THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

From BIM to VR

-The design and development of BIMXplorer

MIKAEL JOHANSSON

Department of Civil and Environmental Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

From BIM to VR
-The design and development of BIMXplorer

MIKAEL JOHANSSON

ISBN 978-91-7597-449-1

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 4130
ISSN 0346-718X

Department of Civil and Environmental Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Chalmers Reproservice
Gothenburg, Sweden 2016

From BIM to VR - The design and development of BIMXplorer

MIKAEL JOHANSSON
Department of Civil and Environmental Engineering
Chalmers University of Technology

Abstract

The architecture, engineering and construction (AEC) industries are currently undergoing a change from a drawing-based form of information exchange to a model-based. Using the concept of Building Information Models (BIM), the content produced by architects and designers has evolved from traditional 2D-drawings to object-oriented 3D-models embedded with information to describe any building in detail. This, in turn, has opened up new possibilities of using real-time visualization and Virtual Reality (VR) as a tool for communication and understanding during the design process. However, as primarily created to describe a complete building in detail, many 3D dataset extracted from BIMs are too large and complex in order to be directly used as real-time visualization models. Because of this, it is still difficult to integrate VR and real-time visualizations as a commonly used tool during the design process. The recent introduction of a new generation of Head-Mounted Displays (HMD) has also made the situation even more challenging. Although these new types of VR devices offer huge potential in terms of realism, sense of scale and overall suitability for design and decision-making tasks, they are also far more demanding when it comes to real-time rendering performance.

In order to address the current situation this thesis contributes with the design and evaluation of a new software application that provides a simple interface from BIM to VR. Following a design science research approach this application has been developed in order to fulfil a set of requirements that has been identified as important in order for VR and real-time visualization to become an everyday used tool for design and communication during the building design process. Along that path, three new technical solutions have been developed:

- An efficient cells- and portals culling system automatically realized from BIM-data.
- An efficient approach for integrating occlusion culling and hardware-accelerated geometry instancing.
- An efficient single-pass stereo rendering technique.

The final system – BIMXplorer – has been evaluated using several BIMs received from real-world projects. Regarding rendering performance, navigation interface and the ability to support fast design iterations, it has been shown to have all the needed properties in order to function well in practice. To some extent this can also be considered formally validated, as the system is already in active use within both industry and education.

Key words: Building Information Modeling, BIM, Virtual Reality (VR), Real-time rendering, Head-mounted display (HMD).

Acknowledgements

First of all, I would like to thank the people that were with me at The Visualization Studio at Chalmers Lindholmen – the place where all of this work really started! Claes and Börje – I hope that you are enjoying your time in retirement! Special thanks go to Mattias Roupé, my friend and colleague with whom I have had very close collaboration during this entire project. We have had a lot of fun over the years and I especially value your way of always seeing the potential and opportunities instead of problems in any given situation.

I would also like to thank all my other colleagues at the Construction Management department for their support, especially Mikael Viklund Tallgren and my main supervisor, the amazing Petra Bosch. Huge thanks also to my examiner Christian Koch as well as my academic writing coach, Christine Räisänen.

I would also like to acknowledge my former examiner, the late Professor Per-Erik Josephson, who made it possible for me to become a PhD student in the first place.

Last, but certainly not least, I would like to thank my family and friends for their company and support, especially the three most important people in my life, Jessica, Noah and Esther – thank you for all the love and happiness you give me every day!

Göteborg, August 2016

Mikael Johansson

Appended papers

Paper I:
"Efficient Real-Time Rendering of Building Information Models"
Johansson, Mikael; Roupé, Mattias. In *Proceedings of the 2009 international conference on computer graphics and virtual reality (CGVR09)*, July 13-16, 2009, Las Vegas, Nevada, USA, Pages 97-103.

Paper II:
"Real-time visualization of Building Information Models (BIM)"
Johansson, Mikael; Roupé, Mattias; Bosch-Sijtsema, Petra. *Automation in Construction*, Vol. 54 (2015), June 2015, Pages 69-82.

Paper III:
"Integrating Occlusion Culling and Hardware Instancing for Efficient Real-Time Rendering of Building Information Models"
Johansson, Mikael. In *GRAPP 2013: Proceedings of the International Conference on Computer Graphics Theory and Applications*, Barcelona, Spain, 21-24 February, 2013, Pages 197-206.

Paper IV:
"From BIM to VR – Integrating immersive visualizations in the current design process"
Johansson, Mikael; Roupé, Mattias; Viklund Tallgren, Mikael. In *Fusion - Proceedings of the 32nd eCAADe Conference - Volume 2 (eCAADe 2014)*, 10-12 September, 2014, Newcastle upon Tyne, England, Pages 261-269.

Paper V:
"Efficient Stereoscopic Rendering of Building Information Models (BIM)"
Johansson, Mikael. *Journal of Computer Graphics Techniques (JCGT)*, Vol. 5, No. 3, 2016, Pages 1-17.

Additional publications

"Immersive visualisation of building information models: Usage and future possibilities during design and construction"
Roupé, Mattias; Johansson, Mikael; Viklund Tallgren, Mikael; Jörnebrant, Fredrik; Tomsa, Petru Andrei. In *Proceedings of the 21st International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2016)*, 30 March-2 April, 2016, Melbourne, Australia, Pages 673-682.

"A BIM-supported framework for enhancing joint planning in construction"
Viklund Tallgren, Mikael; Roupé, Mattias; Johansson, Mikael; Andersson, Roger. In *Proceedings of the 32nd CIB W78 Conference 2015*, October 27th-29th, 2015, Eindhoven, The Netherlands, Pages 696-705.

"An empowered collaborative planning method in a Swedish construction company - A case study"
Viklund Tallgren, Mikael; Roupé, Mattias; Johansson, Mikael. In *Proceedings 31st Annual ARCOM Conference*, Lincoln, 2015, Pages 793-802.

"Interactive navigation interface for Virtual Reality using the human body"
Roupé, Mattias; Bosch-Sijtsema, Petra; Johansson, Mikael. *Computers, Environment and Urban Systems*, Vol. 43, 2015, Pages 42-50.

"Real-Time Rendering of large Building Information Models - Current state vs. state-of-the-art"
Johansson, Mikael; Roupé, Mattias. In *Proceedings of the 17th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2012)*, 25-28 April, 2012, Chennai, India, Pages 647-656.

"Using the human body as an interactive interface for navigation in VR models"
Roupé, Mattias; Johansson, Mikael. In *Proceedings of the 17th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2012)*, 25-28 April, 2012, Chennai, India, Pages 79-88.

"3D-City Modeling: A Semi-automatic Framework for Integrating Different Terrain Models"
Roupé, Mattias; Johansson, Mikael. *Advances in Visual Computing, 7th International Symposium, ISVC 2011*, September 26-28, 2011, Las Vegas, NV, USA, Pages 725-734.

"Towards a Framework for Efficient Use of Virtual Reality in Urban Planning and Building Design"
Johansson, Mikael. Thesis for licentiate degree, Chalmers University of Technology, Gothenburg, Sweden, 2010.

"How can GIS and BIM be integrated?"
Johansson, Mikael; Roupé, Mattias. Poster, *The 15th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2010)*, 7-10 April, 2010, Hong Kong.

"Supporting 3D City Modelling, Collaboration and Maintenance through an Open-Source Revision Control System"
Roupé, Mattias; Johansson, Mikael. In *Proceedings of the 15th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2010)*, 7-10 April, 2010, Hong Kong, Pages 347-356.

"Visual quality of the ground in 3D models: using color-coded images to blend aerial photos with tiled detail-textures"
Roupé, Mattias; Johansson, Mikael. In *Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa (Afrigraph 2009)*, February 4-6, 2009, Pretoria, South Africa, Pages 73-79.

"Virtual Reality Supporting Environmental Planning Processes: A Case Study of the City Library in Gothenburg"
Suneson, Kaj; Allwood, Carl Martin; Heldal, Ilona; Paulin, Dan; Roupé, Mattias; Johansson, Mikael; Westerdahl, Börje. *Knowledge-Based Intelligent Information and Engineering Systems, 12th International Conference, KES 2008*, Zagreb, Croatia, September 3-5, 2008, Pages 481-490.

"Virtual Reality As a New Tool in the City Planning Process"
Suneson, Kaj; Allwood, Carl Martin; Paulin, Dan; Heldal, Ilona; Roupé, Mattias; Johansson, Mikael; Westerdahl, Börje. *Tsinghua Science and Technology*, Vol. 13, No. S1, 2008, Pages 255-260.

"User' evaluations of a virtual reality architectural model compared with the experience of the completed building"
Westerdahl, Börje; Suneson, Kaj; Wernemyr, Claes; Roupé, Mattias; Johansson, Mikael; Allwood, Carl Martin. *Automation in Construction*, Vol. 15, Iss. 2, 2006, Pages 150-165.

"Building Information Modelling for Visualisation in AEC Education"
Horne, Margaret; Roupé, Mattias; Johansson, Mikael. *The 5th conference of Construction Applications of Virtual Reality (CONVR 2005)*, 12-13 September, 2005, Durham, United Kingdom.

"From CAD to VR - Implementations for Urban Planning and Building Design"
Johansson, Mikael; Roupé, Mattias. *Digital Design: The Quest for New Paradigms (23nd eCAADe Conference Proceedings)*, 21-24 September, 2005, Lisbon, Portugal, Pages 399-405.

"From CAD to VR – focusing on urban planning and building design"
Roupé, Mattias; Johansson, Mikael. *AVR III, Conference and Workshop on Applied Virtual Reality and Open Source VR programming*, Gothenburg, Sweden, May 27-28, 2004.

Table of contents

1 Introduction

Real-time visualization and Virtual Reality (VR), have many applications within the Architecture, Engineering and Construction (AEC) industries (Bouchlaghem et al., 2005; Woksepp and Olofsson; Greenwood et al., 2008). With the ability to navigate freely through 3D scenes from a first-person perspective, it is possible to present and communicate ideas regarding future projects in a way that facilitates understanding among all involved parties, despite their background or professional expertise (Kjems, 2005; Westerdahl et al., 2006). For people with limited experience of interpreting traditional design documents, such as 2D drawings, the technology offers a representation that avoids misunderstanding and allows for a thorough apprehension of any type of building or facility (Mobach, 2008).

However, despite all the documented benefits that VR technology offers, it is still not used as an everyday tool during the design process. Instead, its use remains restricted to certain projects of high importance (Bullinger et al., 2010; Liu at al., 2014). In the past, this was mainly due to lack of affordable hardware offering sufficient computing power, but also the fact that the actual design was almost always performed in 2D. As a consequence, any use of VR required the time-consuming creation of a separate 3D model (Westerdahl et al., 2006; Sunesson et al. 2008), using the original design documents as a reference. As this was typically done by someone else than the architect, it severely affected a natural integration of the technology. Even when 3D CAD data was available from the actual design, it still had to be converted to a representation suitable for real-time visualization by optimizing it and adding material properties, such as textures.

Nevertheless, with the introduction of Building Information Models (BIM) within the AEC field new possibilities have emerged. Using modern modelling tools, such as Autodesk Revit or ArchiCad, the content produced by architects and designers has evolved from traditional 2D-plans and elevations to object-oriented 3D-models embedded with information to describe any building or facility in detail (Eastman et al., 2011). In theory, BIM then supports an easier and more natural integration of VR during the design process. As 3D data is available from the actual design work, there is no longer a need to create a separate 3D-model for the sole purpose of visualization.

However, as primarily created to describe a complete building in detail, many 3D dataset extracted from BIMs are too large and complex in order to be directly used as real-time visualization models (Dvorak et al., 2005; Jongeling et al., 2007; Pelosi, 2010; Steel et al., 2012; Dalton and Parfitt, 2013; Shi et al., 2015). To support interactive, real-time navigation the dataset often has to be significantly reduced or otherwise optimized – a process that may involve hours or even days to perform (Dubler et al., 2010; Liu at al., 2014). With current solutions users and stakeholder thus have to either resort to time-consuming pre-processing steps or accept that the visualization may not always be considered fully interactive or free of visual artefacts, i.e. errors. Because of this, it is still difficult to integrate VR as a commonly used tool for design and communication. Although visualizing large amounts of 3D-data in

real-time is an active research topic by itself (Yoon et al., 2008), there has been surprisingly little attention given to the specific case of visualizing large BIMs in real-time.

With the recent introduction of a new type of consumer-directed Head Mounted Displays (HMD), such as the Oculus Rift, the problem of managing large BIMs interactively has become even more relevant to solve. Although these new types of VR devices offer huge potential in terms of realism, sense of scale and overall suitability for design and decision-making tasks, they are also far more demanding when it comes to real-time rendering performance. Not only do they require a 3D scene to be rendered twice in order to produce the stereoscopic effect, but they also require a much higher update rate. Compared to typical desktop VR applications, the performance requirements have essentially increased three-folded, making an existing problem become far worse. Fortunately, this thesis has only one real purpose – to deliver a solution to this problem.

1.1 Aim and objectives

The main aim of the work presented in this thesis is to develop a software application that will allow VR to become an everyday used tool for design and communication during the building design process. As already outlined, one of the biggest obstacle for this to be realized today lies in the difficulties to directly – i.e. without any time-consuming pre-process – visualize BIMs in real-time. The main objectives are therefore to develop techniques and algorithms that allow large and complex BIMs to be directly visualized in real-time.

To better guide this process, the following research questions are investigated and considered:

RQ1: What is the current state when considering real-time visualization of BIMs?

The problem of visualizing BIMs has been highlighted in earlier studies, but it hasn't really been thoroughly investigated. The current literature contains many examples where problems of using large BIMs for visualization purposes have been expressed but often the exact details, such as what type of models, software and hardware that has been used are simply omitted. Similarly, this question also relates to the type of models that can be expected in real-world cases.

RQ2: What is a suitable acceleration technique for typical 3D building models?

