Multi-modal Non-linear Continuous 3D Presentations

Daniil Rodin and Gershon Elber
Technion, Haifa, Israel
E-mail: daniilrodin@cs.technion.ac.il, gershon@cs.technion.ac.il

Abstract. Computer presentations are an integral part of our lives, yet the contemporary presentation tools have not changed for about three decades, still emulating old overhead slide projectors. Combined with the rapid evolution of computer graphics technologies, this results in contemporary tools being ill-suited for many modern needs where linear sequences of fixed-sized 2D slides consisting of text and images is not nearly sufficient. In this work, we propose a presentation system aimed at overcoming the limitations of the contemporary presentation software. To achieve this goal, we abandon the usual concept of a presentation as a 2D slide sequence, and instead treat them as continuous automatically created 3D scenes of non-linear structure with multi-modal content, all without requiring any professional skills from the end user.

1. Introduction
Presenting new ideas and information is an important part of various human activities. At the same time, modern computer technologies provide vast powers when dealing with information. Despite that, presentation software is in stagnation for about three decades and is far from fully utilizing those technologies. Fundamentally, contemporary presentation abilities are more like old non-computer-based overhead slide projectors than what we see in modern digital narration-heavy media like movies and games. This limits not only the expressive power of the presentation, but also the types of content that can be presented in a clear way.

The limitations in contemporary presentation tools have attracted much criticism over the recent years, even to the point of claiming that “PowerPoint deals intellectual damage” [1], suggesting that “PowerPoint may edit our thoughts” [2], and popularizing the expression “Death by PowerPoint” [3].

One of the main criticisms of PowerPoint is that it is discrete. It forces the presenter to quantize everything into slides of fixed size and encourages everything inside a slide to be cut into bullets. As shown in [1], the size limits of a slide often cause omission of significant details, which, in turn, may cause dangerous mismanagement or even affect how we think [4].

Another major limitation of PowerPoint is the lack of native support for 3D content. The real world that surrounds us is three-dimensional, and therefore, the need to show 3D objects and phenomena arises more often than one may think: architecture, engineering/mechanical prototypes, medicine (CT/MRI and anatomy), biology (e.g., molecules), and many more. Note that these 3D data sets are already created by professionals on a daily basis. Furthermore, it is inconceivable from the PowerPoint perspective to create presentations as 3D scenes themselves, while such concept could lead to much more natural navigation as it is in the world we live in. While the construction of 3D scenes by novice users is considered challenging, navigation
Figure 1. Examples of using our presentation system, demonstrating the benefits of native 3D support and non-linear (non-sequential) navigation. The building on the right is generated automatically from an abstract story-graph (see Section 4.1). The white lines at the bottom right are the default generated paths between rooms.

in a 3D scene is quite common (e.g., in 3D games). Hence, herein, the 3D scenes are created automatically. Having the presentations be part of 3D scenes would also open the possibility for making virtual reality presentations more commonplace.

PowerPoint presentations have a flat structure (it lacks hierarchy). Other forms of storytelling like books and games, however, often tend to be structured in a hierarchical way (e.g., chapters, sections, paragraphs). Some of the other presentation systems also introduce a hierarchical structure (see Section 2.2). In addition to helping the end users better organize their presentations, a hierarchical structure is invaluable for automatic 3D scene generation for 3D presentations (see Section 4.1).

PowerPoint presentations also have a strictly linear (sequential) structure. However, it is often necessary to adjust the flow of the presentation to follow the audience interest or time limits. Non-linear structure would also allow presentations to be usable under different time limitations or for different audiences without the need to create various short/long or basic/advanced versions of the presentations, which need to be maintained separately. We, therefore, see it as beneficial to have an ability to create branches and shortcuts in the presentation structure.

In this work, we propose a presentation system that overcomes many of the aforementioned limitations, and thus opens the path to much more effective presentations that satisfy modern real-life needs by utilizing modern technologies. We are also aiming at minimizing the overhead on the end user (e.g., by using automatic synthesis of the 3D scenes). The main contribution of this work is, thus, providing a way of combining modalities (such as text, images, 3D geometry, simulation, interactivity, etc.) together in a user-friendly way for all those who need these modalities, including scientists, engineers, teachers, designers, and more.

The rest of this work is organized as follows. In Section 2, we explore the state of the art. In Section 3, we provide an overview of our system. In Section 4, we discuss in more detail its capabilities and how they make our system unique and beneficial, and we conclude in Section 5.

