

MComp Computer Games Development

Project Management & Research Methods. (ACDA5004)

Project Proposal.

Developing a Real Time Ray Tracer on
Simple Reflective Geometric Polygons

Contents

Contents	2
Introduction	3
Background of Ray Tracing	3
Real Time Ray Tracing	3
Project Research Proposal	4
Proposed Research Project Title: Developing a Real Time Ray Tracer on Simple Reflective Geometric Polygons	4
Objectives	5
Expected Result.....	5
Stretch Outcome	5
Type of Research.....	5
Research philosophy	5
Data Collection.....	6
Methods of Development.....	6
Risks and limitations	6
Data Analysis.....	6
Resources and planning	7
Resources and Requirements	7

Introduction

In this Research Proposal, there will be a short review into the background on the field of Ray Tracing; its purpose and the introduction of the Real Time Ray Tracing or RTX. Furthermore this proposal will be presenting a research topic, covering the objectives, and expected outcomes with the desired resources.

Background of Ray Tracing

Ray Tracing (RT), used in 3d rendering, is a technique in computer graphics[1] that simulates the path that light 'travels' through the 3d environment via 'bounces'. [2] This is the technical terminology for when a light reflects off of another object's surface on the computer screen between the pixel to the source of light. This produces a photo realistic image that other methods like rasterization cannot achieve. [3] The picture on a computer screen is comprised of pixels: a unit of measurement in computing. A tiny square of colour on the screen and a congregate of pixels make up the image on the monitor that the eye sees, representing a tiny portion of the screen. Pixels can vary in size depending on the range of set amounts of resolution, such as 1280 x 1024, 640 x 640 or 4K. The resolution represents how many pixels are in a rendered image, which is calculated by the width times the height of the image, approximately between a few hundred thousand pixels and millions of pixels per screen. [2][4]

Ray Tracing is an algorithm that calculates the correct colour of each pixel by tracing the direction of light between the pixel and the light source or reflections of the light source that reaches the eye from the screen. [5] One would assume that to find this out they could just start at the light's origin and reflect all the colours encountered, from objects, to the required pixel. This is called Forward Ray Tracing, but this is too computationally expensive because determining the direction of the rays to the next colour of the object or the correct pixel would be difficult. The number of rays required would be incalculable and many of those rays would be unnecessary, as they would be going in the wrong direction costing more power for the computer. [5]–[7] A different approach to this is the Backward Ray Tracing method which only traces the rays that are specific to the pixel. [5], [7]–[10] This traces from the eye to the screen and to the object reflecting the light source, unless there is no object reflecting the light. The ray would then not need to bounce until it finds the light source but considering the number of objects and independent light sources that could be within a scene, multiple rays could be bouncing all around the scene before hitting the screen. [5] There is another far more advanced way called Hybrid Ray Tracing which has multiple uses, but it effectively breaks down to combining the two; forward and backward ray tracing to give more realistic shadows, depth perception and reflections. [5], [7], [11], [12] This research proposal will not however be exploring this highly advanced method, unless of course, it is recommended by the supervisor.

Real Time Ray Tracing

Recently, Ray Tracing had an upgrade; Real Time Ray Tracing or RTX, which has ushered in an age of hyper-realistic imagery, photo-realistic reflections, and shadow casting. [2], [3], [13]–[15] RTX is solely dependent on hardware capabilities and will not be easy on the computer's power if not equipped to handle these advanced features. NVIDIA's patent in 2009 (not the earliest patent, but the most accessible one), [16] had a unique solution to this which was to create a dedicated hardware capable of doing the RTX algorithms. Over a decade after the patent NVIDIA released their

RTX 20 series,[17] which is now capable of doing the complicated Ray Tracing algorithms in real time, meaning when the image is needed on the screen. Many developers utilise the RTX cards' capabilities in different ways ranging from all the features it can handle down to concentration on singular features such as shadow casting or light from a fire.[18] Lifelike game environments in real time, that is not prerendered imagery, are now possible yet they still have their shortcomings. One being that the overall power of the computer must be suitable to use the graphics card's capabilities of Real Time Ray Tracing, even on lower polygonal objects in games like Minecraft.[15], [19], [20] The RTX graphics cards will require an adequately powerful computer with enough memory to serve the card's purpose. Although Real Time Ray Tracing is available to use on some older graphics cards it is not without problems, such as framerate drops[18], [21]

Project Research Proposal

Upon introducing this new development in game graphics there are some things that would need to be researched to acquire a better understanding of the subject.

