Computer-Aided Design and Applications
© 2009 CAD Solutions, LLC
http://www.cadanda.com

Sketch Based Design of 2D and 3D Freeform Geometry

Masha Nikolski[1] and Gershon Elber[2]

[1]Technion, Israel Institute of Technology, nikol@cs.technion.ac.il
[2]Technion, Israel Institute of Technology, gershon@cs.technion.ac.il

ABSTRACT

We present an interface for intuitive creation and modification of 2D and 3D freeform, spline based, geometry. The user interactively draws freehand shapes, either in the plane or in space over arbitrary freeform surface(s) and the system automatically reconstructs a plausible result, a new freeform curve or surface. Several intuitive direct modeling operations are offered that a casual or novice user is likely to find easy to use. As we consider Boolean operations counter-intuitive for novice users, a direct modeling scheme of complex freeform geometry of similar power is proposed as an alternative. Once a curve is sketched on a surface it can be further manipulated on the same surface. That is, scaled, translated, rotated, etc. Further, a sketched curve can be employed directly to create a hole (cut) in the geometry and/or to sweep a handle out of (into the) geometry. The entire framework allows the modeling of complex freeform geometry in an intuitive manner, all while avoiding Booleans.

Keywords: freeform curve, freeform surface, sketch, splines, intuitive modeling.
DOI: 10.3722/cadaps.2009.157-166

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 tools, but also very complex environments and are therefore often seen as a separate scientific discipline. Designers, typically engineers, take a long time learning the different modeling functions and how to handle them correctly. The idea of a novice or a casual user employing a CAD system is impractical using contemporary modeling environments. Moreover, contemporary CAD systems are, for the most part, designed to create precise representations of a new artifact, and fail to support the generation of raw sketches, which are often the only one created during the *conceptual stages* of design.

Because CAD systems are not very intuitive and do not efficiently support the early phases of design, a large body of research has gone into the developments of new conceptual sketching modeling tools. Perhaps the earliest computerized sketching system (in fact the earliest CAD system) was Sutherland's Sketchpad [14]. 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 drawing packages have been developed, but not many of them aimed at understanding the picture being drawn, detecting relationships not explicitly specified by the user, or connecting individual components and forming a larger context, as humans do when looking at a sketch. Moreover, even fewer of these systems support true freehand sketching, let alone freehand sketches of and over three-dimensional freeform objects[1].

This work aimed at offering a different approach to modeling complex geometry. The environment was designed in the hope that novice and/or casual users will find it easy and convenient to use. While most of the known CAD systems use Boolean operations to construct complex geometry from simpler one, we postulate that Boolean operations are a counter-intuitive modeling concept for non professional users. This powerful concept is drawing on set-theoretical ideas that not every novice geometer grasps. Moreover, the result of a Boolean application can be non-intuitive at times as, for example, parts of the input disappear in the output. In this work, we offer an alternative modeling scheme to Boolean operations that is of similar power, and yet, we believe, is more intuitive by allowing direct modeling and editing of the complex geometry.

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/~GuIrit for further information). This research results are implemented as two new shared library extensions to the GuIrit modeling environment, the curve sketching and the surface sketching extensions.

The remainder of this paper is organized as follows. Section 2 summarizes related work that has been conducted in this area. Sections 3 and 4 detail the geometric constructors used to create freeform curves and surfaces, respectfully. In Section 5, we show some examples of objects created with our new modeling environment. Finally, in Section 6, conclusions are drawn.

2. PREVIOUS WORK

In Section 2.1 we present the contemporary knowledge in creating 3D sketching tools; Section 2.2 surveys existing commercial products that offer such capabilities and in Section 2.3, we briefly depict the approach proposed by this work.

2.1 Existing Approaches

In the past, quite a few sketch-based modeling systems were developed, which interpret the user's freeform strokes and interactively construct 3D geometry. Most of those systems create polygonal models, while systems for processing of sketches depicting freeform geometry are less common. Our goal is to develop such an interface for freeform shapes that can be used to create complex (freeform) models with ease and yet does provide some control over the precision of the result.

