3DBrushVR: From Virtual Reality Primitives to Complex Manifold Objects

Yuzhen Zhu*[1], Xiangjun Tang†[1], Jing Zhang‡[1], Ye Pan§[2], Jingjing Shen¶[3], and Xiaogang Jin‖**[1]

[1]State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China
[2]Shanghai Jiaotong University, Shanghai, China
[3]Mixed Reality AI Lab – Cambridge, Cambridge Microsoft Research Ltd, Cambridge CB1 2FB, United Kingdom

ABSTRACT

SurfaceBrush and Brush2Model are two systems which enable users to create 3D objects intuitively using a hand-held controller in virtual reality (VR). These state-of-the-art methods either start modeling from dense collections of stroke ribbons drawn by professional artists, or from the most basic point skeletons, line skeletons, and polygon skeletons. Thus, it is very challenging for novices and amateurs to design complex models efficiently. We propose 3DBrushVR, a novel VR modeling tool that uses volume skeleton-based convolution surfaces. It enables the user to draw with arbitrarily shaped brushes and generate 3D manifold objects by fusing the brushed primitives. Unlike existing VR drawing and modeling tools, our approach can directly take some common but complex objects as primitives, and assemble them using implicit surfaces, thus providing a more flexible and powerful modeling ability. To achieve real-time performance, we introduce a new GPU-based method to calculate the volume fields of the resulting convolution surfaces. We also introduce several specially designed time-varying shaders to render the designed model for a better and more appealing modeling experience. We demonstrate the usability and modeling ability of our 3DBrushVR interface by comparing it with the state-of-the-art methods in an observational study. Experimental results further validate the effectiveness and flexibility of our approach.

Index Terms: Computing methodologies—Computer graphics—Graphics systems and interfaces—Virtual reality; Computing methodologies—Computer graphics—Shape modeling—Volumetric models

1 INTRODUCTION

Professional artists leverage various brushes of pre-defined complex shapes in traditional 2D graphic design software like PhotoShop [2]. We extend this pre-defined complex brush concept from 2D brushes to 3D brushes and propose a 3D modeling method that enables the user to draw directly in 3D space using such complex 3D brushes. By combining our method with VR technology, we introduce **3DBrushVR** to enable users without enough drawing skills to easily create a unique model. The modeling result will be a single watertight surface 3D model.

Google has released a commercial software called Tilt Brush [12]. This VR app allows the user to paint in the virtual 3D space easily using 3D brushes. However, drawing with Tilt Brush, the user ends up with lots of disconnected sheet-like surfaces instead of a watertight 3D model. Based on Google's Tilt Brush, SurfaceBrush [26] tries to convert the resulting strokes into a manifold model. However, this system is more suitable for experienced artists because it requires plenty of "good" strokes to form a surface model. Sketching the intended strokes precisely in mid-air can be quite challenging [4] for users with limited experience in drawing and modeling.

Brush2Model [35] uses skeleton-based convolution surfaces as 3D brushes. As a skeleton-based implicit surface, convolution surfaces are widely used in sketch-based modeling because their embedded skeletons can accelerate the process of modeling, and their superposition property is suitable for compositing complex shapes. Brush2Model allows the user to draw with point skeleton-based surface brushes, line skeleton-based surface brushes, and polygon-based surface brushes in a VR environment. However, it is difficult for novice users to create a detailed complex model using only those basic brushes, which prevents them from easily creating intended models. Furthermore, the interactions in Brush2Model are quite limited, such as the user's inability to undo operations.

Motivated by the solutions of assembly-based modeling [8] and Brush2Model, we propose volume-based 3D brushes which allow the user to design complex shapes by assembling primitives in the virtual environment. The user is able to use arbitrary 3D meshes as 3D brushes, place them in the drawing space, erase the unwanted parts, add simple primitives on top of them and combine them into a single watertight model. The user can export the completed model in a standard format for subsequent processing like fabrication. To allow the user to create a manifold model by fusing different shaped brushes, *one key challenge is to merge various complex primitives into one smooth object automatically in real time*. In our system, we leverage convolution surfaces to blend the closely-drawn model parts into a closed and smooth surface, with GPU acceleration.

To summarize, we present a novel immersive modeling method based on convolution surfaces and develop a software package named **3DBrushVR** based on game engine Unity3D [31]. Our main contributions are:

A new real-time algorithm to convert arbitrary 3D model meshes into smooth convolution surfaces on the GPU. We calculate the volume skeleton-based convolution surface by computing the potential value for each vertex in the volume grid in real time through a parallel mesh voxelization step.