Proposed Research Project Title: Developing a Real Time Ray Tracer on Simple Reflective Geometric Polygons

There are many ways that a ray tracer can be implemented[5], [8], [13], although this research would most likely start using the simplest of ray tracing algorithms and advancing to the next possible solution.

As Real Time Ray Tracing in a game environment would require heavy computational power to render complex scenes, this research is only interested in exploring the basic algorithms in real time using simple geometry and basic surface physics[5], [13] Pending supervisory review this research will look at one of the methods of Ray Tracing in real time implementation; either Backwards or Forwards Ray Tracing in Real Time.

Forward Ray Tracing has its limitations as previously mentioned. Backwards Ray tracing has its own limitations although they are arbitrary due to the nature of the research. Lu et al. [22] states that calculating light backwards passing through a lens would have undesirable results. Such as, if a light ray were to pass through a lens in the opposite direction of the light's "natural" path, that light's direction would be somewhat skewed, giving off an inaccurate path and rendering the wrong information.

Ray tracing has always been integral to video game development.[5], [13], [23] Technological advances, explicitly the aforementioned real time ray tracing implementation in games, concurs with this research in the area of computer game development.

Due to the revolutionary upgrade in ray tracing, a better understanding of RTX would establish increased knowledge and development of Ray Tracing, expanding upon current and previous research with a view to strengthening it to benefit the field of games development. Having increased knowledge in the field of real time ray tracing would be beneficial for any game that would need a graphical upgrade and allow for further projects to have a better visual aesthetic.

Using lower polygonal geometric objects will allow for simpler raytracing bounces on the objects. The objects will be of a primitive type: cubes, cuboids, spheres. These primitive objects will be

created using triangle meshes, as Wald[24] cited in Christen's[25] Journal explains it is much easier to effectively ray trace objects comprised of triangles, as it simplifies the algorithms and allows for better optimisation, meaning improved speed and accuracy.

Objectives

Overarching aim : Research and Develop a Real Time Ray Tracer on Low Polygonal Reflective Geometric Objects

Objectives:

1. To research, investigate and simulate ray tracing algorithms applying to a real time implementation.
2. To better understand the operations of Real Time Ray Tracing.
3. To develop and assess Ray Tracing algorithms in real time on low polygonal reflective surface(s)

Expected Result

The expected result of the research is to have a fully rendered 3D environment with multiple low polygonal reflective surfaces at a suitable framerate, approximately 30 frames per second(fps), to demonstrate that the real time reflections are possible throughout the environment, using the most effective method or algorithm.

Stretch Outcome

The ray tracer works at a higher framerate, preferably 60fps, with smooth animation and using multiple objects with at least one high polygonal geometric object.

Type of Research

Research philosophy

The philosophy of this proposed research is that of a positivistic paradigm and a deductive experiment. The chosen paradigm, positivism is a method of research that acquires knowledge already connected to a topic of research. This will bolster the already known information and methods of Real Time Ray Tracing that the research is intending for use in development of a simulation. The research is of an epistemological approach; finding out the knowledge and how and why it is used with an experimental, simulated approach to the whole of the research. Kivunja et al[26] explains the epistemological approach is concentrated on human expertise, whether it is to acquire or develop the researcher's ability or, in this research's case, understand the conceptualisation of the topic in the field that is being researched. Therefore, epistemology is beneficial to the research and will help gather the data that will determine the hypothesis' accuracy.

