Interacting with Computers 21 (2009) 201–211

Contents lists available at ScienceDirect

Interacting with Computers

journal homepage: www.elsevier.com/locate/intcom

A survey of sketch-based 3-D modeling techniques

Matthew T. Cook *, Arvin Agah

Department of Electrical Engineering and Computer Science, The University of Kansas, 1520 West 15th Street, Lawrence, KS 66045-7621, United States

ARTICLE INFO

Article history:
Received 1 October 2008
Received in revised form 28 April 2009
Accepted 7 May 2009
Available online 19 May 2009

Keywords:
Human computer interaction
User interface
3-D modeling
Interaction styles
Sketch-based modeling

ABSTRACT

As 3-D modeling applications transition from engineering environments into the hands of artists, designers, and the consumer market, there is an increasing demand for more intuitive interfaces. In response, 3-D modeling and interface design communities have begun to develop systems based on traditional artistic techniques, particularly sketching. Collectively this growing field of research has come to be known as *sketch-based modeling*, however the name belies a diversity of promising techniques and unique approaches. This paper presents a survey of current research in sketch-based modeling, including a basic introduction to the topic, the challenges of sketch-based input, and an examination of a number of popular approaches, including representative examples and a general analysis of the benefits and challenges inherent to each.

1. Introduction

Once the sole realm of architects and animators, 3-D computer modeling is swiftly becoming a common tool for the modern designer, artist, researcher, budding filmmaker, and even weekend hobbyist. New, more affordable and user friendly modeling applications are now filtering down from the professional markets and are poised to proliferate the general consumer market like photo and video editing software before them.

However, despite the growing demand for modeling software, both professional designers and laypersons are often put off by the complexity, difficulty, and unintuitive nature of the current modeling interfaces. Based on the underlying structure of a model or its mathematical foundations, techniques like subdivision surfaces and control point manipulation may offer the designer fine-grained control, but can require hours of tedious work to create even simple forms. No matter what level of realism a new package can bring to a final model, newcomers see little reason to invest such effort to model an object that could be sketched out on a cocktail napkin in mere minutes.

Responding to these concerns, over the past decade the 3-D modeling and user interface design communities have begun to address these issues. Rather than the fine-grained controls of current systems, many researchers are now focusing on more intuitive, simplified modeling techniques targeted at the early, preparatory stages of the design process. For the traditional artist, these phases of early design are characterized by sketching techniques, and so many researchers have based their new interface methods on these basic artistic forms, hence the name *sketch-based modeling*.

Although targeting a common goal, the field of sketch-based modeling encompasses a wide variety of techniques and approaches, and although none has, as of yet, emerged as a clear direction for the future, each offers its own unique benefits. In this paper, we hope to offer a basic survey of some of the most popular and promising approaches in the field. We will begin with a few opening remarks about traditional sketching and its use as a modeling interface. Next, we will briefly touch upon the unique input challenges of these interfaces. Following this, we review seven specific techniques through representative examples in the literature. We conclude the paper with a brief discussion, and closing remarks.

2. The practice of sketching

In common parlance the term "sketching" can refer to a fairly broad range of activities, so let us begin by formally defining what we mean by sketching. For our purposes, sketching is a form of drawing. Sketching is generally done as preparation, planning, and idea generation, for another activity, be it an engineer's scrawled diagrams or the underdrawing of an oil painter. Because a sketch can be made quickly and simply with the tools at hand, sketching is a powerful tool to help make abstract ideas and nascent designs more concrete. For this same reason, sketches are characteristically "sketchy" in appearance; rough, messy and disorganized. These qualities both reflect the creative mental process a sketch represents, and drive that thinking process from a vague notion to a developed idea.

* Corresponding author. Tel.: +1 785 864 8821; fax: +1 785 864 3326.
E-mail address: mcook@xn--m3h.net (M.T. Cook).

0953-5438/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.intcom.2009.05.004

It is important to emphasize the close connection between sketching and the mental process that underlies it because it is this connection that represents sketching's primary utility. We can think of traditional paper and pencil sketching as composed of three elements. The first is the mental component we call *feedback* (Do and Gross, 1996). As a sketch is drawn, the artist continually sees the results of each stroke of the pencil, and reinterprets the visual image on the page, comparing it with his or her mental concept. While the artist can make corrections to the sketch to bring it closer to his or her mental image, he or she can also use those differences to update the mental image, trying new concepts or fleshing out areas that were not yet concrete.

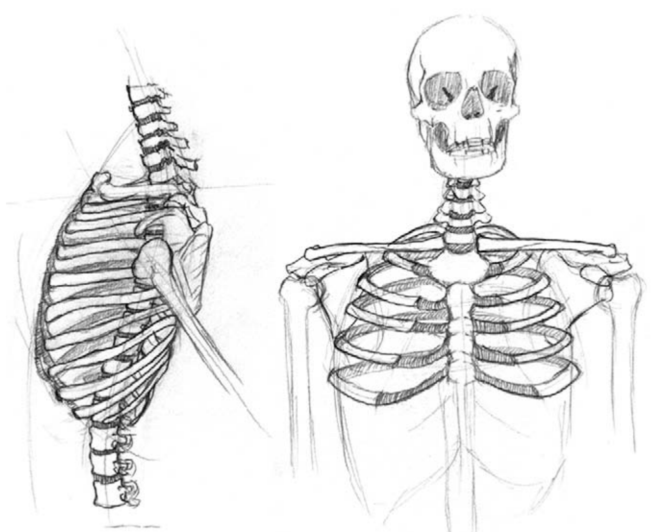

Fig. 1. In this typical sketch we can see how the artist uses faint broad strokes to feel out the general shapes of the subject. The image is then slowly refined by overdrawing those original stokes, each time adding additional detail and clarity of form. Note also that, although continuos contours are suggested, most forms are defined by the totality of many lines, rather than any single contour curve.

The second element of sketching is the physical technique of the artist, which we call *overdrawing*. Overdrawing goes by many names in the literature including oversketching (Zeleznik et al., 1996), re-sketching, re-drawing, overtracing (Do and Gross, 1996), scribbling, and 'nervous' hand (Henzen et al., 2005). In this process the artist gradually adds new marks over previously drawn lines, building up and emphasizing some elements of a sketch while de-emphasizing others – see Fig. 1. It is overdrawing that gives sketches their characteristic sketchy appearance, and allows the artist to change the drawing just as feedback changes his or her mental image.

The third and final component of sketching we call *incremental refinement*, and describes how feedback and overdrawing work together over time to develop an idea (Michalik et al., 2002; Do and Gross, 1996). Incremental refinement can be summed up the maxim: "*work from the macroscopic to the microscopic*". As the sketch progresses the artist begins with simple shapes and broad ideas which are then refined through experimentation and exploration into more concrete and detailed descriptions.

Sketching is utilized by a wide variety of fields, but is probably most associated with the fine and professional arts. Painters, illustrators, sculptors and designers are taught early on to use sketching at all stages of their work – see Fig. 2. Artists often begin a new project with a series of thumbnail sketches. Composed of basic lines and shapes these help to flesh out in the broadest sense the elements and arrangement of a composition. These are followed by refined studies and more detailed sketches. Involving not only line but shading and tone, these secondary sketches explore the potential of the composition and will later sever as a framework or plan for a finished work of art. For commercial artists, sketches also allow the artist and client to discussed and refine a commission before its final construction begins.

