Signifier-Based Immersive and Interactive 3D Modeling

Andreas Bærentzen
Technical University of Denmark
janba@dtu.dk

Jeppe Revall Frisvad
Technical University of Denmark
jerf@dtu.dk

Karan Singh
University of Toronto
karan@dgp.toronto.edu

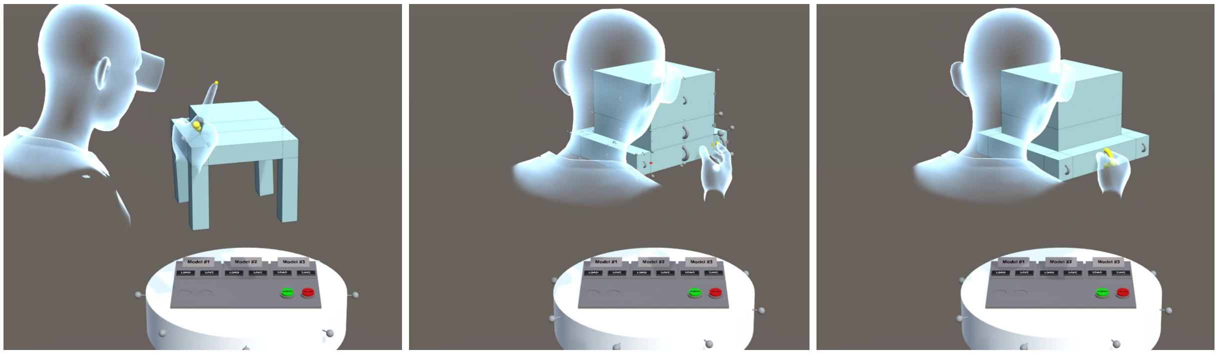

Figure 1: Left: a user moves a vertex by pulling its vertex handle. Middle: face latches are flicked to create a selection set. Right: the user subsequently pulls a face handle to perform a multi-face extrusion.

ABSTRACT

Interactive 3D modeling in VR is both aided by immersive 3D input and hampered by model disjunct, tool-based or selection-action user interfaces. We propose a direct, signifier-based approach to the popular interactive technique of creating 3D models through a sequence of extrusion operations. Motivated by handles and signifiers that communicate the affordances of everyday objects, we define a set of design principles for an immersive, signifier-based modeling interface. We then present an interactive 3D modeling system where all modeling affordances are modelessly reachable and signified on the model itself.

CCS CONCEPTS

• **Human-centered computing → Virtual reality**; • **Information systems → Multimedia content creation**.

KEYWORDS

3D Modeling, signifiers, affordances

ACM Reference Format:
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 (VRST '19), November 12–15, 2019, Parramatta, NSW, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3359996.3364257

1 INTRODUCTION

The possible ways by which a human can interact with a real world object are often called *affordances*, as described in Donald Norman's seminal "The Design of Everyday Things" [Norman 2002]. Parts of the object (e.g. a door handle) that communicate an affordance of the object to a user (e.g. pull to open the door) are known as *signifiers*. An affordance is a relation between the object and its user rather than simply a property of the object.

Most affordances transfer naturally from the physical to the virtual world, the main difference being in the haptics of the interaction between object and user. As the physical proxy for the virtual object is typically a VR controller, the mechanics of engaging an affordance on the object (like grabbing and wielding it) may be different, as will its weight (perceptually heavy objects may be wielded in VR with ease). The visual *signifiers*, however, for transferred affordances remain largely intact, and experience with similar signifiers in the real world makes the *discoverability* and *learning* of their affordances in VR easier [Norman 2008]. Affordances in VR with no physical counterpart are still able to borrow signifiers from real world objects that have a similar pattern of user interaction. Moreover, the mapping of all interactive functionality to signifiers or handles in the virtual scene keeps the user persistently immersed in the scene, and engaging with these handles further enhances a sense of presence.

We present a tool-free, modeless interface, where all operations are captured as affordances directly associated with the object being modelled or a virtual worktable (as exemplified in Figure 1).