A typical approach to modeling new geometry is to start with a simple primitive such as a sphere or a cube and gradually construct a more complex geometry using CSG operators. Even in sketched based design this approach is employed and one example is the ShapeShop sketch-based modeling system. It uses hierarchical implicit volume models (blob- trees) as an underlying shape representation and supports interactive creation of solid models using the standard CSG operators. In contrast, and as stated already, we foresee direct manipulation of geometry as a more intuitive editing scheme, compared to Boolean operations.

Another approach is the surface inflation technique, where a polygonal mesh is extruded from a 2D drawing of the models' silhouette. Igarashi [Igar99] suggested a system called Teddy, which allowed modeling of primitive freeform surfaces with a very simple interface. The procedure requires of drawing the object's silhouette, and then the application will provide a polygonal mesh adapted to it. Similar result was introduced by Karpenko [11] in his SmoothSketch system. While this approach is very intuitive to use, it is not general and can be used to construct only a primitive set of rounded and non-self-occluding models. Furthermore, the silhouette based modeling scheme is imprecise and so far is not adapted to freeform representations (while possible).

[1] While in some contexts freeform geometry might also denote smooth polygonal meshes, in this work we only refer to spline based geometry as freeform.

Some modeling systems use freeform strokes to create a gestured interface and others for specifying 2D and 3D curves. The SKETCH application is an interesting attempt to create an environment for rapidly conceptualizing and editing approximate 3D scenes using simple non-photorealistic rendering. A purely gestured interface based on simplified line drawings of primitives is offered that allows all operations to be specified within the 3D world. Eggli et al [5] proposed a 2D/3D modeling tool for pen based computers. Users of this system define a model by simple pen strokes, drawn on a screen based PC. The system can also be used to sketch 3D solid objects and B-spline surfaces. Herein, we propose an extended interface for curves' sketching and manipulation not only on a plane, but on any general (set of) surface(s).

Yet another method for creation new freeform models is using various deformations and shape manipulation techniques on an existing geometry. One example is Han [Han06] that presented an approach which adopted a simple 3D sketching technique and a finite element deformation method to create freeform models. In the proposed method the user applies interactive spline sculpting to modify a surface in a predictable way.

2.2. Existing Commercial Products

Numerous 3D modeling packages and kernels are available today on the market. Here we survey three sketching tools in popular systems for the naïve and the professional user.

SketchUp (http://sketchup.google.com/): Google 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 three a dimensional form. The last 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 the handles attached to it.

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 curve endpoints. SolidWorks can also build 3D curves using what it calls a 3dsketch. This type of sketch allows defining 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 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 guide curves tangent or C^2 continuous with adjacent shapes.

Rhino (http://www.rhino3d.com/): Rhino allows the creating of 2D/3D curves through selecting 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", in order to be used in further operations. (The command reduces the number of control points.) 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).

The above reveals that modern modeling packages continue to lack sketching abilities. Complex geometry is still built via Boolean operations, 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.

2.3 Current Approach

This work introduces a system for intuitive geometric modeling of freeform complex models. It permits both a novice user and a professional one to easily create complex geometry, with precision if so desired. The requirements for a large geometric modeling experience and/or the tedious learning experience of the systems' functions and capabilities are greatly alleviated. In this modeling environment that employs no Boolean operations, direct interactive manipulation provides an easy to use interface and the resulting geometry is a set of precise B-spline curves or surfaces, which can be further edited with simple and intuitive gestures.

3. CURVE CREATION AND EDITTING

In this section, we examine ways of creating planar and spatial curves from sketches of input devices (i.e. a mouse). Aiming a 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. [1], [7]). Herein, we portray the representation we chose, a representation that will allow us further manipulation of these curves. In Section 3.1, we discuss this selected representation. Section 3.2 presents an alternative way of creating spatial curves over surfaces and Section 3.3 introduces a manipulation scheme of curves on the surfaces, once created.