Data Collection

Each of the objectives will require a set of data to be collected to allow for the creation of a working, measurable Ray Tracer in Real Time. These objectives will need to collect data in the form of test cases, measuring accuracy and speed, which contributes to the frame rate of the algorithm. This observation of the Ray tracers speed and accuracy will help best determine the calibration and develop an understanding of the uses of effective ray tracer methods.

Methods of Development.

The methods that will be used during the experimentation and simulation of the Ray Tracing algorithms will be C++ Object Orientated Programming. The tracer will most likely be of its own programmed C++ class and will be easily identifiable and measurable in accordance to the data collection methods previously defined. To document this class in a more readable fashion, use of Unified Modelling Language(UML) will better display its connection with other C++ Class Objects and again its identification within the program. The proposed methods of development will be developed in an OpenGL Application Programming Interface or OpenGL API.

Risks and limitations

Using Real Time Ray Tracing is can be quite a computationally expensive procedure[5], [14], [15], so this could accrue some undesired effects and difficulties during development. These risks are largely impacted by the limitations of hardware and computational power. Such risks are as follows:

- The speed may differ between different workstations depending on hardware capabilities.
- The accuracy may be skewed if working on different peripheral outputs.
- The Ray tracer may not work on certain workstations due to incorrect software or updates installed.
- Due to computational expense of the Real Time Ray Tracer some workstations could experience a system failure or “crash”.

Data Analysis

To better display the accuracy and speed of the test cases, a simple plot point graph or scatter graph combined with a line graph, the data should display the variance and relation between the two variables being measured. This will allow accurate checks between the methods and calibration of the methods. Using the line of best fit with the scatter graph will mark the ideal calibration of each test case, as the correlation will determine if the method is running too slow or too fast in association to the accuracy. The Line graph will accurately display the change in speed and accuracy. Combining these two analytical methods will give a general overview of the best measurements of speed and accuracy to run the simulations.

Resources

This research will carry out some computationally expensive algorithms and using capable hardware with dedicated Real Time Ray Tracing hardware is essential for this research. Real Time Ray Tracing(RTX) uses powerful Graphics Processing Units and the overall computational capabilities must be able to effectively use the GPU's to carry out the expensive methods and algorithms.

Resources and Requirements

Multiple sources recommend that the best hardware for Real Time Ray Tracing (using the video game Control[27] as an example[28] which uses all available types of RTX) is:

- GPU: GeForce RTX 2060
- CPU: Intel Core i5-7600K or AMD Ryzen 5 1600X
- RAM: 16GB
- OS: Windows 10 64-bit

[28]

Nvidia's GeForce RTX 2060 user manual recommends as a minimum:

- **Motherboard:** PCI Express® or up to PCI Express 3.0-compliant motherboard with one dual-width x16 graphics slot.
- **System Power Supply:** Minimum 500 W or greater system power supply with one 8-pin PCI Express supplementary power connectors.
- **Operating System:**
 - • Microsoft® Windows® 10 64-bit (April 2018 Update or later)
 - • Microsoft Windows 7, 64-bit
 - • Linux, 64-bit
- **System Memory:** 8 GB system memory (16 GB recommended)
- **Hard Disk Space:** 1.5 GB of available hard disk space

[29]

Due to how expensive this one particular GPU is[30], a middle ground using the NVIDIA's GTX 1080Ti card is showing promise as the most capable card of handling Real Time Ray Tracing in a fully rendered commercial game[ref] at around 30 fps. This card unfortunately is no longer stocked by the manufacturer[31]. Considering that the research is not expected to create a whole 3d Game Environment, an NVIDIA GTX 1080Ti should be more than satisfactory for achieving the expected result of the hypothesis.

Nvidia's GeForce GTX 1080Ti user minimum requirements are as follows: :

- **Motherboard:** PCI Express® or PCI Express 3.0-compliant motherboard with one dual- width x16 graphics slot.
- **System Power Supply:** Minimum 600 W or greater system power supply with one 6-pin and one 8-pin PCI Express supplementary power connectors.

