Sketch Based Design
of
2D and 3D Freeform Geometry

Masha Belchich
SKETCH BASED DESIGN
OF
2D AND 3D FREEFORM GEOMETRY

RESEARCH THESIS

Submitted in partial fulfillment of the requirements
for the degree of
Master of Science in Computer Sciences

Masha Belchich

Submitted to the Senate of
the Technion—Israel Institute of Technology

Shvat, 5770 Haifa
January 2010
Acknowledgement

The Research Thesis was done under the Supervision of Prof. Gershon Elber in the Faculty of Computer Science.

I would like to thank Prof. Gershon Elber for his help and guidance in pursuing this research.

I would also like to thank my mother, Evgenia Nikolsky, and my husband, Vadim Belchich, for their infinite support and patience.

THE GENEROUS FINANCIAL HELP OF THE TECHNION IS GRATEFULLY ACKNOWLEDGED.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgement</td>
<td>iii</td>
</tr>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>Notations</td>
<td>3</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>5</td>
</tr>
<tr>
<td>2 Related Work</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Creation of Polygonal Geometry</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Creation of Freeform Geometry</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Commercial Products</td>
<td>9</td>
</tr>
<tr>
<td>2.3.1 SketchUp (<a href="http://sketchup.google.com/">http://sketchup.google.com/</a>)</td>
<td>9</td>
</tr>
<tr>
<td>2.3.2 SolidWorks (<a href="http://www.solidworks.com/">http://www.solidworks.com/</a>)</td>
<td>10</td>
</tr>
<tr>
<td>2.3.3 Rhino (<a href="http://www.rhino3d.com/">http://www.rhino3d.com/</a>)</td>
<td>10</td>
</tr>
<tr>
<td>3 Background</td>
<td>13</td>
</tr>
<tr>
<td>3.1 B-spline Representation of Curves and Surfaces</td>
<td>13</td>
</tr>
<tr>
<td>3.2 Euclidean and Parametric Representation of Curves</td>
<td>16</td>
</tr>
<tr>
<td>3.3 Parameterizations for B-spline Curves</td>
<td>17</td>
</tr>
<tr>
<td>3.4 Least Squares Fitting using B-spline Functions</td>
<td>18</td>
</tr>
<tr>
<td>3.5 Rotation Minimizing Frame</td>
<td>19</td>
</tr>
<tr>
<td>4 Curve Creation and Editing</td>
<td>21</td>
</tr>
<tr>
<td>4.1 Sketching of Curves</td>
<td>21</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>4.2</td>
<td>Projection of Curves</td>
</tr>
<tr>
<td>4.3</td>
<td>Direct Editing of Curves on a Surface</td>
</tr>
<tr>
<td>5</td>
<td>Geometric Constructors for Surface Creation and Editing</td>
</tr>
<tr>
<td>5.1</td>
<td>Cutting Surfaces</td>
</tr>
<tr>
<td>5.2</td>
<td>Sweeping Surfaces Out of Surfaces</td>
</tr>
<tr>
<td>6</td>
<td>Results</td>
</tr>
<tr>
<td>7</td>
<td>Conclusions</td>
</tr>
</tbody>
</table>

Bibliography: 53
List of Tables
List of Figures

3.1 Curves with similar control polygon and different knot vectors. (a) Open-end curve. (b) Floating-end curve. (c) Periodic curve. .................................................................................. 14

3.2 B-spline surfaces with different end conditions. (a) Open-end conditions in both $u$ and $v$ directions. (b) Floating-end condition in $u$ direction and open-end condition in $v$ direction. .................................................................................. 16

4.1 A drawing of a face is sketched on a plane. Same input set of points is used to create the drawn curves with different end conditions. In all cases the eyes are created with periodic end conditions. The final sketch and the control polygon of the curves are shown. In (a) and (d) the nose and the mouth are created with an open-end condition. In (b) and (e) the nose is created with an open-end condition and the mouth with a floating-end condition. Finally, in (c) and (f) the nose is created with a periodic end condition and the mouth with an open-end condition. .................................................................................. 23

4.2 A drawing of a face is sketched on a plane. The same set of input points is used for the mouth curve with different parameterizations. In (a), least squares approximation with centripetal parameterization, in (b), least squares approximation with chord length parameterization, and in (c), least squares approximation with uniform parameterization. .................................................................................. 24
4.3 Same set of input points used to create the mouth curves with different number of control points during the least squares fitting. In (a), six percent of the original set of points is used in the control polygon’s construction. In (b), twenty percent of the original set of points is used in the control polygon’s construction. Finally in (c), ninety percent of the original set of points is used in the control polygon’s construction.

4.4 Two different views of a single curve, that is sketched by one stroke over three different surfaces - a cube, a sphere and a cylinder.

4.5 (a) A drawing of a sand watch is projected onto a model of a laptop computer screen. (b) A face drawing and the words "Be Happy!" are projected onto a model of a cup.

4.6 (a) Projecting a curve $C$ on a cylinder in the $V = Y$ direction. (b) Result of the projection - the curve is projected both on the front and the back walls of the cylinder. $C_{p1}$ is considered as the actual projected curve.

4.7 (a) Projecting a circle on a sphere in the $V = -Z$ direction. (b) Result of the projection - only part of the circle is projected, the other part misses the sphere surface.

4.8 (a) Projecting a circle on a cone with a hole in $V = Y$ direction. (b) Result of the projection - the upper part of the circle is projected on the back wall of the cone and the lower part of the circle is projected on the front wall of the cone.

4.9 (a) A transformation controller, that can be attached to a transformed curve. The square handles are used for translation (with holding the Ctrl key) and non uniform scaling (with holding the Shift key). The circular ones are used for rotation (with holding the Ctrl key) and uniform scaling (with holding the Shift key). (b) The transformation controller applied to a curve on a surface of a wine glass.

4.10 (a) A drawing of a truck is projected on a cylinder and selected for direct manipulation on the surface. (b) The drawing is rotated on the cylinder. (c) A uniform scaling and translation is applied to the truck’s drawing on the cylinder surface.
5.1 A drawing of an animal is trimmed from a surface. As a result two new trimmed surfaces are created. (a) First trimmed surface. (b) Second trimmed surface.

5.2 A horse is drawn on a cone and passes through the surface boundary. Hence, the trimming operation results in a creation of three new trimmed surfaces. (a) First trimmed surface. (b) Second and third trimmed surfaces viewed together. (c) Second trimmed surface. (d) Third trimmed surface.

5.3 Creation of a sweep surface. (a) Two input curves: an axis curve $A(t)$ and a section curve $C(r)$. (b) Curve $C(r)$ is transformed along the axis curve $A(t)$. (c) The resulting sweep surface $W(r, t) = A(t) + H(t)[C(r)]$.

5.4 Connecting the original surface $S$ with a sweep surface $W$. (a) $C^0$ continuity between $S$ and $W$. (b) $G^1$ continuity between $S$ and $W$.

5.5 Two alternatives for computing the offset curve $D$. (a) Offset all section curves in direction $F(u, v)$ from size $C$ to size $D$. (b) Offset the first section, $D$, of the sweep surface in the direction of $-F(u, v)$.

5.6 Different amount of offset on curve $D$ is used, controlling the size of the created rounding that achieves $G^1$ continuity over the surfaces at $C$.

5.7 Two alternatives for positioning the created sweep surface $W$ in space. (a) Location of the axis curve, $A$, and section curve, $C$, in space. (b) On the left - the centroid location of $C$ is placed at the starting location of the axis curve and the sweep is constructed around the axis curve $A$; on the right - the axis curve, $A$, is brought to the centroid location/orientation of $C$.