3.1 The Representation of Curves

The sketched data is converted to a freeform B-spline curve with (control) points in R^5 as (x, y, z, u, v) tuples. The (x, y, z) triplet is the Euclidean represents of the point whereas the (u, v) pair provides the parametric representation of the point in the underneath surface. When a curve is sketched on parametric surfaces $S(u, v) = (S_x(u, v), S_y(u, v), S_z(u, v))$, 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 $S(C(t))$ [Elbe92] 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.

The curve drawing operation is not confined to one underneath surface. A single continuous stroke can lie on multiple surfaces, simultaneously and can circumvent all directions, by allowing the user to interleave the sketching process with arbitrary transformations of the scene. Figure 1 demonstrates such a result. Since automatic surface identification is performed during the sketching process, for each point acquisition, a change in the underneath surface can be noted. Each stroked point should now carry a reference to its underneath surface as well as its (u, v) coordinates in that surface.

3.2 Projection of Curves

The creation of general curves on surfaces is inherently imprecise. While direct sketching of curves has its obvious advantages, at times, a more precisely controlled shape 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 curves could be projected on arbitrary surface(s) along a prescribed projection direction, V. Conceptually, from each point on a spatial curve

C, a ray is fired in the V direction and the first hit, if any, of V with the target surface S is recorded. Clearly V can miss the surface S altogether or hit S twice. Moreover, two adjacent locations on C can become discontinuous, when projected onto S. All this ill conditioned cases must be detected by carefully analyzing the intersection locations of V and S and treated.

This projection operation could be seen as a surface-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 very 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 1 also shows an example of projecting a group of curves onto a freeform model.

hello
hello

Fig. 1: The left image shows a single curve sketched over several surfaces all around by interleaving the sketching process with scene's transformations. The right image presents a precise projection of a group of curves over a cylinder.

3.4 Direct Editing of Curve on the Surface

Once a curve is created on the surface, one might be interested in further editing and manipulating it. In [7], multi-resolution editing schemes where adapted to allow direct manipulation of curves over surfaces. Herein, we propose to similarly adapt the notion of direct editing curves over freeform surfaces but for 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 transformations of curves that are embedded in freeform surfaces.

Consider 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, one can apply T, a transformation to rotate, translate, and (uniformly or non-uniformly) scale $(u(t), v(t))$, as $T(u(t), v(t))$. Then, $S(T(u(t), v(t)))$ 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 one is required to approximate 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 (possibly 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 T 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 T, a *transformation control frame*, is attached to curve C once selected for transformation over S. See Figure 2. 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-) select the rotation, translation and scaling of C. Because the underneath representation of the frame and C is the same, the frame and C can be transformed together as T is applied in the parametric domain of S, creating an intuitive and efficient transformation interface for transforming curves over surfaces.

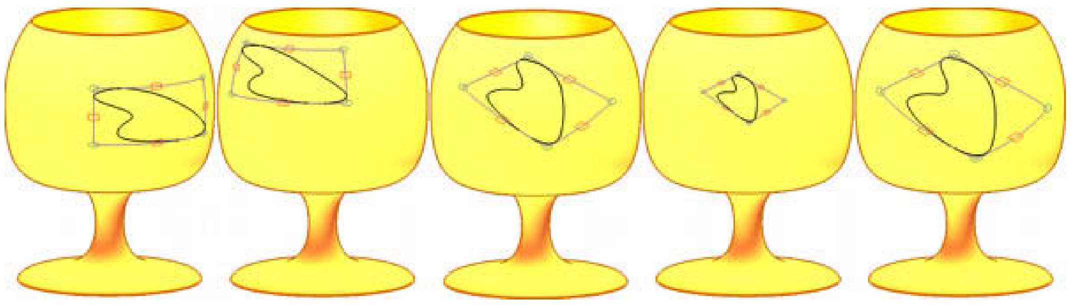

Fig. 2: A curve C (in black) on surface S is selected for transformation over surface S, in yellow. A *transformation control frame* is presented to the user around the curve on S with handles for rotations (green circles at corners) and translations (red squares on edges) to select the desired transformation. Several transformations are presented. Scale is provided also via the green circles with the combination of a Shift key.