2 RELATED WORK

In the VR book, Jerald [2016] discusses the importance of consistent affordances and unambiguous signifiers in the context of VR. The notion that well designed objects have signifiers that clearly signal their affordances [Norman 2002] is central to our work. Hutchins et al. [1985] posit that interfaces feel *direct* when they ease the cognitive load by giving the user a sense of interacting directly with the object itself. To a large extent, this is what we aim for as it pertains to manipulation of 3D objects [Strauss and Carey 1992]. 3D interaction has been surveyed by Jankowski and Hachet [2013] and very recently by Mendes et al. [2019]. In their terms, our interface is based on mid-air interactions, but via manipulators that are integral to the object being modeled as opposed to relying on surrogate objects [Strauss and Carey 1992].

Recently, there has been both research [Arora et al. 2018; Li et al. 2017] and industrial software in the space of 3D modeling via mid-air sketching [Tilt Brush 2018], and this has led to the development of a method for converting 3D sketched strokes to a manifold surface [Rosales et al. 2019]. In addition to 6DOF VR controllers, modeling techniques have explored the use of high DOF physical proxies like curve strips [Grossman et al. 2003], a sponge block [Sheng et al. 2006] and bare hands [Kim and Bae 2016].

The modeling style supported by our system is often referred to as *box modeling*. So called because the artist generally starts modeling from an initial hexahedron (cube) and proceeds to extrude and refine locally or subdivide. In box modeling, the flow of edges and face loops is typically very important, and nearly all faces are quadrilateral. All major 3D packages support box modeling, but some smaller applications, such as the free program Wings3D [2018], are more specifically designed for this type of work process. Other research [Zeleznik et al. 2007] and software such as SketchUp [2018] rely heavily on extrusion (push/pull in SketchUp) but would not be associated with box modeling because the emphasis is not on quadrilateral (dominant) meshes. There is also a large body of research in box modeling [Lipp et al. 2014].

Closer to the realm of VR, Minecraft [2018] is both a game and a modeling tool used for world building. In Minecraft, the geometry consists of grid aligned, mostly identical cubes. ProBuilder [2019] is a polygonal modeling tool for Unity with a broad palette of features. A VR interface is being developed for ProBuilder [ProBuilder VR 2019]. This seems similar in functionality to Minecraft modeling but without grid alignment. Compared to other VR modeling tools, our tool is perhaps most similar to Google Blocks [2018] which also allows the user to create simple polygonal models. In the Google Blocks system, the user holds a tool palette in one hand and, having selected a tool, manipulates the model with the tool. This is modal, while our approach is modeless. Another VR modeling tool is Medium from Oculus [2018] which is really a voxel based sculpting tool that does not directly expose users to polygons. Consequently, it has tools with very different affordances.

The academic work most similar to ours seems to be Mockup Builder by De Araújo et al. [2013]. Mid-air gestures are also used as a means of performing extrusion, but De Araújo et al. rely on contextual menus and a more abstract pinch gesture instead of signifiers. Moreover, the stereoscopic images are projected onto a surface which makes it natural to allow for curve drawing. This leads to a modeling style akin to SketchUp. Later, SketchUp itself was brought into VR by Mine et al. from Disney Imagineering [Mine et al. 2015] using dual tracked touchscreen devices for input. Recently, Mendes et al. [2017] studied a purely gestural technique for boolean operations in VR and compared it to a menu based solution. While interesting, these VR modeling tools all provide a mode of operation quite different from our signifier-based approach.

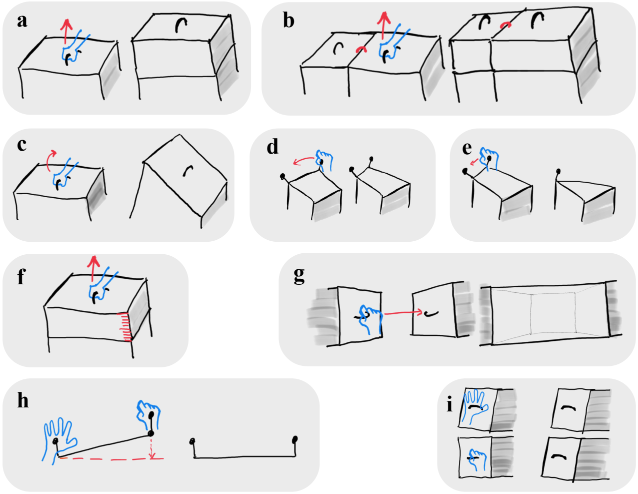

Figure 2: This figure illustrates how the affordances relate to signifiers. The affordances are a) single face extrusion, b) multi-face extrusion (using latches), c) face rotation and translation, d) vertex movement, e) vertex merging, f) extrusion with tape measure, g) face bridging, h) snapping a vertex to an axis-aligned line, i) aligning a face to the plane of another face.