An interactive virtual 3D brush-based modeling system that supports volume skeleton-based convolution surface brushes. Through a user-friendly interface, the system allows both novices and experts to create watertight complex free-form 3D models. We create an erase function that uses a simple point brush with a negative field to reduce the implicit volume, and it is seamlessly integrated into our system in the context of convolution surface modeling. To display the drawn models, the system also provides several materials with

*e-mail: aelinuial@163.com
†e-mail: fcsx1tf@163.com
‡e-mail: jing$_z$99@163.*com*
§e-mail: whitneypanye@sjtu.edu.cn
¶e-mail: jshen2036@gmail.com
‖e-mail: jin@cad.zju.edu.cn
**Corresponding author

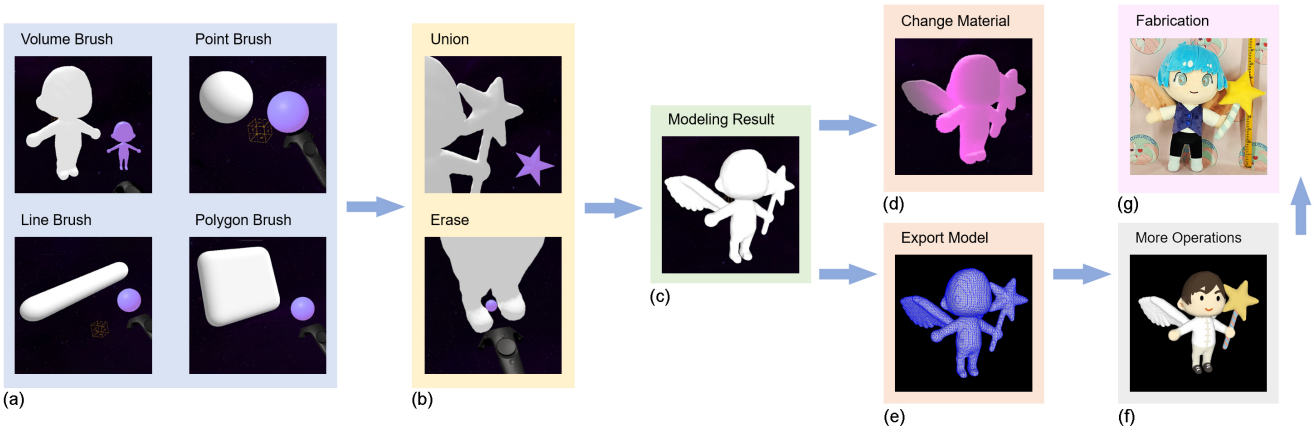

Figure 1: System flowchart. (a) The example shapes created using the four types of skeleton-based brushes in our system. For line or polygon skeleton-based brush, the user only needs to specify the points defining the line or polygon. The field potential value for the resulting surface will be changed by the brushes and be calculated by the GPU automatically. Marching Cubes algorithm is used to extract the iso-surface. (b) While drawing, the shapes of different brushes will blend smoothly, and the user can erase any bits easily. (c) The created model result. After drawing, the user can (d) change the materials and (e) export the model to a standard format. The user can (f) color the model in other professional modeling systems and (g) fabricate it into a plush toy.

appealing visual effects that use time-varying shaders.

2 Previous Work

3D modeling in virtual reality is an inspiring design option, the head-mounted display can simplify the problem of manipulating and understanding 3D models [7]. Some virtual reality modeling systems require the user to do sketch-based modeling. For example, early systems like Teddy [16], FiberMesh [23] and ShapeShop [28] ask the user to draw several 2D freeform strokes interactively and the systems automatically construct plausible 3D polygon surfaces. Surface Drawing [27] allows users to create 3D shapes through repeated marking, just like in traditional painting. FreeDrawer [32], Lift-Off [17] and CASSIE [33] can convert the 3D curve networks created by the user to a model surface. Another similar type of system also includes SurfaceBrush [26], which can create models with closed surfaces using the strokes drawn by the user in Tilt Brush [12]. AdaptiBrush [25] improves SurfaceBrush so that users can draw strokes more comfortably. In addition there is a very popular modeling tool known as Gravity Sketch [29], which has a variety of diverse drawing and modeling methods, supports real-time collaboration, and can be modeled across devices

All of the above systems can create good model results, however, they are not suitable for the novices without enough drawing and modeling experience. It is difficult for those users to sculpt model details and to imagine how to use simple strokes to indicate a model surface. To help novices create model, researchers have proposed modeling by example system [10]. These modeling systems allow users to select existing model components from a database and combine them together to build new models. However, this method, as well as the previously mentioned methods, has the disadvantage that it is difficult to generate watertight models with closed and smooth surfaces. Although many researchers have continued working on the topic of using mesh components to create a new model [1,8,9,13,24], few have come up with new ideas about smooth model blending.

