

**Honours Project - MHW225671**

**INTERIM REPORT**

**2023-2024**

**Department of Computing**

**Submitted for the Degree of: Computer Games (Software Development)**

**BSc Computing**

**BSc ITMB**

**BSc SDfB**

**Project Title: Voxels and Polygons, competing for the standard geometric primitive in 3D Visualization.**

**Name: Lewis McIntyre**

**Programme: BSc (Hons) Computer Games (Software Development)**

**Matriculation Number: S2028246**

**Project Supervisor: Bryan Young**

**Second Marker:**

**Word Count: Approx. 4000 Words**(***excluding contents pages, figures, tables, references and Appendices***)

**“Except where explicitly stated, all work in this report, is my own original work and has not been submitted elsewhere in fulfilment of the requirement of this or any other award”.**

**Signed by Student: Date: 15/12/2023**

Contents

[1. Introduction (1067 words) 3](#_Toc153262760)

[1.1 The Standard Geometric Primitive – Polygons 3](#_Toc153262761)

[1.2 An Alternative – Voxels 3](#_Toc153262762)

[1.3 Voxel Development 4](#_Toc153262763)

[1.4 Research Question 5](#_Toc153262764)

[2. Contextual Review (1013 words) 5](#_Toc153262765)

[2.1 Performance Enhancing Techniques 5](#_Toc153262766)

[2.2 Current Tools 7](#_Toc153262767)

[2.3 Voxelization 7](#_Toc153262768)

[2.4 Voxels and Polygons (about 300 words) 7](#_Toc153262769)

[3. Methods & Design (1000 words) 7](#_Toc153262770)

[3.1 Method Plan 7](#_Toc153262771)

[3.2 Hypothesis 8](#_Toc153262772)

[3.3 Development Environment 8](#_Toc153262773)

[3.4 Data Gathering 8](#_Toc153262774)

[3.5 Data Analysis 8](#_Toc153262775)

[3.6 Relevance/Usefulness of analysis towards question 8](#_Toc153262776)

[3.7 Ethical Considerations 8](#_Toc153262777)

[4. Project Timeline 8](#_Toc153262778)

[5. Project Plan 8](#_Toc153262779)

[6. Appendices 8](#_Toc153262780)

[*7.* References 8](#_Toc153262781)

[1. Introduction Notes (part of 60%) 11](#_Toc153262782)

[2. Contextual Review Notes (part of 60%) 12](#_Toc153262783)

[3. Methods & Design Notes (20%) 12](#_Toc153262784)

[4. Project Plan (20%) 12](#_Toc153262785)

[Notes 12](#_Toc153262786)

[Topic 12](#_Toc153262787)

[Unused References 12](#_Toc153262788)

# Introduction (1067 words)

## The Standard Geometric Primitive – Polygons

Polygons are accepted as the current standard geometric primitive for both the 3D Modelling and Video Games industries; they can be defined as any closed 2D shape made of entirely straight lines. Current GPU manufacturers and tech giants have heavily invested into polygons, originating back to the late 1990s when the term was popularized by the GeForce 256 GPU. (TheBat!, 2023) This can be ascribed to their simplicity and efficiency with the many rasterization techniques at that time. However, their most prominent drawback is within its ‘*imitation*’ of real-world objects, as many techniques are only aesthetically, and not physically simulating the mesh. This is more specifically seen in games within the 3D dissection of polygonal models at runtime, where the model holds no data for the new face required to be created.

