Sculpting Gestures

Chaoyu Du
chaoyu.du@arch.ethz.ch

Kladeftira Marirena
kladeftira@arch.ethz.ch

Ioanna Mitropoulou
mitropoulou@arch.ethz.ch

Jiahong Wang
wangjiah@student.ethz.ch

Abstract

We present a Mixed-Reality application that enables the sculpting of geometries using the user's hands directly on-site. Various digital tools that enable the design of custom geometry rely on keyboard and mouse input, which can be non-intuitive for creative work. Artists are often used to working with their hands and in direct relation with the surrounding environment. In response to that, we propose to use the HoloLens2 device to enable an intuitive digital sculpting process. Our investigations are assembled in an application called 'Sculpting Gestures', which was presented and tested by various users on the Demo day of the Mixed-Reality course.

Figure 1. Incidental Space: Inside the Swiss Pavilion at the 2016 Venice Biennale. source: archdaily.com [1]

1. Introduction

The geometric freedom enabled by computational design allows the creation of very complex and bespoke objects. However, the most common design platforms nowadays are CAD software packages that utilize the mouse and keyboard of the computer as design instruments. The latter are not particularly intuitive design tools. Hence, designers and artists often seek different solutions that allow them to engage with a digital platform in a more analog manner. Digital fabrication enabled by computational design allows manufacturing such bespoke and highly detailed objects. However, the lack of intuitive ways to design those has directed digital fabrication for niche and highly controlled computational methods or simply for the reproduction of existing objects.

Our application targets the gap between analog and tactile craft and advanced manufacturing methods like 3D printing. More specifically, sculpting, a common method to create bespoke objects, is traditionally perceived as both an art form and a craft. Existing digital sculpting applications lack the tactile approach, as designers need to use instruments or controllers that do not feature the intuitive stroke of a person's hand. Such an example is the exhibition *Incidental Space* that was featured at the 15th Venice Architecture Biennale, where a freeform cloud-like space was 3D printed. Still, due to the lack of tactile digital modeling methods, the space was crafted in small-scale models that were later scanned and reproduced with additive manufacturing (Fig. 1). Therefore, we try to simulate a digital craft that can be materialized in the physical world with digital manufacturing methods and can have a digitally crafted twin in the virtual world.

In this project, we explore a more intuitive way of designing free-form 3-Dimensional geometry using Mixed Reality (MR). To that end, we create an application that enables the implicit modeling of geometry using the hands directly on-site with the Hololens2 device. In that way, the user can sculpt a variety of shapes on a 1:1 scale and in relation to the real environment.

Our application combines digital design with the reality constraints. Use cases include furniture on a narrow space, structures that can only fit within specific areas in a room, bespoke sculptures designed on-site, etc. In a traditional design process, the complex surroundings are represented by 2D plans or 3D models, which requires manual measurement or process and is hard to perceive the spatial designs precisely. In contrast, our application provides an accurate

view of the design in physical space, which increases efficiency and accuracy in terms of time, money, and resources needed in the design to production stage, but also benefits the collaboration between client, designers, engineers, and manufacturers.

2. State of the art

2.1. Digital sculpting software

Sculpting software allows designers to insert details on meshes intuitively using brush-like tools. The main geometrical representations in commercial sculpting applications are polygon-based geometry and voxel-based geometry.

Polygon-based sculpting software requires consideration of vertices and typology changes of the mesh. Vertices are pulled around to achieve the target geometries. *Autodesk Maya* [5] is the most-known polygonal sculpting application, where users can move regions of vertices using various operations. *Autodesk Mudbox* [3] and *Pixologic ZBrush* [14] features dynamic tessellation of the polygon mesh, which dynamically updates the polygon mesh topology. *Autodesk Meshmixer* [4] combines polygonal sculpting with dynamic remeshing infrastructure to create highly detailed surfaces with much lower polycounts.