Having efficient and effective interface to create and manipulate curves over surfaces, it can now be employed toward the creation of new freeform surfaces as well as manipulating existing surfaces. The next section presents these capabilities.

4. SURFACE CREATION AND EDITTING

Having created basic curves, in the plane and in space, over surfaces, we are ready to further employ the curves to construct surfaces. Section 4.1 discusses a simple cutting operation which allows cutting away holes or parts off a surface and Section 4.2 presents an extrusion and/or sweeping option.

4.1 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 defined 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. All surface pieces are presented to the user as new entities and the user can select to purge 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 might be allowed in the closed and/or boundary starting/ending curve, the user must extend the curves to bridge these gap(s) before a cutting operation can be commenced successfully.

4.2 Sweeping/Extruding Surfaces Out of Surfaces

More interesting is the fact that curves over surfaces could also be intuitively exploited toward the creation of new surfaces. Consider a constructor of an extrusion surface, E, that accepts a curve, C, and a direction vector, V, and creates a new surface by extruding C along V as $E(t, r) = C(t) + Vr$. C can clearly be a space curve. Recall the circular cross section we projected onto a cylinder in Section 3.2. This projected curve can now be extruded into space along V to create the desired new cylindrical protrusion. In other words, we have completed the creation of a precise union of two cylinders without resorting to Boolean operations. Only direct manipulation with immediate interactive feedback is employed.

In a similar manner, a sweep surface could be forged out a curve on a surface. The sweep constructor is very powerful, and, in essence, is already containing the special case of extrusion. We pay some special attention to the creation of a general sweep constructor that is also intuitive and easy to directly use. Given a curve, *C*, on some surface, *S*, the user needs to prescribe another initial axis curve, *A*, along which *C* is to be swept. In each control points of *A*, an orientation frame is used to place one section of *C*. While it is well accepted that the Frenet frame [4] is not the method of choice for orientation computations due to its instability near inflection points, there are numerous methods to derive orientation frames that one can use [2] and any of them can be employed in this work. Figure 3 presents one example of a sweep formed out of a curve sketched over a surface.

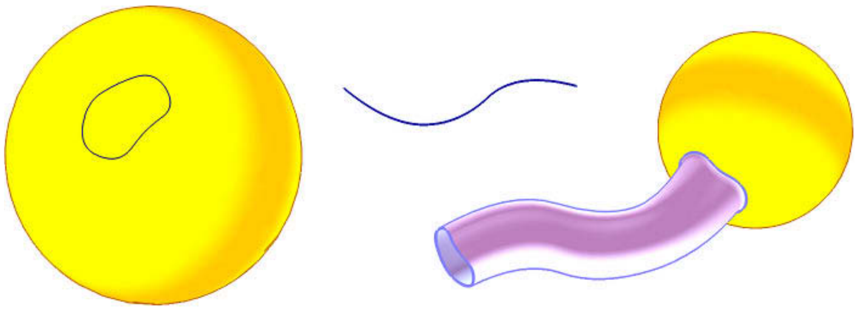

Fig. 3: A sweep surface construction using a curve drawn on a surface *C*, in yellow, and an axis curve, *A*, in black. The initial constructed sweep surface is presented on the right.