- **Operating System:**
 - Microsoft Windows 10, 32-bit or 64-bit
 - Microsoft Windows 8, 32-bit or 64-bit
 - Microsoft Windows 7, 32-bit or 64-bit
 - Linux, 32-bit or 64-bit
 - FreeBSD x86
- **System Memory:** 8 GB system memory (16 GB recommended)
- **Hard Disk Space:** 350 MB of available hard disk space

[32]

The Equipment in the Games Laboratory in 204 of IQ Building should be sufficient to handle the computational power and the graphical requirement. However due to a recent pandemic it is not possible to personally evaluate the current available workstation's computational and hardware capabilities.

References and Bibliography

References

- [1] P. Rademacher, "The ACM Student Magazine Ray Tracing: Graphics for the Masses."
- [2] D. Rose, "2020: The year of Real Time Ray Tracing - GameTextures - Medium," 2020. [Online]. Available: <https://medium.com/gametextures/2020-the-year-of-real-time-ray-tracing-f8b29e89523b>. [Accessed: 14-May-2020].
- [3] P. Slusallek, P. Shirley, B. Mark, G. Stoll, and I. Wald, "Introduction to Realtime Ray Tracing," 2005. [Online]. Available: <https://dl.acm.org/doi/pdf/10.1145/1198555.1198740>. [Accessed: 17-May-2020].
- [4] "What is a Pixel? - Definition from Techopedia." [Online]. Available: <https://www.techopedia.com/definition/24012/pixel>. [Accessed: 14-Jun-2020].
- [5] A. S. Glassner *et al.*, *An Introduction to Ray Tracing*. 2019.
- [6] J. J. Snellenburg, B. Braaf, E. A. Hermans, R. G. L. van der Heijde, and V. A. D. P. Sicam, "Forward ray tracing for image projection prediction and surface reconstruction in the evaluation of corneal topography systems," *Opt. Express*, vol. 18, no. 18, p. 19324, 2010.
- [7] "Ray Tracing Algorithm | Various Types of Ray Tracing Algorithm." [Online]. Available: <https://www.educba.com/ray-tracing-algorithm/>. [Accessed: 14-Jun-2020].
- [8] Z. Popovi, "Ray Tracing," pp. 81–103, 1972.
- [9] J. Arvo, "Backward ray tracing," *Dev. Ray Tracing*, vol. 12, pp. 259–263, 1986.
- [10] S. Xing *et al.*, "High-efficient computer-generated integral imaging based on the backward ray-tracing technique and optical reconstruction," *Opt. Express*, vol. 25, no. 1, p. 330, 2017.
- [11] Z. S. Hakura and J. M. Snyder, "Realistic Reflections and Refractions on Graphics Hardware With Hybrid Rendering and Layered Environment Maps," pp. 289–300, 2001.