2. Related Work
For the reasons discussed in Section 1, the topic of presentations has always kept some attention of the academia and the industry. In this section, we review the effort that is done towards analyzing and overcoming the limitations of contemporary presentation systems. In Section 2.1, we discuss techniques that are uncommon in contemporary presentations, and in Section 2.2 we provide an overview of presentation systems utilizing these techniques.
2.1. Modern Presentation Techniques and their Benefits

While some researchers have investigated methods to improve the effectiveness of “plain old slides” (e.g., [5]), a lot of effort goes into designing new ways of presenting information and evaluating their effectiveness.

As discussed in Section 1, discreteness and lack of hierarchy are major limitations of PowerPoint-like software. To overcome these limitations, [6] has come up with the idea of Zoomable User Interface (ZUI). The idea of ZUI is to structure the content as a mind-map and place it on a giant 2D canvas with deeper nodes typically being much smaller than the ones closer to the root. This way, the spatial positioning and size of the content reflects its internal structure. Navigation is supported as continuous 2D translation, rotation, and zooming in and out, which makes the structure much more obvious to the viewer. ZUI-based presentations have also shown to have some (though not significant) advantage over PowerPoint in education [7, 8].

One step further from the hierarchical structure is a story-graph structure (discussed in detail in Section 3.3). In a story-graph, the story (or the presentation) is structured into nodes connected by arrows, thus forming a directed graph. Such structure is mostly found in games, and is defined either with game scripting (e.g., in KiriKiri (http://kirikirikag.sourceforge.net/) and RenPy (https://www.renpy.org/) game engines) or with a game visual editor (e.g., Twine (https://twinery.org/) and articy:draft3 (https://www.articy.com/en/)).

While the benefit of 3D support is self-evident for use cases that deal with inherently 3D objects and phenomena, researchers are investigating the general effect of the 3D modality on presentation perception, especially in education and training. The addition of a third dimension to a field where it is not required may introduce distraction and cognitive overload [9]. However, when used appropriately, 3D can improve perception of material [10]. The benefits are especially significant in the fields that encourage a ‘hands-on’ interaction with three-dimensional objects, such as biology, medicine [11], geometry, and engineering.

Utilizing modern real-time techniques in presentation software opens up possibilities for better support of animations. It is shown [12] that animation improves material perception when used correctly [13, 14]. Using animations is relevant when dealing with inherently continuous processes, which can be found in various areas.

Another modality that can be supported when utilizing modern real-time techniques is interactivity. Interactivity allows the presenter (or the student) to manipulate various objects, including the ability to transform, expose, or explore a 3D model during the presentation, to adjust the starting conditions of a simulation, and many more. But the most important aspect of interactivity is that it allows the learner to experiment with the material directly, which brings a whole new level of education process [15]. With support for interactivity, the presentation turns into a more effective learning tool by itself. Furthermore, according to [16], interactivity is significantly more effective as an education tool than a one-sided passive linear viewing/listening process, not to say the increased entertainment value of the presentation.

2.2. Presentation Systems Utilizing the Modern Techniques

The limitations of PowerPoint-like presentations have been known for some time now, and there are attempts to overcome those, at least partially. In this subsection, we make an overview of such presentation systems and discuss how the solutions are insufficient to reach the goal we are aiming at.

Prezi (https://prezi.com/) is the most well-known presentation system that breaks away from the concept of slides by embracing the ZUI paradigm. This solves the problems of discreteness and lack of hierarchy. However, Prezi is inherently 2D, and thus cannot achieve our goals of natively supporting 3D content on a fundamental level.

Emaze (https://www.emaze.com/) introduces 3D transitions, which helps with the perception of flow. All the content however is still a linear sequence of 2D slides of fixed size...
Aurora 3D Presentation (https://www.presentation-3d.com/) allows 3D content. The presentation is separated into independent hierarchical scenes, with the ability to simulate transitions within the same scene by moving the objects. However, all the 3D placement is done manually, which requires more professional skills from the end user than what we believe should be necessary.

CL3VER (http://www.cl3ver.com/) and Ventuz (http://www.ventuz.com/) are 3D presentation tools that offer vast capabilities, but are exclusively aimed at professional content creators. Furthermore, they still require manual placement of views, which, as we show in 4.1, can be done automatically, significantly reducing the workload, especially for novice users.

