Intuitive Change of 3D Wand Function in Surface Design

Sang-Hun Nam[1], Hark-Su Kim[1], and Young-Ho Chai[2]

[1,2] The Graduate School of Advanced Imaging Science, Multimedia & Film
Chung-Ang University, 221 Heukseok-dong, Dongjak-gu,
Seoul 156-756, Korea
{sanghunnam,harksu}@gmail.com,
yhchai@cau.ac.kr

Abstract. According to the target model for a designer to sketch, an effective style or shape of input device can be defined differently. The spatial sketch system that supports various types of wand can help to sketch efficiently. We suggest the idea of changing wand style by altering the posture of a 3D wand. This method allows a designer to work in intuitive ways without being interrupted by complicated menus. We implement the surface drawing and merging technique with the grid based data structure which deals with multiple strokes from various types of wand.

Keywords: Virtual Reality, Virtual Conceptual Sketch, Surface Modeling, Interaction Technique.

1 Introduction

A number of tangible wands and software have been developed for sketching a model effectively in surface design. In hardware, a haptic device provides 3D interaction with force-feedback to users [1, 2] and enables them to reduce input errors in the depth position due to an incorrect cognition concerning depth cues [3, 4]. The similar shape of input device that user used to sketch on a 2D plane helps users to understand and adapt easily to a new device in spatial sketch system [5, 6]. In software, the wand shape in display can be changed for the purpose. BLUI [7] has a sphere-shaped wand and allow extruded surface to be created by moving a wand in space. Sketch-based 3D modeling systems, such as Teddy [8] and FiberMesh [9], enable the user to create a 3D model by inflating the region described by a silhouette. Thus, various styles of interaction with a 3D wand allow a designer to create spatial input suitable for different purpose. However, if it is necessary to navigate a complicated menu system to change the style of wand, the designer's train of thought will be interrupted. We propose a more intuitive method of changing wand brush style, based on the posture of the wand.

This research used a 3D input device that allowed designers to intuitively sketch within a spatial sketch system. Like the method through which a designer sketches on a 2D plane, the research used a method of drawing and deforming through the repetitive input of strokes [10, 11]. However, since curved lines and surfaces that are drawn overlapping, the drawing and merging algorithm is required to handle a large amount of data. Our system generates a model directly from the paths of different wand

R. Shumaker (Ed.): Virtual and Mixed Reality, LNCS 5622, pp. 216–224, 2009.

brushes, each of which creates a different type of geometry. Our system uses a 3D input device to sketch and deform surfaces through a series of strokes. The drawing space contains pre-defined grids with movable internal vertices, which is similar to marching cubes [12].

This paper is comprised as follows. Chapter 2 explains the definition of wand brushes and the change of the brush style based on the posture of the wand. Chapter 3 explains surface modeling tool with multiple strokes. Chapter 4 explains the results of the paper.

2 Spatial Sketch Inputs with Various Brush Types

The way in which a designer uses a sketching system depends on the models they are trying to construct, and different types of brush would be appropriate for rectilinear and freeform models. Sketch systems therefore provide several brush types, for drawing different forms of geometry, such as curved lines and surfaces.

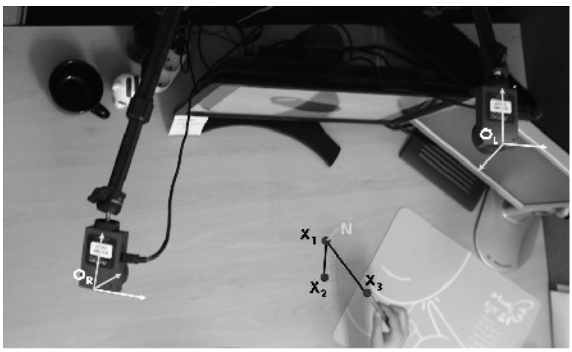
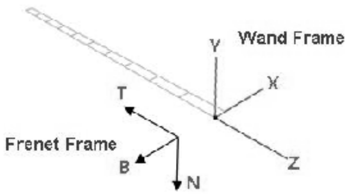


