# Comparative analysis aimed at optimization and visualization of BIM for VR applications.

by

Diego Zambrano Guerrero

A thesis submitted to
the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Information Technology

in

Digital Media

School of Information Technology

Carleton University

Ottawa, Ontario

## Abstract

In this thesis, we explore the proper visualization of Autodesk Revit data inside virtual reality applications by presenting a new optimization framework leveraging a custom plugin / tool inside Autodesk 3DS Max. This tool leverages new Autodesk software integration APIs, in conjunction with Maxscript (native language of 3DS Max) to read and translate BIM data from Revit, while preserving its metadata, materials and textures. Multiple model optimization procedures are applied automatically with the most important being the implementation of a Level of Detail (LOD) system for every model in the scene.

To test our proposed framework, we conducted a comparative analysis of quantitative data gathered from multiple virtual reality applications deployed and running inside an Oculus Quest mobile VR device. Four different Revit case studies were used to develop these applications. They were chosen to exemplify the different levels of complexity that a BIM project can reach in real situations. The first set of applications were developed using our proposed framework, while the second set were developed using the Unreal Datasmith toolset by Epic Games.

Results show that the applications developed using our proposed framework performed better than the applications developed using the Unreal Datasmith toolset, in terms of higher average frames per second, higher visual fidelity and lower GPU usage. Additionally, results show that while the implementation of LOD systems requires extra memory, its benefits regarding performance are substantial.

## Acknowledgements

To God for always being by my side. To my parents for always listening, trusting and believing in me. Throughout my life they have proven to be the best role models I could have ever hoped for. To my brother for his support and understanding while we both live in a foreign country. To my best friend Jose Alejandro Enriquez and the friendly competition we have maintained over the years, pushing us to be the best version of ourselves.

To my supervisor Dr. Steve Fai, Lara Chow and everyone at the Carleton Immersive Media Studio for their support and comradery these past two years, they have challenged me professionally and personally to grow and develop myself beyond my cultural background.

To Dr. Mario Santana, Laurie Smith and everyone involved in the NSERC CREATE Heritage Engineering program, without your support and encouragement I would have not taken the challenge of moving back to Canada to pursue a graduate degree.

To my supervisor Dr. Ali Arya for always asking the right questions and pushing me to complete this degree in a timely manner. To Dr. Arya's IMG group for their constructive criticism and awesome lunches at the Mike's Place pub.

Finally, to the School of Information Technology at Carleton University, Dr. Chris Joslin, Dr. Audrey Girouard, and the graduate administrators Erenia Oliver and Ria Akaiwa. Thank you for your help, and for allowing me the privilege of working as a teaching assistant, helping young students in their quest to expand their knowledge.

# Table of Contents

## List of Abbreviations

***API,*** Application Programming Interface

***BIM,*** Building Information Modeling

***LOD,*** Level Of Detail

***VR,*** Virtual Reality

***GPU,*** Graphics Processing Unit

***CPU,*** Central Processing Unit

***CAD,*** Computer Assisted Drafting

***B-REP,*** Boundary Representation

***NURBS,*** Non-Uniform Rational B-Spline

***HVAC,*** Heating Ventilation and Air Conditioning

***AEC,*** Architecture, Engineering and Construction

***ACIS Solids,*** Alan, Charles, Ian and Spatial Solids.

***FPS,*** Frames Per Second

***CAVE,*** Cave Automatic Virtual Environment

***IFC,*** Industry Foundation Classes

***XML,*** Extensive Markup Language

***FBX,*** Filmbox

***UI,*** User Interface

***HMD,*** Head Mounted Display

***DK,*** Development Kit

***UE,*** Unreal Engine

## List of Tables

## List of Illustrations

# Chapter 1: Introduction

## 1.1 Background and Motivation.

### 1.1.1 Computer Assisted Drafting

Computer Assisted Drafting (CAD) emerged in the 1960s as a way to leverage computer technologies to aid and facilitate any design process, it expanded quickly as it was proven to increase the productivity of the designer while boosting the overall quality of the design (Herron, 2010). Computer Aided Design is closely related to the field of three-dimensional (3D) modeling in computer graphics and particularly uses two important technologies in that field: Facet Modeling and Boundary Representation.

Facet Modeling (Owen et al., 2002) represents surfaces as a mesh of many connected planar triangles (or polygons, in general), where the position, size and shape of these surfaces can be changed in order to represent different 3D objects in digital space.

It was favorited, particularly in computer games and animation, for its simplicity and relative ease of programming (Russo et al., 2006). But its uses in the engineering field were limited as high accuracy models require millions of triangles to be rendered.

Boundary Representation (B-REP), on the other hand, uses mathematically defined surfaces connected by topology to represent 3D objects, the most used mathematical method are the Non-uniform Rational B-Spline curves, usually referred to as NURBS curves (Schneider, 2014).

B-Rep methods can create concise and precise models that could easily represent complex shapes without the need to store large amount of polygonal information. Figure 1 illustrates the use of Facet Modeling and B-REP for 3D objects.

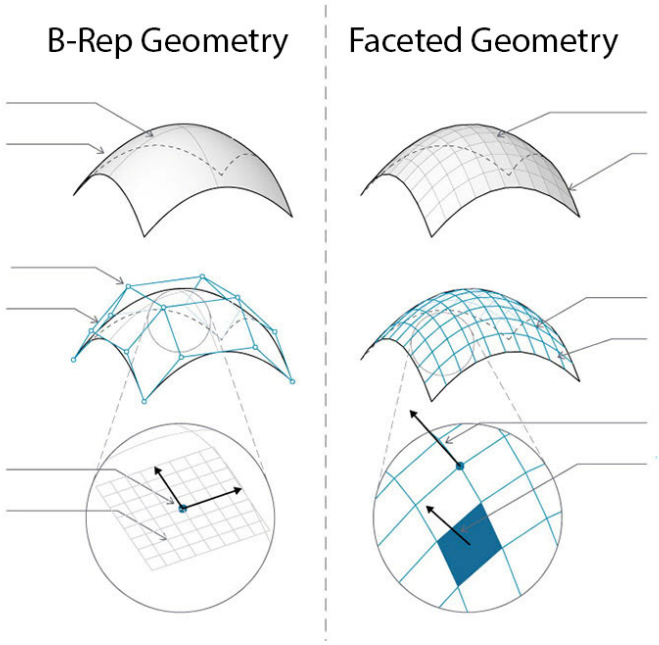


**Figure 1 Shows the differences between B-Rep and Faceted Geometry.**

B-REP proved the best solution for engineering and is used in most of the computer aided design software aimed at manufacturing of industrial and consumer products, design, and simulation. Facet Modeling was developed as the preferred solution for developing less precise, high speed shapes needed for gaming, animation, and digital mock-ups.

Recently, the use of facet modeling has become more common and more valuable in engineering workflows with the rise of new high-precision, low-cost technologies aimed at 3D scanning, medical scanning, 3D printing, and real-time virtual reality applications.

Multiple algorithms have been developed for translating boundary representation models such as NURBS into faceted data (polygons).

Nevertheless, these algorithms strive to preserve the precise curves of B-REP and use millions of polygons in the process. While all these polygons could still be painstakingly rendered into a flat 2D image, and multiple of these images can be used in video animations, the same models are too complex to be used inside real-time graphics applications.

Real-time and virtual reality (VR) applications, demand a new method of translating B-REP models into faceted models. The focus has moved away from the accurate and precise representation of a curve, into the ability to render the same curve as fast as 90 times every second, making mesh optimization one of the most pressing challenges in this process. This is where the development of Levels of Detail (LODs) comes in.

The concept of Levels of Detail, in computer graphics, refers to the act of automatically reducing or increasing the visual quality of a 3D object based on its position and actual size on the screen. These techniques should remain unnoticed to the user since models affected are either distant or moving fast (Biljecki, et al., 2016). Multiple LOD techniques exist and they usually target model complexity (number of triangles comprising a mesh); shader complexity (number of shader instructions); or texture complexity (texture compression, commonly referred to as mipmapping).

For the purposes of this research the concept of Level of Detail (LODs) was used only to refer to the mesh complexity of any given 3D model.

### 1.1.2 The Building Information Model (BIM)

The development of a Building Information Model (BIM) is a collaborative process that involves the generation and management of a digital repository, which includes representations of all the physical and functional characteristics of places (Eastman, et al., 2008). This repository dynamically connects 2D blueprints for a building with its 3D model and its parameters inside the context menus. When a new window is added, or an existing window is resized inside the 3D view, this change will be automatically reflected in the 2D blueprints and on the context menu parameters. This repository of all the building data is commonly referred to as the "Digital Twin" (Borrmann et al., 2018).

A BIM project can include multiple extra dimensions that go beyond the 3D model found in classic computer aided design development. 4D BIM adds the time dimension, making it possible to develop a construction schedule. 5D BIM adds the costs of materials, so that total construction costs can be updated automatically for new designs and concepts. 6D BIM bridges the construction and operation phases of the building, providing a system that will facilitate and streamline facility management. Finally, 7D BIM leverages all the previous data to calculate and estimate operation and life-cycle costs, all based only on the Digital Twin, while the project is still on its design phase (Aengenvoort et al., 2016).

BIM allows professionals from all the different disciplines needed in the project, structural, construction, electrical, HVAC, and others to work inside the same digital repository were their models will be constantly compared and overlaid to avoid any collision or error, preventing any unpleasant surprises that might come up on-site during the construction phase.

### 1.1.3 Gamification in the AEC field.

Gaming has always been associated with having fun, so integrating gaming concepts into other, less entertaining tasks has proven to be a successful way to learn a new concept or get an idea across.

In other words, gamification works (Wang et al., 2014). In the Architecture, Engineering and Construction (AEC) field, serious games are particularly useful to replicate dangerous environments or situations that could occur, these interactive simulations help train and educate users about potential treats as seen in various papers covered in my Related Works section. But, since the inception and expansion of BIM, the development of serious games has evolved into a tool that could fully replicate all the physical and functional characteristics of a building. This "Digital Twin" can help AEC professionals avoid any unforeseen problems, by consolidating all the project information in a single source that allows for real-time access to any changes and updates made.

## 1.2 Problem Statement

With BIM becoming the standard of the industry, software programs like Autodesk Revit (Autodesk 2019) have become a staple in architecture firms.

Revit modeling is based on B-REP technology, producing complex Computer Aided Design (CAD) inspired drawings that use mathematical curves, known as ACIS solids. One of the most important issues facing the expansion of BIM is providing a better level of understanding of the overall project and all its components. BIM development needs to step away from static models and spreadsheets and provide interactive real-time information that will:

- Better communicate the design Intent to project Stakeholders.
- Get stakeholders to buy into the visuals of the building.
- Provide content that can be used for marketing.
- Immerse the user inside the space and its components.

Video game and virtual reality technologies have seen a steady increase in various fields due to the immersion and sense of presence they provide. Game engines (special software tools for making videogames) are becoming increasingly popular as illustrated by the growth of the Unreal Engine 4 and Unity 3D programs, whose adoption continues to grow (Forrester Consulting, 2018).

Companies in the AEC field have had various degrees of success in implementing game engine and VR technologies, while encountering multiple limitations. One of the biggest limitations is model complexity. VR applications have set a standard of 90 frames per second (72 frames per second for the Oculus Quest mobile VR headset) so that the user feels comfortable while using the headset.

In order to achieve this desired frame rate, the meshes and textures used inside the applications need to be highly optimized. Common translation practices require intermediary software and conversion between multiple non-native formats. This usually introduces mesh errors that need to be cleaned up manually even before meshing. Finally, a VR ready model is obtained after multiple conversions and a long clean up procedure that requires the manual labor of trained individuals, raising production time and costs (Merschbrock, et al., 2014), (Vajak, et al., 2017), (Dinis, et al., 2017), (Yu-Cheng Lin, et al., 2018).

In the recent months, multiple software companies have identified this problem and are working towards the development of new software tools that can partially or completely automate this desired model translation. The developers of the Unreal Engine took this matter into their own hands and launched their own format aimed at translating Computer Assisted Drafting data, named Unreal Datasmith (Epic Games, 2019).

This new plugin gained popularity inside the AEC field due to it being released for free while still in beta form. It addresses model optimization using its own Level of Detail system. While it is a step in the right direction, my experiments show that this system has critical limitations when properly translating BIM data. This current approach at LOD creation is still based on a facet modeling workflow aimed at videogame content and introduces mesh errors when used for BIM data, while keeping a high triangle count.

Our primary research hypothesis is aimed at answering the following questions regarding the visualization of Autodesk Revit data inside virtual reality applications:

- Could the new Autodesk software integration APIs be utilized to develop a new framework and automated pipeline?
- What new tools are required?
- How would this new framework perform against the current industry standards?

## 1.3 Contribution

The primary contribution of the research reported here is a new BIM optimization framework that offers a more streamlined and optimized way of translating Autodesk Revit projects into virtual reality applications while maintaining parametric metadata, materials and textures for every model.

This pipeline leverages new Autodesk software integration features, in conjunction with Maxscript (native language of 3DS Max), to do as follows. Read and translate Revit data into 3DS Max. Get every model and create three extra versions of it, each with a different model complexity and lower triangle count. Check and fix possible mesh and lighting errors (by welding vertices, unifying normals and recreating smoothing groups).

For every model, create a new UVW map channel with a new set of UV coordinates to be used for light baking. Models with the same name are grouped and added to a unique LOD system that leverages their different complexities inside the game engine.

The second contribution is a comparative analysis that evaluates the performance of the new pipeline. Four different Revit data case studies were compared. They were specifically chosen to exemplify the different levels of complexity that a Revit project can reach in real situations. Every translated Revit project was implemented inside a game application developed using the Unreal Engine 4 and deployed to the Oculus Quest VR device for comparison and performance analysis.

## 1.4 Research Approach

This research serves two purposes. Its first purpose is to present a new framework for the automatic translation and optimization of BIM data from Autodesk Revit.

Its second purpose is to test the capabilities of the presented framework via a comparative analysis of quantitative data gathered from two sets of identical virtual reality game applications. The first set was created using the new proposed framework leveraging a new pipeline involving a new plugin / tool inside 3DS Max, while the second set was developed using the Unreal Datasmith toolset by Epic Games Enterprise. This toolset is considered the current standard of the industry, firstly because it is marketed and developed by a different branch of the same company that created the Unreal Engine. And secondly because a beta version of this toolset was made available free of charge through a subscription system (Kharvari et al., 2019).

Both sets of virtual reality game applications were deployed to run inside an Oculus Quest mobile VR device. The comparison metrics for this study were taken from the Oculus developer documentation (Oculus, 2019) and are as follows:

- Average frames per second (FPS).
- CPU utilization percentage.
- GPU utilization percentage.
- Available and Used Memory (measured in megabytes).

## 1.5 Thesis Outline

In Chapter 2, we will review prior research in the creation of a "Digital Twin" focusing on the topic of improving visualization techniques. Then we will review the usage of BIM data for the development of serious games that leverage videogame technologies, followed by a review of the first attempts at Levels of Detail inside BIM. We will also review the firsts attempts at the creation of virtual reality environments, starting with the “CAVE system” and finally expanding on the adoption of Head Mounted Displays as the current standard for VR consumption.

In Chapter 3 we will overview and list the multiple challenges found in the literature, these limitations will be grouped under common topics to discuss how can they be properly addressed.

Chapter 4 will overview the development of the proposed plugin / tool and its components. Its usage, inputs and outputs will be discussed and explained.

Chapter 5 will cover the comparative analysis. The computer software and hardware used for the comparison will be presented and discussed. Both sets of Revit game applications created will be compared by retrieving data from the Oculus Quest VR device, the methods used to retrieve this data will also be explained.

Afterwards, a new comparison component will be introduced. Both Revit game applications with an LOD system will be compared with similar applications that don't utilize any LOD system. This will clearly showcase the benefits that implementing an LOD system has in every case study.

In Chapter 6 the data gathered from all the case studies and Revit game applications will be showcased. Comparisons will be discussed as well as the limitations found. Finally, some concluding remarks will be provided in Chapter 7. Following these chapters, we will have our appendices including any extra information.

# Chapter 2: Related Work

The concept of an interactive Building Information Model (BIM) as a tool to improve the way information is conveyed and shared has already been approached by previous researchers with various degrees of success. In this chapter, I give an overview of the related work focusing on research that leverages the interactivity provided by rendering graphics in real-time.

## 2.1 The "Digital Twin"

(Woksepp, el al. 2008) developed a study involving the complete digital recreation of a construction site using 3D graphics. With special attention given to systems provided by subcontractors like ventilation, drainage, etc. For a total of more than 10,000 3D objects. This “VR Model” was showcased on a special meeting room on site using multiple projectors for an added sense of immersion. While skeptical at first, construction workers rapidly got accustomed at having meetings while exploring the site using its digital counterpart and praised the increased level of understanding they get of the project especially in areas outside of their profession. They can now easily detect errors in the design or placement of certain elements and interactively explore different alternatives.

The authors conclude that distributing information using 2D CAD drawings is ineffective and doesn’t provide the same level of understanding of the overall project as an interactive 3D model. They also acknowledge that designing in 2D and then creating a 3D representation for every object in every iteration of the design can be costly. They claim that to reduce costs designs could be directly implemented as 3D models. This provides

some of the fundamental ideals that paved the way for the development of the Building Information Modeling theorem.

## 2.2 Serious Games made from BIM

The steady advancements in real-time graphics and game engine technology piqued the interest of researchers like (Yan et al., 2011) trying to add BIM projects into a serious game. They theorized and explained a “crossover module” that would leverage APIs and programing languages to properly translate 3D models from BIM and their corresponding parametric metadata to game applications. The theorized module would also properly handle texture files from BIM projects and properly import them to a game engine. While a complete implementation of the theorized module is not presented, the authors do discuss and present a simple game prototype that allows the user to visualize a BIM project from different perspectives by dynamically changing from a first-person view to a third-person view while also providing dynamic on-the-fly controls using game controllers.

Developing computer games but with a meaningful serious purpose continued its expansion and is known as Triadic Game Design according to (Rüppel et al., 2011). Their paper presents a fire safety evacuation simulation using the game engine Ogre3D for rendering and a BIM made in Revit. The simulation done involves an explosion taking place inside the space. Two interfaces for BIM data extraction are discussed. The first one is claimed to leverage the IFC format (though this is never expanded later in the paper). While the second interface uses the Revit API to process BIM data via a plugin named "RevitToOgre" that generates separate files for 3D meshes, PhysX simulation, and model metadata using XML.

The authors only present one image of the simulated explosion inside the space showing that material textures are not supported. It's apparent that the authors are only focused on the graphical quality of the explosion with disregard for 3D mesh optimization, and even metadata implementation losing the concept of BIM.

Another interesting fire emergency evacuation simulation based on a BIM was developed in 2014 (Wang et al., 2014). The application allows users to create multiple custom virtual fire scenarios by loading desired emergency components through an inventory library. Behaviour scripts, particle systems and an adjustable path finding algorithm (based on semantic BIM data) were also implemented. The prototype splits BIM data into an FBX file for 3D meshes and a semantic information file (metadata). Data is merged back inside Unity using Object IDs. The authors discuss issues with low performance encountered when trying to test their prototype in multiple handheld devices. A simple BIM of a building floor was used.

Research focusing on human behavior simulation and dynamic fire evolution simulation was developed by (Park et al., 2018) using Revit to create the BIM, and Pyrosim DSF for realistic fire simulations. The Unreal Engine 4 was used for rendering and for the development of AI-driven characters as evacuation agents that avoid intense fire areas. Authors mention future work on how big data analysis could be employed to generate real-world situations of fire histories, drills and regulations making design decisions "more vivid" for architects.

A serious game using the BIM of a new hospital in Oslo was created by (Merschbrock et al, 2014). It was developed so that the 2500 healthcare professionals at work can familiarize themselves with the new space and contribute ideas for its design. A series of interviews were conducted with key persons to properly simulate their workdays inside the space. Changes in the design and layout were proposed to maximize productivity. The game was criticized since the hospital appeared to be empty and not busy like in real life. Design changes to the BIM had to be manually implemented inside the game version since the two are not connected.

A serious game case study presented by (Bille et al., 2014) leverages the Revit DB Link to export BIM metadata, 3D models are exported from Revit using the FBX format. The authors theorize that the Unity AssetPostProcessor API could be used to import metadata into Unity and attach it to the correct model, they advocate for dividing the BIM into two core sets of components to be processed separately. One for the 3D geometry data and the second one for the associated model metadata.

(Meža, et al., 2014) created a mobile augmented reality (AR) application to display a 4D BIM leveraging IFC data. The 3D model data was used by first converting it to the OBJ format using IFCOpenShell. The OBJs are then fed into the Layar3D Model Converter so they can be imported into the mobile environment. The complete 3D model would not run on mobile due to its complexity. So, the authors decided to only use the load-bearing structure. Tests show that faster, newer devices provide a better experience.

Authors never mention mesh optimization, poly count or frames per second suggesting a lack of understanding of these topics.

The research by (Pouke et al., 2018) allows for the visualization of parametric BIM data in real-time with sensor architecture implemented in two workflows for web-based 3D smart home visualizations using ThingSpeak (JSON format) and a WSDL SOAP webserver. Only temperature data was implemented providing only a single data type. XML tags generated by SOAP result in large file sizes and slower transfer speeds. The authors' mention that the Building Information Model was meshed manually using a low-poly approach since they consider that BIMs might be too complex for real-time utilization, they contain elements that must be removed, and must be converted in other formats to be utilized in game engines.

## 2.3 The Level of Detail inside BIM

The research done by (Boeykens, et al., 2011) not only discusses the concept of model complexity and its effects on performance, it also proposes a way to tackle this problem inside and outside of Revit. The authors showcase a parametric column model with a "resolution" parameter that controls 5 different LODs within. Custom "tagging" options through parameters in the BIM control which of the five meshes will be displayed on screen. On export, optimization is approached by hosting five different FBX files, each one with its own LOD, inside a custom object that renders only one model at a time depending on its distance to the user.

While this is the correct, game industry standard approach, the manual workflows presented cannot be implemented in real projects since they require five versions of every model inside the project.