Aside from commercial products, there are presentation systems produced by the academia. Academic Presenter [17] expands on the ideas of Prezi by combining ZUI with regular slides and infographics (imagery, charts, and minimal text that gives an easy-to-understand overview of a topic) [18]. Classroom Presenter [19] heavily integrates digital ink into the learning process, also allowing sharing handwritten annotations between students. Both, however, are inherently 2D, and thus, do not meet our goals.

3. System Overview
In this section, we give an overview of our system and describe the concepts it is based upon.

3.1. Architecture Overview
Our main goals when designing the system architecture were extensibility, multi-modality and real-time support. Fulfilling these goals necessitates our system to be designed as a real-time rendering application at its core, which also bolsters such aspects as animation and interactivity.

An overview of the system is shown in Figure 2. The system has a small core (in red) that includes the minimal object database of the content as well as the story graph. The core is connected to the rest of the system through the extension mechanism, allowing us to add new back-ends (i.e., windowing and rendering frameworks) and support for new modalities. This architecture is discussed in more detail in the following subsections.

![Figure 2](image)

**Figure 2.** An overview of the system architecture. A small core (in red) handles the most fundamental data, while most capabilities are provided as extensions through the extension mechanism.

3.2. Extension Mechanism
In addition to our goals of multimodality and real-time support, another important goal when designing the architecture of our system was extensibility. To facilitate this goal, we designed
the system with modular architecture based on the “Dependency Injection”[20] idea that allows dynamically loading extensions, and is denoted here as an “extension mechanism”.

Two important examples of such extensions are the rendering and windowing (OS GUI interaction) subsystems. Our system can use EtoForms https://github.com/picoe/Eto and OpenGL [21] to run on the Windows and Linux systems natively, or it can use the Unity Engine (https://unity.com/) to run on platforms supported by Unity and enable additional capabilities such as virtual reality (VR). More examples of extensions will be discussed in Section 4.2.

3.3. The Representation of a Presentation
Abandoning the idea of a presentation being a linear sequence of discrete fixed-sized 2D slides requires a new concept of a ‘presentation’ and abstract structures that represent it. One of the decisions we made to overcome these issues is to turn presentations into continuous 3D scenes, improve the perceived flow of the presentation by smoothing transitions, and give a notion of context. Since the presentations are continuous 3D scenes, all the (visible) content is inevitably represented as 3D objects. All 3D objects are visualized as polylines and polygons as the least common denominator representation, including text and images.

To structure the presentation, we use the concepts of “presentation views” and “hierarchical story graph” as defined in [22]. We define them again here for completeness.

**Definition** A **Presentation View** (or **View** for short) is an abstraction of a “slide” for smooth non-linear multi-modal 3D presentations. It contains all the content that is presented together at a certain time point of a presentation, such as text, images, 3D models, etc. A view also has a local 3D camera, relative to which the content is placed. A view can also contain other views as children when it is a part of a hierarchical story-graph (see definitions below). The spatial positions of views, however, are not fixed, by default.

**Definition** A **Story-Graph** is a directed graph where each of its **nodes** is a view. Arrows of a story graph are directed and represent previous → next relationships, i.e., that one view is expected to be immediately followed by another during the normal flow of the presentation. In contemporary presentation tools, these relationships are, for the vast majority, linear.

Instead of a simple story-graph, where all views are equal, it is often convenient to organize views into a hierarchical structure. For example: a “presentation as a whole” then a “topic” then a “subtopic” then a leaf “view”.

**Definition** A **Hierarchical Story-Graph** is a story graph where views also form a hierarchy.

**Definition** A **Story Layout** is a 3D scene (automatically) generated from a story graph, which defines actual spatial positions of the views.

The content of each presentation view is itself a hierarchy of objects. For the sake of flexibility (and ultimately, extensibility) instead of introducing a fixed number of object types like ‘text’ or ‘image’, the objects themselves are just abstract containers of object components, which (actually) define object’s look and behavior. In a more formal way:

**Definition** An **Object Component** is a piece of data that defines looks, behavior, and/or other properties.

**Definition** An **Object** is a collection of one or more object components.

Having object components, thus, enables an additional type of extensions: extensions that introduce new components, and thus new objects that look or behave in a new way.