- [12] T. Willberger, C. Musterle, and S. Bergmann, "Deferred hybrid path tracing," *Ray Tracing Gems High-Quality Real-Time Render. with DXR Other APIs*, pp. 475–492, 2019.
- [13] E. Haines and T. Akenine-Möller, *Ray tracing gems: High-quality and real-time rendering with DXR and other APIs*. 2019.
- [14] E. Liu, I. Llamas, J. Cañada, and P. Kelly, *Cinematic rendering in UE4 with real-time ray tracing and denoising*. 2019.
- [15] V. V. Sanzharov, A. I. Gorbonosov, V. A. Frolov, and A. G. Voloboy, "Examination of the Nvidia RTX," *CEUR Workshop Proc.*, vol. 2485, pp. 7–12, 2019.
- [16] U. (DE); C. Alexander Keller and U. (DE) Waechter, "REAL-TIME PRECISION RAY TRACING," US 7,499,053 B2, 2009.
- [17] "GeForce RTX 20 Series Graphics Cards | NVIDIA GeForce." [Online]. Available: <https://www.nvidia.com/en-gb/geforce/20-series/>. [Accessed: 14-Jun-2020].
- [18] "Ray Tracing, Your Questions Answered: Types of Ray Tracing, Performance On GeForce GPUs, and More." [Online]. Available: <https://www.nvidia.com/en-us/geforce/news/geforce-gtx-dxr-ray-tracing-available-now/>. [Accessed: 14-Jun-2020].
- [19] "Minecraft RTX: System Requirements | NVIDIA." [Online]. Available: https://nvidia.custhelp.com/app/answers/detail/a_id/5014/~minecraft-rtx%3A-system-requirements. [Accessed: 14-Jun-2020].
- [20] "Official site | Minecraft." [Online]. Available: <https://www.minecraft.net/en-us/>. [Accessed: 29-Mar-2019].
- [21] "Ray tracing now runs on older Nvidia GPUs - but how fast is it? • Eurogamer.net." [Online]. Available: <https://www.eurogamer.net/articles/digitalfoundry-2019-real-time-ray-tracing-tested-on-gtx-pascal-hardware>. [Accessed: 14-Jun-2020].
- [22] C. Lu, A. Roetter, and A. Schultz, "Types Of Tracing - 1997 Sophomore College Ray Tracing Site," 1997. [Online]. Available: <https://cs.stanford.edu/people/eroberts/courses/soco/projects/1997-98/ray-tracing/types.html>. [Accessed: 14-Jun-2020].
- [23] Y. Deng, Y. Ni, Z. Li, S. Mu, and W. Zhang, "Toward real-time ray tracing: A survey on hardware acceleration and microarchitecture techniques," *ACM Comput. Surv.*, vol. 50, no. 4, 2017.
- [24] I. Wald and D. Der Ingenieurwissenschaften, "Realtime Ray Tracing and Interactive Global Illumination Dissertation zur Erlangung des Grades," *Optimization*, vol. 48, no. 4, p. 242, 2006.
- [25] M. Christen, "Ray Tracing on GPU Assistant Professor : Marcus Hudritsch Supervisor : Wolfgang Engel," *Thesis*, 2005.
- [26] C. Kivunja and A. B. Kuyini, "Understanding and Applying Research Paradigms in Educational Contexts," *Int. J. High. Educ.*, vol. 6, no. 5, p. 26, 2017.
- [27] "Control - Discover a World Unknown." [Online]. Available: <https://controlgame.com/>. [Accessed: 14-Jun-2020].
- [28] "Control System Requirements and New Ray Tracing Trailer Released." [Online]. Available: <https://www.nvidia.com/en-us/geforce/news/control-pc-system-requirements-ray-tracing-trailer/>. [Accessed: 14-Jun-2020].

- [29] Nvidia, "GeForce RTX 2060 User Guide," 2019.
- [30] "GeForce RTX 2060 SUPER Graphics Cards | NVIDIA." [Online]. Available: <https://www.nvidia.com/en-gb/geforce/graphics-cards/rtx-2060-super/>. [Accessed: 14-Jun-2020].
- [31] "GeForce 10 Series Graphics Cards." [Online]. Available: <https://www.nvidia.com/en-gb/geforce/10-series/>. [Accessed: 14-Jun-2020].
- [32] Nvidia, "GeForce GTX 1080Ti User Guide," 2018.

