



DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2016

Concealing rendering simplifications using gaze contingent depth of field

TIM LINDEBERG



**KTH Computer Science
and Communication**

Concealing rendering simplifications using gaze contingent depth of field

Tim Lindeberg
timlin@kth.se

Master's Thesis at CSC

Supervisor at CSC: Christopher Peters
Examiner: Danica Kragic
Principal: Tobii AB
Supervisor at Tobii: Martin Dechant

June, 2016

Abstract

One way of increasing 3D rendering performance is the use of foveated rendering. In this thesis a novel foveated rendering technique called *gaze contingent depth of field tessellation* (GC DOF tessellation) is proposed. Tessellation is the process of subdividing geometry to increase detail. The technique works by applying tessellation to all objects within the focal plane, gradually decreasing tessellation levels as applied blur increases. As the user moves their gaze the focal plane shifts and objects go from blurry to sharp at the same time as the fidelity of the object increases. This can help hide the pops that occur as objects change shape.

The technique was evaluated in a user study with 32 participants. For the evaluated scene the technique helped reduce the number of primitives rendered by around 70 % and frame time by around 9 % compared to using full adaptive tessellation. The user study showed that as the level of blur increased the detection rate for pops decreased, suggesting that the technique could be used to hide pops that occur due to tessellation. However, further research is needed to solidify these findings.

Referat

Användning av ögonstyrt skärpedjup för att dölja renderingssimplifieringar

Ett sätt att öka renderingsprestanda i 3D applikationer är att använda foveated rendering. I denna uppsats presenteras en ny foveated rendering-teknik som kallas *gaze contingent depth of field tessellation* (GC DOF tessellation). Tessellation är när geometri delas i mindre delar för att öka detaljrikedom. Tekniken fungerar genom att applicera tessellation på alla objekt i fokalplanet och gradvis minska tesselleringsnivåer när suddigheten ökar. När användaren flyttar sin blick så flyttas fokalplanet och suddiga objekt blir skarpa samtidigt som detaljrikedomen i objektet ökar. Det kan hjälpa till att dölja de 'pops' som uppstår när objekt ändrar form.

Tekniken utvärderades i en användarstudie med 32 deltagare. I den utvärderade scenen visade sig tekniken minska antalet renderade primitiver med ca 70 % och minska renderingstiden med ca 9 % jämfört med att använda full adaptiv tessellation. Användarstudien visade att när oskärpa ökade så minskade antalet som såg sig se 'pops', vilket tyder på att tekniken kan användas för att dölja de 'pops' som uppstår på grund av tessellation. Det behövs dock ytterligare forskning för att säkerställa dessa fynd.

Acknowledgements

I would like to express my sincere gratitude to all the people who have helped me with this thesis. A special thank you to Martin Dechant for all the help he has provided with proofreading, structuring the report and general scientific knowledge and eye tracking expertise. I would also like to thank my supervisor at KTH, Christopher Peters, for the great help with designing and executing the user study as well as his excellent feedback on the report. Additionally I would like to thank Fredrik Lindh for all his help with the technical aspects of the project and all of his valuable input.

Thanks to Martin Schön and all the other thesis workers at Tobii for the great company during this semester. Last but not least I would like to thank all the great people at Tobii for making this project possible as well as their continuous assistance and encouragement.

Contents

Glossary	1
1 Introduction	3
1.1 Problem statement	4
1.2 Contribution	4
1.3 Thesis structure	5
2 Background	7
2.1 Level of detail	7
2.1.1 Level of detail in meshes	7
2.1.2 Tessellation	8
2.2 Eye tracking	10
2.2.1 The eye and visual attention	11
2.2.2 Video based eye tracking	13
2.2.3 Analysis of eye movement	14
2.2.4 Measuring data quality	15
2.3 Foveated rendering	16
2.4 Depth of field	18
2.4.1 Depth of field in 3D graphics applications	19
2.4.2 Gaze contingent depth of field	19
3 Methodology	21
3.1 Implementation details	22
3.1.1 Gaze contingent depth of field	23
3.1.2 Tessellation shader	25
3.2 Evaluation	27
3.2.1 Experiment scenario	27
3.2.2 Details of the user study	30
4 Results	33
4.1 Performance results	33
4.2 Detection of pops	35

5	Discussion	41
5.1	Performance	41
5.2	Noticeability	42
5.3	Methodology	43
5.4	Ethics and sustainability	44
6	Conclusion	45
6.1	Future work	45
	Bibliography	47
	Appendices	52
A	Further details of the user study	53
B	User study information sheet and consent form	55

Glossary

Point of regard (POR) - the position of a someones visual attention.

Eye tracking - to track a user's eyes in order to estimate his or her point of regard.

Gaze contingent (GC) - a gaze contingent system is one that can be controlled or manipulated using gaze.

Level of detail (LOD) - an umbrella term for techniques used in real time rendering applications to control the level of detail of objects in order to increase performance.

Geometric primitive - geometric primitives are the simplest geometric objects usable in a graphics application. Commonly used to refer to points, lines, triangles or quads.

Patch - a geometric primitive which can be subdivided, e.g. a line, triangle or a quad.

Polygon mesh - a collection of vertices, edges and faces that together form a 3D object.

Tessellation - the act of subdividing geometric patches to obtain further detail. Can be used to regulate LOD dynamically.

Foveated rendering - rendering techniques that accounts for the users gaze, often used to increase performance.

Depth of field (DOF) - the area which is in focus in an image is called the Depth of Field.

Focal plane - the plane where objects are in full focus.

Pop - a pop occurs when an object suddenly changes shape, for instance when swapping a low quality object for a high quality one.

GPU - Graphics Processing Unit, a specialised circuit designed to accelerate the creation of computer generated images.

Shader - a program which is executed on the GPU, often times to create an image.

CONTENTS

Post processing - methods of changing an image after rasterization. Includes effects such as depth of field, bloom, lens flares and more.

Virtual reality (VR) - virtual reality applications simulate a user's physical presence within a virtual world, often through the use of head mounted displays (HMD:s).

Aliasing - a distortion caused by representing high resolution images at a lower resolution. In 3D graphics this often results in jagged edges in polygons. Anti-aliasing is used to combat these artifacts.

Ray cast - an intersection test between a ray and a surface.

Chapter 1

Introduction

Real time 3D graphics applications are some of the most performance intensive in all of computer science. The demand for more and more realistic 3D renderings have been one of the major drivers of innovation in both hardware and software development during the last 20 years. While the performance of graphic cards have increased exponentially during this period the increase in performance is not enough to produce photo realistic renderings in real time. Computer graphics programming is therefore full of approximations, optimisations and shortcuts which can yield good looking results despite not adhering to the physical reality. Instead of treating lighting like photons bouncing off of surfaces, lighting is often times precalculated and stored as light maps. Meshes are switched from low fidelity to high fidelity as they get closer to the camera and physics calculations are heavily simplified. With the introduction of virtual reality (VR) applications this is true now more than ever since VR applications requires parts of the scene to be rendered twice. VR applications are shown on displays very close to the eye which requires higher resolution than for regular applications. This further drives the need for more powerful graphics cards and smarter rendering algorithms.

One solution to this problem could be eye tracking and foveated rendering. Foveated rendering techniques uses information about the user's point of regard to reduce the detail in areas where the user is not looking or to increase detail in the focused area. This can be used to increase rendering performance by a large amount. Since the area where things are seen clearly is very small, detail can be reduced in the periphery without it being noticeable. This does however require a system with very low latency in order to update the screen quickly enough that one does not perceive the changes, otherwise the user will see a *pop* as objects change shape.

Another use for eye tracking is to simulate the effect called depth of field (DOF). A lens can only focus sharply on one distance at a time, something that also occurs in the human eye. Using eye tracking this effect can be simulated in 3D applications which can be used to increase the sense of depth in flat images. Research also indicates that gaze contingent DOF can be used to decrease discomfort caused by

stereoscopic displays in VR settings.

This thesis proposes a novel foveated rendering technique that combines tessellation and gaze contingent DOF, called *gaze contingent DOF tessellation* (GC DOF tessellation). The key idea behind the GC DOF tessellation technique is that an object's transition from blurry to sharp can hide the transition from low quality to high quality. The technique could therefore be usable even with systems that have higher latency.

1.1 Problem statement

The aim of this thesis is to develop a novel foveated rendering technique based on combining tessellation and gaze contingent depth of field. This technique is called *gaze contingent depth of field tessellation* (GC DOF tessellation). Foveated rendering can be used to improve performance which in turn can be used to increase visual quality. However, if the changes are noticed by the user then the result might actually be that visual quality is decreased. It is therefore important to evaluate both the performance and the noticeability aspects of foveated rendering techniques. The thesis aims to answer the following questions.

- Can gaze contingent DOF be used to hide pops that occur when an object's detail is increased through tessellation?
- What performance gains can be had from employing GC DOF tessellation?

1.2 Contribution

One of the main beneficiaries of the work in this thesis is Tobii¹ who proposed the topic of foveated rendering. This work will provide them with additional knowledge of foveated rendering techniques and gaze contingent DOF. Academia in general will also benefit since the body of knowledge increases in these fields.

If the technique proves viable it can also benefit industries related to computer graphics such as the games industry. This could enable higher fidelity graphics or reduce power consumption. As screen resolution increases, foveated rendering techniques such as the one proposed in this thesis could become mandatory in order to ensure a smooth frame rate. This is especially true for VR applications since they require extremely high resolutions such as 8k or even 16k to match the resolution per visual angle of desktop displays or smart phones. Rendering such high resolutions with current techniques does not seem viable for the foreseeable future unless something dramatically changes in GPU:s.

¹<http://www.tobii.com/> [Accessed: 2016-06-09]

1.3 Thesis structure

Chapter 1 gives an introduction to the subject of the thesis, defines the goal and problem statement and gives a motivation for the work done. Chapter 2 explains relevant theory and related works in the fields of level of detail, eye tracking, visual attention, foveated rendering and gaze contingent DOF. Chapter 3 describes how the GC DOF tessellation technique was implemented and how the user study conducted to evaluate the technique was put together. Chapter 4 shows the result of the user study and in chapter 5 these results are discussed and analysed. Lastly the conclusion of the thesis is presented in chapter 6.

Chapter 2

Background

This chapter of the thesis gives some needed theoretical background and describes some of the related work in the fields of 3D graphics and level of detail methods. It also gives an introduction to the fields of eye tracking, foveated rendering and gaze contingent depth of field.

2.1 Level of detail

3D graphics is one of the most performance intensive fields in computer science. In order to achieve high quality graphics running at a smooth frame rate many compromises have to be made concerning the quality of meshes, the resolution of textures, the detail of shaders etc. One way of increasing the quality of 3D graphics is by employing level of detail (LOD) strategies [38].

2.1.1 Level of detail in meshes

One of the most commonly used LOD strategies is using simplified meshes at varying distances to the camera. This technique is called discrete LOD and was described as early as 1978 [6]. It is still used by most 3D graphics applications today. In this approach multiple versions of each object are created, either by hand or using mesh simplification algorithms. At run time a mesh is chosen depending on the distance between the object and the camera or how much of the screen the object occupies. Small objects or objects that are far away use a mesh with fewer vertices. This technique can vastly reduce the number of triangles that need to be rendered without reducing the visual quality of the scene. Using meshes with a high triangle count at large distances can result in each triangle occupying less space than a pixel which is an unnecessary amount of detail. At such distances it can be impossible to tell the difference between a high detail and a low detail mesh, see figure 2.1. An advantage to the discrete LOD technique is its simplicity and its very small overhead cost.

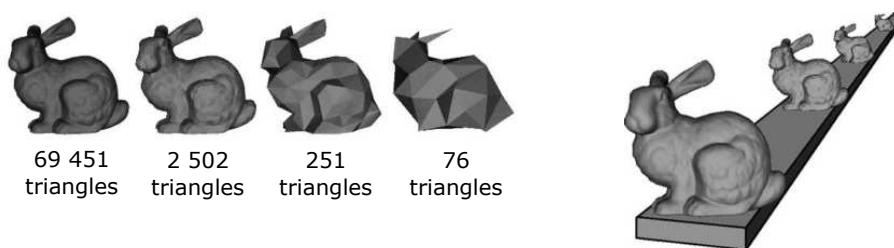


Figure 2.1: An example of discrete LOD. Meshes with different level of detail are used at different distances to the camera [38].