4. System Capabilities
In this section, we describe in details the capabilities of our system and how it facilitates the creation of non-linear multi-modal presentations by end users.
4.1. Non-linearity Extensions

One of the biggest obstacles when creating presentations that are continuous 3D scenes is the creation of said scenes, which includes placement of presentation views and content. We solve the problem by abstracting away the exact spatial positions of presentation views and only leaving their topological relationships (i.e., the story graph). The exact spatial positions and auxiliary visual elements are defined by a story layout which is automatically built as a 3D scene. A 3D story layout can be completely replaced if a new or a different layout is picked. Automatic view and camera placement is an important topic in itself. See, for example, [22], an approach we employ here.

We support two 3D story layouts (as extensions) out of the box: “Nested Spheres” and a ‘Building’. The “Nested Spheres” layout (Figure 3), as the name suggests, structures the presentation into spheres, where the spheres represent nodes of the (hierarchical) story graph. The spheres are placed inside the sphere that corresponds to their respective parent story graph node. Furthermore, the spheres are arranged in a way that (approximately) minimizes spatial distances and orientations between the neighboring views (i.e., the ones between which a direct transition exists within the graph).

The ‘Building’ layout (Figure 1 right (from the inside), Figure 4 (from above)) structures the presentation into a museum-like building, where second level nodes (i.e., direct children of the root) are represented as separate floors, and the leaf nodes are represented as rooms. Corridors are automatically generated that connect neighboring nodes, including the cases of forks.

Furthermore, we support a free exploration mode where the user is able to freely move around the scene. For even further immersive experience, the layouts support navigation and interaction in VR (see Section 4.3).

To allow the end user to control the sequence of the a presentation in this paradigm, we present a topological 2D story graph editor (Figure 5) which allows the user to perform basic operations on the presentation structure (a 2D graph). Those changes are immediately reflected in the main 3D presentation layout.

An extension that adds a 3D story layout provides a function that, given the 2D abstract story graph (like the one in Figure 5), constructs a 3D scene and places the views (that hold 3D objects and cameras) inside it. The main task of the construction of the 3D scene layout is the placement of the views in 3D space. Views are stored as objects in the core, without a global fixed position and orientation in 3D space. Each view has its content bound to it, so moving the view causes all its content (3D objects and cameras) to follow, thus avoiding the disruption of content visibility within a view. In addition to placing each view in the 3D space, the 3D scene can also have its own visual representation, path-finding mechanism for the direct
Figure 5. The 2D editor of the story graph. Changes are immediately reflected in the actual 3D scene layout (e.g., Figures 3, 4).

movement between views, and a collision detection and handling model for the free exploration mode. Corridors in the 3D ‘Building’ layout (Figure 1 right and Figure 4) are a good example of these last optional capabilities.

4.2. Multimodality Extensions
One of the main goals of our system is to support any modality in presentations. Since the content is (mostly) 3D and defined by object components, our system is capable of supporting a vast range of modalities through extensions. We now explain the different modalities we currently support:

- **Images**: Any presentation system must support the ‘usual’ presentation modalities, found in any other system. One such modality is images. We fully support images in our system by treating them as 2D textured rectangles in 3D space. Support for various image formats is added through extensions (an extension must be provided with a function to extract the pixel data from a given file). Support for the the default formats (PNG, JPEG, GIF, TIFF, and BMP) uses the standard libraries.

- **Text**: Another modality ubiquitous to presentation systems is text. Similar to images, we represent text with textured rectangles in 3D space. Since the text can change at runtime (in fact, we support WYSIWYG (What You See Is What You Get) rich text editing), the texture is regenerated on every text change. We use the OS capabilities (through the “System.Drawing” library) to render individual words as 2D images. Then, we position the images of all the words on the overall texture plane according to the text formatting and spacing rules. The extension mechanism also allows other types of data to be embedded into the text, as shown in the LaTeX example below.

- **LaTeX formulas**: Having a simple text editor is not always enough and often there is a need for formulas. We provide an efficient way of working with formulas, in which users edit the formulas in the LaTeX formula syntax [23] — a ubiquitous formula editing syntax among people, mostly in academia, who are the main users of formulas in presentations to begin with.

To support LaTeX, we use the WpfMath library (https://github.com/ForNeVeR/wpf-math) which allows the conversion of LaTeX formulas into images. Then again, the images are placed onto the overall text texture plane in the same way we place words. The user can embed a LaTeX formula into the text by clicking a context menu button or by pressing a keyboard shortcut and edit the LaTeX formula in a popup window (Figure 6). The user can then edit existing LaTeX formulas by double-clicking on the formulas. A similar text extension would have to provide functions to declare additional items for the text context menu, rendering the new data, and handling input (e.g., mouse double-clicks).

