




Working with Volumetric Meshes in a Game Engine: a Unity Prototype

Luca Pitzalis^{1,2}, Gianmarco Cherchi¹ , Riccardo Scateni¹  and Lucio Davide Spano¹ 

¹University of Cagliari, Department of Mathematics and Computer Science, Italy

²Visual Computing, CRS4, Italy

Abstract

Volumetric meshes are useful assets in many different research and application fields, like physical simulations, FEM or IGA. In the last decade, the Computer Graphics community dedicated a lot of effort in studying and developing new algorithms for the creation, manipulation, and visualization of this family of meshes. In the meantime, Game Development became a relevant field of application for CG practitioners entangled with AR and VR techniques. In this work, we position ourselves at the confluence of these two broad research and development paths. We introduce a custom data structure aiming at using volumetric meshes in Unity. To this purpose, we combine gaming techniques and interactions with typical operations of volumetric meshes. Besides this, to make the researcher experience more realistic, we also introduce features to manipulate volumetric meshes for their projects in an immersive environment using VR techniques. We think this feature can be useful in developing tools for 3D Sculpting or Digital Fabrication.

CCS Concepts

• **Computing methodologies** → *Rendering; Physical simulation; Volumetric models; Mesh geometry models;* • **Human-centered computing** → *Interaction techniques;*

1. Introduction

Meshes are the ubiquitous objects populating a game scenario. In a typical video-game, meshes represent characters, props, and scenes. Depending on the type of setup, we use either triangular or quadrilateral meshes, but they represent the visible items' skin in any case. Game engines are becoming more and more efficient, and the same applies to game platforms. But more complex objects and actions can, possibly, appear in the future in the game scenario for getting a higher level of (perceived) fidelity in games. Think about the physics: it evolved, in the last decades, adding dramatic effects, like fragmentation, fog, or particles. In contrast to such an evolution, we represent the objects through the same kind of meshes used thirty years ago, even if they are bigger, textured, sometimes deformed, and more realistic. What could be a step ahead in realism in games, then? The possibility to represent objects as skins and whole volumes could improve the range of effects to include. Object cuttable, or deformable with volume preservation could appear in the game scenario in the next future, even if not around the corner of the commercial video games. But this will happen only if the representations improve adding volumetric meshes to the tools that game developers can use in a game engine. Stemming from this idea, in this paper, we propose a proof of concept to show how Unity, the most common game platform, can incorporate volumetric objects. In the following, we will mainly describe the definition of a simple data structure to store volumetric

meshes and implement two essential features: cutting and heating. Based on the consideration that Unity is not only a game development platform, but it also became a toolbox for broader use in computer graphics, we also present another possible use of our system as a geometry processing tool. This topic gained interest in the research community (see, for instance, the work in [Sch20]). We introduce an interactive three-dimensional sculpting tool that, while the user in an immersive environment interacts with the object, keeps track of the fabrication properties. We incorporate the rudimentary semantics of additive and subtractive fabrication in the tool to give hints to the user on the possible outcome of the whole process, down to the real object's production.

2. State of the art

2.1. Volumetric Mesh Interaction

The introduction of consumer devices tracking gestures fostered the research on interactive methods for editing 3D models using direct manipulation methods, relying on the user's hand movements or tracking 6 degrees of freedom remotes. One of the most researched metaphors for supporting such interaction is the clay modelling: the user changes the shape and the volume of the 3D object adding or removing material and using tools while the object is rotating on a lathe. In this specific field, different solutions employ volumetric meshes, for better mapping the manipulation gestures to their ef-

fects on the object shape and volume.

For instance, Cho et al. [CBB*14, CHB12a] focused on a high-fidelity simulation of the clay modelling metaphor, including the simulation of tools. But the fidelity to the metaphor also represents the main work limitation since the spinning-wheel technique allows for creating only symmetrical shapes. Volarevic et al. [VMM15] propose a Kinect-based system removing such restriction and exploiting the clay modelling metaphor. Other works apply the same metaphor using different input devices such as remotes [Kre13] or finger tracking cameras such as the Leap Motion [PCLC17].