These same techniques are also used in other visually or spatially oriented fields like industrial design and architecture – see Fig. 3. In fact, Herbert (1993) goes so far as to say that sketches are "the designer's principle means of thinking". The ways in which each of these professions utilizes sketching differs slightly owing to the particular requirements of the field. An interior designer for example

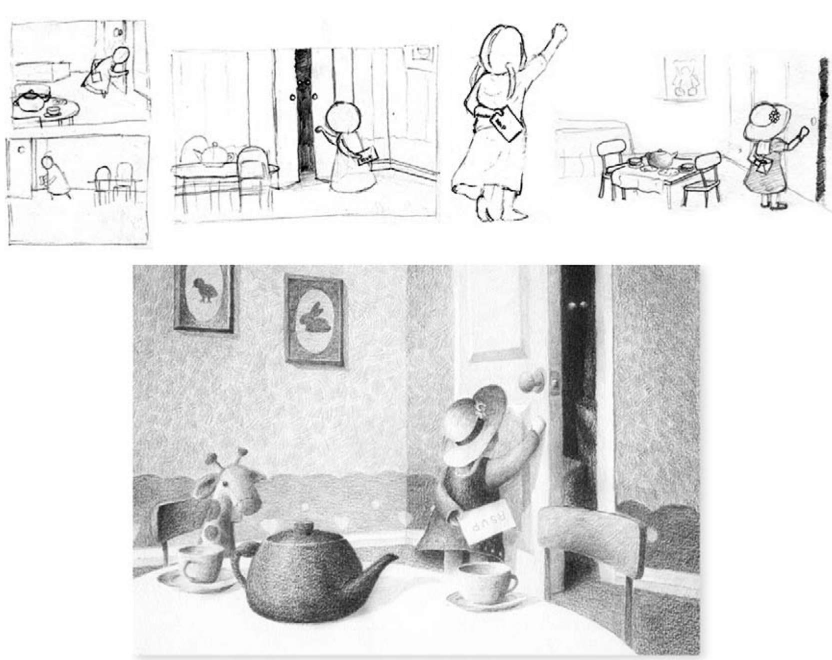

Fig. 2. Here we can see the progression of an idea from basic thumbnail sketches at the top left, through refined sketches, subject studies, and compositional experimentation, to a final illustration.

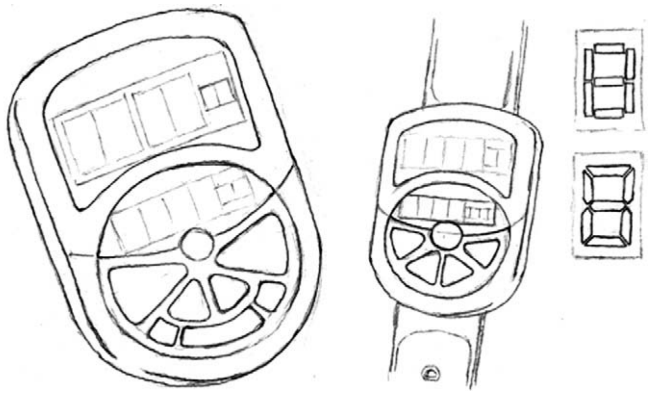

Fig. 3. Industrial designer's sketches of a wrist watch concept.

might compose sketches as floorpan views, making heavy use of symbols and text to indicate the location of furnishings or utilities, while an architect may be more interested in 3-D shapes and how they sit together in an environment (Do, 2005). Despite the diversity of techniques, several observational studies have revealed that there are common patterns and methods to sketching used by professionals in different fields (Garner, 2001; Do, 2005; Huot et al., 2003; Igarashi et al., 1997). These studies show the use of common techniques like abstraction and analogy, and suggest that the sketching activity can be divided into a number of related and overlapping stages, each characterized by the kinds of strokes the artist makes, and the role those strokes play in the final sketch.

Similar practices also carry over into other professions, especially those in which there is a focus on abstract or complex ideas that are difficult to visualize. Engineers, physicists, teachers, and scientist of all descriptions make widespread use of sketching to help develop and communicate their ideas. The sketching techniques of professionals in all of these fields are simply more formal incarnations of the thinking processes innately familiar to everyone.

3. Why sketch-based modeling?

The next logical question is, why sketching as the basis for a new modeling method? The traditional practice of sketching makes an attractive target for two reasons. First, both the physical activity and mental processes associated with sketching form the basis for problem solving, development, and general creative thought for most of the professions with a vested interest in 3-D modeling. Second, sketching is most utilized at early stages of the creative process when designs are vague and details scarce—precisely the stage of 3-D modeling that is underserved by the current market.

However, despite the importance of sketching as a traditional technique, it is important to consider sketching's suitability to the modeling task. After all, traditional sketching is used to produce 2-D projections, not 3-D models. Can sketching skills even translate to this new medium?

In this respect, three aspects of sketching are important to note. First, although the result of a traditional sketch is indeed a 2-D depiction, the artist's mental picture is necessarily more detailed. It is the artist's close attention to the spatial relationships of their subject that allow 2-D depictions to visually simulate a 3-D object.

Second, under some circumstances, sketching may offer advantages to working directly in 3-D. Despite the spatial reasoning abilities of the brain, working directly in 3-D turns out to be more difficult for most than working through a well designed intermediate projection (Durand, 2002).

Third, and perhaps most importantly, although computers may be able to represent 3-D objects, commodity hardware limits our interactions with those objects to 2-D input and output devices. Thus in many ways, working with the 3-D modeling environment is not unlike a traditional sketching process in which 3-D subjects are translated and rendered in 2-dimensions. The computer simply offers an opportunity to make that translation process a more dynamic one.

4. Sketch input

In order to allow artists and designers to interact with a sketch-based modeling program, the application needs to provide a comfortable and intuitive means for the user to 'draw' into the system. On a hardware level, artists generally find mice at least distasteful and at most deplorable as drawing implements. Derry (1996), for example, describes drawing with a mouse as something akin to drawing with a bar of soap. Instead, most systems now offer support for—or require the use of—digitizing tablet devices. The user draws strokes onto the tablet's surface with a pen-like digitizing stylus, and the pen's position on the tablet, as well as angle, pressure, and proximity to the surface, are detected by the tablet and relayed to the system.

From a software perspective, drawing involves two related activities. The first, and most straightforward for the system, is line creation. Because most modeling systems are based on vector graphic representations, this can be easily achieved with any number of appropriate curve or spine fitting algorithms. More challenging for the system is the need to replicate overdrawing, the continuous process of correction and revision that is a hallmark of the traditional sketching process. This can be particularly difficult because the artist's actual intention behind a collection of overdrawn strokes—the so-called 'line hypothesis' (Henzen et al., 2005)—is only visually suggested by the totality of the existing lines, but is not necessarily coincident with any one of them – see Fig. 1.

To address this functionality, researchers have investigated a number of line editing techniques. The most straightforward approach to this issue is a so-called mark-based editing system, such as the one described by Thomas (1994). Baudel's system allows a user to make corrections and alterations to existing spline curves by overdrawing a portion of the original curve. The user's overdrawn stroke is then spliced and blended into the original, replacing a portion of the original with an updated path.[1]

The mark-based interaction paradigm is simple and straightforward, but not without its drawbacks. The system must distinguish between drawing and editing strokes, either through explicit modes or an interpretive system. A further issue arises as the system attempts to smoothly combine strokes. Because the strokes were not generated by a single input, there is no single smooth path that is likely to interpolate or even closely approximate the combination of multiple segments. Where corners or other discontinuous features are desired this arrangement suffices, but in most cases some degree of smoothing and blending will need to be applied. The biggest drawback to this method is the fact that corrections replace rather than augment segments of the curve. This is in contrast to a traditional sketch in which corrective strokes pile up, one on another, creating a visual record of alterations, and fodder for the feedback process.

This last point has not gone unnoticed, and a number of systems have attempted to more closely mimic this aspect of sketching. A system by Fiore and Reeth (2002), for example, uses additional

[1] Baudel's approach has proven popular, and similar systems appear in a number of later works including Michalik et al. (2002), Pereira et al. (2003), and Karpenko et al. (2002). In addition, Adobe also appears to be using a system at least similar to Baudel's in action in current versions of its Illustrator vector graphics art application (Alex, 2005).

strokes as attractors that pull the path of a curve, but don't replace the curve segments all together. Similarly, Fleisch et al. (2004), rather than replacing the segment outright, provide a user adjustable parameter to weight the effect of the new curve, allowing the edit to act as either a strict replacement or a partial attractor depending on the user's preference. Fleisch's work also explores the sketching and adjustment of 3-D space curves with 3-D input devices. Alex (2005) incorporates similar adjustment features into a contextual widget, allowing users to make such adjustments within the context of the drawing.

Perhaps the most similar to traditional sketching is a system presented by Henzen et al. (2005). Dubbed 'nervous hand', each stroke entered by the user is painted to the screen and slowly begins to fade over time. As strokes are built up along an existing line, the line is drawn towards the most saturated, acting like an attractor system. This system is unique in that it continues to display the user's editing strokes, providing a visual similar to the physical version of overdrawing.