5.8 After the initial construction, the sweep surface $W$ is presented to the user in editable mode. (a) Section curves that can be further manipulated are highlighted in bold. (b) Selecting a specific section curve - the curve changes its color and a transformation controller appears surrounding it. More than one section curve can be selected in a time. In this case, the applied transformation will effect all the selected curves simultaneously.
5.9 Building the sweep surface, $W$, from different section curves. (a) Input curves - axis $A$ and three different section curves $C_i$. (b) Using $C_1$ for the first section curve, $C_2$ for the last section curve and $C_3$ for the rest of the curves.

5.10 Various transformation applied to the constructed surface $W$. (a) Initialy constructed surface $W$. (b) (From left to right) The second and forth section curves are scaled, the last one is rotated. (c) Introducing additional section curves to the original surface $W$ from (a). (d) (From left to right) The new second and forth section curves are rotated, the sixth one is scaled.

5.11 Rotating the axis curve, $A$, around the centroid location of $C$. (a) Rotation of 0 degrees. (b) Rotation of 145 degrees.

6.1 A model of a teapot.
6.2 A model of a cactus.
6.3 A model of a clock.
6.4 A model of a phone.
Abstract

This thesis deals with sketch based creation and modification of 2D and 3D freeform geometry. Such capabilities ease the conversion of inaccurate rough sketches, either two-dimensional or three-dimensional, into an accurate geometric representation, thus making the modeling process a more natural and intuitive task. The user mainly specifies his/her desired operation using freeform strokes on the screen, and the system infers the users intent and executes the appropriate creation and editing operations.

An essential part of this research effort is focused on creating a solid modeling environment that will provide all the functionality of traditional geometric design systems, yet will introduce new modeling methods that are intuitive to novice users. As we consider Boolean operations counter-intuitive for novice users, a direct modeling scheme of complex freeform geometry of a similar power is proposed as an alternative. Such operations include curve manipulation on surfaces, trimming interior regions of the surface along predefined curves and creating new freeform surfaces from scratch based only on a few strokes. The entire framework allows the modeling of complex freeform geometry in an intuitive manner, all while avoiding Boolean operations.
## Notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(t)$</td>
<td>B-spline curve</td>
</tr>
<tr>
<td>$P_i(x_i, y_i, z_i)$</td>
<td>$i$'th control point of curve $C$</td>
</tr>
<tr>
<td>$B_{i,n}(t)$</td>
<td>$i$'th B-spline basis function of degree $n$ of curve $C$</td>
</tr>
<tr>
<td>$Q_j$</td>
<td>$j$'th collected data point from user input</td>
</tr>
<tr>
<td>$U(t)$</td>
<td>Moving orthonormal 3D frame attached to curve $C$</td>
</tr>
<tr>
<td>$S(u,v)$</td>
<td>B-spline surface</td>
</tr>
<tr>
<td>$C_p(t)$</td>
<td>The result projection curve of curve $C$ onto surface $S$</td>
</tr>
<tr>
<td>$M(u(t), v(t))$</td>
<td>A transformation applied to curve $C(t)$ on surface $S$</td>
</tr>
<tr>
<td>$W(r,t)$</td>
<td>A sweep surface constructed from an axis curve $A$ and a section curve $C$</td>
</tr>
<tr>
<td>$A(t)$</td>
<td>An axis curve of a sweep surface $W$</td>
</tr>
<tr>
<td>$H(t)$</td>
<td>A transform of curve $C$ at location $A(t)$</td>
</tr>
<tr>
<td>$N(u,v)$</td>
<td>Surface normal</td>
</tr>
<tr>
<td>$F(u,v)$</td>
<td>Vector field of surface $S$ along $C$</td>
</tr>
<tr>
<td>$D(u,v)$</td>
<td>Offset curve on a surface $S$</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Geometric CAD systems have been a standard tool in the product development process for a long time. Over the years they have become very powerful, but also very complex and are therefore often seen as a separate scientific discipline. A typical designer takes a long time learning the CAD modeling functions and how to handle them correctly. Even when he/she has reaches this level, the CAD system is still not very intuitive to use. This holds especially in the early stages of the product design in the CAD system - stages that are usually not flexible enough for the designer’s needs. CAD systems are designed to model exact representations, and do not support well fast and intuitive generation of raw sketches, which are often the only one created during the conceptual stage of design.

Because CAD systems are not very intuitive and do not efficiently support the early phases of design, significant research has gone into developing new conceptual sketching modeling tools. Perhaps the earliest computerized sketching system (in fact the earliest CAD system) was Sutherland’s Sketchpad [27]. In this system, the user uses a light pen to draw on a screen, and manipulates graphic primitives such as arcs and lines. Since Sketchpad, numerous graphic’s drawing packages have been developed, but only a few of them aimed to understand the picture being drawn, in the sense that they detect relationships not explicitly specified by the user, or connect individual components to form a larger context, as humans may do when looking at a sketch. Moreover, even fewer of these systems support true freehand sketching, let
CHAPTER 1. INTRODUCTION

alone freehand sketches of three-dimensional objects.

This research aims at providing a simple and intuitive environment for the creation of computerized freeform geometry, in 2D as well as 3D, an environment that both novice and professional users will find convenient to use. A major goal has been the development of intuitive constructors based on free-hand sketching toward simple creation of freeform curves and surfaces both in 2D and 3D. The designed system allows the creation of curves by drawing simple strokes on the screen, resulting in 2D geometry, or on an underlying surface, thus creating freeform 3D curves. Surfaces are interpreted from numerous strokes defining their boundaries or skeleton. In addition, wide range of editing operations are supported once the creation process is complete.

This work was developed as a modeling extension to the GuIrit modeling environment. The GuIrit package is developed at the Technion and serves as a graphical user interface (GUI) for the Irit solid modeling kernel (see www.cs.technion.ac.il/~irit and www.cs.technion.ac.il/~gershon/GuIrit for further information). The research results of this work are implemented as two new shared library extensions to the GuIrit modeling environment, the “curve sketching” and the “surface sketching” extensions.

The following chapters are organized as follows. Chapter 2 summarizes recent related work that has been conducted in the geometric modeling field. Chapter 3 provides the mathematical background for our research. Chapters 4 and 5 detail the geometric constructors used to create freeform curves and surfaces respectfully. In Chapter 6, we present a variety of objects created with our new modeling environment. Finally, in Chapter 7, we summarize this research and discuss possible future work.
Chapter 2

Related Work

In this chapter, we present the state-of-the-art knowledge in creating 2D and 3D sketching tools; Section 2.1 surveys contemporary approaches for polygonal models creation; Section 2.2 discusses the creation of freeform geometry and Section 2.3 presents existing commercial products that offer such capabilities.