## 2.4 The Dawn of VR, the CAVE system

In the quest of improving the ways a BIM can be visualized and experienced, (Kang et al., 2014) proposed the use of a CAVE system. This system places users in a room were multiple connected computers and projectors display a continuous image into the walls, floors, and ceilings. Covering the entire field of view of the user creating a sense of presence and immersion. The authors developed a plugin that connects and creates the same camera movement inside the same NavisWorks scene running in three different computers for a coordinated image in four different screens. Model metadata was not implemented or displayed.

The research by (Kieferle et al, 2015) adds interactivity to the CAVE system using body tracking inside the software COVISE. Users are fitted with a glove and a head mount containing tracking points visible to motion capture cameras. The glove can be used to teleport around the scene pointing at “Viewport” icons. It can also be used to selected and move objects. The user carries a tablet connected to the system that displays parametric metadata information of the selected object, this information can be changed using the tablet’s touch screen or inside a 3D UI that follows their head movement. The graphical capabilities of COVISE are shown to be subpar to commercial game engines.

In their case study of a cancer hospital, (Yu-Cheng Lin et al., 2018) propose an interactive BIM design application using the Unreal Engine and a CAVE system. The game was leveraged in a workshop to gather feedback on the space design from multiple groups of people. Users can select models and access their metadata contained in its own database while navigating the space with a game controller.

## 2.5 The arrival of the Head Mounted Displays

The development and commercialization of Head Mounted Displays ushered in the new era of interactive virtual reality. The complicated and expensive installations necessary for a CAVE system became a thing of the past. These new hardware technologies have sparked the interest of multiple researchers trying to correctly display their BIM data in an interactive Virtual Reality space.

(Shen, et al., 2012) created a serious game for virtual on-site visiting of building HVAC systems and based it around the newly released Oculus Rift Head Mounted Display (HMD) prototype. The Virtual Reality Toolkit plugin for Unity 3D was used to get their content in VR. A PHP based air-flow database was implemented within the Unity 3D web plugin for the training scenarios.

(Johansson, Roupe, et al., 2014, 2015 and 2016). Developed extensive research based around BIM, real-time graphics and the Oculus Rift prototype and published three papers. The first one details how they developed their own OpenGL rendering engine using the Oculus API to make it VR capable, it also includes the creation of a Revit plugin for

extracting BIM data. The second paper was based on the user studies conducted using the developed system.

The authors mention that all the participants praised the experience of being in a 1:1 scale with the digital building and consider it allowed then to better interpret and understand the space. An architect praised VR as a good way to communicate and share ideas with coworkers, stakeholders and others. It is also mentioned that it could be used for the decision-making process. The third paper is focused on the visualization of large BIM projects in real-time.

The authors present a comparative analysis of four commonly used software to create BIMs based on their real-time rendering performance. After a detailed performance comparison between the software and a detailed analysis of each one, Solibri OPT is claimed to offer the best performance. Afterwards, the development of their own BIM viewer is presented. It uses Visibility culling (occlusion) as its main acceleration technique and managed to outperform Solibri OPT. Finally, the authors mention the need for LODs and data streaming techniques to be implemented for a better performance in general.

A Comparative study were user performance was evaluated on a set of identical tasks taking place in a physical office environment and in an immersive VR replica was developed by (Heydarian, et al., 2015). A BIM of the space was created in Revit and manually polished in 3DS Max. The Worldviz Vizard VR software was used for rendering and it was connected to an Oculus Rift DK1, Microsoft Kinect 1 and a Wand Tracker. 112 participants completed questionnaires after performing the real and VR tasks. Results showed that the VR experience was a satisfactory representation of the real environment.

No significant difference was discovered in the performance of users between the VR or physical spaces during everyday office-related tasks.

Only space navigation was not perceived as realistic since the users were not physically moving around the space, the users also pointed out the low resolution of the DK1 screens. This research provides evidence that VR spaces can be used to study user behaviour, document user preferences and measure their performance inside architectural spaces. Providing useful data that can be used to improve the design of any space.

(Dinis, et al. 2017) present multiple small AR and VR prototypes developed by Masters students in an introductory class using CAD with manual translation of models into Unity 3D. These prototypes are designed to teach K -12 and pre university students about the role and professional activities of a Civil Engineer. They were presented in an outreach event were the authors used a Likert Scale to survey the students. Data gathered shows that 75% of the students considered that the VR interface was significantly important to understand the concepts presented.

(Yang, et al. 2018) present a VR application focused on Facilities Management. Unity and the Virtual Reality Toolkit (VRTK) API are leveraged to add interactive pointer controls that overlay model metadata on the screen per laser point or controller touch from the user. The authors claim that Unity and C# allow for a higher degree of customization when compared to Fuzor, a Revit plugin that offers native integration tools.

Fuzor needs and external database based on Excel to display metadata, its API provides limited functionality in the development of customized interactive applications, and its graphical quality is subpar to game engines like Unity or UE4.

The implementation of a Virtual Reality serious game for a Public Library was created by (Minyaev, et al. 2018). Workshops conducted brought together researchers, library staff and library users to develop ideas and preliminary concepts. The experiences developed were selected based on the workshop proceedings, this are a VR book search and book recommendation menus and a VR art gallery. Finally, only one installation was created inside the library with no user study conducted to determine user satisfaction, or other research topics.

The research by (Boeykens, et al., 2018) proposes a more polished game based on a BIM that splits-up complex projects into separate modules, adding extra detail to the project by incorporating complex facades created in modeling software like 3DS Max. Extra interactivity is also included, utilizing animations that present the assembly of the model, with explanatory text and dynamic color-coding of the models. The authors mention how most current workflows still rely on splitting the transfer of geometry and information, usually into separate formats and files and treating them as separate components. A better and faster integration could be achieved by using IFC files that contain and process both model data and parametric metadata.

(Wong et al., 2019) present an experience-based interactive lighting design approach that simulates multiple lighting approaches inside a BIM, Unity and the FBX file format. Authors used the Universal Material Converter plugin (UMC) to manually recreate the materials in 3DS Max. There is no mention of metadata being connected back to its corresponding model inside Unity.

## 2.6 **Real-time communication between BIM and Game applications**

Multiple research has focused on having direct, back and forth communication between a BIM project and its game counterpart. The work by (Edwards, et al., 2015) proposes the use of a Unity webserver connecting the game application to the BIM project inside Revit. Unity server and client modules run the same game. The client modules can place and move elements in the scene, this is interactively replicated in the server module. A user-study was conducted with pre and post surveys, it only compared multiple input devices ranging from Kinect, HMD and tablets to regular mouse and keyboard. While the multiplayer networking features inside Unity allowed multiple users to interact inside the game, the proposed goal to translate user changes inside the game engine back to the BIM project was not achieved.

(Hilfert, et al., 2015) conceptualized advanced interactions using the Oculus Rift with a Leap Motion device attached for hand tracking and the Microsoft Kinect for full-body tracking. The authors theorized workflow includes an open source BIMServer with user credentials to centralize data. A c++ plugin to translate IFC models into UE4 actors. A material database combining BIMServer data with UE4 materials.

Network connectivity between multiple clients. And finally, a method for detecting already existing IFC models and instancing new copies. These concepts are only theorized and explained with diagrams. No prototype is presented, only one test involving an IFC model imported into UE4 is discussed and shown in a figure. Referring to this test, the authors mention lag spikes, and the FPS data presented is below VR standards. Metadata integration is not shown. Future work is needed to demonstrate the feasibility of the complete proposed methodology and its theorized components.

(Kado, et al. 2018) developed a VR application that incorporates a two-way cooperation system were changes done in VR are reflected in the BIM and vice-versa. ArchiCAD 20 and UE4 are combined with a relay database based on PostgreSQL.
A Delta Update Function uses timestamps and unique IDs given to all the components.
If a timestamp changes then the database looks inside the element for more changes to automatically update things like elements, parameters, materials and meshes. Even with a simple BIM project, the app performance is way below VR standards. Delays of several seconds occur when trying to visualize any changes made to the scene, most likely due to large file data conversions done inside the server.

Researchers (Grandon, et al. 2018) envisioned the creation of a unique tool that will allow for BIM files to be directly translated to Game engines with minimal steps in between. Unity Assetbundles are used to encapsulate models, their materials, and even whole scenes. An FBX file is used within a custom Unity virtual machine to automatically generate Assetbundles and give them an ID.

These are saved inside a server setup using RESTful protocols. The local Unity software connects to the server and displays a list of the available Assetbundles, downloading them on request, saving them temporally and loading the scene in VR. Sadly, the concept of BIM was lost since no parametric metadata was implemented or discussed. The issue relies on the workflow's dependence on the FBX format. Only this format was used when extracting BIM data and when creating Assetbundles, limiting the capabilities of the game application.

An interesting BIM inside VR system created by (Du, et al. 2018) interactively synchronizes data between a Revit project and the Unity 3D application. A Cloud Server reads and constantly updates elements in the project based on Element IDs.
The application creates requests for updated information every few seconds, tagged elements in the database are recognized as changed and automatically requested. A pointer system displays Revit BIM metadata inside Unity. Authors mention that big BIM projects with multiple objects will require more computing power in the searching / scanning process, slowing down the synchronization and creating lag. Mesh optimization is not discussed, most likely not needed for the simple BIM showcased.

The application by (Nandavar, et al. 2018) lets users interact and manipulate models in VR and synchronize the changes to the original model, it is based on the vendor-neutral platform OpenBIM. Data exchange between BIM and VR based is based on the IFC and XML formats using the xBIM Toolkit (open-source SDK for IFC).

A prototype is presented with three different scenarios showcasing different levels of complexity, after the user terminates the VR application, the system creates a new IFC file containing all user changes as theorized. The Realtime performance in FPS of the three different scenarios is not discussed. Optimization of complex BIM components is not considered. Screen freezes when querying the file for model metadata display. Materials and textures inside the BIM are not supported. The plugin requires the user to pick a color for the different model categories. The authors state that the system lacks the capability to handle large BIM projects and is yet to achieve a high-quality rendering of the model in VR.

(Liu et al., 2019) Present a Web3D based real-time visualization of large-scale BIM scenes, including data light weighting, indexing, semantics analysis and scene management. The proposed algorithms for semantics light weighting, repetitive objects removal and scene index structure are implemented on Away3D using c++, sadly, they are roughly explained. The authors used an engine based on obsolete technologies like Flash3D and Action Script. Optimization is done to the B-REP geometry with no discussion on meshing and its benefits. Only a few pictures of the prototype are shown, and they show poor graphical quality with no materials, no textures and no implementation of metadata. It is mentioned that the application runs at around 30 FPS which is not acceptable by today's standards.

(Chenaux et al., 2019) present a new concept the labeled City Information Modeling (CIM). It combines geospatial data and building attributes and behaviors into an integrated multiplatform. Bringing together BIM with GIS. Python was used to import a SketchUp file into ArcGIS. Datasets are handled by a PostgreSQL spatial database. Geomagic Wrap was used for meshing. The authors mention limitations in the manual segmentation and conversion of point cloud data from Lidar scans. Also, mesh optimization was not discussed or implemented. The performance of the application was not discussed, metadata integration not discussed. Processing times are elevated for these large datasets.

# Chapter 3: Gap Analysis

As shown in the previous section, companies in the Architecture, Engineering and Construction field have had various degrees of success in implementing game technologies, while encountering multiple limitations. In the next section I give an overview of the most common limitations found in the literature.

## 3.1 Summary of Existing Limitations

### 3.1.1 Manually recreating models

As mentioned before, in order to correctly import 3D models from a BIM project to game engine applications they need to be translated from B-REP models to faceted polygonal models. Various researchers decided not to translate their models or materials and opted for their manual recreation inside digital modeling software like 3DS Max or Maya.

The workflow by (Boeykens, et al., 2011) requires the manual modeling of each of the five different levels of detail. On their serious games, (Merschbrock, et al., 2014), (Vajak, et al., 2017), (Dinis, et al., 2017), (Yu-Cheng Lin, et al., 2018) and (Pouke, et al., 2018) only used the BIM as a reference for the remodeling of the whole environment. They mentioned this was a time-consuming, work-intensive and tedious. The prototype developed by (Bille, et al., 2014) and (Wong, et al., 2019) required the manual recreation of materials using 3DS Max and the UMC plugin. For their VR experience, (Yang, et al., 2018) used Maya for the manual optimization of meshes and for material conversion.

### 3.1.2 Parametric Metadata Ignored.

One of the cornerstones inside a BIM is its inclusion of the parametric metadata of every model and element inside the digital scene. The fire safety evacuation simulation by (Rüppel, et al., 2011), the virtual on-site visiting app by (Shen, et al., 2012), the hospital BIM game by (Merschbrock, et al., 2014), the VR Revit plugin by (Johansson , et al., 2015), the VR serious game by (Hilfert, et al., 2015), the multiuser BIM based collaborative app by (Edwards, et al., 2015), the Web3D app by (Liu, et al., 2016), the virtual tour by (Grandon, et al., 2018), the human behavior simulation and dynamic fire evolution simulation by (Park, et al., 2018), all of them glanced over, or completely ignored the parametric metadata that every BIM project requires. Without the parametric metadata, the project becomes just an interactive architectural visualization and can no longer be considered a BIM.

### 3.1.3 Utilizing a Basic BIM.

The research done by (Edwards, et al., 2015), (Kieferle, et al., 2015), (Dinis, et al., 2017), (Du, et al., 2018), (Kado, et al., 2018), used a simple BIM model of only walls, floors, doors, and stairs for their prototypes. A model like that does not exemplify a real-life BIM project.

### 3.1.4 No mesh optimization performed.

Authors like (Rüppel, et al., 2011), (Wang, et al., 2014), (Hilfert, et al. 2015), (Grandon, et al., 2018), and (Park, et al., 2018) decided to handle mesh translation by exporting an FBX file from Revit, this properly translated 3D models, but it did not optimize them in any way. Larger and more complex models caused low performance. The mobile AR application by (Meža, et al., 2014) could not load the complete 3D model due to its complexity. So, the authors decided to only use the load-bearing structure, limiting the use and scope of the application. The work by (Kieferle, et al., 2015) mentions a database optimized for "rendering and interaction" but only simple models are showcased running in the system with no discussion on its performance. In the case of (Du, et al., 2018) and (Chenaux, et al. 2019) the lack of optimization meant larger datasets that increased processing times.

A lack of proper mesh optimization in the work of (Nandavar, et al., 2019) caused screen freezes when querying the file for model metadata display. The authors stated that the system lacks the capability to handle large BIM projects and is yet to achieve high-quality rendering of a model in VR.

The work by (Johansson, et al., 2015) mentions that LODs and data streaming techniques should be implemented to achieve a better performance in general. As mentioned before, the work of (Boeykens, et al., 2011) discusses the concept of model complexity and its effects on performance, with the authors proposing the manual creation of five different levels of detail for their model.

### 3.1.5 Merging back BIM Components.

The work by (Rüppel, et al., 2011), mentions the creation of separated files for meshes, PhysX simulations, and model parameters using XML files. In their paper, (Bille, et al., 2014), advocate for dividing the BIM into two core sets of components to be processed separately. One for the 3D geometry data and the second one for the associated model metadata without any explanation on how to merge these components back. (Boeykens, et al., 2018) mention how most current workflows still rely on splitting the transfer of geometry and information, usually into separate formats and files and treating them as separate components. They state that a better and faster integration could be achieved by using IFC files that contain and process both model data and parametric metadata.

## 3.2 Proposed Solution

This research addresses all the presented limitations. The first limitation presented is based on the manual recreation of models. Our solution uses the 3D models inside the Autodesk Revit file and translates them to be used without the need of any manual work.

The second limitation presented is based on ignoring or discarding the parametric metadata. Our solution retains the parametric metadata from Revit inside every single model so that it can be accessed inside the real-time application.

The third limitation presented is based on previous work only using a simple BIM dataset. Our solution presents four different Revit projects specifically chosen to exemplify the different levels of complexity that a BIM can reach.

The fourth limitation presented is based on previous work not discussing or implementing mesh optimization. Our solution addresses mesh optimization with the automatic generation of four Levels of Detail (LODs) for every model. This LODs dynamically change based on the size of the model on screen.

Finally, the fifth limitation presented is based on previous work separating BIM data into multiple files and the issues related to bringing the data back together. Our solution keeps all the models, levels of detail, parametric metadata, materials and textures inside a single file.

# Chapter 4: Proposed Workflow Design.

## 4.1 Overview.

This chapter will cover the development of a new optimization framework for Building Information Models, implemented as a plugin / tool inside Autodesk 3DS Max. This plugin / tool was created entirely using Maxscript, the built-in scripting language of 3DS Max (Autodesk, 2019) and leverages new scripting functions that expanded the interoperability between Revit and 3DS Max in recent versions of the software.

The steps taken by this tool are:

1. Read and translate Revit data into 3DSMax (preserving metadata, materials and textures).
2. Get every model and create three extra versions of it, each with a different model complexity and lower triangle count.
3. Check and fix possible mesh and lighting errors (by welding vertices, unifying normals and recreating smoothing groups).
4. For every model, create a new UVW map channel with a new set of UV coordinates to be used for light baking.
5. Group models with the same name and add them to their own LOD system.
6. Convert the Autodesk materials applied to the models to standard materials that can be read by inside a game engine.
7. Export the scene using the FBX format.

Figure 2 presents a graphical representation of these steps:

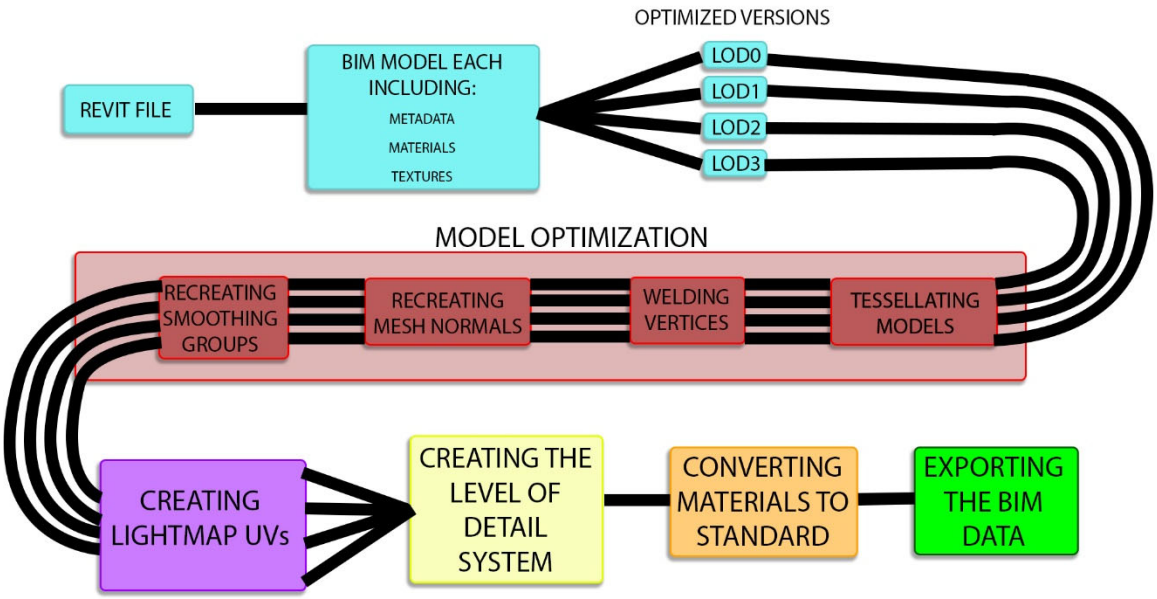


**Figure 2 shows a diagram representing all the steps the script takes.**

## 4.2 Revit translation using functions in Maxscript.

Now the main Maxscript functions used for translating BIM data will be presented. We started by creating the main window to house all buttons that will run my code. This is done using the "newRolloutFloater" function, it takes the name of the window and its width and height.

```
(
    MainFloater = newRolloutFloater "DIEGOZ REVIT TOOL" 350 700
```

**Figure 3 Function that creates the floating window housing the script.**

We had to make sure that the units are properly setup inside the software, the Unreal Engine's default units are centimeters, so this unit was used.

```
--SET UNITS TO CENTIMETERS
units.SystemType = #Centimeters
```

**Figure 4 Function that sets units to centimeters.**

The first button is implemented using a "button" function. When pressed this will open a window asking the user to locate a Revit file.

```
button FileNames "LOCATE RVT FILE" width:200 height: 40
  on FileNames pressed do
    (
      zfile = getOpenFileName caption: "Open an .RVT file:"
```

**Figure 5 Button Function.**

The "RevitEngine" custom function was used to turn off certain elements in the process since by default the software translates elements like cameras, lights and the daylight environment system. They are not needed in my opinion since they do not translate properly to UE4 (specially the Revit environment system).

```
-- Turn off lighting and other objects
RevitEngine "camera" false
RevitEngine "light" false
RevitEngine "daylight" false
```

**Figure 6 Function to ignore certain components inside the BIM data.**

Afterwards, the combine options had to be set. The software can combine multiple objects inside the Revit scene, it can combine all objects into a single mesh, combine by family, combine by category, combine by material, and others.

This could be useful to improve performance, but sadly the parametric metadata for every object is lost when combining. This metadata is needed to properly represent BIM data, so the combine option must be set to not combine.

```
--Option to combine Revit models:
RevitEngine "combineoption" 0
```

**Figure 7 Setting the combine option.**

Both “view detail” and “simplify mesh” mesh optimization functions provided by the software were applied and set to their maximum values for every LOD version. Now, the main “geometry detail” function was used to create the necessary optimized versions with lower triangle counts. The LOD0, LOD1, LOD2 and LOD3 versions of every model were made running the code four different times with different "geometry detail" values.

```
-- LOD1
RevitEngine "material" false
RevitEngine "viewdetail" 3
RevitEngine "geometrydetail" 9
RevitEngine "simplifymesh" 0.5
RevitEngine "extrudedetect" true
```

**Figure 8 Setting the mesh optimization functions for the LOD1**

The next step is to handle materials. The material function translates any material and texture found in the Revit data. This works as expected, but when using this for the LOD system, issues arose. The code was creating four different versions of each model and every version was bringing their own material (by default 3DS Max just adds an incremental number like “_2” at the end of every new material name and moves on without using the existing material). This means that every material and texture are repeated four times, this is not acceptable and will affect performance.