5. Sketch-based modeling methods

As a field, sketch-based modeling is still in its infancy, and as a result there is little consensus on which interface paradigms are most effective, or for that matter, what constitutes an effective interface to begin with. Instead, there is a flourishing ecosystem of diverse approaches to the problem, each with unique strengths and weaknesses. As a modeling method, sketch-based modeling is primarily concerned with how to take drawing input from the user and convert that input into a model. Therefore, in the following sections we present seven general classifications that focus primarily on how a system interprets a user's input, and how that input in converted into model construction and alteration.

An early approach to the sketch-based modeling interface was to use drawing input as symbolic instructions, an approach we call *gesture created primitives*, and discussed in Section 5.1. This method allows a designer quick and intuitive access to the multitude of commands in a modeling interface, and was well suited to the limitations of early hardware. As technology has progressed, so too has the utility of this approach, often involving the interpretation of more complicated input, or working in tandem with other interface methods. Extending the gesture metaphor to its limit logically leads to a system that can interpret a user's drawing directly, a method we classify as *reconstruction*, and discussed in Section 5.2. Ideally reconstruction more closely resembles the experience of sketching on paper, but in practice its complexity presents a serious challenge. Straddling the fence between these two extremes is a class of solutions we call *blobby inflation*, which accept generalized drawing input from the user, but apply the same interpretation to any input, creating a more narrow class of models with distinctive "blobby" features. While this may seem counterintuitive, this approach has proven to be a particularly effective interface. Blobby inflation is discussed in Section 5.5.

While gestures and reconstruction focus on line drawings as input, there are a number of methods which instead focus on other artistic metaphors. For example, traditional artists use shading and tone to give a 2-D drawing the appearance of 3-D volume. Some interface paradigms attempt to interpret this and related information to infer volume from a shaded or tonal drawing. We classify these approaches as *height-fields and shape from shading*, and discussed them in Section 5.3. Another common tact is to approach 3-D modeling from the perspective of sculpture in which virtual tools are used to build up a model like clay, or cut it down with tools like a sculptor, methods we classify as *deformation and sculpture*, and discussed in Section 5.4.

Another approach that has seen a good deal of exploration considers how exactly 2-D drawing input should be interpreted in a 3-D environment, an approach we classify as *contour curves and drawing surfaces*, and discussed in Section 5.6. This includes both drawing onto 3-D surfaces, as well as constructing truly 3-D space curves. However, where as these approaches tend to focus on wire frame models and contour drawings, similar interface concepts can be extended to create 3-D surfaces and volumes from user-drawn 2- and 3-D strokes, a technique we classify as *stroke based constructions*, and discussed in Section 5.7. Like blobby inflation, these interfaces also fall in the continuum between pure interpretation and static gesture recognition, providing the user with a wider range of modeling methods than blobby inflation, while treating input strokes like functional parameters to be followed rather than data to be interpreted.

In the following sections we present a detailed discussion of each of these seven classifications, including a variety of examples. It should be noted that the lines between many of these classifications can, at times, be fuzzy, as many researchers have combined aspects from one or more categories into a single interface. We have tried in each section to focus on those aspects of each project that tie them together while still presenting a representative range of examples.

5.1. Gesture created primitives

Although drawing represents the user's input to the system, the crux of a sketch-based modeling interface is how best to convert this drawing input into a 3-D model. Because many 3-D models can be generally described by a combination of basic shapes—cubes, cylinders, planes, etc.—an early approach to this issue was to interpret the user's drawing input as gestures symbolic of these basic forms.

The classic example of this type of sketch-based modeling is provided by Zeleznik et al. (1996) SKETCH application. In SKETCH, new geometry is created though sequences of strokes that combine into a gesture defining both the shape to be created, and details of its form. So, for example, the user might draw three perpendicular lines, which would then be interpreted to define a box with sides matching the lengths of each stroke. Similar methods allow the user to create a variety of basic shapes very quickly, and to edit and position the forms once created.

Zeleznik et al.'s original system provided only a small set of basic shapes, allowing the gesture set to remain fairly small, and the gestures themselves iconic of the shapes they represent. A number of other researches have attempted to build on the success of the SKETCH system, adding additional functionality. However as the number of gestures increases, the ability to recognize and differentiate them becomes more difficult.

A popular solution to this issue has been the use of an interactive contextual disambiguation system (Eggli et al., 1995), or 'expectation list' (Pereira et al., 2000). A good example of such a system is the GIDeS modeling prototype developed by Pereira et al. (2004). GIDeS uses expectation lists to negotiate the meaning of a user's input interactively as part of the modeling process. When the user enters a gesture, a small contextual window appears presenting icons describing possible interpretations of the gesture. The user can then select the appropriate interpretation, or correct their input if it has been misinterpreted. By allowing the user to help differentiate the meanings of ambiguous gestures the interface can accommodate a greater number of similar gestures that better approximate traditional drawing.

One drawback to expectation lists is that they tend to descritize the modeling process into a call and response pattern of presenting a gesture and then approving a recognition. 'Suggestive interfaces' or 'mediated gesture system' (Igarashi and Hughes, 2001) provide a more conversational alternative. Chateau, designed by Igarashi and Hughes, allows users to generate simple, planer models from a

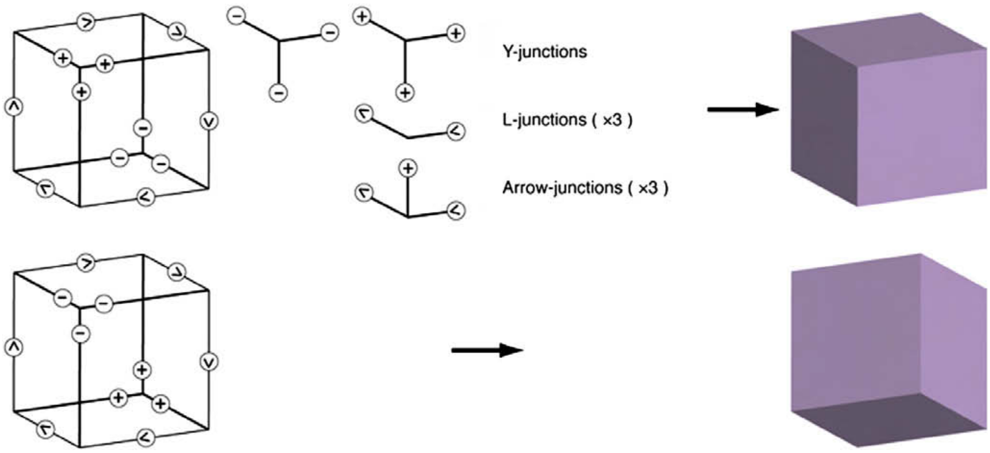

Fig. 4. Huffman–Clowes (1971) labeled line drawings: A (+) indicates a convex edge, (−) a concave edge, and (>) an occluding edge with the surface to the right. To the right of each figure is the associated 3-D reconstruction. Note that for this simple line drawing of a cube there exists two equally valid labelings corresponding to the two possible 3-D interpretations. This is known as the Necker Cube Illusion.

number of basic operations. After each operation a suggestion engine analyzes any selected lines and offers the user a set of possible construction operations. Like an expectation list, the suggestions are presented to the user on the lower portion of the screen as a series of smaller, thumbnail-sided complete renderings. Although the user can make an explicit selection after providing input, the user is also free to continue entering additional strokes. This means that the user can shift the system's focus at any time, allowing the user to come at a single operation from many different paths and incorporate components created at different times.

5.2. Reconstruction

Interpreting a user's drawings as gestures may provide a quick and easy way to generate 3-D content, but it also distances the user's drawn input from the resulting geometry. In response a number of researchers have attempted to develop systems that interpret the user's drawings more directly.

Although a 2-D projection has no real depth, when we look at a drawing or picture our brains can reconstruct the 3-D shape of each object and describe the relative distances between them. This reconstruction process is so fluid and natural that our brains make it look like child's play, but the underlying task is surprisingly complex. Only by accepting some level of ambiguity, drawing on clues in the image, and applying basic knowledge about how the world works, are our brains able to see past the literal 2-D projection and construct a plausible 3-D model (Lehar, 2004). Despite its apparent effectiveness, we are all familiar with simple optical illusions and "impossible figures" which easily thwart our perceptive systems. This feat is still beyond the capabilities of even the most advanced artificial intelligence system, however researchers are now applying many of these same techniques under more controlled conditions in order to reconstruct 3-D scenes from 2-D images.