To create watertight models with closed and smooth surfaces, convolution surfaces are introduced into skeleton-based modeling [5, 6, 30]. A convolution surface is an iso-surface embedded in a scalar field by convolving a geometric skeleton with a filter kernel function [6,22]. Closed-form solutions are the most efficient way to reduce the considerable computation of convolutional integrals. Besides spherical implicit functions [3], researchers have raised the computation methods for creating line skeletons [18, 19, 34], cylindrical skeletons [5, 23, 30] and triangle skeletons [15, 23]. Inspired by their work, Zhu et al. created a 3D modeling system called Brush2Model [35], which allows the user to draw models directly with point, line and skeleton-based convolution surface brushes.

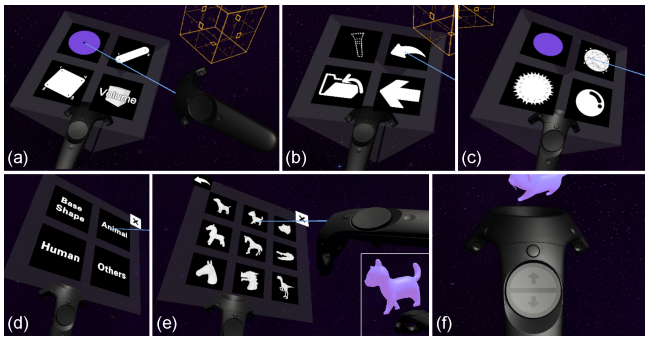

Figure 2: User interface. Users use the ray emitted from the right VR handle to choose the options on the panel. (a) Four types of skeleton-based brushes. (b) Erase, undo, save, and return functions. (c) Material options. (d) Volume skeleton-based brush library. (e) A cat brush selected from the library. (f) The brush size is adjustable by the up/down arrows on the right VR handle's panel.

3 Workflow

3DBrushVR is an easy-to-use modeling system in the VR environment. The user can use it to create smooth surfaces with arbitrary 3D skeleton-based brushes. Our work is inspired by Tilt brush's 3D brushes [12] and Brush2Model's skeleton-based modeling [35]. To reduce the difficulties that the user encounters when creating complex models, volume skeleton-based brushes and user-friendly interfaces are introduced to assist the drawing process. Fig. 1 shows the modeling flowchart of our system.

The user can choose to start drawing the model either from a blank canvas or on top of a particular model. For the latter, the user can import any model from our library or their own model (in a well-known 3D format) using our interface. Our system will automatically convert the loaded 3D model into a convolution surface, and the user can use any skeleton-based brushes to draw (e.g., a point, line,

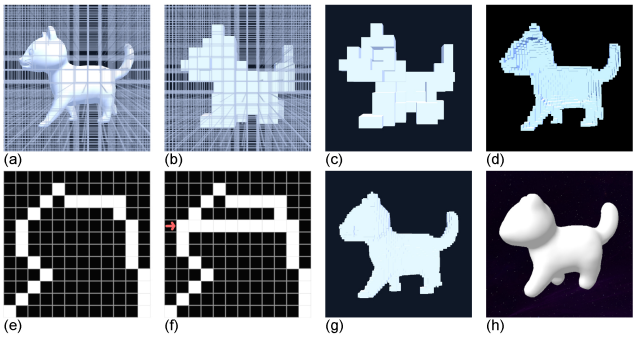

Figure 3: Triangle mesh voxelization process. (a) Divide the drawing space into small cubes. (b) For the triangular faces on the model mesh, compute their intersecting cubes and fill these cubes. This will produce a voxel model shell. (c) A cross section of this model shell. (d) The cross section with higher resolution. (e) A slice of the model shell on arbitrary plane. (f) Apply the scanline method to successively fill the cubes inside the model in each line. (g) The resulting solid voxelized model. (h) Converted into a convolution surface, with a smooth 3D convolution kernel applied.

polygon or volume) on top of it.

Fig. 2 shows our main user interface. While drawing, the user can use the ray emitted from the right VR handle and the option panels attached to the left VR handle to choose different types and shapes of volume skeleton-based brushes. Once the position, size and angle of the brush are set, the user can press the trigger button on the right VR handle to draw the corresponding model. The drawing result of the volume skeleton-based brush will be seamlessly merged into the resulting convolution surface model automatically.

