EasyVRModeling: Easily Create 3D Models by an Immersive VR System

ZHIYING FU, Shandong University, China
RUI XU, Shandong University, China
SHIQING XIN, Shandong University, China
SHUANGMIN CHEN, Qingdao University of Science and Technology, China
CHANGHE TU, Shandong University, China
CHENGLEI YANG, Shandong University, China
LIN LU, Shandong University, China

The latest innovations of VR make it possible to construct 3D models in a holographic immersive simulation environment. In this paper, we develop a user-friendly mid-air interactive modeling system named *EasyVR-Modeling*. We first prepare a dataset consisting of diverse components and precompute the discrete signed distance function (SDF) for each component. During the modeling phase, users can freely design complicated shapes with a pair of VR controllers. Based on the discrete SDF representation, any CSG-like operation (union, intersect, subtract) can be performed voxel-wise. Throughout the modeling process, we maintain one single dynamic SDF for the whole scene so that the zero-level set surface of the SDF exactly encodes the up-to-date constructed shape. Both SDF fusion and surface extraction are implemented via GPU to allow for smooth user experience. We asked 34 volunteers to create their favorite models using EasyVRModeling. With a simple training process for several minutes, most of them can create a fascinating shape or even a descriptive scene very quickly.

CCS Concepts: • **Human-centered computing** → **Virtual reality**; • **Computing methodologies** → *Modeling methodologies*; *Mesh models*.

Additional Key Words and Phrases: construct 3D models, virtual reality, CSG-like operation, user-friendly interaction, SDF fusion

ACM Reference Format:
Zhiying Fu, Rui Xu, Shiqing Xin, Shuangmin Chen, Changhe Tu, Chenglei Yang, and Lin Lu. 2022. EasyVR-Modeling: Easily Create 3D Models by an Immersive VR System. *Proc. ACM Comput. Graph. Interact. Tech.* 5, 1, Article 5773733 (May 2022), 14 pages. https://doi.org/10.1145/3522613

1 INTRODUCTION

Nowadays, VR experience halls can be seen everywhere in shopping malls, and VR/AR technology has gradually used in the education of primary and middle school students. An easy-to-use and WYSIWYG modeling system allows students to get started quickly and focus on creative design, which can greatly stimulate students' interest in VR/AR learning. However, building a modeling

*Corresponding author: S. Xin (xinshiqing@sdu.edu.cn).
Authors' addresses: Zhiying Fu, Shandong University, Qingdao, China, 266000, zhiyingf@mail.sdu.edu.cn; Rui Xu, Shandong University, Qingdao, China, xrvitd@163.com; Shiqing Xin, Shandong University, Qingdao, China, xinshiqing@sdu.edu.cn; Shuangmin Chen, Qingdao University of Science and Technology, Qingdao, China, csmqq@163.com; Changhe Tu, Shandong University, 30 Shuangqing Rd, Qingdao, China, chtu@sdu.edu.cn; Chenglei Yang, Shandong University, Jinan, China, 250000, chl_yang@sdu.edu.cn; Lin Lu, Shandong University, Qingdao, China, llu@sdu.edu.cn.

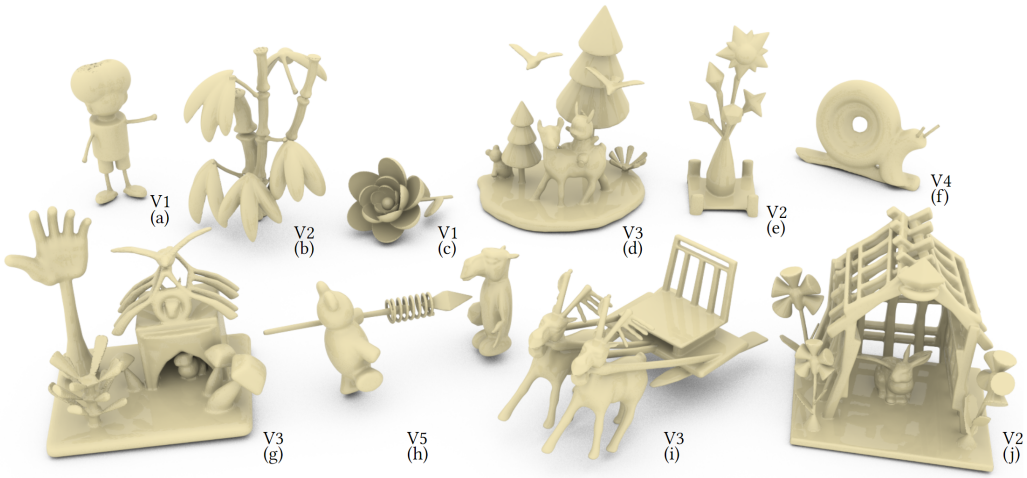


Fig. 1. Some appealing and realistic 3D examples created by volunteers (marked with V1, V2, V3, V4, V5). All the vivid characters and descriptive scenes are created based on a dataset of 155 basic components. (a) Little kid; V1; 8 min. (b) Bamboo; V2; 17 min. (c) Flower; V1; 15 min. (d) Pikachu Spring Tour; V3; 8 min. (e) Illustration; V2; 13 min. (f) Snail; V4; 7 min. (g) Rabbit House; V3; 18 min. (h) Animal war; V5; 5 min. (i) Merry Christmas; V3; 12 min. (j) Bunny Pavilion; V2; 9 min. (See the accompanying video for more details)

oriented VR modeling system is highly non-trivial. On one hand, it has to integrate many functionalities into one system - developing a user-friendly interface to support the diverse functionalities is not easy. On the other hand, different from simply arranging the layout of basic units, it is better to synthesize the components in real time. Therefore, in this paper, we develop a VR based mid-air interactive modeling system named *EasyVRModeling*, which is used to overcome the couple of difficulties.

Our system consists of a HTC Vive headset, two wireless controllers and two base stations. In the beginning, we prepare a large set of primitive components, and each component serves as a generator. We precompute the discrete SDF representation for each generator. The central idea is to maintain a global discrete SDF to facilitate real-time fusion of generators [Duan et al. 2020; Fedkiw and Osher 2002; Osher and Fedkiw 2006]. We denote the up-to-date shape of the work scene as A. During each step, users need to select a generator B from the given dataset, translate/scale/rotate it to a desirable positional and rotational status, and specify a kind of atomic operations, either $A \cup B$, or $A \cap B$, or $A - B$. In implementation, we use a voxelized representation of SDF, which enables instant voxel-wise fusion of two SDFs and voxel-wise surface extraction [Kobbelt et al. 2001; Lewiner et al. 2003; Lorensen and Cline 1987; Newman and Yi 2006].