Voxel-sculpting stores data in the voxels. Voxels are two-dimensional pixels for three-dimensional space analog, and the voxel model is filled inside. Thus, voxel-sculpting is entirely free from polygon considerations and topology. *Pilgway 3D-Coat* [13] features voxel sculpting with user-specified spatial resolution, and topology changes are handled smoothly by the nature of the voxel data representation. *Blender* [6] is a free, open-source software. Its Voxel Remesher tool uses an OpenVDB to generate a new manifold mesh from the current geometry.

2.2. Related Work

Besides the traditional way of manipulating a mesh on a desktop environment typically accessed via a 2D interface, there already are some applications that allow users to sculpt immersively. *Geomagic Touch* is a haptic 3D input device that enables the user to sculpt in 3D. However, it remains confined to a 2D display. With VR headsets, such as HTC Vive, Oculus, Playstation VR, or MR equipment, uses can sculpt and visualize the results in 3D.

When sculpting in virtual reality environments, users utilize input devices, such as 6-axis space mouse, sensors, physical props, or special VR gloves to track finger trajectory or hand movement. *Gravity Sketch* [2] allows users to draw freehand in 3D space using smooth curves, then extrude surfaces into 3D space. Google's *Tilt Brush* [7] provides users with a variety of brushes, including ink, smoke, snow, and fire, etc. With *Kanova* [8] (Fig. 2) and *Unbound Alpha* [16], people can create artwork collaboratively.

Figure 2. Kanova: multi-platform VR sculpting tool [8]

Augmented reality environments are usually equipped with calibrated cameras to track the users' hand motion. Leap Motion's *Freefrom* [15] is a 3D sculpting software that runs on a computer. Users can create digital sculptures by manipulating clay-like objects with their fingers and hand gestures. *AiRSculpt* [10] is an HMD-based AR system that allows users to naturally explore and interact with virtual objects from various angles and distances within their environment.

3. Method

Our application was developed on the game engine Unity [17], using the Mixed Reality Toolkit (MRTK) libraries [12] that are compatible with the HoloLens 2 and implement basic functionality such as hand tracking or voice control. Our application was created using a combination of C# scripts, MRTK scripts, and unity game objects.

Implicit modeling is well known for its suitability to create complex geometry that always remains valid and doesn't suffer from the disadvantages of discrete representations that can become non-manifold, have holes, or invalid parts. In implicit modeling, changes in the shape topology are easy to handle without the need to make changes in the underlying representation [9]. As a result, we chose to base the sculpting functionality on implicit shape functions. The sculpted shape is defined as the isosurface of the distance values where they equal 0. As an input, we use analytical distance functions that are applied on a discrete regular grid and are saved and updated as discretized values on grid points.

3.1. Features

The application *Sculpting Gestures* has the following basic features

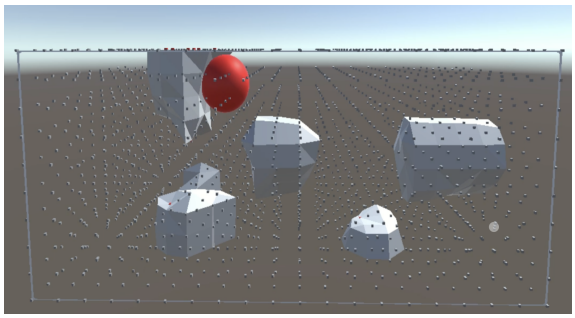

Figure 3. Regular grid inside the sculpting bounding box

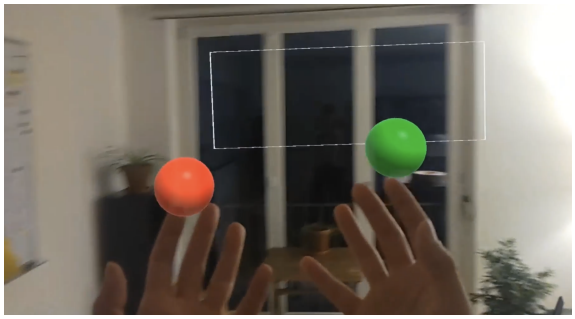

Figure 4. The sculpting primitives that update the grid values are attached to the user's index fingers.