There are a number of existing techniques and algorithms that can be utilized in order to accelerate real-time rendering. These have all strengths and weaknesses and a suitable choice is highly dependent on the type of 3D environment that it should be applied to. For instance, a vast, open landscape seen in a flight simulator is very different from a detailed city environment seen from the ground level when it comes to performance optimization. When considering buildings in general – which BIMs typically represent – they often feature a lot of occlusion, i.e. enclosed regions. Can we take advantage of this in an efficient manner?

RQ3: How can we take advantage of BIM-data in order to accelerate rendering?

When considering existing acceleration techniques they have primarily been developed with general 3D-models in mind, i.e. as received from CAD or DCC tools. As such, they become inherently limited by the lack of information beyond pure geometrical data. BIMs, on the other hand, contain much more information, i.e. metadata, such as detailed object properties as well as any relations to other objects. Can this additional information be utilized in order to accelerate real-time rendering?

RQ4: How can we support a natural integration of VR within the building design process?

In order for VR to become a natural and integrated part of the building design process, several barriers needs to be overcome. In this context the technical ability to visualize large and complex BIMs in real-time only represents one of them. How these techniques are actually implemented with regards to accessibility, as well as usability and interface becomes equally important consider. For instance, if time-consuming processes – although automated – are required to realize a VR session this will probably affect an integration negatively. Similarly, if the actual navigation interface is found too complex for non-experts, it will be difficult to support end-user participation.

2 Background and related work

2.1 Definition of Virtual Reality (VR)

Ever since Sutherland (1965) first articulated the term Virtual Reality (VR) it has been defined and used in many different ways. Ranging from simple environments presented on a desktop computer to fully immersive environments experienced through head-mounted displays and tracker systems, the term now means different things in various contexts. Within the scope of this thesis, VR is defined as a computer-generated visualization of spatial data that can be interactively controlled by a user and displayed on any type of screen. Furthermore, the primary application that has been considered is that of *real-time architectural walkthroughs* where users can explore and navigate through interior and exterior spaces of a virtual building model (Mobach, 2008; Liu at al., 2014).

2.2 Real-time rendering

The field of 3D computer graphics includes several techniques that aim at producing 2D images – often called frames – of three-dimensional geometric data. This is realized using a process known as 3D rendering, where the input 3D-data is converted to pixels in a 2D image based on the location of a virtual camera (Figure 1). When the resulting 2D image is displayed onto a computer screen it basically represents a window into a three-dimensional world from a specific location. The 3D rendering process can either be done in real-time or performed offline, i.e. non-real-time. When non-real time rendering is used the purpose is to produce a single image – or multiple images that are composed into an animation sequence or film – of high quality that later can be displayed on a computer screen. Depending on the size and complexity of the input 3D-data, the processing time ranges from minutes to hours or even days, but the end result will in turn be able to realistically simulate lighting, shadows, reflections and other natural phenomena. The end result, however, is a static image or film sequence that once created cannot be changed by the user – it only represents a single view of the input 3D-data.

Real-time rendering, on the other hand, takes a different approach as the process is repeated continuously. During this process, the virtual camera can be moved freely, thus giving the user the impression that he or she is travelling around in a virtual world. The actual navigation can be controlled by an input device, such as a mouse or joystick, and gives the user the ability to interact with the system. However, to make this "virtual journey" smooth and interactive, the computer has to generate a sufficient number of frames per second to prevent the user from experiencing motion sickness (Hettinger, 1992). In order to be able to perform this computational expensive process, dedicated graphics hardware, also known as GPU (Graphics Processing Unit), has to be used. Using a technique known as *rasterization*, the GPU handles all the computations required to produce 2D image views of the 3D-data (Akenine-Möller et al., 2008). Although powerful, the GPU still has limitations on the amount of data that can be processed within a given time frame. This basically means that there always exists an upper limit on the amount of 3D-data that can be interactively visualized.

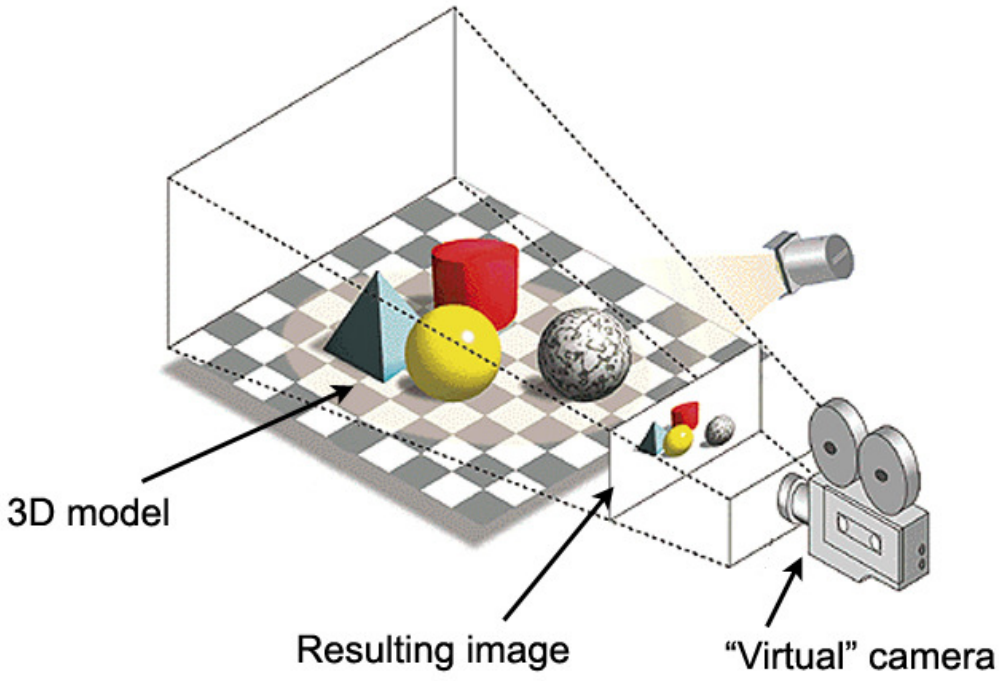

Figure 1: With 3D rendering, an input 3D model is converted into an image based on the position of a "virtual" camera.

2.3 Display systems

Although real-time rendering is the main technology for producing images of non-existing objects or environments, the result may be displayed in a wide variety of ways. Ranging from display on regular computer screens to solutions where a user is wearing a head-mounted display (HMD), the basic difference is the level of immersion they enable. Here, the level of immersion may be defined as the degree to which a user feels completely surrounded by the virtual world. When considering *Immersive VR*, the most used solutions today are either a CAVE (Cruz-Neira et al., 1992) or Head-Mounted Display (HMD) (Burdea and Coiffet, 2003). Examples of these are shown in Figure 2.

The HMD naturally supports stereoscopic vision in that it uses a small display for each eye: one for the left and one for the right. By rendering the virtual world from two slightly horizontally separated (virtual) cameras for each eye, the user experiences stereoscopic vision similar to that in real life (Kjellin, 2008). For the CAVE solution LCD shutter glasses are often used in order to provide the stereoscopic vision. These shutter glasses work by blocking one eye's view of the screen (or wall). When an image is rendered for the left eye, the shutter glasses block the right eye view. For the right eye the process is then reversed. The switching process is synchronized with the graphics card and performed at a high frequency, thus giving the user a perceived true stereoscopic vision.

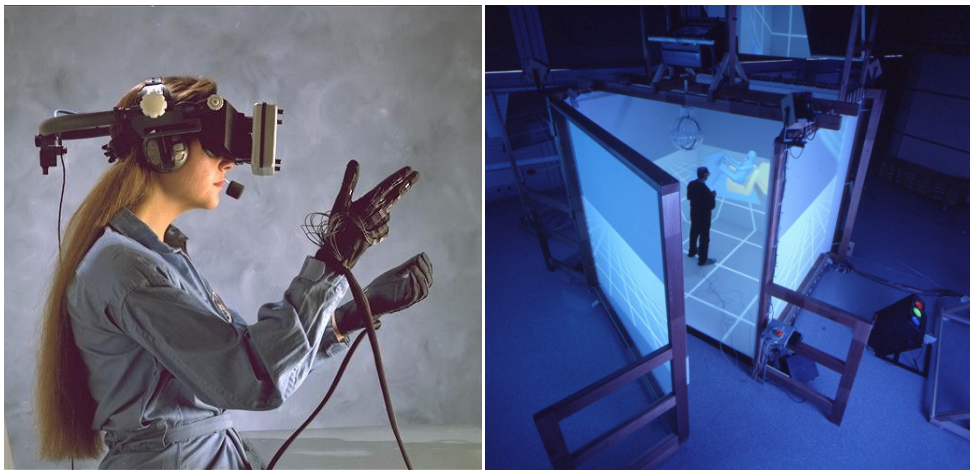

Figure 2: HMD (left) and CAVE (right) (Courtesy NASA and Chalmers)

A semi-immersive alternative to the CAVE is the *Powerwall* (Westerdahl et al., 2006). As shown in Figure 3, it is basically a small cinema screen allowing for many people to view the VR simulation simultaneously. By using stereographic shutter glasses or polarized glasses, stereoscopic vision is enabled.

Figure 3: A Powerwall solution.

Finally, *Desktop VR* (Modjeska, 2003) refers to the case when the real-time rendering is displayed on a regular computer screen or portable computer screen, without stereoscopic vision. This solution is the simplest in terms of required hardware, and by using a projector the system is also suitable for a larger audience. When using the "projector-assisted" approach

of Desktop VR, the solution thus basically becomes a small Powerwall without stereoscopic vision.

Until very recently, HMDs have been either low-cost-low-performance or high-cost-high-performance devices (Dörner et al., 2011), making them less useful in practice. However, with the introduction of a new generation of consumer-directed HMDs, such as the Oculus Rift and HTC Vive (Figure 4) this has completely changed. These devices provide a high resolution, large field-of-view as well as orientation and positional tracking ability at a very competitive price (around $600-1000). Within the scope of this thesis, both the Desktop VR solution as well as the new generation of HMDs has been targeted.

Figure 4: Oculus Rift (left) and HTC Vive (right)

2.4 BIM and real-time visualization

A Building Information Model (BIM) may be defined as a digital representation of the physical and functional characteristics of a building. Compared to a general 3D-CAD model, a BIM is a different kind of representation since it defines not only geometrical data but also specifications and information regarding spatial relations and connections among the included components. The creation of a BIM is typically done in a modern modeling tool, such as *ArchiCAD* or *Autodesk Revit*. These systems represent each component in a building as an object with parametric properties and relations to other parts of the building. The collection of objects is not seen as a 2D-drawing or 3D-model, but is instead stored in a single database that represents the complete building. From this database it is then possible to derive different representations, such as a 2D drawing or 3D model (Figure 5). As all representations originate from the same data, any change in the database will automatically update all representations. This property makes the system especially efficient when considering revisions and updates. As a repository of information a BIM support a multitude of applications along the design and construction process, including cost-estimation, energy analysis and production planning (Eastman et al., 2011).

For the majority of BIM authoring tools the underlying data-model closely resembles that of the Industry Foundation Classes (buildingSMART, 2007). The IFC was designed to provide a universal basis for the information sharing over the whole building lifecycle (Eastman, 1999), and is the de facto standard for representing BIMs. It differs from general 3D-file formats, such as 3D Studio, FBX or COLLADA (Arnaud and Barnes, 2006), in that it represents a

building or facility with specific (virtual) building objects instead of pure geometrical entities. The IFC scheme supports a wide variety of buildings objects, such as IfcWall, IfcDoor, IfcWindow, IfcSlab and IfcRoof together with an unlimited set of properties connected to each object. Using the IfcRelation feature, any object can also relate to other objects, making it possible to form constraints and relations between building parts. For instance, a door "knows" that it is placed in a particular wall. Another major difference between IFC and general 3D-file formats is the representation of space. Every instance of an IFC-object must belong to a spatial context. Special space-enclosing structures are the sites (IfcSite), buildings (IfcBuilding), storeys (IfcBuildingStorey) and rooms (IfcSpace). Additionally, any window or door placed in a wall results in an opening element (IfcOpening) that represents the cut-out in the affected wall.

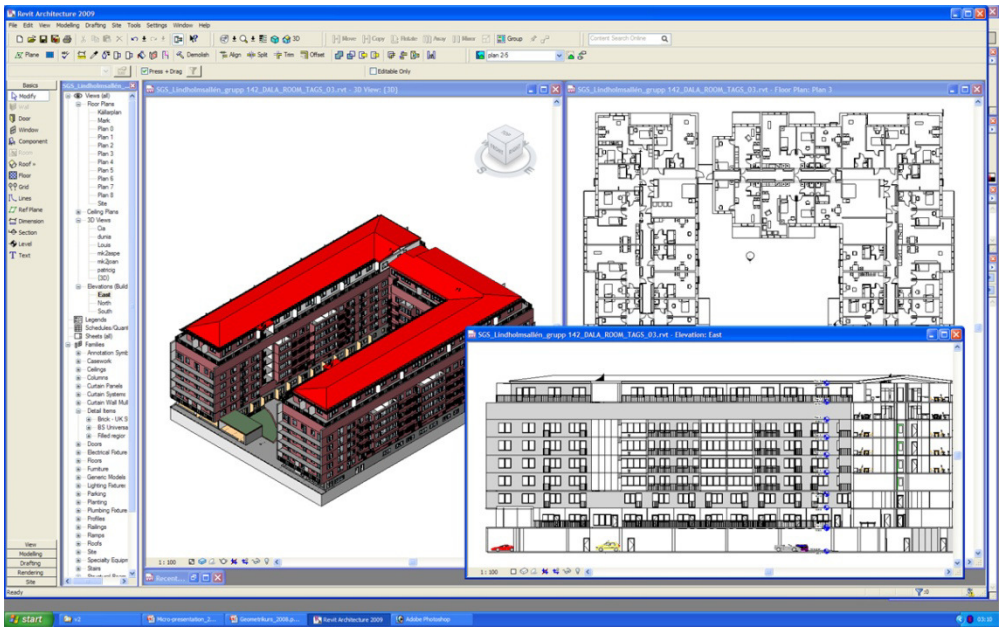

Figure 5: A BIM created in Autodesk Revit.