All LODs inside a model need to share the same materials for optimization purposes. To achieve this, only the main instance of the LOD system (LOD0) was set to translate the materials and textures from the Revit data, then the other 3 LODs were set to ignore their materials and had a standard placeholder material applied to them, avoiding material duplicates.

```
myNewMat = multimaterial numsubs: 7 name:"newMultimaterial"
```

**Figure 9 Placeholder material added to LOD1 - 3.**

### 4.3 Automating the model optimization.

With the models and their proper metadata, materials, textures and LOD versions in place inside 3DS Max some extra steps need to be taken to avoid any possible mesh errors that might occur inside the Unreal Engine.

#### 4.3.1 Tessellating the models.

The first step is to fully convert all the meshes to triangles. The previous process still maintains some NURBS curves features specially in models with multiple curved surfaces. The "ProOptimizer" modifier function was called within Maxscript to leave only the triangles inside the mesh.

```
addModifier obj (ProOptimizer()) ui:on
obj.modifiers[#ProOptimizer].KeepUV = true
obj.modifiers[#ProOptimizer].LockUV = true
obj.modifiers[#ProOptimizer].Calculate = on
```

**Figure 10 ProOptimizer Modifier Function**

### 4.3.2 Welding vertices to seal the meshes.

With all the meshes fully tessellated, they need to be sealed to avoid any light leaks or black spots when viewed inside the Unreal Engine. Maxscript functions selected all the vertices inside a mesh and welded them using a threshold of 0.01. This value targets only open faces and keeps the rest intact.

```
subobjectLevel = 1
obj.weldThreshold = 0.01
max select all
polyop.weldVertsByThreshold obj #all
update obj
```

**Figure 11 Welding the vertices inside every object.**

### 4.3.3 Recreating the mesh Normals.

The code selects the faces inside all the sealed meshes and resets their normals. This fixes any shading errors that might have occurred in the translation and the tessellation process.

### 4.3.4 Recreating the Smoothing Groups.

The code can also select all the faces inside all the sealed models and auto-smooth them based on their coordinates and normals. This improves the look and flow of curved models especially of curves with lower triangle counts in their LOD3 versions.

## 4.4 Creating Lightmap UVs

The performance limitations of developing for mobile Android platforms require that all the shadows inside a scene are baked and displayed as a texture. This extra texture is then applied to any models receiving shadows. To achieve this, an extra UV map channel needs to be created. This new set of UVs saved to the second map channel are commonly known as “Lightmaps”. The “UVWUnwrap” modifier function was used inside Maxscript in conjunction with the “Flatten Mapping” algorithm to automatically create these maps for every LOD version of every model.

## 4.5 Creating the LOD System.

With all the LOD versions created and optimized they were added to an LOD system that tells the Unreal Engine that they belong together and should be imported as the same Static Mesh. To achieve this the Maxscript code inspects the names of all the models in the scene and groups the ones with the same name. Then the code runs functions inside the Level of Detail utility to convert the simple group into an LOD system that can be properly read by the Unreal Engine.

```
setCommandPanelTaskMode #utility
UtilityPanel.OpenUtility Level_of_Detail
```

**Figure 12 Part of the utility function that creates the Level of Detail systems.**

### 4.6 Converting Materials to Standard.

In their current form, the materials applied to all the models in scene are not supported by the FBX exporter or the Unreal Engine. All the materials and bitmap textures need to be converted to the standard material. The last section of the plugin transforms all the Autodesk Bitmap texture functions into standard functions. Autodesk Noise components are not supported and should be deleted (this will be covered in detail in the discussion and limitations section). Finally, the Scene Converter utility was used to convert every Autodesk material from Revit into a Standard material.

### 4.7 Exporting the Data.

Finally, the complete scene is exported using the Autodesk FBX format. This unique file holds every model within its own LOD System including parametric metadata and materials with their corresponding textures.

## 4.8 Overview of the Plugin UI

The following figure showcases the UI of the Plugin and how every button includes multiple labels that explain their functionality.

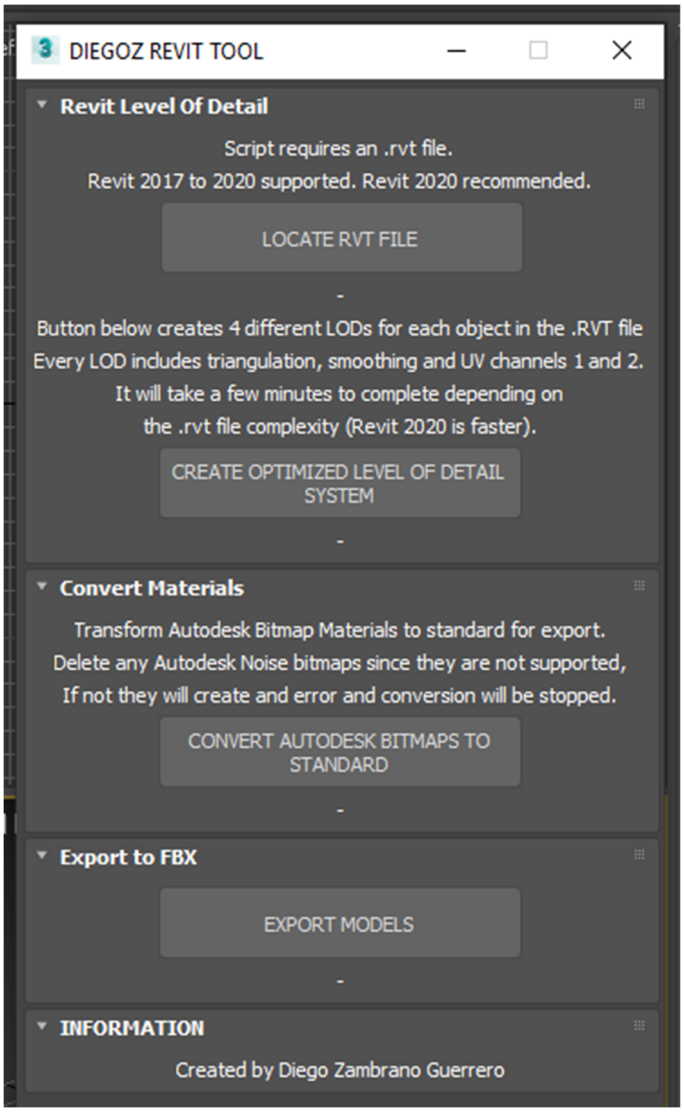


**Figure 13 The plugin UI inside 3DS Max.**

# Chapter 5: Comparative Analysis.

## 5.1 Head Mounted Displays.

When conceptualizing and designing a new structure, flat sketches, blueprints and spreadsheets of data can only convey a limited portion of the design concept and intention that architects and engineers have in their minds. The development and adoption of Head Mounted Displays (HMDs) that track user movement and replicate it inside a virtual environment has disrupted multiple fields due to the level of immersion they provide (Johansson, Roupe, et al., 2014, 2015, and 2016). The sense of presence provided by these technologies is exactly what architects and engineers need to properly get their ideas across. Now, the "Digital Twin" derived from the BIM interactive database can move from a flat 2D screen to an immersive digital environment were designers, contractors, investors and ultimately buyers can literally take a walking tour inside it and experience everything firsthand and at real scale.

## 5.2 Equipment and Hardware.

The same computer was used to process and create every application. It has the following specifications:

- Windows 10 Pro.
- Intel Core i7-6820HK processor.
- NVIDIA GeForce 1070 graphics card.
- Samsung 970 PRO NVMe SSD boot hard drive with a capacity of 512 gigabytes.
- Samsung 970 EVO Plus NVMe SSD data hard drive with a capacity of one terabyte.

Software used includes Autodesk Revit 2020 and Autodesk 3DS Max 2020 under a student license. The Unreal Engine 4.23.1 version was used with the Unreal Datasmith plugin also on its 4.23.1 version. NVIDIA Codeworks for Android version 1R7u1 was used to compile and deploy the applications. Finally, the Oculus Quest mobile Virtual Reality device was used to run all the Revit game applications. Metrics were obtained using the OVR Metrics Tool version 1.4 software provided free of charge by Oculus. This software runs in the background inside the VR device and exports multiple metrics as .CSV files that can be retrieved inside Windows.

## 5.3 Evaluation Criteria.

The comparison metrics for this study were taken from the Oculus developer documentation (Oculus, 2019) and represent the standard data that should be gathered to measure and troubleshoot the performance on an application inside the VR device. They are as follows:

### 5.3.1 Average Frames Per Second.

This represents the number of frames that the hardware can render every second. This number is limited by the display refresh rate of the device. The Oculus Quest displays have a refresh rate of 72 frames per second (FPS), meaning that applications should run at 72 FPS for optimal performance. Applications running at 60 FPS are still considered acceptable, while applications running at less than 30 FPS will most likely cause discomfort on the user (Oculus, 2019).

### 5.3.2 CPU ad GPU Utilization.

This represents the percentage of processing power that is currently in use by the application. Both for CPU and GPU are treated and examined separately. Since, optimizing CPU performance is not the same that optimizing GPU performance.

### 5.3.3 Available Memory.

This number represents the amount of free memory available inside the device. This metric is measured in Megabytes. We used it to track real-time memory usage of every application.

## 5.4 Why the Oculus Quest.

While any HMD can provide an immersive experience, the latest device released by Oculus on May 2019 provides an elevated sense of freedom and range of movement that is not available to tethered devices. The Oculus Quest device is a self-contained unit that leverages smartphone technologies to stream, download and render real-time applications or video while at the same time tracking the environment around it and the two controllers representing the hands of the user. This device was chosen not only for its extra level of immersion and freedom of movement, but also for its ease of use, and even easier setup.

Users can just wear the HMD and hand controllers, and everything works as expected. This eliminates the need for intricate setups were users need to mount external tracking stations that need their own power source, and USB connection to a high-end computer. Then, after setting the hardware, the user must run a long calibration test to make sure the sensors, HMD and controllers are properly tracked.

The Oculus Quest still has multiple limitations. The most important being the limited graphics rendering performance coming from the Qualcomm Snapdragon 835 mobile processor. The device is also battery powered, so graphically intense applications will drain the battery in just under two hours.

## 5.5 Case Studies and Results

In this section I will present every case study and the recorded data results, both from the application developed using our proposed framework (from now on referred to as RVTSCRIPT) and the application developed using the Unreal Datasmith toolset (from now on referred to as DATASMITH). As mentioned before data was recorded as .CSV files using the OVR Metrics Tool application. The user followed a specific walkthrough inside the application, with designated stops inside the environment. This kept the data consistent between the applications.

### 5.5.1 Small House Concept

This case study showcases a small Revit project of a single space house. The house includes multiple furniture and cabinet models as seen in figure 21.

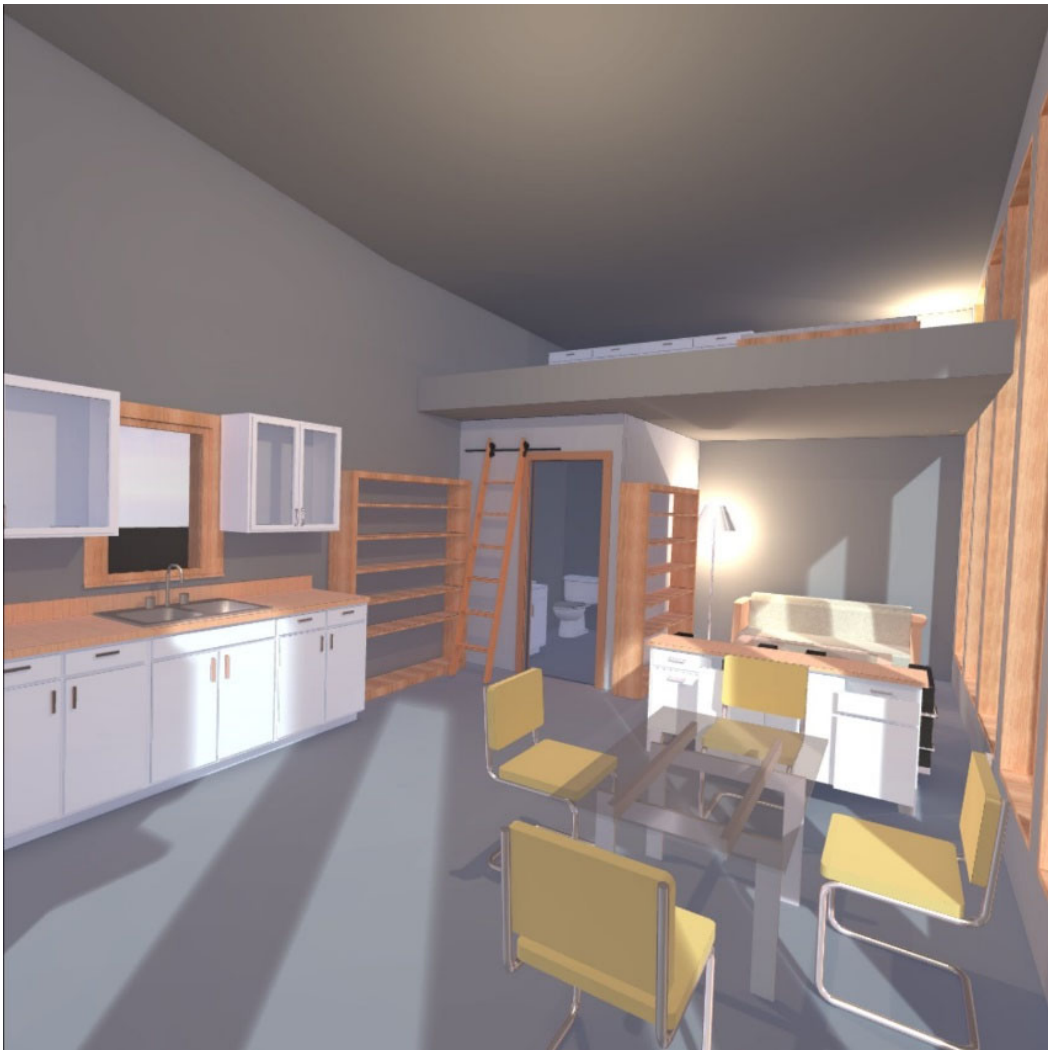

**Figure 14 Image of the single space house captured inside the Oculus Quest.**

The first criteria compared was the Average Frame rate. The optimal results should be as close to 72 Frames per Second as possible. Data can be visualized in Figure 18.

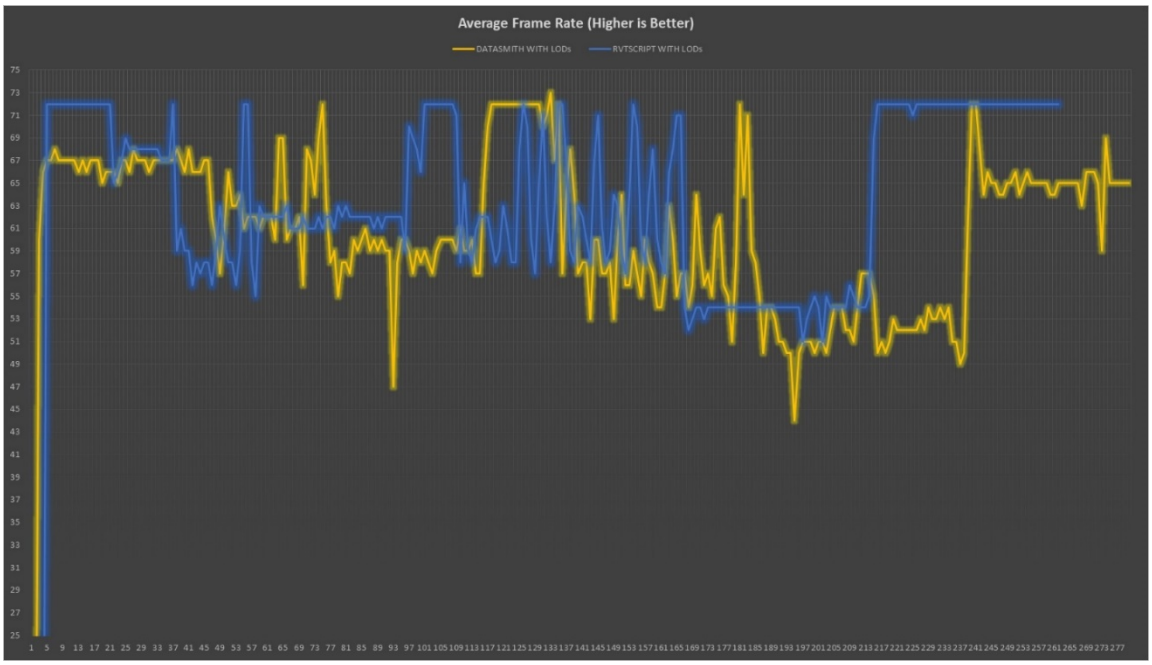


**Figure 15 Average Frame Rate data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 62.73 | 62 | 72 | 72 | 51 |
| DATASMITH | 60.54 | 61 | 67 | 72 | 44 |

**Table 1 Analyzing the Average Frames per Second data.**

Data shows that the RVTSCRIPT application reaches the optimal mode of 72 FPS with a minimum value of 51. While the DATASMITH application only reaches a mode of 67, with a minimum value of 44. Based on this it can be considered that the RVTSCRIPT application has an improved performance of around 5 FPS from the DATASMITH application.

The second criteria compared is the CPU utilization within the VR device. Data can be visualized in figure 19.

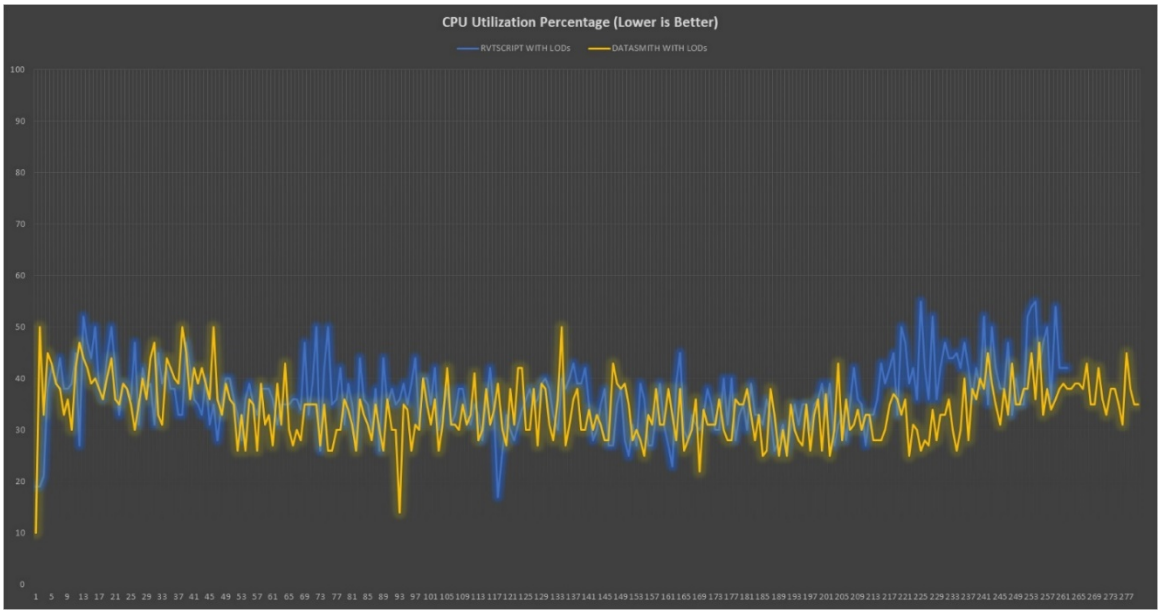


**Figure 16 Average CPU Utilization data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 36.65 | 36 | 35 | 55 | 17 |
| DATASMITH | 34.07 | 34.5 | 30 | 50 | 14 |

**Table 2 Analyzing the CPU Utilization data.**

Data shows that the RVTSCRIPT application reaches a mode of 35% with a minimum value of 17% while the DATASMITH application reaches a mode of 30% with a minimum value of 14%. Based on this it can be said that the RVTSCRIPT application CPU utilization is around 5% higher than the CPU utilization of the DATASMITH application. Since both applications are only using about half the processing power of the CPU inside the device, it can be said that this metric doesn't affect the overall performance of both applications.

The third criteria compared was the GPU utilization within the VR device. Data can be visualized in figure 20.

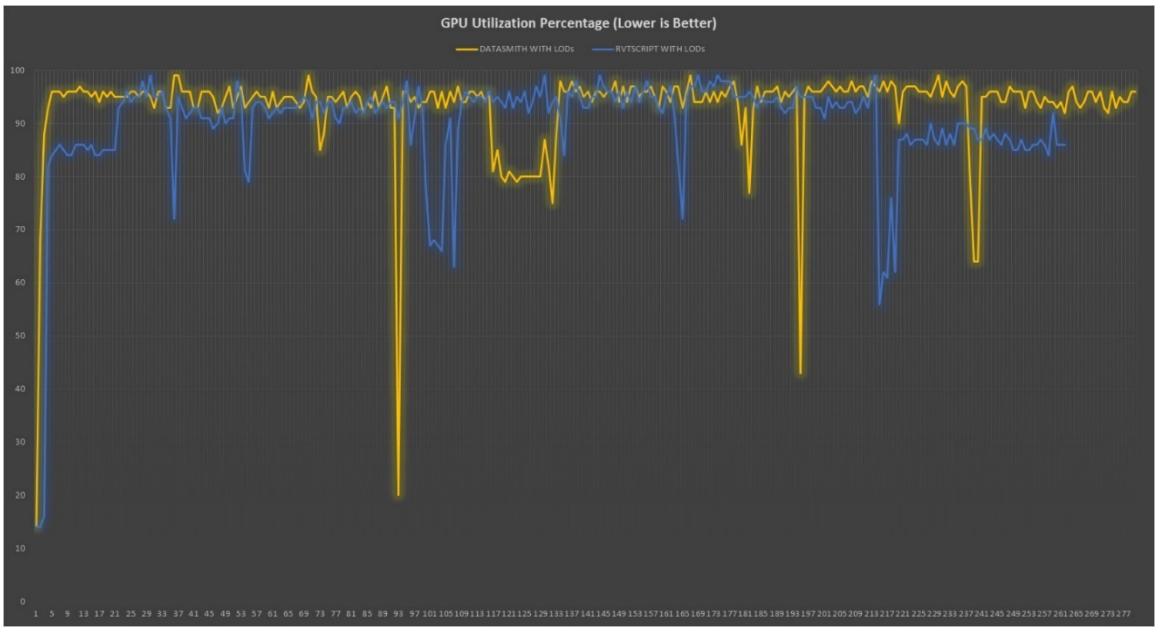


**Figure 17 Average GPU Utilization data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 89.87 | 93 | 95 | 99 | 56 |
| DATASMITH | 93.18 | 95 | 96 | 99 | 20 |

**Table 3 Analyzing the GPU Utilization data.**

Data shows that the RVTSCRIPT application reaches a mode of 95% with a minimum value of 56% while the DATASMITH application reaches a mode of 96% with a minimum value of 20%. Based on this it can be said that both applications are highly dependent on the processing power of the GPU. It can also be said that the RVTSCRIPT's GPU utilization mean is 3.31% lower than the GPU utilization of the DATASMITH application.