- **Video**: Another modality often supported by contemporary presentation systems is video. Herein, we support video playback using the FFmpeg library (https://ffmpeg.org/). The
extension provides an object component that works in a similar way to the image component (i.e., a textured rectangle). Here, however the texture is replaced every video frame and there are interaction handlers for playing/pausing/seeking/reverting the video playback. Furthermore, with FFmpeg’s support for live stream playback, the same component can be used for playback of live video feeds over the web, which is a separate modality in itself.

- **3D Geometric Models**: Since all the content is part of a 3D scene, we natively support 3D models, with ability to add support for loading various formats using extensions. Consider adding an ability to load OBJ geometric files [24], with the assumption that one has an external library to read OBJ files. The extensions mechanism requires, in this case, an implementation of a function that reads OBJ geometry and returns the loaded geometric model in a polygonal form that the core supports. Aside from OBJ, we also support the DICOM (see next item) and ITD (http://www.cs.technion.ac.il/~irit/) file formats.

- **Medical Geometric Data**: In addition to the usual 3D models, we also support the loading of DICOM [25] files. DICOM files are mostly used in the medical field to store 2D and 3D scan data in the form of voxel grid [26]. Apart from extracting the voxel data from the files, the extension uses the marching cubes algorithm [27] to convert the 3D data into polygons (Figure 7).

- **Animation and Simulation**: The 3D content must not necessarily be static. The content can change its visual properties, animate, or react to user actions. Being a real-time 3D environment at its core, our system can run simulations (e.g., physics) right inside the presentation (Figure 8), or present real-time data from the web. This modality is added by an extension that adds an object component that defines the needed behavior and handles interaction.

In addition, it is also possible to link different modalities together to achieve various inter-modality interactions. For example, a part of a 3D model can be highlighted when the mouse cursor is over the part of text describing this model part or vice versa.

### 4.3. Other Extensions

Having an ability to work with presentations capabilities that exceed the limitations of contemporary presentation systems opens a plethora of new possibilities, and at the same time produces new demands.

One of the main obstacles of transitioning to a new presentation system is having many existing presentations prepared in other presentation systems. To alleviate this difficulty, we provide an easy import method for legacy PowerPoint data. The user is able to load a .PPTX file into our system, which automatically converts text (preserving most of the formatting) and images into our system, allowing full further editing of the content. Import methods can be added as extension, and the PPTX one is implemented using the “Microsoft.Office.Interop” API [28].

With real-time 3D graphics and interactive capabilities, it is possible to go outside the bounds of contemporary ‘presentations’ and use our system for more general (albeit related) interactive experiences. One such possibility would be ‘tours’ in virtual reality (VR), either free or guided. For such a use case, we offer VR support for our system, fully functioning, for example, in a ‘Building’ story layout, allowing the user to explore the ‘presentation’ on their own in a much more immersive way, complete with a tutorial, to guide through the VR controls (Figure 9).

The presentation system described in this work has been implemented as a real-time application with an extension system, as described in Section 3.1. The two supported backends are OpenGL with native windowing system (currently runs on Windows, portable to Linux and OS X (Mac)) and Unity (with VR support).
Figure 6. Editing a LaTeX formula inside a text box using a Formula Editor popup window.

Figure 7. Import of a voxel-based CT scan into our system.

Figure 8. Fluid Simulation (in blue) running, in real time, right inside the presentation.

Figure 9. A tutorial ‘presentation’ teaching the user how to interact with 3D objects in VR.

The system has already been used to successfully make presentations on several unrelated topics. Further, to test its benefits, an experienced high school lecturer used the system to perform a lesson on solid geometry for high school students living in remote areas, with a highly positive reception.

5. Conclusions and Future Work
We have portrayed a presentation systems that aims at overcoming numerous limitations of contemporary presentation software. A user-friendly tool for creating presentations with multi-modal content, non-linear structure, and continuous flow has been achieved by utilizing modern real-time graphics technologies. We believe this will enable even novice users to present information in a much more efficient manner.

Of course, considering the scope of the project and its ambition, there is an enormous room for improvement. While establishing a framework, many technical features are missing that the end users may be accustomed to, including but not limited to multi-selection, certain features of text formatting, spell checking, etc., as well as IME integration into the WYSIWYG text editing, importing objects other than text and images from PPTX files, etc.