## An Alternative – Voxels

A suggested alternative, Voxels; also known as volumetric pixels; are a geometric primitive holding data for the entire model, including what cannot be seen. This primitive is used to represent values in three-dimensional space on a grid. Voxels function similarly to physical particles, therefore creating a more sophisticated implementation used to imitate the real world. Often, Voxels are referred to as 3D pixels and have a wide use in Procedural Generation, Particle Simulation, and Destructible Physics. (Zadick, Kenwright, & Mitchell, 2016) (Sekanina, 2023) Voxels are uniquely stored within a grid, allowing efficient usage of Object-Oriented Programming (OOP) and the Entity Component System (ECS), allowing each voxel to hold any number of unique properties. This is shown within the 3D voxel game ‘*MakeFarm*’ by David Szymon Grobert, as each block holds whether it is breakable, and if so, what item should be dropped. (GROBERT, 2023)

Polygons have held their position as the standard geometric primitive in many of the 3D visualization industries for over 20 years. This has resulted in many of the competing primitives to fall behind, as new technologies advance polygons forward. Many rendering practices for these primitives require the conversion of their structures into polygons, a major of which being the Marching Cubes algorithm. (Lorenson & Cline, 1987) Most previous rendering techniques for mainstream geometric primitives are insufficient for modern day usage, as researchers are now aiming to create greater realism. Due to the observation that is Moore’s law, which dictates that the number of transistors within computer chips double every two years, graphics are no longer restricted by our hardware, but by our software approach. A common implementation includes Ray Tracing, a method of emulating the light reflections and refractions of the real world. Ray Tracing is entirely possible within Voxels, although a similar process named Ray Marching is generally used, however, both of which do not require the need to convert to a polygonal mesh. These techniques are potentially creating a new use case for Voxels, with the possibility of gaining a wider acceptance as an alternative to polygons. The improved graphics that come with Ray techniques leads way into the necessity for Particle Simulation and Destructible Objects, creating a greater sense of realism, and both of which voxels are suited towards.

## Voxel Development

Atomontage, a leader in voxel development with a 13-year running micro-voxel engine, are consistently pushing the boundaries of voxel development forward. Atomontage are known for their usage of projection based voxelization, soft-body dynamics and their voxel editor’s recent open beta launch in 2021. (Atomontage, 2023) Daniel Tabar, and Branislav Siles, the founders of Atomontage, shared that the future of voxels lies within cloud services, as with their current implementation, the voxels are relatively inexpensive on both the view, and the server. The future of Atomontage aims to revolutionize both the cloud gaming industry, and the interactive sandbox games genre, and Daniel Tabar has stated that their engine could be compared to “*Roblox + Minecraft + a higher resolution*”. (JTVentures, 2023)

Atomontage is not only invested within the games industry, but also has ties to medicine from one of its most supportive angel investors, Tommy Palm (Siles, 2019). Voxel-based 3D visualization is an incredibly large sector within medicine, with competing software such as ‘*3D Slicer*’, ‘*Voxel-Man ENT*’ and ‘*SolidWorks 3D-Doctor*’. These provides a multitude of tools ranging from CT Lung Analysis to Virtual Training Simulators, all of which involve intensive usage of volumetric data. The usage of voxels within medicine comes from the necessity of creating not only a realistic model, but also for accurate mapping of different tissue types. (Slicer Community, 2023) (Voxel-Man, 2023) (Dassault Systemes, 2023) As voxel techniques are researched, these software could potentially receive updates allowing more realistic, and accurate models for its users. A study within the effect of Field-Of-View and Visual Realism on virtual training tasks found that visual realism may impact the virtual training performance negatively, but that it impacts the future real-life assessment positively. (Ragan, et al., 2015) This suggests that if voxels gain more realistic rendering techniques, that it could positively impact training procedures for users of Medical Simulations.