The paper [DCK13] introduced a different approach based on adapting typical CAD interactions to the gesture modality. The tool supports constructing and deforming solids through CAD tools, which limit the gesture expressiveness to the techniques available in 2D interaction.

Our main goal is including volumetric meshes in a game engine. Wang et al. [WAK18] present the first attempt to include this family of meshes in Unity, but it limited to tetrahedral meshes, without adjacencies information.

2.2. Game Development

Given the platform we considered for developing our prototype, it's worth summarising the usage of volumetric meshes in videogames implementation. The full list is available in [Wik20]. Here we report some of the main examples. The real-time strategy game *Command and Conquer* exploits voxels for representing most of the vehicles. *Robocraft* uses the same approach again for representing vehicles, this time for modelling the loss of pieces during robot fighting. *Minecraft*, differently from what it seems for its graphics, uses a voxel representation only for storing the terrain, but it exploits surface meshes for the gameplay. *Worms 4: Mayhem* uses a voxel-based engine to simulate land deformation similar to the older 2D *Worms* games.

2.3. HCI for Fabrication-Oriented 3D Modeling

Traditional 3D modelling approaches rely on standard mouse and keyboard input and 2D screen output, requiring different operations to fill the mismatch between the object and its representation. Such limitation represents a barrier for artists and craftsman, as they usually are not able to use even the simplest 3D modelling tools (e.g., SketchUp).

HCI research has provided 3D gestural input metaphors for shaping 3D objects, focusing on specific domains such as clay modeling [CHB12b] or dress tailoring [WSMI12]. The research on obtaining a 3D output resulted in different mixed reality systems, mostly applied to furniture design [WLK*14] or sculpting [YZY*17].

All these approaches exploit a strict separation between the design and the fabrication phase, iteratively repeating them until the user gets the desired result. To limit the time and material consumption involved in these iterations, different low-fidelity fabrication techniques exist in the literature (e.g. faBrickator [MMG*14], WirePrint [MIG*14], Plantener [BGM*15]).

Other solutions exploit a simple 3D preview after editing the object through a 2D interface, for instance, CutCAD [HTL*18]. More

recent solutions apply a compositional approach for the volume-building based on boxes that allow the construction of closed box structures that are possible to unfold in 2D for cutting [BSK*19] automatically. However, how to move from such phases separation towards a fully interactive fabrication environment as envisioned by Willis et al. [WXW*10] and applying the Shneiderman's direct manipulation principles remains an open research challenge [BM16]. Early solutions exploit the space physical sketching tools [AUK*15] or the coordination between the modelling tool and the printing process [PWMG16].

3. Contribution

The main contribution of this paper is a set of assets (in the form of C# scripts) to store, manipulate and visualize both tetrahedral and hexahedral meshes in the Unity Game Engine. We implemented and tested the mesh slice functionality in a VR environment (Oculus Rift and Hololens) to test the feasibility of real-time manipulation of volumetric objects in videogames. Finally, we implemented a basic 3D sculpting tool with VR interaction and real-time fabricability check. We also perform a user test to validate the usability of the implemented rudimental interaction techniques.

4. Data Structure

To make the Unity Game Engine compatible with the volumetric meshes, we implemented three different C# scripts to represent the tetrahedral and the hexahedral meshes. The work in [CPFS19] widely inspires the structure of the code. In particular, we have three main data structures (Abstractmesh, Tetmesh and Hexmesh) organized as follow: The Abstractmesh contains all the attributes and the methods shared between the two volumetric mesh types. The vertices are expressed as a List of Vector3 containing three floats coordinates for each vertex. We represent the simplices as vertex ids, referring to the vertices list. We use the same approach to represent the faces. Finally, we have additional lists to represent adjacencies between the mesh elements. Tetmesh and Hexmesh are subclasses of Abstractmesh. They implement the methods necessary to extract the surface of the mesh and to render it through the components we describe in Section 5.