Bibliography

- [1] T. Whitted, "An improved illumination model for shaded display," *ACM SIGGRAPH 2005 Courses, SIGGRAPH 2005*, vol. 23, no. 6, 2005.
- [2] T.-Y. Kim and U. Neumann, "Opacity Shadow Maps," pp. 177–182, 2001.
- [3] C. Dachsbacher and M. Stamminger, "Reflective shadow maps," *Proc. Symp. Interact. 3D Graph.*, pp. 203–208, 2005.
- [4] L. Persson and L. Persson, "Performance in game environments with Shadow Maps based on Level of Detail," no. June, 2019.
- [5] T. H. E. Firm and O. F. Karlf, "Patent Application Publication (10) Pub . No .: US 2007 / 0009410 A1," vol. 1, no. 19, pp. 10–13, 2007.
- [6] C. Waechter, "(12) Patent Application Publication (10) Pub. No.: US 2009/0225081 A1," vol. 1, no. 19, 2009.
- [7] "(No Title)." [Online]. Available: <https://cs.stanford.edu/people/eroberts/courses/soco/projects/1997-98/ray-tracing/types.html>. [Accessed: 02-Jun-2020].
- [8] "(No Title)." [Online]. Available: <https://uk.pcmag.com/gaming-gear/125153/amd-to-bring-ray-tracing-effects-to-playstation-5-xbox-series-x>. [Accessed: 14-Jun-2020].
- [9] R. L. Cook, "Computer graphics in commercial and broadcast production (Panel)," *ACM SIGGRAPH Comput. Graph.*, vol. 18, no. 3, p. 205, 1984.
- [10] D. Lin and C. Yuksel, "Real-Time Rendering with Lighting Grid Hierarchy," *Proc. ACM Comput. Graph. Interact. Tech.*, vol. 2, no. 1, pp. 1–17, 2019.
- [11] M. Magro, K. Bugeja, S. Spina, K. Napoli, and A. De Barro, "Atlas shrugged: Device-agnostic radiance megatextures," *VISIGRAPP 2020 - Proc. 15th Int. Jt. Conf. Comput. Vision, Imaging Comput. Graph. Theory Appl.*, vol. 1, no. Visigrapp, pp. 255–262, 2020.

- [12] A. Hirvonen, A. Seppälä, M. Aizenshtein, and N. Smal, *Accurate real-time specular reflections with radiance caching*. 2019.
- [13] P. Shirley, *Ray Tracing: The Rest of Your Life*. 2019.
- [14] “What is ray tracing (raytracing, ray-tracing or ray casting)? - Definition from WhatIs.com.” [Online]. Available: <https://whatis.techtarget.com/definition/ray-tracing-raytracing-ray-tracing-or-ray-casting>. [Accessed: 14-May-2020].
- [15] D. C. Anderson and J. Cychosz, “An introduction to ray tracing,” *Image Vis. Comput.*, vol. 8, no. 2, p. 169, 1990.
- [16] Scratchapixel, “An Overview of the Ray-Tracing Rendering Technique,” 2016. [Online]. Available: <https://www.scratchapixel.com/lessons/3d-basic-rendering/ray-tracing-overview>. [Accessed: 14-May-2020].
- [17] A. s Glassner, “An introduction to ray tracing,” *Comput. Graph.*, vol. 17, no. 1, p. 107, Jan. 1993.
- [18] Viitanen (Thesis), *Hardware Accelerators for Animated Ray Tracing Timo Viitanen Hardware Accelerators for Animated Ray Tracing Julkaisu 1551 • Publication 1551 Tampere 2018*, vol. 1551, no. 2018. 2020.
- [19] T. J. Purcell, I. Buck, W. R. Mark, and P. Hanrahan, “Ray tracing on programmable graphics hardware,” *Proc. 29th Annu. Conf. Comput. Graph. Interact. Tech. SIGGRAPH '02*, pp. 703–712, 2002.
- [20] H. Weghorst, G. Hooper, and D. P. Greenberg, “Improved Computational Methods for Ray Tracing,” *ACM Trans. Graph.*, vol. 3, no. 1, pp. 52–69, 1984.
- [21] F. H. Brittain and M. E. Hale, “Some limitations of ray-tracing software for predicting community noise from industrial facilities,” *Inst. Noise Control Eng. USA - 23rd Natl. Conf. Noise Control Eng. NOISE-CON 08 Sound Qual. Symp. SQS 08*, vol. 2, pp. 895–906, 2008.
- [22] D. Fleet, A. Hertzmann, and F. Estrada, “Advanced Ray Tracing,” pp. 92–113.
- [23] F. Xie, E. Tabellion, and A. Pearce, “Soft Shadows by Ray Tracing Multilayer Transparent Shadow Maps,” *Most*, 2007.
- [24] B. De Greve, “Reflections and Refractions in Ray Tracing,” *October*, vol. 2, no. October, pp. 1–8, 2004.
- [25] P. Moreau, M. Pharr, and P. Clarberg, “Dynamic Many-Light Sampling for Real-Time Ray Tracing,” *High-Performance Graph. - Short Pap.*, 2019.
- [26] W. J. Lee, G. Liktov, and K. Vaidyanathan, “Flexible ray traversal with an extended programming model,” *SIGGRAPH Asia 2019 Tech. Briefs, SA 2019*, pp. 17–20, 2019.