To summarize, our contribution is twofold:

(1) We present an easy-to-operate VR based modeling system, with which users can create appealing and realistic shapes/scenes based on component synthesis; See some examples created by volunteers in Figure 1.
(2) We precompute the discrete SDF for each component in a diverse dataset. We also define a transformation mechanism to align a local component with the global working scene based on SDF representation, then perform 3D Boolean operations to synthesize components, and finally transform the synthesized result to a renderable mesh. All the operations can be parallelized, which is indispensable to ensure comfortable interactive experience.

2 RELATED WORK

At least two kinds of topics are very related to the theme of this paper, including VR based interaction and VR based 3D modeling.

2.1 Interaction in VR

VR has the capability to provide users with a sense of immersion and realism, but how to interact with the virtual environment pleasantly and comfortably is highly non-trivial [Jerald and Marks 2016]. There are mainly three interaction paradigms including VR controller direct control, VR extended menu, and gesture control. For example, CASSIE [Yu et al. 2021] proposed to use different buttons of the VR controller to realize various actions such as draw, delete, add, and save. Mendes et al. [Mendes et al. 2017] used gesture control and menu interaction for mid-air modeling. Arora [Arora 2020] presented a gesture-based animation method that directly controls the direction, diffusion and turbulence of the smoke simulation.

2.2 3D Modeling in VR

3D modeling is a central topic in the computer graphics field, and there is a large of literature on this topic. There are several common ways to virtually create a 3D model including sketching [Drey et al. 2020; Igarashi et al. 2006; Jackson and Keefe 2016; Liu et al. 2011; Lopes et al. 2011; Schmidt et al. 2007; Yu et al. 2021], drawing and sculpting [Arora and Singh 2021; Evans 2015; Peng et al. 2018], direct manipulation of mesh vertices [Armin Rigo 2020; Bærentzen et al. 2019], and procedural modeling with boolean operations [Jerald et al. 2013; Mendes et al. 2017].

Sketching is a common and natural 3D modeling way in VR. For example, CASSIE [Yu et al. 2021], as a lightweight sketch system, is able to transform sparse strokes in 3D space to a 3D mesh, which can automatically detect closed loops of the curve network and predict the surface. Liu et al. [Liu et al. 2011] used sketching techniques for plush toy generation, but their resulting models do not have a high precision. The main disadvantage of the above mentioned sketching systems lies in that they require relatively high sketching skills, and it is difficult to generate very complex models and tedious to edit the final shape.

In addition to sketching, painting and sculpting can be also accomplished in virtual reality to generate beautiful textures or elaborate carvings. Rahul et at. [Arora and Singh 2021] proposed a 3D stroke projection technique to draw mid-air strokes precisely on the surface of a 3D virtual object. Peng et al. [Peng et al. 2018] presented a 3D sculpting system with a 2D brushing interface that consists of a sculpting canvas and a widget panel. They considered how a model is authored via dynamic workflows in addition to what is shaped in static geometry, thus supporting accurate analysis of user intentions and more general synthesis of shape structures.

Signifier-based approach [Bærentzen et al. 2019] realizes 3D modeling by directly manipulating the vertices of polygonal meshes. The system performs operations such as face rotation and translation, vertex movement and merging through the correspondence between the affordances and signifiers. Sketchup is a CAD modeling tool that extends VRSketch [Armin Rigo 2020]. It includes drawing layout functionality and surface rendering in different styles, and enables placement of its models within Google Earth. Because of the limited resolution of the mesh model, mesh manipulation based approaches usually require further refinement processing.

There are also some VR based modeling approaches that depend on boolean operations, such as [Jerald et al. 2013] and [Mendes et al. 2017]. They respectively use the magic wand, menu or gesture to define boolean operation types. The biggest problem lies in that they do not include an efficient CSG solver, and thus cannot run very smoothly. For example, the modeling system presented by Mendes et al. [Mendes et al. 2017] has to stop for half a second when one boolean

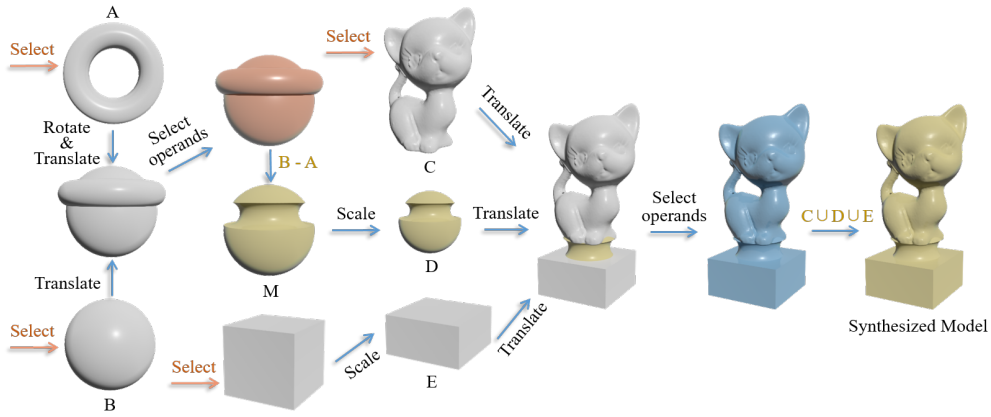


Fig. 2. A typical workflow to create a 3D model with EasyVRModeling. During modeling, users need to select a favorite component, transform (translate, scale or rotate) it to a desirable configuration, and specify the boolean operation type. By a sequence of simple interactions in VR, users finally obtain a synthesized model.

operation is triggered. Dreams [Evans 2015] directly evaluates the SDF to generate a dense point cloud on the surface of CSG sculpts. Dreams focuses on sculpting and rendering, without producing a watertight manifold polygonal mesh.

Besides, there are some VR based systems to simply combine a number of basic objects into a scene, without really synthesizing them into a whole object [Beever et al. 2020; De Leon et al. 2016; Ferreira et al. 2021; Wang et al. 2013].