The constructed sweep surface will be C^0 connected to the original surface, *S*, at *C*. Because in many cases G^1 continuity might be desired, the following process can be used, near $C(t) = (u(t), v(t))$. Let $N(u,v) = \frac{\partial S}{\partial u} \times \frac{\partial S}{\partial v}$ 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* of

$$F(u(t), v(t)) = \frac{d(u(t), v(t))}{dt} \times N(u(t), v(t)). \tag{1}$$

$F(u, v)$ is in the tangent space of *S* for all location 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

$$D(u,v) = C(u,v) + F(u,v), \tag{2}$$

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 G^1 at *C*. Several issues must be noted. Equation (1) could be computed analytically as it is polynomial (See [12]) in which case the magnitude of *N* and *C'* and hence *F* must be (approximately) normalized. See [8] for a possible approximate normalization method of vector fields. Alternatively, *D* can be approximated by sampling points along *C* while evaluating Equations (1) and (2) locally.

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 $-F(u,v)$. Additional Control is also offered on the amount of offset used (the magnitude of normalized field of

F(*u,v*)), controlling the side of the created rounding that achieves G^1 continuity over the surfaces at *C*. See Figure 4 for a few examples, achieving a similar result to blending that is common when smooth connection between two surfaces is desired.

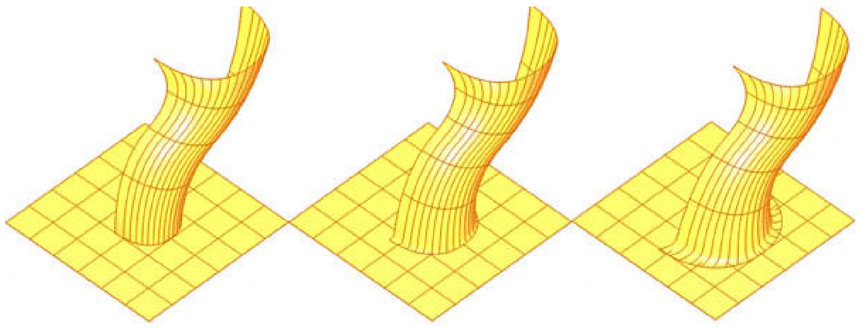

Fig. 4: Several examples of stitching the constructed surface at *C* with G^1 continuity.

Finally, it is not always required to stitch *S* and the constructed a sweep at *C*. In fact, it can happen that *S* is only an auxiliary surface aiding to create a space curve *C* and the sweep surface from *C* stands on its own right. To facilitate this, 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.

Once the initial sweep surface construction is complete, the created surface is presented to the user in an editable mode. Along with the surface, *transformation controllers* for section curves are presented (See Figure 5). 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 translate, rotate, or scale a section curve with a presented transformation controller or he/she can completely replace the selected curve with a whole different curve (see Figure 6). It is also possible to refine the created surface in regions of interest and introduce additional section curves (and transformation controllers) into the surface. See also Figure 5.

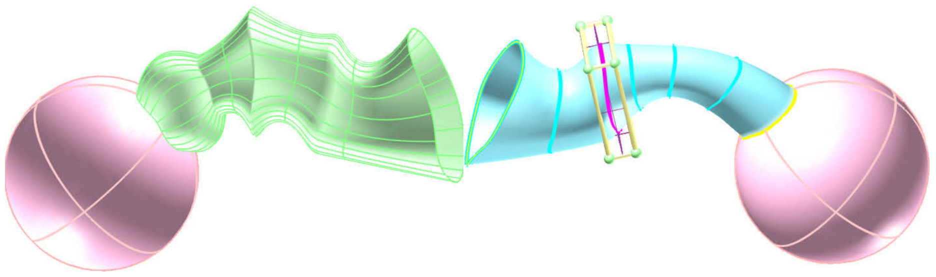

Fig. 5: Each section of the initial sweep in Figure 3 is amendable to further manipulation with the aid of a *3D transformation controller* shown on one section on the right figure. The figure on the left shows one possible outcome, also after additional sections were added to refine the shape.

5. RESULTS

This section presents four additional examples of somewhat more complex objects, created using the introduced tools. Figure 7a presents a model of a teapot; Figure 7b presents a model of a cactus; Figure 7c shows a model of a clock and Figure 7d – a model of a telephone.