Fig. 1. Spatial sketch system

The position and posture of the wand is acquired by two infrared cameras which have three infrared reflection markers, as shown in Fig. 1. The velocity and acceleration of a moving wand can be calculated from a series of the position data. Also, the curvature (κ_w) and torsion (τ_w) of the wand can be calculated from (1). There are various kinds of brush types according to the relation between these properties and the changing rule of brush. We can associate three types of brush with a 3D wand, using rules which take account of the position and also the posture of the wand.

$$\kappa_w(t) = \frac{\left\| r'(t) \times r''(t) \right\|}{\left\| r'(t) \right\|^3}, \ \tau_w(t) = \frac{(r'(t) \times r''(t)) \cdot r'''(t)}{\left\| r'(t) \times r''(t) \right\|^2} \ . \quad (1)$$

First operation is used to define an initial swept cross-sectional curve as shown in Fig. 2(a). It is based on the Frenet formulas as shown in Fig. 2(b) and does not require menu interactions.



$$\begin{cases} T' = \kappa N, \\ N' = -\kappa T + \tau B, \\ B' = -\tau N. \end{cases}$$

(a) (b)

Fig. 2. Frenet frame and formulas

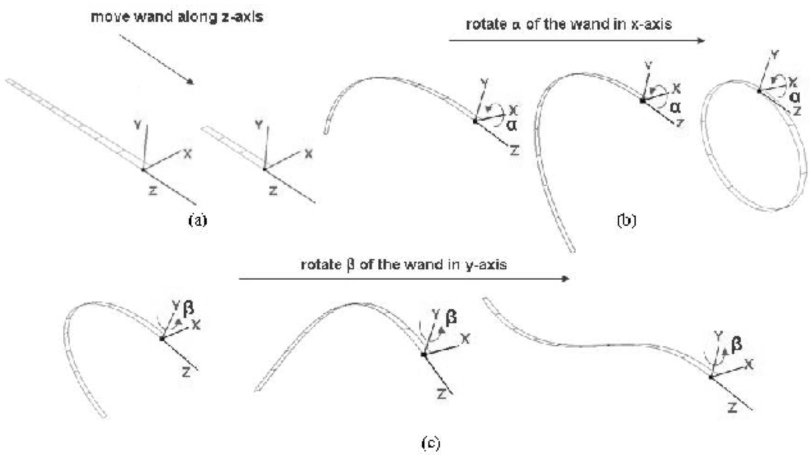


Fig. 3. Curve brush type

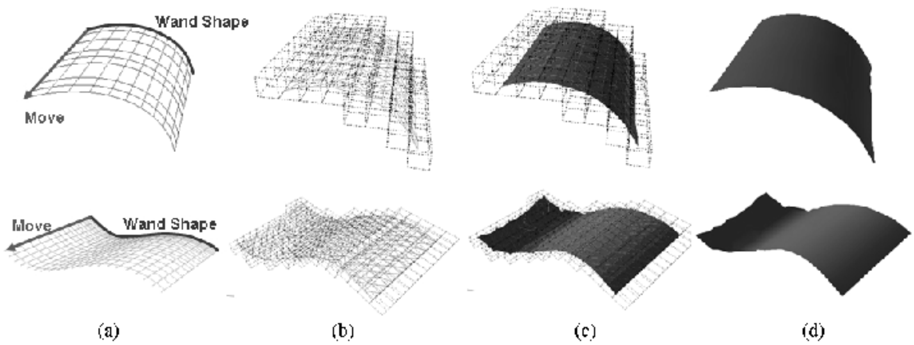


Fig. 4. Drawing surfaces with swept cross-sectional curves

As shown in Fig. 3(a), the length of the initial swept cross-sectional curve is adjusted by moving the wand in the *z*-direction. The curvature in the *z*-direction is created by the rotation α of the wand about the *x*-axis shown in Fig. 3(b), and the torsion in the *z*-direction is determined by the rotation β of the wand about the *z*-axis, as shown in Fig. 3(c). This arrangement can be changed at the beginning of the drawing process to be suitable for a designer.