3 WORKFLOW

Our EasyVRModeling system uses two interactive ways at the same time, i.e., menu interaction for selecting the next generator and direct control by the VR controller for grabbing/transforming objects and selecting boolean operation types. In our system, the boolean operation types include union, intersect and subtract, and the transform bahaviors include translation, scaling and rotation. Figure 3 shows the snapshot of the working scene of the EasyVRModeling system, which the menu is the dataset panel. Currently, our system has a dataset of 155 basic models, which can be divided into three categories: simple geometric primitives, animal models and chair components (from COSEG [Wang et al. 2012]). Users are allowed to freely augment the dataset to increase the shape diversity.

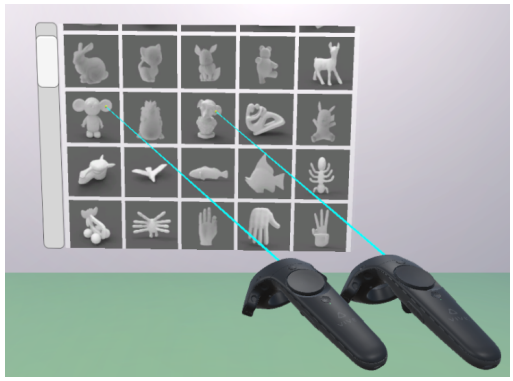
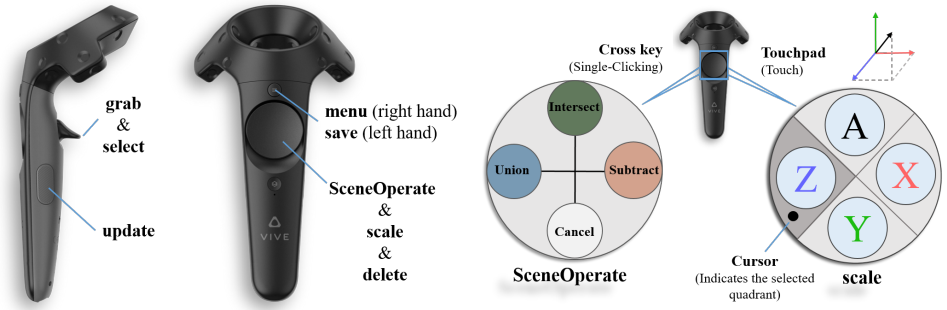


Fig. 3. The menu display EasyVRModeling system's component dataset which provides 155 models. Users can select favorite components using the magic wand.

EasyVRModeling accomplishes a modeling task by decomposing the whole task into a sequence of atomic operations. Figure 2 illustrates the modeling process of how to add a pedestal onto the Kitten model. Starting from opening the menu, the user can select a torus A and a sphere B from the dataset. Then the user places the two models in the appropriate position and orientation, selects them as the subtraction operands and finally performs the boolean subtraction $B - A$, yielding a new model M. After that, the result M can be shrunk to D. The user then selects a cube and improportionally scales it to E. Finally, the user selects the Kitten model C, as well as D and E, as



(a) Controllers (b) SceneOperate and Scale mode

Fig. 4. (a)We use a pair of controllers to define all necessary operations for boolean operations, coordinate transformation and import/export operations. (b)The *SceneOperate* and *Scale* panel. For left, the SceneOperate mode is activated when single-clicking "Cross Key". For right, the scaling panel appears when the right-hand "Touchpad" is touched during the selection mode.

the union operands, and perform $C \cup D \cup E$ to get a synthesized model, which finishes the operation of adding a pedestal.

4 MID-AIR USER INTERACTION

In this section, we present the 3D modeling interface of our system. More technical details on geometric representation of a 3D shape and the voxel-based SDF fusion in Section 5.

4.1 Various operation modes

In order to facilitate users to perform real-time modeling, we use a pair of controllers (see Figure 4(a)) to define all necessary operations for boolean operations, coordinate transformation and import/export operations.

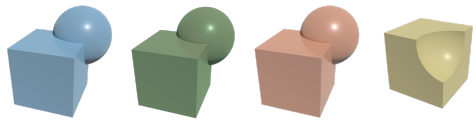
- **Model selection:** pop the dataset panel and specify a favorite component;
- **Coordinate transformation:** translate, rotate or scale an object;
- **Boolean operations:** union, intersect, and subtract;
- **File export:** save the final shape/scene into a file.

All the operations are implemented with a VR extended menu and a pair of 6-DOF controllers. EasyVRModeling provides eight operation modes: *grab, select, update, menu, save, SceneOperate, scale, delete* as Figure 4(a) shows. For ease of user interaction, most of the operations can be done using any one of the two controllers without any difference. Only *menu* (pop the dataset panel) and *save* (save the final shape/scene into a file) distinguish between the left and right hands.

When any of the controllers hits a model, it indicates that the model is selected. The user can keep grabbing the model by pressing the *grab* button, and move it to a different position or rotate it to a different orientation. Clicking the *menu* button of the right-hand controller can pop the dataset panel. Users can select a favorite component by press *select* button with the help of the magic wand, as shown in Figure 3. The menu can be toggled off by clicking the *menu* button again. The *save* button of the left controller, instead, is used to save the object hit by the left controller. The *SceneOperate* mode allows users to select (or deselect) operands, and the *update* mode activates the real boolean operations (synthesis of SDFs) and extract mesh surface. The touchpad of HTC VIVE controller is a multi-function button. When it is single-clicked the cross key area, the *SceneOperate* mode is toggled on. When it is double-clicked the center area, the *delete* mode is toggled on. When the touchpad of the right-hand controller is touched, the *scale* mode is toggled on.

4.2 SceneOperate mode

As Figure 4(b) shows, the *SceneOperate* mode has 4 kinds of operations, i.e., *union*, *intersect*, *subtract* and *cancel*. We always interpret a CSG task as a sequence of boolean operations. For example, in order to compute $A \cup B \cap C$, we interpret it as $(A \cup B) \cap C$. Users can hit A first (representative selects the model) to click *union* (be ignored), then hit B to click *union* again, and finally hit C to click *intersect*. The *cancel* option is to remove an operand from the CSG calculation sequence. Furthermore, in order to give more visual cues to users, we render the operands in a specified color according to the boolean operation type. When it enters the *union/intersect/subtract* modes, the operands are respectively rendered in blue, green and orange. After *update* is finished, the result is rendered in yellow as Figure 5 shows.



(a) (b) (c) (d)

Fig. 5. (a-c) the two components are selected for union, intersect, subtract respectively. (d) After *update* is finished, the result is rendered in yellow.