2.1 Creation of Polygonal Geometry

In the past, quite a few systems were developed to simplify and automate the process of 3D geometry creation using freehand sketches. The end result of most of those systems is a polygonal geometric model. Kato et al [18] described a system for interactive processing of freehand sketched diagrams. Their system is two-dimensional. It recognizes two-dimensional primitives such as lines, circles, flow-chart symbols and some Chinese characters. Spur and Jansen [12] presented a system for automatic recognition of hand drawn contours for CAD applications. Jenkins et al. [16] described a system called Easel for online (interactive) freehand sketching of two-dimensional graphics composed of lines, arcs and Bezier curves. Easel accepts direct freehand sketching and tolerates inaccuracies. It avoids the use of menus so as not to impede the creative process, and, therefore, automatically distinguishes between stroke types and infers the implicit constraints among them. Bengi and Ozguc [24] described a system for architectural sketch recognition. Their system processes freehand sketches of
two-dimensional architectural plans, and identifies and recognizes linear segments and circular arcs. Hwang and Ullman [14] described a system that unites ideas from design with three-dimensional layouts and knowledge engineering, based on an extensive study of mechanical engineers in action. Eggli et al [5] proposed a solid modeler incorporating a sketching tool. Their system is three-dimensional but the sketching itself is always constrained to some plane, thereby avoiding the problematic reconstruction phase. A similar system for designing solid objects using interactive sketch interpretation is described by Pugh [25]. The SKETCH application described in [28] attempted to create an environment for rapidly conceptualizing and editing approximate 3D scenes. SKETCH uses simple non-photo realistic rendering and a purely gestured based interface on simplified line drawings of primitives that allow all operations to be specified within the 3D world. Igarashi [15] suggested a system called Teddy, which allowed modeling of freeform surfaces with a very simple interface. The procedure requires of drawing the object’s silhouette, and then the application provides a polygonal mesh adapted to the object’s silhouette. [17] introduced SmoothSketch - a system for inferring plausible 3D shapes from visible-contour sketches. The system allowed completion of hidden contours, topological shape reconstruction and smooth embedding of the shape via relaxation. The ShapeShop [26] sketch-based modeling system introduced Hierarchical Implicit VolumeModels (BlobTrees) as an underlying shape representation. The BlobTree framework supported interactive creation of complex, detailed solid models with arbitrary topology. Sketch-based modeling operations were defined that combine these basic shapes using standard blending and CSG operators. [23] proposed a new approach of reconstructing the object from its projection. This new approach to the reconstruction problem was based on the concept of geometric correlations. These correlations reflect statistical geometric relationships between spatial and projected man-made objects. They were learned based on observations made from a database of sketches that was also used for scene recovery.
2.2 Creation of Freeform Geometry

Systems for processing of sketches depicting freeform geometry are less common. Lamb and Bandopadhay [21] described a system for interpreting a three-dimensional object from a rough two-dimensional sketch. Their system accepts the sketch from a variety of sources, and then proceeds to reconstruct the three-dimensional object by sequential reconstruction of adjacent faces, starting at a corner to which they assign an orthogonal coordinate system. Eggli, Hsu, Bruderlin and Elber [6] proposed a 2D/3D modeling tool for pen based computers. Users of this system define a model by simple pen strokes. The system can also be used to sketch 3D solid objects and B-spline surfaces. [13] presented a novel approach which adopted a simple 3d sketching technique and a finite element deformation method to create full free-form models. In the proposed method, the user applies interactive spline sculpting to modify a surface in a predictable way.

2.3 Commercial Products

Numerous 3D modeling packages and kernels are available today on the market. Here, we sample three sketching tools in popular systems for the naive as well as the professional user.

2.3.1 SketchUp (http://sketchup.google.com/)

Google’s SketchUp is a tool that allows unprofessional users to create and edit 3D models with ease. However, the resulting geometry is polygonal and rather simplistic. It is mainly based on a basic extrusion operation of a flat rectangular polygon into a three dimensional form. The recent SketchUp version also allows the creation of a sweep and a surface of revolution by the standard known methods. The available line tool allows one to define and edit a curve through points lying on it (interpolation). A point can be added to a curve, removed from it or can change the curve’s continuity by dragging handles attached to it.
2.3.2 SolidWorks (http://www.solidworks.com/)

SolidWorks can construct 2D sketch curves using splines and analytic geometry such as lines, arcs and conic sections. It can construct curves through three-dimensional reference points that are either fixed in space or attached to an existing model. But - it cannot ensure tangency or $C^2$ continuity at the curve’s endpoints. SolidWorks can also build 3D curves using what it calls a 3dsketch. This type of sketch allows one to define a curve on a plane or a 3D surface through number of points (interpolation) with the constraint that the curve will lay on the surface. No free sketching and no direct manipulation of the curve on a surface is available in this case. SolidWorks also allows the projection of a planar curve onto a surface, but the projection direction is chosen automatically and is parallel to the plane’s normal on which the (planar) curve resides. SolidWorks also allows sweep surfaces with multiple guidelines (section curves) swept along a trajectory. More complex shapes can be created using the variable sweep section feature. It sweeps a profile along a path and allows the designer to define multiple paths (often called guides or rails) that constrain the profile to follow them. The guides alter the length and shape of the swept section as it follows the path. An unlimited number of guides can be used with the path and profile, but one cannot make the guide curves tangent or $C^2$ continuous with adjacent shapes.

2.3.3 Rhino (http://www.rhino3d.com/)

Rhino allows the creating of 2D/3D curves through the selection of control points’ locations, interpolation through selected points, and free 3D sketching. Sketching on a surface is limited to one surface (no multiple surface drawing) and the underlying surface must be explicitly selected first. Further more, one must select whether s/he desires to draw on a mesh (polygonal) or on a NURBS surface. Projection option is available only for planar curves and is always orthogonal to the construction plane of the curve. The resulting projected curve must be then manually simplified by the “rebuild command”, that reduces the number of control points, in order to be used in further operations. Sweeps are created by choosing one or more section curves and an axis curve. For closed section curve, one must manually align them to the same
direction for a correct sweep output. Once the sweep surface is created, its section curves cannot be edited, and only general NURB surface editing is provided (editing the iso curves and the control mesh).

All the above reveals that modern modeling packages continue to lack fully free-hand sketching abilities. Complex geometry is still built via Boolean operations between simpler entities, general sketching on complex geometry is unavailable, curves’ projections are restricted and the curves on surfaces and sweep surface constructors are limited and/or counter-intuitive to use.
Chapter 3

Background

In this chapter, we discuss the necessary background and define the mathematical tools that are employed in this work. Section 3.1 surveys the B-spline representation of curves and surfaces; Section 3.2 discusses Euclidean versus parametric representations; Section 3.3 presents different parameterizations that can be used to create curves and surfaces; Section 3.4 shows how to fit a parametric spline curve through a set of points using a Least Squares Fitting; and Section 3.5 discusses a possible way to establish reference frames for locating cross-sections on an axis of a sweep surface.

3.1 B-spline Representation of Curves and Surfaces

Given \( m + 1 \) real values \( t_i \in [0,1] \), called knots, \( t_0 \leq t_1 \leq ... \leq t_m \), a parametric B-spline curve of degree \( n \), \( C : [t_n, t_{m-n}) \rightarrow \mathbb{R}^k \), is composed of B-spline basic functions of degree \( n \), \( B_{i,n}(t) \):

\[
C(t) = \sum_{i=0}^{m-n-1} P_i B_{i,n}(t), \quad t \in [t_n, t_{m-n}).
\]  

(3.1)

The \( P_i \) are called \textit{control points}. A polygon can be constructed by connecting the control points, in order, with edges, from \( P_0 \) and to \( P_{m-n-1} \). The \( m - n \) B-spline
basis functions of degree $n$ are defined using the following recursion formula:

$$B_{j,0}(t) = \begin{cases} 
1 & \text{if } t_j \leq t < t_{j+1}, \\
0 & \text{otherwise.}
\end{cases}$$

$$B_{j,n}(t) = \begin{cases} 
\frac{t - t_j}{t_{j+n} - t_j} B_{j,n-1}(t) + \frac{t_{j+n+1} - t}{t_{j+n+1} - t_{j+1}} B_{j+1,n-1}(t) & \text{if } t_j \leq t < t_{j+n+1}, \\
0 & \text{otherwise.}
\end{cases}$$

(3.2)

We assume that $t_{j+n} > t_j, \forall j$. That is, no multiplicity of knots of more than $n$ is allowed. Moreover, if one of the denominators in Equation (3.2) is zero, the whole term is defined to be equal to zero.