Grid. The application initializes a bounding box, filled with a regular grid of *GridPoints* (Fig. 3). This defines the extents of the volume within which the user can sculpt. Each GridPoint is initialized with a positive distance value so that initially, there is no isosurface, starting, as a result, with an empty shape. The resolution of the grid determines the refinement and detail of the sculpting shape. However, increasing the resolution poses significant performance challenges, discussed in detail in Section 3.2.

Sculpting Tools. The sculpting tools are analytical distance functions. They update the GridPoints distance values according to the tools' parameters when applied on the grid. The two main types are the *activators*, which activate the grid points, thus adding volume, and the *deactivators*, which deactivate the grid points, thus removing volume. Two different distance functions were implemented; the *sphere*, which is suitable for creating curved shapes, and the *cube*, meant for creating straight lines and sharp corners. These are tied to Unity user objects that also have a mesh displaying the correct

User Interface. A main goal in our work was to make the application as intuitive and easy to use without prior knowledge as possible. Initially, we experimented with making the sculpting tools grabbable by the user. However, after testing this idea, we realized it is nonintuitive and tiresome to sculpt while clenching the fists to maintain the grabbing gesture. After a series of tests, we concluded that the easiest interaction is achieved by attaching the sculpting tools directly on the index fingers so that the user does not need to make any effort for the tools to follow their hands (Fig. 4).

As the sculpting tools fully occupy the hands, we decided against adding further hand-driven actions to keep the interactions simple. Instead, we use voice commands to allow the user to change the tools' parameters and carry out further actions such as saving the sculpting mesh or positioning the grid. The following voice commands are implemented:

- *On/Off* : Enables/Disables the sculpting tools.
- *Bigger/Smaller* : Increases/Decreases the size of the sculpting tools, multiplying their dimensions by 1.2 or 0.8 respectively.
- *Reset* : Resets the sculpting tools to their initial size.
- *Cubes/Spheres* : Allows to change the shape of the sculpting tools into cubes or spheres respectively. If the sculpting tools already have that shape, it has no effect.
- *Save* : Saves the current sculpted mesh into the memory of the Hololens on the 3D Objects folder, as an *.obj* file. When the application is run on the computer, it saves the file on the memory of the computer instead.
- *Grid* : Toggles a boolean that allows to re-position the grid which defines the sculpting volume. When this boolean is set to *True*, then the grid is attached to the right hand of the user. The user can place the grid on the desired position, and repeat this command, to toggle the grid movement back to *False*. This command is usually called at the beginning of the sculpting process, to position the grid on the space before the sculpting starts. However, it also works on any stage of the sculpting process.

Materials. To avoid making the mesh generation heavier in terms of computing and memory requirements, we decided against adding UV coordinates on the sculpted mesh. As a result, we could not apply textures that needed to be mapped on the mesh. Instead, we only used a metallic-looking shader and implemented sliders to enable the change of colors (Fig. 5).

3.2. Performance optimizations

To trade-off for longer battery life, Hololens 2 sacrifices its performance for a low-power processing unit, Qualcomm Snapdragon 850, and a small run-time memory – only 4 GB. Our application is doomed to be computationally intensive because its design nature is graphically

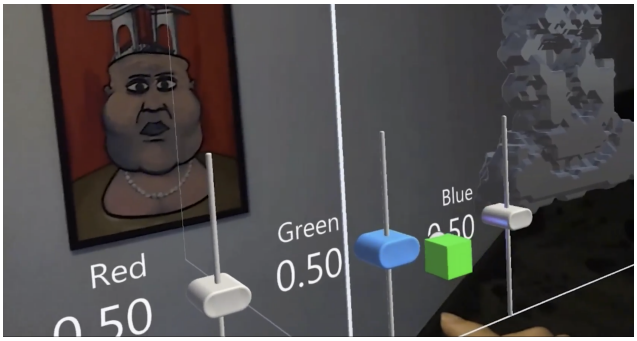

Figure 5. RGB sliders that enable the dynamic changing of the color of the sculpted shape.

demanding and requires large memory for storing the art piece. We did two major optimizations on collision detection and Marching Cubes to reduce latency and mitigate memory pressure.