3 A SIGNIFIER-BASED UI FOR 3D MODELING

The idea for the proposed system sprang from the observation that extrusion of a mesh face is analogous to pulling out a drawer. This led to the design of an interface where the 3D model becomes a slightly magical, mechanical device that can be manipulated in many ways that all cause its geometry and structure to reconfigure. Inspired by Norman [2002], we decided that rather than relying on abstract user interface components, we would let all mesh entities belonging to the 3D model (faces, edges, and vertices) be possessed of a signifier (see Figure 2) which signals the affordance associated with the mesh element.

Vertex handles enable the user to move vertices and to refine the 3D model. When a user grabs and pulls a vertex, the vertex is moved, making this tool effective for (coarse) edits. This function motivated the design of the vertex handle as a knob that does not signal in which direction force should be applied. This seems appropriate since the vertex can be moved in any direction using the handle. If a vertex is brought close to another vertex to which it is connected via an edge, the edge is collapsed and the two vertices are merged. Clearly, if an edge is collapsed, the two incident quads become triangles. These actions are shown in Figure 2 (d & e).

Two-handed interaction. Grabbing two vertices which are connected by an edge allows the user to introduce a new edge loop. If the user proceeds to bend the edge to a breaking point, a new edge loop is created by inserting vertices on all edges which separate faces belonging to the face loop of the "broken" edge.

Edge latches are used to create selection sets. Closing a latch associated with an edge joins the two faces in a selection set. If the handle of either face is grabbed, both faces are affected by the operation (rotation or extrusion). Several faces can be latched together to form larger selection sets. The latches are designed to look like they can close. This might not immediately lead the user to guess that it locks incident faces together, but it should invite the user to experiment.

Face handles enable the user to transform and extrude faces. If the handle is pulled, it will lead to an extrusion, and this is the reason for the design of the handle as a simple pull handle. However, the handle can also be used to reorient faces by rotation as illustrated in Figure 2 (c). Whenever the translational part of the motion exceeds a threshold an extrusion is performed as illustrated in Figure 2 (a). In (b), the figure shows what happens if the faces are separated by an edge with a closed latch: in this case both faces are extruded together. More generally, a set of faces all joined by closed latches will be extruded together if the user pulls on the handle of just one of these faces. Note that during the extrusion it is still possible to reorient the extruded face by rotating the handle.

While the face handles were quite effective for modeling of organic objects, we also realized that the system could be used to model regular structures. To facilitate this type of modeling, face handles are subjected to two types of snapping: orientation snapping and direction snapping. The latter means that when a face is extruded, the extrusion happens in the direction of the parent face normal unless the user pulls significantly in another direction. If so the snapping is disabled for the remainder of the operation. Angle snapping means that the orientation of the extruded face matches its parent unless the face is rotated significantly. In this case, angle snapping is disabled.

Unfortunately, it is somewhat difficult for most users to pull out a face very far without exceeding the snapping thresholds for both orientation and direction. To facilitate efficient modeling of regular structures, we lock the snapping if the user's thumb does not stick out. In the current implementation, the thumb sticks out if it does not rest on the controller. To allow users to measure out a particular length during extrusion, we render tick marks on the side of the extruded part as illustrated in Figure 2 (f). Clearly, these tick marks will be seen as a signifier which is arguably self explanatory. Once more, we wanted to aid the user in creating regular structures: if the user's thumb does not stick out, extrusions are in increments of a tick.

Symmetry by Sensing. Based on testing of the initial prototypes, we realized that we needed to provides users with a mechanism to make already created geometry more symmetrical. For instance, we may have a situation where the user would like to place a vertex such that it lies precisely on a line that is parallel to one of the main axes and passes through a different vertex. In a very similar scenario, it is also common that users would like to align a face with the plane containing another face.

These two related actions seemed fairly abstract, but a solution which relies on the existing signifiers (i.e. handles) presented itself. The basic idea is to think of the handles not only as instrumental but also as conveyers of information about their corresponding mesh entities. Whenever a user touches a vertex handle or a face handle with one open hand she effectively "senses" the information of the corresponding vertex (or face). If she proceeds to grab another handle of the same type (i.e. closing the other hand around this second handle) then the information is transferred and the grabbed vertex (face) is aligned with the sensed vertex (face). The interaction is illustrated in Figure 2 (h & i).