Use of different knot vectors with the same control polygon and the same degree of B-spline basis functions results in similar yet different curves. If the knot vector $\{t_j\}_{j=0}^{m}$ satisfies $t_0 = \ldots = t_n$ and $t_{m-n} = \ldots = t_m$, the resulting curve is called an open-end B-spline curve, see Figure 3.1(a), and its end points interpolate $P_0$ and $P_{m-n-1}$. A curve for which $t_0 < t_n$ or $t_{m-n} < t_m$ is called a floating-end B-spline curve and such a curve interpolates neither $P_0$ nor $P_{m-n-1}$. Figure 3.1(b) is an example of a floating curve. A periodic B-spline is a curve which closes on itself and is defined with a periodic parametric domain, that is $C(t) = C(t + \sigma(t_n - t_{m-n}))$, where $\sigma \in \mathbb{Z}$. See Figure 3.1(c) for an example.

![Figure 3.1](image)

Figure 3.1: Curves with similar control polygon and different knot vectors. (a) Open-end curve. (b) Floating-end curve. (c) Periodic curve.

Let $\varphi_j = t_{j+1} - t_j$. If $\varphi_j = \varphi_{j+1}, \forall j$, all the B-spline basis functions of this knot
sequence are the same up to translation, and produce curves that are called uniform B-splines curves. Any curves whose knot sequence doesn’t meet this uniformity condition is called a nonuniform B-spline curve.

When the number of control points in a B-spline curve is the same as the degree + 1, the B-spline curve degenerates into a polynomial curve. If one sets the knot vector such that $\alpha = t_0 = \ldots t_n < t_{n+1} = \ldots t_{2n+1} = \beta$, then the curve using this knot sequence identifies with a Bezier curve of degree $n$ over the parametric domain $[\alpha, \beta]$, and has the same control points.

We define the tensor product B-spline surface as the following piecewise polynomial surface:

$$S(u, v) = \sum_{i=0}^{l-n_u-1} \sum_{j=0}^{m-n_v-1} P_{i,j} B_{j,n_v}(v) B_{i,n_u}(u),$$

(3.3)

where $B_{j,n_v}(v)$ are the B-spline basis functions of degree $n_v$ on the knot vector $v = \{v_j\}_{j=0}^l$ in the $v$ direction, and $B_{i,n_u}(u)$ are the B-spline basis functions of degree $n_u$ on the knot vector $u = \{u_j\}_{j=0}^m$ in the $u$ direction. The set of points $\{P_{i,j}\}$ is called the control mesh. If both knot vectors satisfy the Bezier condition, defined above, the surface is called a Bezier surface. If, in Equation (3.3), we set the $u$ parameter to be constant, $u = u_0$, and let $v$ vary inside its parametric domain, we get a curve, $S(u_0, v)$, that is called an isoparametric curve of surface $S(u, v)$ in the $v$ direction. The same holds for isoparametric curves of surface $S(u, v)$ in the $u$ direction. Like a B-spline curve with open-ends, floating-ends or periodic, a B-spline surface can also have different end conditions in either $u$ or $v$. If a B-spline surface is open-ended in both directions, this surface interpolates the four corner points $P_{0,0}$, $P_{l-n_u-1,0}$, $P_{0,m-n_v-1}$ and $P_{l-n_v-1,m-n_v-1}$ and is tangent to the eight edges of the control mesh at these four corner points. If a B-spline surface has floating end conditions, then the surface does not interpolate these corner points. If a B-spline surface is periodic in some direction, all isoparametric curves in this direction are periodic as well. Figure 3.2 depicts some of these conditions.
CHAPTER 3. BACKGROUND

3.2 Euclidean and Parametric Representation of Curves

Typically, a three dimensional B-spline curve is using a Euclidean representation in \( \mathbb{R}^3 \), where a triplet \( P_i(x_i, y_i, z_i) \) depicts the control points’ position in space. Alternatively, if the curve happens to lay on a surface \( S(u, v) \), then \((u_i, v_i)\) pairs can provide a parametric representation of the curve’s control points, in the underneath surface. The 3D Euclidean locations can be recovered from the parametric representation at any time as

\[
P_i(x_i, y_i, z_i) = \{S_x(u_i, v_i), S_y(u_i, v_i), S_z(u_i, v_i)\}.
\]

In general, a space curve that all its control points are on an underneath surface \( S \) is unlikely to lay completely inside \( S \). This holds not only for a general B-spline curve but also for a linear B-spline curve. This means that even a polyline sampled along the curve so its points are on \( S \) will not be completely in \( S \), in general. Let \( C(t) = (u(t), v(t)) \) be a regular parametric curve in surface \( S(u, v) \). Then, the composition \( S(C(t)) \) [7] is the precise Euclidean realization of \((u(t), v(t))\) on \( S \). Unfortunately, the computation of this composition can be expensive at times and of high degree (that is a function of the degrees of both \( C \) and \( S \)). By carrying \( C \) along, as \((u(t), v(t))\), we can

![Figure 3.2: B-spline surfaces with different end conditions. (a) Open-end conditions in both \( u \) and \( v \) directions. (b) Floating-end condition in \( u \) direction and open-end condition in \( v \) direction.](image-url)
possibly avoid the need for explicitly computing this composition while evaluating the Euclidean location of \( S(C(t)) \) at every point and as needed, and to arbitrary precision.

### 3.3 Parameterizations for B-spline Curves

A common task in geometric modeling is to interpolate/approximate a sequence of points \( P_0, ..., P_n \), sampled from a curve in \( \mathbb{R}^k \), with a parametric spline curve. We choose parameter values \( t_0 < ... < t_n \) and find a parametric curve \( C : [t_0, t_n] \rightarrow \mathbb{R}^k \), with \( k \geq 2 \), such that \( C(t_i) = P_i \) for \( i = 0, ..., n \). Empirical evidence has shown that the choice of parameterization has a significant effect on the visual appearance of the interpolant \( C \). Hence in this work we allow the user to choose from four different parameterizations that better suit his/her needs:

1. The simplest choice is the uniform parameterization defined by \( t_i = i \), where the values \( t_i \) are uniformly spaced.

2. Another parametrization is the chord length parametrization proposed in 1967 by Ahlberg, Nilson, and Walsh [1] in which \( t_{i+1} = t_i + \| P_{i+1} - P_i \| \) and \( \| \cdot \| \) denotes the Euclidean norm in \( \mathbb{R}^k \). The motivation behind this choice is that the distance between two points on a curve is a reasonable approximation to the length of the associated curve segment. Thus, the speed \( |C'(t)| \) of the spline curve might be closer to a unit speed at all \( t \in [t_0, t_n) \).

3. The third option, which was termed by Lee [22] as the centripetal parametrization is defined as \( t_{i+1} = t_i + \| P_{i+1} - P_i \|^\frac{1}{2} \). It is suggested that it sometimes leads to a spline curve which is closer to the polygon passing through the data points \( P_i \).

4. More recently, Foley and Nielsen [11] observed that the chordal parameterization lacks affine invariance. If a non-uniform scaling is applied to the data points \( P_i \) and the new chord lengths are computed, then the resulting spline curve will not in general be a scaling of the original. They suggested replacing the Euclidean distance from above by a metric which is invariant to affine
transformations of the data points. The *Nielson-Foley parameterization* is defined with the aid of both the affine-invariant distance between \( P_i \) and \( P_{i+1} \) and the angles of the polygon passing through \( P_{i-1}, P_i, P_{i+1}, P_{i+2} \). Thus, unlike the previous methods, every value of the metric depends on all the data points \( P_0, ..., P_n \).