Collision Detection. To speed up the computation, we brought the concepts from collision detection in computer graphics. The collision detection runs in a two-phase approach: a broad-phase detection followed by a narrow-phase detection. The broad-phase detection is fast, and it roughly sieves out unlikely collider pairs. The narrow-phase detection is exact but requires costly computations.

Sculpting Gestures. uses axis-aligned bounding boxes (AABB) for the broad phase and truncated signed distance function (TSDF) for the narrow phase. In the broad phase, the bounding box of the sculpting tool is compared to each grid points. However, since all points lie in the grid, we further optimize AABB by computing the intersection volume of the bounding box with the grid. After that, we can easily extract all potential grid points from the intersection volume. By this means, we effectively reduce the number of broad-phase collision detections from n^3 to 1, where n^3 is the number of grid points. We define TSDF for the cube and the sphere tools separately in the narrow phase. The TSDF is in the interval [-1, 1]. The TSDF function for the cube is

$$f(x) = clip(s^{-1} \cdot \|\mathbf{x} - \mathbf{x_0}\|_\infty - 1, -1, 1) \qquad (1)$$

where s is the cube's scale, and x_0 is its centroid. Similarly, the TSDF function for the sphere is

$$g(x) = clip(s^{-1} \cdot \|\mathbf{x} - \mathbf{x_0}\|_2 - 1, -1, 1) \qquad (2)$$

where s is the sphere's scale, and x_0 is its centroid. The final algorithm is described in the Algorithm 1.

Marching Cubes To reconstruct mesh from a grid of TSDF values, we use the Marching Cubes algorithm [11]. As its name suggests, the algorithm divides a 3D space into voxels or cubes, and it iterates and assigns a mesh for each

Algorithm 1: Collision Detection

```
Data: Design grid Ω_g
Result: Updated grid points l as a linked list
foreach frame do
    if sculpting tool moves then
        Ω_t ← bounding box of the tool;
        Ω ← Ω_t ∩ Ω_g;      /* broad phase */
        foreach grid point p in Ω do
            d_p ← TSDF of p; /* narrow phase
              */
            if d_p is different from the old value then
                Append p to l;
            end
        end
    end
end
```

cube according to the TSDF values of the cube's eight vertices. If we have a cubic design grid of length n, the time complexity will be $\Theta(n^3)$. So there will be a cubic growth of computation cost when the user scales up the resolution, resulting in an uncomfortable user experience.

To cope with this issue, we adopt a just-in-time update approach. The idea is that the application updates a cube only when its related grid points have TSDF value changes. We modify the collision detection to keep track of updated grid points in a linked list. Note that each grid point corresponds to 8 cubes less the boundary edge cases. So the optimized Marching Cubes first translates the list of updated grid points to a hash set of cubes. After that, it iterates over all updated cubes and reconstructs their meshes. To store the meshes, we add a hash table as the buffer. The hash table maps a cube's index to its mesh. In this way, meshes are updated independently from cubes to cubes. The reasoning behind using a hash table is that most cubes in the design grid are empty; therefore, it saves both time and space complexity from storing meshes in a sparse representation. The last step is to trivially concatenate entries in the hash table into a vertex list and an index list. The two lists are then copied to GPU vertex and index buffers for the redraw procedure. The complete routine is shown in Algorithm 2.

Outcome With the aforementioned optimizations, the latency is largely reduced. Our application can work without any latency up to resolution 256x128x128, when the sculpting tool is as large as a hand. The resolution is even higher when user sculpts in a finer resolution since the computation cost is not related to the design grid size but proportional to sculpting tool size.