A surface is defined by the swept cross-sectional curve shown in Fig. 4(a). This is then represented as polygons in the cells of the grid, as shown in Fig. 4(b). The rendered surface is shown in Fig. 4(c) and (d).

Second operation is a similar method to the first operation. However, this method has no need to make initial swept cross-sectional curves, because the cross-sectional curves are reshaped in proportion as the path of the wand automatically [8, 9]. As a designer moves the wand to draw a model in space, the operation gets the paths of the wand and calculates the curvature and torsion simultaneously in (1) as shown Fig. 5(a). The calculated curvature and torsion is applied to the Frenet formulas in Fig. 2(b). The designer can draw surfaces easily without the menu to change initial swept cross-sectional curves as shown in Fig. 5(b).

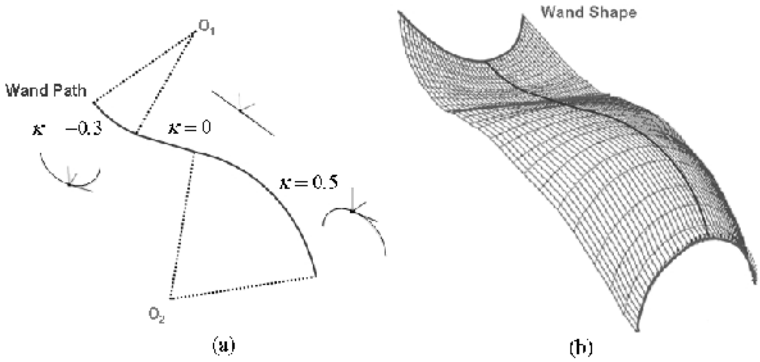


Fig. 5. Drawing a surface with automatic swept cross-sectional curves

Third operation is definition of the initial surface uses a similar method to that employed by the first brush, again using the Frenet formulas as shown in Fig. 2. Fig. 6(a) shows how the width of the initial surface is adjusted by moving the wand in the *x* and *y*-directions. The curvature in the *y*-direction is determined by the rotation α of the wand about the *x*-axis, and the curvature in the *x*-direction is determined by the rotation β of the wand about the *y*-axis, as shown in Fig. 6(b).

After the initial surface has been created, the designer moves the brush to the intended position and then stamps it. The surface is stored in the grid. Some rendered surfaces are shown in Fig. 7. This brush type is useful to draw shapes with simple geometries such as planes, cylinders, or spheres.