One method of resolving ambiguity is to apply previous knowledge, a common technique of machine learning. Lipson and Shpitalni (2002) for example describe a modeling system that can reconstruct a user's 2-D wire-frame drawing by comparing the geometry to models it has seen in the past. The program's 'memory' is in the form of correlation tables—statistical data derived from 100,000 randomly generated 3-D models and their 2-D projections.

An alternative method of reconstructing line drawings focuses on optimizing a number of smaller elements in order to gain a global understanding of an image. Huffman (1971), Clowes (1971) independently developed a formal method for reconstructing a specific class of line drawings based on properties of individual lines and their intersections. Today this method is referred to as *Huffman–Clowes line labeling*. The Huffman–Clowes system deals with what are called 'trihedral planar objects', forms consisting of only flat faces and containing vertices with no more than three incident edges.

The basis of the system is the fact that, no matter the overall shape of a model, given three incident edges there are only so many ways the edges can meet at a vertex that make physical sense. In turn, each line entering the junction can be assigned one of three labels: *convex* edges protrude out as in a mountain fold or the edge of a cube, *concave* edges sink in as in a valley fold, or *occluding* edges, which are convex edges along the silhouette of a 3-D shape from the viewpoint of the 2-D projection – see Fig. 4. Huffman and Clowes produced a catalogue of all possible junction labeling for trihedral planer objects. Because each line in the drawing is straight, the label assigned to the line never changes along its length, thus a valid 3-D projection must necessarily have a consistent labeling. In other words, the label assigned by selecting a junction type at one end of an edge must match the assignment at the other extreme.[2]

In the past, line labeling has primarily been used as an approach to computer vision, but the technique was first applied to sketch-based modeling by Grimstead and Martin (1995). In this system, the user provides a 2-D hidden line drawing, which is analyzed to generate possible labelings. The system then negotiates with the user to select the proper configuration and then a system of linear equations determines the *z*-coordinates of each vertex in the drawing. This basic design was taken a step further by Varley et al. (2004) in their RIBALD modeling system. Following the basic reconstruction, RIBALD allows the user to use the reconstructed edges of the model as a framework onto which curved edges can be placed to create a more expressive model.[3]

Although these systems are effective at reconstructing certain classes of objects, their primary drawback is the immense computational complexity of the reconstruction algorithms. Systems are generally limited to reconstructing simple shapes with few edges,

[2] Varley et al. (2004) discussed extensions of these techniques into more complex figures, and Williams (1997) discusses the inclusion of curved as well as straight lines.

[3] An alternative reconstruction algorithm based on propagation is discussed by Masry et al. (2005).

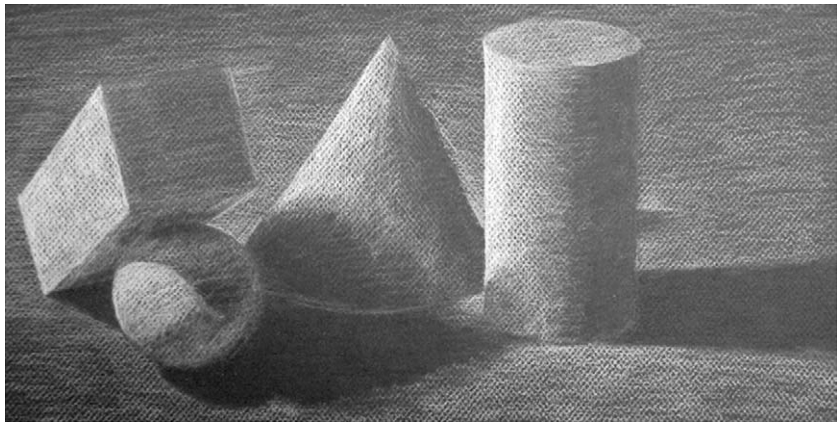

Fig. 5. A typical art student's exercise: a collection of basic 3-D forms is defined by the play of light and shadow across each object's surface rather than through the use of outlines.

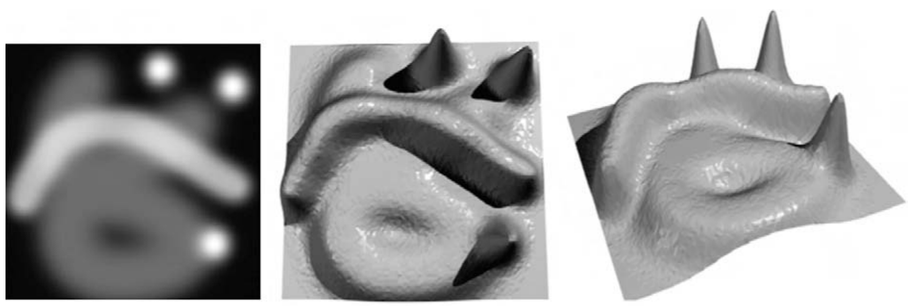

Fig. 6. An example of height-field modeling. The 2-D image to the far left is interpreted by a modeling system to generate the surface displacement to the right. Light areas of the image are translated into peaks in the 3-D model, whereas darker areas result in valleys.

and even then processing times can be less than interactive. Even under tight constraints, these systems must also deal with many ambiguous cases in which multiple interpretations of a line drawing are equally plausible—situations that can only be resolved through user interaction.

Some have suggested that reconstruction systems could be aided by providing additional contextual information. Turner et al. (1999)'s Stilton system, for example, allows the user to draw their geometry into an existing 3-D model. The reconstruction system then uses clues from the preexisting geometry such as the ground plane to aid in the labeling process.

At the other extreme are approaches to reconstruction that severely restrict the parameters of the reconstruction task. Davis et al. (2003) for example applied a specialized version of line labeling to character animation to allow 3-D animators to quickly create motion sequences by drawing simple stick figures. By narrowly limiting the interpretation of a user's sketches down to a specifically defined set of stick figure components, issues of line labeling such as efficiency and ambiguity are diminished.

Given the inherent limitations of line labeling and related optimization techniques, the flexibility and creativity of the various sketch-based modeling applications that use this process is astonishing. For some limited applications in which the desired models consist of planar trihedral volumes or other related shapes, line labeling might be a workable solution, however for general modeling the exclusion of curves and the heavy computational cost of these methods make them infeasible as a basis for a general modeling interface. As the work by Davis et al. suggests, line labeling may find more use in tangential systems or as an abstracted interface for manipulating more complex attributes in an intuitive way.

5.3. Height-fields and shape from shading

Although traditional sketching makes heavy use of line, to the artist these lines serve as a convenient and universal shorthand to describe abrupt transitions in value. In some situations, artists find it more appropriate to deal with these values directly in a process called shading—creating variations in tone and shadow that generate the illusion of light on a 3-D surface—see Fig. 5. Many researchers have recognized that the depth and surface normal information described by an artist's shading offers a potential source of information that could be used in reconstructing the 3-D shape of an object from a 2-D shaded image.

In the past, perturbations of the lighting model have been used to add surface details to 3-D models in a process called 'bump maps' (Blinn, 1978), however these effects are only an illusion. This process can be taken one step further into displacement mapping, which rather than disturbing the surface normals, actually changes the geometry of a surface based on a displacement map (Williams, 1990). However, this method still relies on underlying geometry.

A first step towards the use of shading in modeling is the use of *height-fields* or *digital elevation models* (DEM) – see Fig. 6. In the same way that a grayscale raster image assigns an intensity value to each pixel within a composition, a height-field image assigns a value to each pixel corresponding to the distance between the viewer and the surface of an object. The result is a grayscale image that can then be interpreted by a modeling system to generate a 3-D surface—think, for example, of a topographical map with darkly colored valleys, bright peaks, and a smooth gradation in between. Because this height field data is represented as a 2-D raster graphic, it can be edited using off the shelf image manipulation programs (Williams, 1990). By applying familiar painting tools such as smearing brushes and lasso selection the user can manipulate the height-field data in 2-D, and then convert the model back to a 3-D surface representation.

However, as Williams notes, working directly with height-fields has some difficulties. First, most users are unfamiliar with interpreting a greyscale image as height rather than light information. Furthermore, Rushmeier et al. (2003) point out that the value scale of a height-field image is always oriented towards the viewer. Interpreted (incorrectly, but intuitively) as a shaded image this corresponds to a light source shining directly from the viewer's position, a direction that tends to wash out many small details.