### 3.4 Least Squares Fitting using B-spline Functions

Given a set of collected data points, \( Q_j, j = 0, ..., q \), we would like to approximate a parametric spline curve through the collected points, in such a way that will minimize the error between the original data set and the resulting curve. The parameters \( t_j \) can be selected in one of the ways described in Section 3.3.

For a specified set of control points \( \{P_i\}, i = 0, ..., m - n - 1, (m - n < q) \), the least-squares error function between the B-spline curve and sample points \( Q_j \) is the scalar-valued function

\[
E(\hat{P}) = \sum_{j=0}^{q} \left( \sum_{i=0}^{m-n-1} B_{i,n}(t_j) P_i - Q_j \right)^2.
\]

(3.4)

\[\sum_{i=0}^{m-n-1} B_{i,n}(t_j) P_i\] is the point on the B-spline curve at time \( t_j \). The term within the summation on the right-hand side of Equation (3.4) measures the squared distance between the sample point and its corresponding curve location. The error function \( E(\hat{P}) \) measures the total accumulation of squared distances. The hope is that we can choose the control points while minimizing this error.

The minimization is an optimization problem. The function \( E \) is quadratic in the components of \( \hat{P} \), its graph a paraboloid, so it must have a global minimum that occurs when all its first-order partial derivatives vanish. The first-order partial
derivatives are written in terms of the control points $P_k$

$$\frac{\partial E}{\partial P_k} = 2 \sum_{j=0}^{q} \left( B_{k,n}(t_j) \sum_{i=0}^{m-n-1} (B_{i,n}(t_j)P_i - Q_j) \right)$$

$$= 2 \sum_{j=0}^{q} \sum_{i=0}^{m-n-1} B_{k,n}(t_j)B_{i,n}(t_j)P_i - 2 \sum_{j=0}^{q} B_{k,n}(t_j)Q_j$$

$$= 2 \sum_{j=0}^{q} \sum_{i=0}^{m-n-1} a_{j,k}a_{j,i}P_i - 2 \sum_{j=0}^{q} a_{j,k}Q_j, \quad (3.5)$$

where $a_{r,c} = B_{c,n}(t_r)$ for all $0 < k < m - n - 1$. Setting the partial derivatives equal to the zero vector leads to the system of equations,

$$0 = \sum_{i=0}^{m-n-1} a_{j,k}a_{j,i}P_i - \sum_{j=0}^{q} a_{j,k}Q_j = A^T \hat{A} \hat{P} - A^T \hat{Q}, \quad (3.6)$$

where $A = [a_{r,c}]$ is a matrix with $m - n$ rows and $q + 1$ columns, $\hat{P} = \{P_i\}$ and $\hat{Q} = \{Q_j\}$.

### 3.5 Rotation Minimizing Frame

The construction of a sweep surface from an axis and a set of cross section curves, given as an input, requires establishing reference frames to locate the cross-sections on the axis. The rotation minimizing frame (RMF), which is a moving orthonormal 3D frame $U(t)$ attached to a smooth curve $C(t)$ in 3D such that $U(t)$ does not rotate about the instantaneous tangent of $C(t)$, can be used for this purpose. In general, one often needs to approximate the RMF by a sequence of orthonormal frames at sampled points on the axis curve.

The RMF computation problem can be formulated as follows. Let $U(t)$ denote an RMF of a $C^1$ regular axis curve $C(t)$ in 3D, $t \in [0, L]$, with the initial condition, $U(0) = U_0$, which is some fixed orthonormal frame at the initial point $C(0)$. Suppose that a sequence of points $C_i = C(t_i)$ and the unit tangent vectors $T_i$ at $C_i$ are sampled on the curve $C(t)$, with $t_i = i \ast h, i = 0, 1, \ldots, k$, where $h = L/k$ is called the step size.
The goal is to compute a sequence of orthonormal frames $U_i$ at $C_i$ that approximates the RMF frame $U(t)$ at the sampled points, i.e., each $U_i$ is an approximation to $U(t_i), i = 0, 1, 2..., k$.

In this work, we used a popular discrete approximation method, called the rotation method [3]. The rotation method needs as input the sampled points $C_i$ on the axis curve and the unit tangent vectors $T_i$ of the curve at $C_i$. Consider two adjacent sampled points $C_i$ and $C_{i+1}$. Given the frame $U_i$ at $C_i$, compute the next frame $U_{i+1}$ at $C_{i+1}$ from the boundary data ($C_i, T_i; C_{i+1}, T_{i+1}$). To minimize the rotation about the tangent of the axis curve, this method rotates $U_i$ into $U_{i+1}$ about an axis $B_i$ perpendicular to $T_i$ and $T_{i+1}$, that is, $B_i = T_i \times T_{i+1}$; the rotation angle is such that the frame vector $T_i$ of $U_i$ is brought into alignment with the frame vector $T_{i+1}$ of $U_{i+1}$, i.e., $\alpha = \arccos(T_i \cdot T_{i+1})$. Note that if $T_i \equiv T_{i+1}$ then no rotation is necessary.
Chapter 4

Curve Creation and Editing

This section introduces some geometric modeling techniques that, we believe, are intuitive for novice users. We examine different ways of creating planar and spatial curves from sketches of input devices (i.e. a mouse). Aiming at direct manipulation and immediate feedback, we adapt techniques to reconstruct B-spline curves from a set of input points (possibly from a mouse input device). Techniques to interactively and incrementally build B-spline curves where explored in the past (i.e. [2], [10]). Herein, we portray the representation we chose, a representation that will allow us further manipulation of these curves. In Section 4.1, we discuss how the curves are created with this selected representation. Section 4.2 presents an alternative way of creating spatial curves over surfaces and Section 4.3 introduces a manipulation scheme of curves on the surfaces, once created.

4.1 Sketching of Curves

We allow the creation of planar curves in several ways. The user can simply sketch a curve on a plane, that is displayed in the drawing area or he/she can draw the curve in space without choosing any visible underlying plane. In this later case, the curve will lie on a plane that is parallel to the screen and passes through the center of the current scene’s bounding box. It is also possible to construct non planar curves by sketching on arbitrary surfaces.
Sketching of a curve is reduced to clicking and dragging the mouse in space or over the surface, whether planar or not. Every click-and-drag event is processed, and the mouse position on the screen is evaluated. The point of intersection of the ray starting at the mouse position and perpendicular to the screen, and the surface that the curve is sketched on, becomes one of the sketched points. The sketched data is converted to a freeform B-spline curve with (control) points in $\mathbb{R}^3$ as $P_i = (x, y, z)$ triplets, possibly using the Least Squares Fitting technique (Section 3.4). The $(x, y, z)$ triplet is the Euclidean representation of the point in space. When a curve is sketched on parametric surface, $S(u, v)$, then the $(u, v)$ pair provides additional information for each control point - the parametric representation of the point in the underneath surface $S$. Hence, when a curve is sketched on a parametric surface $S(u, v) = (S_x(u, v), S_y(u, v), S_z(u, v))$, the $(u, v)$ coordinates of every location introduced by the input device are determined and are considered the true representation. Carrying the underneath surface as part of the representation, the Euclidean locations could always be recovered from the parametric representation as $(S_x(u, v), S_y(u, v), S_z(u, v))$.

In general, a space curve that its control points are on the underneath surface $S$ is unlikely to lay completely inside $S$. This holds not only for a general B-spline curve but also for a linear B-spline curve. This means that even a polyline sampled along the curve so its points are on $S$ will not be completely in $S$. Let $C(t) = (u(t), v(t))$. Then, the composition $S(C(t))$ is the precise true realization of $(u(t), v(t))$ on $S$. Unfortunately the computation of this composition can be expensive at times and of high degree (that is a function of the degrees of both $C$ and $S$). By carrying $C$ along, as $(u(t), v(t))$, we can possibly avoid the need for computing the composition explicitly and evaluate the Euclidean location of $S(C(t))$ at every point and as needed, and to arbitrary precision.