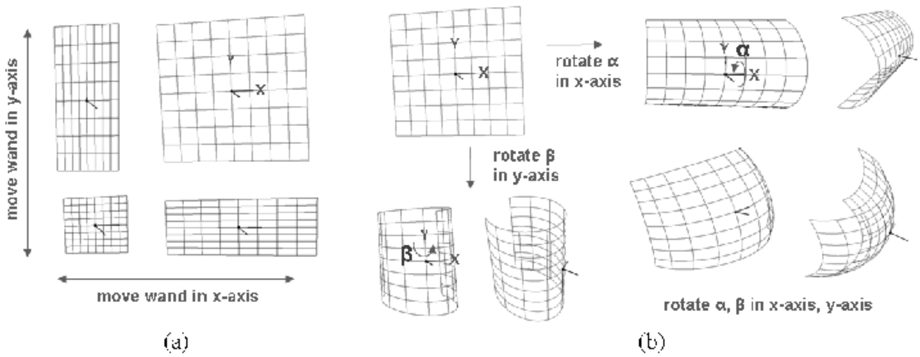


Fig. 6. Surface brush type

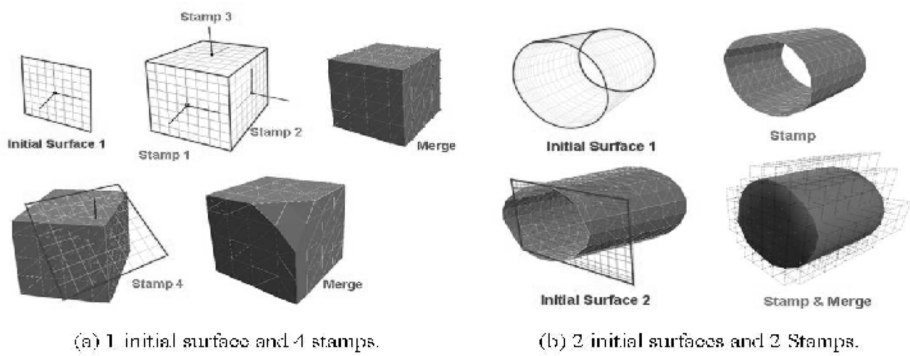


Fig. 7. Drawing objects with a surface brush type

Three kinds of brush types have different advantage for use. So, it is useful to change brush types as occasion demands. Three kinds of brush types can be selected by the rotation of wand about the z-axis. A designer can change various brush types easily similar to changing initial curved lines and surfaces.

3 Surface Modeling with Grid-Based Data Structure

This paper suggests a modeling tool for conceptual surface design that draws a desired surface according to the distribution and the combination patterns of input data among grids, which is similar to marching cubes [12]. In the method using grids a surface is drawn based on the 3D input by applying to the spatial sketching a method for digitalizing a plane sketch into grid unit inputs. This has an effect on reducing the position errors caused by the repetitive input with additional stroke to modify a surface.

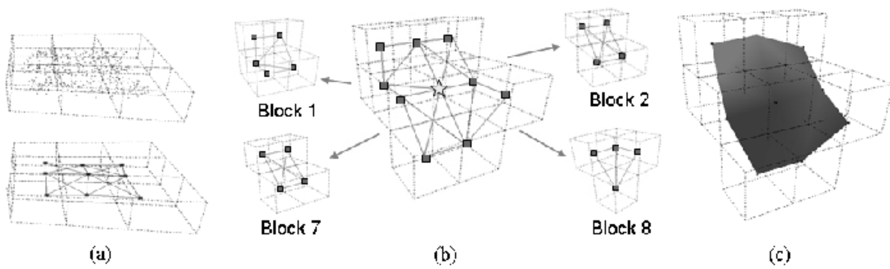


Fig. 8. The cubes with internal vertex and surface connected with internal vertices

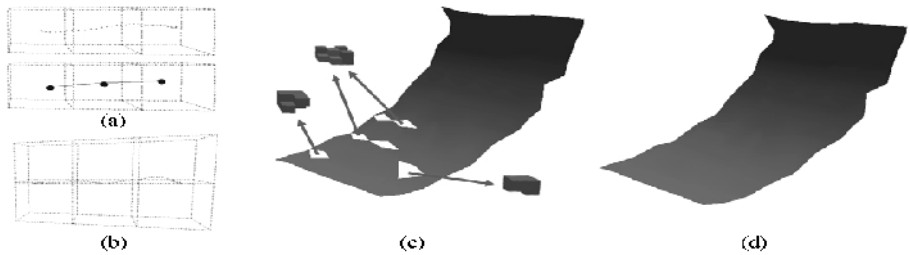


Fig. 9. A proper and improper input types in grids and a median surface regulation

As a designer moves the wand in drawing space, many points are inputted into grids from the input device and one internal vertex is calculated by a combination of input points in each grid and can be used as the real vertex of a real surface, as shown in Fig. 8(a). According to the relationship between the central grid and neighboring grids, a pre-defined pattern is selected as shown in Fig. 8(b). A surface is rendered by connecting the internal vertices with pre-defined patterns as shown in Fig. 8(c).

In a case in which an input stroke passes through grids that are simply connected as in Fig. 9(a), the intended curve can be easily drawn. However, as shown in Fig. 9(b), the input data can pass through the boundaries between the grids, or through the surrounding grids. In this case, the grids have unusual relationship with neighborhood grids, and so parts of surface with improper pattern are not drawn, as shown in Fig. 9(c). In order to minimize ambiguity by determining the valid grids, we regulate invalid parts of surface with the thinning algorithm [13]. However, the thinning algorithm erodes surface boundary well, which we want to preserve. We have extended thinning algorithm to preserve boundary and to avoid formation of a hole.