4.3 Scaling mode

The EasyVRModeling system also supports various scaling operations. In order to enter the scaling mode, users can use the two controllers to hit the target model at the same time and hold the grab button. After that, users can touch the "Touchpad" of the right-hand controller and then see a scaling panel as Figure 4(b) shows. X/Y/Z/A respectively represent a scaling operation along X-axis, Y-axis, Z-axis and an isotropic scaling. The cursor tracks the touched position to indicate the selected quadrant. Suppose that the two controllers are respectively located at (x_1, y_1, z_1) and (x_2, y_2, z_2) at the moment that the two grab buttons are pressed simultaneously. Once the positions of the two controllers are updated, we render the scaled model immediately. The scaling ratio is defined as

$$\frac{\sqrt{(x_1' - x_2')^2 + (y_1' - y_2')^2 + (z_1' - z_2')^2}}{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}}, \tag{1}$$

where (x_1', y_1', z_1') and (x_2', y_2', z_2') are the new positions. Currently we limit the ratio within $[0.4, 4]$, which suffices for most applications.

4.4 Other modes

During the modeling process, it's observed that the *copy* and *delete* functions are also very useful. In our implementation, the *delete* function is activated when users double-click the *delete* button for the selected model and the *copy* function is activated only one operand when the *update* button is pressed.

5 SDF SYNTHESIS

Explicit and implicit representations are two ways to represent geometric models in computer graphics. Compared to explicit approaches, Implicit representations are proven to be very helpful for synthesizing multiple models [De Araújo et al. 2015; Duan et al. 2020; McInemey and Terzopoulos 1999; Pasko et al. 1995; Wyvill et al. 1999]. In this section, we propose to use SDF as the implicit representation, with which we discuss various boolean operations and surface extraction.

5.1 Continuous and discrete SDF

Let Ω^-, $\partial\Omega$ and Ω^+ respectively denote the point set inside the surface, that on the surface and that outside the surface. The SDF $\phi(p)$ [Duan et al. 2020] is thus defined as

$$\phi(p) = \begin{cases} -d(p, \partial\Omega), & p \in \Omega^- \\ 0, & p \in \partial\Omega \\ d(p, \partial\Omega), & p \in \Omega^+, \end{cases} \tag{2}$$

where $d(p, \partial\Omega)$ is the straight-line distance from point p to the closest point on $\partial\Omega$. Given a SDF, the surface can be obtained by extracting the zero-value level set surface.

For a continuous SDF, it can report the signed distance value at any point. But it is hard to operate since there are infinitely many points in the bounding volume D. Therefore, by discretizing D along X-axis, Y-axis and Z-axis in a step of S, the bounding volume D can be decomposed into $\lfloor \frac{L}{S} \rfloor \times \lfloor \frac{W}{S} \rfloor \times \lfloor \frac{H}{S} \rfloor$ cells, where L, W, H are respectively the side lengths of D along X-axis, Y-axis and Z-axis. It's obvious that smaller S leads to higher resolution yet larger memory usage and computational cost. The grid points can be thus indexed by

$$\left\{ p_{i,j,k} : 0 \le i \le \left\lfloor \frac{L}{S} \right\rfloor, 0 \le j \le \left\lfloor \frac{W}{S} \right\rfloor, 0 \le k \le \left\lfloor \frac{H}{S} \right\rfloor \right\}, \tag{3}$$

The signed distance field Φ_D is the discrete SDF in the bounding volume D, which relationship is as follows:

$$\Phi_D = \{\phi(p_{i,j,k})\}. \tag{4}$$

Finally, we can use Marching Cubes [Lewiner et al. 2003; Lorensen and Cline 1987; Newman and Yi 2006] to extract the polygonal surface from Φ_D.

5.2 Precomputing SDFs and local-global alignment

In the preprocessing phase, we need to compute Φ_D for each component (typically a polygonal mesh), which denoted *local* Φ_D:

(1) Compute the axis-aligned bounding box D;
(2) Move the component such that its center coincides with the origin;
(3) Scale the component such that the longest side length of D is 0.4 and set the bounding box padding is 0.1;
(4) Discretize D into a set of regular grid points, with the step $S_{local} = 0.01$ by default;
(5) For each grid point $p_{i,j,k}$, we compute the distance from $p_{i,j,k}$ to the surface using PQP [Gottschalk et al. 1996; Larsen et al. 1999] while determining whether $p_{i,j,k}$ is located inside or not with CGAL [The CGAL Project 2021]. In this way, we get a signed distance value $\phi_l(p_{i,j,k})$ for $p_{i,j,k}$;
(6) Transform Φ_D into a 3D texture and keep it in a ScriptableObject asset of Unity.

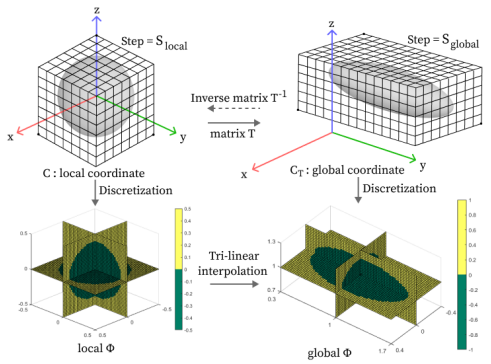


Fig. 6. Computing the *global* Φ of the ellipsoid using precomputed *local* Φ of the sphere.

See Figure 6 for illustration, a component C is placed in the scene, transformed by a matrix $\mathbf{T}$, during the modeling phase. Therefore, the counterpart after transformed is C_T, which can be

discretized in a step of S_{global}, denoted as *global* Φ_D, as shown in Figure 6. we have to align *global* Φ_D with the *local* Φ_D. First, we need to compute the bounding box D_{T} of C_{T} and discretize it in the step S_{global}. Second, for each grid point $p_{i,j,k}$ in D_{T}, we compute $\phi_g\left(p_{i,j,k}\right)$ by querying $\phi_l\left(\mathbf{T}^{-1}p_{i,j,k}\right)$ with a tri-linear interpolation technique.