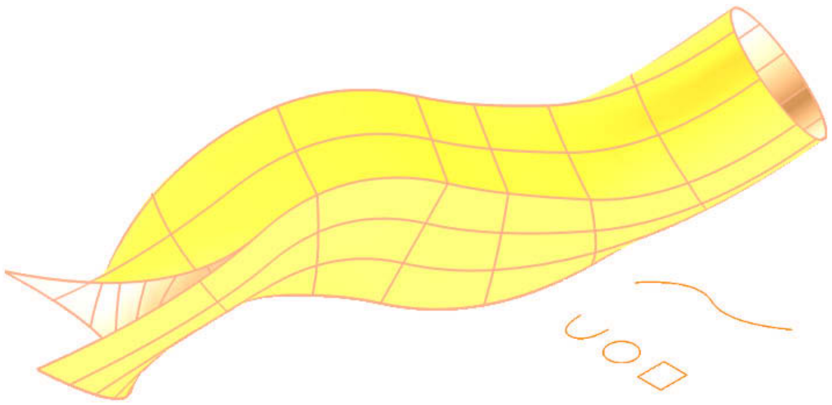

Fig. 6: Each section of the initial sweep could be replaced with a whole new cross section curve. In this example, three section curves are used for a sweep along the green axis curve.

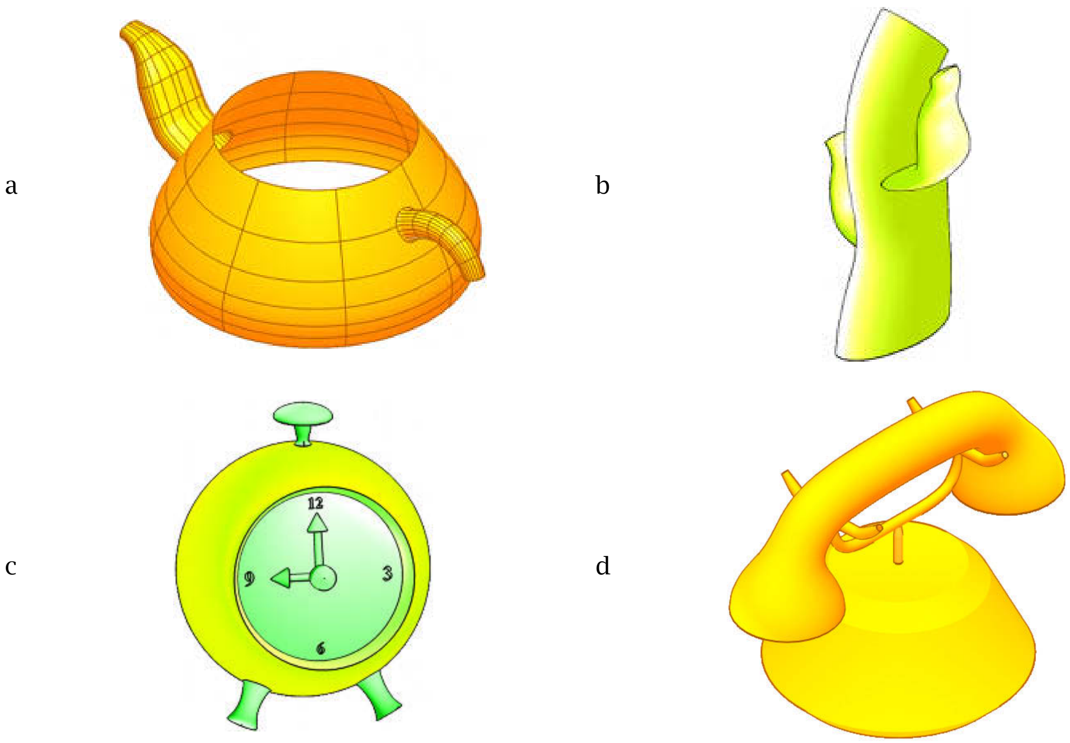

Fig. 7: (a) A model of a teapot created by sketching two closed curves over the body and sweeping the spout and the handle out. The spout also underwent some editing, narrowing it down, (b) A cactus created by three sweep surfaces, (c) A model of a clock using sweep surfaces, projection and trimming options, (d) A phone created from sweep surfaces with additional editing.