The games industry is at the forefront of realistic voxel development, a recent voxel success story includes Tuxedo Labs ‘*Teardown*’. Featuring a fully destructible environment, particle simulations of fire and smoke, and an incredible in-house lighting system, it has garnered almost 76,000 positive reviews on steam, gaining the award for ‘*Excellence in Design*’ at the Independent Games Festival in 2021. (Tuxedo Labs, 2023) (Informa Tech, 2023) It is a game that has shown the practicality of voxels within indie game development, and the massive support such a game can receive. According to the Tuxedo Labs modding wiki, the artists used ‘*MagicaVoxel*’, a lightweight voxel art editor, to create all assets for the game. (Tuxedo Labs, 2023) This art editor is still supported, is fully open source and has hundreds of modelling tutorials. It is an incredibly powerful tool within the modelling of 3D voxel environments; however, they require hundreds of hours in development to take form. A more efficient method, used for both rapid prototyping and final model products is LiDAR (Light Detection and Ranging). LiDAR uses pulses of light to rapidly generate point clouds, which are simply a collection of points in three-dimensional space. As suggested by Xu, Tong & Stilla, point clouds can be efficiently converted into a voxel representation, and then can be effectively used within structural analysis, 3D simulation of fluids, volume estimation and pathfinding. Voxels are mentioned to be one of the best primitives used to transfer point cloud data into a higher data format, due to their speed and efficiency, although suggests they may lack accuracy in conversion as compared to some others. (Tong, Stilla, & Xu, 2021) If too much accuracy is lost, this technique may still be useful within rapid prototyping, or give artists a reference prop to work from. (Yuan, Peng, & Zhang, 2018)

## Research Question

The aim of this paper is the investigation of the possible voxel future, as a replacement for polygons as the standard geometric primitive for 3D visualization. The strengths and weaknesses of voxels should be highlighted, whilst further research avenues should be opened and suggested. The research question is:

*“Could voxels replace polygons, as the standard geometric primitive for 3D visualization?”*

# Contextual Review (1013 words)

## Performance Enhancing Techniques

Research within Design & Technology (Telea & Jalba, 2011), Engineering (McNeely, Puterbaugh, & Troy, 2005) and Biomedical Studies (Mason, et al., 2000) suggests a common limitation of large memory consumption whilst using the standard implementation of regular voxel grids. The regular grid structure creates a memory footprint of O(n^3) where n represents the width, length and height. Recent studies have explored the manipulation of a regular grid using warp fields to create dynamic irregular grids, which significantly reduced memory cost within their use case. (Lombardi, et al., 2019) There are many other solutions to this issue, a notable case involving the major Australian software company ‘*Euclideon*’. A video posted in 2010 by the CEO of Euclideon, Bruce Dell, showcases an ‘*Unlimited Detail Technology*’, which garnered mass attention towards an engine which could dynamically render any number of objects in any scene no matter how large. (Quipster99, 2010) It appears to do so by flexibly changing the Level of Detail (LOD) of a voxel grid at runtime. A similar technology was used in Funk & Borner’s research of efficient LOD visualization tools and solved this issue through the usage of Sparse Voxel Octrees (SVOs). (Funk & Borner, 2016) SVOs are a technique used within rendering, created through the combination of Sparse Voxels, and Octree Voxels. These are defined as the following:

The rendering technique Sparse Voxel removes the data of all unoccupied values. (Figure 1)

The rendering technique Octree Voxel divides the 3D space into a hierarchy of octants, that are recursively divided until a goal is achieve, e.g. the final level of detail is reached. (Figure 2)

A diagram of a circle and a cross

Description automatically generated

Sparse Voxel (Figure 1)

A diagram of a circle and a cross

Description automatically generated

Octree Voxel (Figure 2)

Studies have found that both the rendering methods Sparse Voxel, and Octree Voxel, can be incredibly efficient in the optimisation of non-volatile memory, a mentionable study by Gebhardt, Scott et al. finding both file sizes to reach as low as 0.0008x that of their traditional regular grid counterparts. (Gebhardt, et al., 2009) It can be implied that this technique could be further improved by the combination of each technique, to create the Sparse Voxel Octree structure. However, these techniques are not perfect, due to the nature of their compression, models that are more densely populated, with zero empty space or duplication would not receive the same benefits as those without. Therefore, a greater level of compression which can be processed with a more efficient algorithm could help support voxels and increase the feasibility of voxels within more 3D visualization software. (Laine & Karras, 2010)