Where as height-fields work with depth information directly, another technique called 'shape from shading' attempts to reconstruct this information from a more traditional shaded image. However, this reconstruction process is particularly difficult. Because the shading is a property not only of the position and shape of the object but also the location of the light source, even under ideal conditions it is necessary to know or estimate additional information about the scene such as each light's position and intensity in order to make a successful reconstruction. Furthermore, ambiguities and mistakes in artist's renderings create additional problems. Thus, systems that successfully apply shape from shading are those that can narrow these variables to some extent.

Rushmeier et al. for example tackle this problem by generating a shaded image from an existing 3-D model. The user can edit this shaded image, then reenter the changes to the system where edits are reinterpreted as changes to the 3-D shape. The key to the reintegration process is that the original image was based on a rendering created by the system, meaning that the exact and unambiguous size and orientation of the model, and the positions of any and all light sources are known quantities.

As Kerautret et al. (2005) points out, this method is appropriate for small scale edits and repairs, but is not well suited for creating new geometry from scratch. In response, Kerautret and colleagues describe a system in which the user provides several 2-D shaded sketches, each with its light source in a different location. These sketches are then combined and interpreted by the reconstruction algorithm to generate a single $2\frac{1}{2}$-D model—that is, a model with depth extending from a single direction, like the shape of a sheet laid over an object placed on a table.

This sort of $2\frac{1}{2}$-D/3-D depth painting interface to modeling has also found its way into the commercial sphere. First introduced in 1999, ZBrush (Pixologic, 2007) is a sort of hybrid paint and modeling application that uses a unique pixel representation to store both standard pixel information (color and alpha values), as well as modeling information (depth, texture, and material properties). Using ZBrush, the artist can not only paint an image, but also push and pull the canvas surface in and out. This allows the artist to apply lighting effects and other traditionally 3-D modeling techniques to the creation of 2-D artwork.

Although applications like ZBrush make effective use of this sort of modeling interaction, as we can see from the efforts of researchers like Rushemier et al. and Kerautret et al., as the focus of a modeling interface this method of creating 3-D geometry has some serious drawbacks. Working directly with height-field data is effective for generating minor details, but for general modeling its visual interpretation is both unintuitive and uninformative to the artist. Working with shading information may be more familiar to the artist, but for the computer this mode of interaction complicates matters beyond the range of an interactive application.

5.4. Deformation and sculpture

Whereas height-fields and shape from shading techniques manipulate a model from a single direction, this idea can be extended to working more directly in 3-dimensions in a technique called 'deformation'. Here, the surface of a 3-D object is interactively pushed in, puffed out, pulled, smudged, smoothed, gouged, or otherwise deformed to create the model's features.

Deformations are not a new technology in the 3-D modeling arena, but it is their resemblance to physical artistic techniques that relates them to the field of sketch-based modeling. In many ways, deformation systems allow users to 'sculpt' the surface of a model by applying deformation operations.

The most general form of deformation operations are *global deformations*, and can be thought of much like an image processing filter—applying a single change to the whole model. An example of a system using global deformations is provided by Wyvill and Guy (1998), who's application uses spatial warping functions and their hierarchical arrangement into BlobTree structures to create models.

A more popular approach to deformation as a modeling tool has been the use of *local deformations*. These are operations that effect only a small portion of the model, and can often be equated to activities like sculpting or carving. An early example of this method is presented by Galyean and Hughes (1991). The authors' system uses a voxel based volume representation coupled with a carving tool controlled by a force feedback 3-D pointing device to add and remove material from a virtual sculpture.[4]

A major limiting factor for these voxel-carving applications is the fact that as the model grows bigger, there is an exponential growth in the memory requirements of the underlying voxel structures. This ultimately places a low ceiling on the level of detail that can be expressed by the user. One possible alternative to is the use of an adaptive voxel grid or 'adaptively sampled distance field' (ADF), a technique proposed by Frisken et al. (2000).[5]

Implicit modeling offers still another alternative representation for volumetric modeling in which the model is represented by implicit functions. Andreas and Niels Jørgen (2002), describe a volume sculpting system based around the level-set implicit surface representation, and provide a number of references on volumetric modeling. Han and Medioni (1997)'s 3-DSketch system uses an equipotential surface representation to allow users to quickly trace and then refine clay-like models using a 3-D stylus.

Because of the resemblance to 3-D artistic techniques like sculpture, a number of systems in this category make use of 6-degree-of-freedom pointing devices and other spatially tracked input sources as a means of modeling. Ferley et al. (1999) for example demonstrate a system in which gobs of material can be deposited in space using a 'toothpaste' tool directed in such a manner. Deering (1996)'s HoloSketch system uses a 6-DOF wand device along with a head tracking and 3-D display system to allow a user to sweep out shapes in mid air. Schkolne et al. (2001)'s surface drawing project pushes this manual creation idea one step further. A user wearing a special data glove creates ribbon like surfaces by sweeping the gloved hand through space over a workbench display.

Clay like deformation can also be applied to more standard polygonal model representations. The commercial application Mudbox (Skymatter, 2007) for example allows users to create basic shapes like blocks and spheres or import existing models and then subdivide and deform their surfaces using a variety of 'brush' tools. Resulting models are polygon meshes that can easily be imported into other software for texturing or animation.

A more novel approach to deformation called 'velocity paint' is described by Lawrence and Funkhouser (2003). In this system, rather than deforming a model directly, a user paints the surface

[4] In 1995 Wang and Kaufman (1995) developed a similar system called VolVis that uses a more traditional 2-D mouse as input.

[5] Based on this technique, Perry and Frisken (2001) developed a sculpture based modeling application called Kizamu.

of a model signifying areas he or she would like to be distorted. Each color of paint signifies a different distortion operation like growing, shrinking, etc. Once the user has applied paint, the model is simulated and the distortions gradually take effect as an interactive animation.

Finally, along with paint-like methods, several researchers have investigated sketch-based deformation interfaces that use strokes as their control mechanism. A representative paper in this area, authored by Singh and Fiume (1998), describes a sketch-based deformation method called 'wires' in which parametric curves are bound to an arbitrary model to act as manipulation handles.[6]

Deformations provide artists and designers with powerful tools to affect the look of their models. However, just as a sculptor begins with a block of stone or a lump of clay, for the most part, deformation operations must be applied to an existing model. It can be difficult to create an entirely new model using these techniques. Deformation can instead be integrated into other modeling techniques, where it functions as one of a number of tools at the artist's disposal. Fully 3-D sculpting systems offer another approach where matter can be created and manipulated manually. While this approach more closely fits with working in a 3-D environment, such systems require additional space and expensive display and input equipment. 3-D interfaces can also be more physically taxing on the user, who must manipulate and hold their hands and arms in space for extended periods of time (Deering, 1996).

5.5. Blobby inflation

Although many of the other methods discussed above are capable of creating 3-D models from user input, few approach the idealized image that a name like 'sketch-based modeling' conjures up—the ability to draw an object and have it literally pop into 3-dimensions. While the previously mentioned approaches tend to provide a user with a high degree of control, some researchers have explored how a limited set of modeling capabilities might provide a more satisfying, if less capable interface.

Developed by Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka, Teddy is a 3-D modeling system designed for children that is capable of generating charmingly bulbous 3-D models from a user's simple silhouette drawing (Igarashi et al., 1999). At the center of the system is an ingenious inflation method that converts the user's input into a 3-D shape.

To create a model, the user begins by drawing a simple, 2-D outline. The user's stroke is collected as a closed polyline loop, and then analyzed by the system to find a central chordal axis or 'spine', a single branching line that passes through the middle of the closed shape. Vertices of the spine are elevated from the plane of the initial stroke based on their distance from the stroke, and used to form a tessellated mesh dome that is mirrored to the other side and sewn together to create a symmetric, watertight model topologically equivalent to a sphere.

Visually the effect is as if the user's outline described a balloon that is then inflated (hence the name). Circles become spheres, long ovals become cigars, etc so that resulting models have a characteristically rounded shape. Once created, users can further augment their models by creating extrusions or bites into the surface using similar techniques. Although a wide variety of shapes can be created using this method, models are limited to single pieces, and do not contain sharp features like corners or edges. Despite these limitations, the whole process is simple and straightforward, and requires no interpretation or negotiation with the user, meaning that models can be created extremely quickly, and with a minimum of interface complexity.