Tri-linear interpolation. Suppose that in the local coordinate system of the component, its bounding box has been discretized into a set of cubic elements. Generally the point $\mathbf{T}^{-1}p_{i,j,k} \triangleq (x^*, y^*, z^*)$ is in one cubic element that is bounded by $x_1 \le x^* \le x_2, y_1 \le y^* \le y_2, z_1 \le z^* \le z_2$. We define

$$\lambda = \frac{x_2 - x^*}{x_2 - x_1}, \mu = \frac{y_2 - y^*}{y_2 - y_1}, \nu = \frac{z_2 - z^*}{z_2 - z_1}, \tag{5}$$

and denote the values at the 8 grid points by

$$\phi_{000}, \phi_{100}, \phi_{010}, \phi_{001}, \phi_{011}, \phi_{101}, \phi_{110}, \phi_{111}. \tag{6}$$

Then $\phi_g\left(p_{i,j,k}\right)$ can be estimated by

$$\begin{aligned}&\lambda\mu\nu\phi_{000} + (1-\lambda)\mu\nu\phi_{100} + \lambda(1-\mu)\nu\phi_{010} + \lambda\mu(1-\nu)\phi_{001} + \lambda(1-\mu)(1-\nu)\phi_{011}\\ &+ (1-\lambda)\mu(1-\nu)\phi_{101} + (1-\lambda)(1-\mu)\nu\phi_{110} + (1-\lambda)(1-\mu)(1-\nu)\phi_{111}.\end{aligned}$$

In fact, it is likely that after the grid point $p_{i,j,k}$ is mapped from the global coordinate system to the local coordinate system, the position (x^*, y^*, z^*) is located outside the bounding box D so that one cannot find a suitable cubic element to contain (x^*, y^*, z^*). We simply set $\phi_g\left(p_{i,j,k}\right) = \infty$, which does not influence the extraction the zero-value level-set surface. In this way, we can quickly update the SDF of a component after it is transformed, without re-computing the distance between each grid point and the surface.

5.3 CSG-like boolean operations

At this moment, we suppose that the SDF of the first object M_1 is ϕ_1 and we come to synthesize ϕ_1 with a new object M_2 whose SDF is ϕ_2. We consider the union/intersect/subtract operation between M_1 and M_2. For each grid point $p_{i,j,k}$ in the scene, we set

$$\phi_{(p_{i,j,k})} = \begin{cases}\min\left(\phi_1(p_{i,j,k}), \phi_2(p_{i,j,k})\right), & M_1 \cup M_2\\ \max\left(\phi_1(p_{i,j,k}), \phi_2(p_{i,j,k})\right), & M_1 \cap M_2\\ \max\left(\phi_1(p_{i,j,k}), -\phi_2(p_{i,j,k})\right), & M_1 - M_2.\end{cases} \tag{7}$$

where $\phi_{(p_{i,j,k})}$ is the synthesized SDF value at $p_{i,j,k}$.

In order to reduce the computational cost, it suffices to restrict the computation in a smaller region. Let D_1 and D_2 be respectively the bounding boxes of M_1 and M_2. We define the restricted region $D_{\text{restricted}}$ as follows:

$$D_{\text{restricted}} = \begin{cases}D_1 \cup D_2, & \text{Union}\\ D_1 \cap D_2, & \text{Intersect}\\ D_1, & \text{Subtract.}\end{cases} \tag{8}$$

5.4 Speedup strategies

Since the real-time experience is crucial to develop a VR based modeling system, we have a set of speedup strategies to accelerate the computation and provide instant feedback to users.

Parallel SDF fusion. We maintain a global discrete SDF throughout the whole modeling process. During each step of boolean operations, we first get the restricted bounding region (see Eq. (8)) where the SDF values need to updated. Then we map the task of SDF fusion to Compute Shader by using two 3D textures. We recommend to use a 3D local size of 8x8x8 for internally storing 3D texture data on GPU. The detailed fusion rules are available in Eq. (7).

Parallel marching cubes. In the modeling phase, it is important to render the surface (instead of the SDF) in real time in the virtual scene. For this purpose, each time when we finish a set of boolean operation, we need to immediately extract the zero-value level-set surface from the fused SDF. Similarly, we map the task of surface extraction to Compute Shader by importing $\{(p_{ijk}, \phi(p_{ijk}))\}$ to GPU, where p_{ijk} is the coordinate of a grid point while $\phi(p_{ijk})$ is the SDF value at this grid point. Finally, we need to synchronize the mesh pieces from GPU to CPU and stitch them together to a single mesh surface. The use of precomputed SDF, parallel SDF fusion and marching cubes enables EasyVRModeling to run in real time.

Lazy boolean operation update. Generally, the CSG computation task specified by users each time can be decomposed into a sequence of atomic computation sub-tasks, where sub-task operates between two operands. Suppose that the CSG computation task is like

$$((A \cup B) \cap C) - D, \tag{9}$$

which contains three steps:

$$(A \cup B) \triangleq E_1 \;\rightarrow\; (E_1 \cap C) \triangleq E_2 \;\rightarrow\; (E_2 - D) \triangleq E_3. \tag{10}$$

In order to enable users have a smooth experience, rather than run marching cubes for 3 times, it is better to perform a single marching cubes computation. Therefore, we suggest a lazy boolean operation update scheme. Users first take the following actions in order: hit A to click *union* (be ignored), hit B to click *union* ($\cup$), hit C to click *intersect* ($\cap$), and hit D to click *subtract* ($-$). After that, users can click *update* and see the fused surface. In this way, it requires one single surface extraction operation for the compound CSG task. Another benefit of the lazy scheme lies in that after the user selects these operands, we still allow the user to have a chance to transform (move, scale or rotate) each operand. Although we implement the marching cubes in parallel, it is impossible to perform the surface extraction at 90 FPS.

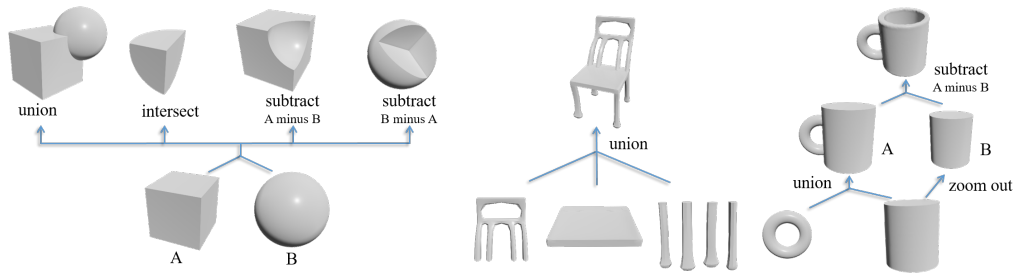
Choice of the step parameter. It's also important to seek for a trade-off between accuracy and efficiency. S_{global} is a hyperparameter that users can set according to their needs. In implementation, we tried various step parameters S_{global} about $global\,\Phi$ within $[0.001, 0.02]$ and found that $S_{global} = 0.01$ is a good choice. If S_{global} is too small, it requires a huge computational workload, causing non-smooth experience. But if S_{global} is too large, the extracted models have a low resolution, causing serious visual artifacts.