A designer sketches a model with multiple strokes with various wand brushes. Several surfaces with various shapes are inputted and merged to make a desired model. To connect and merge two surfaces, the modeling algorithm needs the relationships between both surfaces. So, we use the boundary contours for the regulation of grid based surface. When a surface is drawn by a stroke as shown in Fig. 10(a), the boundary cubes of outer boundary are searched as shown in Fig. 10(b). The boundary cubes are connected with near boundary cubes and made as a contour as shown in

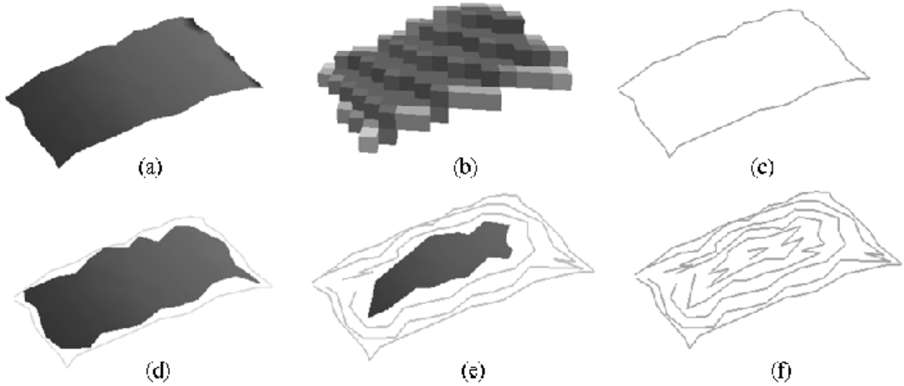


Fig. 10. Boundary contours of a surface

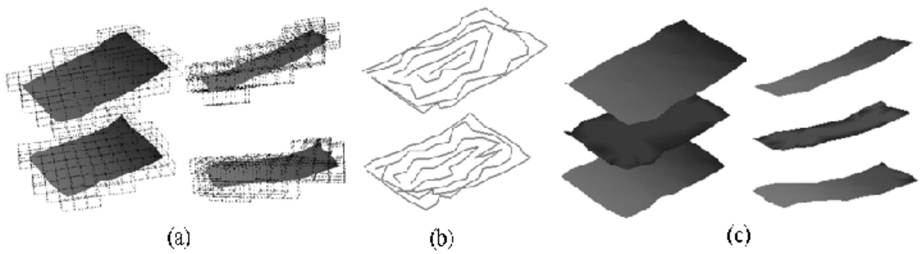


Fig. 11. Surface merging with parallel surfaces

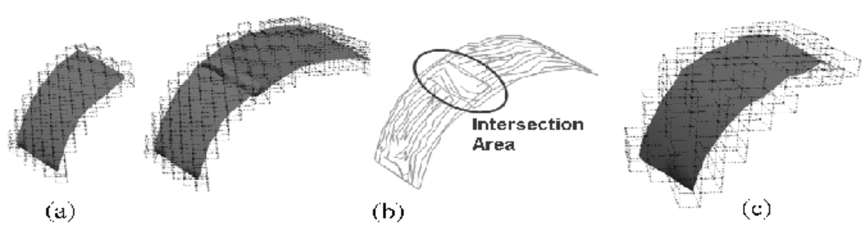


Fig. 12. Surface merging with intersected surfaces

Fig. 10(c). And the selected boundary cubes of the contour are removed as shown in Fig. 10(d). We make contours from the outer boundary one by one as shown Fig. 10(e) and (f).

There are two kinds of merging method in this paper. First method is merging two parallel surfaces. When a new surface is drawn, the new surface searches parallel surface with no intersection as shown in Fig. 11(a). If the parallel surface is found out, both surfaces make the boundary contours as shown in Fig. 11(b) and calculate the middle contours with both boundary contours. And a new surface is created with middle boundary contours as shown in Fig. 11(c). Second method is merging two surfaces that meet together. When a new surface meet existing surface as shown Fig. 12(a), both surface make the boundary contours and search intersection cubes as shown in Fig. 12(b). The intersection area is regulated with the thinning algorithm and the rule for removing protruding cubes as shown in Fig. 12(c).