Teddy's simple interface proved to be inspirational to a number of other researchers. SmoothSketch developed by Karpenko et al. (2002) is an inflation-based modeler built around variational implicit surfaces. A system by Ohwada et al. (2003) allows for the creation of objects with complex topologies and internal structures. A similar approach using an implicit-modeling representation called convolution surfaces is presented by Tai et al. (2004)'s ConvoMo. Here a user adjustable cross section is convolved along the chordal axis to create the model shape, allowing greater flexibility and the creation of 'semi-sharp' rather than blobby features. Finally, ShapeShop, developed by Schmidt et al. (2005), attempts to expand inflational modeling to a more fully featured application.

The inflation based modeling systems provide a prime example of how even simple modeling interfaces can provide effective and even powerful means of creating 3-D models quickly and more intuitively. Although the modeling vocabularies of early systems like Teddy are limited, by focusing on simplified interfaces and drawing as a primary input method, users are able to get just enough functionality and an unobstructed view of the modeling process to not be bothered by its limitations. We can also see from later examples like that of Schmidt et al.'s ShapeShop that these blobby inflation techniques can still form the basis of a more advanced and well rounded system.

5.6. Contour curves and drawing surfaces

Where as traditional techniques like sculpture and carving bear an obvious link to 3-D modeling, the connection between the physical activity of drawing and 3-D modeling is more tenuous. From a practical standpoint, for designers and artists drawing is a 2-D activity, and so some research has been directed at how best to use 2-D drawing input in a 3-D environment. This includes both utilizing 2-D strokes in 3-D, and allowing the user to draw fully 3-D space curves.

When working with traditional 2-D input, a common technique has been the use of drawing planes or drawing surfaces—artificial 2-D structures onto which the user's strokes are projected, thus positioning them in 3-D space. Some projects have taken this approach quite literally. Bimber et al. (2000) for example describe an immersive modeling system where in the user stands above a 3-D workbench display wearing special glasses. The user is provided both a stylus, as well as a translucent plexiglass sketchpad, both of which are tracked in 3-D. The pad allows the user position 2-D sketching input in space, and provides a surface for other inputs like gestures and handwritten notes. Sachs et al.'s (1991) 3-draw system takes a similar tact, affixing the model to a 3-Dly tracked physical ''palette'', allowing the user to not only draw on a solid surface, but to move and rotate the model in progress by physically manipulating the palette in space.

A less physical approach can be found in work by Grossman et al. (2001). Automotive design is one industry in which contour and profile curve modeling is heavily utilized. In order to create characteristic smooth flowing lines automotive designers traditionally employ a unique one-to-one drawing technique known as tape drawing in which long strips of photographic tape are applied to large vertical work surfaces, creating something akin to a wire frame projection of an auto-body design. In their paper, Grossman et al. (2001) explored how this technique could be extended to creating 3-D wireframe models. In their system, the sensation of tape drawing is simulated by two 6-degree-of-freedom 3-D pointing devices directed against a large screen. As the user draws out virtual strips of tape, the lines and curves they form are projected onto planar work surfaces within the 3-D environment. By positioning the planes the user can construct a 3-D

[6] This method of deformation was applied to mesh-based models by Kho and Garland (2005).

wireframe model from these planar curves. Based on Grossman et al.'s interface designs, Tsang et al. (2004) developed a similar tape drawing system using a more standard digitizing tablet interface.

Whereas Grossman et al.'s system is limited to planer curves, in their 1999 paper Cohen et al. (1999) developed a method of drawing 3-D space curves from a 2-D interface. The researcher's approach was to allow the user to draw the curve from a single perspective, and then draw the shadow cast by the curve onto a nearby plane. The system then combines these two strokes to create a single space curve by projecting the first curve onto a surface drawn out from the shadow perpendicular to plane.[7]

The idea of drawing in space has also been explored as a method of altering or annotating existing models or interacting in 3-D environments. Perhaps the first example of this approach was inspired by the 1955 classic children's book *Harold and the Purple Crayon* by Johnson (1981). Taking this imaginative tale as a cue, researchers at Brown University developed Harold, a prototype program designed to facilitate creative exploration and storytelling for young children (Cohen et al., 2000). In their application, users explore a virtual landscape in which they can draw objects in mid air. As the user draws, the system automatically creates an invisible planar surface parallel to the viewing plane called a billboard, and projects the marks onto its surface. Because the artwork only defines one side of an object, as the user moves around the environment the boards turn to face them from any angle.

A similar tact was taken by Bourguignon et al. (2001) to allow what the authors call 'guided design'. Here users might begin, for example, with an exiting 3-D model of a dressmaker's dummy, and draw their clothing designs over the form in 3-dimensions. To generate models from silhouette strokes, the system converts the user's drawing input into transparent troughs or partial tube-shaped 3-D surfaces onto which the visible line is projected. The trough is shaped by the curvature of the stroke and the visible portion is located at its apex. As the user's view changes, the visibility of the stroke attenuates, giving the illusion of a 3-D surface contour. Because these strokes actually have 3-D surfaces associated with them the system can perform limited occlusion with other elements.

Together, these projects demonstrate that it is possible to create and situate curves in three dimensions. They also provide an example of how general 3-D space curves can be defined by the same system. However, indications from the researchers' publications suggest that the creation of completely general 3-D space curves can be difficult and even frustrating for the user. Although some of this difficulty can be blamed on deficiencies in the interfaces themselves or the limitations of 2-D display devices, remarks by the authors suggest that to some degree users simply find it taxing to fully consider the 3-D structure of the curves. Grossman et al. (2002) for example noted that the majority of modeling time by test subjects using their system was not spent creating curves, but carefully considering what curves to create in order to best define the desired model.

5.7. Stroke based constructions

Although 3-D curves and wireframe models are appropriate for some applications, the vast majority of 3-D modeling activity focuses on the creation of surface or volume-based models. Many traditional modeling systems now incorporate skinning and lofting features which allow surfaces to be fit to a series of curves. Sketch-based modeling provides an intuitive means of creating those initial curves, as well as editing both curves and surfaces once they are created.

One example of this sort of system is FreeDrawer (Wesche and Seidel, 2001). Built around an immersive 3-D workbench and two 3-D pointers, this system allows a designer to draw and edit networks of spline curves. Loops within this network can then be filled with surfaces to create a model. A more powerful example of this sort of sketch-based curve and surface editing is provided by Michalik et al. (2002). To create a model the user draws strokes with a digitizing tablet or mouse into the 3-D environment, which are then projected onto planar drawing surfaces. These strokes then enter a constraint solving system that generates a b-spline surface approximating the shape suggested by the curves. Because curves are used as constraints rather than scaffolding, the variety of shapes that can be created by this system is more diverse.

Systems of this sort provide a much more intuitive method of quickly generating parametric surfaces. However, as one might expect the complexity of the fitting process can be quite high. For their part, Michalik et al. employ a number of techniques to cope with the explosive growth of the problem, but admit that as the number of constraints increases, performance is adversely effected.

It may at first seem counterintuitive to forgo additional functionality to arrive at a more functional system. However it is important to remember that the underlying techniques of traditional sketching are not based on precision or quality of output, but on the speed and ease with which that output can be created by the artist. We have already seen how seemingly simple systems based on the inflation techniques make this tradeoff, forgoing user control of the 3-D aspect of the geometry in exchange for rapid development from silhouette strokes alone.

Other researchers have taken note of this as well. Starting from the same basic interface as Igarashi et al.'s Teddy, Levet et al. (2006) for example developed a more expressive interface by allowing the user slightly more control. The Levet et al. system, rather than working with a silhouette shape alone, requires two inputs from the user: a silhouette stroke, and a profile curve. Profile curves replace the standard rounded cross sectional profile used in Teddy, allowing not only rounded shapes to be created, but also forms with an arbitrary cross section.

This basic idea has been expanded on by a number of researchers. An approach that has shown some recent promise is the use of procedural modeling methods, especially those based on generalized sweeps and extrusions. A good example is provided by Cherlin et al. (2005). Starting from a traditional drawing technique in which the artist draws tight spirals to feel out a 3-D shape, the authors developed a construction method they call *rotational* and *cross sectional blending surfaces* in which the user defines a closed cross sectional shape, and a pair of contour curves. A parametric surface is then generated by sweeping the cross section along the path defined by the two curves, while dynamically scaling the cross section based on the curve's relative separation. Once a number of modeling components have been constructed, the user can further distort their shapes using a stroke based deformation system, and position them in 3-D space to construct a complete model.