The costs of both compression and decompression techniques used within voxels can have a heavy cost within the CPU and the GPU, specifically when implementing real-time graphics and dynamic construction and deconstruction of voxels. There are multiple grid traversal techniques used to mitigate this cost, a notable recent usage includes Morton Order, also known as the Z-order curve. (Baert, Lagae, & Dutre, 2013) It is used to convert any set of multidimensional data into a linear sequence for fast traversal, in our case three-dimensional voxel space. Pan, Yucong found when investigating the traversal of Sparse Voxel Octrees using Morton Order, that without the usage of this technique, or one of similar standard, real-time global illumination would not be feasible, especially when working with a voxel grid that is modifiable at runtime. (Pan, 2021)

The goal of Global Illumination is to create a high level of visual realism, which is a crucial component to many of the 3D visualization industries, with notable industries such as product design and advertising needing extra realism. (Dutre, Bekaert, & Bala, 2018) (Kim, Choi, & Wakslak, 2019) This goal can be found using a variety of techniques, although at the highest level of realism it is generally by a form of light-simulating rays. Notable practices include Ray Casting, Ray Tracing and Ray Marching. These techniques are incredibly similar; however, the key technique Ray Marching stands out within voxel development, which traverses the voxel grid using distance estimations. Dreams, a voxel-based game creation system with in-built 3D modelling software uses a form of Ray Marching known as Sphere Tracing, which has the added benefit of smoothing all surfaces hit. (Alex Evans, 2015) Dreams runs entirely on the PlayStation 4 and shows the incredible possibilities of voxels as a base for model storage.

## Current Tools

Technology within games constantly expands the possibilities of 3D visualization, the tool Unreal Engine by Epic Games standing out as a main contributor. Although Unreal Engine was initially created as an in-house engine for the First Person Shooter ‘*Unreal*’ (1998), it quickly gained notoriety and opened for developer use, and then public usage in late 2009. (Kao & Wang, 2023) Unreal Engine is a 3D visualization tool which is not only used within the Games Industry, but has seen usage within media, with over six hundred usages in film and TV shows, and has in-built templates supporting Architecture, Automotive Design, Simulation and Animations. (Epic Games, Inc., 2023) Voxels are supported within Unreal Engine through the Voxel Plugin which has both a free legacy version, and an updated paid version which includes a perpetual license for any future version of Voxel Plugin. (Voxel Plugin, 2023) (Unreal Engine, 2023) This plugin supports many of the desired features for voxels, including real-time destruction and creation, procedural generation and physics simulations of terrains. Voxel Plugin functions well with the newest release of Unreal Engine 5, allowing the benefits of both ‘*Nanite*’, which dynamically edits the level of detail in the scene in real-time, and ‘*Lumen*’, which allows the real-time edits of all lighting features. Unreal Engine also allows the usage of a node-based scripting system named ‘*Blueprints*’ which is user friendly and provides designers the ability to create systems without the need for a programming background. (El-Wajeh, Hatton, & Lee, 2022) These suggested tools make Unreal Engine a major competitor as one of the tools used within 3D visualization.

Unity is a large competitor to Unreal Engine, holding a 48% global market share of game developers as compared to Unreal Engines 13% in 2020. (Kao & Wang, 2023) Unity has many solutions for 3D visualization, some of which include Architecture, Engineering and Product Design. (Unity Technologies, 2023) A paper on voxel options with Unity involved the usage of mainly native techniques, and ongoing open-source projects supported by the community. (Aleksandrov, Zlatanova, & Heslop, 2022) It was found that voxels are supported within Unity, but many of the tools were left in the hands of the programmer, and within these tools were many limitations regarding large physics simulations. The open-source projects on the other hand generally use some form of octree compression, however all were rendered using some form of conversion to polygons to then be rasterized.