There are always quantitative improvements that can me made. Considering the difficulties of novice users in synthesizing 3D data, more 3D story layouts can be added like 3D terrains, space, etc. Another obvious quantitative improvement is adding support for more 3D object...
formats, and additional modalities such as 3D molecules.

We are yet to explore the ways for utilizing multiple displays during presentation. The additional screen real estate can be used for preserving context (like keeping the big picture while discussing details, keeping previous pieces of information while showing next ones, etc.), or for comparing two or more objects side-by-side, etc., and we are eager to explore these possibilities.

As discussed in Section 2.1, basing our system upon real-time 3D graphics technologies opens access to much more advanced animation capabilities that is usually possible in contemporary presentation systems. While we have successfully experimented with various animations in our prototypes, we are still yet to design an animation authoring interface that would be suitable for novice users, nor offered abilities to import animation data from animation authoring tools.

The same can also be said about interactivity, which, however, opens even more possibilities. Aside from the obvious use case of a presenter manipulating an object while explaining it, interactivity also opens a possibility of turning the presentation into self-study materials (“interactive textbooks”) that the (no longer) viewers can explore on their own. Our free exploration mode already can provide such an experience, especially when combined with VR. However, this only scratches the surface of the interaction possibilities that can be used for such self-study use cases.

References

[1] Tufte E R 2003 The cognitive style of PowerPoint vol 2006 (Graphics Press Cheshire, CT)
[2] Parker I 2001 The New Yorker 28 76–87
[3] Taylor D 2007 Developmental Medicine & Child Neurology 49 395–395
[4] Kjeldsen J 2006 Seminar.net vol 2
[5] Alley M and Robertshaw H 2004 ASME 2004 International Mechanical Engineering Congress and Exposition (American Society of Mechanical Engineers) pp 445–450
[6] Good L and Bederson B B 2002 Information Visualization 1 35–49
[7] Bederson B B 2011 Behaviour & Information Technology 30 853–866
[8] Chou P N, Chang C C and Lu P F 2015 Computers & Education 91 73–82
[9] Richards D and Taylor M 2015 Computers & Education 86 157–171
[10] Chittaro L and Ranzan R 2007 Computers & Education 49 3–18
[11] Kamsin A and Abdullah M H 2006 International Conference on Computer Graphics, Imaging and Visualisation (CGIV’06) (IEEE) pp 434–439
[12] Mayer R E 1997 Educational psychologist 32 1–19
[13] Mayer R E and Moreno R 2002 Educational psychology review 14 87–99
[14] Mayer R E and Moreno R 2002 Learning and instruction 12 107–119
[15] Wouters P, Tabbers H K and Paas F 2007 Educational Psychology Review 19 327–342
[16] Betrancourt M 2005 The Cambridge handbook of multimedia learning 287–296
[17] Avasar B, Aliabadi D E, Aliabadi E E and Yousefnejhad R 2016 arXiv preprint arXiv:1607.06979
[18] Lankow J, Ritchie J and Crooks R 2012 Infographics: The power of visual storytelling (John Wiley & Sons)
[19] Anderson R, Anderson R, Davis P, Linnell N, Prince C, Razmov V and Vidoen F 2007 Computer 40 56–61
[20] Prasanna D R 2009 Dependency injection (Manning)
[21] Shreiner D, Sellers G, Kessenich J and Licea-Kane B 2013 OpenGL programming guide: The Official guide to learning OpenGL version 4.3 (Addison-Wesley)
[22] Rodin D and Elber G 2018 Proceedings of the 11th International Symposium on Visual Information Communication and Interaction (ACM) pp 56–63
[23] Lamport L 1994 LATEX: a document preparation system: user’s guide and reference manual (Addison-Wesley)
[24] Wikipedia contributors 2019 Wavefront .obj file — Wikipedia, the free encyclopedia https://en.wikipedia.org/wiki/index.php?title=Wavefront._obj_file&oldid=893827771 [Online; accessed 20-May-2019]
[25] Mildenberger P, Eichelberg M and Martin E 2002 European radiology 12 920–927
[26] Wikipedia contributors 2019 Volume rendering — Wikipedia, the free encyclopedia https://en.wikipedia.org/wiki/index.php?title=Volume_rendering&oldid=887719874 [Online; accessed 19-May-2019]
[27] Lorensen W E and Cline H E 1987 ACM siggraph computer graphics vol 21 (ACM) pp 163–169
[28] Carter E and Lippert E 2006 Visual Studio Tools for Office (Addison-Wesley, Boston, MA)