Since Unity does not natively support volumetric meshes, we added to our data structure all the methods required to make our meshes fully compatible with the Unity Mesh component. To do so, we extracted the surface of the volumetric mesh. If the mesh surface is composed of quads, then we proceed to triangulate it. The Unity Mesh component, in fact, can handle only triangular faces expressed as arrays of nx3 integers.

Then the mesh is visualized by a Mesh Filter. It is also possible to apply materials to the mesh through a Mesh Renderer.

5. Usage

Unity assets are usually imported by dragging and dropping a supported file into the asset folder, or directly into the scene editor. In our case, Unity does not support the volumetric mesh files, so we cannot exploit the drag and drop feature.

We tried to make the import operation as straightforward as possible. To import a .mesh file, the first operation is creating a Unity

Empty GameObject in the desired position into the scene editor. There are three fundamental components required to make importing volumetric mesh possible. The first one is one of our data structures (Tetmesh or Hexmesh), and the second one is a Mesh Filter, which is a Unity component that takes as input a surface mesh and renders it through another component called Mesh Renderer. Once all the required components are attached to the Empty Game Object, it is possible to specify the path of the mesh in the apposite text field provided as a public attribute by our data structure (see Figure 1). It is also possible to add only our data structure as a component of a GameObject. In this case, the environment adds all the components described above. Our data structure also supports materials. It's possible to drag and drop a Material in the dedicated field and, if no material is specified, then the script will use a default one.

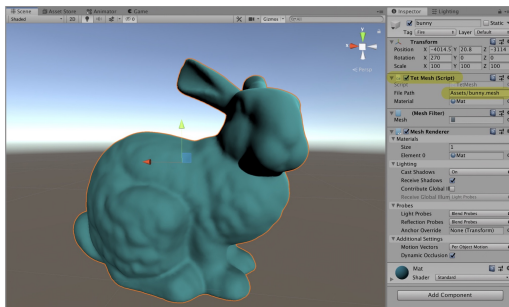


Figure 1: Example of importing a volumetric mesh in Unity.

Other functionalities, like mesh slicing and mesh heating, can be included by adding the corresponding scripts as components of the GameObject. For example, we implemented the slicing functionality we proposed in the script VolumeMeshSlicer that presents three sliders in the editor for slicing the mesh in the three axis-aligned directions. The slicing functionalities can be used both in the editor, as shown in Figure 2, and in the game.

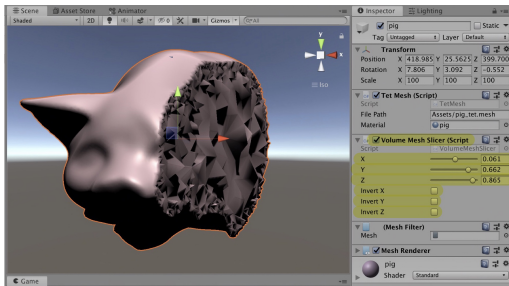


Figure 2: Slicing a mesh using the Unity inspector.

6. Basic Sculpting Tool for Fabrication

We implemented a prototype sculpting tool to test the interactive support provided by our data structure in an interactive fabrication task (see Section 2.3). At the moment, this demo works only on hexahedral meshes. We start from a single hexahedron mesh, which we extrude to create quite complex volumetric shapes. With

this feature, the user can create an entire simple game environment, from the characters to the scene objects, with volumetric properties. Figure 3 shows an example of this class of object produced with our demo.