Although practically all BIM authoring applications follows the IFC-specification they still differ in many ways when considering the level of information they contain and the ability to extract that information. In that sense, it can be said that each application defines BIM in its own way. In order to support the whole range of possible authoring environments, the work presented in this thesis therefore defines a BIM according to the IFC-specification. That is, the acceleration techniques presented within the scope of this thesis have been designed to work with models defined according to the IFC-specification, and do not rely on any additional data or features beyond that.

The introduction of BIM within the AEC field is interesting, as it makes it possible to use a single source of data for 2D-drawings, offline renderings as well as real-time renderings and VR. In theory, this should make it much easier to integrate real-time visualizations as a design and communication tool during the actual design process. As 3D-models can be extracted directly from the BIM-systems, there is no longer a need for the additional creation of a

separate 3D-model for visualization purposes. However, in practice, this development has also introduced a new set of challenges. As primarily created to describe a complete building in detail, BIMs can be too large and complex in order to be directly used for real-time rendering (Dvorak et al., 2005; Svidt and Christiansson, 2008; Steel et al., 2012; Dalton and Parfitt, 2013). Although commonly used software tools for BIM visualization is able to directly load models regardless of size and complexity it is often difficult to achieve smooth frame rates without further processing of the input dataset or by introducing non-conservative acceleration techniques (Dubler et al., 2010). It is therefore common that a visualization session has to be preceded by an optimization step in order to make the dataset more suitable for use in a real-time environment. However, as this step has to be repeated as soon as the design changes it severely affects an efficient integration of real-time visualization as a communication tool (Dubler et al., 2010; Liu at al., 2014).

Recent times have also seen the use of so-called game engines for the purpose of real-time visualization of BIMs (Yan et al., 2011; Shi et al., 2015; Merschbrock et al., 2016). Given the image quality and immersion offered by modern computer games, game engines are often put forward as a better alternative to conventional BIM-viewers, such as Navisworks, as they offer high rendering performance and more elements of interactivity. However, although it's true that it is possible to use game engines to produce impressive visualizations, they still require a lot of manual work in order for this to be realized (Shi et al., 2015; Merschbrock et al., 2016). In Figure 6, a typical workflow from BIM to VR using game engines is illustrated (Halaby, 2015). As can be seen, there are several steps that needs to be performed, including model optimizations and other processes to provide real-time performance, e.g. occlusion bake. Even with a streamlined process as the one described in Figure 6, this can easily add up to several hours or even days, depending on the size and complexity of the BIM. As discussed previously, this is a process that needs to be repeated for every major change of the design.

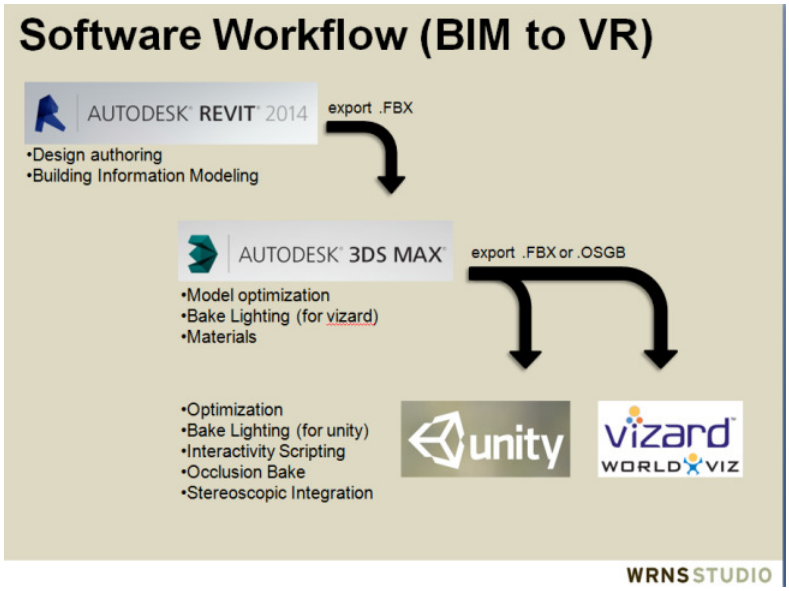

Figure 6: Typical software workflow from BIM to game engine (Halaby, 2015)

Looking forward, it could be argued that the ever increasing speed of CPUs and GPUs will solve this problem simply by brute-force performance. However, at the same time we also see that BIMs tend to become more detailed in terms of geometry and amount of objects as this field matures. In addition, new display hardware puts even higher performance demands as resolution (e.g. 4K screens) and frame rate requirements (e.g. Oculus Rift and HTC Vive) increase. As such, the problem of interactivity and real-time performance is still important to solve.

2.5 Frame rate and interactivity

An important property for any type of real-time visualization system is its ability to maintain a sufficiently high frame rate. As defined by the frequency at which new images are presented on screen it inherently affects user experience and task performance. A too low frame rate will make the system less responsive and make navigation and other interaction tasks more demanding at the same time as it greatly diminishes the sense of continuous motion. When considering a minimum frame rate, many studies have found that user performance becomes significantly reduced below 15 Hz for a number of different applications. Reddy (1997) investigated the effects of different frame rates on human performance when faced with a simple heading task in a virtual environment. During these tests low frame rate substantially degraded user performance and a frame rate of around 15 Hz was suggested to serve as a minimum requirement for a generally acceptable degree of performance. However, it was also suggested that a higher frame rate should be strived for, as this was shown to improve performance even further.

Barfield et al (1998) studied the perceived level of presence within a virtual environment as a function of input device type and update rate. Although the type of input device had limited effect, it was found that an update rate of at least 15 Hz was a critical value in order to experience a sense of presence. Regarding ease and comfort of navigation, interactivity and smoothness of motion, an update rate of 15 Hz was considered equally important. The study further revealed consistently higher ratings for all factors when the update rate was increased to 20 Hz.

In a more recent study, Claypool (2007) investigated the impact of frame rate on player performance in first person shooter games. For movement tasks it was found that an increase of frame rate from 7 Hz to 15 Hz significantly improved player performance. Unfortunately, no data were collected beyond 15 Hz. For shooting tasks, the frame rate was found to be even more important. Although the rate of improvement was steeper below 15 Hz, player performance was increased all the way up to 60 Hz.

For everyday 3D design and engineering applications the importance of maintaining a sufficiently high frame rate has also been highlighted. Experiments at The Boeing Company show that low frame rate decreases the feeling of continuous motion and that improved continuity helps user performance when searching for objects in a complex 3D model (Kasik

et al., 2002). Furthermore, empirical studies revealed that, although 10 Hz was considered useful, massive model visualization users require at least 16 Hz in order to be considered acceptable (Yoon et al., 2008).

Still, in practice, 15 Hz is not generally considered a sufficient level of interactivity. For applications such as architectural walkthroughs, 30 Hz is often recommended as a minimum frame rate in order to provide a suitable experience (Shiratuddin and Fletcher, 2006). Below this number users will typically start to experience lag to such a degree that the impression of continuous movement is lost (Herwig and Paar, 2002; Göttig et al., 2004).

Looking at the computer games industry it also becomes clear that 15 Hz is not enough to satisfy player demands. For so-called first-person shooter games (FPS) 30 Hz is generally considered the absolute minimum and many game developers even target 60 Hz in order to give players a smooth and responsive experience (Rubino and Power, 2008).

In this context it is also important to highlight the main difference between film (e.g. motion pictures) and real-time rendering in terms of frame rate. With current cinema using a standardized frame rate of 24 Hz, it may not be obvious to see the benefit of going beyond this number. However, while the rendered images represent singular moments in time, film can record motion-blurred images which effectively integrate information over time. As a consequence, smooth motion can be displayed at a comparably lower frame rate without suffering from apparent "jumps" between discrete moments in time (Rubino and Power, 2008).

Nevertheless, with the introduction of a new generation of HMDs, such as the Oculus Rift and HTC Vive the frame rate requirements have changed significantly compared to that of desktop VR. In order to provide a useful VR experience 90Hz is now considered critical (Hasan and Yu, 2015). This requirement is also different compared to that of desktop VR, where a lower frame rate may still give a sufficient experience, albeit with less fidelity. Due to a very strong connection to the tracker system, any update rate below 90Hz will produce such visual artefacts that the system essentially becomes useless.

2.6 Rendering performance and acceleration techniques

In Figure 7 the graphics pipeline is illustrated. As can be seen it is built up by a number of different stages and involves both the CPU and the GPU. The interaction between the CPU and GPU can be thought of as a client-server pair, where an application acts as the client on the CPU-side and uploads data and issues commands through the graphics driver that the GPU then processes. To take advantage of hardware-accelerated rendering an application will upload 3D positions that represent the vertices (i.e. corners) of a set of triangles to GPU memory, and then issue draw commands that specifies which triangles to draw on screen. The GPU will then transform and project each one of these triangles according to the position of a

"virtual" camera and, finally, convert it to a two-dimensional image using a process known as rasterization.

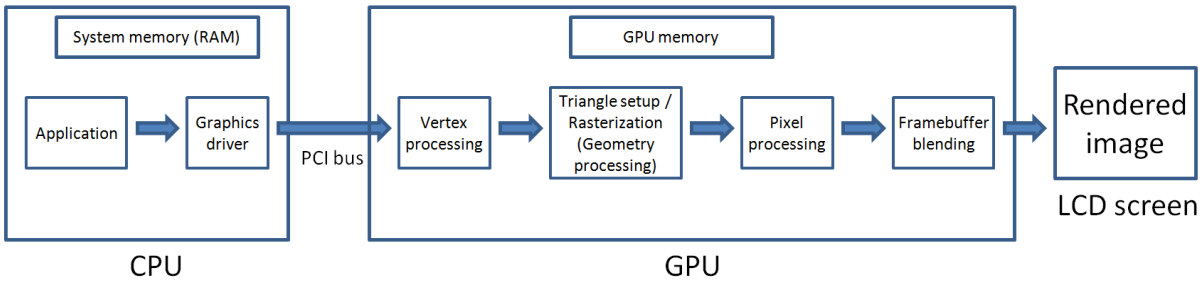

Figure 7: The graphics pipeline.

While the processes performed on the GPU used to be "fixed-function", it is nowadays fully programmable using so-called shader programs which offer developers more flexibility when it comes to processing geometry and compute lighting. As illustrated in more detail in Figure 8 there is typically three main shader stages – vertex, geometry and fragment.

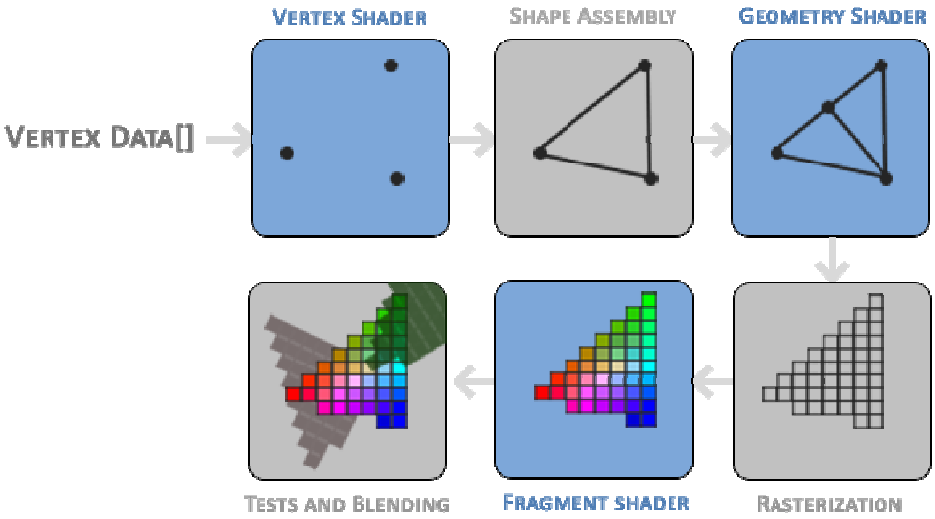

Figure 8: Different shader stages in the graphics pipeline.

During the vertex shader stage, each vertex is individually processed. This stage transforms and projects vertices to match the current view position (i.e. camera). During the geometry shader stage complete primitives (i.e. triangles) are processed. A geometry shader is optional and takes a single primitive as input and may output zero or more primitives. As such, it has the ability to amplify geometry. Finally, the fragment shader processes individual fragments (i.e. pixels) generated by the rasterization into colors that will appear on screen.

When an application takes advantage of hardware-accelerated 3D rendering "out-of-the-box", all of the geometry of a given 3D scene has to go through at least some parts of the graphics pipeline every frame. Even with very powerful CPUs and GPUs such a scenario will therefore

always have an upper limit in the amount of geometry that can be processed at a certain frame rate. However, by taking advantage of additional acceleration techniques it is often possible to go beyond this limitation (Akenine-Möller et al., 2008). In general, these acceleration techniques can be assigned into three different categories: *pipeline optimizations* which increase performance by streamlining data flow through the graphics pipeline, *Level-of-detail (LOD)* which increases performance by reducing the geometric complexity of far-away objects and *visibility culling* which increase performance by rejecting non-visible objects. In the following subsections each one of these acceleration techniques are explained in more detail.