6. CONCLUTIONS

In our research, 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 desired precision.

Not every freeform model could be realized using the presented tools, certainly not with ease. Figure 7 presents an interesting topological problem where it might be desired to glue the handle back into the body, affecting the Genus of the model. Further, 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. Other tools that can ease the aggregation process include 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).

Beyond aggregation, another modeling capability that must be provided as part of a complete solution is probably the 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. More control over the editing of individual cross section should be added so arbitrary cross sections could be formed. Complex section curves could also be editing and created by (multi-resolution) editing as in [7], an option that should be seen part of a complete solution. Moreover, possible direct multi-resolution editing [13] and intuitive deformation tools of freeform surfaces should probably be made available as well.

Finally, one can also consider adding more intuitive manipulation and deformation tools to further ease the editing process. These simple deformation tools will be then converted to general deformations internally. For example, bending, twisting and deforming tools (See [3] are highly intuitive to novice users and the outcome is also easy to anticipate.

7. REFERENCES

[1] Banks, M.; Cohen, E.: Real Time Spline Curves from Interactively Sketched Data, Symposium on Interactive 3D Graphics, 1990, 99-107.

[2] Cohen, E.; Riesenfeld, R. F.; Elber, G.: Geometric Modeling with Splines: An Introduction, AK Peters, 2001.

[3] Conner, D. B.; Snibbe, S. S.; Herndon, K. P.; Robbins, D. C.; Zeleznik, R. C.; Van Dam, A.: Three dimensional widgets, proceedings of the 1992 symposium on Interactive 3D graphics, Cambridge, Massachusetts, United States, 1992, 183 – 188.

[4] Do Carmo, M.: Differential Geometry of Curves and Surfaces, Prentice-Hall, 1976.

[5] Eggli, L.; Hsu, C. Y.; Bruderlin, B. D.; Elber, G.: Inferring 3D Models from Freehand Sketches and Constraints, Computer-Aided Design, 29(2), 1997, 101-112.

[6] Elber, G.: Free Form Surface Analysis using a Hybrid of Symbolic and Numeric Computation, Ph.D. Thesis, University of Utah, Computer Science Department, 1992.

[7] Elber, G.: Multiresolution Curve Editing with Linear Constraints, The Journal of Computing & Information Science in Engineering, 1(4), 2001, 347-355.

[8] Elber, G.: Symbolic and Numeric Computation in Curve Interrogation, Computer Graphics forum, 14(1), 1995, 25-34.

[9] Han, L.; Amicis, R.; Conti, G.: Interactive Spline-Driven Deformation for Free-Form Surface Styling, Symposium on Solid and Physical Modeling, 2006, 139-147.

[10] Igarashi, T.; Matsuoka, S.; Tanaka, H.: Teddy: A Sketching Interface for 3D Freeform Design, SIGGRAPH, 1999, 409-416.

[11] Karpenko, O.; Hughes, J.; Raskar, R.: SmoothSketch: 3D Free-Form Shapes from Complex Sketches, ACM Trans. Graph, 25(3), 2006, 589-598.

[12] Kim, K.; Elber, G.: New Approaches to Freeform Surface Fillets, The Journal of Visualization and Computer Animation, 8(2), 1997, 69-80.

[13] Kazinnik, R.; Elber, G.: Orthogonal Decomposition of Non-Uniform Bspline Spaces using Wavelets, Computer Graphics Forum, 16(3), 1997, 27-38.

[14] Sutherland, I.: A Man-Mashine Graphical Communication System, Ph.D. Thesis, MIT, Boston, Massachusetts, 1963.