The fourth criteria compared was the available memory in the VR device while the application is running. Data can be visualized in figure 21.

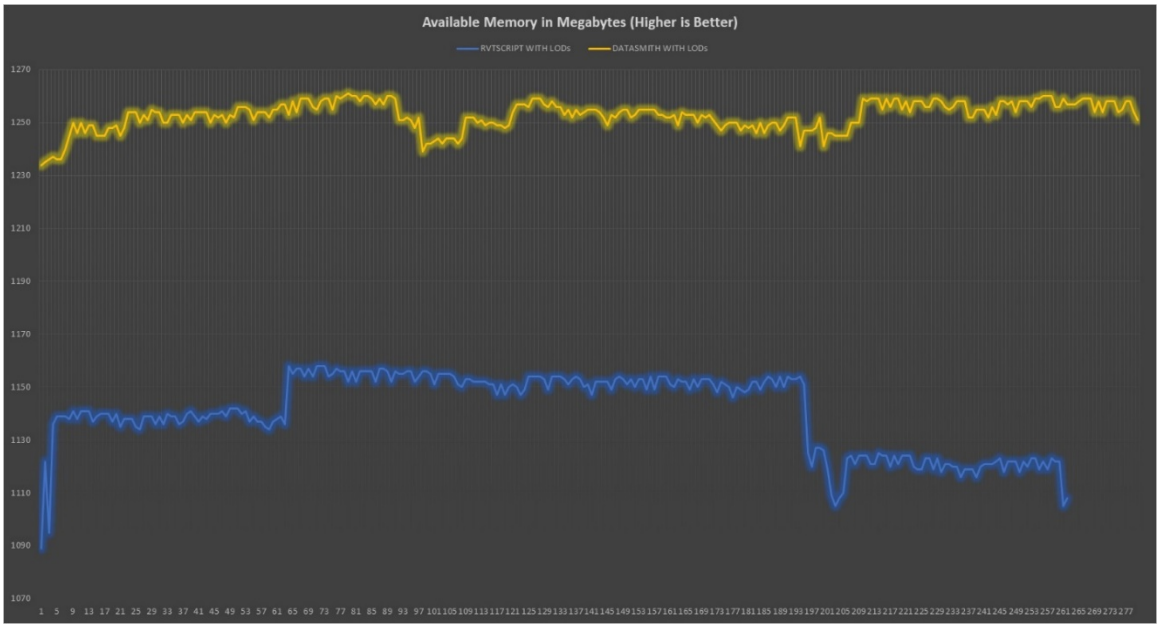


**Figure 18 Available memory data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 1140.63 | 1146.5 | 1154 | 1158 | 1105 |
| DATASMITH | 1252.81 | 1254 | 1259 | 1261 | 1239 |

**Table 4 Analyzing the Available memory data.**

The available memory metric gathered by the software also tells us the amount of memory that is being used. Data shows that the RVTSCRIPT application has a mode of 1154 megabytes available while running, while the DATASMITH application has a mode of 1259 megabytes available. After analyzing the data is clear that the RVTSCRIPT application uses around 100 megabytes more than the DATASMITH application. As stated before, the total memory of the device is 4000 megabytes (4gb).

This means that both applications still have more than a quarter of the total memory available. So, it's safe to say that this metric is not impacting the performance of both applications.

Finally, the applications with an LOD system were compared to identical applications without an LOD system. These extra applications are labeled as LOD0 (highest quality and triangle count), and LOD3 (lowest quality and triangle count). Based on their importance for the performance of the application, only the Average Frames per Second and Available memory metrics were used.

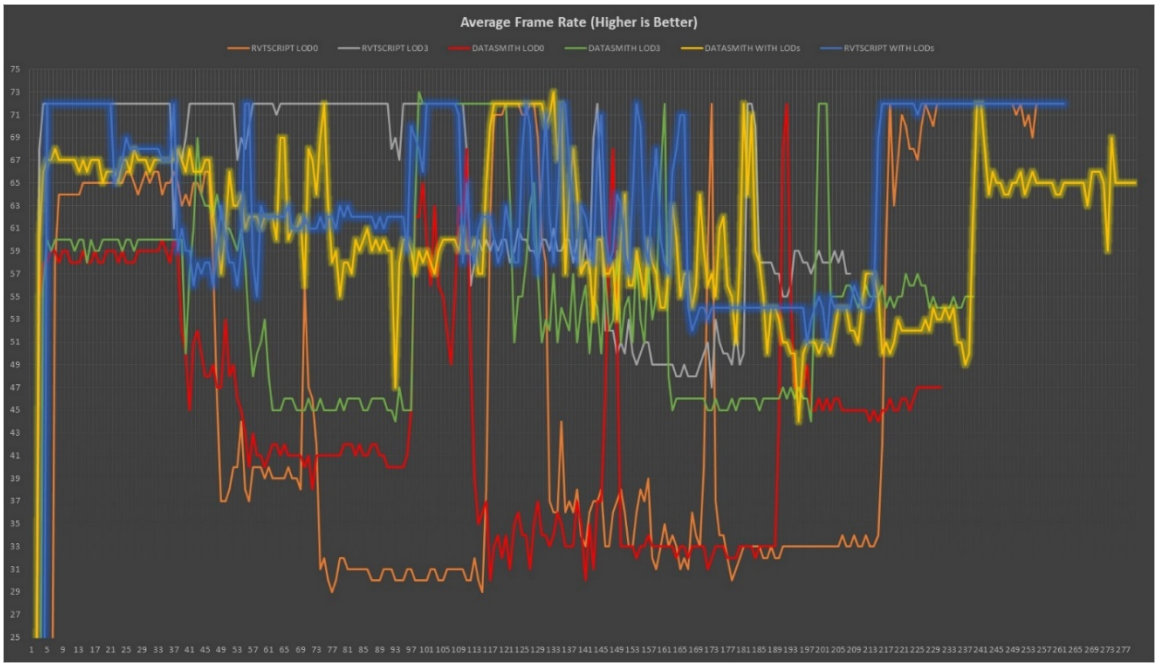


**Figure 19 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 62.73 | 62 | 72 | 72 | 51 |
| DATASMITH WITH LODs | 60.54 | 61 | 67 | 72 | 44 |
| RVTSCRIPT LOD0 | 46.33 | 38 | 33 | 72 | 29 |
| DATASMITH LOD0 | 44.07 | 43 | 33 | 72 | 30 |
| RVTSCRIPT LOD3 | 63.57 | 69 | 72 | 72 | 47 |
| DATASMITH LOD3 | 54.41 | 55 | 46 | 72 | 44 |

**Table 5 Analyzing the Frame Rate data from six different applications.**

The data analysis revealed that both applications on their LOD0 version only reached a mode of 33 FPS. Almost half the performance of that of their counterparts with LOD systems. Showcasing the importance of the LOD system.

The RVTSCRIPT LOD3 version reached the desired 72 FPS as expected but the DATASMITH LOD3 version struggled and only reached a mode average of 46 FPS.

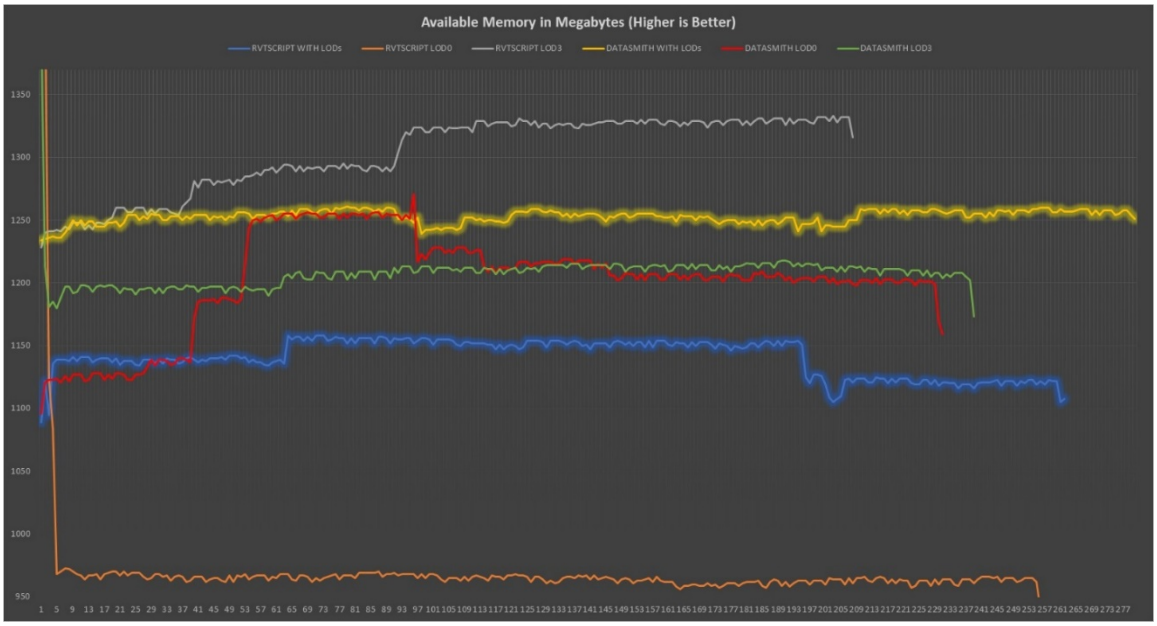


**Figure 20 Available memory data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 1140.63 | 1146.5 | 1154 | 1158 | 1105 |
| DATASMITH WITH LODs | 1252.81 | 1254 | 1259 | 1261 | 1239 |
| RVTSCRIPT LOD0 | 969.25 | 965 | 966 | 973 | 940 |
| DATASMITH LOD0 | 1202.01 | 1206 | 1255 | 1271 | 1121 |
| RVTSCRIPT LOD3 | 1303.16 | 1323.5 | 1329 | 1332 | 1213 |
| DATASMITH LOD3 | 1207.49 | 1209 | 1213 | 1218 | 1173 |

**Table 6 Analyzing available memory data from six different applications.**

The RVTSCRIPT LOD0 version uses around 200 megabytes more than its counterpart with the LOD system and it's the version that uses the most amount of memory. The RVTSCRIPT LOD3 version uses the least amount of memory followed by DATASMITH WITH LODs version. The data revealed that all 6 versions still have a considerable amount of memory available, so memory usage is not affecting their performance.

### 5.5.2 Kitchen Room

This case study represents a real-life Revit project double the size of the previous case study. It includes two rooms with multiple different appliances and extras as seen in Figure 24.

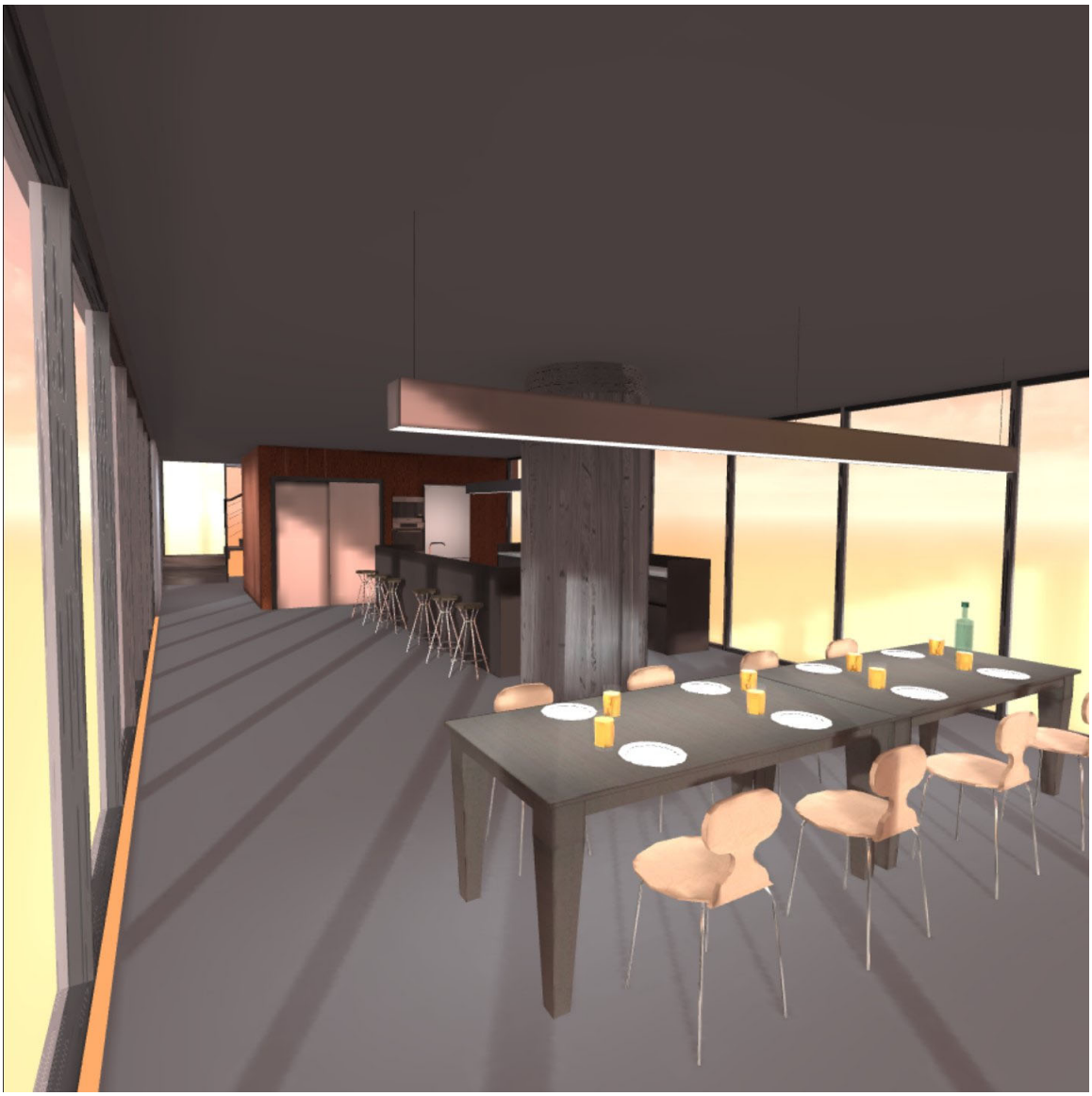

**Figure 21 Image showing the Kitchen room scene, captured inside the Oculus Quest.**

The first criteria compared was the Average Frame rate. The optimal results should be as close to 72 Frames per Second as possible. Results are shown in Figure 25.

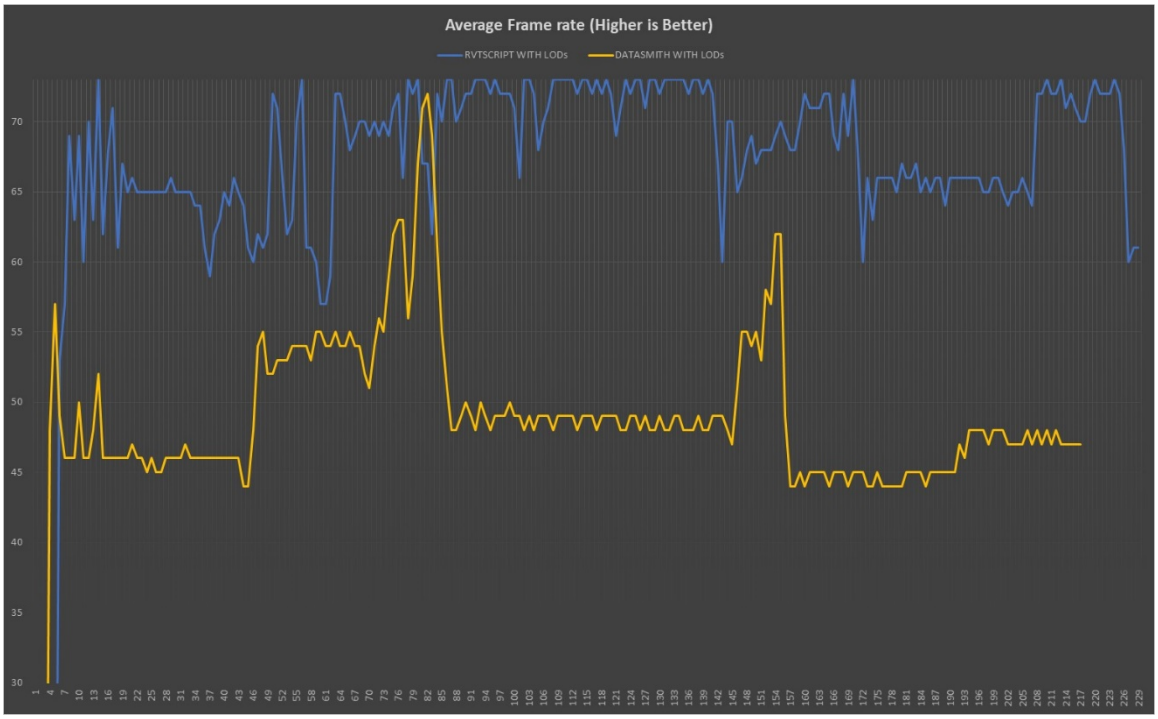


**Figure 22 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 66.67 | 69 | 72 | 72 | 57 |
| DATASMITH | 48.57 | 48 | 48 | 72 | 44 |

**Table 7 Analyzing the Average Frame Rate data.**

Data shows that the RVTSCRIPT application has a mean of 66.67 versus 48.57 from the DATASMITH application, this shows a performance improvement of almost 18 frames per second. The minimum value of the RVTSCRIPT application is also higher by a considerable 12 frames per second.

The second criteria compared is the CPU utilization within the VR device. Data can be visualized in figure 26.

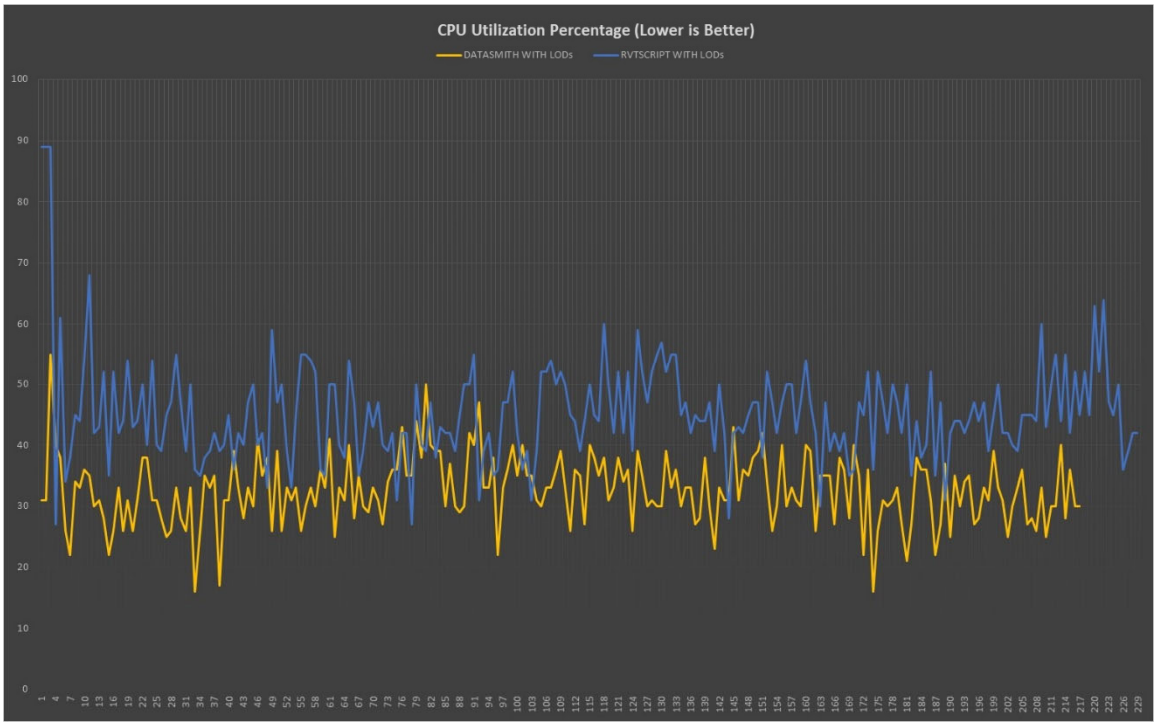


**Figure 23 Average CPU Utilization data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 45.34 | 44 | 42 | 68 | 27 |
| DATASMITH | 32.5 | 33 | 33 | 55 | 16 |

**Table 8 Analyzing the CPU Utilization data.**

Data shows that the RVTSCRIPT application has a mean of 45.34 versus 32.5 from its DATASMITH counterpart. Since both means are under 50% and the maximum value reached is only 68%, it is safe to say that CPU utilization is not impacting the performance of the applications.

The third criteria compared was the GPU utilization within the VR device. Data can be visualized in figure 27.