2.6.1 Pipeline optimizations

As with any other type of pipeline, the speed at which data can flow through the graphics pipeline is inherently dictated by the slowest stage. The general idea behind pipeline optimizations is to remove the bottlenecks without reducing the amount of geometry to process. For instance, it is very often the case that an application is CPU-bound as opposed to GPU-bound. What that means is that the GPU processes data and rendering tasks at a faster rate than the CPU is able to feed it with new data and commands. In essence, the GPU becomes underutilized. One of the main reasons for this behaviour is the amount of draw commands and state changes that are made every frame (Hillaire, 2012). Rendering a set of triangles in graphics APIs such as OpenGL usually involves two main steps: (1) modifying the OpenGL states and objects in order to setup resources (i.e. textures and vertex arrays) used for rendering and (2) issuing the actual draw call to tell the GPU to render the triangles. Both these steps involve multiple calls to the graphics driver and therefore incur a certain cost (i.e. time) on the CPU-side. In CPU-bound situations it is therefore possible to improve performance by reducing state changes and draw calls. When considering state changes many of them can often be removed by sorting the objects to render based on state, such as materials and textures. By doing so, the states required for each material only has to be set once per frame (as opposed to multiple times if objects are rendered in an unsorted way). Furthermore, in order to reduce draw calls there are mainly two ways – *geometry batching* and *geometry instancing*. With batching the idea is to combine geometry that share the same state (e.g. material) in order to form larger, but fewer, “chunks” of geometry to render (Wloka, 2003). By doing so, the same amount of geometry can be rendered, but with much fewer draw calls. In contrast, geometry instancing takes advantage of the fact that many 3D scenes contain replicated geometry, such as the wheels on a car or all the chairs around a dining table (Dudash, 2007). With the instancing abilities on modern GPUs it is possible to submit a single draw call when rendering several objects that share the same geometry. By uploading each instance´s unique position to the GPU in a previous steps, they can all be transformed to their correct place during the vertex shader stage.

2.6.2 Level-of-detail (LOD)

In contrast to pipeline optimizations, the idea behind LOD is to reduce the amount of geometry that has to be drawn every frame. When objects are far away from the viewer it is

often possible to use a much simpler representation of them without affecting the visual quality too much (Luebke et al., 2002). As illustrated in Figure 9, this technique involves the creation of several different versions of an object, each one being represented by fewer triangles than the previous one. The selection of which version to use for rendering is then based on the distance to the viewer. However, although techniques exist to automate the creation of simplified versions of an object, they usually involve some sort of manual interaction in order to reach satisfying results (Garland and Heckbert, 1997).

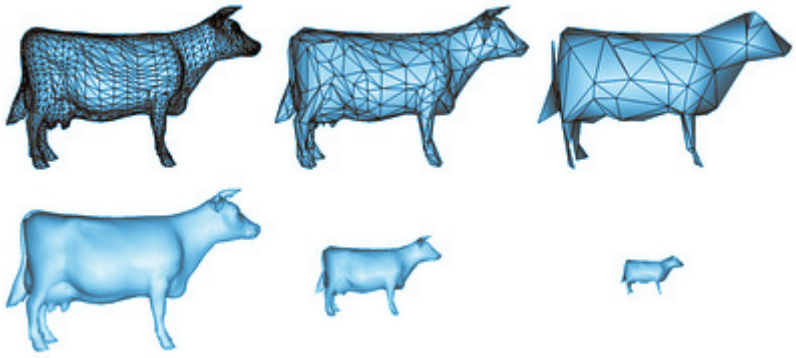

Figure 9: Illustration of LOD. The original object is replaced by simplified representations when far away from the viewer.

2.6.3 Visibility culling

As with LOD, the idea behind visibility culling is to increase performance by reducing the amount of geometry to draw. However, instead of rendering simplified objects, visibility culling tries to identify non-visible objects that don't need to be drawn at all (Cohen-Or et al., 2003). As illustrated in Figure 10, left, there will always be objects in a 3D scene that cannot be seen from a certain point of view, because they are either outside the field-of-view or hidden by other objects. The simplest form of visibility culling is *view-frustum culling*, which rejects objects that are outside the visible field-of-view (Figure 10, middle). This operation is typically not performed per-triangle but instead per-object using the objects bounding box (i.e. a box that fully encloses the object) for quick rejection. A more complex form of visibility culling is *occlusion culling* which rejects objects that are hidden by other objects (Figure 10, right). Compared to view-frustum culling this is a much more complex process as it requires computing how objects in a 3D scene affect each other. However, with the introduction of *occlusion queries* it has been possible to take advantage of the GPU to perform the actual visibility detection (Bartz et al., 1998). In essence, occlusion queries allow an application to render some geometry and then "ask" the GPU if it turned out to be visible. This way, proxy geometries (i.e. bounding boxes) can be used to test an object for visibility before it is actually rendered (Figure 11).

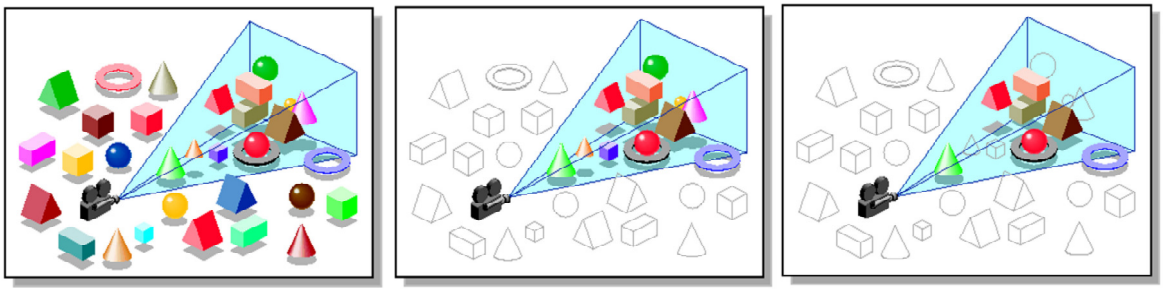

Figure 10: Illustrating no culling (left), view-frustum culling (middle) and occlusion culling (right). "Dimmed" objects are not sent to the GPU for rendering.

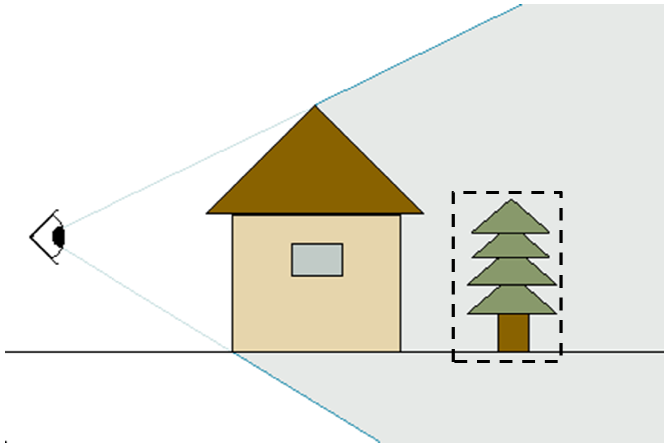

Figure 11: Using hardware occlusion queries it is possible to test the tree for visibility using a bounding box representation (dashed lines) before rendering the actual tree model.

As described above, both view-frustum culling and occlusion culling is performed *online* and therefore requires no offline pre-computations. However, there is also visibility culling techniques that works by pre-computing a *potentially visible set* from multiple regions in the 3D scene. During run-time, this set is then indexed in order to quickly obtain the objects that are *potentially* seen from a certain region in the scene (Funkhouser and Séquin1993).

A somewhat hybrid approach is *cell-and-portal culling* which primarily lends itself for use in indoor environments (Luebke and Georges 1995). As illustrated in Figure 12 it involves the creation of *cells* that are connected by *portals*. By using this data structure it is possible to restrict rendering only to objects that are in the same cell as the camera as well as objects that can be seen in adjacent cells *trough* the portals.

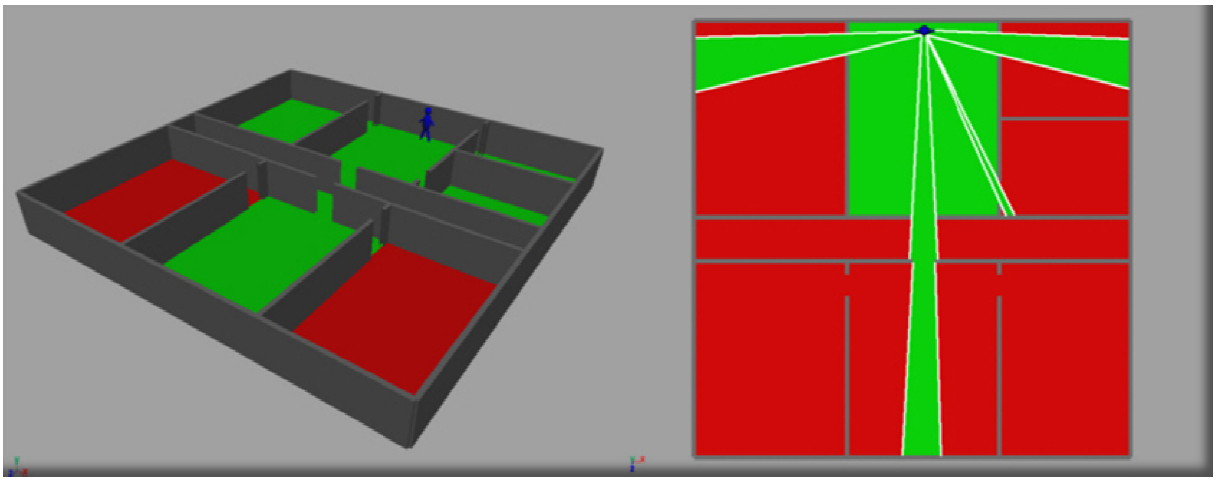

Figure 12: Cell-and-portal culling

3 Research approach

The work presented in this thesis mainly falls into the category of *technology and design science*. As such, it represents *constructive research*. As opposed to natural science, technology and design related research may be considered "artificial" in that it produces new artefacts and knowledge within a problem-solving paradigm. The concept is further explained by March and Smith (1995, p.253) who states: "Whereas natural science tries to understand reality, design science attempts to create things that serve human purposes."

Within this paradigm, several research approaches that share a similar philosophy exists. When considering the area of Information Systems (IS), the design science research approach has mainly been popularized by Hevner et al. (2004). However, in many ways it has already been a principal approach in engineering research and computer science for a long time (Kuechler and Vaishnavi, 2008). Similarly, it is also very closely related to the constructive research approach (Piirainen and Gonzalez, 2014).

Nevertheless, regardless of specific research approach, design–oriented research is mainly concerned with the task of designing and evaluating an artefact. As discussed by Hevner et al. (2004), such an artefact needs to address an existing unsolved problem, should build on and contribute to theoretical knowledge of the problem domain and should be proven to actually improve on existing solutions or attempts to solve the problem.

Since this approach exhibit many similarities to what software developers and engineers do as part of their regular jobs, it may not be obvious what distinguishes design science research – as well as constructive research – from conventional design work. According to Hevner, there are two important differences between design research and the practice of design. First, design science involves thorough and careful use of existing theories and methods from the scientific knowledge base in order to build and evaluate the particular artefact. Secondly, it contributes to the scientific knowledge base by scholarly dissemination. As an effect of the latter, design research also tries to solve a class of problems as opposed to a specific situated problem, which is more common in design practices.

As for the actual artefacts, it has become well established within the design science field to identify four different types; *constructs*, *models*, *methods*, and *instantiations* (March and Smith, 1995; Hevner et al. 2004; Johannesson and Perjons, 2014).

Constructs are definitions and concepts that form the "language" of a domain. They are the smallest conceptual parts that make it possible to understand and communicate about various phenomena. Typical examples are the concepts of method in Java or class in the Unified Modeling Language (UML).

Models represent possible solutions to practical problems. They are sets of propositions or statements that express relationships among constructs. For instance, a database model can be used for developing a database system.

Methods are a set of steps (an algorithm or guideline) used to perform a task or solve a defined problem. Typical examples are methods for database design or a search algorithm.

Instantiations are working systems that can be used in a practice. They are realizations of an artefact in its environment, such as a database for electronic medical records or a Java program realizing a search algorithm. Instantiations can always embed constructs, models, and methods.

Furthermore, Gregor and Hevner (2013) discuss how different design science contributions, i.e. artefacts, can be classified according to the maturity of the solution, as well as the application domain. As illustrated in Figure 13, they identify four different types of contributions – *improvements*, *inventions*, *exaptations*, and *routine design*.

Improvements are new solutions for known problems. These kinds of contributions address an existing problem with a new or enhanced solution, such as one offering better efficiency, usability or utility compared to the previous state of the art.

Inventions are new solutions for new problems. These kinds of contributions involve an innovation that addresses a new and unexplored problem by offering a novel solution, such as the first X-ray machine or the first data mining system. As such, they are typically much less common than improvements.

Exaptations are known solutions extended to new problems. These kinds of contributions adapt or extend an existing solution to address a problem for which it was not intended for in the first place, such as the use of data mining in meteorology.

Routine designs are known solutions to known problems. These types of designs are often the application of existing knowledge to a well-known problem, such as the creation of a business application using best practice solutions extracted from the knowledge base. In contrast to the previous discussed types, routine designs do not offer the same opportunity to contribute to the archival knowledge base of foundations and methodologies. As such they typically do not count as design science research contributions.

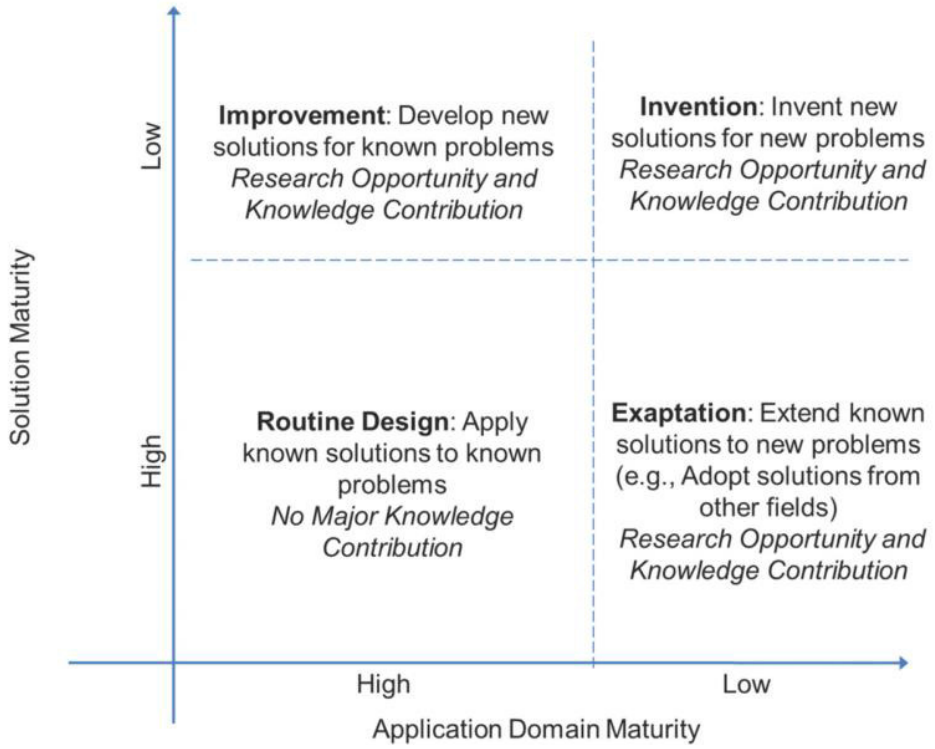

Figure 13: Design science research knowledge contribution framework (Gregor and Hevner, 2013)

When positioning the design science contribution presented in this thesis it falls within the *improvements* quadrant in that it improves on the previous state of the art with respect to efficiency as well as utility. As such, this research offers potential to contribute new knowledge to the scientific knowledgebase. As for the actual artefact, it is considered an *instantiation* in that it is a working system that can be used in practice.