The diagram in Fig. 6 shows an overview of the application's workflow. The user's inputs are; hand gestures triggering the collision detection mechanism, voice commands

Algorithm 2: Optimized Marching Cubes

Data: Hash table h mapping cubes to mesh;
Updated grid points l as a linked list
Result: Vertex list v; Index list i
if $l \neq \emptyset$ **then**
  $q \leftarrow \emptyset$;
  foreach *grid point* p *in* l **do**
    Add neighbor cubes to q;
  end
  foreach *cube* c *in* q **do**
    $h[c] \leftarrow$ Marching cubes on cube c;
  end
  $v \leftarrow []$;
  $i \leftarrow []$;
  foreach *mesh* m *in* h **do**
    Append m to v;
    Append count(v) to i;
  end
end

triggering sculpting tools changes or other utility functions, and slider updates that control the sculpted object's RGB color property. The Marching Cubes remeshing algorithm is called only when there have been updates of GridPoints' truncated distance values from the sculpting primitives. Finally, the 'redraw' function is called only when there have been updates of the mesh from the Marching Cubes remeshing or on the sliders controlling the color of the mesh.

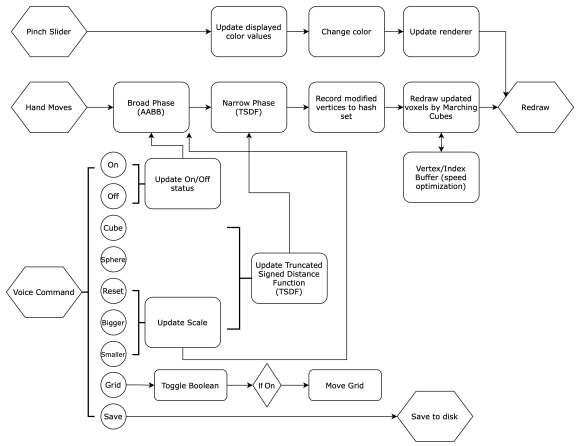

Figure 6. The workflow diagram of *Sculpting Gestures*

4. Results

The methods described in chapter 3 resulted in the AR application *'Sculpting Gestures'*. It is an intuitive application for the Hololens 2 for users with no prior experience of AR or VR technologies. The user loads the application and

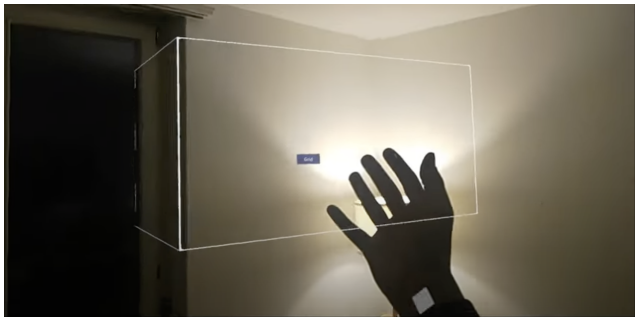

Figure 7. Re-position the design space in the room to start sculpting.

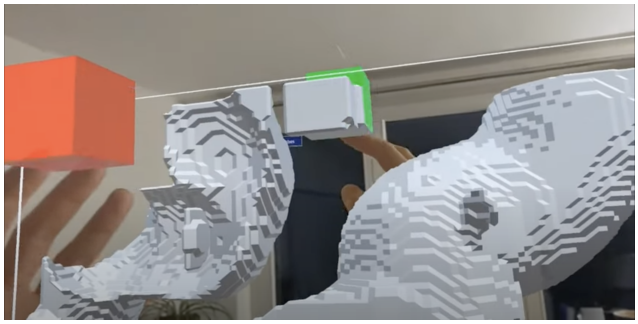

Figure 8. Shaping the general form of the object using large tools.

sees a default design space in the shape of a cube. They can then reposition or resize the default design space according to their liking. This allows the users to fit their design object within a certain spatial context to design/sculpt on-site (Fig. 7).