6 USER STUDY

In this Section, we evaluate the EasyVRModeling system based on user study statistics and training periods. We also give a detailed analysis on the run-time performance of the main operations of our algorithm.

6.1 Participants

We recruited 34 volunteers (8 females and 26 males) aged from 18 to 30. The volunteers consist of undergraduates, graduate students and doctoral students. Most of them have an educational background of computer science. Four of them have a long VR experience of more than one year, five volunteers ever used VR devices but the experience is less than one year, and the rest



(a) Cube and Sphere (b) Chair (c) Cup

Fig. 7. Training tutorial: (a) teach volunteers to learn how to conduct union/interact/subtract between a cube and a sphere. (b) synthesizing a chair model using six components. (c) synthesizing a cup model using a torus and a cylinder.

are VR newcomers. Nearly none of them tried VR based modeling systems before, but most of the volunteers can remember the operation manual of our system and learn how to create new 3D models very quickly. Our tests were divided into two phases. In the first phase, part of the volunteers are asked to use EasyVRModeling and give their feedback about interaction. Based on their feedback, we improved EasyVRModeling and added some new features (copy and delete). In the second phase, we asked volunteers to synthesize their favorite shapes or scenes and then fill out an online questionnaire on simplicity, practicability and versatility.

6.2 User training

In order to enable users to quickly master the basic operations of the EasyVRModeling modeling, such as grasping, moving, rotating, scaling, boolean operations, copy and deletion, we provided the volunteers with an EasyVRModeling instruction in advance. After a simple training of less than ten minutes, users began to create their favorite models and then filled out an online questionnaire.

EasyVRModeling ran on a computer with Intel Core i7-6700 Processor, 24GB memory and NVIDIA GeForce GTX 1070. Before using EasyVRModeling to implement creative modeling, volunteers need to warm up in advance to ensure that they are fully familiar with the use of the system. In the training phase, we asked the volunteers to accomplish three tasks. The first task is to use a cube and a sphere perform the union/intersect/subtract of boolean operations successively (Figure 7(a)). Through this task, users can learn basic interaction operations. Second, we asked the volunteers to build a chair from six components (Figure 7(b)). The purpose of this task is to teach users how to model a complicated model using a sequence of boolean operations. Finally, we asked the volunteers to build a cup model using a cylinder and a torus (Figure 7(c)). By this task, we want to teach users to realize that it is possible to create a complicated model with a carefully designed operation sequence from a quite limited number of basic components.

We asked volunteers to accomplish the three training tasks and kept down the training period for each volunteer. The training period is used to check if the interaction is user friendly and the system is easy to use.

6.3 Evaluation

After users finished the training phase, we asked them to use EasyVRModeling to create their favorite models/scenes, and then score the system based on various indicators. Figure 1 shows some interesting results modeled by 34 volunteers. The results are created from some basic components such as cylinder, torus, cube, sphere, bunny and chair parts. Although most of the volunteers lack the VR modeling experience, they can still build very impressive models/scenes with EasyVRModeling.

In Figure 1, a female amateur Maya modeling enthusiast (V3) showed enthusiasm and talent in scene construction. V3 built three beautiful and descriptive scenes, and named them by *Rabbit House, Pikachu Spring Tour, Merry Christmas*, respectively. *Rabbit House* (Figure 1(g)) describes a summer scene where a rabbit is resting in his house, and there are mushrooms, a hand symbol and a small shrub nearby the tiny house. *Pikachu Spring Tour* (Figure 1(d)) conveys a story that Pikachu is riding on a dog and running in the forest. *Merry Christmas* (Figure 1(i)) conveys a vivid scene that Santa Claus went to deliver Christmas gifts, leaving two elk pulling a sleigh and waiting for him. Another female volunteer (V1) used a sphere, a chair leg, and a chair board to create a delicate flower (Figure 1(c)). Firstly, she constructed a petal by subtracting two scaled spheres and then combined multiple petal into a flower by rotation and copy. Furthermore, she used a sphere to produce the stamen, the chair board to produce the flower leaf, and the chair leg to produce the flower stem. V1 also used a sphere, a cylinder, a chair leg and a chair board as the basic elements to construct a Little-Kid model (Figure 1(a)) by copy, scale, and union operations. What's more, the Snail model (Figure 1(f)) was constructed from a torus, a cylinder and a chair leg by a male volunteer (V4); The bamboo clump (Figure 1(b)) was constructed from a sphere and a chair leg, where the leaf was obtained similarly to V1's petal construction. To summarize, by skillfully using the geometric transformation operations and the boolean operations, users can create very complicated shapes from a limited number of basic components.

Quantitative analysis. The average timing cost for volunteers to learn EasyVRModeling is 233 seconds, where the shortest period is 105s and the longest period is 420s. On one hand, it shows that users can get familiar with the system after a relatively short period of learning. On the other hand, the large variance of training time shows that different volunteers have different backgrounds, and more skilled volunteers can understand/accomplish the training stuff more quickly.

Table 1. For the three tasks of training tutorial in Figure 7, we respectively record the GPU time (in milliseconds) spent for one *update* operation (including multiple SDF fusions and one surface extraction). Note that the timing cost depends on the choice of the step parameter S_{global}.

	Step S_{global}	0.012	0.01	0.008	0.006
Cu-Sp	SDF fusion	8	9	11	12
	surface extraction	14	22	46	98
	total time	22	**31**	57	110
Chair	SDF fusion	26	28	35	45
	surface extraction	27	34	76	186
	total time	53	**62**	111	231
Cup	SDF fusion	15	16	18	21
	surface extraction	24	45	89	208
	total time	39	**61**	104	229

In Table 1, we keep down the timing statistics for performing one *update* operation. To be more detailed, it involves multiple SDF fusions and one surface extraction. It can be seen that the operation of surface extraction is particularly sensitive to the step parameter S_{global} - a smaller S_{global} requires a larger timing cost of surface extraction. If S_{global} is too small, there is a conspicuous frame rate fluctuation. That's why we invent a "lazy boolean operation update" strategy to reduce the effect of stuttering or lag. In our implementation, we set $S_{global} = 0.01$ by default to balance efficiency and accuracy.