More specific tools within Unity (200 words)

## Voxelization (300 words)

Explain voxelization and give examples of a few methods.

## Voxels and Polygons (about 300 words)

Comparison and conclusion to contextual review

Previous research on the comparison between voxels and polygons.

# Methods & Design (1000 words)

## Method Plan

* What are you going to do?
* How are you going to do it?
* How does it help *solve* the research question?
* Describe the methods including its design features.
* Can include references, sources and constraints.

## Hypothesis

## Development Environment

## Data Gathering

## Data Analysis

## Relevance/Usefulness of analysis towards question

## Ethical Considerations

What ethical issues should be considered

Evidence of approved application

# Project Timeline

Gantt Chart

# Project Plan

* What are the main tasks involved within this project?
* What are the intended deliverables at the end of this project?
* Create a work plan, e.g. Using a Gantt Chart

# Appendices

# **References**

Aleksandrov, M., Zlatanova, S., & Heslop, D. J. (2022). Voxelisation and Voxel Management Options in Unity3D. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 13-20.

Alex Evans. (2015, August 12). Learning from failure. *Advances in Real-Time Rendering in Games. MediaMolecule, SIGGRAPH*, Volume 2.

Atomontage. (2023, May 1). *Home: Atomontage*. Retrieved from Atomontage Web Site: https://www.atomontage.com

Baert, J., Lagae, A., & Dutre, P. (2013). Out-of-Core Construction of Sparse Voxel Octrees. *High-Performance Graphics Conference* (pp. 27-32). Anaheim, California: ACM.

Dassault Systemes. (2023, December 3). *3D-DOCTOR: Able Software Corporate*. Retrieved from SolidWorks: https://www.solidworks.com/partner-product/3d-doctor

Dutre, P., Bekaert, P., & Bala, K. (2018). *Advanced Global Illumination.* Boca Raton, Florida: CRC Press.

El-Wajeh, Y. A., Hatton, P. V., & Lee, N. J. (2022). Unreal Engine 5 and Immersive Surgical Training: Translating Advances in Gaming Technology into Extended-Reality Surgical Training Programmers. *British Journal of Surgery*, 470-471.

Epic Games, Inc. (2023, December 15). *Unreal Engine 5.3 brings new potential for media and entertainment projects*. Retrieved from Unreal Engine: https://www.unrealengine.com/

Fedorov, A., Beichel, R., Kalpathy-Cramer, J., Filnet, J., Fillion-Robin, J.-C., Pujol, S., . . . Kikinis, R. (2012). 3D Slicer as an Image Computing Platform for the Quantitative. *Magnetic Resonance Imaging*, 1323-1341.

Funk, E., & Borner, A. (2016). Infinite, Sparse 3D Modelling Volumes. *VISIGRAPP: Computer Vision, Imaging and Computer Graphics Theory and Applications* (pp. 593-605). Rome, Italy: Springer International Publishing.

Gebhardt, S., Payzer, E., Salemann, L., Fettinger, A., Rotenburg, E., & Seher, C. (2009). *Polygons, Point-Clouds, and Voxels, a Comparison of High-Fidelity Terrain Representations.* Environmental Science.

GROBERT, D. S. (2023). *Implementation of 3D game MakeFarm with infinite terrain model and basic physics in OpenGL API.* Gliwice: Silesian University of Technology.

Informa Tech. (2023, December 3). *Independent Games Festival Finalists & Winners 2021*. Retrieved from Independent Games Festival: https://igf.com

JTVentures. (2023, December 3). *Atomontage: We've solved the problems with 3D that even Google and Meta are struggling with*. Retrieved from JTVentures: https://www.jtventures.cz/

Kao, M., & Wang, P. (2023). *Epic Games Thesis.* Contrary Research.

Kim, K. B., Choi, J., & Wakslak, C. J. (2019). The Image Realism Effect: The Effect of Unrealistic Product Images in Advertising. *Joural of Advertising*, 251-270.