Techniques such as these described by Levet et al. and Cherlin et al. have a number of advantages. First, as Cherlin et al. note, because each surfaces is generated from parametric strokes, the surfaces themselves have a parametric definition, allowing them to be evaluated at arbitrary levels of precision, and providing a ready-made coordinate system for the application of surface techniques like textures. Second, from the user's perspective, models are created directly from input strokes, making the construction process highly intuitive. Finally, like the inflation based systems, because the algorithms used to generate the surfaces are straightforward and mechanical, they can easily run at interactive rates.

[7] A more intuitive simplification of this technique was later used in an updated version of Grossman et al. (2002) system.

6. Discussion

In examining these projects we can make a number of important observations. First, although the result of the modeling process is a 3-D form, in most cases every aspect of the user's interaction with that form is through the lens of 2-D interface devices. Mice, tablets, monitors—each of these devices places inherent limits on the user's expressivity and dexterity within the modeling environment. Thus, although interaction methods that mimic 3-D techniques such as carving or sculpting are available, systems that accommodate the user's 2-D experience of the modeling process provide a more manageable interface.

Second, because the practice of sketching spills the user's thought process into a physical (or virtual) form, additional layers of interface between the user and the modeling environment only serve to dilute and distract the user's thought process. Systems that can provide a direct and responsive experience better simulate the conditions of traditional sketching.

Third, no matter the quality or accuracy of an interpretation system, drawing is such an expressive medium that there are likely to always be situations too ambiguous for the computer system to divine on its own. That same expressivity and open nature of sketching means that rather than representing occasional troublesome conditions or the odd edge cases, these ambiguous situations are perhaps more common than not. In confined circumstances an interpretive system can be appropriate, and even a boon, but at early stages of design it is the ambiguity of sketching that both confounds discrete interpretation and fuels the creative process.

Finally, although a goal of sketch-based modeling is to translate traditional sketching and drawing skills into the modeling arena, modeling is an inherently different activity than drawing. Although sketching and drawing may form the basis of a new interface, relying on developed traditional techniques like shading, perspective, or occlusion to provide admittedly related but impoverished spatial information is, given current technology, a difficult prospect. Rather than literally reproducing existing techniques, a more viable approach may be to extract from those techniques the basic physical activities and mental processes that facilitate them, and then cater to those aspects directly in the modeling system (Cook, 2007).

In conclusion, in the present work we have provided the reader an introduction to the topic of sketch-based modeling. We have discussed the basic tenants of traditional sketching, and how those skills translate to the field of 3-D modeling, we have covered the unique requirements of sketch-based modeling interfaces, and we have provided a brief survey of seven approaches in the field.

Given the wide variety of approaches to sketch-based modeling, it is unclear what form the modeling interfaces of even the near future may take. However, if the current crop of research is any indication, no matter their field users should be able to look forward to a future of systems that cater far more to the way *they* work rather than the other way around.

References

Alex, John. 2005. Hybrid Sketching: A New Middle Ground Between 2- and 3D. Ph.D. Thesis, Massachusetts Institute of Technology.

Bærentzen, J. Andreas, Christensen, Niels Jørgen. 2002. Volume sculpting using the level–set method. In: Proceedings of the Shape Modeling International 2002 (SMI'02). IEEE Computer Society, p. 175.

Baudel, Thomas, 1994. A mark-based interaction paradigm for free-hand drawing. In: Szekely, Pedro (Ed.), Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology, pp. 185–192.

Bimber, Oliver, Miguel Encarnação, L., Stork, André, 2000. A multi-layered architecture for sketch-based interaction within virtual environments. Computer Graphics 24 (6), 851–867.

Blinn, James F., 1978. Simulation of wrinkled surfaces. Computer Graphics 12 (3), 286–292.

Bourguignon, David, Cani, Marie-Paule, Drettakis, George, 2001. Drawing for illustration and annotation in 3D. Computer Graphics Forum 20, 114–122.

Cherlin, Joseph Jacob, Samavati, Faramarz, Sousa, Mario Costa, Jorge, Joaquim A. 2005. Sketch-based modeling with few strokes. In: Proceedings of the 21st Spring Conference on Computer Graphics. ACM Press, Budmerice, Slovakia, pp. 137–145.

Clowes, Maxwell B., 1971. On seeing things. Artificial Intelligence 2, 79–116.

Cohen, Jonathan, Hughes, John F., Zeleznik, Robert, 2000. Harold: a world made of drawings. In: Proceedings of the First International Symposium on Non Photorealistic Animation and Rendering (NPAR) for Art and Entertainment.

Cohen, Jonathan M., Markosian, Lee, Zeleznik, Robert C., Hughes, John F., 1999. An interface for sketching 3D curves. In: Proceedings of the 1999 ACM Symposium on Interactive 3D Graphics. ACM Press, Atlanta, Georgia.

Cook, Matthew, 2007. A 3-dimensional Modeling System Inspired by the Cognitive Process of Sketching. Master's thesis, The University of Kansas.

Davis, James, Agrawala, Maneesh, Chuang, Erika, Popović, Zoran, Salesin, David, 2003. A sketching interface for articulated animation. In: Symposium on Computer Animation, pp. 320–328.

Deering, Michael F., 1996. The HoloSketch VR sketching system. Communications of the ACM 39 (5), 54–61.

Derry, John, 1996 (Thursday, September 28, 2006). Of Pianos and Oranges. <http://zonezero.com/magazine/articles/derry/derry.html>.

Do, Ellen Yi-Luen, 2005. Design sketches and sketch design tools. KBS – Knowledge Based Systems 18, 383–405.

Do, Ellen Yi-Luen, Gross, Mark D., 1996. Drawing as a means to design reasoning. In: Visual Representation, Reasoning and Interaction in Design Workshop notes, Artificial Intelligence in Design'96 (AID'96). Stanford University.

Durand, Frédo, 2002. An invitation to discuss computer depiction. In: Proceedings of The 2nd International Symposium on Non-photorealistic Animation and Rendering. ACM Press, Annecy, France, pp. 111–124.

Eggli, Lynn, Brüderlin, Beat D., Elber, Gershon, 1995. Sketching as a solid modeling tool. In: ACM Symposium on Solid and Physical Modeling: Proceedings of the Third ACM Symposium on Solid Modeling and Applications. ACM Press, Salt Lake City, Utah.

Ferley, Eric, Cani, Marie-Paule, Gascuel, Jean-Dominique, 1999. Practical volumetric sculpting. Implicit Surfaces.

Fiore, Fabian Di, Reeth, Frank Van, 2002. A multi-level sketching tool for pencil-and-paper animation. In: American Association for Artificial Intelligence (AAAI 2002) Spring Symposium. Stanford University in Palo Alto, AAAI, California.

Fleisch, Timo, Rechel, Florian, Santos, Pedro, Stork, André, 2004. Constraint stroke-based oversketching for 3D curves. In: Hughes, John F., Jorge, Joaquim A. (Eds.), EUROGRAPHICS Workshop on Sketch-based Interfaces and Modeling. pp. 161–165.

Frisken, Sarah F., Perry, Ronald N., Rockwood, Alyn P., Jones, Thouis R., 2000. Adaptively sampled distance fields: a general representation of shape for computer graphics. In: Akeley, Kurt (Ed.), Siggraph 2000, Computer Graphics Proceedings. ACM Press/ACM SIGGRAPH/ Addison Wesley Longman, pp. 249–254.

Galyean, Tinsley A., Hughes, John F., 1991. Sculpting: an interactive volumetric modeling technique. Computer Graphics 25 (4), 267–274.

Garner, Steve. 2001. Is sketching still relevant in virtual design studios? In: Proc. Design Computing on the Net (DCNet 2000), a Refereed Virtual Conference.

Grimstead, Ian. J., Martin, Ralph, R., 1995. Creating solid models from single 2D sketches. In: SMA'95: Proceedings of the Third Symposium on Solid Modeling and Applications. ACM Press.

Grossman, Tovi, Balakrishnan, Ravin, Kurtenbach, Gordon, Fitzmaurice, George, Khan, Azam, Buxton, Bill, 2001. Interaction techniques for 3D modeling on large displays. In: SI3D'01: Proceedings of the 2001 Symposium on Interactive 3D Graphics. ACM Press.

Grossman, Tovi, Balakrishnan, Ravin, Kurtenbach, Gordon, Fitzmaurice, George W., Khan, Azam, Buxton, William. 2002. Creating principal 3D curves with digital tape drawing. In: CHI 2002 Conference Proceedings ACM Conference on Human Factors in Computing Systems, pp. 121–128 .