Another approach to mesh simplification is to use continuous LOD, first proposed in [26]. This method encodes the mesh as a list of vertices and a series of edge collapse operations. Since the edge collapse operation is invertible these operations can be reverted to reproduce the original mesh. Any number of these operations can be applied in order to produce a continuous LOD spectrum. The order of the vertices and indices can be sorted in the order of removal which allows smaller and smaller blocks of data to be streamed to the GPU as the detail level decreases. The list of indices can be cached and reused between frames to save performance. However, caching the indices for more frames makes the method look more like discrete LOD and some of the benefit of the method is lost. Even though this method sounds good in theory it is not used very often in practice. When vertices are removed from the model the geometry can become cache incoherent which can in fact decrease performance instead of increasing it [38].

A more sophisticated method is called view-dependant LOD which selects vertices depending on the current view [27, 37, 71]. All of these approaches use hierarchies of vertex merge operations that are selected dynamically at run time according to some set of view-dependent criteria. Parts of objects that are further in the background may be rendered with fewer polygons than those in the foreground. Often times more polygons are delegated to silhouettes and contours than areas that are flat relative to the camera. This method comes with some significant drawbacks such as a large overhead in processing time and memory consumption. More recent methods move these computations to the GPU which can increase performance for CPU-bound applications [10, 28]. View-dependent LOD is almost exclusively used for very large objects that cannot benefit from discrete LOD such as terrain [38].

2.1.2 Tessellation

A newer method of providing dynamic LOD is to use hardware accelerated tessellation which was first introduced in Direct3D 11 [45]. Using tessellation a highly detailed mesh can be generated from a simpler base mesh by subdividing its triangles, enabling dynamic LOD without artist involvement. Since GPU:s nowadays are mainly limited by bandwidth between the CPU and the GPU this can increase

2.1. LEVEL OF DETAIL

performance, especially for dynamic or animated meshes where many vertices have to be updated every frame [56].

Direct3D 11 adds three steps to the graphics pipeline between the Vertex shader and the Geometry shader. These are called the Hull shader, the Tessellator and the Domain shader (other names are used in OpenGL). The Hull shader takes a rendering patch as input argument, either a triangle, a quad or an isoline and produces tessellation factors for the Tessellator. It can also be used to calculate constant values for the patch. For a triangle a tessellation factor is calculated for each edge and for the inside of the triangle. The Tessellator is a fixed function that produces UV-coordinates depending on the given tessellation factors. The tessellation factors do not have to be integer values, floating point values can be used to enable smooth transition between different tessellation levels. All this is done directly on the GPU without need to access main memory which allows for very high tessellation performance [56].

The output of the Tessellator and the patch constants are then forwarded to the Domain shader. This shader is similar to the Vertex shader except that it receives a set of barycentric coordinates (for a triangle) from the Tessellator in each invocation instead of a vertex. It also receives the patch constant information from the Hull shader. The Domain shader outputs a vertex as well as per vertex information such as surface normals and texture coordinates. These new pipeline additions allow for fast adaptive tessellation which performs much better than previously available methods [56].

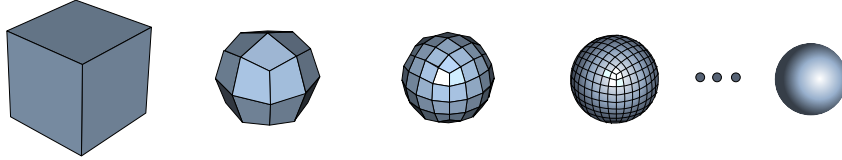


Figure 2.2: Iterations of the Catmull-clark subdivision algorithm applied to a cubic mesh [69].

Subdivision surfaces are a way of describing smooth surfaces in 3D graphics and are used in applications such as animated movies. However, due to their costly performance they're not used in real time rendering applications. They were first described in [5] as a recursive method that can reach smooth surfaces as the number of recursive steps approaches infinity. This technique can be seen in figure 2.2. Many variations and improvements to this method has been made [4, 11, 33] as well as a method to evaluate these surfaces exactly [59].

While these methods give great visual results they do not perform well using tessellation on the GPU. There are however approximate subdivision methods that can be used which provides good visual results and good performance [56]. One of the more popular methods is called PN-triangles [65] which models each triangle as a triangular Bézier patch. Another approach is Phong Tessellation [2] which uses a technique similar to Phong shading to smooth the geometry of the mesh. While

Phong tessellation is faster to compute PN-triangles usually produce a better visual result [51].

A common use of tessellation is to use displacement maps, first introduced in [8]. Height information is stored in the displacement map and each vertex is offset along it's normal by the value sampled from the map. Using tessellation a high number of vertices can be produced resulting in high detail displacement with correct contours. Since flat tessellation can be used the Hull shader is very cheap to compute but since an additional texture look up is required the technique can consume a lot of memory especially when using high resolution textures [56]. Flat tessellation and PN-triangles can be seen in figure 2.3.

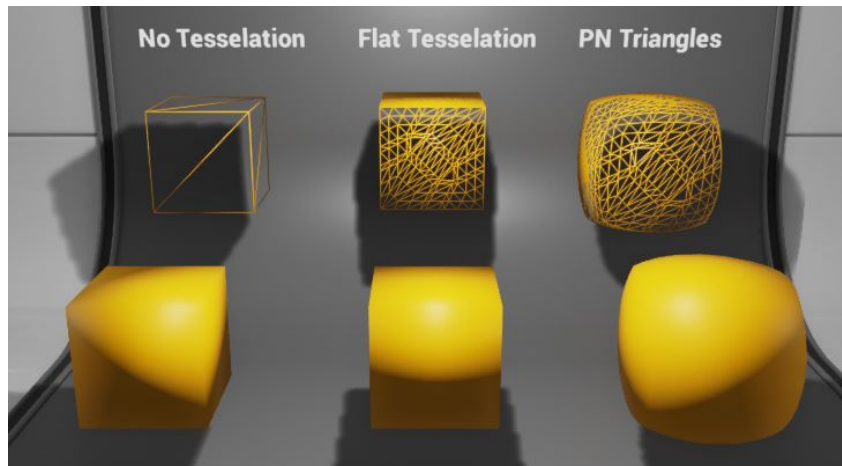


Figure 2.3: Flat tessellation and PN-Triangles tessellation applied to a cube [19].

When using tessellation to achieve dynamic LOD the tessellation levels are computed on the fly depending on different factors. It is important to assign a good tessellation factor to each edge since using a too small factor will lead to under tessellation and poor visual quality while using a too large factor will result in over tessellation. This will reduce rasterization efficiency and result in poor performance. Modern GPU:s are generally not very good at handling pixel-sized triangles [56]. Tessellation factors can be computed as a function of the distance between the camera and the middle of the edge [49] or by fitting a sphere around the edge and determine the factor based on the spheres diameter in screen space [3]. It is important that the tessellation factors for an edge is exactly the same for both patches otherwise cracks will appear in the mesh [56].

2.2 Eye tracking

Eye tracking is the process of tracking someones eyes and estimate their point of regard (POR), the point that holds the persons visual attention. There are

2.2. EYE TRACKING

numerous ways of accomplishing this though some methods are used more than others. One technique is Electro-OculoGraph (EOG) which measures the skins electric potential differences using electrodes placed at different points on the user's face. This technique is however not used very much these days [12].

One of the more accurate methods of tracking eye movement uses a scleral contact lens with a search coil. The coil in the lens is measured when moving through an electromagnetic field which gives very precise readings. The method is accurate to about 5-10 arc-seconds over a range of about 5° but is quite intrusive and can cause discomfort to the wearer. The insertion of the scleral lens can be cumbersome. Also, recent research indicate that modern optical eye trackers rival the scleral search coil method, making this method redundant in most use cases [30].

2.2.1 The eye and visual attention

People move their eyes towards a point of interest in order to bring that point in to the high resolution area of the retina. This lets one draw conclusions about the person's thoughts and what drew their attention. The eyes can thus be seen as a window in to a persons mind which is why visual attention is such a large field of research. Eye contact and gaze direction are central in human communication [50].

In the center of the retina is the fovea which is the part of the retina that has the highest concentration of the photoreceptor cells called cones. This area is where we see things in the highest resolution. Cones are responsible for color vision while rod cells are used to see less intense light. Rod cells are concentrated around the fovea [24]. Further details of the inner structure of the eye can be seen in 2.4.

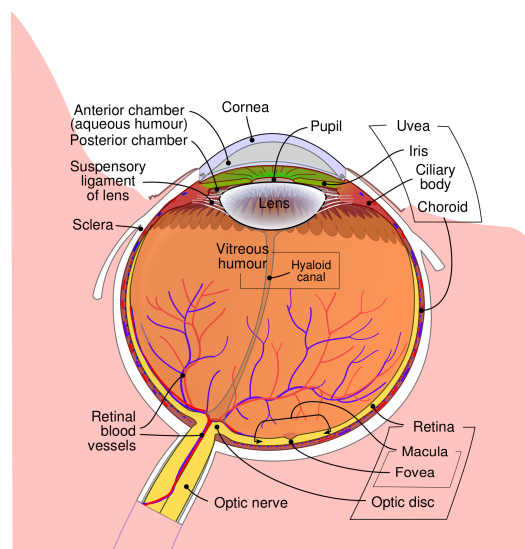


Figure 2.4: Illustration of the human eye [68].

When given a stimulus the eye will quickly scan the object through peripheral vision in low resolution. When an interesting feature is found the eye will quickly move towards that location and focus the region of interest within the fovea. The eye responds more quickly to features such as edges and contours than homogeneous areas and these areas will draw our attention even in the periphery [12].

When discussing vision and quality a common quantity is the visual angle, V , given by the formula

$$V = 2 \arctan \left(\frac{S}{2D} \right) \quad (2.1)$$

where S is the linear size of the object gazed at and the D is the distance from the eye. The fovea only subtends 2° of visual angle and the parafovea extends about $4 - 5^\circ$ and after that visual acuity drops off quickly [12]. Full acuity therefore only occurs in a small area, roughly the size of a thumb nail at arms distance [24]. Up to 30° is considered to be the useful visual field, after this we mostly perceive motion [12].

This means that we are unable to see anything in the periphery with any clarity. Even though the foveal region is more receptive to slower motion, motion is perceived uniformly over the visual field. The velocity of objects seem higher in the foveal region than in the periphery and the periphery is a lot more sensitive to moving targets than stationary ones. Detecting moving objects can be seen as one of the major functions of the periphery. The distribution of cones and rods within the retina also suggests that color sense is highly limited in the periphery [12].

There are different types of eye movements that have to be accounted for when doing eye tracking. The most common eye movement is the fixation which is when the eye is stabilised and is looking at a stationary object. One might think that the eye would not move at all during a fixation but a fixation consists of many very small eye movements [24].

Another type of eye movement is the saccade, which is when the eye transitions from one fixation to another. This movement is very quick, sometimes as fast as $900^\circ s^{-1}$. A saccade typically take between $30 - 80$ ms [24]. During this movement the executor is effectively blind [15].

A third eye movement is the smooth pursuit which is when the eye is tracking a moving target. A smooth pursuit has a maximum velocity of around $50 - 70^\circ$ per second and vision remains clear unlike during a saccade. The eyes are capable of matching the velocity of the target and the movement is a lot smoother than saccadic eye movement [15]. There are also other types of eye movements that will not be described here, see [15].

2.2. EYE TRACKING

2.2.2 Video based eye tracking

The most common method of doing eye tracking nowadays is to use video streams to track eye movements in real time. A video based eye tracking device from Tobii called the EyeX controller can be seen in figure 2.5. In order to estimate a users point of regard (POR) the head of the user either has to be fixed or multiple features of the eye have to be measured. The eye is illuminated using light, usually infrared, which is reflected in the cornea. These reflections are called Purkinje reflections or Purkinje images. There are four such reflections that occur due to the construction of the eye [12], see figure 2.6. P1 through P4 show the four different reflections [70]. P1 is the corneal reflection which is the most commonly used reflection in video based eye tracking. It can be enough to only track one reflection but more of them can be tracked for additional accuracy [9]. Since the first and fourth Purkinje reflection move together during eye translation but change interdistance during eye rotation this feature can be used to separate head movement from eye rotation.



Figure 2.5: The Tobii EyeX Controller video based eye tracker [64].

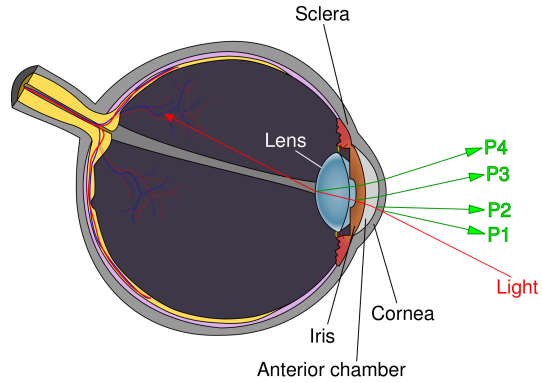


Figure 2.6: The locations of Purkinje reflections in the eye [70].