Bridging Faces. We realized that users might often want to conjoin two parts and that this could be quite naturally supported. Our solution is similar to the symmetry by sensing feature in that it relies on the existing signifiers. If a user starts an extrusion by grabbing and dragging a face handle and proceeds to drag until touching the handle of a different face, then the system does not perform an extrusion, but instead it removes both faces and creates a bridge that connects the resulting holes instead (Figure 2 (g)).

Global Operations and The Work Table. The work table serves two purposes. It provides a spatial anchoring of the 3D model such that it does not seem to float in the void, and it serves as a panel for global controls. This is necessary as some operations do not have a natural association with a single mesh element. These operations are loading and saving of 3D models, as well as undo and redo.

Under the work table, we find a solid disc with knob-handles that allow the user to rotate the disc. This disc provides the interface for rotation (grabbing a single knob and turning the disc) but also translation and scaling are possible by grabbing two knobs followed by motions analogous to the intended transformation: moving the handles synchronously to translate, and spreading or gathering the arms to scale.

3.1 Implementation

The system was developed using the Unity game engine [Unity 2019] and the GEL mesh library [GEL 2019]. The source code and Unity project can be obtained from https://github.com/janba/VR-modelling-system. We used the Oculus Rift CV1 and Oculus Touch controllers which make it easy to emulate hand interaction.

Having implemented the first prototype, we quickly realized that the signifiers can make the model seem rather cluttered. This is particularly true when the model is detailed. We countered this problem by introducing two features. The first feature is to remove all signifiers when the user lowers her hands to below the work table. In a sense this amounts to a viewing mode, but the mode selection is trivial: as soon as either or both hands are raised above the table, the signifiers pop back.

However, the handles and latches still feel large, and not all signifiers are relevant to the user. Moreover, the signifiers can be fairly small until the user needs to grab them. This led to the realization that we could simply scale the handles and latches according to the distance from either controller to the signifier. In the present implementation, the signifiers scale with the inverse of the square distance to a controller giving the user the sense that the signifiers grow to meet their hands as they approach. This feature feels similar to how icons scale in the MacOS Dock.

Several users for whom polygonal meshes are unfamiliar would tend to create polygonal soups during modeling by treating extrusions a bit like a brush in a 3D paint system. This would generally not lead to meaningful results. To dissuade users from creating models that do not make sense as polygonal meshes, we perform a simple test for intersections at the end of each modeling operation. If the user performs an operation that results in a self intersection, we simply roll back the model to just before that operation was commenced.