To allow the user to draw freely and creatively, we also provide three basic skeleton-based brushes. With our friendly user interface, the user can switch brush types freely. The user can use the point skeleton-based brush to draw a sphere, the line skeleton-based brush with two defining points to create a cylinder, and the polygon skeleton-based brush with four defining points to create a polyhedron. Our system also provides erase and undo functions. With all these functions, creating complicated models becomes a much easier task for novice users. After finishing the drawing process, the user can choose different materials for his models. We provide four default materials for displaying the drawn model with appealing effects. Two of the materials are time-varying, which allow the model to change color and vertices positions with time. Finally, to enable the user to use the created model in other applications such as 3D printing, our system supports exporting models in a well-known 3D format.

4 Implementation

In this section, we describe the mathematical model of our method, and elaborate on the implementation details of our 3DBrushVR.

4.1 Convolution Surface

A convolution surface is an iso-surface in Euclidean space implicitly defined by convolving a kernel function on geometric skeletons [6]. Denote $\mathbf{P}(x,y,z)$ as a space point in R^3, and $g : R^3 \to R$ as the geometry function which represents a modeling skeleton V:

$$g(\mathbf{P}) = \begin{cases} 1, & \mathbf{P} \in \text{skeleton } V \\ 0, & \text{otherwise} \end{cases}. \tag{1}$$

Let $f : R^3 \to R$ be a potential function, and $\mathbf{Q}$ a point in the skeleton, then the total field F contributed by the skeleton at a point $\mathbf{P}$ can be calculated as the convolution of two functions f and g [6, 22]:

$$F(\mathbf{P}) = \int_V g(\mathbf{Q}) f(\mathbf{P}-\mathbf{Q}) dV = (f \otimes g)(\mathbf{P}). \tag{2}$$

Here, f is also called the convolution kernel.

Superposition is a unique property for convolution surfaces, and it ensures that summing the convolution surfaces generated by two separate skeletons and calculating the convolution surface directly using their combined skeleton will generate the same result [20]. The superposition feature enables convolution surfaces to overcome the limitation of distance surfaces, where the surfaces usually have bulges and creases. Using this property, we can calculate the convolution surface S in the general form as follows:

$$S = \left\{ \mathbf{P} \mid \sum_{i=1}^{n} \lambda_i F_i(\mathbf{P}) - T = 0 \right\}. \tag{3}$$

In the above formula, F_i and λ_i are the field function and the field contribution weight of the i-th skeleton segment, respectively, and T is the threshold of the iso-surface. With a fixed kernel function, the superposition property guarantees that the resulting surface only depends on the shape of skeleton, which enables us to compute the resulting field in parallel on the GPU by subdividing a skeleton into many sub-skeletons.

In our implementation, we split the drawing space into a cubic grid of size $128 \times 128 \times 128$. We give each vertex of the cubic grid a default potential value 0. While drawing with the skeleton-based brushes, the potential values of these vertices are updated accordingly. We use GPU to accelerate the potential value updates for each vertex in parallel. The Marching Cubes algorithm [21] is used to extract the surfaces from the cubic grid, and the extracted surfaces are rendered and displayed to users in real time. We also implement the Marching Cubes algorithm on GPU.

4.2 Volume Skeleton

We provide volume skeleton to help the user to draw complex models easily. The volume skeleton can be used in two scenarios. Firstly, before any drawing, the user can load arbitrary models into our system. Our system will automatically convert a mesh model into a convolution surface, so that the user can draw with skeleton-based brushes on top of it. Secondly, the user can use the volume skeleton-based brushes in our brush library and perform example-based modeling, which means the user can create models by seamlessly combining different brush shapes.

Converting a mesh into a convolution surface requires two steps. We first voxelize the triangle mesh and update the potential values of the cubic grid vertices mentioned above. Then we smooth the model surface by using a 3D convolution kernel.

4.2.1 Triangle Mesh Voxelization

A voxel is a single cube in a 3D lattice, and mesh voxelization means using the small cubes to fit the shape of the original mesh.

A voxel model is a bounded 3D grid. We use the same cubic grid that holds the potential values for the voxel model. Given a triangle mesh, the voxelization task is to decide which cubes in the grid are solid (i.e., inside the mesh). We use a two-step process. First, we solidify a shell representing the model surface. And second, we fill the model with a method like the scanline rendering algorithm.