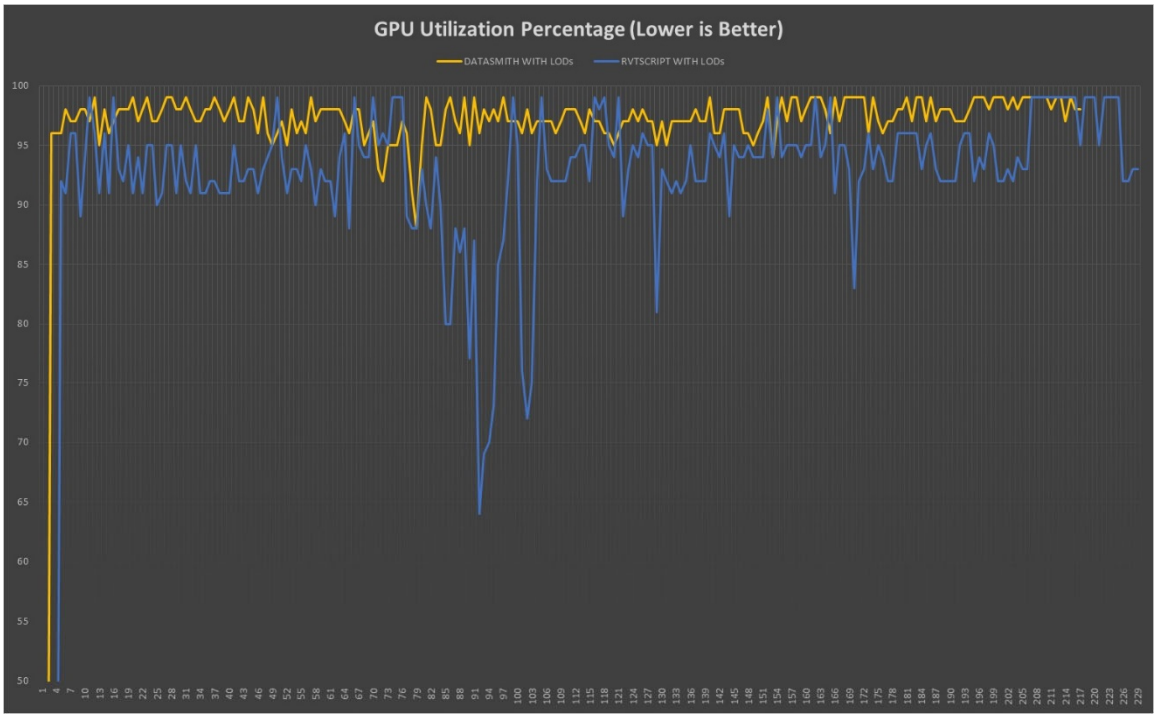


**Figure 24 Average GPU Utilization data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 91.57 | 94 | 95 | 99 | 64 |
| DATASMITH | 96.57 | 97 | 97 | 99 | 88 |

**Table 9 Analyzing the GPU Utilization data.**

Data shows that the RVTSCRIPT application has a mean of 91.57 versus 96.57 on the DATASMITH application. While both applications are GPU bound, and will always demand high usage of the GPU, a lower GPU usage creates less heat from the device and less battery drainage, so it is considered beneficial. Nevertheless, this shouldn't affect performance.

The fourth criteria compared was the available memory in the VR device while the application is running. This metric is measured in megabytes. Data can be visualized in figure 28.

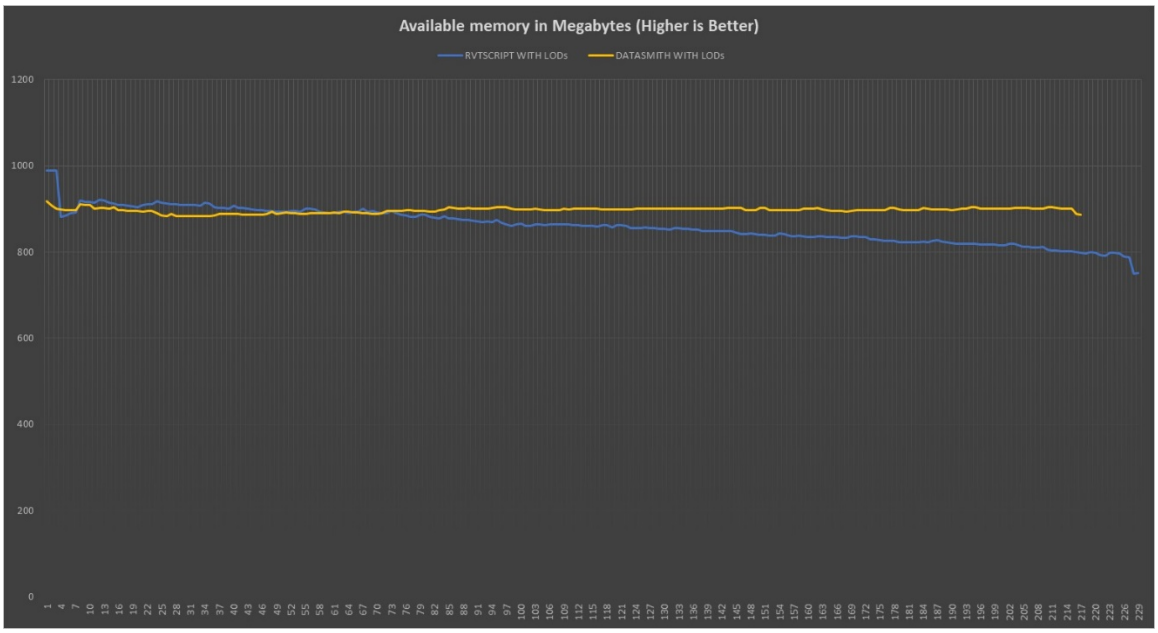


**Figure 25 Available memory data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 861.08 | 861 | 894 | 990 | 749 |
| DATASMITH | 897.24 | 899 | 901 | 918 | 883 |

**Table 10 Analyzing the Available memory data.**

This metric helps us determine how much memory is the application using, data shows that, the same as the last case study, the RVTSCRIPT application uses more memory than its DATASMITH counterpart.

Finally, the applications with an LOD system were compared to identical applications without an LOD system. These extra applications are labeled as LOD0 (highest quality and triangle count), and LOD3 (lowest quality and triangle count). Based on their importance for the performance of the application, only the Average Frames per Second and Available memory metrics were used.

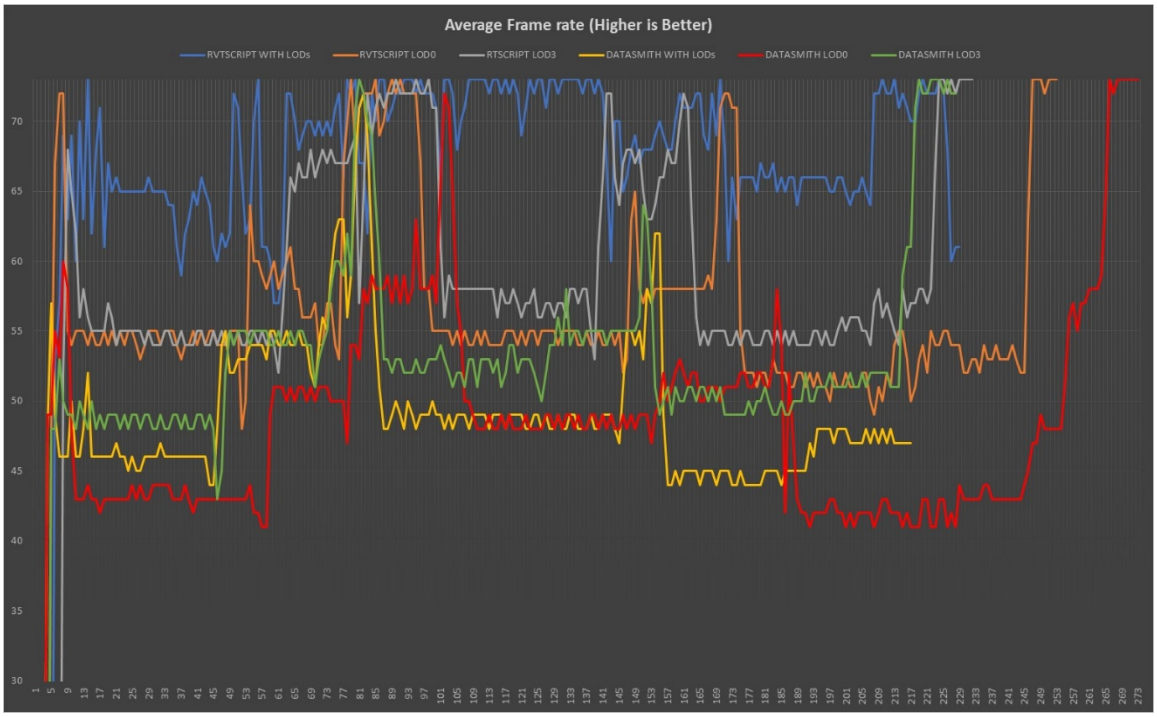


**Figure 26 Average frames per second data from the RVTSCRIPT and DATASMITH applications**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 66.67 | 69 | 72 | 72 | 53 |
| DATASMITH WITH LODs | 48.57 | 48 | 48 | 72 | 44 |
| RVTSCRIPT LOD0 | 56.02 | 55 | 55 | 72 | 48 |
| DATASMITH LOD0 | 48.43 | 48 | 43 | 72 | 41 |
| RVTSCRIPT LOD3 | 68.41 | 72 | 72 | 72 | 52 |
| DATASMITH LOD3 | 52.58 | 52 | 49 | 72 | 43 |

Data shows how the RVTSCRIPT WITH LODs application reaches the highest mean average. Followed by the both LOD3 versions with lower quality. The RVTSCRIPT LOD3 version reached the desired 72 FPS as expected but the DATASMITH LOD3 version struggled and only reached a mode of 49 FPS.

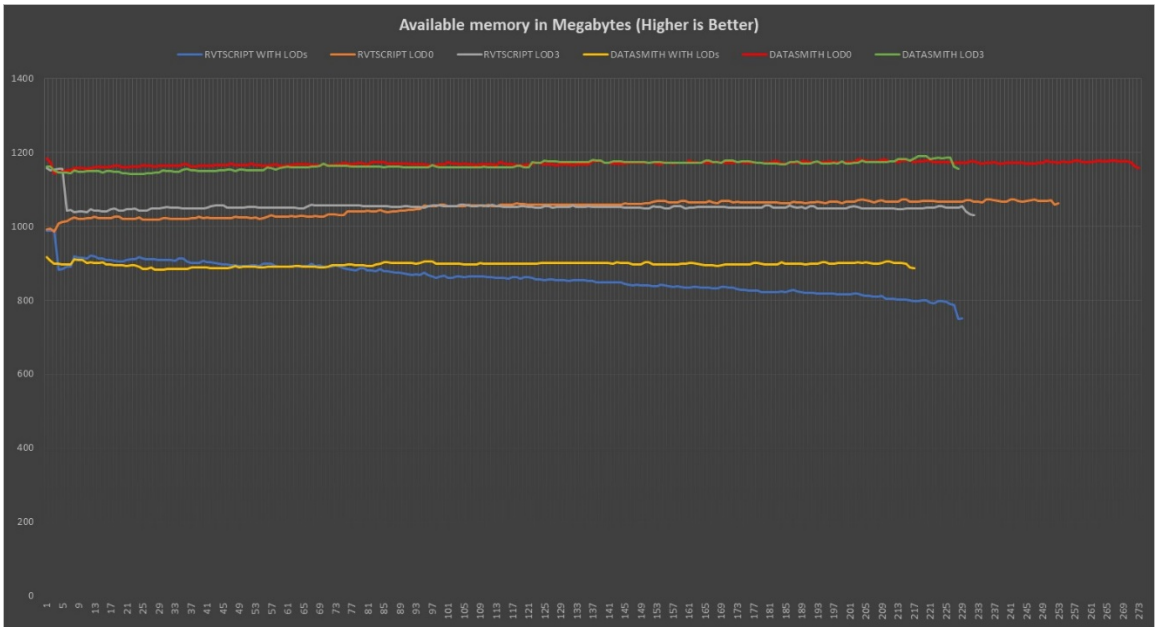


**Figure 27 Available memory data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 861.08 | 861 | 894 | 990 | 749 |
| DATASMITH WITH LODs | 897.24 | 899 | 901 | 918 | 833 |
| RVTSCRIPT LOD0 | 1050.06 | 1059 | 1059 | 1074 | 986 |
| DATASMITH LOD0 | 1170.34 | 1171 | 1173 | 1184 | 1144 |
| RVTSCRIPT LOD3 | 1053.33 | 1051 | 1051 | 1158 | 1031 |
| DATASMITH LOD3 | 1165.18 | 1164 | 1161 | 1190 | 1142 |

**Table 11 Analyzing available memory data from six different applications.**

Data shows that both DATASMITH LOD0 and LOD3 applications use the least amount of memory. Followed by their RVTSCRIPT counterparts. Then both applications with LOD systems start similar, but the RVTSCRIPT application ends up using the most memory of all, following the same trend of the previous case study.

### 5.5.3 Anglican Christ Church

This case study represents a more complex Revit project of a two-story building with multiple decorations on the façade and a complex wood interior. The walkthrough takes a loop through the outside and then into basement floor followed by the main floor.

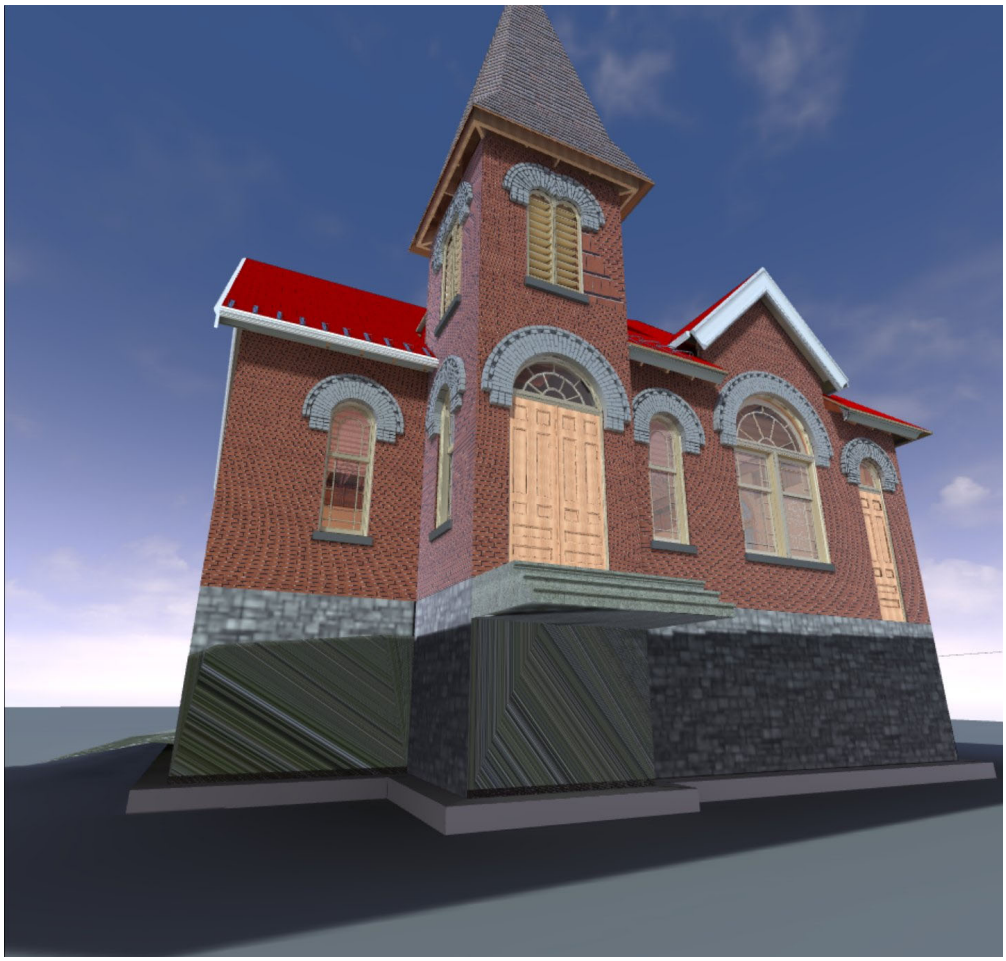

**Figure 28 Anglican Christ Church BIM, image captured inside the Oculus Quest.**

The first criteria compared was the Average Frame rate. The optimal results should be as close to 72 Frames per Second as possible. Data can be visualized in Figure 29.

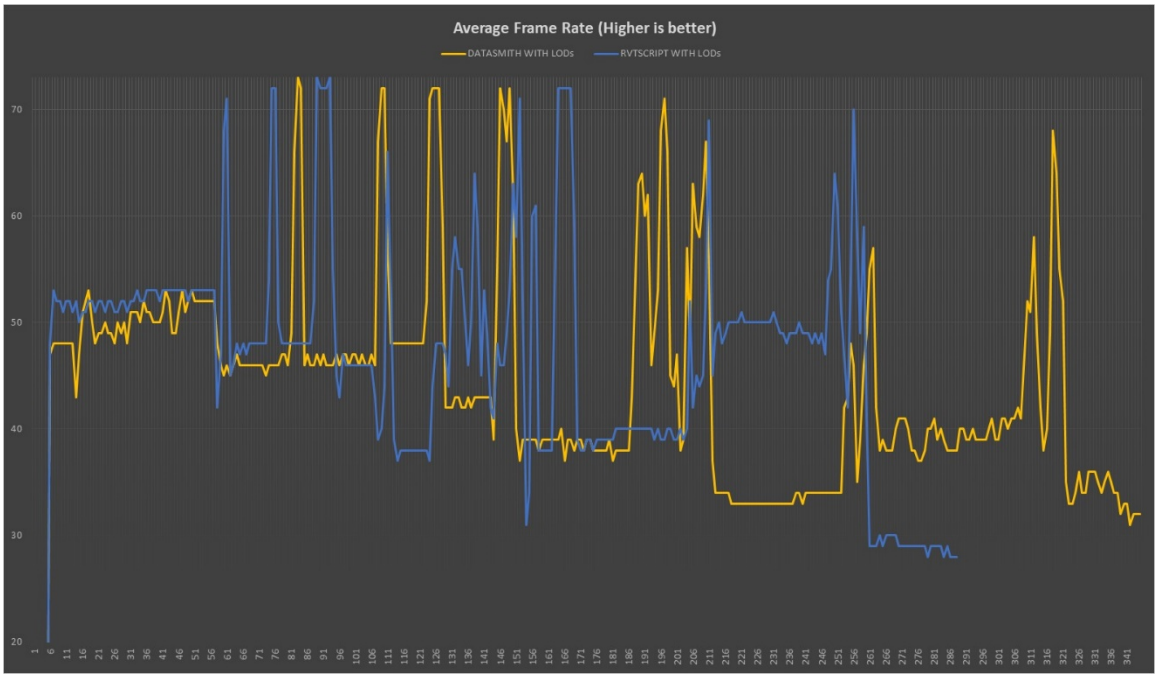


**Figure 29 Average frames per second chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 47.35 | 48 | 48 | 72 | 28 |
| DATASMITH | 43.95 | 43 | 46 | 72 | 31 |

Data shows a mean of 47.35 for the RVTSCRIPT application and 43.95 for its DATASMITH counterpart. This tells us that the RVTSCRIPT application performs 3.4 frames per second better than its DATASMITH counterpart.

The second criteria compared is the CPU utilization within the VR device. Data can be visualized in figure 30.

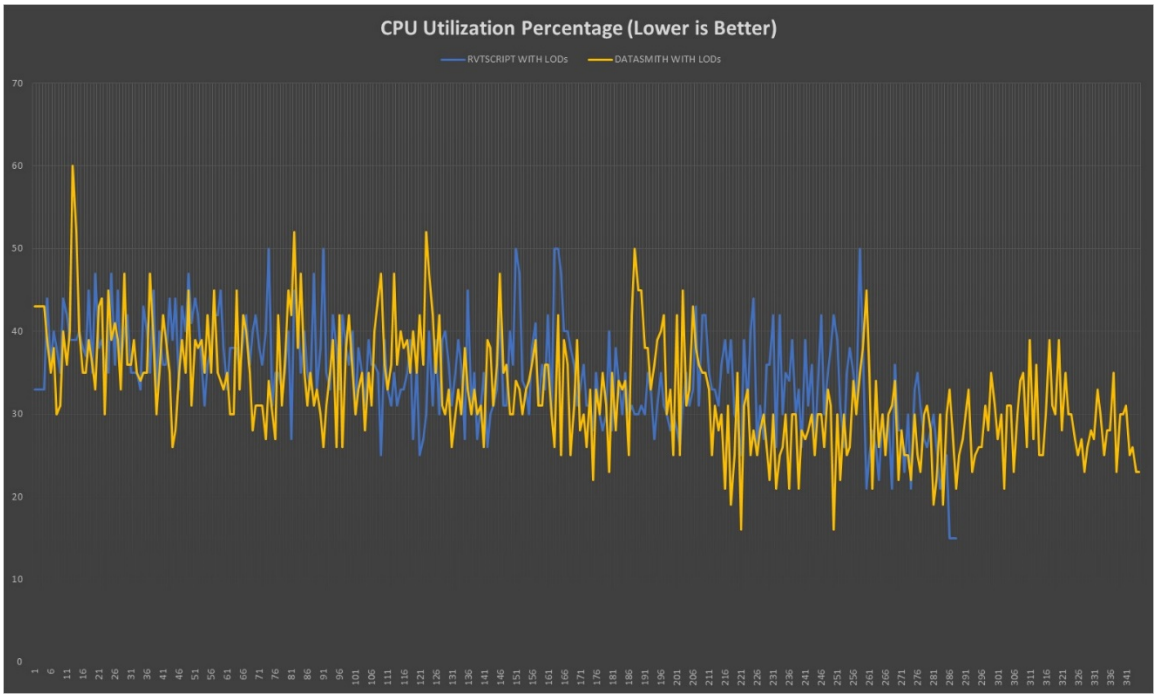


**Figure 30 CPU Utilization percentage chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 34.85 | 35 | 35 | 50 | 15 |
| DATASMITH | 32.79 | 33 | 30 | 60 | 16 |

**Table 12 Analyzing the CPU Utilization data.**

Data shows that the RVTSCRIPT application utilizes an approximate 3% more CPU power than its DATASMITH counterpart. Nevertheless, both applications use less than 50% of the CPU processing power, so this metric shouldn't affect their performance.

The third criteria compared was the GPU utilization within the VR device. Data can be visualized in figure 31.