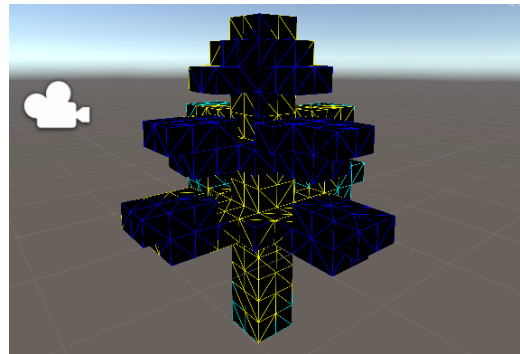


Figure 3: From a single cube to a volumetric tree model.

We also tested all these functionalities in a Mixed Reality setting, trying to give the user a simple and intuitive way to interact with the sculpting environment. In particular, we used Hololens, but it is possible to use a generic VR/MR Headset (i.e., Oculus Rift). In our demo, the user can grab a portion of the mesh surface and pull it. In this way, she can extrude the mesh, generating the inner volume in real-time.

The extrusion pipeline consists of three main functionalities and relative gestures. The first one is the selection of the faces we want to extrude. The users can point the desired faces with their index finger and select them by closing the finger to the hand. To realize the proper extrusion, the user can hold his finger close to the hand and pull the face with a hand movement. Before performing the real mesh modification, the tool shows a preview of the extrusion operation to the user.

At each step of the sculpting process (i.e., after each extrusion operation), we perform a very basic fabricability test. In particular, we perform a local manifoldness and self-intersection check to give the user an idea of the fabricability of the resulting model at the current time. If the fabricability test is not passed, an undo operation on the last change is possible. Of course, we know this test is not exhaustive for the model's complete fabricability condition, and we will face it in the future according to the fabricability requirements described in [LEM*17].

In order to assess the effectiveness of the proposed interaction, we run a preliminary qualitative evaluation of the modelling support when used in a Mixed Reality setting. We deployed the demo sculpting application on a Microsoft Hololens v1 [Mic19] Mixed Reality headset. For overcoming its limited support to interactive gestures, we paired it with a Leap Motion [Mic20] sensor for a fine-grained hand tracking.

The test consisted of two parts. In the first one, each participant learnt how to use the modelling features through a set of basic tasks, consisting of applying a single application feature for obtaining a simple table shape:

1. Increasing the mesh resolution;
2. Extruding the voxels at the corners of one hexahedron face;



Figure 4: Modelling goals for the user test.

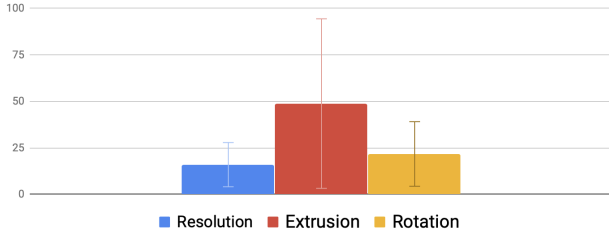


Figure 5: Part 1 - SMEQ [ZVD85] results for the manipulation features evaluated in the first part of the user test. Error bars represent the standard deviation.

3. Rotating the mesh using the grab gesture;
4. Zooming in and out the mesh using the pinch gesture;
5. Saving the result.

In the second part, we provided the users with a target model they had to replicate, starting from a single hexahedron. The target was a step pyramid requiring the extrusion of portions consisting of a decreasing set of voxels. Figure 4 shows the resulting objects for each task.

In the first part, we collected the task load through the SMEQ [ZVD85] questionnaire for each task, consisting of a single rating in a 0 to 150 scale. In the second part, we requested the participants to fill the NASA TLX [HS88] questionnaire and we collected the time spent on the task. Finally, we asked the participants to provide qualitative ratings of the interactive features and also open-ended comments for improving the application, which helped us in identifying possible improvements.

Ten people participated in the evaluation, nine males and one female, aged between 22 and 28 years old. They had different instruction levels, ranging from high school (2), bachelor (3), master (3) and PhD (2). All participants are familiar with 3D applications such as videogames, but none had previous 3D modelling experience. None of them previously used Mixed Reality applications or gestural interfaces. Only two of them used a 3D printed in the past. All users but one completed all the tasks. We registered the only failure in the second part. The lack of haptic feedback and some errors in the hand tracking increased the number of errors during the interaction but, in general, the participants liked the support and expressed positive opinions on the overall experience.