User feedback. The questionnaire for evaluating EasyVRModeling is divided into three parts: the first part is to fill in the tester's personal information, the second part is to investigate the tester's recognition of the practicality of the EasyVRModeling modeling system, and the last part is to give the creation support index (CSI) [Cherry and Latulipe 2014].

Table 2. The practicability statistics of the EasyVRModeling system (between 1 and 10; the higher the score, the better).

EasyVRModeling	Mean (deviation)	Weighted Mean
Playability	9.00 (1.28)	9.52 (6.32)
Ease of utilization	8.26 (1.90)	8.87 (6.22)
Fluency	7.91 (4.20)	8.64 (6.17)
Overall practicability	8.39 (1.82)	9.01 (6.22)

Considering that different users have different backgrounds, we ask the volunteers to rate their professional level about the VR based modeling, ranging from 0.0 to 2.0 (i.e. the mean of weights is 1.0). Using the professional level as the weighting scheme, we get a significance weighted score. See the statistics in Table 2.

In Table 2, we provided the statistics of volunteers' evaluation of the practicability of the EasyVRModeling system. The practicality of the modeling system is evaluated from three perspectives: playability, ease of operation, and fluency. Based on the statistics (mean value and standard deviation), it can be seen that users agree EasyVRModeling is fun and easy to operate. It's also worth noting that some users the system may lag during modeling. In fact, it is due to that currently we didn't limit the maximum length of the boolean operation sequence. Our system assumes that during each step, the operations of SDF fusion and surface extraction can be confined in a small region. However, if users select too many operands at one time (e.g., larger than 6), generally the influenced region becomes very large, which requires a long computation time.

Table 3. The average Creative Support Index (CSI) [Cherry and Latulipe 2014] scores (between 1 and 20; the higher the score, the better).

EasyVRModeling	Mean (deviation)	Weighted Mean
Enjoyment	16.65 (3.40)	16.97 (11.45)
Exploration	17.35 (3.61)	17.64 (12.16)
Expressiveness	17.26 (2.38)	17.54 (11.36)
Immersion	17.91 (2.47)	17.93 (11.59)
Results Worth Effort	17.35 (2.18)	17.44 (11.12)
CSI	17.31 (2.90)	17.51 (11.37)

In Table 3, the volunteers gave their scores in terms of CSI [Cherry and Latulipe 2014]. Statistical data in Table 2 shows that users believe that EasyVRModeling modeling system really helps in creative modeling, and has high entertainment, exploration and strong expression ability, enabling users to give full play to their imagination in modeling.

6.4 Comparison to other VR based modeling tools

MakeVR [Jerald et al. 2013]. It introduces a free-form modeling tool in VR, with a two-handed 3D multi-touch interface, for designers and makers. Since it aims at a professional-grade CAD engine, it has a high accuracy requirement on boolean operations, and cannot ensure smooth interaction when the scene becomes complicated. Our system, instead, maintains the whole scene in a fixed-resolution of volume data and thus does not suffer from the increasing complexity of the scene.

Mendes [Mendes et al. 2017]. To our knowledge, Mendes supports boolean operations between two given operands. However, it has at least two disadvantages. First, the interaction is not user friendly - users need to grab the two objects simultaneously. Second, the computation is not efficient enough to conduct real-time boolean operations between two complex shapes.

VRSketch [Armin Rigo 2020]. It combines Sketch and other CAD tools for VR modeling. VRSketch directly manipulates mesh vertices, edges and faces to build models with diverse shapes. Based on our experience, there is an occurrence of stuttering or lag if the mesh has a high complexity. Generally, it can only deal with low-resolution mesh models, making the resulting mesh not very informative. One has to refine the model as a postprocessing step.

EasyVRModeling. Our EasyVRModeling system enables users to easily and quickly synthesize components, by various geometric transformations and various boolean operations, into a visually appealing model. Generally the output is a closed watertight polygonal model, which shows that our system is much different those layout arrangement approaches. Due to this nice feature, users can directly export the synthesized model to a 3D printer for direct fabrication, which is very helpful for educating artistic students. In Figure 8, we show the printed version of the Bunny Pavilion model (see Figure 1) from 4 different perspectives.

Fig. 8. Users can directly input the synthesized model into a 3D printer for fabrication. We show the printed version of Bunny Pavilion (see the digital model in Figure 1) from 4 different perspectives.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we build an easy-to-use VR based modeling system, without the help of any external devices. The central idea is to help users create a complicated model with a limited number of simple primitives. On one hand, our system provides user-friendly interaction. The system provides a menu for component selection and two controllers to define CSG boolean operation types and control coordinate transformation. On the other hand, we use the discrete SDF to represent each component and further develop a trilinear interpolation technique to support real-time synthesis of two SDFs for quick CSG-like operations (union/intersect/subtract). Both SDF fusion and surface extraction are implemented via GPU in real time. However, EasyVRModeling, in its current form, still needs further improvement. For example, if a component is scaled down to a very little size, it is hard to grasp the object. Furthermore, it is interesting to explore the possibility of collaborative creation in EasyVRModeling.

REFERENCES

Duncan Fraser Armin Rigo, Maciej Fijałkowski. 2020. *version 16.0.0.* Baroque Software. https://vrsketch.eu/

Rahul Arora. 2020. Creative Expression with Immersive 3D Interactions. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3334480.3375028

Rahul Arora and Karan Singh. 2021. Mid-Air Drawing of Curves on 3D Surfaces in Virtual Reality. *ACM Transactions on Graphics (TOG)* (2021).

Andreas Bærentzen, Jeppe Revall Frisvad, and Karan Singh. 2019. Signifier-Based Immersive and Interactive 3D Modeling. In *25th ACM Symposium on Virtual Reality Software and Technology* (Parramatta, NSW, Australia) *(VRST '19)*. Association for Computing Machinery, New York, NY, USA, Article 18, 5 pages. https://doi.org/10.1145/3359996.3364257

Lee Beever, Serban Pop, and Nigel W. John. 2020. LevelEd VR: A virtual reality level editor and workflow for virtual reality level design. In *2020 IEEE Conference on Games (CoG)*. 136–143. https://doi.org/10.1109/CoG47356.2020.9231769

Erin Cherry and Celine Latulipe. 2014. Quantifying the creativity support of digital tools through the creativity support index. *ACM Transactions on Computer-Human Interaction (TOCHI)* 21, 4 (2014), 1–25.