Although the system automatically constructs a plausible curve as a result of a sketching operation, more advanced users can control different parameters that effect the end result of a sketching operation. In both the planar and the non planar cases, the user can decide whether the curve will be created as a periodic, i.e. closed, or non periodic, i.e. with open-end or floating-end conditions, curve (see Section 3.1).
See Figure 4.1 for an example.

Figure 4.1: A drawing of a face is sketched on a plane. Same input set of points is used to create the drawn curves with different end conditions. In all cases the eyes are created with periodic end conditions. The final sketch and the control polygon of the curves are shown. In (a) and (d) the nose and the mouth are created with an open-end condition. In (b) and (e) the nose is created with an open-end condition and the mouth with a floating-end condition. Finally, in (c) and (f) the nose is created with a periodic end condition and the mouth with an open-end condition.

The curve can be constructed from all the sampled points or by applying the least squares approximation technique to the sketched points (Section 3.4). Any one of the four parameterizations described in Section 3.3 can also be chosen. Figure 4.2 shows different curves created from the same set of input points with various parameterizations.

We predefine the order and the size of the B-spline curve, as a function of number of the control points. However, one can control the order of the created B-spline curve,
or the number of control points in the control polygon to be created during the least squares fitting, and it is possible to reuse the set of sketched points in order to recreate the newly desired curve. Figure 4.3 shows such an example.

The curve drawing operation is not confined to one underneath surface. A single continuous stroke can lie on multiple surfaces, simultaneously, and can circumvent
4.2 Projecting Curves

The creation of general shaped curves on surfaces is inherently imprecise. While direct sketching of curves has its obvious advantages, at times, a more precisely controlled shape creation might be desired. Consider a cylinder from which the user seeks to extract another cylindrical protrusion, in an orthogonal direction to the original cylinder’s axis. In order to facilitate this, we also offer more precise alternative that is also available in some CAD systems. Arbitrary planar or spatial curve(s) could be
projected on arbitrary surface(s) along a prescribed projection direction, $V$, resulting in a new projected curve(s) $C_p$.

Figure 4.5: (a) A drawing of a sand watch is projected onto a model of a laptop computer screen. (b) A face drawing and the words "Be Happy!" are projected onto a model of a cup.

This projection operation could be seen as a curve-surface-intersection problem between $S$ and a surface that is extruded from $C$ in the $V$ direction, $R(t,r) = C(t) + Vr$, $r \geq 0$. Having little hope for an analytic solution (unless a special surface-surface-intersection case like plane-cylinder intersection etc.), the result is approximated as a polyline. Nonetheless, the projected curve on $S$ is still saved as a linear B-spline with control points in $R^5$ as $(x,y,z,u,v)$ tuples. Figure 4.5 shows examples of curve projection onto a freeform models.

To calculate the projected curve, consider a point $T_i$ originated from some spatial curve $C$. A ray, $R$, is fired in the $V$ direction and the first hit, if any, of $R$ with the target surface $S$ is recorded. Clearly, $R$ can hit $S$ more than once. In this case, only the first hit will be considered for contributing to the result curve $C_p$. For Figure 4.6 for an example.

The source points $T_i$ on a original spatial curve $C$ can be chosen in two ways. $T_i$
can either be the control points of the curve $C$, i.e. $T_i = P_i$, or the curve $C$ can be first approximated as a polyline $L$ with points $L_i$. In this later case, the polyline’s points can be used as the source points $T_i$. While the second options yields more accurate result, but being more expensive to calculate and results in curves with large number of control points.

Clearly there are situations when the calculation of the projected curve may fail. In those cases no result curve $C_p$ will be created. See Figures 4.7 and 4.8 for an illustration of such cases.

1. $R$ can miss the surface $S$ at some point $T_i$.

2. Two adjacent locations on $C$ can become discontinuous, when projected onto (a discontinuity of) $S$.

4.3 Direct Editing of Curves on a Surface

Once a curve is created on the surface, we are interested in further editing and manipulating it. In [10], multi-resolution editing schemes were adapted to allow direct manipulation of curves over surfaces. In this work, we propose to similarly adapt

![Figure 4.6: (a) Projecting a curve $C$ on a cylinder in the $V = Y$ direction. (b) Result of the projection - the curve is projected both on the front and the back walls of the cylinder. $C_p^1$ is considered as the actual projected curve.](image)
the notion of direct editing of curves over freeform surfaces but for linear transformations. The notion of translating, rotating, and/or scaling of an object in the plane or in space is quite simple and intuitive. This concept is adapted here to allow some

Figure 4.7: (a) Projecting a circle on a sphere in the $V = -Z$ direction. (b) Result of the projection - only part of the circle is projected, the other part misses the sphere surface.

Figure 4.8: (a) Projecting a circle on a cone with a hole in $V = Y$ direction. (b) Result of the projection - the upper part of the circle is projected on the back wall of the cone and the lower part of the circle is projected on the front wall of the cone.
transformations of curves that are embedded in freeform surfaces.

Figure 4.9: (a) A transformation controller, that can be attached to a transformed curve. The square handles are used for translation (with holding the Ctrl key) and non uniform scaling (with holding the Shift key). The circular ones are used for rotation (with holding the Ctrl key) and uniform scaling (with holding the Shift key). (b) The transformation controller applied to a curve on a surface of a wine glass.

Consider a surface \( S(u,v) \). A transformation that is applied in the \((u,v)\) parametric space is mapped to Euclidean space by \( S \). Assume \( S \) is an Isometry. That is, \( S \) preserves distances in its mapping. Then, we can apply \( M \), a transformation to rotate, translate, and (uniformly or non-uniformly) scale \((u(t), v(t))\), as \( M(u(t), v(t)) \). Then, \( S(M(u(t), v(t))) \) will provide an immediate feedback on the new transformed location of \( C \) over \( S \), in the Euclidean space.

Unfortunately, in general, \( S \) is not an Isometry and hence only an approximation is computed while still providing an immediate interactive feedback to the result of the applied transformation. Let \((u_0, v_0)\) be a centroid location of \( C \) (the center of gravity of \( C \)) in the parametric domain of \( S \). By computing the Jacobian of the mapping \( S \) at \((u_0, v_0)\) one can locally approximate the affect of \( M \) onto the Euclidean space. If \( S \) is regular, the Jacobian is well defined everywhere, and one can always locally compensate for the non-Isometric behavior of \( S \) at \((u_0, v_0)\).
Globally, the shape of $C$ is likely to undergo some non-linear deformations due to $S$, as $C$ is linearly transformed in the domain of $S$ which is rarely an Isometry. Nonetheless, the fact that immediate interactive feedback of this transformation is provided turns out to be quite intuitive. In order to further alleviate the difficulty introduced by these non-linear deformations and as a graphical user interface to handle and prescribe $M$, a transformation control frame, is attached to curve $C$ once selected for transformation over $S$. See Figure 4.9. This transformation frame is a set of additional curves in the parametric domain of $S$ that are built to form a bounding box to $C$. This frame also offer handles to (mouse click-and-drag of) the rotation, translation and (possibly non-uniform) scaling of $C$. The underneath representation of the frame and $C$ is the same as sets of curves in the parametric domain of surface $S$. Hence, the frame and $C$ can be transformed together as $M$ is applied in the parametric domain of $S$, creating an intuitive and efficient transformation interface for transforming curves over surfaces. See Figure 4.10 for an example.

