work would be to investigate ways of incorporating extra functionality to create models with more precision, without the interface becoming as complicated as a typical 3D CAD application. The one-handed tape drawing technique, for example, could be extended to allow the use of the free non-

dominant hand to specify control points and centers of curvature while the dominant hand draws the curve. A good example of an interface which supports the creation of both conceptual and precision curves is seen in [35], where users could both sketch curves and manipulate control points.

Finally, an interesting design element of our system was the focus on a thorough set of 2D curve editing techniques, and then using these 2D techniques to create and manipulate 3D curves. As a result, the curves which can be drawn using these systems all exist on a flat plane, or a plane which is curved along one axis. While such curves are adequate for creating the principle curves of most 3D models, it would be interesting to explore interaction techniques for the creation of free form 3D curves, such as a knot or a spring, which do no exist on surfaces.

Bibliography

1. Ravin Balakrishnan, Thomas Baudel, Gordon Kurtenbach, and George Fitzmaurice. *The Rockin'Mouse: Integral 3D manipulation on a plane*. in *ACM CHI 1997 Conference on Human Factors in Computing Systems.* 1997. New York, NY p. 311-318.
2. Ravin Balakrishnan, George Fitzmaurice, Gordon Kurtenbach, and William Buxton. *Digital tape drawing*. in *ACM UIST 1999 Symposium on User Interface Software and Technology*. 1999. New York, NY p. 161-169.
3. Ravin Balakrishnan, George Fitzmaurice, Gordon Kurtenbach, and Karan Singh. *Exploring interactive curve and surface manipulation using a bend and twist sensitive input strip*. in *ACM I3DG 1999 Symposium on Interactive 3D Graphics*. 1999. New York, NY p. 111-118.
4. Ravin Balakrishnan and Gordon Kurtenbach. *Exploring bimanual camera control and object manipulation in 3D graphics interfaces*. in *ACM CHI 1999 Conference on Human Factors in Computing Systems.* 1999. New York, NY p. 56-63.
5. Thomas Baudel. *A mark-based interaction paradigm for free-hand drawing.* in *ACM UIST 1994 Symposium on User Interface Software and Technology*. 1994 p. 185-192.
6. Eric Bier and Maureen Stone. *Snap dragging*. in *ACM SIGGRAPH 1986 Conference on Computer Graphics and Interactive Techniques*. 1986. New York, NY p. 233-240.
7. William Buxton, *Living in augmented reality: Ubiquitous Media and Reactive Environments,* in *Video Mediated Communication*, K. Finn, Sellen, A., Wilber, S., Editor. 1997, Erlbaum: Hillsdale, NJ. p. 363-384.
8. William Buxton, George Fitzmaurice, Ravin Balakrishnan, and Gordon Kurtenbach, *Large displays in automotive design.* IEEE Computer Graphics and Applications, 2000. July/Aug 2000: p. 68-75.

9. William Buxton and Brad Myers. *A study in two-handed input*. in *ACM CHI 1986 Conference on Human Factors in Computing Systems.* 1986. New York, NY p. 321-326.
10. Jonathan Cohen, Lee Markosian, Robert Zeleznik, John Hughes, and Ronen Barzel. *An interface for sketching 3D curves*. in *ACM I3DG 1999 Symposium on Interactive 3D Graphics*. 1999. New York, NY p. 17-21.
11. Brookshire Conner, S. S. Snibbe, K. P. Herndon, Daniel Robbins, Robert Zeleznik, and Andy van Dam, *Three dimensional widgets.* Computer Graphics, 1992. 22(4): p. 121-129.
12. C. Cruz-Neira, D.J. Sandin, T.A. DeFanti, R.V. Kenyon, and J.C. Hart, *The CAVE: Audio visual experience automatic virtual environment.* Communications of the ACM, 1992. 35(6): p. 65-72.
13. M. Czernuszenko, D. Pape, D. Sandin, T. DeFanti, G.L. Dawe, and M.D. Brown, *The ImmersaDesk and infinity wall projection based virtual reality displays.* Computer Graphics, 1997. 31(2): p. 46-49.
14. S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Jannsen, D. Lee, K. McCall, E. Pedersen, K. Pier, J. Tang, and B. Welch. *Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration*. in *ACM CHI 1992 Conference on Human Factors in Computing Systems.* 1991. New York, NY p. 599-607.
15. George W. Fitzmaurice, Hiroshi Ishii, and William Buxton. *Bricks: Laying the foundations for graspable user interfaces*. in *ACM CHI 1995 Conference on Human Factors in Computing Systems.* 1995. New York, NY p. 442-449.
16. Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton. *Interaction techniques for 3D modeling on large displays*. in *ACM I3DG 1999 Symposium on Interactive 3D Graphics*. 2001. New York, NY p. 17-23.
17. Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton. *Creating principal 3D curves with digital tape drawing*. in *ACM CHI 2002 Conference on Human Factors in Computing Systems.* 2002 p. 121-128.