Figure 5 shows the detailed results for the manipulation features evaluated in the first part of the test.

In the second part of the test, we show the participants a target model to re-create through the application. The task took about 10 minutes to complete ($\bar{x} = 10.3$ min, $s = 3.2$ min). We measured the

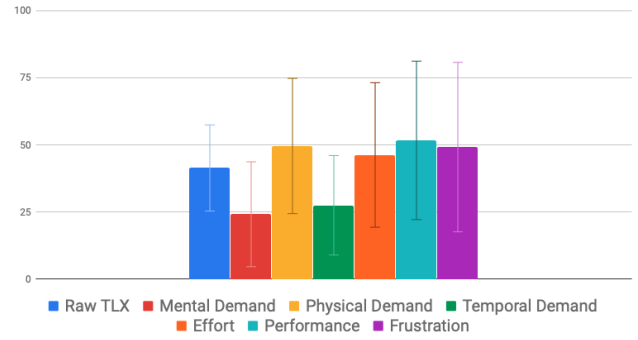


Figure 6: Part 2 - NASA TLX [HS88] results for the modelling task in the second part of the user test. We report the raw index value and the six dimensions that contribute to the overall task load.

task load using the NASA TLX [HS88], for analysing the factors that contributed to this load. The low values for the physical and mental demands, together with the results for the frustration and performance denote that the users were able to establish the intention, but they had some problems in executing the actions. Considering that we evaluated a preliminary prototype, this is encouraging: the users can establish what to do, and they can conclude the task, but we need to improve the interactive support.

The physical effort and the task duration indicate that we need to carefully consider the gorilla arm problem in the gestural interaction for this task: the time spent on the task is long enough for tiring the user's arms. Users may rest the elbows on a table, but this would make it difficult to interact with the lower part of the model. However, the raw index value shows that the task difficulty is already acceptable, but it is possible to improve it ($\bar{x} = 41.38$ $s = 16.03$ in a 0 to 100 scale, the lower, the better).

7. Conclusion and future work

We presented work at the convergence of two fields not strictly related to each other: mesh processing and interactive environments generation. Our goal was to demonstrate that not so much effort is needed to incorporate hypothetically sophisticated features, like volumetric meshes, in a game engine.

Our current proposal, as we stated in the beginning, is a proof of concept. We merely wanted to show that mixing up knowledge in mesh processing and game development can improve the toolbox available in a powerful and popular environment like Unity. This proposal is the cornerstone for a whole set of possible future evolution. First of all, we would like the real game developers to incorporate our rudimentary tools in their games to ensure that the enhancements we foresee are valuable. We would then like to focus on the fabrication guidance tool. It is interesting to interactively sculpt an object in a simple yet complete and user-friendly environment like one that Unity can generate. Full control of the sculpted object's fabrication can be a driver for digital artists that can, at the end of the work, obtain a real object in different materials using additive or subtractive fabrication.

Acknowledgement

Gianmarco Cherchi gratefully acknowledges the support to his research by PON R&I 2014-2020 AIM1895943-1 (<http://www.ponricerca.gov.it>).