![Figure 4.10](image)

**Figure 4.10:** (a) A drawing of a truck is projected on a cylinder and selected for direct manipulation on the surface. (b) The drawing is rotated on the cylinder. (c) A uniform scaling and translation is applied to the truck’s drawing on the cylinder surface.
Chapter 5

Geometric Constructors for Surface Creation and Editing

Having created basic curves, in the plane and in space, over surfaces, we now employ the curves to construct surfaces. Section 5.1 discusses a simple cutting operation which allows cutting away holes or parts off a surface and Section 5.2 presents an extrusion and/or sweeping option.

5.1 Cutting Surfaces

The common representation for surfaces in contemporary modeling environments is of a tensor product. Trimmed surfaces are frequently used to circumvent the inherent restriction in the rectangular topology of the tensor products. Results of Boolean operations are, in most cases, represented as a set of trimmed surfaces that together define the 2-manifold boundary for the model. Direct manipulation of trimmed surfaces is certainly feasible and while we already portrayed Boolean operations as counter-intuitive for novice users, the idea of cutting a surface along an arbitrary curve drawn on the surface is quite simple to grasp. Hence, we do allow end users to cut surfaces along curves sketched over them. As a result, no topological constraints are imposed over the geometry to be a 2-manifold.

If the given stroked curve is closed and/or starts and ends on the boundary, the
surface could be split along the curve and two or (more) new trimmed surfaces are formed. See Figures 5.1 and 5.2. All surface pieces are presented to the user as new entities and the user can select to purge or use some of them. Curves that are neither closed nor connected to the boundary cannot be immediately used in the cutting process. While some gap(s), up to some tolerance, are allowed in the closed and/or boundary starting/ending curve, we must extend the curves to bridge these gap(s) before a cutting operation can be commenced successfully.

Figure 5.1: A drawing of an animal is trimmed from a surface. As a result two new trimmed surfaces are created. (a) First trimmed surface. (b) Second trimmed surface.
5.2 Sweeping Surfaces Out of Surfaces

The sweep constructor is another way we provide the user for creating surfaces out of given input curves. Given a curve, $C$, the user needs to prescribe another initial axis.

![Diagram of boundary with $u_{\text{max}}$ and $u_{\text{min}}$](image)

Figure 5.2: A horse is drawn on a cone and passes through the surface boundary. Hence, the trimming operation results in a creation of three new trimmed surfaces. (a) First trimmed surface. (b) Second and third trimmed surfaces viewed together. (c) Second trimmed surface. (d) Third trimmed surface.
curve, $A$, along which $C$ is to be swept. Then, a sweep surface $W$ can be constructed as $W(r,t) = A(t) + H(t)[C(r)]$, where $H(t)$ is the transform of $C(r)$ at location $A(t)$. In each control point of $A$, an orientation frame is used to place one section of $C$. See Section 3.5 that discusses a possible way to establish reference frames for locating cross-sections on an axis of a sweep surface. Figure 5.3 presents an example of a simple sweep surface.

![Figure 5.3: Creation of a sweep surface. (a) Two input curves: an axis curve $A(t)$ and a section curve $C(r)$. (b) Curve $C(r)$ is transformed along the axis curve $A(t)$. (c) The resulting sweep surface $W(r,t) = A(t) + H(t)[C(r)]$.](image)

If the input curve, $C$, resided on some surface, $S$, then the created sweep, $W$, will
be $C^0$ connected to the original surface, $S$, at $C$. See Figure 5.4(a) for an example. Because in many cases, $G^1$ continuity might be desired at the stitching curve, we apply the following process near $C(r) = (u(r), v(r))$, allowing the user to create a sweep surface that is $G^1$ continuous to the original surface $S$, from which it was swept (Figure 5.4(b)). Let,

\[
N(u,v) = \frac{\partial S}{\partial u} \times \frac{\partial S}{\partial u}, \quad (5.1)
\]

be the normal field of $S(u,v)$. The unit normal field of is $S$ not rational in general. Nonetheless, $N$ is rational, for a rational surface $S$. Consider the following vector field along $C(r) = (u(r), v(r))$ of

\[
F(u(r), v(r)) = \frac{d(S(u(r), v(r)))}{dr} \times N(u(r), v(r)) \quad (5.2)
\]

$F(u,v)$ is in the tangent space of $S$ for all locations on $C$ and is properly oriented.
provided that both $C$ and $S$ are regular, $C$ is not self intersecting over $S$, and $S$ is orientable. Then, define a new curve $D$ as

$$D(u(r), v(r)) = C(u(r), v(r)) + F(u(r), v(r))$$

(5.3)

and use $D$ as the second section curve of the constructed sweep. Because the first order cross derivatives of the sweep surface at $C$ are in the direction of $F$ now, the result is that the sweep surface and the original surface $S$ are $C^1$ at $C$.

Several issues must be noted. Equation (5.2) could be computed analytically as it is polynomial (See [20]) in which case the magnitude of $N$ and $C' = \frac{dS(u(r), v(r))}{dr}$ and hence $F$ must be (approximately) normalized. See [9] for a possible approximate normalization method of vector fields. Alternatively, $D$ can be approximated by sampling points along $C$ while locally evaluating Equations (5.2) and (5.3).

One can combine the above approach with offset approximation of $C$ over $S$. Because $D$ is some 3D offset of $C$ and due to the fact that $D$ is the second section of the sweep, the forthcoming sections must be of the same scale as curve $D$. Therefore, the alternatives are to either offset all forthcoming sections from size $C$ to size $D$, or offset the first section of the sweep surface in the direction of $-F(u, v)$. See Figure 5.5 for an illustration. The user is also offered additional control on the amount of offset used (the magnitude of the normalized field of $F(u, v)$), controlling the size of the created rounding that achieves $C^1$ continuity over the surfaces at $C$. See Figure 5.6 for a few examples, achieving a similar result to blending that is common when smooth connection between two surfaces is desired.

Since it is not always required to stitch $S$ and the constructed sweep $W$ at $C$, we also offer the user the option to position the constructed sweep in two alternative ways. Either the axis curve, $A$, is brought to the centroid location of $C$, or the centroid location of curve $C$ is placed at the starting location of the axis curve and the sweep is constructed around the axis curve. See Figure 5.7 for an example.

Once the initial sweep surface construction is complete, the created surface is presented to the user in an editable mode, highlighting section curves that can be further manipulated in different ways. (See Figure 5.8.) One can select a section
curve and then apply various transformation operations to customize the resulting sweep surface. It is possible to change an existing section curve by two means. One can completely replace the selected curve with a whole different curve (see Figure 5.9), or he/she can rotate or scale a section curve with a presented transformation controller, that appears once the curve is selected (Figure 5.10(b)). It is also possible to refine the created surface, whether globally or locally, in regions of interest and introduce additional section curves (and transformation controllers) into the surface. See Figure 5.10(c) and 5.10(d). Finally, one can also define the rotation angle of the axis curve, $A$, around the centroid location of $C$. Figure 5.11 demonstrates this option.