The raw image is extracted from the camera and sent for analysis. Systems that allow head movement first have to find the location of the face and eyes using image processing techniques. Smaller images containing only the eyes are typically extracted and used for further analysis. These images are used to segment the pupil and corneal reflection. The pupil and cornea are often detected using a feature based approach where regions of similar pixel intensities are analysed. In order to achieve good eye tracking it is important to have high image quality with a clearly separable pupil and corneal reflection. Image processing to detect the pupil and corneal reflection can be seen in figure 2.7. Once the position of the pupil and corneal reflection has been determined the POR can be calculated [24].

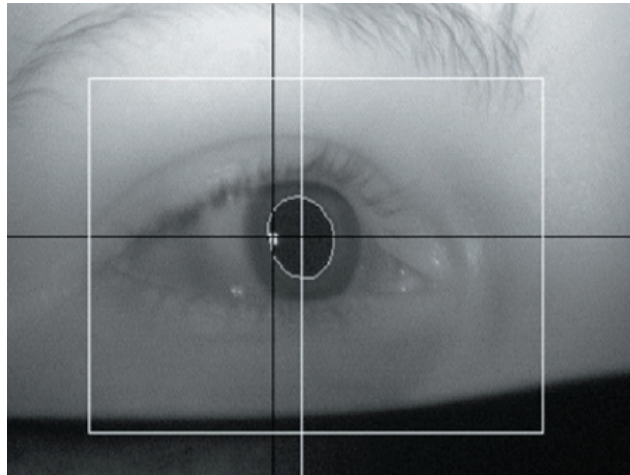


Figure 2.7: Image processing to detect the center of the pupil and the corneal reflection [55]. The white line shows the detection of the center of the pupil and the black cross shows the corneal reflection.

Most video based eye trackers require some form of individual calibration to function properly. The size of the radius of peoples eye balls can vary by up to 10 % and some people wear glasses or contact lenses which can change the size of the eye in the image. The calibration is usually performed by showing some visual stimuli at the extremas of the viewing region. The number of points shown can vary between different trackers from between two and sixteen typically. These points are usually small to ensure an exact POR during calibration. Any misalignment in gaze position during the calibration can introduce a corresponding offset in the gaze tracking data. For each of these positions the POR is calculated by looking at the relative position of the pupil and corneal reflection. These values can be interpolated at intermediate eye positions. The second goal of the calibration is to calibrate the trackers optics to the users eyes. The tracker tries to find the users pupil and Purkinje images automatically during the calibration. Detection thresholds can be adjusted for the specific users eyes. Long eyelashes, contact lenses and glasses can pose problems when detecting these features, which is why eye tracking does not work equally well for all users. Image processing techniques can however help to alleviate some of these problems [24].

2.2.3 Analysis of eye movement

The raw data provided by the eye tracker is not perfect and it is therefore important to reduce noise in the gaze signal. Noise can occur due to imperfect optical technology and the inherent instability of the eye. Blinks can also contribute to noise. However, the largest contribution of error lies in the gaze estimation algorithms. The eye is typically modelled as a sphere to simplify calculations while real eyes generally have grooves and other deformations which results in inaccuracies. Ac-

2.2. EYE TRACKING

curacy typically degrades in the peripheral regions of the screen and some trackers will ignore data outside of the effective operating range of the device [24].

There are different methods of filtering the raw data stream from the eye tracker. In general, filtering is a trade off between stability and responsiveness. A simple method is to use the average of the last N data points. This will give a more stable signal but will introduce a delay since multiple positions have to be added to the average before a significant change happens. To combat the issue one can use finite-impulse response filters. Using this method a weight from a kernel function is assigned to each gaze point where more recent data points typically have a larger weight. This can increase responsiveness while still improving the stability of the signal. There are also other commonly used filtering techniques, see [58].

When analysing visual attention one wants to detect saccades and fixations. There are two main ways of detecting these types of eye movements using the raw data from the tracker. In the first method the signal is averaged over time and if there is little to no movement over a certain amount of time a fixation is ongoing. The second method uses eye movement velocity. If the speed exceeds a predetermined threshold a saccade is ongoing [12]. These thresholds are typically obtained from empirical studies, a common source being [72].

2.2.4 Measuring data quality

When comparing trackers, either for research purposes or commercial purposes, the quality of the eye tracking data is the most important factor. There is however no clear standard for how eye tracking data quality is measured. Such a standard would help greatly in comparing research results and different eye trackers [25]. Work is being done by the COGAIN Association to determine standards for measuring eye tracking data quality [7].

The most important aspects of eye tracking data quality are spatial and temporal accuracy. Spatial accuracy refers to the difference between the actual and the measured gaze position in degrees. The temporal accuracy can be described as the difference between the measured time and the actual time of an eye movement. One can also measure the spatial resolution, the smallest eye movements that can be detected by the tracker [25].

There are many factors that can affect the quality of the eye tracking data which is why it is hard to find a standardised measurement. Different users can have different eye tracking quality depending the physiology of their eye, if they wear glasses or contact lenses or if they have droopy eyelids, strabismus etc. The quality can also depend on factors in the environment, such as lighting conditions and the distance between the user and the eye tracker. Another factor is the task being performed by the user as moving around might result in worse data quality. Different trackers might also perform better or worse during different circumstances [25].

2.3 Foveated rendering

Foveated rendering or gaze-contingent rendering involves using data about where the user is looking to optimise rendering in different ways. Since the human eye can only perceive things with high clarity in a very small region [61] not all parts of the screen have to be rendered in full detail [24]. Many level of detail (LOD) optimisation schemes could potentially take advantage of foveation.

Foveated rendering is not a new idea, the first study on foveated rendering was done in 1990 [31]. In this paper they used eye tracking to determine the users POR and adapt their ray tracing algorithm to use less samples in the users periphery. They concluded that this technique could give a performance saving of up to a factor of five. However, the technique was very noticeable according to the participants due to the high latency of the system.

One of the more significant studies on foveated rendering is a research paper from 2012 [21]. The authors use a multi-resolution foveated rendering technique where they render the scene in three different eccentricity layers. In the middle layer, in the foveated region, the scene is rendered in full resolution. In the second layer the scene is rendered in half the resolution and the out most layer use a quarter of the original resolution. These layers are then smoothly blended together. This technique can be seen in figure 2.8.

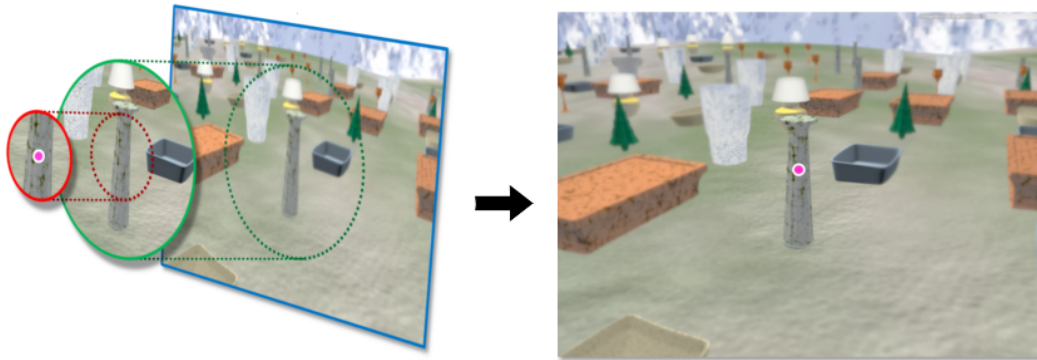


Figure 2.8: Multi-resolution foveated rendering as described in [21]. The pink dot shows the gaze position.

The technique suffers heavily from aliasing problems in the outer eccentricity layer which is remedied by using anti-aliasing techniques. Using this method the authors managed to accelerate graphics computation by a factor of five to six. The participants of the user study rated the foveated rendering quality as equal to or better than the non-foveated version when using conservative foveation quality. Careful consideration was taken to the effects of system latency. The experiment was executed using a Tobii TX300 eye tracker which runs at 300 Hz and a high frequency monitor running at 120 Hz. The Tobii TX300 is connected using an Ethernet cable which reduces latency in data transfer. The total latency of the system used in

2.3. FOVEATED RENDERING

the experiment was between 23 and 40 ms. The technique presented in the paper has been replicated in modern game engines, in VR-settings and expanded upon to use slightly different techniques [54, 60, 62]. Similar techniques have also been described earlier [13].

In the same paper [21] they also used tessellation to increase detail in objects. Objects were tessellated depending on distance to the camera and the maximum level of tessellation was based on which eccentricity layer contained the object. The inner and middle layer used a maximum of 20 000 triangles and the outer layer used a maximum of 8000 triangles per object. Their results show that the low resolution and blur can mask the LOD transitions in the outer layer [21].

There are multiple other techniques that take advantage of foveation. An early example is discrete mesh LOD switching using foveation [52] where objects that are not being focused by the users gaze are switched for simpler ones using six different levels of detail. It was shown that the technique could increase performance but the noticeability of the technique was not formally evaluated.

In [47] the authors propose a new gaze contingent rendering technique which utilises a ray casting technique where sampling is consistent with the limits of the human visual system. This technique was evaluated through a user study which looked at the time to execute a visual search task as the level of detail in the periphery was varied. The results show an inverse correlation between search time and visual detail in the periphery. It was found that keeping the silhouettes of the objects intact improved search times by a significant factor. Another foveated ray tracing technique used in HMD:s can be seen in [20].

Another interesting approach [32] encodes the color in the peripheral regions using fewer bits since humans are less color sensitive in the periphery. This can save up to 66 % of the bandwidth compared to using full 8-bit color. Yet another paper uses foveation to decrease the accuracy of ambient occlusion calculations far away from the gaze point. The number of sampling rays are decreased as a function of distance to the gaze point which yields significant speedups without decreasing the perceived visual quality of the ambient occlusion [41].

Another factor which can greatly affect the noticeability of foveation techniques is what task the users perform as the effect occurs. When performing specific tasks inattentional blindness can occur which is when a someone does not perceive things that are in plain sight. In [34] the authors investigate how inattentional blindness can affect the noticeability of LOD changes. In the experiment the users were not able to detect any of the LOD changes occurring in the periphery while performing a game related task until they were explicitly told about them. Once told they saw about 15 % of the LOD changes implying that performing tasks that require attention limits the detection rate of LOD changes.

As mentioned above the latency of a system significantly affects how noticeable foveation techniques are. This topic is investigated in [35] where participants of a user study were shown grey scale images where blur was applied outside a region around the gaze point. The study indicated that delays as long as 60 ms after an eye movement did not significantly increase detection of image blur or motion transients

due to the update. It was also concluded that longer eye movements led to a higher detection rate, likely because the eyes move further into the low-resolution area. This could be solved by increasing the size of the sharp region. The amount of blurring that occurred in the periphery also greatly affected the results indicating the higher delays could perhaps be used with less extreme foveation.

In order to produce a gaze contingent multi-resolution display that is completely indistinguishable from a full resolution system the display needs to be updated within 5 ms of the start of a fixation [36]. Greater delays produce detectable visual artifacts but do not impact the performance of visual tasks when using a 4.1° radius focused area. The authors also found a trade off between delay and the size of the full resolution area where longer delays required a larger focused area.

2.4 Depth of field

A lens can only focus sharply at one distance at a time and sharpness decreases gradually on each side of this depth. The area which is in focus is called the depth of field (DOF) or the focus range. The DOF effect can be seen in figure 2.9. The plane in focus or focal plane is the plane located at this distance. Any object not located within this plane will project to a region instead of a point. This region is known as the circle of confusion (COC), see figure 2.10. The depth of field effect is often used in photography and in cinema to direct attention towards particular objects and to increase the sense of depth within the image [48]. This effect also applies to the human eye and is affected by factors such as pupil size, optical aberrations as well as wavelength and the spectral compositions of the target object [42]. Human depth perception is affected by a number of different things such as relative size between objects, binocular stereopsis and motion parallax. It has also been shown that DOF and blur is a depth cue in human depth perception [43].