When considering the actual research approach, Hevner et al. (2004) and Hevner (2007) propose a framework containing three cycles that places the design activity into a scientific framework (Figure 14). The three cycles are the *design cycle*, *relevance cycle* and *rigor* cycle. The core of the framework is the design cycle, which represents an iterative process where design alternatives are generated and evaluated against the requirements until a satisfactory design is achieved. The other two cycles connect the design cycle to the environment and to the scientific knowledge base. The relevance cycle first identifies an opportunity or an existing unsolved problem in the environment which then translates into a set of requirements that needs to be addressed by the designed artefact. An evaluation of the artefact then shows how well it meets the requirements to solve the stated problem. If it is shown to improve on existing solutions or attempts to solve the problem, the artefact is then fed back into the environment. However, the cycle repeats if the problem is only partially addressed or new problems emerge. The rigor cycle is the part that separates design science research from conventional design in a work environment (Hevner, 2007). During this cycle, the scientific knowledge base provides past knowledge to the project in order to ensure that the designs produced are research contributions and not routine designs and that appropriate and rigorous

methods are used for evaluation of the artefact. At the end of the cycle the newly developed knowledge on how to solve the identified problem is added to the knowledge base (e.g. by academic publication).

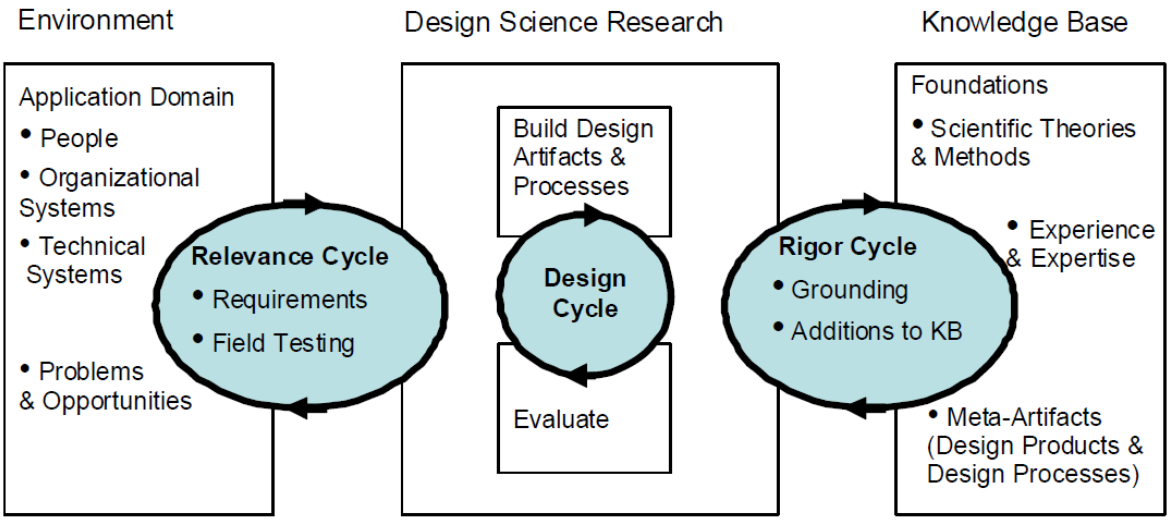

Figure 14: Design science research framework and cycles (Hevner, 2007)

In addition to the framework, Hevner et al. (2004) also outlines a set of guidelines for effective design science research. As illustrated in Figure 15 it consists of seven guidelines that highlight issues that should be addressed when performing this type of research.

When mapping the work in this thesis onto Hevners research framework, it can essentially be said that each appended paper represents a single design loop, encompassing all three cycles. Starting from the recognized opportunity (i.e. the use of BIMs for VR simulations have great potential) as well as related problems (e.g. BIMs provide a challenge to manage in real-time), an initial technology-based solution has been designed using input from the knowledge base. This solution has then been evaluated to show how well it solves the problem which essentially ends the design loop. Using the discoveries from the previous loop together with any changes in the environment and knowledge base as input, the problem formulation and evaluation criteria has then been updated and further addressed for each subsequent paper. Each design loop thus represents an incremental step towards realizing the final software application, which as of the last paper encompasses all the properties that have been identified as important in order support everyday use of VR during the building design process. The complete process will be described in more detail in Section 4, where each of the five appended papers is summarized, followed by a description of the final software application – BIMXplorer. As for the identification of required properties this will be discussed in the following subsection.

Table 1. Design-Science Research Guidelines	
Guideline	**Description**
Guideline 1: Design as an Artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6: Design as a Search Process	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Figure 15: Design science research guidelines (Hevner et al., 2004)

3.1 Requirements

One of the most important parts of a design science project is to formulate the acceptance criteria for the ultimate evaluation of the designed artefact. As already stated, the starting point for the work presented in this thesis was the recognized opportunity of combining BIM and VR in an efficient way, as well as the related performance problems already identified in the literature and observed in an actual practise (Johansson, 2010). Thus, already from the start, the ability to provide real-time rendering performance stood out as a fundamental requirement to satisfy. However, during the course of this work new opportunities and problems have emerged which have called for updates and changes to the list of requirements that the final artefact should be evaluated against. In order to give a better understanding of how the individual contributions relate to the relevance cycle, the final set of requirements is presented in advance (i.e. before the summary of the papers) below:

The system should provide real-time rendering performance. As a fundamental feature of any real-time visualization system this requirement needs no further motivation. Still, when considering the actual definition of what real-time is, the concept becomes somewhat fluid and has to be mapped to the actual use case. As compiled from the literature, this thesis defines the *satisfactory* as well as *optimal* level of frame rate for desktop VR as 30 and 60 Hz,

respectively. However, with the introduction of a new generation of HMDs, this requirement has then been transformed into a strong 90Hz demand.

The system should support architectural BIMs taken from real-world projects. Although posed as a requirement, this should be read more as a delimitation. As of today, a BIM-based design and construction project will, eventually, include several different BIMs, each on representing a single discipline (i.e. architectural, structural, etc.). However, the primary use-case that has been considered in this thesis is that of *architectural walkthroughs*. Given VR's ability to convey scale and overall experience of space it naturally lends itself especially useful in order to study architectural qualities (Westerdahl et al., 2006; Mobach, 2008). In addition, this use-case is related to many non-professional stakeholders (e.g. clients or building end-users) who naturally have less experience in interpreting traditional design documents, such as 2D drawings. This group of people therefore have much to benefit from the use of VR in terms of communication and enhanced understanding. Thus, as reflected by the type of models that has been evaluated in the papers, this requirement (or delimitation) has primarily been posed in order to clarify that the application of visualizing structural or mechanical, electrical and plumbing (MEP) BIMs *in isolation* has not been explicitly addressed. However, this does not necessarily mean that the system cannot be used for this use-case as well.

The system should accurately reflect the input dataset. When considering acceleration techniques it is possible to resort to solutions that favour interactivity at the expense of accuracy, such as contribution or drop culling. However, seen from a scientific perspective this introduces another level of complexity when it comes to verification and evaluation. As the visualization is no longer guaranteed to truly reflect the input dataset, these types of solutions also has to be evaluated based on how well they perform in terms of accuracy. In order to not introduce this level of complexity into the evaluation, this requirement has been posed.

The system should not rely on time-consuming pre-processing steps. A viable option when considering the isolated task of providing real-time rendering performance of large 3D datasets is to perform a pre-computation step that will allow the final visualization session to run at high frame rates. However, such a solution will inherently pose itself as a potential obstacle in that an additional process is needed before any visualization session can be realized. This requirement is thus based on the simple logic that if we can omit any additional process, it will make the use of the technology more accessible and therefore easier to integrate as an everyday tool into real practise.

The system should support a wide range of users. A typical building project will involve a number of different stakeholders with different backgrounds and expertise. When considering a successful integration of VR within this setting, it is thus highly desirable that the medium can be easy to use and, ultimately, controllable by anyone.

4 Summary of the papers

4.1 Paper I

Efficient Real-Time Rendering of Building Information Models

Background and purpose
Due to a large number of individual objects and high geometric complexity, typical BIMs are not easily rendered in real-time. However, compared to a general 3D-model, a BIM defines not only geometrical data, but also information regarding spatial relations and semantics. The idea behind Paper I was to investigate if it's possible to take advantage of the additional data in order to accelerate real-time rendering.

Method
By extracting spaces (cells) and openings (portals) from a BIM we can automatically create a cells-and-portals partitioning. Using this data structure, the rendering is accelerated by rejecting objects that are not in, or can be seen from, the specific room that the viewer is currently in. To make this algorithm efficient also in outdoor cases, additional mechanisms had to be developed. These included a technique that utilizes frame-to-frame coherence and a procedure to efficiently reject non-visible exterior walls. The proposed technique was tested on two fairly large BIMs and evaluated against traditional view-frustum culling.

Results
Compared to traditional view-frustum culling, the new technique was often 10 times faster, for both exterior and interior view points, essentially making real-time rendering of large BIMs possible.

4.2 Paper II

Real-Time Visualization of Building Information Models (BIM)

Background and purpose
Paper I showed the benefit of rejecting hidden objects (i.e. cull away) with respect to real-time performance. However, the technique developed in Paper I relied heavily on specific BIM-data (i.e. spaces) being present in order to function properly. As evident from many BIMs received from real-world projects, the required data is not always present. The idea behind Paper II was to evaluate and analyze commercial BIM viewers in terms of real-time rendering performance and to evaluate more general acceleration techniques (i.e. that do not rely on specific BIM-data).

Method
Four commercial BIM viewers were in-depth analyzed in terms of acceleration techniques and real-time rendering performance. In addition, a general occlusion culling algorithm, CHC++, was implemented in a prototype BIM viewer and further refined. All viewers, including the prototype, were evaluated using four different BIMs taken from real-world projects.

Results
All four commercial viewers shared limitations in their ability to handle large BIMs interactively. The prototype viewer had no such problems. Consequently, the CHC++ algorithm was found to be a suitable acceleration technique for efficient real-time rendering of BIMs.

4.3 Paper III

Integrating Occlusion Culling and Hardware Instancing for Efficient Real-Time Rendering of Building Information Models

Background and purpose
In Paper II, occlusion culling, and more specifically, CHC++ were found to provide a suitable acceleration technique for typical BIMs. However, for viewpoints when many objects are, in fact, visible, occlusion culling alone may not always be able to guarantee sufficiently high performance. Based on the observation that typical BIMs contain many replicated objects, the idea behind Paper III was to evaluate the combination of occlusion culling and hardware-accelerated geometry instancing – a feature of modern GPUs that allow replicated geometry to be rendered very efficiently.

Method
By taking advantage of temporal coherence together with the development of a lightweight data transfer approach, occlusion culling could be performed at the object level at the same time as visible, replicated geometry can be efficiently rendered using hardware-accelerated geometry instancing. The combination of techniques was evaluated on four different BIMs taken from real-world projects.

Results
Compared to only using occlusion culling the new technique were shown to offer additional speed-ups of 1.25x-1.7x in viewpoints that represent the worst case scenarios when only occlusion culling is utilized.

4.4 Paper IV

From BIM to VR – Integrating immersive visualizations in the current design process

Background and purpose
When considering the use of immersive visualization technology within the AEC field, the introduction of consumer-directed, low-cost-high-performance HMDs devices, such as the Oculus Rift, has opened up new possibilities. Compared to previous solutions, such as CAVEs and PowerWalls, many inherent barriers, including investment costs and limited accessibility can now be broken. However, the performance demands required by stereo rendering are still difficult to satisfy without additional acceleration techniques. The idea behind paper IV was to investigate the acceleration techniques proposed in Paper II and III in a stereo setting as well as setup and evaluate a system that allowed immersive visualizations to become a natural and integrated part of the current design process.

Method
The rendering engine developed in Paper II & III was implemented as a plugin in Revit, thereby offering direct visualization from a BIM authoring environment. To support a wide range of users (i.e. from gamers to construction site workers) a simple navigation interface was developed by means of a so-called PowerPoint remote. The proposed system was tested on a BIM taken from a real world project and evaluated from three different perspectives - rendering performance, navigation interface and the ability to support fast design iterations.

Results
Compared to current immersive solutions (i.e. CAVEs and PowerWalls) the proposed system is non-expensive, portable (i.e. accessible) and has very good BIM support. Furthermore, regarding rendering performance, navigation interface and the ability to support fast design iterations, it has all the needed properties to function well in practice.