We calculate the model shell by iterating through all triangles of the mesh. For each triangle, we find its intersections with the cubes and mark these cubes as solid, which means the potential values at the vertices of these cubes will be set to 1 (in our implementation, the threshold for extracting the model surface is 0.5). After this step, we will get a tubular model with two surfaces. The second step is to fill the inside of this model shell. Take a slice of the cubic

grid on arbitrary plane as an example (Fig. 3(e)), like the scanline rendering algorithm, we use a line to scan the grid slice row by row. In each row, the scan line finds the intersecting cubes that have been marked as solid in the first step in order from left to right. At each intersection, we determine whether the current scan line is entering or leaving the model, and set the state accordingly to decide whether to update the next few cubes as solid or not. The same scanline process is repeated for each slice of the grid.

While determining the intersection of the mesh triangles and the cubes, each triangle can be calculated independently. While using the scanline method to fill the model shell, each row can also be calculated independently. Therefore, we use compute shaders in Unity3D to accelerate all these calculations in parallel with the GPU.

4.2.2 Surface Smoothing

The system then smooths the model surface automatically. After the voxelization of the mesh, the model is made of a series of small cubes, thus the surface is grainy and uneven. We smooth the surface with a 3D convolution kernel, i.e., the potential value at each vertex of the cube is determined by the surrounding neighboring vertices. In our implementation, the default influential radius is 1, and the potential value at a specific vertex is the mean value of the 27 potential values around and including it ($3 \times 3 \times 3$). To this end, we get a model with a smooth and closed convolution surface.

4.3 Basic Skeleton

Inspired by the previous work, we also provide point skeleton-based brush, line skeleton-based brush and polygon skeleton-based brush to help users draw points, cylinders and polyhedrons easier. Compared to Brush2Model, we provide a much easier user interface for using those brushes. In our system, only a single hand is needed for creating the model after the user selects the skeleton type, all the parameters can be changed within one VR handle. The convenience of our system greatly increases the user's engagement.

5 Results

Using the above methods, we create a unique and cohesive virtual reality modeling application that allows the user to draw and display models easily. We use Unity3D game engine and SteamVR SDK to implement 3DBrushVR.

Figure 4: Example models drawn with our system. The models are set materials and rendered with Substance Painter.

Table 1: The number of triangles for examples in Fig. 4.

	Fig. 4(a)	Fig. 4(b)	Fig. 4(c)	Fig. 4(d)
Triangles	59,788	69,176	78,784	112,256

We tested our 3DBrushVR on a personal computer (CPU: AMD Ryzen 7 3800X, RAM: 16G, GPU: NVIDIA GeForce GTX 1660 SUPER). Fig. 4 and Fig. 5 show the models we created using the real-time system in VR. The mesh statistics of these some models are shown in Table 1.

We use GPU to accelerate each step of our algorithm. The drawing with a volume skeleton-based brush is the most computationally intensive part, which includes the voxelization of the loaded mesh or

Figure 5: More models created by users using our 3DBrushVR system.

Table 2: The calculation time (voxelization step + smoothing step + mesh extraction step via the Marching Cubes algorithm) for the volume skeleton-based brushes used in drawing the Sagittarius. The grid size is $128 \times 128 \times 128$.

	Horse	Human Body	Wings
Triangles	24,780	24,540	4,238
Calculation Time	23ms	23ms	21ms

brush template, field computation using 3D convolution kernels, and mesh extraction via the Marching Cubes algorithm. Table 2 shows the computation time for the volume skeleton-based brushes used in drawing the Sagittarius (Fig. 6), with a grid size $128 \times 128 \times 128$. Note that this computation happens only once after the user finished a volume skeleton-based brush drawing step. With our GPU acceleration, we are able to compute at 40 - 50fps with each volume skeleton-based brush. For simple skeleton-based brushes like point, line, and polygon, we are able to run at 90fps. Note that comparatively it took 52s to reconstruct a model with 20k vertices in SurfaceBrush.

6 User Study

In order to compare our 3DBrushVR with previous immersive modeling systems, we set up a user study and chose SurfaceBrush and Brush2Model as the representatives which can produce closed-faced models.

6.1 Evaluated Systems

The three systems we have tested:

3DBrushVR: our novel immersive modeling system.

Brush2Model: a modeling system which uses convolution surface.

SurfaceBrush: a system that uses the strokes created by Tilt Brush as inputs and automatically turn these strokes into a surface-closed model using their matching algorithms.

We evaluate and compare the user interface, model drawing experience, results of the model generation, and the display of the models of these systems. For SurfaceBrush, the user's drawing experience refers to the experience of drawing strokes with Tilt Brush.