- [27] J. Günther, S. Popov, H. P. Seidel, and P. Slusallek, "Realtime ray tracing on GPU with BVH-based packet traversal," *IEEE/ EG Symp. Interact. Ray Tracing 2007 Proceedings, IRT*, pp. 113–118, 2007.
- [28] J. M. Singh and P. J. Narayanan, "Real-time ray tracing of implicit surfaces on the GPU," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 2, pp. 261–272, 2010.
- [29] P. A. Navratil, W. R. Mark, P. A. Navr, and W. R. Mark, "An Analysis of Ray Tracing Bandwidth Consumption," *Univ. Texas, CS Tech Rep.*, pp. 1–4.
- [30] "Control PC Requirements and What Ray Tracing Will Look Like | Tom's Hardware." [Online]. Available: <https://www.tomshardware.com/uk/news/control-game-pc-requirements-ray-tracing,40096.html>. [Accessed: 14-Jun-2020].
- [31] "Real-Time Ray Tracing In Games Doesn't Necessarily Require Pricey Hardware." [Online]. Available: <https://www.forbes.com/sites/marcochiappetta/2019/11/25/real-time-ray-tracing-in-games-doesnt-necessarily-require-pricey-hardware/#6fdb632526a>. [Accessed: 14-Jun-2020].
- [32] "Getting started with ray tracing | High Definition RP | 7.1.8." [Online]. Available: <https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@7.1/manual/Ray-Tracing-Getting-Started.html>. [Accessed: 14-Jun-2020].
- [33] "What Is Ray Tracing? | Digital Trends." [Online]. Available: <https://www.digitaltrends.com/computing/what-is-ray-tracing/>. [Accessed: 14-Jun-2020].
- [34] "Ray Tracing Gems, High-Quality and Real-Time Rendering with DXR and Other APIs - Google Play." [Online]. Available: <https://play.google.com/books/reader?id=MNqRDwAAQBAJ&lr=&printsec=frontcover&pg=GBS.PA546>. [Accessed: 14-Jun-2020].
- [37] J. R. Wallace, M. F. Cohen, and D. P. Greenberg, "A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods," *Proc. 14th Annu. Conf. Comput. Graph. Interact. Tech. SIGGRAPH 1987*, vol. 21, no. 4, pp. 311–320, 1987.
- [38] "How important will ray tracing be for Sony's PS5 and Xbox's Project Scarlett? | GamesRadar+." [Online]. Available: <https://www.gamesradar.com/uk/how-important-will-ray-tracing-be-for-sonys-ps5-and-xboxs-project-scarlett/>. [Accessed: 14-Jun-2020].
- [39] C. Wagner, B. Kawulich, and M. Garner, "Selecting a research approach, methodology and methods," *Doing Soc. Res. A Glob. Context*, no. January 2012, pp. 51–61, 2012.
- [40] L. Silva, "Post-positivist review of Technology Acceptance Model," *J. Assoc. Inf. Syst.*, vol. 8, no. 4, pp. 255–266, 2007.
- [41] D. Straub and D. Gefen, "Validation Guidelines for IS Positivist Research," *Commun. Assoc. Inf. Syst.*, vol. 13, no. March, 2004.

- [42] L. Silva, “Post-positivist review of Technology Acceptance Model,” *J. Assoc. Inf. Syst.*, vol. 8, no. 4, pp. 255–266, 2007.