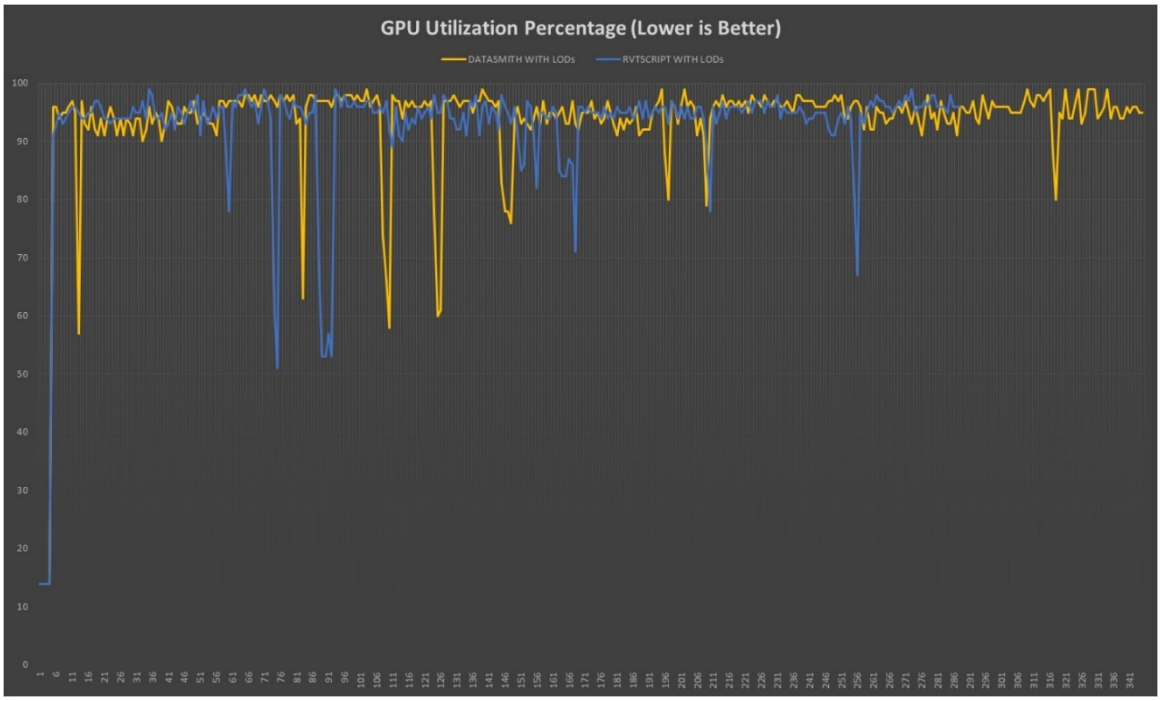


**Figure 31 CPU Utilization percentage chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 92.49 | 95 | 96 | 99 | 14 |
| DATASMITH | 93.51 | 96 | 97 | 99 | 14 |

**Table 13 Analyzing the GPU Utilization data.**

Data shows a difference of only one percent (1%) on the GPU utilization of both RVTSCRIPT and DATASMITH applications. Their minimum and maximum values are also the same.

The fourth criteria compared was the available memory in the VR device while the application is running. This metric is measured in megabytes. Data can be visualized in figure 32.

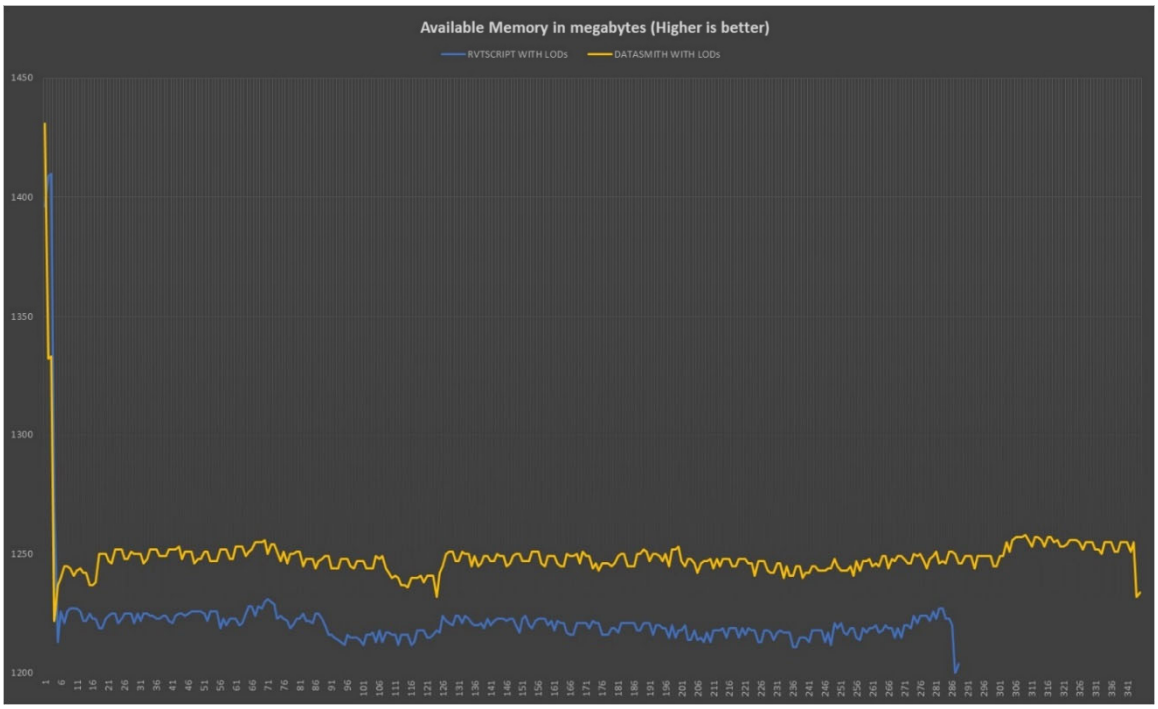


**Figure 32 Available memory chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 1222.02 | 1220 | 1221 | 1410 | 1200 |
| DATASMITH | 1248.77 | 1248 | 1249 | 1431 | 1222 |

**Table 14 Analyzing available memory data.**

Data shows that the RVTSCRIPT application uses more memory than its DATASMITH counterpart and maintains its consistency with the previous case studies. The extra memory used is not enough to affect performance.

Finally, the applications with an LOD system were compared to identical applications without an LOD system. These extra applications are labeled as LOD0 (highest quality and triangle count), and LOD3 (lowest quality and triangle count). Based on their importance for the performance of the application, only the Average Frames per Second and Available memory metrics were used.

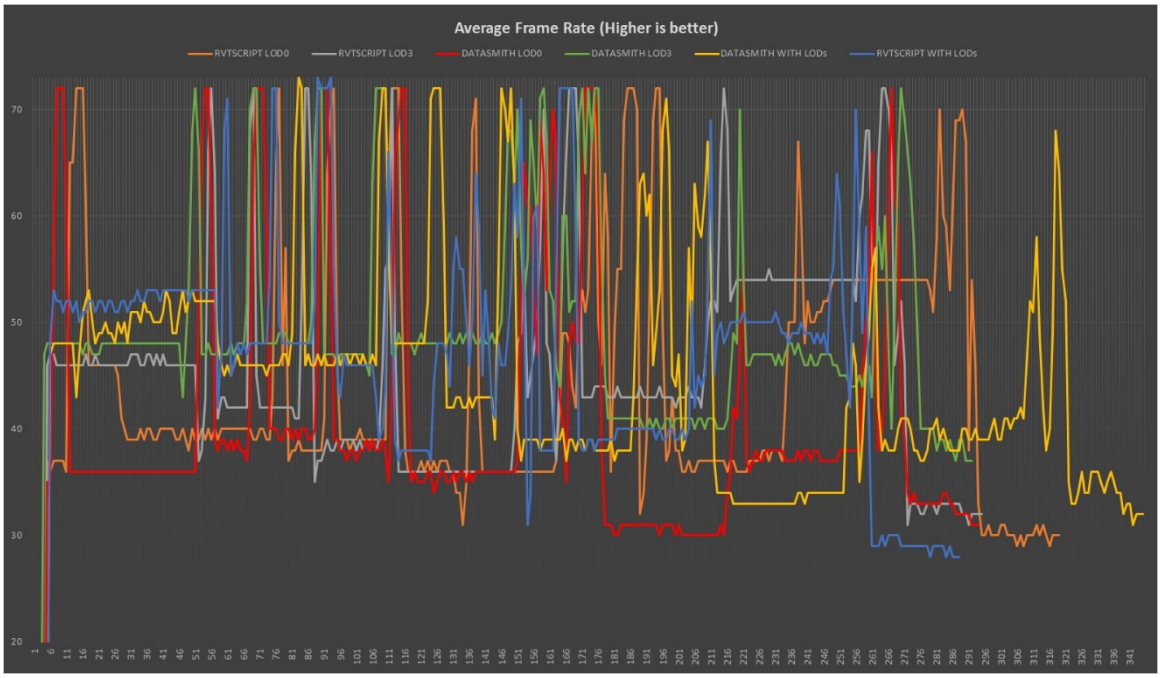


**Figure 33 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 47.35 | 48 | 48 | 72 | 28 |
| DATASMITH WITH LODs | 43.95 | 43 | 46 | 72 | 31 |
| RVTSCRIPT LOD0 | 43.74 | 39 | 36 | 72 | 29 |
| DATASMITH LOD0 | 40.41 | 37 | 36 | 72 | 30 |
| RVTSCRIPT LOD3 | 52.92 | 49 | 49 | 72 | 31 |
| DATASMITH LOD3 | 48.85 | 48 | 48 | 72 | 37 |

Data shows the RVTSCRIPT LOD3 version having the best performance overall, followed closely by the DATASMITH LOD3 and then the RVTSCRIPT with LODs. The worst performance goes to the DATASMITH LOD0 application.

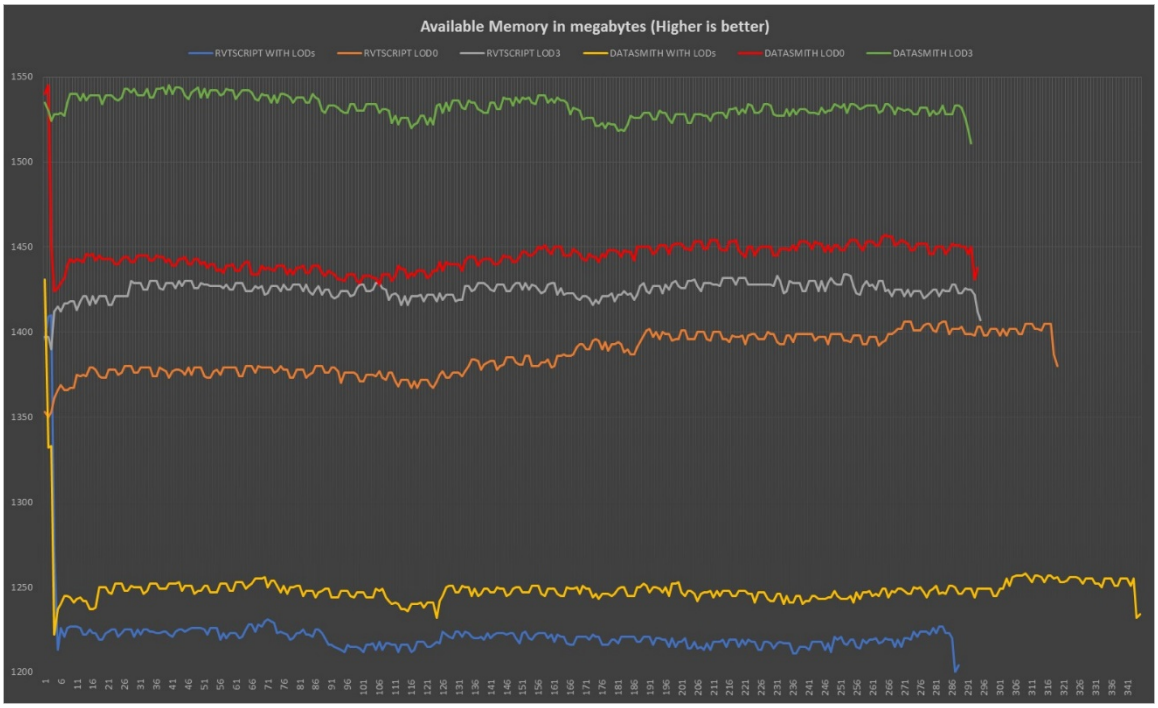


**Figure 34 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 1222.02 | 1220 | 1221 | 1410 | 1200 |
| DATASMITH WITH LODs | 1248.78 | 1248 | 1249 | 1431 | 1222 |
| RVTSCRIPT LOD0 | 1386.65 | 1385 | 1399 | 1406 | 1350 |
| DATASMITH LOD0 | 1444.47 | 1444 | 1450 | 1545 | 1424 |
| RVTSCRIPT LOD3 | 1424.57 | 1425 | 1428 | 1434 | 1390 |
| DATASMITH LOD3 | 1532.46 | 1532 | 1529 | 1545 | 1511 |

**Table 15 Analyzing average memory data from six different applications.**

Data shows that both LOD systems use more memory than the other applications. With the RVTSCRIPT WITH LODs being the one that uses the most memory of all. Both Datasmith LOD3 applications use less memory than their RVTSCRIPT counterparts. With the DATASMITH LOD3 application using the least memory of all.

### 5.5.4 Peace Tower

This case study was designed to stress test the performance of pipeline and of the Oculus Quest VR device by using a high quality, fully detailed BIM of the Peace Tower, and iconic part of Canada's Parliamentary Precinct.

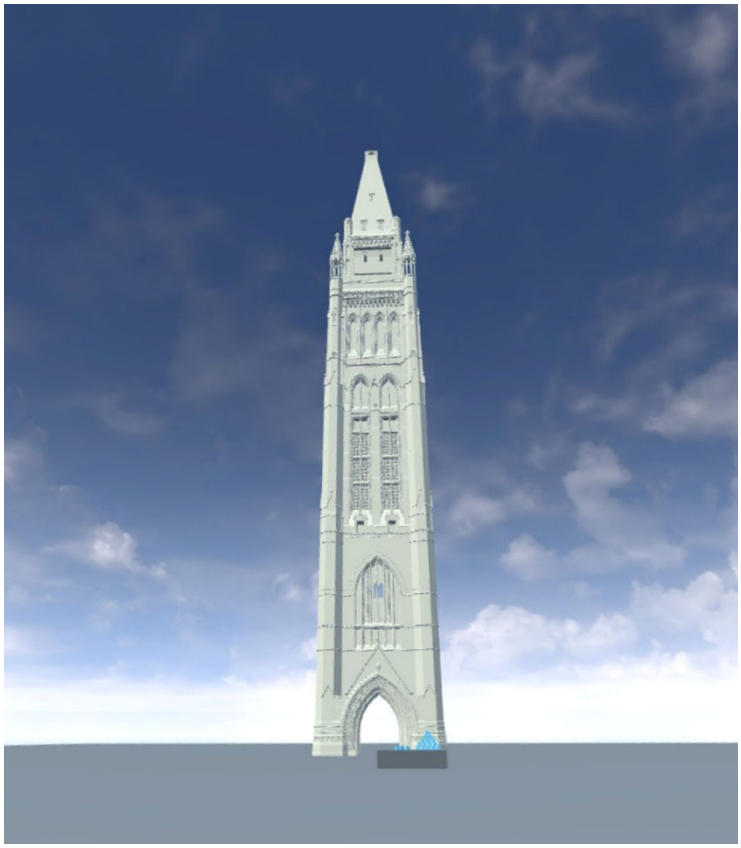

**Figure 35 Image of the Peace Tower captured inside the Oculus Quest**

The first criteria compared was the Average Frame rate. The optimal results should be as close to 72 Frames per Second as possible. Data can be visualized in Figure 36.

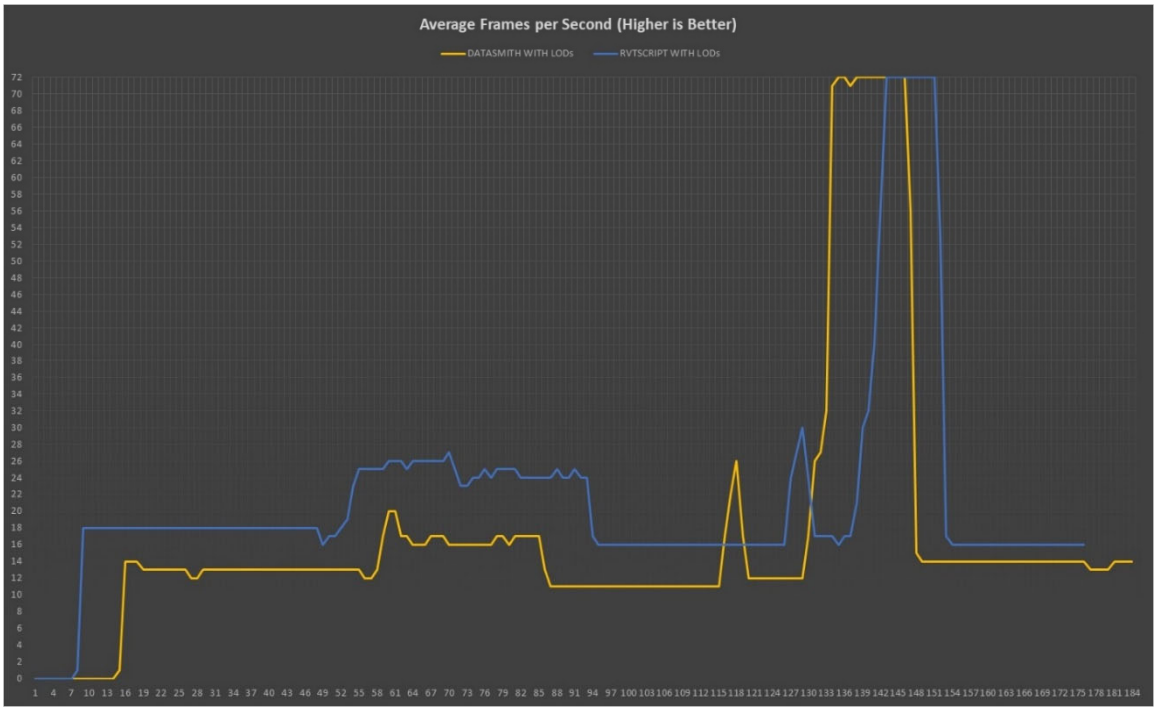


**Figure 36 Average frames per second chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 21.69 | 18 | 16 | 72 | 16 |
| DATASMITH | 17.14 | 13 | 13 | 72 | 11 |

**Table 16 visualizing the average frames per second data.**

Data clearly shows that this complex case study limits the performance of the device. While both applications still reach the desired 72 FPS at times, their mean averages are way lower. The RVTSCRIPT application has a mean of only 21.69 while its DATASMITH counterpart only manages 17.14.

The second criteria compared is the CPU utilization within the VR device. Data can be visualized in figure 37.

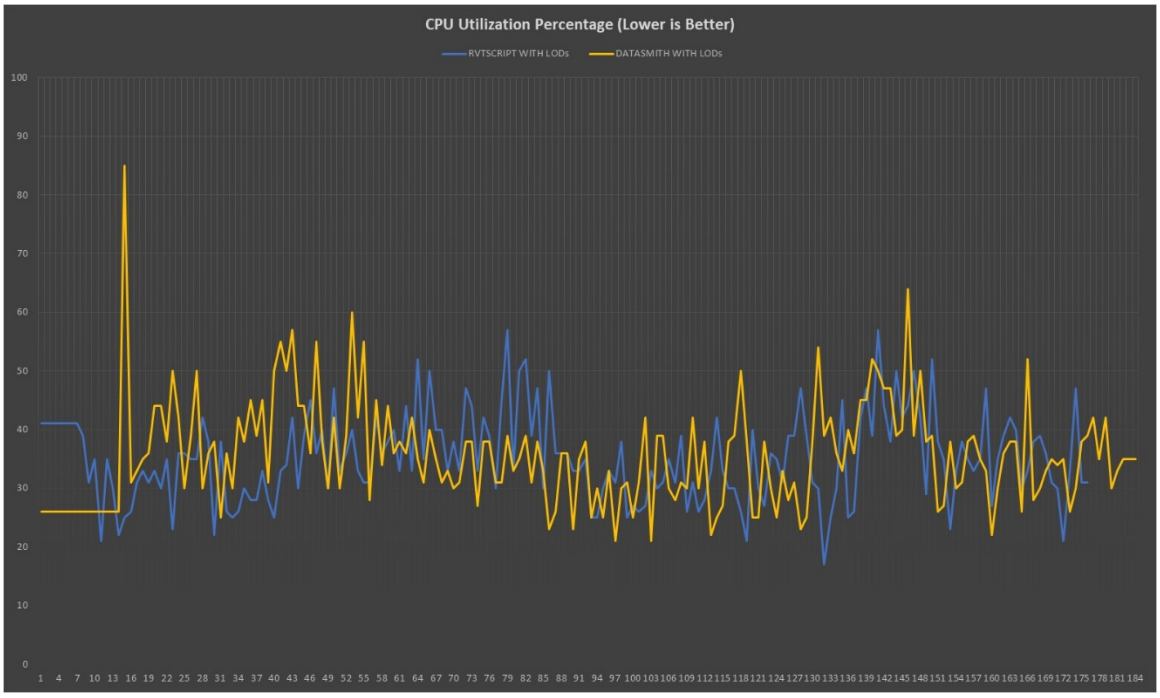


**Figure 37 CPU Utilization Chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 35.18 | 35 | 33 | 57 | 17 |
| DATASMITH | 35.70 | 35 | 38 | 85 | 21 |

**Table 17 visualize the CPU utilization data.**

Data shows an almost identical mean between both applications. It is also shown that the DATASMITH application has a bigger impact on the CPU. The recurring trend continues showing that the CPU utilization is low enough that is not affecting performance in any way.

The third criteria compared was the GPU utilization within the VR device. Data can be visualized in figure 38.