18. Tovi Grossman, Ravin Balakrishnan, and Karan Singh. *An Interface for Creating and Manipulating Curves using a High Degree-of-Freedom Curve Input Device*. in *ACM CHI 2002 Conference on Human Factors in Computing Systems.* 2003 p. 185-192.
19. Yves Guiard, *Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model.* Journal of Motor Behavior, 1987. 19(4): p. 486-517.
20. Francois Guimbretière, Maureen Stone, and Terry Winograd. *Fluid interaction with high-resolution wall-size displays.* in *ACM UIST Symposium on User Interface Software and Technology*. 2001. New York p. 21-30.
21. François Guimbretière and Terry Winograd. *FlowMenus: combining command, text, and data entry*. in *ACM UIST Symposium on User Interface Software and Technology*. 2000 p. 213-216.
22. Kenneth Herndon, Robert Zeleznik, Daniel Robbins, Brookshire Conner, S. Snibbe, and Andy van Dam. *Interactive shadows.* in *ACM UIST 1992 Symposium on User Interface Software and Technology*. 1992. New York, NY p. 1-6.
23. Ken Hinckley, Mary Czerwinski, and Michael Sinclair. *Interaction and modeling techniques for desktop two-handed input*. in *ACM UIST 1998 Symposium on User Interface Software and Technology*. 1998. New York, NY p. 49-58.
24. Ken Hinckley, Randy Pausch, Dennis Proffitt, and Neal Kassell, *Two-handed virtual manipulation.* ACM Transactions on Computer-Human Interaction, 1998. 5(3): p. 260-302.
25. Takeo Igarashi and John Hughes. *A suggestive interface for 3D drawing*. in *ACM UIST Symposium on User Interface Software and Technology*. 2001 p. 173-181.
26. Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. *Teddy: a sketching interface for 3D freeform design*. in *ACM SIGGRAPH.* 1999 p. 409-416.
27. H. Ishii and B. Ullmer. *Tangible bits: towards seamless interfaces between people, bits and atoms.* in *ACM CHI 1997 Conference on Human Factors in Computing Systems.* 1997. New York, NY p. 234-241.
28. Paul Kabbash, William Buxton, and Abigail Sellen. *Two-handed input in a compound task*. in *ACM CHI 1994 Conference on Human Factors in Computing Systems.* 1994. New York, NY p. 417-423.

29. Myron Krueger, *VIDEOPLACE and the interface of the future*, in *The art of human computer interface design*, B. Laurel, Editor. 1991, Addison Wesley: Menlo Park, CA. p. 417-422.
30. Gordon Kurtenbach and William Buxton. *The limits of expert performance using hierarchical marking menus*. in *ACM CHI Conference on Human Factors in Computing Systems.* 1993. New York, NY p. 35-42.
31. Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and William Buxton. *The design of a GUI paradigm based on tablets, two-hands, and transparency*. in *ACM CHI 1997 Conference on Human Factors in Computing Systems.* 1997. New York, NY p. 35-42.
32. Jeffrey Pierce, Matthew Conway, Maarten van Dantzich, and George Robertson. *Toolspaces and glances: storing, accessing, and retrieving objects in 3D desktop applications*. in *ACM CHI 1999 Conference on Human Factors in Computer Systems.* 1999. New York, NY. p. 163-168.
33. Emanual Sachs, A. Roberts, and D. Stoops, *3-draw: A tool for designing 3D shapes.* IEEE Computer Graphics and Applications, 1991. 11(6): p. 18-26.
34. S. Schkolne, M. Pruett, and P. Schroeder. *Surface drawing: Creating organic 3D shapes with the hand and tangible tools*. in *ACM CHI 2001 Conference on Human Factors in Computing Systems.* 2001. New York, NY p. 261-268.
35. Karan Singh. *Interactive curve design using french curves*. in *ACM I3DG'99 Symposium on Interactive 3D Graphics*. 1999. New York p. 23-30.
36. Karan Singh and Eugene Fiume. *Wires: A geometric deformation technique*. in *ACM SIGGRAPH 1997 Conference on Computer Graphics and Interactive Techniques*. 1998. New York, NY p. 405-414.
37. R. C. Zeleznik, K. Herndon, and J. Hughes. *SKETCH: An interface for sketching 3D scenes*. in *ACM SIGGRAPH 1996 Conference on Computer Graphics and Interactive Techniques*. 1996. New York, NY p. 163-170.
38. Robert Zeleznik, A. Forsberg, and P. Strauss. *Two pointer input for 3D interaction*. in *ACM I3D Symposium on Interactive 3D Graphics*. 1997. New York, NY p. 115-120.

39. Robert Zeleznik and Andrew Forsberg. *UniCam - 2D Gestural Camera Controls for 3D Environments.* in *ACM I3D Symposium on Interactive 3D Graphics*. 1999. New York, NY p. 169-173.