References

- [AUK*15] AGRAWAL H., UMAPATHI U., KOVACS R., FROHNHOFEN J., CHEN H.-T., MUELLER S., BAUDISCH P.: Protopiper: Physically sketching room-sized objects at actual scale. *UIST '15*, Association for Computing Machinery, p. 427–436. URL: <https://doi.org/10.1145/2807442.2807505>, doi:10.1145/2807442.2807505. 2
- [BGM*15] BEYER D., GUREVICH S., MUELLER S., CHEN H.-T., BAUDISCH P.: Platener: Low-fidelity fabrication of 3d objects by substituting 3d print with laser-cut plates. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), CHI '15, Association for Computing Machinery, p. 1799–1806. URL: <https://doi.org/10.1145/2702123.2702225>, doi:10.1145/2702123.2702225. 2
- [BM16] BAUDISCH P., MUELLER S.: Personal fabrication: State of the art and future research. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2016), CHI EA '16, Association for Computing Machinery, p. 936–939. URL: <https://doi.org/10.1145/2851581.2856664>, doi:10.1145/2851581.2856664. 2
- [BSK*19] BAUDISCH P., SILBER A., KOMMANA Y., GRUNER M., WALL L., REUSS K., HEILMAN L., KOVACS R., RECHLITZ D., ROUMEN T.: Kyub: A 3d editor for modeling sturdy laser-cut objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2019), CHI '19, Association for Computing Machinery, p. 1–12. doi:10.1145/3290605.3300796. 2
- [CBB*14] CHO S., BAEK D., BAEK S., LEE K., BANG H.: 3d volume drawing on a potter's wheel. *IEEE Computer Graphics and Applications* 34, 03 (may 2014), 50–58. doi:10.1109/MCG.2014.3. 2
- [CHB12a] CHO S., HEO Y., BANG H.: Turn: A virtual pottery by real spinning wheel. In *ACM SIGGRAPH 2012 Emerging Technologies* (New York, NY, USA, 2012), SIGGRAPH '12, Association for Computing Machinery. URL: <https://doi.org/10.1145/2343456.2343481>. 2
- [CHB12b] CHO S., HEO Y., BANG H.: Turn: A virtual pottery by real spinning wheel. In *ACM SIGGRAPH 2012 Emerging Technologies* (New York, NY, USA, 2012), SIGGRAPH '12, Association for Computing Machinery. doi:10.1145/2343456.2343481. 2
- [CPFS19] CHERCHI G., PITZALIS L., FRONGIA G. L., SCATENI R.: The Py3DViewer Project: A Python Library for fast Prototyping in Geometry Processing. In *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference* (2019), The Eurographics Association. doi:10.2312/stag.20191374. 2
- [DCK13] DAVE D., CHOWRIAPPA A., KESAVADAS T.: Gesture interface for 3d cad modeling using Kinect. *Computer-Aided Design and Applications* 10, 4 (2013), 663–669. 2
- [HS88] HART S. G., STAVELAND L. E.: Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52. Elsevier, 1988, pp. 139–183. 4
- [HTL*18] HELLER F., THAR J., LEWANDOWSKI D., HARTMANN M., SCHOONBROOD P., STÖNNER S., VOELKER S., BORCHERS J.: Cutcad - an open-source tool to design 3d objects in 2d. In *Proceedings of the 2018 Designing Interactive Systems Conference* (New York, NY, USA, 2018), DIS '18, Association for Computing Machinery, p. 1135–1139. URL: <https://doi.org/10.1145/3196709.3196800>, doi:10.1145/3196709.3196800. 2
- [Kre13] KREYLOS O.: A developer's perspective on immersive 3d computer graphics, 2013. Online, Accessed 2020-09-28. URL: <http://doc-ok.org/?p=493>. 2
- [LEM*17] LIVESU M., ELLERO S., MARTÍNEZ J., LEFEBVRE S., ATTENE M.: From 3d models to 3d prints: an overview of the processing pipeline. *Computer Graphics Forum* 36, 2 (2017), 537–564. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13147>, arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13147>, doi:10.1111/cgf.13147. 3
- [Mic19] MICROSOFT: Hololens (1st gen) hardware, 2019. Online, Accessed 2020-09-28. URL: <https://docs.microsoft.com/en-us/hololens/hololens1-hardware>. 3