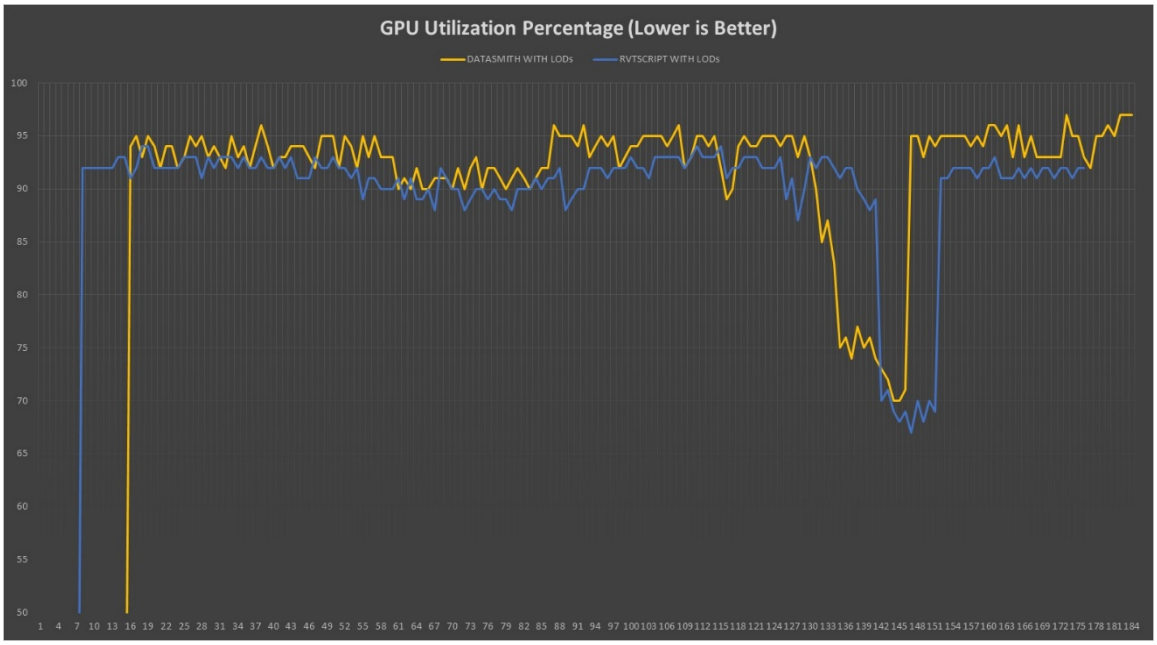


**Figure 38 GPU utilization chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 87.13 | 92 | 92 | 94 | 14 |
| DATASMITH | 85.70 | 93 | 95 | 97 | 14 |

**Table 18 visualize the GPU utilization data.**

Data shows that the RVTSCRIPT application has a higher mean of around 2% compared to its DATASMITH counterpart. Despite this the DATASMITH application has higher values on all the other categories. It could be said that both applications are evenly matched in this metric.

The fourth criteria compared was the available memory in the VR device while the application is running. This metric is measured in megabytes. Data can be visualized in figure 39.

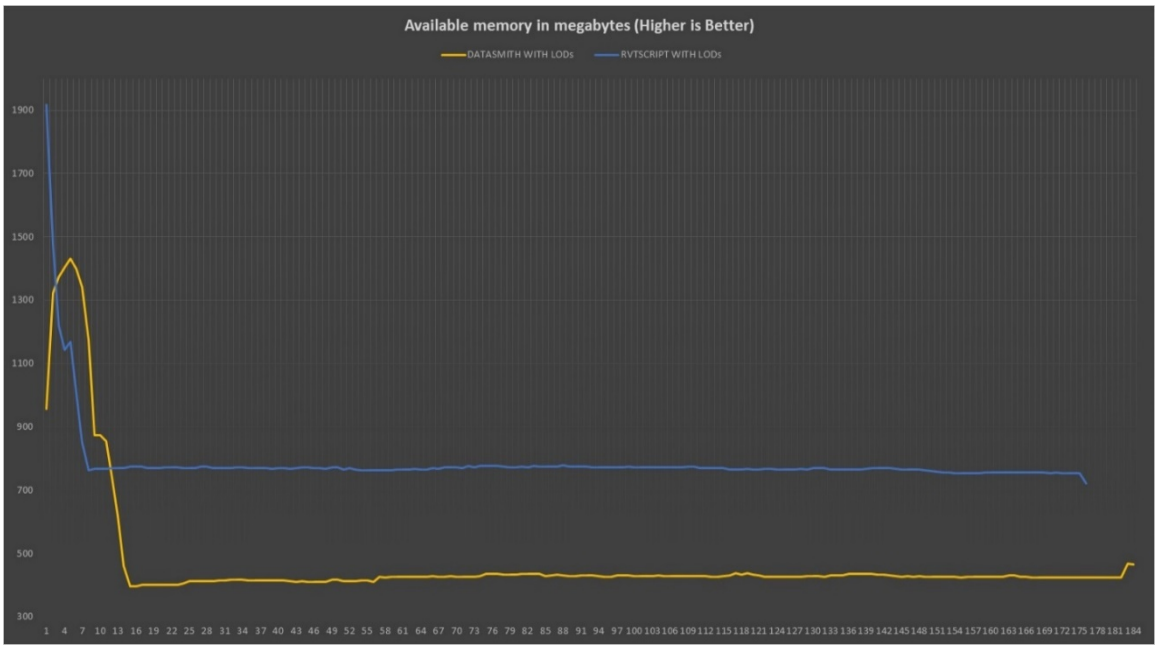


**Figure 39 Available memory chart.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT | 786.63 | 769 | 769 | 1918 | 722 |
| DATASMITH | 472.88 | 427 | 426 | 1432 | 397 |

**Table 19 Analyzing available memory data.**

Data shows that the DATASMITH application has high memory usage, since it is using approximately 300 megabytes more than the RVTSCRIPT application. Both applications are using a considerable amount of extra memory compared to the other case studies.

Finally, the applications with an LOD system were compared to identical applications without an LOD system. These extra applications are labeled as LOD0 (highest quality and triangle count), and LOD3 (lowest quality and triangle count). Based on their importance for the performance of the application, only the Average Frames per Second and Available memory metrics were used.

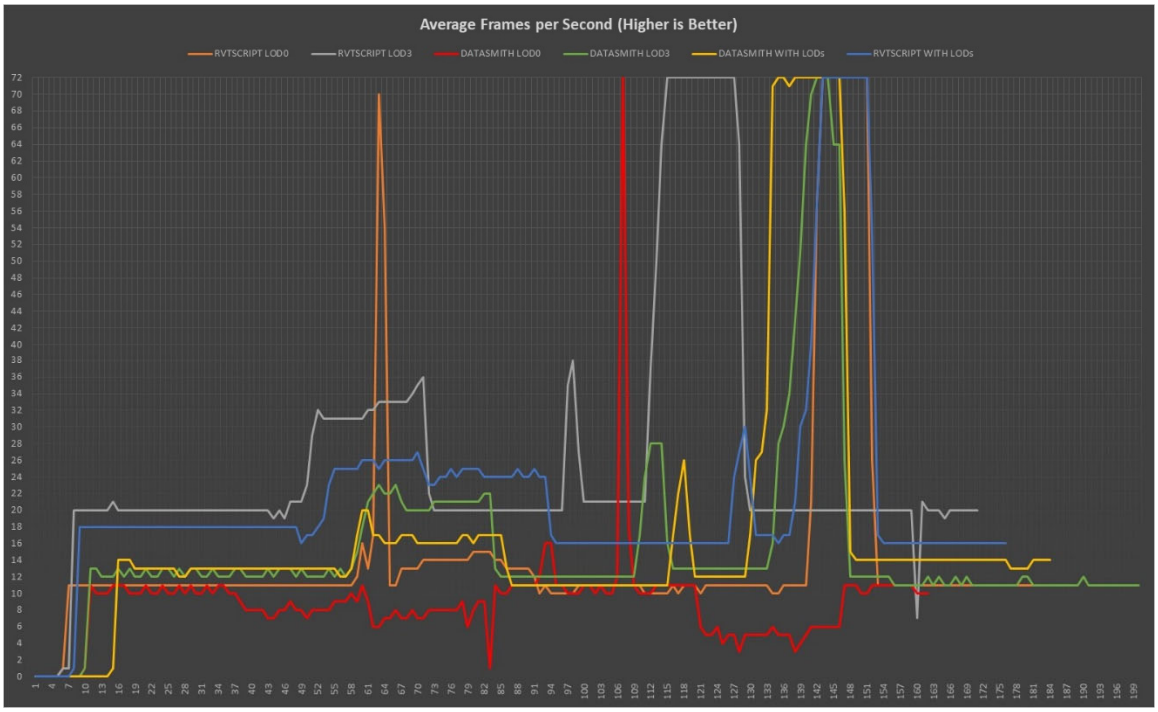


**Figure 40 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 21.69 | 18 | 16 | 72 | 16 |
| DATASMITH WITH LODs | 17.14 | 13 | 13 | 72 | 11 |
| RVTSCRIPT LOD0 | 14.83 | 11 | 11 | 72 | 10 |
| DATASMITH LOD0 | 8.79 | 10 | 11 | 72 | 1 |
| RVTSCRIPT LOD3 | 25.73 | 20 | 20 | 72 | 7 |
| DATASMITH LOD3 | 15.55 | 12 | 12 | 72 | 11 |

**Table 20 Analyzing average frames per second data from six different applications.**

Data shows the continuing trend between case studies. Simplified LOD3 versions achieve greater performance. In this case though, the RVTSCRIPT LOD3 version averages 10 FPS higher than its DATASMITH LOD3 counterpart. LOD0 versions got the lowest numbers. Our RVTSCRIPT WITH LODs version still outperforms all the DATASMITH versions.

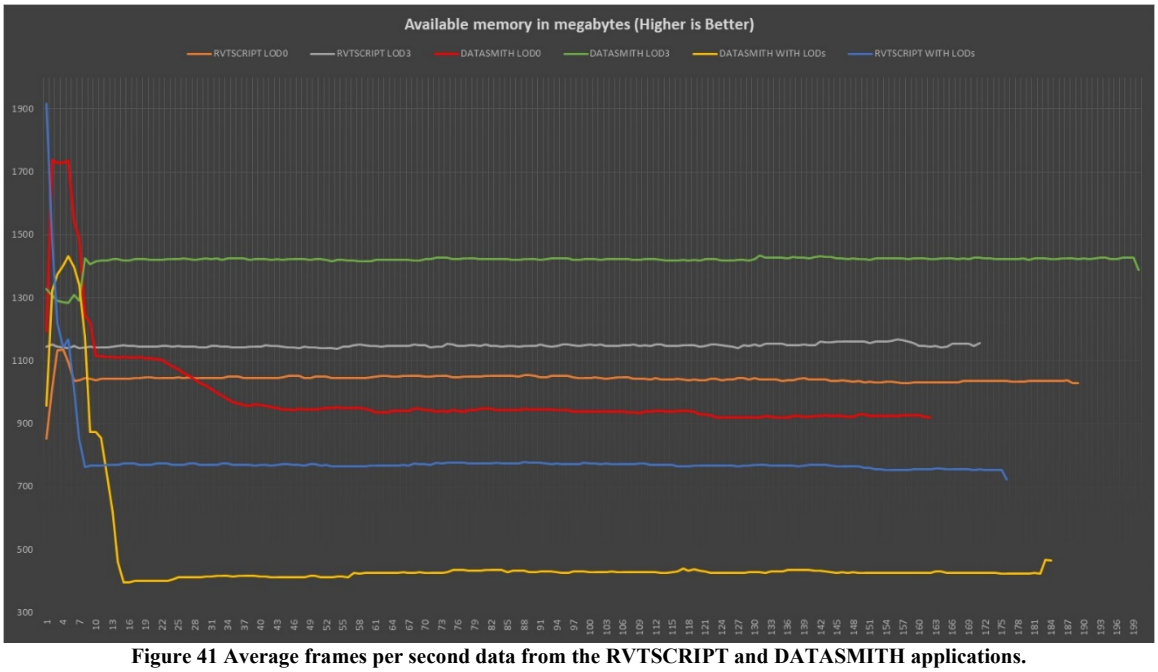


**Figure 41 Average frames per second data from the RVTSCRIPT and DATASMITH applications.**

| Applications | Mean Average | Median Average | Mode Average | Max Value | Min Value |
|---|---|---|---|---|---|
| RVTSCRIPT WITH LODs | 786.63 | 769 | 769 | 1918 | 722 |
| DATASMITH WITH LODs | 472.88 | 427 | 426 | 1432 | 397 |
| RVTSCRIPT LOD0 | 1042.79 | 1044 | 1045 | 1136 | 853 |
| DATASMITH LOD0 | 990.56 | 942 | 938 | 1739 | 919 |
| RVTSCRIPT LOD3 | 1148.47 | 1147 | 1144 | 1168 | 1139 |
| DATASMITH LOD3 | 1418.62 | 1423 | 1424 | 1435 | 1285 |

**Table 21 Analyzing memory usage data from six different applications.**

Data shows that the applications with LOD systems use more memory following the trend from the previous case studies. The DATASMITH LOD3 version uses the least amount of memory followed by the RVTSCRIPT LOD3. There is still a fair amount of memory left, even in the DATASMITH WITH LODs application. This means that memory usage is still not impacting performance.

# Chapter 6: Discussion

## 6.1 Analysis breakdown

Looking at the results, we can see that the Revit game applications developed using our proposed framework performed better than the applications developed using the Unreal Datasmith toolset. Both using a Level of Detail (LOD) system:

| Mode average frames per second by case study (max 72 FPS): | Proposed Framework | Unreal Datasmith |
|---|---|---|
| Small House Concept | 72 | 67 |
| Kitchen Room | 72 | 48 |
| Anglican Christ Church | 48 | 46 |
| Peace Tower | 16 | 13 |

**Table 22 average frames per second data compared.**

Also, without using a Level of Detail system:

| Mode average frames per second by case study (max 72 FPS): | Proposed Framework LOD0 | Proposed Framework LOD3 | Unreal Datasmith LOD0 | Unreal Datasmith LOD3 |
|---|---|---|---|---|
| Small House Concept | 33 | 72 | 33 | 46 |
| Kitchen Room | 55 | 72 | 43 | 49 |
| Anglican Christ Church | 36 | 49 | 36 | 48 |
| Peace Tower | 11 | 20 | 11 | 12 |

**Table 23 average frames per second data compared from apps without the LOD system.**

The figure below provides a better representation of the data and shows how the proposed framework applications outperforming the Unreal Datasmith applications.

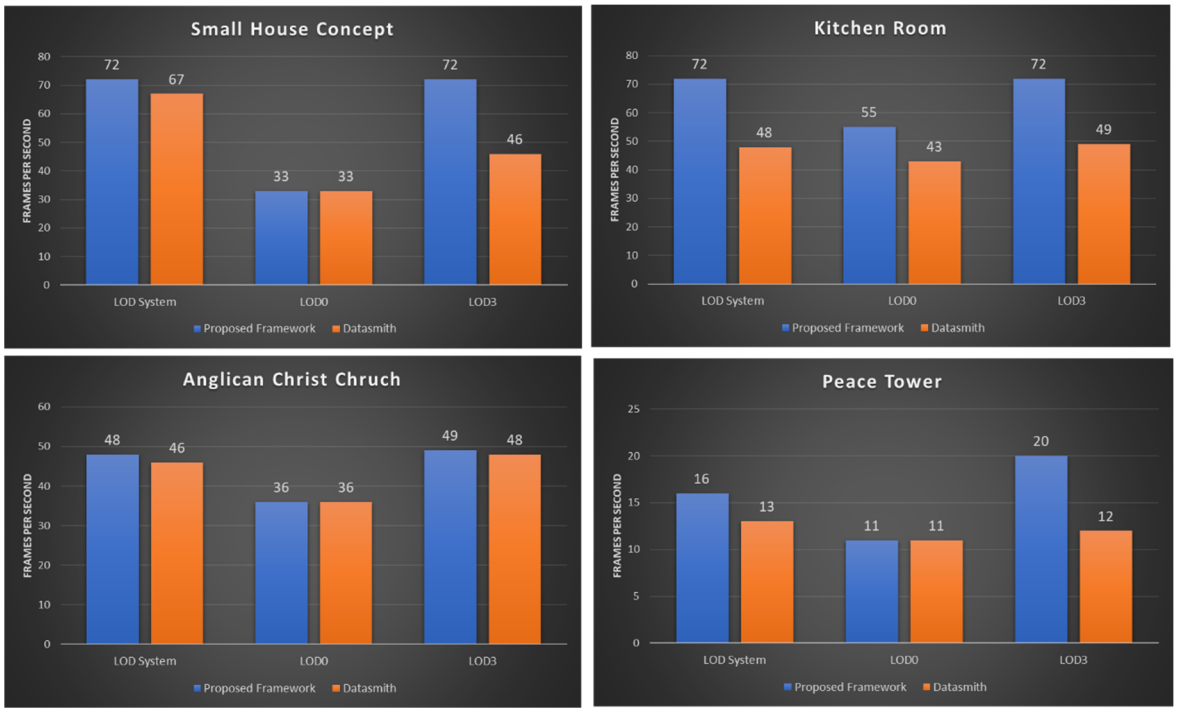


**Figure 42 Frames Per Second of all applications.**

Additionally, the data tells us that implementing LOD systems requires extra memory. The four different versions of every model in the scene (inside the LOD system), need to be saved in memory so they can be easily accessed and swapped dynamically in real-time. The Oculus Quest VR device includes a generous 4 GB of RAM; therefore, extra memory usage is possible and free of impacting performance.

## 6.2 Visual Quality Comparison.

In two out of the four case studies, applications developed with the Unreal Datasmith toolset are missing some key materials and textures, they were automatically replaced by a generic texture not found inside the Revit scene. This can be seen in figure 42.

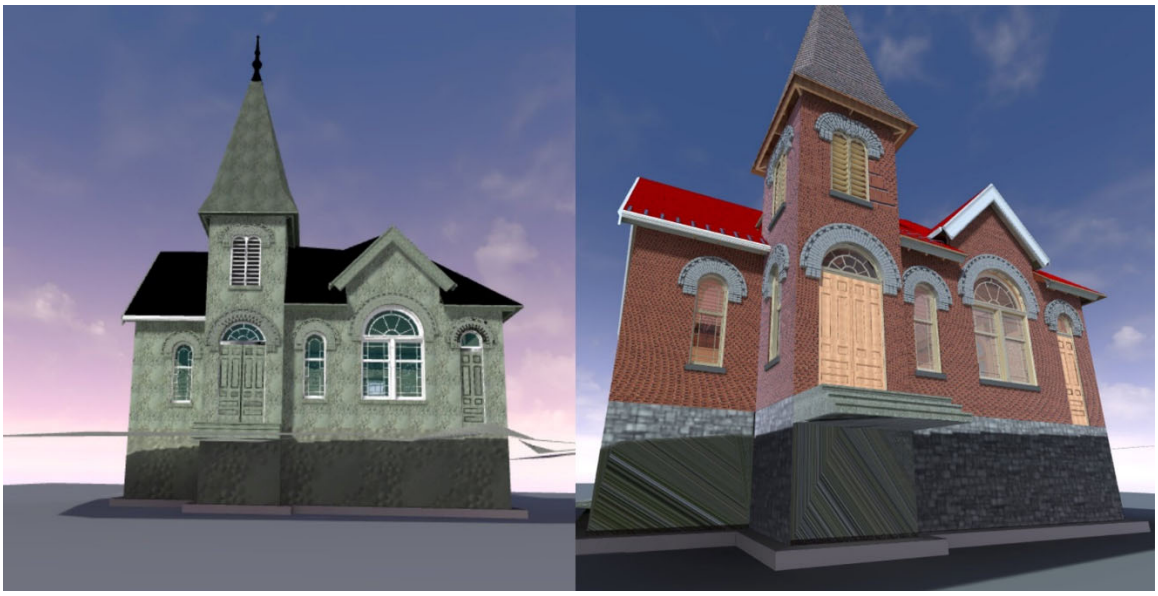

**Figure 43 Datasmith app (Left) missing materials available in the Revit scene and in the proposed framework app (Right).**

In all the case studies, applications developed with the Unreal Datasmith toolset have small but visible mesh errors in their heavily reduced LOD3 versions. This can be seen in Figures 43, 44, 45 and 46.