Laine, S., & Karras, T. (2010). *Efficient Sparse Voxel Octrees - Analysis, Extensions and Implementation.* Santa Clara, California: NVIDIA Research.

Lombardi, S., Simon, T., Saragih, J., Schwartz, G., Lehrmann, A., & Sheikh, Y. (2019). Neural Volumes: Learning Dynamic Renderable Volumes from Images. *ACM SIGGRAPH* (pp. 1-14). New York, NY, USA: Association for Computing Machinery.

Lorenson, W. E., & Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics* (pp. 163-169). New York, N.Y.: ACM.

Mason, P. A., Hurt, W. D., Walters, T. J., D'Andrea, J. A., Gajsek, P., Ryan, K. L., . . . Ziriax, J. M. (2000). Effects of frequency, permittivity, and voxel size on predicted specific absorption rate values in biological tissue during electromagnetic-field exposure. *IEEE Transactions on Microwave Theory and Techniques*, 2050-2058.

McNeely, W. A., Puterbaugh, K. D., & Troy, J. J. (2005). Six degree-of-freedom haptic rendering using voxel sampling. *ACM SIGGRAPH* (pp. 42-49). Los Angelos, California: Association for Computing Machinery.

Pan, Y. (2021). *Dynamic Update of Sparse Voxel Octree Based on Morton Code.* West Lafayette, Indiana: ProQuest Dissertation Publishing.

Quipster99. (2010, February 26). *Unlimited Detail Technology.* Retrieved from Youtube: https://www.youtube.com/

Ragan, E. D., Bowman, D. A., Kopper, R., Stinson, C., Scerbo, S., & McMahan, R. P. (2015). Effects of Field of View and Visual Complexity on Virtual Reality Training Effectiveness for a Visual Scanning Task. *Visualization and Computer Graphics*, 794-807.

Randles , B., Welcher, J., Szabo, T., Jones, B., Elliot, D., & MacAdams, C. (2010). *The Accuracy of Photogrammetry vs. Hands-on Measurement Techniques used in Accident Reconstruction.* Detroit, Michigan, United States: SAE International.

Sekanina, J. (2023). *An Exploration of Algorithms for Real-Time Terrain Destruction.* Brno, Czechia: Masaryk University, Faculty of Informatics.

Siles, B. (2019, June 24). Atomontage Inc.’s Branislav Siles on the Limits of Polygons, the Voxel Future, Streaming, AI and more. (D. Aubrey, Interviewer)

Slicer Community. (2023, December 3). *Slicer: Home Page*. Retrieved from Slicer: https://www.slicer.org/

Telea, A., & Jalba, A. (2011). Voxel-Based Assessment of Printability of 3D Shapes. *Mathematical Morphology and its Applications to Image and Signal Processing* (pp. 393-404). Verbania-Intra, Italy: Springer-Verlag.

TheBat! (2023, May 1). *GeForce 256, GPU Database: Tech Power Up*. Retrieved from Tech Power Up Web site: https://www.techpowerup.com

Tong, X., Stilla, U., & Xu, Y. (2021). Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry. *Automtion In Construction*.

Tuxedo Labs. (2023, May 1). *Teardown Modding. Tuxedo Labs*. Retrieved from Teardown game website: https://www.teardowngame.com

Tuxedo Labs. (2023, May 1). *Teardown, Steam*. Retrieved from Steam/Valve Corporation. Websitie: https://store.steampowered.com

Unity Technologies. (2023, December 15). *Unity Design Visualization*. Retrieved from Unity: https://unity.com

Unreal Engine. (2023, December 15). *Digging Deep: Voxel Plugin 2.0's Next-Gen World Creation Workflows | Inside Unreal*. Retrieved from Youtube: https://www.youtube.com/

Voxel Plugin. (2023, December 15). *Voxel Plugin Docs*. Retrieved from Voxel Plugin: https://docs.voxelplugin.com/

Voxel-Man. (2023, December 3). *Voxel-Man: Home Page*. Retrieved from Voxel-Man: https://www.voxel-man.com/

Wilder, M. W. (2015). *An Investigation in Implementing a C++ Voxel Game Engine with Destructible Terrain.* Akron, Ohio: University of Akron.