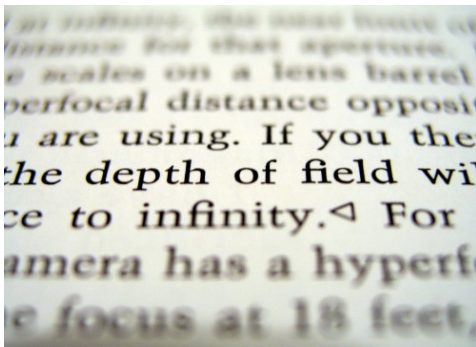


Figure 2.9: A picture taken with a very shallow depth of field [67].

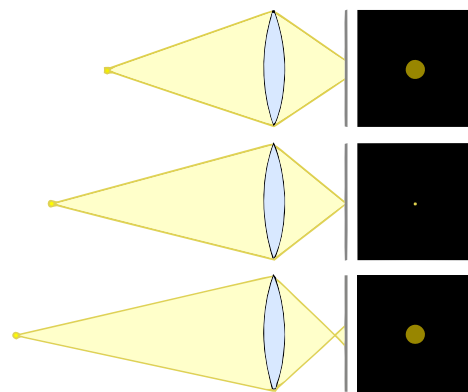


Figure 2.10: A diagram showing the circle of confusion (COC) [66].

2.4. DEPTH OF FIELD

2.4.1 Depth of field in 3D graphics applications

DOF is commonly used in 3D rendering applications but since these typically do not model the camera in a physically accurate manner the DOF effect has to be simulated. The COC can be calculated using the following formula

$$COC = \left| \frac{A * f * (d - P)}{d * (P - f)} \right| \quad (2.2)$$

where A is the aperture of the camera, f is the focal length, d is the distance to the point and P is the distance between the camera and the plane in focus. The COC can be used to determine how much blur to apply at a given distance [48].

There are two main ways of accomplishing DOF in 3D rendering applications. The first category of methods operate in object space and calculates DOF directly using the 3D representation of the scene. This can be accomplished by using realistic camera models which model aperture and focal length. Instead of tracing one ray per pixel multiple rays are traced to sample a finite aperture. This reflects the way an actual image is formed and produces realistic results. These technique are however very expensive and are not suited for real time rendering [1].

The second category of DOF effects use post processing methods. The image is first rendered in perfect focus and a depth map is used to calculate the amount of blur to apply to each pixel. Screen space methods are a lot faster than object space methods but does not produce as good results in all situations [1]. Blur can be applied to the image in different ways. A common method is to use gaussian blur which is fast and produces decent results. Other techniques try to simulate imperfections in the lens which cause bokeh patterns to appear [29].

2.4.2 Gaze contingent depth of field

More recently research has been made which combines eye tracking and depth of field to create gaze contingent DOF. One of the first studies which introduced gaze contingent DOF is [23]. In the paper the focal depth is calculated by sampling pixels in a square around the gaze point received from the eye tracker. This is done since eye tracking cannot provide an exact point of regard. Points closer to the gaze point receive a higher weight and semantically important objects such as player characters are also given a higher weight. This depth is then multiplied with the cameras forward vector to yield the world position at which to place the focal plane. The effect was evaluated in a user study. The participants consistently rated the gaze contingent DOF effect higher than using no DOF or using a constant DOF on the factors rendering realism, fun, depth perception and feeling of immersion. The constant DOF effect was not preferred over using no DOF at all.

A similar gaze contingent DOF implementation is described in [40]. The technique was evaluated using a slightly larger user study and similar conclusions were reached. The participants of the user study noticed and preferred the gaze contingent DOF compared to regular DOF. The authors also tested different levels of blur

and concluded that users liked a medium blur level more than low or high levels. Accuracy of the eye tracker was found to be very important to the experience since deviations between gaze points could lead to difficulties in computing the distance to the focal plane in a consistent manner. This could lead to the blurred region flickering back and forth which was annoying according to the users in the study. In a further study the same authors propose a method using hidden markov models to predict the gaze position [39]. They find that it significantly improves accuracy and stability of the focal plane.

Gaze contingent DOF has also been evaluated in VR environments. Stereoscopic displays used in head mounted displays can cause severe discomfort due to the images being rendered at a fixed display distance. This can result in eye strain and fatigue with prolonged usage. In [14] the effect of gaze contingent DOF is tested in VR environments. The study found that gaze contingent DOF can significantly reduce discomfort caused by stereoscopic displays, at least for those with high stereoacuity. As with earlier studies users reported a dislike of the defocus blur caused by the shifting focal plane. However, the authors conclude that with no smoothing the gaze data is too noisy causing the focal plane to shift rapidly.

Another paper studies what effects gaze contingent DOF can have on depth perception [44]. Through a user study the authors conclude that gaze contingent DOF increases the subjective perceived realism and depth as rated by the participants. The authors also conducted an experiment to gauge whether gaze contingent DOF could convey additional information of depth order, distance and relative spatial position of objects. They found this to be true but to a limited extent.

Chapter 3

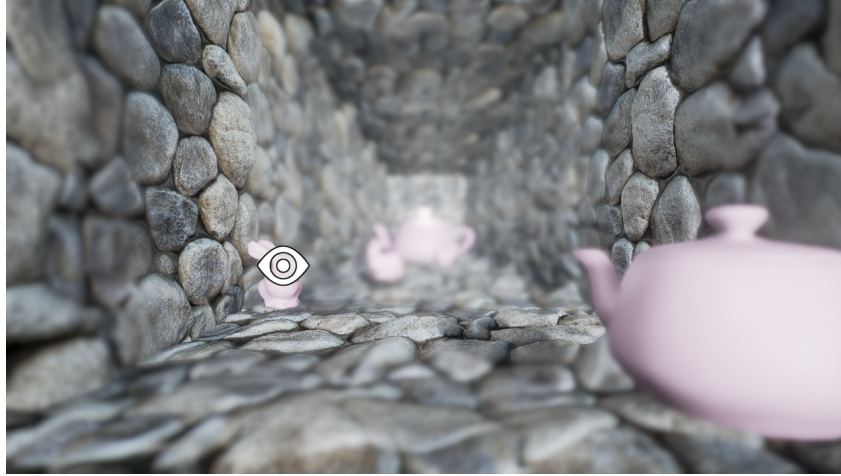
Methodology

The Methodology chapter describes the way the *gaze contingent DOF tessellation* (GC DOF tessellation) technique was implemented as well as the methodology used to evaluate the research questions described in section 1.1. Section 3.1 describes the implementation details and section 3.2 describes the method used to evaluate the technique from a performance perspective and to evaluate whether the technique can be used to hide pops.

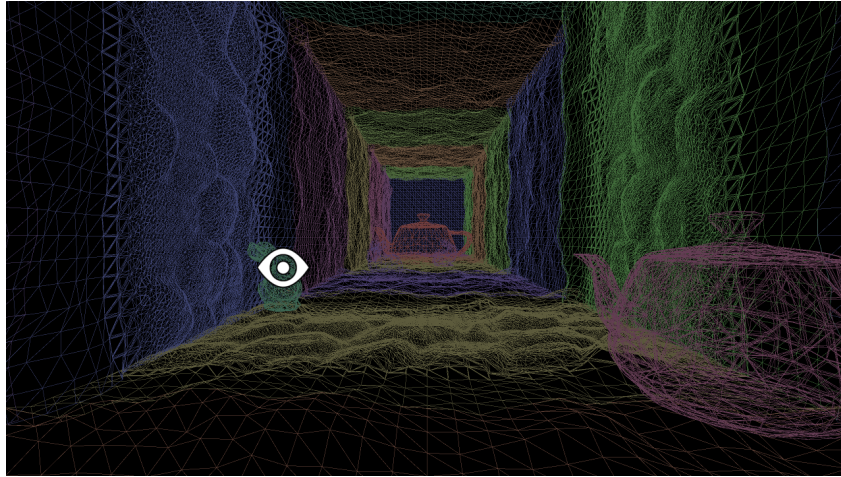
To increase performance in real time 3D-graphics applications it is common to apply different level of detail (LOD) schemes. These are often based on the distance between the object and the camera or the object's size on the screen, see 2.1. One can also use foveated rendering where eye tracking, see 2.2, is used to determine where the user is looking and make sure that rendering resources are delegated to this area, see 2.3. This approach requires a system with low latency, otherwise the user can see pops as the fidelity of an object changes. Large changes in an objects contour can often be noticed in the periphery as well.

Another way eye tracking can be used is to simulate depth of field (DOF) using the gaze position, see 2.4. This can result in increased perception of depth in an image and can increase perceived realism of a 3D rendering. Gaze contingent DOF has also been shown to decrease nausea and discomfort when used in head mounted displays.

When using gaze contingent DOF the focal plane is shifted as the user starts looking at a new object, causing the object to transition from blurry to sharp. The underlying idea of the GC DOF tessellation technique is that this transition can be used to hide an objects transition from low quality to high quality. Using GC DOF tessellation all objects within the focal plane are tessellated, see 2.1.2. The level of tessellation decreases as the applied blur increases and no tessellation is applied outside the focused region.



(a) Rendered using gaze contingent DOF.



(b) The wireframe of the scene.

Figure 3.1: An example scene showing the GC DOF tessellation technique. The eye icon represents the gaze position.

A visualisation of the GC DOF tessellation technique can be seen in 3.1.

3.1 Implementation details

This section describes the implementation of the GC DOF tessellation technique in more detail. Section 3.1.1 describes how the gaze contingent DOF effect was implemented, and section 3.1.2 describes the inner workings of the tessellation shader. The method as a whole was implemented in Unreal Engine 4.11¹, a free to use modern graphics engine.

¹<https://www.unrealengine.com>

3.1. IMPLEMENTATION DETAILS

3.1.1 Gaze contingent depth of field

Unreal Engine 4.11 supports three different DOF post processing methods: *Gaussian*, *Bokeh* and *Circle*. Bokeh DOF produces the most photo like results with proper bokeh shapes but is the most costly. Gaussian DOF will produce an even blur effect with no bokeh shapes and has reasonable performance. It is therefore the best suited for real time applications. Circle DOF is a newer method that can produce bokeh shapes at a lower performance cost compared to the bokeh method [16].

In this thesis Gaussian DOF was used for a number of reasons. It is the most viable method in most games, producing decent results at a relatively cheap computation cost. Bokeh DOF is often too expensive to use in real time rendering applications and is more suited for cinematics and showcases [16]. Secondly, computing the area of focus is easier using gaussian DOF than it is for bokeh DOF. Finding an exact formula for the blur size for a given distance proved quite difficult using bokeh DOF. Lastly, the bokeh DOF effect produced artifacts at the boundary between objects that could be confused with pops at large blur sizes. This could have skewed the results of the experiment. Circle DOF did not function properly at the time this thesis was written.

Gaussian blur is mostly controlled by the parameters **Focal Distance**, **Focal Region**, **Near Transition Range** and **Far Transition Range**.

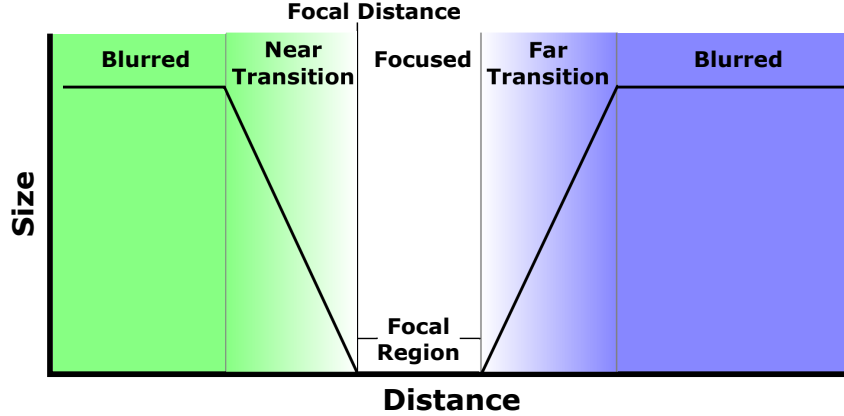


Figure 3.2: Description of how the gaussian DOF effect is controlled in Unreal Engine 4.11 [16].

Figure 3.2 shows how these parameters in combination with orthogonal distance affect the blur size. The horizontal axis shows the orthogonal distance from the camera and the vertical axis shows the blur size. **Focal Distance** determines the distance at which to place the focal plane, the region that is entirely in focus. **Focal Region** determines the length of an artificial region where everything is in focus. The **Near- and Far Transition Range** determine the distance over which the scene transitions from sharp to blurry. In this implementation the near and far transition

ranges were always the same. In addition, the maximum blur size can be controlled using the parameters **Near Blur Size** and **Far Blur Size** [16]. These values were also always the same for this implementation.