4 Conclusions

This research adopted the idea that when a designer sketches a model, he/she uses various wand brush styles to draw it. Each brush style has advantages of drawing proper surfaces. It is useful for a designer to have several types of brush available as shown Fig. 13. We show how the style of brush can be selected by altering the posture of a 3D wand. This allows a designer to work in an intuitive way without being interrupted by complicated menus.

Our spatial sketch system simplifies the input data from different brush types by dividing the drawing space into a grid and makes an intended surface by regulating invalid cube patterns. We also use the boundary contours for surface merging.

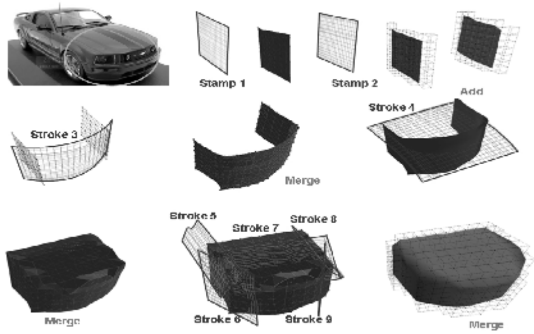


Fig. 13. Combination of surfaces with different brush types

Acknowledgments

This work is funded by KOSEF (Korea Science and Engineering Foundation). (No.R01-2007-000-20283-0)

References

1. McDonnell, K., Qin, H., Wlodarczyk, R.: Virtual clay: A real-time sculpting system with haptic interface. In: Proceedings of ACM Symposium on Interactive 3D Graphics, pp. 179–190 (2001)
2. Keefe, D., Zeleznik, R., Laidlaw, D.: Drawing on Air: Input Techniques for Controlled 3D Line Illustration. IEEE Transactions on Visualization and Computer Graphics 13, 1067–1081 (2007)

3. Fiorentino, M., Monno, G., Renzulli, P., Uva, A.: 3D Pointing in Virtual Reality: Experimental Study. In: XIII ADM–XV INGEFRAF International Conference on Tools and Methods Evolution in Engineering Design (2003)
4. Bowman, D., Johnson, D., Hodges, L.: Testbed Evaluation of Virtual Environment Interaction Techniques. Presence: Teleoperators and Virtual Environments 10, 75–95 (2001)
5. Schkolne, S., Michael Pruett, M., Schroder, P.: Surface Drawing: Creating Organic 3D Shapes with the Hand and Tangible Tools. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 261–268 (2001)
6. Keefe, D., Acevedo, D., Moscovich, T., Laidlaw, D.H., LaViola, J.: CavePainting: A Fully Immersive 3D Artistic Medium and Interactive Experience. In: Proceedings of ACM Symposium on Interactive 3D Graphics, pp. 85–93 (2001)
7. Brody, B., Harman, C.: Painting Space with BLUI. In: Proceedings of ACM SIGGRAPH Abstracts and Applications, p. 242 (2000)
8. Igarashi, T., Matsuoka, S., Tanaka, H.: Teddy: A Sketching Interface for 3D Freeform Design. In: Proceedings of ACM SIGGRAPH, pp. 409–416 (1999)
9. Nealen, A., Igarashi, T., Sorkine, O., Alexa, M.: FiberMesh: Designing Freeform Surfaces with 3D Curves. ACM Transactions on Graphics 26 (2007)
10. Kara, L., D'Eramo, C., Shimada, K.: Pen-Based Styling Design of 3D Geometry Using Concept Sketches and Template Models. In: Proceedings of ACM Solid and Physical Modeling, pp. 149–160 (2004)
11. Fleisch, T., Rechel, F., Santos, P., Stork, A.: Constraint stroke-based oversketching for 3D curves. In: Proceedings of the Eurographics Workshop on Sketch-Based Interfaces and Modeling, pp. 161–166 (2004)
12. Lorenson, E., Cline, E.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In: Proceedings of ACM SIGGRAPH, pp. 163–169 (1987)
13. Borgefors, G., Nystrom, I., Baja, G.S.D.: Computing skeletons in three dimensions. Pattern Recognition 32, 1225–1236 (1999)