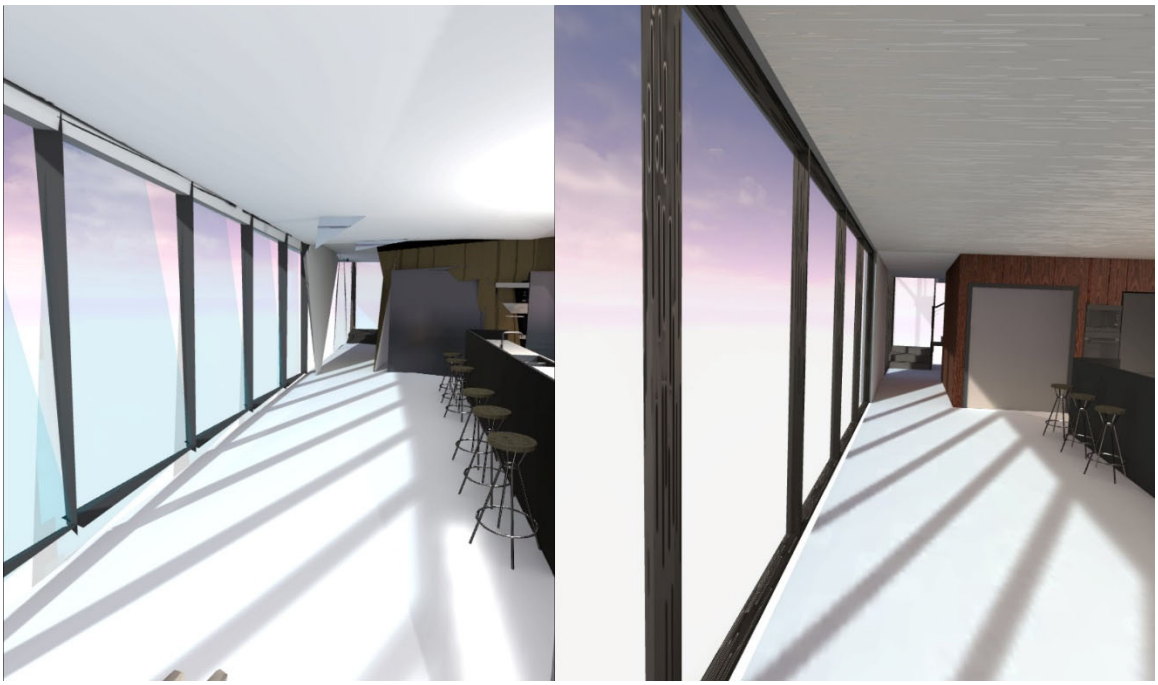

**Figure 44 Kitchen Datasmith LOD3 app (Left), same app but made with the proposed framework (Right).**

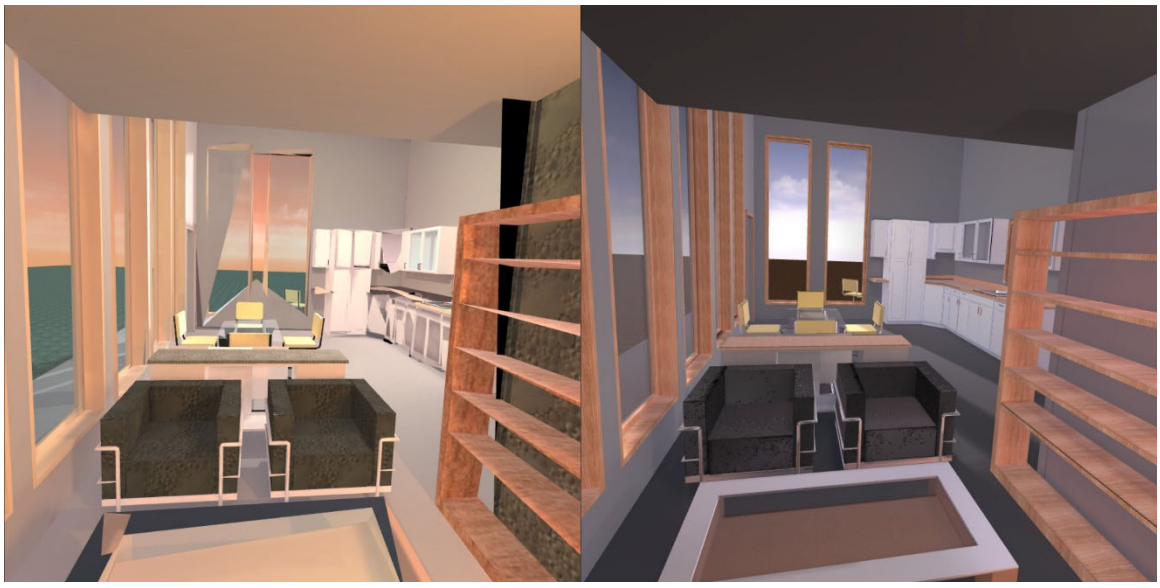

**Figure 45 Small House Datasmith LOD3 app (Left), same app but made with the proposed framework (Right).**

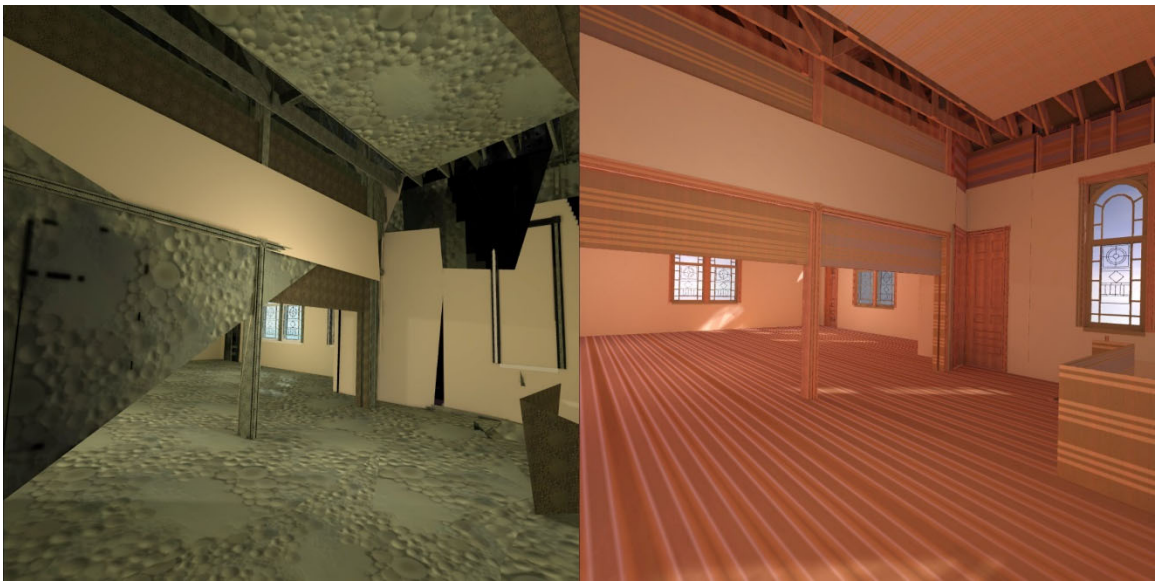

**Figure 46 Church interior datasmith LOD3 app (Left), same app but made with the proposed framework (Right).**

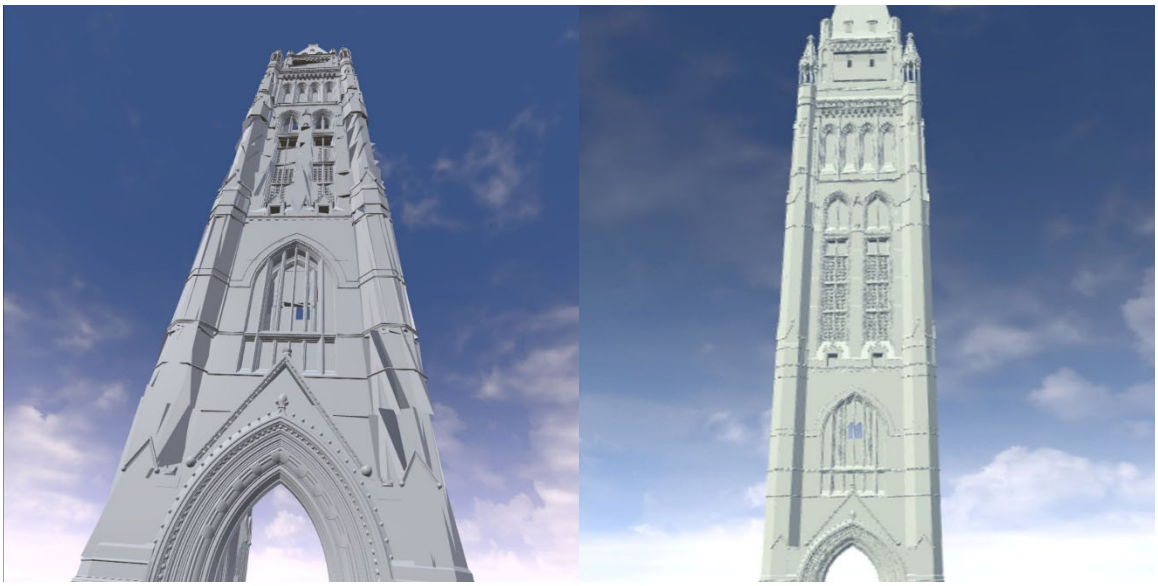

**Figure 47 Peace Tower datasmith LOD3 app (Left), same app but made with the proposed framework (Right).**

## 6.3 Limitations.

Further analysis showed that two of the four case studies couldn't perform at the recommended mean average of 72 frames per second (Quest VRC, 2019). This metric is inversely correlated with the GPU utilization percentage that was measured in the upper nineties for every application. This tells us that application performance is limited by the GPU's processing capabilities. Further manual optimization techniques could be applied inside the Unreal Engine to reach this desired frame rate, but they are considered external to the automated framework presented.

A limitation found in the proposed framework is the conversion of certain materials. The Autodesk Noise bitmap found in two of the four case studies could not be automatically converted to a map that a game engine could interpret. A warning for this limitation has been added as a label inside the plugin UI (see figure 13 in Chapter 4.8). Any Autodesk Noise bitmaps need to be deleted to avoid error messages from the code.

Additionally, transparent materials like glass are not properly read inside the Unreal Engine and they are imported as opaque materials. The user must manually change these materials so they can be transparent again. Also, imported lights have a high intensity value that needs to be lowered manually so that it properly reflects the value inside the Revit project.

Further research would involve the development of additional automation tools inside the Unreal Engine leveraging the newly integrated experimental Python API (Unreal Python API, 2019). A system called Hierarchical Level of Detail (HLOD) recently implemented inside the Unreal Engine on its 4.22 version (Unreal HLOD, 2019) could be implemented to create more optimized LOD3 versions, or even LOD4 versions reducing the performance impact of elements further in the background.

Finally, additional qualitative data could be gathered in the form of a user study, this could introduce new metrics related to application comfort, ease of use of the metadata display system and overall perceived visual quality of the scene.

# Chapter 7: Conclusion

This research aimed at answering two main questions regarding the visualization of Autodesk Revit data inside virtual reality applications:

- Can a more optimized and streamlined framework be developed for accurate data translation?
- How would this new framework perform against the current industry standards?

We explored these questions by identifying the current limitations found in the literature and the right tools to address them. We then presented a new optimization framework as a plugin / tool inside Autodesk 3DS Max. This tool leverages new Autodesk software integration APIs, in conjunction with Maxscript (native language of 3DS Max) to read and translate BIM data while preserving its metadata, materials and textures. The tool automatically performs multiple model optimization procedures (presented in detail in section 4.1), with the most important being the implementation of a Level of Detail (LOD) system for every model in the scene.

This system holds four different versions of the same model ranging from the highest quality model (known as LOD0) to the more simplified and low-quality model (known as LOD3). Then it dynamically swaps between them based on their size on the screen, so that models at a distance are not displayed at full quality.

To answer the second question proposed, we presented a comparative analysis of quantitative data gathered from multiple virtual reality applications deployed and running inside an Oculus Quest mobile VR device. Four different BIM data case studies were used to develop these applications. The first set of applications were developed using our proposed framework, while the second set were developed using the Unreal Datasmith toolset by Epic Games.

The performance implications of implementing a Level of Detail system were also tested and compared between the two sets of applications. LOD0 and LOD3 versions of every case study were developed and compared with their counterparts with LOD systems in place.

Results show that the applications developed using our proposed framework had fewer visible mesh errors and performed better than the applications developed using the Unreal Datasmith toolset, both in terms of higher Average frames per second and lower GPU usage. This was true for both sets of applications, the ones using a Level of Detail (LOD) system and the ones that don't. Additionally, results show that while the implementation of LOD systems requires extra memory, its benefits regarding performance are substantial.

Future work needs to tackle limitations in the framework regarding material conversion. We also consider that conducting a user study could be beneficial to introduce some additional qualitative comparison metrics.

## References


Autodesk Revit Software 2019, https://www.autodesk.ca/en/products/revit/overview

Autodesk 3DS Max Maxscript, https://knowledge.autodesk.com/support/3ds-max/getting-started/caas/CloudHelp/cloudhelp/2020/ENU/3DSMax-Getting-Started/files/GUID-FE40F021-5444-4709-BF08-DF1F1F0960C3-htm.html

Aengenvoort, K. and Krämer, M., 2018. BIM in the Operation of Buildings. In Building Information Modeling (pp. 477-491). Springer, Cham.

Brecher, C., Lange, S., Merz, M., Niehaus, F., Wenzel, C., Winterschladen, M. and Weck, M., 2006. NURBS based ultra-precision free-form machining. CIRP annals, 55(1), pp.547-550.

Biljecki, F., Ledoux, H. and Stoter, J., 2016. An improved LOD specification for 3D building models. Computers, Environment and Urban Systems, 59, pp.25-37.

Bille, R., Smith, S.P., Maund, K. and Brewer, G., 2014, December. Extending building information models into game engines. In Proceedings of the 2014 Conference on Interactive Entertainment (pp. 1-8). ACM. https://dl.acm.org/citation.cfm?id=2677764

Boeykens, S., 2011. Using 3D Design Software, BIM and Game Engines for Architectural Historical Reconstruction. CAAD Futures 2011 : Designing Together, ULg, 2011

Boeykens, S., Maekelberg, S. and De Jonge, K., 2018. (Re-) Creating the past: 10 years of digital historical reconstructions using BIM. International Journal for Digital Art History, (3). https://journals.ub.uni-heidelberg.de/index.php/dah/article/view/32544

Borrmann A., Abualdenien J., "A meta-model approach for formal specification and consistent management of multi-LOD building models," Advanced Engineering Informatics, vol. 40, pp. 135-153, 2019.

Chenaux, A., Murphy, M., Pavia, S., Fai, S., Molnar, T., Cahill, J., Lenihan, S. and Corns, A., 2019. a Review of 3d GIS for Use in Creating Virtual Historic Dublin. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 422, pp.249-254. http://adsabs.harvard.edu/abs/2019ISPAr.422..249C

Dinis, F.M., Guimarães, A.S., Carvalho, B.R. and Martins, J.P.P., 2017, April. Virtual and augmented reality game-based applications to civil engineering education. In 2017 IEEE Global Engineering Education Conference (EDUCON) (pp. 1683-1688). IEEE. https://ieeexplore.ieee.org/abstract/document/7943075

Du, J., Zou, Z., Shi, Y. and Zhao, D., 2018. Zero latency: Real-time synchronization of BIM data in virtual reality for collaborative decision-making. *Automation in Construction*, *85*, pp.51-64. https://www.sciencedirect.com/science/article/pii/S0926580517309172

Eastman, C., Teicholz, P., Sacks, R. and Liston, K., 2011. BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors. John Wiley & Sons.

Epic Games Enterprise, Unreal Studio, https://www.unrealengine.com/en-US/studio

Edwards, G., Li, H. and Wang, B., 2015. BIM based collaborative and interactive design process using computer game engine for general end-users. Visualization in engineering https://viejournal.springeropen.com/articles/10.1186/s40327-015-0018-2

Forrester Consulting (2018) Real-Time Rendering Solutions: Unlocking The Power OF Now. Thought leadership Spotlight Commissioned by Epic Games. https://cdn2.unrealengine.com/ Unreal+Engine%2Fresources%2FEpic-Games-Real-Time-Rendering-TLP_post-production_R3-2f4769b9b2adfca45af876c2f701f65ec6ef1228.pdf

Grandon, N., Peldoza, H. and Besoain, F., 2018, June. Automatizing the generation of a virtual tour of an architecture model through an information system. In 2018 IEEE Biennial

Congress of Argentina (ARGENCON) (pp. 1-7). IEEE. https://ieeexplore.ieee.org/abstract/document/8646152

Herron, Jennifer (2010). "3D Model-Based Design: Setting the Definitions Straight". MCADCafe. http://www10.mcadcafe.com/nbc/articles/2/867959/3D-Model-Based-Design-Setting-Definitions-Straight.

Heydarian, A., Carneiro, J.P., Gerber, D., Becerik-Gerber, B., Hayes, T. and Wood, W., 2015. Immersive virtual environments versus physical built environments: A benchmarking study for building design and user-built environment explorations. Automation in Construction, 54, pp.116-126. https://www.sciencedirect.com/science/article/pii/S0926580515000606

Hilfert, T., & König, M. (2015). Low-cost virtual reality environment for engineering and construction. Paper presented at the , 32 1-8. Retrieved from https://link.springer.com/article/10.1186/s40327-015-0031-5

Johansson, M., Roupé, M. and Viklund Tallgren, M., 2014. From BIM to VR-Integrating immersive visualizations in the current design process. In *Fusion-Proceedings of the 32nd eCAADe Conference-Volume 2 (eCAADe 2014)* (pp. 261-269).

Kharvari, F. and Höhl, W., 2019, September. The Role of Serious Gaming using Virtual Reality Applications for 3D Architectural Visualization. In 2019 11th International

Conference on Virtual Worlds and Games for Serious Applications (VS-Games) (pp. 1-2). IEEE.

Kang, J. and Kuncham, K., 2014. BIM CAVE for 4D immersive virtual reality. In Proceeding of the Creative Construction Conference 2014 (CCC 2014) (pp. 568-574).

Kieferle, J. and Woessner, U., 2015. BIM interactive-about combining BIM and virtual reality-a bidirectional interaction method for BIM models in different environments. http://papers.cumincad.org/cgi-bin/works/paper/ecaade2015_329

Kado, K. and Hirasawa, G., 2018, December. Two-way cooperation of architectural 3d cad and game engine. In Proceedings of the 16th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry (p. 22). ACM. https://dl.acm.org/citation.cfm?id=3284420

Liu, X., Xie, N., Tang, K. and Jia, J., 2016. Lightweighting for Web3D visualization of large-scale BIM scenes in real-time. Graphical Models, 88, pp.40-56. https://www.sciencedirect.com/science/article/pii/S1524070316300170

Merschbrock, C., Lassen, A.K. and Tollnes, T., 2014. Integrating BIM and gaming to support building operation: the case of a new hospital. https://oda-hioa.archive.knowledgearc.net/handle/10642/2442

Minyaev, I., Pouke, M., Ylipulli, J. and Ojala, T., 2018, November. Implementation of a Virtual Reality Interface for a Public Library. In Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia (pp. 513-519). ACM. https://dl.acm.org/citation.cfm?id=3289718

Meža, S., Turk, Ž. and Dolenc, M., 2014. Component based engineering of a mobile BIM-based augmented reality system. Automation in construction, 42, pp.1-12. https://www.sciencedirect.com/science/article/pii/S0926580514000363

Nandavar, Anirudh & Petzold, Frank & Nassif, Dr & Schubert, Gerhard & Ag, Bmw. (2018). INTERACTIVE VIRTUAL REALITY TOOL FOR BIM BASED ON IFC. Presented at CAADRIA 2018, Tsinghua University, Beijing, China.

Owen, Steven J., David R. White, and Timothy J. Tautges. "Facet-Based Surfaces for 3D Mesh Generation." IMR. 2002.

Oculus Developers Support Blog, https://developer.oculus.com/blog/ovr-metrics-tool-vrapi-what-do-these-metrics-mean/

Pouke, M., Virtanen, J.P., Badri, M. and Ojala, T., 2018, January. Comparison of two workflows for Web-based 3D smart home visualizations. In 2018 IEEE International Conference on Future IoT Technologies (pp. 1-8). IEEE. https://ieeexplore.ieee.org/abstract/document/8325599

Park, H., Panya, D.S., Goo, H., Kim, T. and Seo, J., (2018). BIM-based Virtual Reality and Human Behavior Simulation For Safety Design. http://papers.cumincad.org/cgi-bin/works/paper/ecaade2018_251

Quest Virtual Reality Check (VRC) Guidelines, 2019. https://developer.oculus.com/distribute/latest/concepts/publish-quest-req/\

Roupé, M., Johansson, M., Viklund Tallgren, M., Jörnebrant, F. and Tomsa, P.A., 2016. Immersive visualization of Building Information Models. In Living Systems and Micro-Utopias (CAADRIA 2016) (pp. 673-682). http://publications.lib.chalmers.se/records/fulltext/233949/local_233949.pdf

Rüppel, U. and Schatz, K., 2011. Designing a BIM-based serious game for fire safety evacuation simulations. Advanced engineering informatics, 25(4), pp.600-611. https://www.sciencedirect.com/science/article/pii/S1474034611000589

Schneider, Philip. "NURB Curves: A Guide for the Uninitiated". MACTECH. Link: http://preserve.mactech.com/articles/develop/issue_25/schneider.html. Retrieved 01 December 2019.

Shen, Z., Jiang, L., Grosskopf, K. and Berryman, C., 2012. Creating 3D web-based game environment using BIM models for virtual on-site visiting of building HVAC systems. In Construction Research Congress 2012: Construction Challenges in a Flat World (pp. 1212-1221). https://ascelibrary.org/doi/abs/10.1061/9780784412329.122

Unreal Python API, 2019. https://docs.unrealengine.com/en-US/PythonAPI/introduction.html

Unreal HLOD, 2019, https://docs.unrealengine.com/en-US/Engine/HLOD/index.html

D. Vajak and Č. Livada, "Combining photogrammetry, 3D modeling and real time information gathering for highly immersive VR experience," 2017 Zooming Innovation in Consumer Electronics International Conference (ZINC), Novi Sad, 2017, pp. 82-85. URL: http://ieeexplore.ieee.org/document/7968669/

Wang, B., Li, H., Rezgui, Y., Bradley, A. and Ong, H.N., 2014. BIM based virtual environment for fire emergency evacuation. The Scientific World Journal, 2014. https://www.hindawi.com/journals/tswj/2014/589016 /abs/

Woksepp, S. and Olofsson, T., 2008. Credibility and applicability of virtual reality models in design and construction. Advanced Engineering Informatics, 22(4), pp.520-528. https://www.ssimpleciencedirect.com/science/article/pii/S1474034608000517

Unity Technologies, 2019, https://unity3d.com/unity/features/multiplatform.

Wong, M.O., Du, J., Zhang, Z.Q., Liu, Y.Q., Chen, S.M. and Lee, S.H., 2019, February. An experience-based interactive lighting design approach using BIM and VR: a case study. In IOP Conference Series: Earth and Environmental Science (Vol. 238, No. 1, p. 012006). IOP Publishing. https://iopscience.iop.org/article/10.1088/1755-1315/238/1/012006/meta

Yan, Wei, Culp, C., Graf, R., 2011. Integrating BIM and gaming for real-time interactive architectural visualization. Automation in Construction 20 (4):446-458. http://dx.doi.org/10.1016/j.autcon.2010.11.013.

Yang, Z. and Kensek, K. (2018). Building Information Modeling and Virtual Reality: Workflows for Design and Facility Management. In ARCC Conference Repository. https://www.arcc-journal.org/index.php/repository/article/view/546

Yu-Cheng Lin, Yen-Pei Chen, Huey-Wen Yien, Chao-Yung Huang, Yu-Chih Su, Integrated BIM, game engine and VR technologies for healthcare design: A case study in cancer hospital, Advanced Engineering Informatics, Volume 36, 2018, Pages 130-145, ISSN 1474-0346, https://doi.org/10.1016/j.aei.2018.03.005.