In order to enable gaze contingent DOF a Tobii EyeX Controller² was used. It is an eye tracker aimed at the consumer market which can be used in combination with the Tobii SDK for the Unreal Engine. This allows one to access gaze data in real time.

The gaze contingent DOF algorithm works as follows. First the object of focus is decided using a gaze-to-object mapping algorithm which was implemented in another thesis work at Tobii [57]. The technique is based on performing a second render pass to extract the 2D shape of the objects in the scene. Each object is rendered to a stencil buffer using a unique color. The buffer is then scanned and a stencil mask is generated for each object at 1 / 16 resolution. The stencil masks are then passed to the EyeX Engine, a software suite provided by Tobii, which decides which object holds the users focus. More information about stencil masks can be found in [57, 63]. The algorithm employed by the EyeX Engine is however proprietary so not all details are included.

Once an object of focus has been determined the gaze point is projected from screen space to world space and a ray cast is performed from this position into the scene. The collision point of this ray is where the focal plane should be placed. If the ray hits a different object than the one determined by the gaze-to-object mapping algorithm the process stops. Otherwise the focal distance, F , is calculated using the following formula

$$F = O \cdot \frac{C}{\|C\|} \quad (3.1)$$

where O is the vector from the camera's position to the collision point of the ray cast and C is the cameras forward vector. This gives the length of the projection of O on to C which is the orthogonal distance to the object, see figure 3.3. The focal distance is recalculated each frame.

²<http://www.tobii.com/xperience/products/>

3.1. IMPLEMENTATION DETAILS

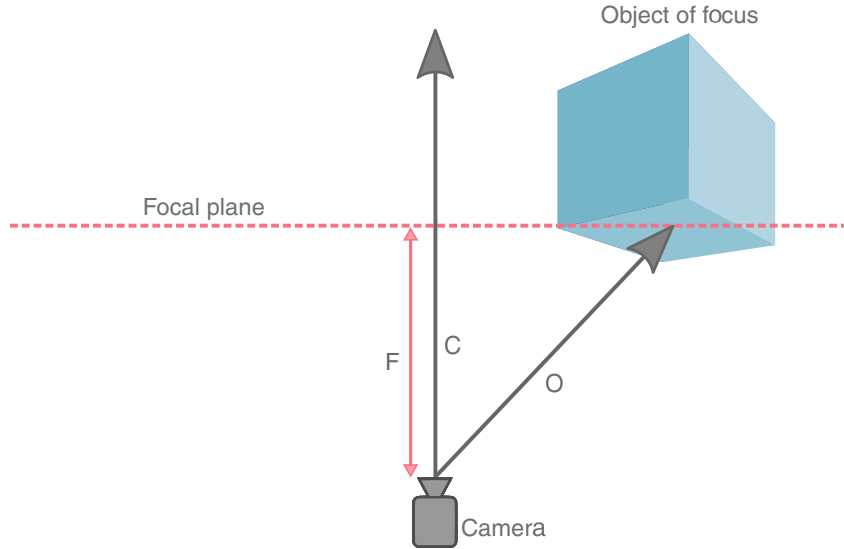


Figure 3.3: Diagram showing how the focal distance is calculated.

If the calculated focal distance is used directly the focal plane can move large distances between each frame which can be distracting. Therefore interpolation is used to move the focal plane towards the calculated focal distance each frame. The interpolated value is then used as the focal distance for the DOF algorithm. If the interpolation is too slow a delay will occur when switching the point of focus and if it's too fast the focal plane can jitter.

When the focal plane is placed far away from the camera the focused region will look smaller when using a perspective projection even though the size of the region is constant. The focal region and transition regions are therefore scaled by distance so that the focused region looks approximately the same size at all distances.

3.1.2 Tessellation shader

To add detail to the DOF region tessellation was used, see 2.1.2. Unreal Engine 4 has built in support for flat tessellation and PN-Triangles tessellation which can be used to smooth objects [19]. It also supports adaptive tessellation which uses a heuristic to keep the triangle size at a constant size (48 pixels by default) to avoid over tessellation. The technique is approximate, some triangles might be larger than this and some might be smaller.

The tessellation shader needs access to the parameters of the DOF post process in order to calculate the correct tessellation level. These values are passed to the tessellation shader using material parameter collections, one of Unreal Engine 4's way of passing data between the CPU and the GPU [18].

Most of the computation is done in the Hull shader where a tessellation level is determined for each edge. First the orthogonal distance between the camera and the vertex is calculated the same way as described in 3.1.1 by projecting the vertex

position on to the cameras forward vector. A tessellation level is then decided based on the blur size at the given vertex coordinate. The tessellation factor for each vertex is averaged and used as tessellation factor for the edge. The center of the triangle uses the average of the three edges as its tessellation factor.

For gaussian blur the blur percentage at a given orthogonal distance, d , is given by the function

$$Blur(d) = \max \left(\min \left(\frac{1 - \frac{R}{2} - |d - F - \frac{R}{2}|}{T}, 1 \right), 0 \right) \quad (3.2)$$

where R is the length of the focal region, T is the length of the transition regions and F is the current focal distance. The tessellation factor is then given by $1 - Blur(d)$.

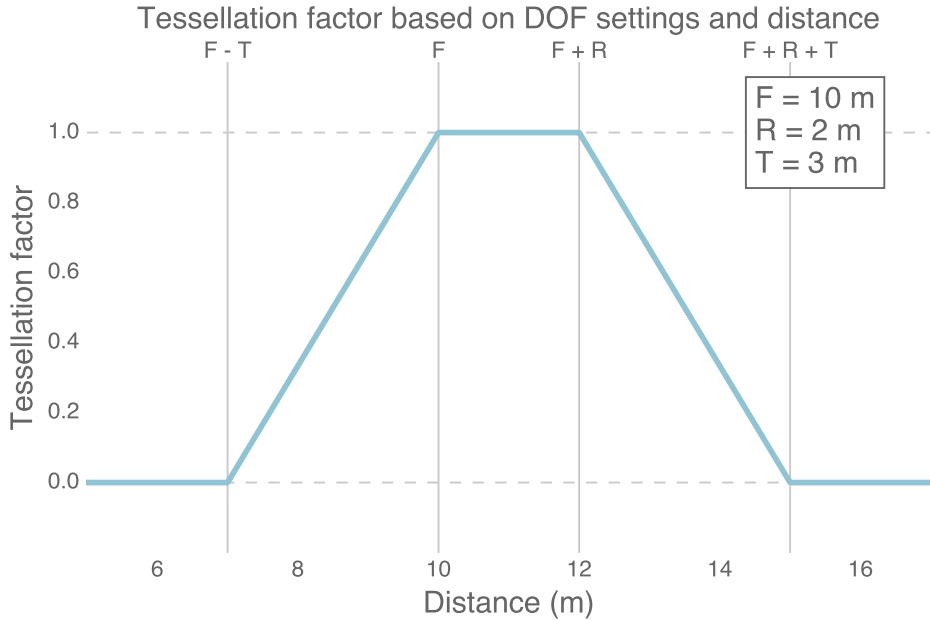


Figure 3.4: A description of the relation between distance, DOF settings and tessellation.

An example of how the tessellation factor is calculated for a specific set of DOF parameters can be seen in figure 3.4. In the example the **Focal Distance** is set to 10 m, the **Focal Region** is set to 2 m and the **Transition Regions** are set to 3 m. The horizontal axis shows the vertex's orthogonal distance from the camera and the vertical axis shows the tessellation factor. As can be seen, the equation gives a tessellation factor of 1 inside the focused area and a factor of 0 in the blurred area. In the transition regions the value is between 0 and 1. This value is then multiplied with the adaptive tessellation factor which results in no tessellation in the blurred areas and full adaptive tessellation in the focused region. This makes sure that geometry is not over tessellated.

3.2 Evaluation

In order to answer the research questions posed in 1.1 the technique described in 3.1 was evaluated through a user study. This section describes the details of the evaluation method and the user study.

An experiment scene was constructed which contained objects that could be affected by the GC DOF tessellation technique described in 3.1. The meshes in the scene mostly had a lower triangle count in order to enable increased level of detail through tessellation. To evaluate whether the gaze contingent DOF could hide the pops caused by tessellation the participants of the user study were told to look around the scene and search for pops. This was repeated multiple times for different levels of blur. Each scene was shown for ten seconds. If they saw a pop they were instructed to press space.

The experiment was executed using the GC DOF tessellation technique for levels of blur between 0.0 and 5.0 with intervals of 0.25 for a total of 20 levels of blur. A value of 0.0 means no blur was applied while 5.0 is quite an extreme level of blur, see figure 3.7. Each level of blur was tested twice. In addition, a number of control scenes were added where all objects used full adaptive tessellation to see whether the participants reported seeing pops when none occurred. There were 12 of these scenes, two for each level of blur between 0.0 and 5.0 in intervals of 1.0. In total there were 52 experiment scenes. These were presented in a predetermined order which was randomly selected. Every participant saw the experiment scenes in the same order.

To evaluate the performance of the method performance data was gathered during the experiment to ensure that the data reflected real world usage. For each scene the average time to render a frame was recorded. In addition to this the average number of rendered primitives each frame was recorded. This data was collected using the Direct3D11 API's `D3D11_QUERY_DATA_PIPELINE_STATISTICS` struct. It contains a value called `CInvocations` which gives the number of primitives sent to the rasterizer [46].

3.2.1 Experiment scenario

The experiment scene was constructed in Unreal Engine 4.11. The scenario is a modern 3D graphics scene with dynamic lighting and the type of post processing effects that one might find in a modern game. This type of scenario was chosen to see whether the method could be applicable in a modern game setting. The scene was constructed using free assets from Epic Games, mostly from the mobile game *Infinity Blade* [17]. These assets were selected because they were freely available and held a high graphical standard. Another important aspect is that the meshes in this game do not contain too many triangles since mobile games usually need lower fidelity meshes to run smoothly. This made the assets a good fit for the experiment since the meshes could be tessellated to increase detail. Some meshes had smooth groups added to them which is a requirement for the PN-Triangles

tessellation technique to work.

The scene used a static perspective, all movement of the camera was disabled. This was to make sure that all participants saw the same things and ensure a controlled environment. Allowing free movement also could have been distracting for participants who were not used to playing video games. The scene contained large distances and objects spaced at different depths to enhance the DOF effect. To encourage the participants to look around the scene some animation was added. The torches in the scene were animated as well as the flower and vines which moved with the wind. A snow particle effect was also added. The final scenario scene can be seen in figure 3.5.

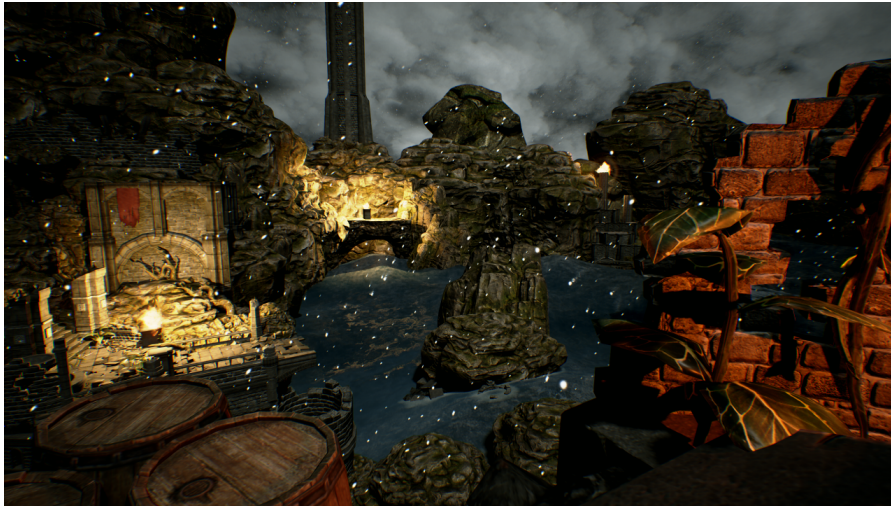


Figure 3.5: The experiment scenario scene.