6.2 Experimental Design and Procedure

Our user study includes 20 participants. The participants either have no drawing, modeling and VR experience or little experience, and none of them are experts in these domains.

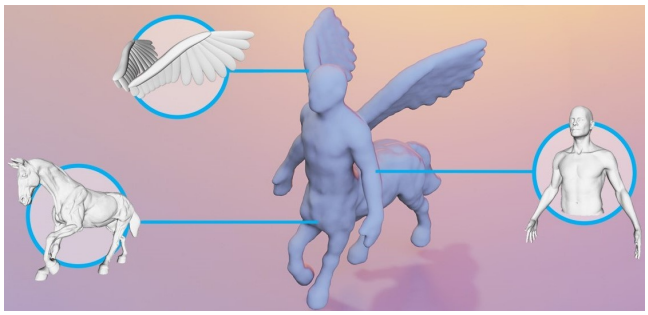

Figure 6: A Sagittarius model created by using three volume skeleton-based brushes and the erase function.

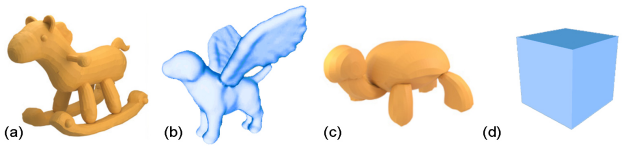

Figure 7: Our four reference images. Images (a) and (c) are from Rosales et al.'s paper [26].

For each of the three systems, the subject first had a 10-minutes pre-task preparation to learn how to use the system. Then, each subject was given a reference image of a model and asked to finish the tasks of creating such a model in the three systems, respectively.

Fig. 7 shows our reference images. In our 3DBrushVR, the user can use the horse brush, dog brush, and the wing brush to help finish the task shown in Fig. 7(a) and Fig. 7(b).

The subject was given maximum 20 minutes to finish each task. After accomplishing all the tasks, the subject was firstly asked to finish a *NASA-TLX Questionnaire* [14] for each system, and then a *Scoring Questionnaire* on whether these systems are easy to use and appealing, 1 to 5 for totally disagree to totally agree.

6.3 Study Result

Figure 8: (a) Task results with our 3DBrushVR system. (b) Task results with Brush2Model.

Fig. 8 and Fig. 9 separately shows the models created by the subjects with 3DBrushVR, Brush2Model and SurfaceBrush. Most of the subjects (P = 85%) agreed that the models created by our system are the most exquisite ones. For SurfaceBrush, when the user draws relatively simple strokes like Fig. 9(a), the reconstruction will get a good result. However, if the model is drawn with many complex strokes, the reconstruction progress may fail, which is shown in Fig. 9(b).

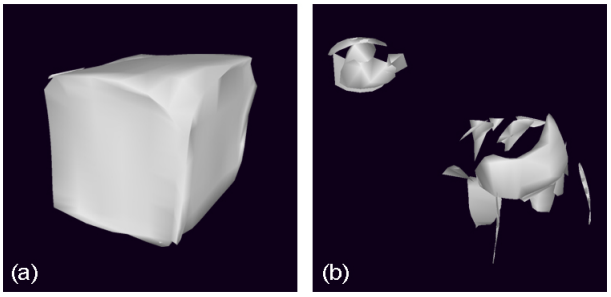

Figure 9: Task results with SurfaceBrush. This system is designed for professionals. The subjects in the study could only draw relatively simple strokes, which works for cube model, but failed on others which require complex strokes.

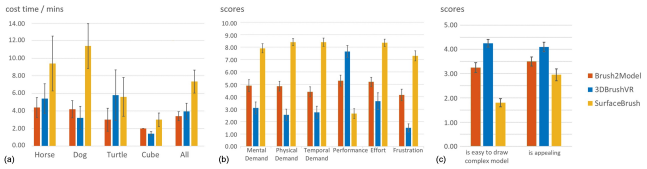

Figure 10: User study results. (a) For each reference image, the average time cost for the three tasks. Note that the timings shown for SurfaceBrush do not contain the post processing time. (b) Mean values of NASA-TLX's six dimensions. For performance, the higher score the better. For others, the lower the better. (c) The result of the Scoring Questionnaire. Higher score means more agreement.

Fig. 10(a) clearly shows that finishing the tasks with 3DBrushVR (M = 3.95, SD = 4.11, blue) costs much less time than with SurfaceBrush (M = 7.35, SD = 5.83, yellow). Finishing the tasks with Brush2Model (M = 3.4, SD = 2.26, red) costs almost the same time as with 3DBrushVR (p = 0.60).