The app features sculpting tools that are available through voice commands. The tools only perform two actions: add material or remove material. The latter can be identified by their color (green for add and red for remove). Using hand tracking on Hololens, the "add" tool is attached to the right index finger while the "remove" tool is attached to the left index finger. The shape and size of tools can also be controlled through voice commands. For example, the user can change from spheres to cubes and make them bigger or smaller, as described in the method section. Although the design space has a defined resolution, adjusting the sculpting tools allows for better control over the geometry. That means that bulk gestures to define the overall shape of the sculpting object can be done quickly with larger tools while details can be added or subtracted by downsizing the tools to the desired detailing size (Fig. 8, Fig. 9). This allows the user to have a similar sequence of gestures as they would interact with a physical material in the real world like clay or marble. They would first form a bulk of material that approximates the general volume of the object they intend to make and then refine the form using more elaborate tools.

Figure 9. Creating details using small tools.

The user can also change the color of the displayed object in the app using three sliders for RGB colors. This is particularly useful if color is part of the design of the object and if the object will be manufactured with a 3D printing method that allows coloring like FDM, SLS, or MJF technologies or simply it will be color coated after its production.

After the user is satisfied with their creation, they can export a digital copy in an .obj file, a common file format that can be read by most CAD, visualization, or 3D printing software.

5. Conclusion

5.1. Contribution

Our application uses AR technology that allows users to sculpt beyond the screen through natural bare-hand gestures. It is intuitive, fun, playful, that encourages creative exploration and experimentation, especially suitable for artistic work. It seamlessly integrates the virtual realm with our physical environment as if one were a natural extension of the other. With sound control, the user doesn't need to remember complicated gestures, thus making our application more easy-to-learn.

5.2. Next Steps

Gestures. Some functions are only available through voice detection for the moment, which can be implemented by both gesture and voice. Also, gestures for tool transitioning and object manipulation can be further explored in the next step. For example, toggle the tools, scale the brushes, rotate the brush, undo/redo, etc.

Features. More features can be added to our application. One feature is the undo and redo function. Currently, there's no error correction and mistake recovery function. Thus, the last few history states should be saved. After every modification, the grid resolution and value changes should be kept in memory. Gestures should not be saved because they are memory uneconomical. When the user wants to undo, there should be a logical reverse of value changes.

Materials. Texture can give the user a sense of material about the sculpture. We have experimented 2D texture mapping in our application, which turned out to be the wrong direction in volume rendering. Thus, we propose implementing 3D texture rendering in the future.3D texture is a bitmap image containing information in three dimensions rather than the standard two, commonly used in volumetric modeling to store animated textures and blend between them smoothly. 3D texture will solve the 2D unwrapping problem and give better visual results.

Haptic Input. Input devices, such as haptic gloves, can deliver the sensation of touch during the sculpting process. For example, the user would feel the fingertips vibrating with a different intensity depending on the strength of the contact, which makes the sculpting experience more immersive and realistic.

References

[1] Incidental Space, 2016 Venice Biennale. 1

[2] Gravity sketch - 3d design and modelling software, 2021. 2

[3] Autodesk, INC. Mudbox, 2016. 2

[4] Autodesk, INC. Meshmixer, 2018. 2

[5] Autodesk, INC. Maya, 2019. 2

[6] Blender Online Community. Blender - 3d modelling and rendering package, 2018. 2

[7] P. Hackett D. Skillman. Tilt brush, 2016. 2

[8] Foundry. Kanova a vr sculpting app, 2018. 2

[9] Sarah Frisken and Ronald Perry. Designing with distance fields. pages 58–59, 01 2005. 2

[10] Woo W. Wakefield G. Jang SA., Kim H. Airsculpt: A wearable augmented reality 3d sculpting system. *Distributed, Ambient, and Pervasive Interactions. DAPI 2014. Lecture Notes in Computer Science*, 8530, 2014. 2

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

[12] MRTK, Mixed Reality ToolKit. 2

[13] Pilgway. 3d coat, 2021. 2

[14] Pixologic. Zbrush - 3d sculpting software, 2021. 2

[15] Ultraleap. Freeform, 2021. 2

[16] Unbound. Alpha - collaborative 3d design tools for teams, 2018. 2

[17] Unity Technologies. Unity, 2019. 2