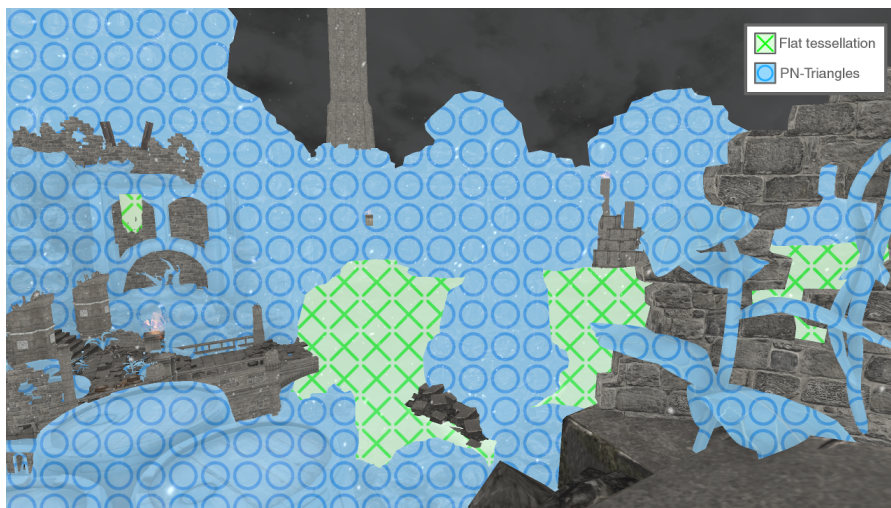


Figure 3.6: Objects affected by tessellation in the scenario scene.

3.2. EVALUATION

Figure 3.6 shows all the objects that were affected by the tessellation shader. The objects marked with circles were affected by PN-Triangles tessellation which is used to smooth objects. The objects marked with crosses were affected by flat tessellation and displacement mapping. The marked objects were the only objects in the scene that could produce visual pops. Some objects were not affected by tessellation at all since there is little point in adding detail to flat surfaces or edged objects.

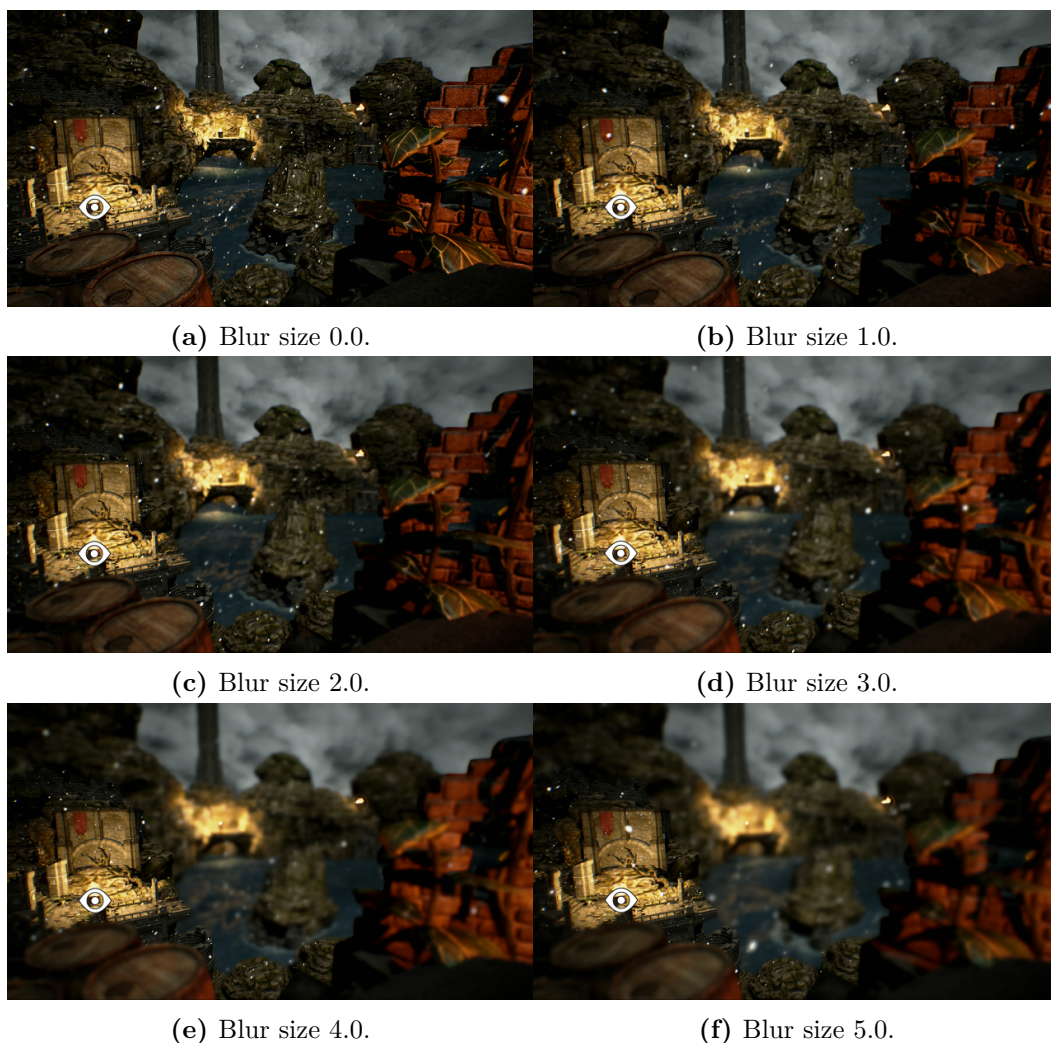


Figure 3.7: Different levels of blur applied to the experiment scene. The eye icon represents the gaze position.

Different levels of blur can be seen in figure 3.7. Values between 0.0 and 5.0 were tested in the experiment. No values greater than 5.0 was tested since it was assumed higher values would give similar results.

3.2.2 Details of the user study

The participants in the user study consisted of 32 students at KTH (The Royal Institute of Technology) most of whom were part of an introductory course in computer graphics and interaction (all but three participants). There were also five additional participants who did not show up for the experiments. The participants were 81% male and 19% female, aged between 20 – 43 with a median age of 24. 41% wore glasses, 19% had contact lenses and 40% used no corrective lenses. The participants of the computer graphics course were told that they would receive a bump in their grade if they were close to a higher grade as compensation for participating in the study.

The study took place at KTH in a private office, see appendix A. This environment was free from visual and audial distractions. The curtains were drawn at all times to ensure that each participant had a similar lighting environment. This is important since external light sources can negatively affect the eye tracking device. The participants were placed in a chair without wheels to ensure that the participants were placed at the same distance to the screen for the duration of the experiment. On the floor were markings where the back legs of the chair were to be placed to ensure that the distance between the participant and the screen was roughly 60 cm which is in the optimal tracking range for the eye tracker. The desk could be raised and lowered mechanically so that the screen never had to be moved or adjusted.

The experiment was executed on a modern computer with an NVIDIA GTX 970 graphics card which is known to have very high tessellation performance [22]. The monitor used was a 27" monitor running at 2560x1440 resolution and 60 Hz. The experiment scene ran at between 40–45 frames per second. The eye tracker used was a consumer level tracker called Tobii EyeX Controller which runs at approximately 60 Hz. For full details of the hardware and software used see appendix A.

As the participants entered the room they were given an information sheet and a consent form which all participants signed. Appendix B shows these documents in full. They were based on the consent form and information sheet used in [53].

The eye tracker was then calibrated for the participants eyes using a seven point calibration. Since the accuracy and precision of the eye tracking can vary greatly between individuals the tracking for each participant was visually examined. Though there were differences in accuracy and precision none of the participants tracking was deemed bad enough to remove them from the study.

Once the calibration was completed the experiment would start. The participants took part in two user studies, this one and another eye tracking related study by another masters thesis student [57]. In order to avoid bias from tiredness etc. half of the participants started with this study and the other half started with the other user study. If deemed necessary the tracking quality was examined once again between the experiments.

The participants were first shown what a pop could look like through three objects that changed from no tessellation to high tessellation. They were told to look

3.2. EVALUATION

around the scene actively and that no pops would be visible if their gaze remained stationary. If they saw a pop they were told to press space. The DOF effect was also described and the difference between a pop occurring through geometry changes and through change in blur level was explained. The experiment lasted for, on average, 7 minutes and 25 seconds per participant.

Chapter 4

Results

The Results chapter shows the results obtained from the user study described in 3.2. The performance results can be seen in section 4.1 detailing the number of primitives rendered and the frame time recorded during the experiment. Section 4.2 shows the noticeability results acquired from the user study.

4.1 Performance results

This section shows the performance data that was gathered during the user study.

The horizontal axes in figures 4.1 and 4.2 shows the experiment number in the order that they were executed. In total there were 1344 scenes that used GC DOF tessellation, 42 per participant. Each scene lasted between 0.8 and 10 seconds which was the timeout limit. On average each scene lasted 8.3 seconds.

Figure 4.1 shows the average number of primitives rendered per frame for each experiment scene. The vertical axis shows the average number of primitives rendered in millions. The line labelled full tessellation shows the number of primitives rendered when using full adaptive tessellation (8.01 million) and the line labelled GC DOF tessellation shows the number of primitives when using the GC DOF tessellation technique (2.50 million). This is an improvement of 68.7 % compared to using full adaptive tessellation. The minimum amount of primitives rendered for a scene was 1.01 and the maximum was 3.93 million.

Figure 4.2 shows the average frame time for each experiment scene. The vertical axis shows the average time to render each frame in milliseconds. The scenes which produced the spikes seen in the graph all have short durations. The spikes most likely occur due to start up times affecting the result to a higher degree for scenes with short durations in combination with the user looking at a part of the scene that resulted in many primitives rendered. Scenes that lasted a longer duration were closer to the average. Other uncontrollable factors could also affect long frame times. The line labelled full tessellation shows the frame time when using full adaptive tessellation (24.25 ms) and the line labelled GC DOF tessellation shows the frame time when using the GC DOF tessellation technique (22.02 ms). This is

an improvement of 9.2 % compared to using full adaptive tessellation.

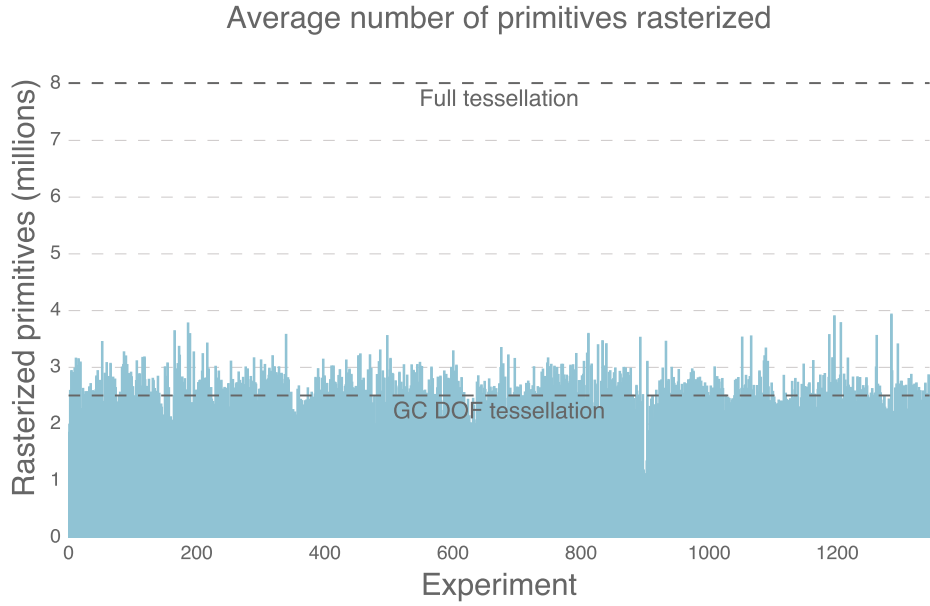


Figure 4.1: The average number of rasterized primitives per frame for each experiment scene.

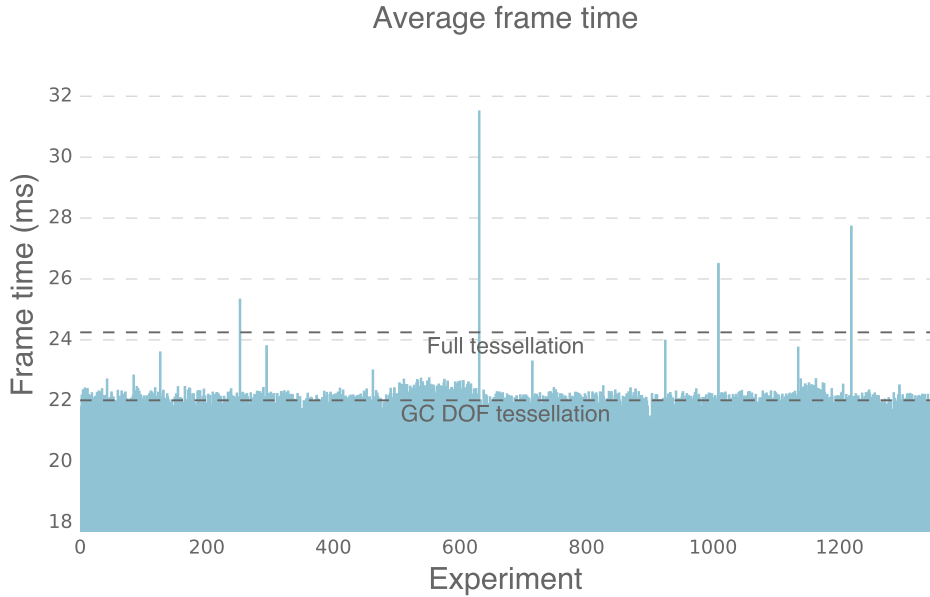


Figure 4.2: The average time to render a frame for each experiment scene.