Han, Song, Medioni, Gérard. 1997. 3DSketch: modeling by digitizing with a smart 3D pen. In: MULTIMEDIA'97: Proceedings of the Fifth ACM International Conference on Multimedia. ACM Press, Seattle, Washington.

Henzen, Alex, Ailenei, Neculai, Fiore, Fabian Di, Reeth, Frank Van, Patterson, John, 2005. Sketching with a low-latency electronic ink drawing tablet. In: Computer Graphics and Interactive Techniques in Australasia and South East Asia Archive Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia, pp. 51 – 60.

Herbert, Daniel M., 1993. Architectural Study Drawings. Wiley.

Huffman, David A., 1971. Impossible objects as nonsense sentences. Machine Intelligence 8, 475–492.

Huot, Stéphane, Dumas, Cédric, Hégron, Gérard, 2003. Toward creative 3D modeling: an architects' sketches study. In: INTERACT 2003 – Bringing the Bits together Ninth IFIP TC13 International Conference on Human–Computer Interaction.

Igarashi, Takeo, Hughes, John F., 2001. A suggestive interface for 3D drawing. In: 14th Annual Symposium on User Interface Software and Technology, ACM UIST'01. ACM Press, Orlando, Florida.

Igarashi, Takeo, Kawachiya, Sachiko, Matsuoka, Satoshi, Tanaka, Hidehiko. 1997. In search for an ideal computer-assisted drawing system. In: The Sixth IFIP Conference on Human–Computer Interaction, pp. 104–111.

Igarashi, Takeo, Matsuoka, Satoshi, Tanaka, Hidehiko, 1999. Teddy: a sketching interface for 3D freeform design. In: ACM SIGGRAPH'99. ACM, Los Angels, California.

Johnson, Crockett., 1981. Harold and the Purple Crayon, 50th anniversary edition. HarperTrophy.
Karpenko, Olga, Hughes, John F., Raskar, Ramesh, 2002. Free-form sketching with variational implicit surfaces. In: Eurographics 2002.
Kerautret, Bertrand, Granier, Xavier, Braquelaire, Achille, 2005. Intuitive shape modeling by shading design. In: Butz, Andreas, Fisher, Brian, Krüger, Antonio (Eds.), International Symposium on Smart Graphics Lecture Notes in Computer Science, vol. 3638. Springer-Verlag GmbH, pp. 163–174.
Kho, Youngihn, Garland, Michael, 2005. Sketching mesh deformations. In: Symposium on Interactive 3D Graphics. Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games. ACM, Washington, District of Columbia, pp. 147–154.
Lawrence, Jason, Funkhouser, Thomas, 2003. A painting interface for interactive surface deformations. In: Pacific Graphics.
Lehar, Steven, 2004. Gestalt isomorphism and the primacy of subjective conscious experience: a Gestalt bubble model. Behavioral and Brain Sciences 26 (4), 375–444.
Levet, Florian, Granier, Xavier, Schlick, Christophe, 2006. 3D sketching with profile curves. In: International Symposium on Smart Graphics.
Lipson, Hod, Shpitalni, Moshe, 2002. Correlation-based reconstruction of a 3D object from a single freehand sketch. In: Davis, Randall, Landay, James, Stahovich, Tom (Eds.), American Association for Artificial Intelligence Spring Symposium: Sketch Understanding. Stanford University in Palo Alto, AAAI, California.
Masry, Mark, Kang, Dong Joong, Lipson, Hod, 2005. A freehand sketching interface for progressive construction of 3-D objects. Computers and Graphics 29, 563–575.
Michalik, Paul, Kim, Dae Hyun, Bruderlin, Beat, D., 2002. Sketch- and constraint-based design of B-spline surfaces. In: ACM Symposium on Solid and Physical Modeling Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications. ACM Press, Saarbrücken, Germany, pp. 297–304.
Ohwada, Shigeru, Nielsen, Frank, Nakazawa, Kazuo, Igarashi, Takeo, 2003. A sketching interface for modeling the internal structures of 3-D shapes. In: International Symposium on Smart Graphics, Lecture Notes in Computer Science (LNCS), vol. 2733. Springer-Verlag, pp. 49–57.
Pereira, João P., Jorge, Joaquim A., Branco, Vasco, Ferreira, F., Nunes, 2000. Towards Calligraphic Interfaces: Sketching 3d Scenes with Gestures and Context Icons. WSCG2000.
Pereira, João P., Jorge, Joaquim A., Branco, Vasco A., Ferreira, F., Nunes. 2003. Calligraphic interfaces: mixed metaphors for design. In: 10th International Workshop, DSV-IS, vol. 2844/2003, pp. 154–170.
Pereira, João P., Branco, Vasco A., Jorge, Joaquim A., Silva, Nelson F., Cardoso, Tiago D., Ferreira, F., Nunes. 2004. Cascading recognizers for ambiguous calligraphic interaction. In: Hughes, John F., Jorge, Joaquim A. (Eds.), EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling.
Perry, Ronald N., Frisken, Sarah F., 2001. Kizamu: a system for sculpting digital characters. In: International Conference on Computer Graphics and Interactive Techniques Proceedings of the 28th Annual conference on Computer Graphics and Interactive Techniques, pp. 47–56.
Pixologic, 2007. ZBrush.
Rushmeier, Holly, Gomes, Jose, Balmelli, Laurent, Bernardini, Fausto, Taubin, Gabriel, 2003. Image-based object editing. In: Fourth International Conference on 3-D Digital Imaging and Modeling (3DIM'03), p. 20.
Sachs, Emanuel, Roberts, Andrew, Stoops, David, 1991. 3-Draw: a tool for designing 3D shapes. IEEE Computer Graphics and Applications 11 (6), 18–26.
Schkolne, Steven, Pruett, Michael, Schröder, Peter, 2001. Surface drawing: creating organic 3D shapes with the hand and tangible tools. In: Proceedings of CHI 2001.
Schmidt, Ryan, Wyvill, Brian, Sousa, Mario Costa, Jorge, Joaquim A., 2005. ShapeShop: sketch-based solid modeling with the BlobTree. In: 2nd Eurographics Workshop on Sketch-based Interfaces and Modeling.
Singh, Karan, Fiume, Eugene, 1998. Wires: a geometric deformation technique. Computer Graphics 32, 405–414.
Skymatter. 2007. Mudbox 1.06.
Tai, Chiew-Lan, Zhang, Hongxin, Fong, Jacky Chun-Kin, 2004. Prototype Modeling from sketched silhouettes based on convolution surfaces. Computer Graphics Forum 23 (1), 71–83.
Tsang, Steve, Balakrishnan, Ravin, Singh, Karan, Ranjan, Abhishek. 2004. A suggestive interface for image guided 3D sketching. In: Conference on Human Factors in Computing Systems Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM Press, pp. 591–598.
Turner, Alasdair, Chapman, David, Penn, Alan, 1999. Sketching a virtual environment: modeling using line-drawing interpretation. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology, pp. 155–161.
Varley, P.A.C., Martin, R.R., Suzuki, H. 2004. Can machines interpret line drawings? In: Hughes, John F., Jorge, Joaquim A. (Eds.), EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling.
Wang, Sidney W., Kaufman, Arie E., 1995. Volume sculpting. In: Proceedings of the 1995 Symposium on Interactive 3D Graphics, pp. 151–ff.
Wesche, Gerold, Seidel, Hans-Peter, 2001. FreeDrawer: a free-form sketching system on the responsive workbench. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology. ACM Press, pp. 167–174.
Williams, Lance, 1990. 3D paint. In: Symposium on Interactive 3D Graphics Archive Proceedings of the 1990 Symposium on Interactive 3D graphics, vol. 24. pp. 225–233.
Williams, Lance R., 1997. Topological reconstruction of a smooth manifold-solid from its occluding contour. International Journal of Computer Vision 23 (1), 93–108.
Wyvill, Brian, Guy, Andrew, 1998. The BlobTree – warping, blending and boolean operations in an implicit surface modeling system. Computer Science Technical Reports, 1998-618-09.
Zeleznik, Robert C., Herndon, Kenneth P., Hughes, John F., 1996. SKETCH: an interface for sketching 3D scenes. In: Rushmeier, Holly (Ed.), SIGGRAPH'96. Addison Wesley, pp. 163–170.