From Fig. 10(b), we can conclude the following. First, the mental demand, physical demand, and temporal demand values of drawing with 3DBrushVR are all lower than the other two systems, which indicates that our 3DBrushVR is less time-consuming and laborious than others. The scores of performance (the higher the better) and effort (the lower the better) show that with 3DBrushVR, the subjects can create the best results with minimal efforts. Meanwhile, the frustration of 3DBrushVR (M = 1.50) is much lower than Brush2Model (M = 4.15) and SurfaceBrush (M = 7.30), which also shows that the subjects are more willing to use our system for modeling work in future. Fig. 10(c) shows that our 3DBrushVR received the highest mean scores for both usability and appealingness. A repeated measures ANOVA [11] and a post-hoc test were used to prove that all the scores significantly differ from each other (p <0.05). These results show that our system does have better usability and attraction.

All the subjects agreed that 3DBrushVR is an interesting concept design tool. We exported one of the subjects' free-drawing models and fabricated it into a plush toy (see Fig. 11).

7 Future Work

Our system still has its limitations. First, in our system, the post-drawing undo function is still limited. For line skeletons and polygon skeletons, it is able to undo several drawing steps, but for continuous point skeletons and volume skeletons, it can only undo one step because of the memory limitations. We plan to extend the undo function and add skeleton editing functions in the future.

Second, because the drawing space has a limited resolution and we use a 3D convolution kernel to smooth the voxel model, the level of details are limited and are largely affected by the canvas resolution

Figure 11: We use 3DBrushVR to design the toy. (a)The modeling result. (b)The color design draft. (c)The 3D print results. (d)The plush toy.

and the radius of the convolution kernel. If the brush size is too small compared to the cube in the grid, the drawing result will be unsatisfactory. If we increase the canvas resolution for better model details, the system will run substantially slower. With a resolution of $256 \times 256 \times 256$, the performance of our system will drop to 5 fps while drawing volume skeletons. In the future, we plan to improve the volume skeleton-based brushes by automatically adjusting the size of the convolution kernel with reference to the resolution of the grid and the size of the brush model, and further accelerate the computation by leveraging an octree instead of a uniform grid.

8 Conclusion

We present 3DBrushVR, a creative virtual reality modeling system that allows novice users to create their own models easily using volume skeleton-based brushes. Users can switch different brushes to design concept models. With the help of our convolution surface-based volume skeleton-based brushes, the erase function using a point skeleton brush and the undo function, creating a complicated model becomes much easier. After drawing, users can use different materials to display the drawn models. Finally, designed models can be exported in standard formats to be further used in other professional 3D modeling software. Our experiments and user study show that our 3DBrushVR has major advantages over other similar modeling systems in terms of usability, efficiency and satisfaction of the modeling results.

Acknowledgments

This work was supported by Key R&D Program of Zhejiang (No. 2022C03126) .

References

[1] Adobe. Medium by adobe. `https://www.adobe.com/ca/products/medium.html`, 2021.

[2] Adobe Inc. Adobe photoshop. `https://www.adobe.com/products/photoshop.html`, 1990.

[3] I. A. Alexe, V. Gaildrat, and L. Barthe. Interactive modelling from sketches using spherical implicit functions. In *Proc. ICCGV*, pp. 25–34. ACM, New York, NY, USA, 2004.

[4] R. Arora, R. H. Kazi, F. Anderson, T. Grossman, K. Singh, and G. Fitzmaurice. Experimental evaluation of sketching on surfaces in VR. In *Proc. CHI*, p. 5643–5654. ACM, New York, NY, USA, 2017.

[5] A. Bernhardt, A. Pihuit, M. Cani, and L. Barthe. Matisse: Painting 2d regions for modeling free-form shapes. In *5th Eurographics Workshop on Sketch-Based Interfaces and Modeling*, pp. 57–64. Eurographics Association, Goslar, DEU, 2008.

[6] J. Bloomenthal and K. Shoemake. Convolution surfaces. In *Proc. SIGGRAPH*, pp. 251–256. ACM, New York, NY, USA, 1991.

[7] J. Butterworth, A. Davidson, S. Hench, and M. Olano. 3dm: A three dimensional modeler using a head-mounted display. In *Proc. I3D*, pp. 135–138. ACM, New York, NY, USA, 1992.

[8] S. Chaudhuri, E. Kalogerakis, S. Giguere, and T. Funkhouser. AttribIt: Content creation with semantic attributes. *ACM Symposium on User Interface Software and Technology (UIST)*, p. 193–202, Oct. 2013.