4.2. DETECTION OF POPS

4.2 Detection of pops

This section shows the data regarding the detection of pops that were gathered during the user study.

Figures 4.3 and 4.4 shows the participants answers. The horizontal axis shows the blur size for the tested scene and the vertical axis shows the participants number. A filled in square means the participant said they saw a pop during that scene and a blank square means that they did not report seeing a pop within 10 seconds. Each level of blur was tested twice and are shown here in order of appearance. Figure 4.3 shows the answers for the scenes that used GC DOF tessellation and figure 4.4 shows the answers for the control scenes that used full adaptive tessellation where no pops could occur.

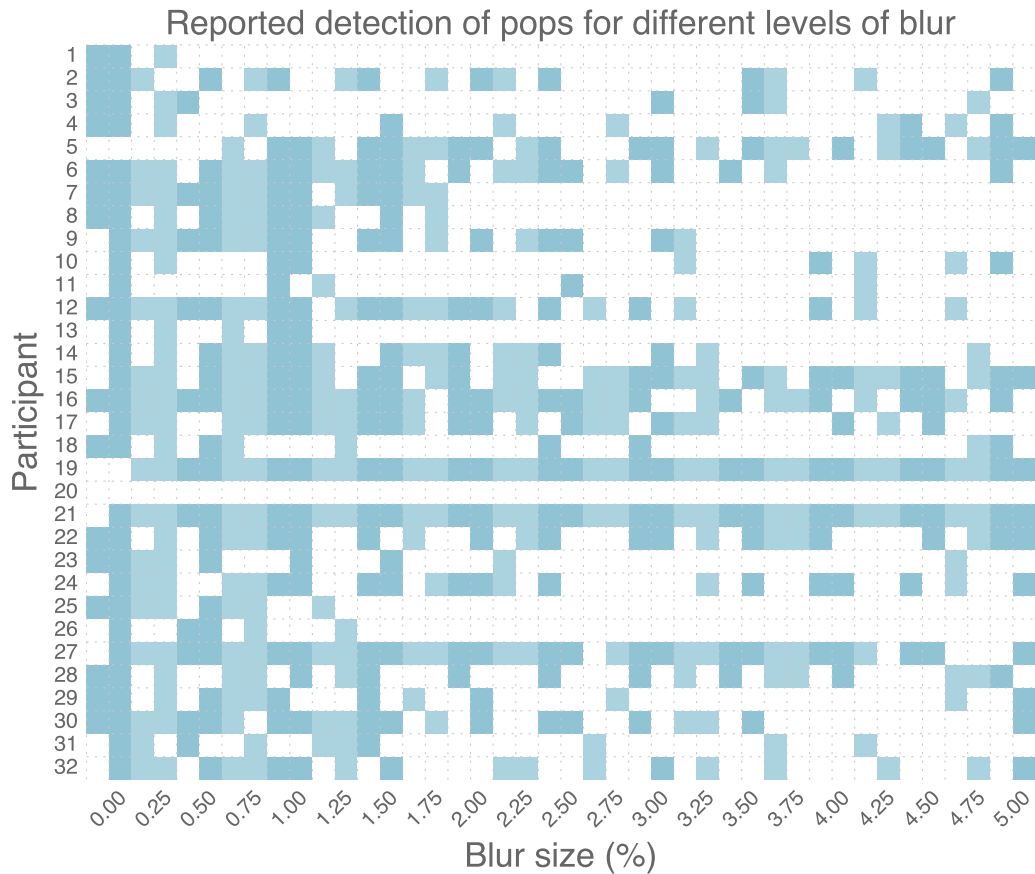


Figure 4.3: The participants answers for the scenes that used GC DOF tessellation.

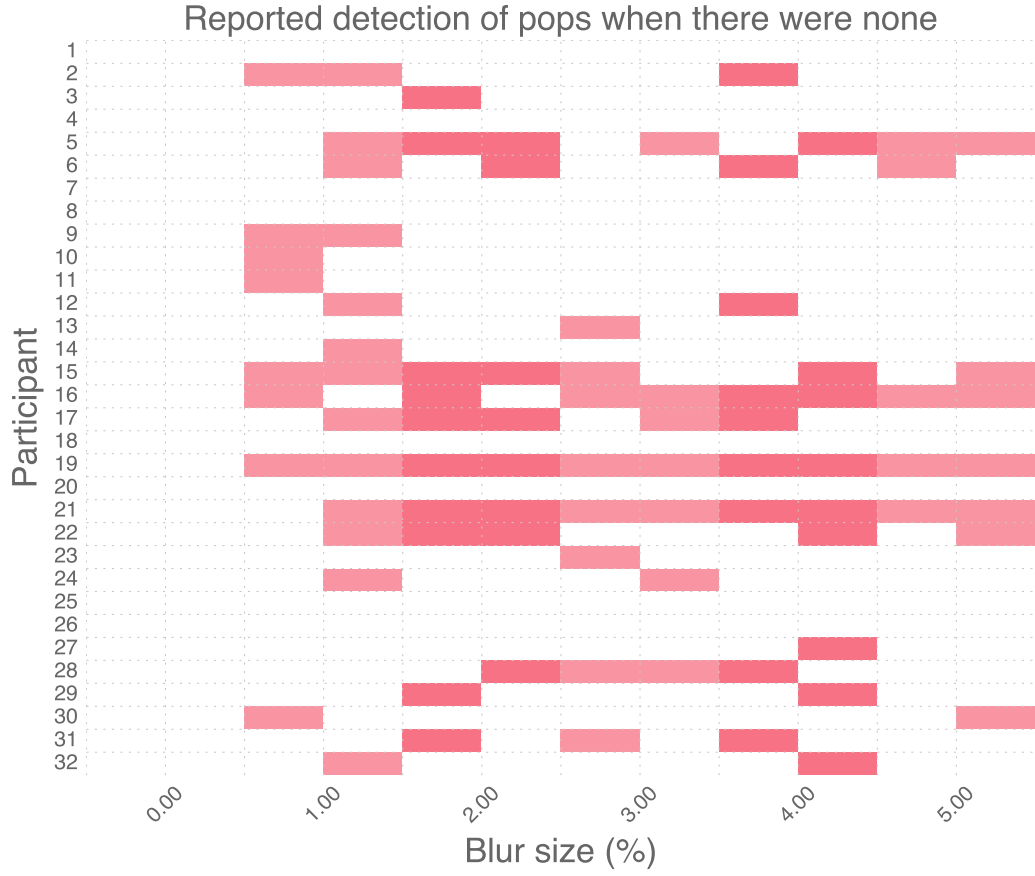


Figure 4.4: The participants answer for the control scenes that used full adaptive tessellation.

Figures 4.5, 4.6 and 4.7 show the detection rate for different levels of blur. The horizontal axis shows the blur size and the vertical axis shows the detection rate for that blur size in percent. The error bars show a 95% confidence interval for each blur size. The dashed line show a quadratic regression fit with the detection rate as dependant variable. The data gathered from participant number 20 was not used for these graphs since the participant did not provide any input during the experiment.

Figure 4.5 show these results among all 31 participants with 62 data points for each level of blur. The quadratic regression yields an R^2 value of 0.836.

Figure 4.6 shows the results obtained when removing all participants who said they saw a pop in 50 % or more of the control scenes where there were no pops. When removing these participants 26 participants are left resulting in 51 data points for each level of blur. This yields an R^2 value of 0.878 which is a slightly better fit for a quadratic model compared to using all participants data.

Figure 4.7 shows the results obtained when being even more selective and re-

4.2. DETECTION OF POPS

moving participants with an error rate above 10%. After these have been removed 14 participants are left resulting in 28 data points per blur level. This filtering results in an R^2 value of 0.915 which is an even better fit for a quadratic model.

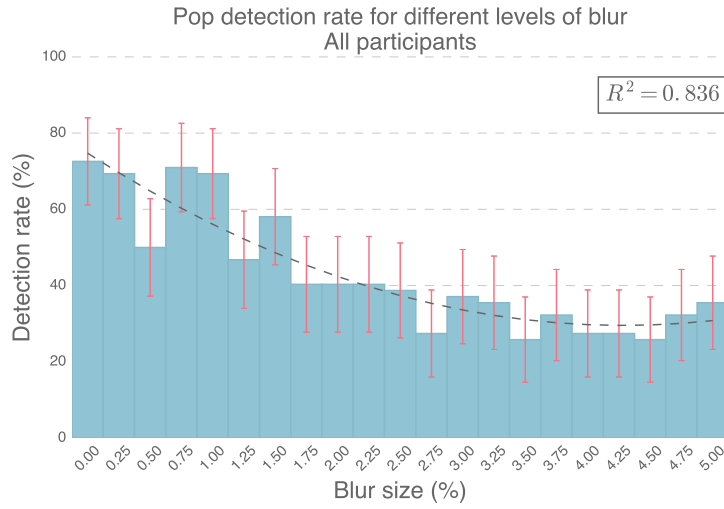


Figure 4.5: The percentage of times a pop was seen for different levels of blur among all participants.

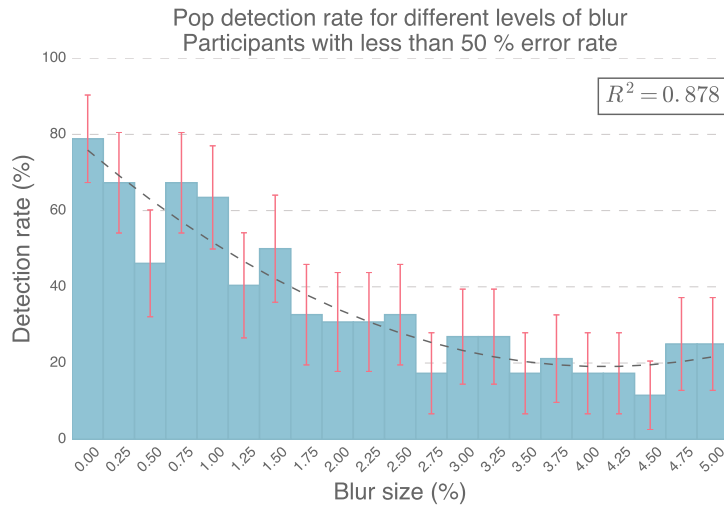


Figure 4.6: The percentage of times a pop was seen for different levels of blur among participants who saw a pop in 50% or less of the control scenes.

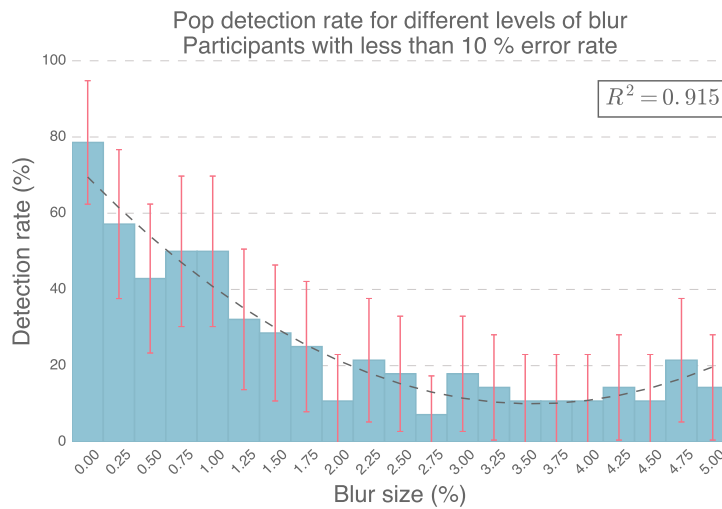


Figure 4.7: The percentage of times a pop was seen for different levels of blur among participants who saw a pop in 10% or less of the control scenes.