4.5 Paper V

Efficient Stereoscopic Rendering of Building Information Models (BIM)

Background and purpose
Stereo rendering is traditionally done by performing two individual and serial rendering passes – one for the left eye and one for the right eye. This was the method used in Paper IV and compared to monoscopic rendering, this setup essentially increase the number of draw calls and rasterized triangles by a factor of two. One way to remove the requirement of a second pass is by taking advantage of the geometry shader in order to duplicate and present the geometry for the left and right eye. However, even if this reduces the number of draw calls, the geometry shader typically introduces significant overhead on the GPU side. The idea

behind paper V was to explore the possibilities of using hardware-accelerated geometry instancing in order to provide a single-pass stereoscopic rendering in a split-screen stereo setup (i.e. as found in the Oculus Rift)

Method

With the instancing capabilities of modern GPUs it is possible to produce multiple output primitives from a single input, without introducing the geometry shader. As such, it becomes suitable for producing both the left and right eye view of the scene within a single rendering pass. However, the main difficulty with this approach is that current graphics API does not support multiple viewport output from the vertex shader. In the proposed technique this is solved by performing a screen-space transformation of the geometry, together with user-defined clipping planes. In addition to reduce the number of draw calls the proposed technique were shown to integrate very well with occlusion culling based on hardware-accelerated occlusion queries (i.e. as used in Paper II, III and IV). With a single depth buffer used for both the left and right eye, only a single occlusion query is ever needed per visibility test, effectively reducing the number of occlusion tests by a factor of two compared to the traditional two-pass stereo rendering technique. Furthermore, with little modifications, the new stereo instancing technique could be extended to also support the geometry instancing technique developed in Paper III.

Results

The new stereo instancing technique is very well suited for integration with occlusion query-based occlusion culling as well as conventional geometry instancing and has been shown to outperform traditional two-pass stereo rendering approach, geometry shader-based stereo duplication, as well as brute-force stereo rendering of typical BIMs on recent hardware.

5 BIMXplorer v1.0

BIMXplorer represents the final system (i.e. artefact) that is the result of the research presented in this thesis (Figure 16). In essence, this is a working software application that allows large and complex BIMs to be directly visualized in real-time, either through a traditional desktop interface (i.e. screen, mouse and a keyboard) or by using a modern HMDs such as the Oculus Rift. Many of the systems features, such as the navigation interface and the integration as a plugin within Autodesk Revit are described in detail in Paper IV. It also incorporates the acceleration techniques developed and presented in Paper II, III and V. In addition, BIMXplorer has support to directly load IFC-files through the xBIM (eXtensible Building Information Modelling) software development toolkit. Also, in order to allow for a more user-friendly navigation the system takes advantage of the PhysX SDK to support collision detection. Upon model loading, collision meshes can be automatically generated which prevents user from navigating "through" objects, such as walls and floors, in a similar fashion as modern 3D games.

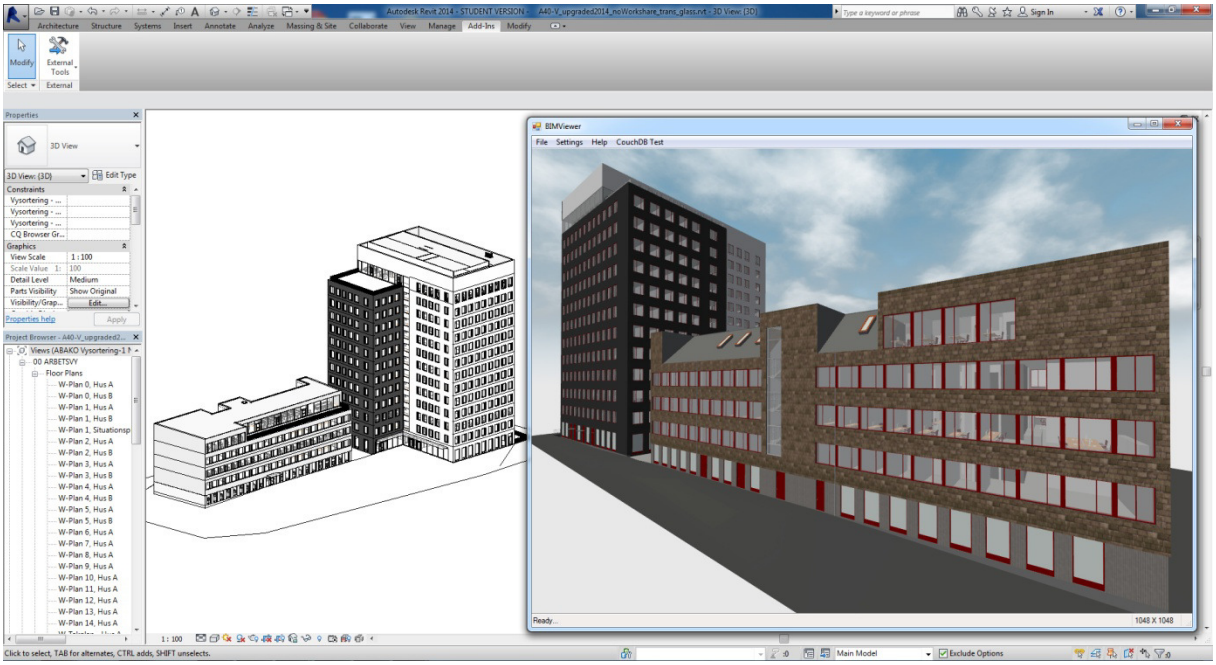

Figure 16: BIMXplorer interface as a plugin in Autodesk Revit.

To improve the visual quality BIMXplorer takes advantage of a technique known as Screen-Space Ambient Occlusion (SSAO), which calculates how exposed each point in a 3D scene is to ambient lighting (McGuire et al., 2012). Compared to a constant ambient term, this gives much better depth perception and provides clues on how objects relate to each other as seen in Figure 17.

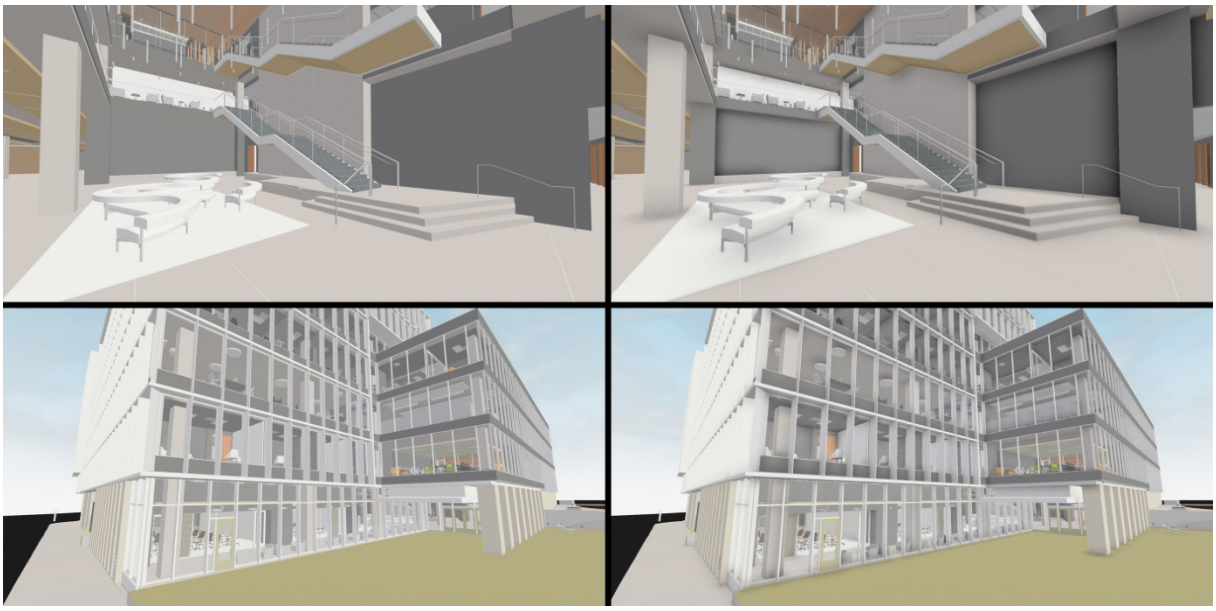

Figure 17: A BIM containing approx. 40,000,000 triangles rendered in real-time in BIMXplorer with constant ambient term (left) and SSAO (right) (Revit model courtesy of Jason Halaby, WRNS Studio).

Although not yet publicly available, BIMXplorer has already been used during several courses at Chalmers University of Technology. These courses involve the design of a suburban area as well as the design of a new university campus area featuring several new buildings created in Autodesk Revit. Throughout these projects BIMXplorer has been used as an integrated visualization tool in order evaluate different designs and to communicate ideas among team members. At the end of these courses each team then presents their proposal by performing live walkthroughs during a final seminar (Figure 18). Being that a diverse set of BIMs have been created during these projects the courses have served as a form of continuous beta testing of the software.

In addition BIMXplorer has been in active use for over a year at NCC Construction Sweden (Jörnebrant and Tomsa, 2015; Roupé et al., 2016; Brännström and Ljusteräng, 2016) and has also been used during several projects at WRNS Studio, an architecture and planning firm located in San Francisco, California. As such, it has been proven useful in actual practise.

Figure 18: Students at Chalmers University of Technology presenting design proposals using live walkthroughs in BIMXplorer.

6 Discussion

In this section the results from the five appended papers are discussed in relation to the four research questions posed in Section 1 as well as the requirements outlined in Section 4.

6.1 Current state of BIM visualization

Going back to the initial problem statement, previous literature had already recognized the challenges of using BIMs for the purpose of real-time visualization. Still, many questions remained, such as the magnitude of the problems and how these were related to hardware and model complexity. However, based on the results from the five appended papers we can now conclude that this is, in fact, a real issue. In essence, all papers reveal this in that additional acceleration techniques – i.e. beyond that of conventional view-frustum culling or brute-force rendering – are needed in order to provide a suitable level of interactivity when rendering large BIMs taken from real-world projects. Furthermore, the in-depth analysis in Paper II shows that existing BIM-viewers are currently unable to address the problem in a satisfying manner. Also, given the huge spread in terms of rendering performance these problems can no longer only be discussed in relation to model complexity and lack of efficient hardware, but needs to include *software capacity* as an additional variable. For instance, with BIMSight any model may be seen as large and complex.

Still, it is also very important to acknowledge that several BIM-viewers have techniques to guarantee a certain level of interactivity by sacrificing correctness (e.g. drop culling). However, as identified in Paper II, the use of drop culling does not only produce an incorrect image but also gives very obvious “popping” artefacts as the priority of which objects to render constantly changes. Although no formal evaluation has been conducted as to what degree, if any, that this influences experience and usage negatively, these ”popping” artefacts has been reported as very distracting in previous literature (Willmott et al., 2001; Giegl and Wimmer, 2007). As such, this thesis argues that non-conservative acceleration techniques such as drop culling are not an adequate solution to the interactivity problem.

Furthermore, when considering the actual interactivity problem as well as the corresponding requirements on frame rate, this is something that has changed during the course of this work. Although 60 Hz was initially considered an optimal level of interactivity this is no longer the case for all display systems. With the introduction of a new generation of HMDs, such as the Oculus Rift, 90Hz is now considered the absolute minimum (Hasan and Yu, 2015). Due to very strong connection to the tracker system, any update rate below 90 Hz will provide such visual artefacts that the system essentially becomes useless. As such, this requirement is different compared to desktop VR, where lower frame rates may still give an ok experience, although with less fidelity (Claypool, 2007; Rubino and Power, 2008). When also taking into account that these devices require the 3D scene to be rendered twice it should be no doubt that existing systems or techniques will not be able support HMDs in their current state.

In this context it is also important to highlight the recent trend within the AEC industry to use game engines for the purpose of real-time visualizations. Although game engines typically share the same performance problems as dedicated BIM-viewer they often have in-built tools or overall support to optimize or prepare 3D models for real-time performance. As such, it is then possible to prepare or optimize a BIM for the purpose of real-time visualization. However, this process typically involves a lot of labour-intensive, manual work making it less suitable in practise (Shi et al., 2015; Merschbrock et al., 2016).

Recent times have also seen the introduction of a couple of commercial applications, such as Revizto and Enscape, which mimics the behaviour of BIMXplorer in that they provide a plugin interface to a BIM authoring environment, e.g. Revit. These applications have not been thoroughly evaluated as part of this thesis. However, as of an initial investigation it is clear that these systems primarily rely on brute-force performance, and therefore will have difficulties to scale up to the large type of BIMs that have been evaluated in this thesis.

In relation to the above discussion it is also relevant to pose the question of *whether or not we truly need to be able to visualize large BIMs in the first place*. A simple solution to the interactivity problem would instead be to only visualize sub-sets of a complete building model, which is already a commonly used approach (Dubler et al., 2010; Shi et al., 2015). Assuming a reasonably powerful hardware it is very likely that the interactivity demands can be fulfilled as long as we restrict the visualization session to a certain region of a building. However, being forced to work with a sub-set of a complete model for the sake of performance is, by all means, a restriction. Although this thesis have only addressed actual user studies to a small degree (i.e. Paper IV), there are other recent publications that have conducted user studies with this particular system (Jörnebrant and Tomsa, 2015; Roupé et al., 2016; Brännström and Ljusteräng, 2016). Here, the ability to interactively navigate a *complete* building have been identified as a major feature in that it allows stakeholders to inspect the building as a whole and study and get an understanding of internal logistics and communication, e.g. ”how many doors needs to be passed in order to get from the entrance to the lab area?”. Also, the ability to have everything in a single model has been expressed as an important aspect in order to truly have a virtual representation of the actual building (Roupé et al., 2016). Nevertheless, the techniques and, ultimately, the final system – BIMXplorer – developed within the scope of this thesis naturally supports the ability to only show sub-set of a complete model.

Connecting to all of this is of course on what premises existing solutions as well as the developed techniques have been evaluated. In other words, are the models used relevant test cases? Except for the two models used in Paper I, all models have been received from real-world projects. Ranging from apartment and office buildings to more open ones, such as a library or hotel, they represent the wide variety of building types that can be encountered in practise. Also, these models have not primarily been chosen to showcase the developed technique, but instead been used as *drivers* for the development of new techniques. To better simulate a worst-case scenario two of the models (the hotel and the office building) have been

”completed” in that furniture and other interior equipment have been placed at all levels (i.e. floors) in the building(s). Among Swedish architects the general strategy today appears to be to only add interior equipment at certain levels in order to not make the models too large and complex. This, however, appears not to be the case in the US (Maller, 2011) and in order to better mimic an international situation additional levels have therefore been ”completed”.