[9] S. Chaudhuri and V. Koltun. Data-driven suggestions for creativity support in 3d modeling. *ACM Trans. Graph.*, 29(6):183, 2010.

[10] T. A. Funkhouser, M. M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. P. Dobkin. Modeling by example. *ACM Trans. Graph.*, 23(3):652–663, 2004.

[11] E. R. Girden. *ANOVA: Repeated measures*. Number 84. Sage, 1992.

[12] Google. Tilt brush. `www.tiltbrush.com`, 2016.

[13] X. Guo, J. Lin, K. Xu, S. Chaudhuri, and X. Jin. Customcut: On-demand extraction of customized 3d parts with 2d sketches. *Comput. Graph. Forum*, 35(5):89–100, 2016.

[14] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52, pp. 139–183. Elsevier, 1988.

[15] E. Hubert. Convolution surfaces based on polygons for infinite and compact support kernels. *Graph. Model.*, 74(1):1–13, 2012.

[16] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: a sketching interface for 3d freeform design. In *International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2006*, p. 11. ACM, New York, NY, USA, 2006.

[17] B. Jackson and D. F. Keefe. Lift-off: Using reference imagery and freehand sketching to create 3d models in vr. *IEEE Transactions on Visualization and Computer Graphics*, 22(4):1442–1451, 2016.

[18] X. Jin and C. Tai. Analytical methods for polynomial weighted convolution surfaces with various kernels. *Comput. Graph.*, 26(3):437–447, 2002.

[19] X. Jin and C. Tai. Convolution surfaces for arcs and quadratic curves with a varying kernel. *Vis. Comput.*, 18(8):530–546, 2002.

[20] X. Jin, C. Tai, J. Feng, and Q. Peng. Convolution surfaces for line skeletons with polynomial weight distributions. *J. Graphics, GPU, & Game Tools*, 6(3):17–28, 2001.

[21] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH*, 21:163–169, 08 1987.

[22] J. McCormack and A. Sherstyuk. Creating and rendering convolution surfaces. *Comput. Graph. Forum*, 17(2):113–120, 1998.

[23] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa. Fibermesh: designing freeform surfaces with 3d curves. *ACM Trans. Graph.*, 26(3):41, 2007.

[24] P. Ren, Y. Wang, M. Zhou, Z. Wu, P. Zhou, and J. Zhang. Data-driven modeling for chinese ancient architecture. *PRESENCE: Teleoperators and Virtual Environments*, 26(4):389–401, 2018.

[25] E. Rosales, C. Araújo, J. Rodriguez, N. Vining, D. Yoon, and A. Sheffer. Adaptibrush: adaptive general and predictable vr ribbon brush. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021.

[26] E. Rosales, J. Rodriguez, and A. Sheffer. Surfacebrush: from virtual reality drawings to manifold surfaces. *ACM Trans. Graph.*, 38(4):96:1–96:15, 2019.

[27] S. Schkolne, M. Pruett, and P. Schröder. 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.

[28] R. M. Schmidt and B. Wyvill. Shapeshop: Free-form 3d design with implicit solid modeling. In *Sketch-based Interfaces and Modeling*, pp. 287–312. Springer, 2011.

[29] G. Sketch. Gravity sketch. `www.gravitysketch.com`, 2017.

[30] C. Tai, H. Zhang, and J. C. Fong. Prototype modeling from sketched silhouettes based on convolution surfaces. *Comput. Graph. Forum*, 23(1):71–84, 2004.

[31] Unity Technologies. Unity. `https://unity.com`, 2005.

[32] G. Wesche and H. Seidel. Freedrawer: a free-form sketching system on the responsive workbench. In *Proc. VRST*, pp. 167–174. ACM, New York, NY, USA, 2001.

[33] E. Yu, R. Arora, T. Stanko, J. A. Bærentzen, K. Singh, and A. Bousseau. CASSIE: curve and surface sketching in immersive environments. In *Proc. CHI*, pp. 190:1–190:14. ACM, 2021.

[34] X. Zhu, X. Jin, S. Liu, and H. Zhao. Analytical solutions for sketch-based convolution surface modeling on the GPU. *Vis. Comput.*, 28(11):1115–1125, 2012.

[35] X. Zhu, L. Song, L. You, M. Zhu, X. Wang, and X. Jin. Brush2model: Convolution surface-based brushes for 3d modelling in head-mounted display-based virtual environments. *Comput. Animat. Virtual Worlds*, 28(3-4):e1761, 2017.