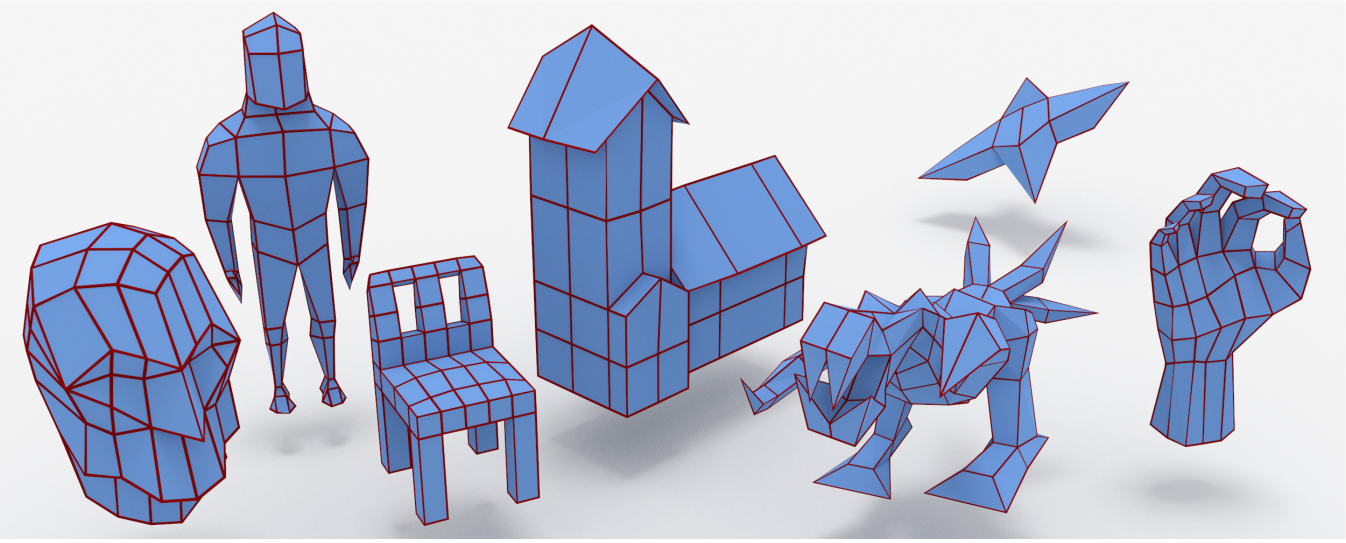

Figure 3: The seven models above were created over the course of several sittings by an expert user. The small models were created in a few minutes whereas the larger ones took up to half an hour to model.

4 DISCUSSION

Several users, mostly novice 3D sculptors with some VR experience, have tried our system. Based on their testing as well as our own use, we made a number of observations which we discuss in the following. Example creations are shown in Figure 3.

Coarse-to-Fine modeling. Digital box-modeling (in general, and our system is no exception) works best if the user models as much as possible at a low polygonal resolution before subdividing the model to add details. Users without this mindset easily become frustrated.

Surface topology. Our system undoes modeling operations that cause geometry to self-intersect, While this was sometimes viewed as *unforgiving*, it provided users with a playful way [Shneiderman 2007] to throw away a modeling operation by simply twisting the geometry into a self-intersecting mess. However, when users try to extrude adjacent faces sequentially with the intent of performing a multi-face extrusion, self-intersections also arise. Instead of undoing such operations, the system could transform them into multi-face extrusions and automatically latch the adjacent face to convey this to the user – in keeping with the *Principle of Least Astonishment* [Seebach 2001].

Feature-set and System Design. Our debutant users found the modeling feature set quite large, whereas advanced users missed design functionality like bevels and chamfers. Distance and angle thresholds for snapping and collapsing elements in our systems were too small for some users and prevented the creation of very fine detail for others. Ultimately, parameters for such a modeling system would warrant user customization.

Signifiers. While our handles work well as signifiers, further visual feedback of their operational impact on the model could be provided. For example, users suggested that closed latches (colored red) in our system, could also change the color of incident faces, to better reflect group membership. As another example, when sensing one vertex (or face) and grabbing another, visual feedback of the sensed handle can be provided near the grabbed handle, to ensure that the sensing hand does not drift away from the handle while the operation is performed.

Snapping and Symmetry. An important feature of our system is the bimanual control to aid snapping and symmetry in model creation. Many users were prolific in their use of this feature to produce self-symmetric parts of a model such as legs on a chair, or the slant of a roof on a house. At the same time, our models have free-form flexibility, unlike say Minecraft where regularity is enforced to an extreme on a grid. Some users expressed the desire for widget-like axis controlled manipulation (translations and rotations) of elements [Mendes et al. 2019]. Such control is readily supported by a signifier-based interface, the challenge, as with widgets, being the design of compact yet clear and controllable signifiers.

Work Table. We found the concept of a work table with various controls to transfer well from its physical analogue. In some cases, features such as load and save could be protected by, say, a safety cover or step to prevent inadvertent button pressing. Users frequently move and scale the model, and this was well supported by the affordances of the worktable. In general, the worktable can be augmented with affordances that provide global control over the modeling process.

Conclusion. Box modeling (and 3D modeling in general) is fairly challenging. That being said, the basic affordances in our system can be discovered in a few minutes, and for an experienced user, it lends itself to the effective creation of a range of model types. Going forward, we plan to generalize and consolidate the presented system. Replacing controllers with hand tracking is also an interesting avenue for future work.

Acknowledgements. The authors would like to thank Morten Nobel-Jørgensen, Valery Kamalov, Lars Kofoed Jensen, and Egill Ingi Jacobsen for work on the presented system in various stages of the development.

REFERENCES

Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. SymbiosisSketch: Combining 2D & 3D Sketching for Designing Detailed 3D Objects in Situ. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 185, 15 pages. https://doi.org/10.1145/3173574.3173759

Bruno R. De Araújo, Géry Casiez, Joaquim A. Jorge, and Martin Hachet. 2013. Mockup Builder: 3D modeling on and above the surface. *Computers & Graphics* 37, 3 (2013), 165–178.