6.2 Efficient real-time rendering of BIMs

Regarding ways to accelerate real-time rendering of BIMs two main ideas were initially considered: (1) to take advantage of the natural occlusion present in typical buildings and (2) to take advantage of the additional information (e.g. metadata) contained in a BIM. Both these ideas were explored and combined in Paper I, where information about spaces and openings was used to take advantage of portal culling (Luebke and Georges, 1995) *without* the need for any manual interaction or offline pre-processing. Although portal culling has been primarily used for indoor scenes it was shown to be efficient also for exterior viewpoints if additional techniques were added. However, even if the developed technique solved the performance problem of rendering large BIMs it turned out to be less suitable in practise. Because of the strong requirement that spaces and openings has to be correctly defined in the model, it would only be fully functional in situations where complete BIMs (i.e. complete in the sense that all spaces and openings are present) can be guaranteed. Although openings are typically generated automatically in a BIM authoring system as soon as a door or window is placed in a wall, the same is not true for spaces. Instead, spaces have to be manually added to the model as any other object - a process typically done during later stages of the design. Consequently, to support real-time visualization during *all* stages of the design process, it becomes difficult to have the main acceleration technique depend on a specific type of object - or specific data - being present in the model. Instead, a much more versatile solution could be found simply by looking at the *general characteristics* of a typical BIM – high level of geometry occlusion. As identified in Paper II, CHC++ (Mattausch et al., 2008), a state-of-the art occlusion culling algorithm, turned out to be a very good fit for real-time rendering of BIMs, essentially outperforming all existing BIM viewers on the market. Although perhaps seen as an obvious choice in retrospect, a general occlusion culling system is by far not guaranteed to always be a suitable solution. As seen from the performance results from the Navisworks occlusion culling system, it may very well perform worse than simple view frustum culling for many viewpoints. Nevertheless, CHC++ turned out to be very efficient, not only in interior, but also for exterior viewpoints, making it a very good starting point for further improvements. Still, even if the rejection of hidden objects provide a huge speed-up, there are typically several viewpoints that contain many objects that are, in fact, visible. Based on the simple logic that there is a high probability that such viewpoint contain many replicated objects, e.g. imagine all the windows seen in a hotel facade , Paper III then explored the possibility of combining occlusion culling with hardware-accelerated instancing – a feature on modern GPUs to render replicated objects efficiently. Although the use of instancing as well as occlusion culling has many examples in the literature, no known efforts had been previously made to combine these two techniques.

As seen from the results in Paper III, the idea of providing instanced rendering of un-occluded replicated objects, turned out to be a very good complement to the original CHC++ algorithm. As with the portal culling approach, the use of instancing requires specific data being present in the BIM. However, this information forms an integral part of any modern BIM authoring system – once an object of a certain *type* is added to the model it essentially becomes an *instance* of that *type*, meaning that the required data becomes available from the model. Also, the main difference compared to the portal culling approach is that the instancing technique is implemented in a way that takes advantage of replicated objects if present, but simply degrades to a standard CHC++ solution if there are no visible replicated objects. In other words, the technique does not negatively affect performance if no replicated objects are found visible.

However, with the introduction of a new type of consumer-directed HMDs, the rendering performance requirements changed. Not only became frame-rate requirements higher, but also the requirement of producing two different views every frame. As identified in Paper IV, the developed techniques were suitable also for stereo rendering. Even if two rendering passes was now required (i.e. one for each eye) the current acceleration technique was still efficient enough to support real-time frame rates. Nevertheless, as the performance demands, in terms of resolution and frame rate, became higher for each version of the HMDs it became clear that additional acceleration techniques were eventually needed. Fortunately, the concept of hardware instancing turned out to offer a solution also in this case. As identified in Paper II, the occlusion culling system was mainly CPU-bound on high end systems due to a large number of draw-calls. With a traditional stereo setup, this amount essentially doubled. Although the instancing technique (Paper III, IV) made the situation better, the amount of draw-calls still needed to be reduced in order to reach the required frame rates. The successful use of instancing then naturally sparked the idea of also using it for stereo rendering. After all, stereo rendering is essentially a process of rendering almost two replicates of the complete 3D-scene. Ultimately, this idea then became *stereo instancing* – an efficient single-pass stereo rendering technique. As seen from the results in Paper V, this also made a perfect fit for the occlusion culling system, in that not only draw-calls, but also occlusion tests became reduced by a factor of two, i.e. as compared to a traditional two-pass stereo setup.

As it would turn out, however, the concept and potential of stereo instancing had independently been recognized by developers from the game development community (Wilson, 2015). At the time of formal publication of Paper V, the stereo instancing technique was already considered a best practise within the game development industry (Vlachos, 2015). Still, the paper contributes the first detailed description of the technique and a thorough performance evaluation. For the purpose of efficiently rendering BIMs it is also the actual *combination* of the different techniques – i.e. occlusion culling, stereo instancing, conventional instancing and batching of walls – that is important in order to provide the required frame rates.

So, in perspective, the initial idea of taking advantage of the natural occlusion present in buildings as well as metadata actually turned out to be a successful approach, albeit in a different form.

Nevertheless, in this context it may also be relevant to further discuss the brute-force approach. Given that this alternative was surprisingly close to deliver sufficient frame rates for the Hotel model (Paper V), it does seem like a viable option in the near future. However, although this was true for the Hotel model, this was not nearly the case for the Student house or Office buildings. With the Hotel model only having roughly half the amount of triangles compared to the Office building, this puts things in perspective. Another aspect to consider is that with the brute-force approach essentially all of the GPUs power will be used simply to rasterize triangles. That is, even if the brute-force performance of future GPUs will be able to manage all of the dataset within an acceptable time frame, there will be less processing power left to do better shading, like SSAO, for instance. As such, it will always make sense to use additional acceleration techniques in order to better utilize the GPUs resources.

6.3 Integration of VR within the AEC field

In order for VR to become a natural and integrated tool within the design process there is more to consider than just the ability to visualize large BIMs in real-time. For instance, if time-consuming pre-processing steps or manual interactions are needed in order to support real-time performance it is highly likely that this will affect a natural integration negatively (Liu at al., 2014). However, following a design science approach, the techniques presented as part of this thesis have all been developed with the requirements and the integrational aspects in mind. Going back to the requirements that the final artefact should be evaluated against (Section 4), these can essentially be summarized as "*Being able to support instant/direct, artifact-free and user-friendly real-time VR walkthroughs of architectural BIMs taken from real-world projects on systems that exist today*". When considering real-time performance as well as the actual definition of what real-time is the previous subsections have already discussed this. It has been shown that, when combined, the developed techniques and algorithms allow architectural BIMs taken from real-world projects to be rendered, in stereo, at more than 90 frames per second on off-the-shelf laptops. Moreover, contrary to the approach taken by several existing BIM-viewer, the developed techniques do not introduce any visual artefacts, e.g. omitting objects that should be visible.

When considering the actual integration of VR within the design process, Paper IV introduced the idea of using the rendering engine as a plugin in a BIM authoring application instead of a stand-alone application. Together with the use of a new type of consumer-directed HMD, the Oculus Rift, it essentially offers a "one button click" connection between the design environment, i.e. the BIM authoring software, and immersive VR. Compared to previous immersive solutions, like CAVEs and Powerwalls, this opens up a number of possibilities within the design process. With a low cost, portable solution that directly support BIMs it is possible to take advantage of immersive visualizations anywhere at any time during the

design process. Although the very existence of the Oculus and HTC Vive made much of this possible, it must be highlighted that the techniques developed in this thesis are very important in order for the VR-technology to be used as an everyday tool during the design. As already discussed, the real-time rendering strategy used by any of the tested BIM viewers will not support modern HMDs. That is, even *if* stereo VR support was formally added, none of the tested viewers would be able to provide the required frame rates without introducing severe visual artefacts. Nevertheless, if we also include the use of game engines several examples can be found where modern HMDs has been used to provide immersive visualizations of BIMs. Still, as discussed, the use of game engines for the purpose of BIM visualizations currently requires additional optimization and preparation time. Even if an efficient pipeline or work procedure has been established (Halaby, 2015), this process can easily range from hours to days. In comparison, the technical contributions presented in this thesis cut this process down to, on average, 60 seconds for complete architectural BIMs. Not only does this make the use of immersive VR highly accessible in the first place, but it also supports an *active* use during fast design iterations.

Furthermore, as evaluated in Paper IV, the simple navigation interface makes it suitable also for inexperienced users. Especially when considering building end-users this becomes an important property. As of today user involvement is mostly restricted to reviewing traditional 2D-plans which may be difficult to fully interpret for all the different stakeholders in a project. With the ability for any type of user to freely navigate proposed designs from an immersive, first person perspective a much better understanding can often be achieved (Heydarian at al., 2015).

To what extent the technical contributions and, ultimately, the final system will pave the way for a more integrated use of VR during the design process remains somewhat an unanswered question. As evaluated against the technical requirements it has all the properties needed in order to function well in practise. Still, except for the evaluation of the navigation interface, no formal/documented user-studies have been conducted within the scope of this thesis. However, early versions of the system, as well as the final system have already been used in several other studies, where its suitability as a tool to enhance understanding and communication has been highlighted (Kreutzberg, 2015; Roupé et al., 2016; Hermund and Klint, 2016). It has also been used during several courses at Chalmers University of Technology. In addition, it has been in active use in several construction projects for over a year at NCC Construction Sweden (Jörnebrant and Tomsa, 2015; Roupé et al., 2016; Brännström and Ljusteräng, 2016). Simple logic tells us that this would not have been the case if the system was found unfit for use in real-world projects.

7 Conclusions and future work

The research presented in this thesis has contributed to a better understanding regarding the complexity and challenges involved in visualizing large and detailed BIMs in real-time. It has been shown that additional acceleration techniques are, indeed, needed in order to solve the interactivity problem and that existing BIM-viewers are currently unable to address this issue in a satisfying manner – this at the same time as a new generation of VR hardware calls for even higher performance demands.

In order to address the current situation this thesis contributes with the design and evaluation of a new software application that provides a "one-button-click" solution from BIM to VR. Following a design science research approach this application has been developed in order to fulfil a set of requirements that has been identified as important in order for VR and real-time visualization to become an everyday used tool for design and communication during the building design process. Along that path, three new technical solutions have been developed:

- An efficient cells- and portals culling system that is automatically realized from BIM-data.
- An efficient approach for integrating occlusion culling and hardware-accelerated geometry instancing.
- An efficient single-pass stereo rendering technique based on hardware-accelerated geometry instancing.

The final system – BIMXplorer – has been evaluated using several BIMs received from real-world projects. Regarding rendering performance, navigation interface and the ability to support fast design iterations, it has been shown to have all the needed properties in order to function well in practice. To some extent this can also be considered formally validated, as the system is already in active use within both industry and education.

For future work there are several different directions possible. For instance, when considering ways to improve rendering performance there is still much work to be done within the following areas:

> **Spatial hierarchy** The culling efficiency is inherently dependent on how well the spatial hierarchy can represent the 3D scene with respect to occluding and enclosing objects, such as walls and floors. As of now, the bounding volume hierarchy (BHV) is constructed based on surface area and do not take into account any natural containment, e.g. due to occluding walls or floors. It would be interesting to see how the spatial hierarchy could be improved by taking advantage of any space objects – e.g. *IfcSpace* or *Rooms* in Revit – present in the model. Interior objects, e.g. furniture, could then be clustered per space object before feeding them to the BVH construction procedure.

Another way to improve the spatial hierarchy and, hence, the culling efficiency, would be to take advantage of oriented bounding boxes (OBB) instead of axis-aligned ones (AABB). As OBBs typically provides a much "tighter" fit around objects, the number of visibility tests that provides false negatives would decrease. However, instead of creating "true" oriented bounding boxes, it would be interesting to explore the concept of *building-oriented bounding boxes*. With the interior and exterior walls of most buildings following a local, orthonormal coordinate frame, it makes much sense to then orient all of the 3D scenes bounding boxes accordingly. This would then prevent many situations where object bounds intersect walls and becomes (falsely) detected as visible from the other side. Even if the local coordinate frame is not explicitly known, it should be straightforward to calculate it based on the normals of the wall geometry weighted against its surface area for all the walls contained in a BIM.

Occlusion culling The biggest disadvantage of using occlusion queries is the latency introduced by waiting for the result of the queries to return to the CPU-side of the application. CHC++ hides this latency fairly efficient by rendering previously visible objects during the wait-time, but it is still not an optimal solution. The readback of data from the GPU is required mainly because of the GPUs inability to feed itself with draw calls (Rákos, 2012). However, with recent extensions to the OpenGL API this restriction has been relaxed and it has been shown possible to implement an occlusion culling system mainly on the GPU (Boudier and Kubisch, 2015). Further exploring these features thus represents an obvious direction for future research.

Level-of-detail (LOD) As BIMs become even more detailed and several of them are to be visualized together, the concept of LOD needs to be considered in order to reduce the sheer amount of triangles that has to be rendered. Ultimately, in order to provide a scalable solution, this has to be done both locally, i.e. per object, as well as globally, i.e. for a whole facade or a whole building. However, neither of these tasks is trivial to address automatically – especially if it also has to be done in a short amount of time (Luebke et al., 2002; Arroyo Ohori, 2016). When considering per-object simplification it makes sense to initially focus on furniture and other interior equipment, as these types of objects often contain an excessive amount of triangles. A good starting point would probably be to first explore and evaluate the results that can be achieved by simplifying individual interior objects using *edge collapse* (Luebke et al., 2002; Melax, 1998).