Bruno Rodrigues De Araújo, Daniel S Lopes, Pauline Jepp, Joaquim A Jorge, and Brian Wyvill. 2015. A survey on implicit surface polygonization. *ACM Computing Surveys (CSUR)* 47, 4 (2015), 1–39.

Josen Daniel O. De Leon, Romelio P. Tavas, Rodolfo A. Aranzanso, and Rowel O. Atienza. 2016. Genesys: A Virtual Reality scene builder. In *2016 IEEE Region 10 Conference (TENCON)*. 3708–3711. https://doi.org/10.1109/TENCON.2016.7848751

Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. 2020. *VRSketchIn: Exploring the Design Space of Pen and Tablet Interaction for 3D Sketching in Virtual Reality.* Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376628

Zhongxiang Duan, Qin Yang, Xianhai Meng, and Jigang Li. 2020. Detailed Voxel-Based Implicit Modeling With Local Boolean Composition of Discrete Level Sets. *IEEE Access* 8 (2020), 48376–48385. https://doi.org/10.1109/ACCESS.2020.2979571

Alex Evans. 2015. Learning from failure: a survey of promising, unconventional and mostly abandoned renderers for 'dreams ps4', a geometrically dense, painterly ugc game. *Advances in Real-Time Rendering in Games. MediaMolecule, SIGGRAPH* (2015).

Stanley Osher Ronald Fedkiw and Stanley Osher. 2002. Level set methods and dynamic implicit surfaces. *Surfaces* 44, 77 (2002), 685.

João Ferreira, Daniel Mendes, Rui Nóbrega, and Rui Rodrigues. 2021. Immersive Multimodal and Procedurally-Assisted Creation of VR Environments. In *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 30–37. https://doi.org/10.1109/VRW52623.2021.00012

Stefan Gottschalk, Ming C Lin, and Dinesh Manocha. 1996. OBBTree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 171–180.

Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 2006. Teddy: a sketching interface for 3D freeform design. In *ACM SIGGRAPH 2006 Courses*. 11–es.

Bret Jackson and Daniel F. Keefe. 2016. Lift-Off: Using Reference Imagery and Freehand Sketching to Create 3D Models in VR. *IEEE Transactions on Visualization and Computer Graphics* 22, 4 (2016), 1442–1451. https://doi.org/10.1109/TVCG.2016.2518099

Jason Jerald and Richard Marks. 2016. Human-Centered Design for VR Interactions. In *ACM SIGGRAPH 2016 Courses* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 15, 60 pages. https://doi.org/10.1145/2897826.2927320

Jason Jerald, Paul Mlyniec, Arun Yoganandan, Amir Rubin, Dan Paullus, and Simon Solotko. 2013. Makevr: A 3d world-building interface. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 197–198.

Leif P Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. 2001. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 57–66.

Eric Larsen, Stefan Gottschalk, Ming C Lin, and Dinesh Manocha. 1999. *Fast proximity queries with swept sphere volumes*. Technical Report.

Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. 2003. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of graphics tools* 8, 2 (2003), 1–15.

Yong-Jin Liu, Cui-Xia Ma, and Dong-Liang Zhang. 2011. EasyToy: Plush Toy Design Using Editable Sketching Curves. *IEEE Computer Graphics and Applications* 31, 2 (2011), 49–57. https://doi.org/10.1109/MCG.2009.147

Pedro Lopes, Daniel Mendes, Bruno Araújo, and Joaquim A Jorge. 2011. Combining bimanual manipulation and pen-based input for 3d modelling. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*. 15–22.

William E Lorensen and Harvey E Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics* 21, 4 (1987), 163–169.

Tim McInemey and Demetri Terzopoulos. 1999. Topology adaptive deformable surfaces for medical image volume segmentation. *IEEE transactions on medical imaging* 18, 10 (1999), 840–850.

Daniel Mendes, Daniel Medeiros, Maurício Sousa, Ricardo Ferreira, Alberto Raposo, Alfredo Ferreira, and Joaquim Jorge. 2017. Mid-air modeling with Boolean operations in VR. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*. 154–157. https://doi.org/10.1109/3DUI.2017.7893332

Timothy S Newman and Hong Yi. 2006. A survey of the marching cubes algorithm. *Computers & Graphics* 30, 5 (2006), 854–879.

Stanley Osher and Ronald Fedkiw. 2006. *Level set methods and dynamic implicit surfaces*. Vol. 153. Springer Science & Business Media.

Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir Savchenko. 1995. Function representation in geometric modeling: concepts, implementation and applications. *The visual computer* 11, 8 (1995), 429–446.

Mengqi Peng, Jun Xing, and Li Yi Wei. 2018. Autocomplete 3D sculpting. *ACM Transactions on Graphics (TOG)* (2018).

Ryan Schmidt, Brian Wyvill, Mario Costa Sousa, and Joaquim A Jorge. 2007. Shapeshop: Sketch-based solid modeling with blobtrees. In *ACM SIGGRAPH 2007 courses*. 43–es.

The CGAL Project. 2021. *CGAL User and Reference Manual* (5.3 ed.). CGAL Editorial Board. https://doc.cgal.org/5.3/Manual/packages.html

Jia Wang, Owen Leach, and Robert W. Lindeman. 2013. DIY World Builder: An immersive level-editing system. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. 195–196. https://doi.org/10.1109/3DUI.2013.6550245

Yunhai Wang, Shmulik Asafi, Oliver van Kaick, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2012. Active Co-Analysis of a Set of Shapes. *ACM Trans. Graph.* 31, 6, Article 165 (nov 2012), 10 pages. https://doi.org/10.1145/2366145.2366184

Brian Wyvill, Andrew Guy, and Eric Galin. 1999. Extending the csg tree. warping, blending and boolean operations in an implicit surface modeling system. In *Computer Graphics Forum*, Vol. 18. Wiley Online Library, 149–158.

Emilie Yu, Rahul Arora, Tibor Stanko, J. Andreas Bærentzen, Karan Singh, and Adrien Bousseau. 2021. CASSIE: Curve and Surface Sketching in Immersive Environments. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohoma, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3411764.3445158