GEL. 2019. https://github.com/janba/GEL.

Google Blocks. 2018. Google. https://vr.google.com/blocks/.

Tovi Grossman, Ravin Balakrishnan, and Karan Singh. 2003. An Interface for Creating and Manipulating Curves Using a High Degree-of-freedom Curve Input Device. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 185–192. https://doi.org/10.1145/642611.642645

Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. 1985. Direct Manipulation Interfaces. *Hum.-Comput. Interact.* 1, 4 (Dec. 1985), 311–338. https://doi.org/10.1207/s15327051hci0104_2

Jacek Jankowski and Martin Hachet. 2013. A Survey of Interaction Techniques for Interactive 3D Environments. In *Eurographics 2013 - STAR*.

Jason Jerald. 2016. *The VR Book: Human-Centered Design for Virtual Reality*. ACM and Morgan & Claypool, New York, NY, USA.

Yongkwan Kim and Seok-Hyung Bae. 2016. SketchingWithHands: 3D Sketching Handheld Products with First-Person Hand Posture. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 797–808. https://doi.org/10.1145/2984511.2984567

Yuwei Li, Xi Luo, Youyi Zheng, Pengfei Xu, and Hongbo Fu. 2017. SweepCanvas: Sketch-based 3D Prototyping on an RGB-D Image. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 387–399. https://doi.org/10.1145/3126594.3126611

Markus Lipp, Peter Wonka, and Pascal Müller. 2014. PushPull++. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 130.

Daniel Mendes, Fabio Marco Caputo, Andrea Giachetti, Alfredo Ferreira, and J. Jorge. 2019. A survey on 3D virtual object manipulation: From the desktop to immersive virtual environments. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 21–45.

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 *IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 154–157.

Mark Mine, Arun Yoganandan, and Dane Coffey. 2015. Principles, interactions and devices for real-world immersive modeling. *Computers & Graphics* 48 (2015), 84 – 98. https://doi.org/10.1016/j.cag.2015.02.004

Minecraft. 2018. Microsoft. https://minecraft.net.

Donald A. Norman. 2002. *The Design of Everyday Things*. Basic Books, Inc., New York, NY, USA.

Donald A. Norman. 2008. THE WAY I SEE IT: Signifiers, Not Affordances. *Interactions* 15, 6 (Nov. 2008), 18–19. https://doi.org/10.1145/1409040.1409044

Oculus Medium. 2018. Oculus. https://www.oculus.com/medium/.

ProBuilder. 2019. ProCore, Unity. https://unity3d.com/unity/features/worldbuilding/probuilder.

ProBuilder VR. 2019. ProCore, Unity. http://www.procore3d.com/probuilder-vr/.

Enrique Rosales, Jafet Rodriguez, and Alla Sheffer. 2019. SurfaceBrush: From Virtual Reality Drawings to Manifold Surfaces. *ACM Transaction on Graphics* 38, 4 (2019). https://doi.org/10.1145/3306346.3322970

Peter Seebach. 2001. The cranky user: The principle of least astonishment. *IBM DeveloperWorks* (2001).

Jia Sheng, Ravin Balakrishnan, and Karan Singh. 2006. An Interface for Virtual 3D Sculpting via Physical Proxy. In *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia (GRAPHITE '06)*. ACM, New York, NY, USA, 213–220. https://doi.org/10.1145/1174429.1174467

Ben Shneiderman. 2007. Creativity Support Tools: Accelerating Discovery and Innovation. *Communications of the ACM* 50 (December 2007), 20–32. https://doi.org/10.1145/1323688.1323689

SketchUp. 2018. Trimble. http://sketchup.com.

Paul S Strauss and Rikk Carey. 1992. An object-oriented 3D graphics toolkit. In *ACM SIGGRAPH Computer Graphics*, Vol. 26. ACM, 341–349.

Tilt Brush. 2018. Google. https://www.tiltbrush.com/.

Unity. 2019. https://unity.com/.

Wings3D. 2018. http://www.wings3d.com.

Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. 2007. SKETCH: An Interface for Sketching 3D Scenes. In *ACM SIGGRAPH 2007 Courses (SIGGRAPH '07)*. ACM, New York, NY, USA, Article 19. https://doi.org/10.1145/1281500.1281530