Yuan, T., Peng, X., & Zhang, D. (2018). Direct Rapid Prototyping from Point Cloud Data without Surface Reconstruction. *Computer-Aided Design & Applications* , 390-398.

Zadick, J., Kenwright, B., & Mitchell, K. (2016). Integrating Real-Time Fluid Simulation with a Voxel Engine. *The Computer Games Journal*, 55-64.

## 1. Introduction Notes (part of 60%)

* Opening statement should be clear, the issue I am looking to solve should be well argued and should naturally lead into the research question.

1. Introduce current ‘standard’ geometric primitive.
2. Give insight as to why polygons are the standard geometric primitive.
3. Introduce alterative of voxels and give definitions.
4. Give insight as to why voxels are not the current standard geometric primitive.
5. Suggest that voxels have been overlooked when looking at its specific use case/the future of voxels could have more voxels use cases.

## 2. Contextual Review Notes (part of 60%)

* Extensive use of literature from scholarly articles. Follows smoothly from statement to statement and shows critical engagement with literature.

## 3. Methods & Design Notes (20%)

* Clear description of design methods that are to be employed. Methods are well justified over any alternative methods, which is backed up by supporting materials. Description of procedures and tools is clear.

## 4. Project Plan (20%)

* High level of understanding of the tasks involved within the project objectives. Document is fully chaptered with sufficient detail. Work plan describing all considerations. Ethics is fully presented, approved, or not required based on evidence.

## Notes

### Topic

Reaffirm already existing research that Morton’s code/Z-Order curve can help Sparse Voxel Octree tree traversal.

Show the point of contention within voxels and showcase best use cases for each. Main point – Mention although Polygons may be better than voxels in terms of many rasterization techniques, however looking into the future of computer graphics, ray marching can hold its own against ray tracing.

## Unused References

Project uses a voxel-based approach to haptic feedback. Has limitations within its voxel size due to memory consumption and states it accepts a common engineer rule of 0.5-inch clearance wherever possible.

(McNeely, Puterbaugh, & Troy, 2005)

Project uses a neural voxel volume to dynamically render 3D objects from 2D images. Discusses limtiations of voxels cubic relationship of resolution and memory and solves this limitation using warp fields to create dynamic irregular grids.

(Lombardi, et al., 2019)

Project discusses printability of models sent to a 3D printer using a voxel-based representation. Defines the projects main limitation as the voxelization resolution capable by their GPUs, as certain characteristics have not been calculated within the model. Discusses the solution of this problem as more GPU memory.

(Telea & Jalba, 2011)

Project discussing various factors within biological tissue during electromagnetic-field exposure. Explains that generally in this field, they need incredibly small voxel sizes, of which when calculated came out to much more RAM usage than their computers could handle. This was one of the main limitations of this project.

(Mason, et al., 2000)

Investigating destructible terrain using voxels in C++. Currently limited by terrain size as entire terrain is rendered, suggests the usage of an Octree.

(Wilder, 2015)

Compare two things

* FPS
* Memory (Volatile, RAM, GPU)

Unreal Engine 5 (Voxel Plugin Paid)

Unity (Many open-source usage of SVOs)

Godot (Has voxel modules but not sure how well they function)

Other engines (Most other engines only render one primitive type (voxels or polygons), would be hard to compare)

Compare the RAM usage of a both a polygonal model and a voxel model in Unity using open-source voxel frameworks. Voxel model created using a voxelization algorithm from default polygonal model with internal voxel structure.