Going beyond that of acceleration techniques, there is also much room for further research and development regarding suitable interaction interfaces for different types of applications – e.g. design review or information extraction on-site – as well as user-studies related to perception and spatial understanding with modern HMDs. Moreover, when looking at these things in a larger context, it would be interesting to see to what degree an integrated use of VR will affect the design process and, ultimately, the buildings that are a result of it.

References

Akenine-Möller, T., Haines, E., & Hoffman, N. (2008). Real-time rendering. CRC Press.

Arnaud, R., & Barnes, M. C. (2006). COLLADA: sailing the gulf of 3D digital content creation. CRC Press.

Arroyo Ohori, K. (2016) Higher-dimensional modelling of geographic information. PhD-thesis, Delft University of Technology. Delft, Netherlands.

Barfield, W., Baird, K. M., & Bjorneseth, O. J. (1998). Presence in virtual environments as a function of type of input device and display update rate. Displays, 19(2), 91-98.

Bartz, D., Meißner, M., & Hüttner, T. (1998). Extending graphics hardware for occlusion queries in OpenGL. In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware (pp. 97-ff). ACM.

Bouchlaghem, D., Shang, H., Whyte, J., & Ganah, A. (2005). Visualisation in architecture, engineering and construction (AEC). Automation in construction, 14(3), 287-295.

Boudier, P., Kubisch, C. (2015), GPU-driven large scene rendering, Presentation at the GPU Technology Conference (GTC 2015), San Jose, CA, USA.

Brännström, E. och Ljusteräng, F. (2016) VR och VR-glasögon inom byggbranschen. Chalmers University of Technology (Examensarbete - Institutionen för bygg- och miljöteknik, Chalmers tekniska högskola, nr: BOMX03-16-08).

buildingSMART, I. F. C. (2007). 2x Edition 3 Technical Corrigendum 1. International Alliance for Interoperability, URL: http://www. buildingsmart-tech. org/ifc/IFC2x3/TC1/html/index. htm.

Bullinger, H. J., Bauer, W., Wenzel, G., & Blach, R. (2010). Towards user centred design (UCD) in architecture based on immersive virtual environments. Computers in Industry, 61(4), 372-379.

Burdea, G. C., & Coiffet, P. (2003). Virtual reality technology (Vol. 1). John Wiley & Sons.

Cohen-Or, D., Chrysanthou, Y. L., Silva, C. T., & Durand, F. (2003). A survey of visibility for walkthrough applications. IEEE Transactions on Visualization and Computer Graphics, 9(3), 412-431.

Claypool, K. T., & Claypool, M. (2007). On frame rate and player performance in first person shooter games. Multimedia systems, 13(1), 3-17.

Cruz-Neira, C., Sandin, D. J., DeFanti, T. A., Kenyon, R. V., & Hart, J. C. (1992). The CAVE: audio visual experience automatic virtual environment. Communications of the ACM, 35(6), 64-73.

Dalton, B. and Parfitt, M. (2013). Immersive visualization of building information models. Design Innovation Research Center Working Paper, 6(1.0).

Dubler, C. R., Messner, J., & Anumba, C. J. (2010). Using lean theory to identify waste associated with information exchanges on a building project. In Proceedings Construction Research Congress/ASCE Conference.

Dudash, B. (2007). Animated crowd rendering. GPU Gems, 3, 39-52.

Dvorak, J., Hamata, V., Skacilik, J., & Benes, B. (2005), Boosting up architectural design education with virtual reality, CEMVR, 5, 95-200.

Dörner, R., Lok, B., & Broll, W. (2011). Social Gaming and Learning Applications: A Driving Force for the Future of Virtual and Augmented Reality?. In Virtual Realities (pp. 51-76). Springer Vienna.

Eastman, C. M. (1999). Building product models: computer environments, supporting design and construction. CRC press.

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

Funkhouser, T. A., & Séquin, C. H. (1993). Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In Proceedings of the 20th annual conference on Computer graphics and interactive techniques (pp. 247-254). ACM.

Garland, M., & Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques (pp. 209-216). ACM Press/Addison-Wesley Publishing Co.

Giegl, M., & Wimmer, M. (2007). Unpopping: Solving the Image-Space Blend Problem for Smooth Discrete LOD Transitions. In Computer Graphics Forum (Vol. 26, No. 1, pp. 46-49). Blackwell Publishing Ltd.

Greenwood, D., Horne, M., Thompson, E., Allwood, C. M., Wernemyr, C., Westerdahl, B. (2008), Strategic Perspectives on the Use of Virtual Reality within the Building Industries of Four Countries, Architectural Engineering and Design Management, Vol. 4, No. 2, pp. 85-98.

Gregor, S., & Hevner, A. R. (2013). Positioning and presenting design science research for maximum impact. MIS quarterly, 37(2), 337-355.

Göttig, R., Newton, J., & Kaufmann, S. (2004). A Comparison of 3D Visualization Technologies and their User Interfaces with Data Specific to Architecture. In Recent Advances in Design and Decision Support Systems in Architecture and Urban Planning (pp. 99-111). Springer Netherlands.

Halaby, J. (2015), Simulating Presence: BIM to Virtual Reality, Presentation at BAYA: Emerging Technologies in Architecture, 2015, San Francisco, CA.

Hasan, M. S., & Yu, H. (2015). Innovative developments in HCI and future trends. In 21st IEEE, International Conference on Automation & Computing (ICAC 2015), University of Strathclyde, Glasgow, UK.

Hermund, A., Klint, L. (2016) VIRTUAL AND PHYSICAL ARCHITECTURAL ATMOSPHERE. In Proceedings of the International Conference on Architecture, Landscape and Built Environment (ICALBE 2016), Kuala Lumpur, Malaysia, pp. 3-4

Herwig, A., & Paar, P. (2002). Game engines: tools for landscape visualization and planning. Trends in GIS and virtualization in environmental planning and design, 161, 172.

Hettinger, L. and Riccio, G. (1992), Visually induced motion sickness in virtual environments, Presence 1(3), pp. 306–310.

Hevner, A. R. (2007). A three cycle view of design science research. Scandinavian journal of information systems, 19(2), 87-92.

Hevner, A., March, S., Park, J., and Ram, S. (2004), Design Science in Information Systems Research, MIS Quarterly (28:1), pp. 75-105.

Heydarian, A., Carneiro, J. P., Gerber, D., Becerik-Gerber, B., Hayes, T., & 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, 116-126.

Hillaire, S. (2012). Improving Performance by Reducing Calls to the Driver. OpenGL Insights, A K Peters/CRC Press, 353-364.

Johannesson, P., & Perjons, E. (2014). An introduction to design science. Springer.

Johansson, M. (2010). Towards a Framework for Efficient Use of Virtual Reality in Urban Planning and Building Design. Licentiate thesis, Chalmers University of Technology, Sweden

Jongeling, R., Asp, M., Thall, D., Jakobsson, P., & Olofsson, T. (2007). VIPP: Visualization in Design and Construction. Luleå: Luleå tekniska universitet. (Technical report / Luleå University of Technology; Nr 2007:07).

Jörnebrant, F. and Tomsa, P. (2015) The BIM Head Mounted Display as an integrative part of project phases A case study of applying new technologies in a construction project. Göteborg : Chalmers University of Technology (Examensarbete - Institutionen för bygg- och miljöteknik, Chalmers tekniska högskola, nr: 2015:89).

Kasik, D. J., Troy, J. J., Amorosi, S. R., Murray, M. O., & Swamy, S. N. (2002). Evaluating graphics displays for complex 3d models. IEEE Computer Graphics and Applications, 22(3), 56-64.

Kjellin, A. (2008), Visualizing Dynamics –The Perception of Spatiotemporal Data in 2D and 3D. Uppsala: Acta Universitatis Upsaliensis; Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Social Sciences.

Kjems, E. (2005), VR applications in an architectural competition: Case: House of Music in Aalborg. /. I: Realitat Virtual a l'Arquitectura i la Construcció : Taller 2. Barcelona : Khora ll, pp. 47-58.

Kreutzberg, A. (2015). Conveying Architectural Form and Space with Virtual Reality. Proceedings of the 33rd eCAADe Conference - Volume 1, Vienna University of Technology, Vienna, Austria, 16-18 September 2015, pp. 117-124

Kuechler, W., & Vaishnavi, V. (2008). The emergence of design research in information systems in North America. Journal of Design Research, 7(1), 1–16.

Liu, Y., Lather, J., & Messner, J. (2014). Virtual Reality to Support the Integrated Design Process: A Retrofit Case Study. in Computing in Civil and Building Engineering ASCE.

Luebke, D., & Georges, C. (1995). Portals and mirrors: Simple, fast evaluation of potentially visible sets. In Proceedings of the 1995 symposium on Interactive 3D graphics (pp. 105-ff). ACM.

Luebke, D., Reddy, M., Cohen, J., Varshney, A., & Watson, B., Huebner. R. (2002) Level of detail for 3D graphics, Elsevier.

Maller, A. (2011), Autodesk Revit Links, Groups, and Documentation: How to Make It Really Work!, Presentation at Autodesk University 2011, Las Vegas, Nevada.

March, S. and Smith, G. (1995), Design and natural science research on information technology, Decision Support Systems, Vol. 15, No. 4, pp. 251-266.

Mattausch, O., Bittner, J., & Wimmer, M. (2008). Chc++: Coherent hierarchical culling revisited. In Computer Graphics Forum (Vol. 27, No. 2, pp. 221-230). Blackwell Publishing Ltd.

McGuire, M., Mara, M., & Luebke, D. (2012, June). Scalable ambient obscurance. In Proceedings of the Fourth ACM SIGGRAPH/Eurographics conference on High-Performance Graphics (pp. 97-103). Eurographics Association.

Melax, S. (1998). A simple, fast, and effective polygon reduction algorithm. Game Developer, 11, 44-49.

Merschbrock, C., Lassen, A., Tollnes, T., Munkvold, Bjørn, E. (2016). Serious games as a virtual training ground for relocation to a new healthcare facility. Facilities. Vol. 34.

Mobach, M. (2008), Do virtual worlds create better real worlds?, Virtual Reality, Vol. 12, No. 3, pp. 163-179.

Modjeska, D. and Chignell, M. (2003), Individual Differences in Exploration Using Desktop VR, Journal of the American Society for Information Science and Technology, Vol. 54, No. 3, pp. 216-228.

Pelosi, A. W. (2010). Obstacles of utilising real-time 3D visualisation in architectural representations and documentation. In Proceedings of the 15th International Conference on Computer Aided Architectural Design Research in Asia/Hong Kong (Vol. 7, pp. 391-400).

Piirainen, K. A., & Gonzalez, R. A. (2014). Constructive synergy in design science research: a comparative analysis of design science research and the constructive research approach. Liiketaloudellinen Aikakauskirja, 3-4.

Rákos, D. (2012). Programmable Vertex Pulling. OpenGL Insights, A K Peters/CRC Press, 293-300.

Reddy, M. (1997). The effects of low frame rate on a measure for user performance in virtual environments. University of Edinburgh, Computer Systems Group.

Roupé, M., Johansson, M., Viklund Tallgren, M., Jörnebrant, F. och TOMSA, P. (2016) Immersive visualization of Building Information Models, Proceedings of the 21st International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA 2016) p. 673-682

Rubino, C., & Power, J. (2008). Level design optimization guidelines for game artists using the epic games: Unreal editor and unreal engine 2. Computers in Entertainment (CIE), 6(4), 55.

Shi, Y., Ferlet, E., Crawfis, R., Phillis, P., & Durano, K. (2015). 3D Hospital: Design and implement quest-based game framework for transitional training. In Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES), 2015 (pp. 119-125). IEEE.

Shiratuddin, M. F., & Fletcher, D. (2006, August). Southern Miss' Innovation and Commercialization Park: Development of a Large Scale Real-Time Virtual Reality Environment. In Proceedings of 6th International Conference on Construction Applications of Virtual Reality.

Steel, J., Drogemuller, R., & Toth, B. (2012). Model interoperability in building information modelling. *Software & Systems Modeling*, *11*(1), 99-109.

Sutherland, I. (1965), Ultimate display, in W.A. Kalenich (ed), Proceedings of IFIP Congress 2, New York, Spartan Books, 506–508.

Sunesson, K., Allwood, C. M., Paulin, D., Heldal, I., Roupé, M., Johansson, M., & Westerdahl, B. (2008). Virtual reality as a new tool in the city planning process. Tsinghua Science & Technology, 13, 255-260.

Svidt, K., & Christiansson, P. (2008). REQUIREMENTS ON 3D BUILDING INFORMATION MODELS AND ELECTRONIC COMMUNICATION–EXPERIENCES FROM AN ARCHITECTURAL COMPETITION. In CIB W78 25th International Conference on Information Technology: Improving the Management of Construction Projects Through IT Adoption, Chile (pp. 231-238).

Vlachos, A. (2015) Advanced VR rendering. Presentation at the Game Developers Conference 2015, San Francisco, California

Westerdahl, B., Suneson, K., Wernemyr, C., Roupé, M., Johansson, M., & Allwood, C. M. (2006). Users' evaluation of a virtual reality architectural model compared with the experience of the completed building. Automation in construction, 15(2), 150-165.

Willmott, J., Wright, L. I., Arnold, D. B., & Day, A. M. (2001). Rendering of large and complex urban environments for real time heritage reconstructions. In Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage (pp. 111-120). ACM.

Wilson, T. (2015) High performance stereo rendering for VR. Presentation at San Diego Virtual Reality Meetup #3, San Diego, California.

Wloka, M. (2003), Batch, batch, batch: What does it really mean?, Presentation at Game Developers Conference 2003.

Woksepp, S., & Olofsson, T. (2008). Credibility and applicability of virtual reality models in design and construction. Advanced Engineering Informatics, 22(4), 520-528.

Yan, W., Culp, C., & Graf, R. (2011). Integrating BIM and gaming for real-time interactive architectural visualization. Automation in Construction, 20(4), 446-458.

Yoon, S. E., Gobbetti, E., Kasik, D., & Manocha, D. (2008). Real-time massive model rendering. Synthesis Lectures on Computer Graphics and Animation, 2(1), 1-122.