Figure 5.5: Two alternatives for computing the offset curve $D$. (a) Offset all section curves in direction $F(u, v)$ from size $C$ to size $D$. (b) Offset the first section, $D$, of the sweep surface in the direction of $-F(u, v)$. 
Figure 5.6: Different amount of offset on curve $D$ is used, controlling the size of the created rounding that achieves $G^1$ continuity over the surfaces at $C$. 
5.2. SWEEPING SURFACES OUT OF SURFACES

Figure 5.7: Two alternatives for positioning the created sweep surface $W$ in space. (a) Location of the axis curve, $A$, and section curve, $C$, in space. (b) On the left - the centroid location of $C$ is placed at the starting location of the axis curve and the sweep is constructed around the axis curve $A$; on the right - the axis curve, $A$, is brought to the centroid location/orientation of $C$. 
Figure 5.8: After the initial construction, the sweep surface $W$ is presented to the user in editable mode. (a) Section curves that can be further manipulated are highlighted in bold. (b) Selecting a specific section curve - the curve changes its color and a transformation controller appears surrounding it. More than one section curve can be selected in a time. In this case, the applied transformation will effect all the selected curves simultaneously.
5.2. SWEEPING SURFACES OUT OF SURFACES

Figure 5.9: Building the sweep surface, $W$, from different section curves. (a) Input curves - axis $A$ and three different section curves $C_i$. (b) Using $C_1$ for the first section curve, $C_2$ for the last section curve and $C_3$ for the rest of the curves.
Figure 5.10: Various transformation applied to the constructed surface $W$. (a) Initially constructed surface $W$. (b) (From left to right) The second and forth section curves are scaled, the last one is rotated. (c) Introducing additional section curves to the original surface $W$ from (a). (d) (From left to right) The new second and forth section curves are rotated, the sixth one is scaled.
5.2. SWEEPING SURFACES OUT OF SURFACES

Figure 5.11: Rotating the axis curve, \( A \), around the centroid location of \( C \). (a) Rotation of 0 degrees. (b) Rotation of 145 degrees.
CHAPTER 5. GEOMETRIC CONSTRUCTORS FOR SURFACE CREATION AND EDITING
Chapter 6

Results

This section presents a variety of additional objects that were created using the geometric operators developed in our research, and implemented as part of the GuIrit modeling environment.

For a teapot object in Figure 6.1, first a surface of revolution was created to represent its body. Then, two ellipses were sketched on the body at the locations of the spout and the handle. Next, the spout and the handle were swept out of the body surface, rotated around the centroid location of the stitching curves and their section curves selectively scaled for non uniform look. At the end, two holes were trimmed at the stitching curves between the body and the spout, and the body and the handle.

A cactus object in Figure 6.2 was created using the sketching, sweeping and trimming operators. First, the stem of the cactus was created by a sweep operation from a drawing of two input curves, and gradually scaled towards the apex. Then, two branches were recursively swept out of it using drawings of closed curves on the stem. The branches also underwent additional scaling and rotation editing.

The body of a clock in Figure 6.3 was created by combining two objects - a sweep and a planer surface that was trimmed out of a plane by a circular sketch. The ringer, legs and clock pointers were all swept out of the body and their section curves refined and edited. The digits were created by a projection of a text-font drawing.

A phone model in Figure 6.4 was created using only the sketching and sweeping operations. First, the body was created and then the handles were swept out of it,
refined and edited. In the end, a receiver was created from an additional sweep object, that was scaled at the far ends of the sweep.

Figure 6.1: A model of a teapot.
Figure 6.2: A model of a cactus.
Figure 6.3: A model of a clock.
Figure 6.4: A model of a phone.
Chapter 7

Conclusions

In our work, we have developed and adopted techniques for three dimensional direct modeling of freeform curves and surfaces. We presented intuitive modeling tools, where novice designers can easily create and intuitively edit relatively complex shape, while possibly keeping the end result within the desired precision. Those tools include curve sketching and direct editing on 2D and 3D freeform surfaces and use of those curves to create new surfaces by sweeping or editing existing ones by cutting. The presented approach provides the user with intuitive sketching and modeling capabilities in an interactive speed for creating relatively complex freeform shapes. Future work involves developing the following tools:

1. Connecting the swept object back to the original surface - For example, consider Figure 6.1 which presents an interesting topological problem where it might be desired to glue the handle back into the body, affecting the Genus of the model.

2. Automatic intuitive aggregation of curves over surfaces - Consider for instance a case where multiple holes should be made in a surface for keys in a keyboard. The creation of these holes one at a time could be highly tedious while an automatic intuitive aggregation of curves over surfaces could render such a process trivial.

3. Use of different symmetries for object creation - The use of Euclidean plane symmetry (once the right leg is complete, duplicate it to create the left leg by
plane symmetry) or rotational symmetry (once one hole is modeled, duplicate it around the surface, creating \( n \) holes) in intuitive modeling.

4. Ability to deform and create complex geometry in one surface - While the sweep is indeed a very powerful modeling operator, not every surface can be constructed using sweeps. For example, bending, twisting and deforming operations (See [4]) are highly intuitive to novice users and the outcome is also easy to anticipate.

5. More control over the editing of individual cross sections should be added so arbitrary cross sections could be formed.

6. Complex section curves could also be edited and created by (multi-resolution) editing as in [10], an option that should be seen part of a complete solution.

7. Direct multi-resolution editing [19] of freeform curves and surfaces should probably be made available as well.
Bibliography


לפי עקומות קיימות. בנוסח הפעולה המצוינת, גוף למס קולס חזרה על צורת גאומטריה מוארכת ייחודי. כנפיים לユニיט, כנפיים וẉרו מוזר של מControlItem והתקדים של עקומות ומישורים בשורטית תנורמות לעריכה של הקולוס תתרむ תודוביטית multi resolution editing יש להאבק ולהיות נוחות בינו לבין המשטחים.
The thesis is about the process of modeling and simulating the behavior of curves in a computer program.

The main focus is on the transformation controller, which is used to manipulate the curves. The transformation includes the ability to change the resolution of the curves, and it is done by manipulating the control points of the curves.

The thesis also discusses the use of a shared library extension for the program, which allows for easier manipulation of the curves.

In conclusion, the thesis presents a new method for simulating the behavior of curves in a computer program, which can be used in various applications, such as computer-aided design (CAD) and computer graphics (CG).

The thesis is written in Hebrew and contains an abstract in English.
B-splines, and their use in the development of modern software tools for creating curves and surfaces. Today, this approach allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces.

The use of B-splines in computer-aided design (CAD) and computer-aided manufacturing (CAM) allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces. Today, this approach allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces.

The use of B-splines in computer-aided design (CAD) and computer-aided manufacturing (CAM) allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces. Today, this approach allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces.

The use of B-splines in computer-aided design (CAD) and computer-aided manufacturing (CAM) allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces. Today, this approach allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces.

The use of B-splines in computer-aided design (CAD) and computer-aided manufacturing (CAM) allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces. Today, this approach allows for the creation of complex geometries, such as freeform curves and surfaces, within a single environment. The use of such software tools enables the user to create objects with high geometric accuracy and to manipulate them in real-time, using intuitive and powerful interfaces.
המחק נועש בהנחייה פורפ' גרשון אלבר בפקולטה למדעי המחשב.

תודה מוייחה לפורפ' גרשון אלבר על עזרתו ותמיכתו בתוכן המחק.

המחק מוייח לאמיר יבנגייך ניקולסקיק, רצייל, רימ בליג'ינ, על תמיכתם וסבלנותם.

אניך מודה לסכנינו על התמיכחה והכפיפה הנדרשת ב_deinitוניות.
תكنיק里斯ומית עבור גאומטריה ודימויים

הערכה על מחקר

לשם מילוי חלקי של הדרישות ל科学发展 התאור
מიיסטר למדעי במדעי המחשב

מאתון בליץ י"ע

הוגש לסנטה טכנולוגיה – מכון טכנולוגי לישראל
שבט החשמ"צ

הרפה ינואר 2010
.Marker ריווומי
עבורי
גאומטריה וوكالة מימית חוויתית

מאתה בלצ'יימ'