- [Mic20] MICHAEL BUCKWALD, DAVID HOLZ: Leap motion developer, 2020. Online, Accessed 2020-09-28. URL: <https://developer.leapmotion.com>. 3
- [MIG*14] MUELLER S., IM S., GUREVICH S., TEIBRICH A., PFISTERER L., GUIMBRETIERE F., BAUDISCH P.: Wireprint: 3d printed previews for fast prototyping. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2014), UIST '14, Association for Computing Machinery, p. 273–280. URL: <https://doi.org/10.1145/2642918.2647359>, doi:10.1145/2642918.2647359. 2
- [MMG*14] MUELLER S., MOHR T., GUENTHER K., FROHNHOFEN J., BAUDISCH P.: Fabrication: Fast 3d printing of functional objects by integrating construction kit building blocks. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2014), CHI EA '14, Association for Computing Machinery, p. 187–188. URL: <https://doi.org/10.1145/2559206.2582209>, doi:10.1145/2559206.2582209. 2
- [PCLC17] PARK G., CHOI H., LEE U., CHIN S.: Virtual figure model crafting with vr hmd and leap motion. *The Imaging Science Journal* 65, 6 (2017), 358–370. 2
- [PWMG16] PENG H., WU R., MARSCHNER S., GUIMBRETIERE F.: On-the-fly print: Incremental printing while modelling. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2016), CHI '16, Association for Computing Machinery, p. 887–896. URL: <https://doi.org/10.1145/2858036.2858106>, doi:10.1145/2858036.2858106. 2
- [Sch20] SCHMIDT R.: Command-line mesh processing with unreal engine 4.26, 2020. Online, Accessed 2020-10-28. URL: <http://www.gradientspace.com/tutorials/2020/9/21/command-line-geometry-processing-with-unreal-engine>. 1
- [VMM15] VOLAREVIĆ M., MRAZOVIĆ P., MIHAJLOVIĆ Ž.: Freeform spatial modelling using depth-sensing camera. In *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (2015), IEEE, pp. 318–323. 2
- [WAK18] WANG K., ADIMULAM K., KESAVADAS T.: Tetrahedral mesh visualization in a game engine. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (2018), pp. 719–720. doi:10.1109/VR.2018.8446544. 2
- [Wik20] WIKIPEDIA: Voxel - computer games, 2020. Online, Accessed 2020-09-28. URL: https://en.wikipedia.org/wiki/Voxel#Computer_games. 2
- [WLK*14] WEICHEL C., LAU M., KIM D., VILLAR N., GELLERSEN H. W.: Mixfab: A mixed-reality environment for personal fabrication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2014), CHI '14, Association for Computing Machinery, p. 3855–3864. URL: <https://doi.org/10.1145/2556288.2557090>, doi:10.1145/2556288.2557090. 2
- [WSMI12] WIBOWO A., SAKAMOTO D., MITANI J., IGARASHI T.: Dressup: A 3d interface for clothing design with a physical mannequin. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction* (New York, NY, USA, 2012), TEI '12, Association for Computing Machinery, p. 99–102. doi:10.1145/2148131.2148153. 2

- [WXW*10] WILLIS K. D., XU C., WU K.-J., LEVIN G., GROSS M. D.: Interactive fabrication: New interfaces for digital fabrication. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2010), TEI '11, Association for Computing Machinery, p. 69–72. URL: <https://doi.org/10.1145/1935701.1935716>, doi:10.1145/1935701.1935716. 2
- [YZY*17] YUE Y.-T., ZHANG X., YANG Y., REN G., CHOI Y.-K., WANG W.: Wiredraw: 3d wire sculpturing guided with mixed reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2017), CHI '17, Association for Computing Machinery, p. 3693–3704. URL: <https://doi.org/10.1145/3025453.3025792>, doi:10.1145/3025453.3025792. 2
- [ZVD85] ZIJLSTRA F. R. H., VAN DOORN L.: The construction of a scale to measure subjective effort. *Delft, Netherlands 43* (1985), 124–139. 4