The aggregated answers in order of appearance can be seen in figure 4.8. The horizontal axis shows the tested levels of blur in the order in which they were shown to the participants. The vertical axis shows the detection rate for each blur value. The error bars shown are 95% confidence intervals. A linear regression test was applied to the data which is represented by the dotted line. The slope for the regression is not statistically significant. The data was gathered among the same 31 participants as for figure 4.5.

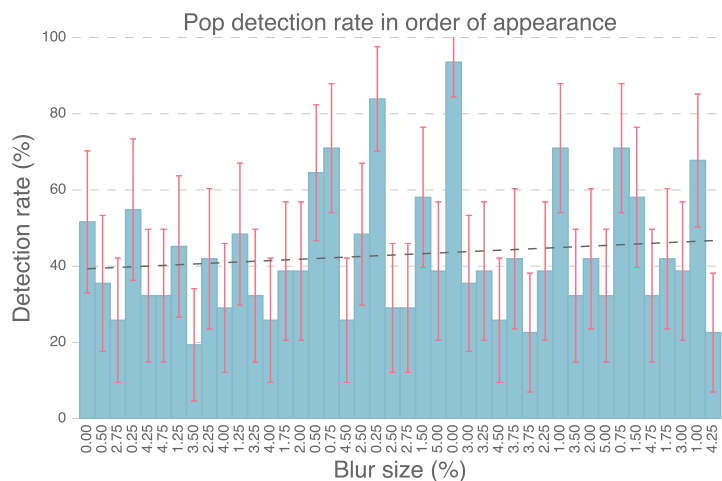


Figure 4.8: The percentage of times a pop was seen for different levels of blur in the order that they were shown to the participants.

4.2. DETECTION OF POPS

A heat map showing the aggregated gaze position among all participants can be seen in figure 4.9. The color signifies the amount of gaze points that fall within a particular area with blue meaning a few gaze points and red meaning many gaze points.

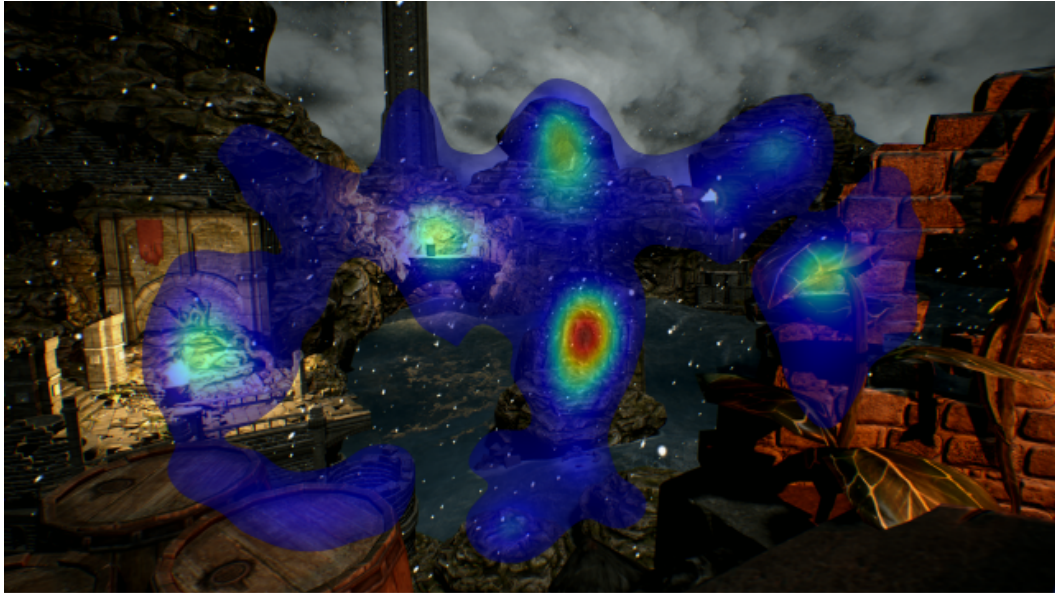


Figure 4.9: Heat map showing the aggregated gaze position among all participants.

Chapter 5

Discussion

In the Discussion chapter the results presented in chapter 4 are discussed in order to answer the research question posed in 1.1. Section 5.1 discusses the results regarding performance and section 5.2 discusses the noticeability results of the user study. Section 5.3 discusses some of the decisions made with regards to the methodology and the user study and suggests some improvements. Lastly, section 5.4 takes up some important aspects regarding ethics and sustainability.

5.1 Performance

As can be seen in figure 4.1 the GC DOF tessellation technique reduces the number of primitives that have to be rendered by a significant amount, around 70 % on average. This number will however vary greatly depending on where the current gaze position resides, looking at some parts of the screen will result in a large number of triangles and some parts in very few triangles. The maximum amount of primitives rendered was almost four times as large as the minimum value. This large variation can be a detriment since one optimally wants to execute the same amount of calculations each frame to ensure a smooth frame rate. The variation can depend on the way the scene is constructed, the placement of objects and their triangle counts. Using regular discrete LOD techniques in combination with this technique could perhaps reduce the variation to a degree by ensuring that each level of depth has similar triangle count. Of course this will not always be possible since some areas can be more rich in detail than others. Compared to using full adaptive tessellation the decrease is still significant even in the worst case.

The meshes used in the experiment scene were not constructed with this technique in mind, they were originally used with regular LOD techniques and no tessellation. If the meshes were constructed for this technique specifically the performance improvement could perhaps be increased. Some of the meshes used in the scene had a higher triangle count than necessary, especially in objects far away from the camera. Another aspect that will affect performance is the depth in the scene. The technique will not perform as well for scenes with a more shallow depth such as

indoor environments.

The frame time improvement amounts to a 9 % increase on average which can be seen in figure 4.2. As with the primitive count this can vary depending on where in the scene the user is looking. The NVIDIA GTX 970 and 980 GPU:s have very high tessellation performance. It is therefore likely that a larger performance increase can be had when using older graphics cards or cards from AMD that don't perform as well with high levels of tessellation. Measuring performance in terms of frame time can be misleading since the frame time can depend on many factors. Some setups will probably see a larger performance increase while some setups will see a smaller one.

5.2 Noticeability

Figure 4.5 shows an inverse correlation between detection of pops and blur size. The difference between no blur and blur sizes higher than 1.75 is statistically significant showing that using gaze contingent DOF can be used to hide pops caused by tessellation. Filtering out participants who had high error rates (reporting pops when none occurred) yields a greater difference as can be seen in figures 4.6 and 4.7. When filtering out participants with over 50 % error one can see a statistically significant difference at a blur size of 1.25. For the less forgiving level of filtering seen in figure 4.7 the decrease in detection is even more steep.

A quadratic model was fit to the data and results in relatively high R^2 -values, suggesting that a quadratic model is a good fit for the data. It seems reasonable that the detection rate would at first decrease with the level of blur and then rise again for extreme levels of blur since these levels can cause significant changes in the appearance of the object.

The number of positive answers given in the control scenes seen in figure 4.4 suggest that it was difficult to understand the experiment and what they as participants were supposed to look for. The difference between an object changing shape due to changes in geometry compared to transitioning from blurry to sharp was difficult to explain.

Figure 4.8 seems to suggest a slight increase in the pop detection rate as the experiment progressed. Though the slope in the linear regression fit is not statistically significant at a 95% level it would make sense that participants got better at detecting pops as the experiment progressed. This introduces a slight bias since the scenes were not presented in a randomised order. It especially seems true for the first few scenes. Comparing the first time the participants were shown a scene with blur size 0.0 with the second time yields a statistically significant difference. Around 52% of the participant reported seeing a pop the first time they were shown this level of blur (which was the very first scene of the experiment) compared to around 94% the second time. The same goes for blur size 0.5 which was shown in the second scene and the 16:th scene where 35% saw a pop the first time and 65% the second time. This helps explain the difference seen in figures 4.5, 4.6 and 4.7

5.3. METHODOLOGY

for blur values 0.0, 0.5, 1.25 and 2.75. This difference could be due to participants feeling uncertain as to what constitutes a pop in the beginning of the experiment. It could also be because of participants feeling tired or bored and wanting the experiment to be over with, resulting in less attention paid towards the end of the experiment.

Generalising these findings could prove difficult since many factors can affect the results. Things like object shape and lighting conditions within the scene could make a large difference. Some objects will change more noticeably than others when tessellation is applied. Studying the heat map in figure 4.9 shows that many people looked at the rock in the middle of the screen and at the plant in the foreground. This is likely due to pops being more easily detected in these objects than others. Almost no one looked at the water which suggests that pops that occurred due to the water changing tessellation level were difficult to spot. A style with higher contrasts could likely make it easier to spot changes in an object. The technique might therefore not be as applicable in all settings.

In general the results seem to indicate that the method could be used to conceal pops in modern game settings. However, further research would have to be conducted to surely answer this question. While this experiment indicate that a blur size at around 1.25 – 1.75 would make a difference it remains to be seen whether this level of blur is acceptable to the end user and whether it improves visual quality compared to doing no tessellation at all.

5.3 Methodology

If time and resources allowed, the user study would have been conducted anew. Not randomising the order of the scenes introduced an obvious bias to lower detection rates for values that appeared early in the experiment. Using a random order would have ensured no bias towards any levels of blur due to ordering. Another way of alleviating this issue would have been to introduce test scenes before the actual experiment started. This would have allowed the participants to get used to the experiment and their task before recording any data. This would probably have resulted in a higher detection rate for low blur values.

Another aspect of the user study that could be improved is the explanation of the task. It was difficult to explain the difference between a pop occurring due to tessellation and a pop occurring due to blur changes. Introducing a longer explanation with a video and further explanation of how tessellation and DOF works might have helped the participants understanding of the experiment. It would also be interesting to conduct a study more focused on participants with a previous experience in gaming and 3D graphics.

5.4 Ethics and sustainability

Careful considerations were taken regarding the ethical aspects of the user study. Every participant had to read a consent form stating the participants rights, detailing the confidentiality aspects of the experiment and how the data gathered from each participant was handled. It was emphasised that the participant could choose to withdraw their participation at any time and that the participants could ask to have their data removed even after the experiment. Other than this the project and thesis contain no ethical aspects worth mentioning.

Since this technique deals with optimisation one could argue that a sustainability perspective exists. Improving the efficiency of the rendering process could let the GPU run at a lower clock rate which consumes less electricity. Increasing performance could also lead to older graphics cards being able to run applications that they otherwise could not meaning that people do not have to upgrade their graphics cards as often. One could therefore say that the technique is environmentally friendly.

Chapter 6

Conclusion

In this thesis a novel foveated rendering technique called *gaze contingent depth of field tessellation* (GC DOF tessellation) is presented. The experiment conducted showed that using this technique the amount of primitives that needed to be rendered could be reduced by up to 70% on average compared to using full adaptive tessellation. The frame time could be improved by around 9% in the tested scene using the specific hardware. Performance gains will however most likely vary depending on the scene, objects and meshes used. Depending on the current gaze position of the user these number will vary greatly meaning that performance gains can fluctuate between frames. This can be a detriment since 3D graphics applications should optimally run at a constant frame rate. However, even the largest amount of primitives that needed to be rendered using GC DOF tessellation was significantly fewer compared to using full adaptive tessellation.

A user study was conducted with 32 participants which showed that the technique can be used to conceal pops. Using a eye tracker which runs at around 60 Hz most participants said that they saw pops when using no level of blur. As the level of blur increased the detection rate sank. At a blur level of around 1.75 a statistically significant difference could be seen. When only looking at the results from participants who reported few to no pops in the control scenes this difference occurs even earlier, at a blur size of 1.25. The decrease in detection rate with blur size is also a lot steeper when looking at the data from these participants.

While these results seem promising it is difficult to claim with certainty that the technique is viable for concealing pops. Some errors in the way the user study was conducted introduced a slight bias in the data which makes the results less reliable. It is also clear that not all participants understood their task which skewed the results of the experiment to a degree.

6.1 Future work

Further research is needed to solidify these findings. While the user study conducted in this thesis show that GC DOF tessellation has promise, conducting a new

user study with the changes proposed in section 5.3 would be good. Filtering out participants with large error rates yielded much more promising results. It would therefore be interesting to conduct a user study that is targeted more at people with experience of video games and 3D graphics since explaining the details of the experiments might prove easier to such participants. Finding participants with that type of background knowledge could prove difficult however.

It remains to see whether the technique is actually preferable to doing no tessellation at all. This could also be evaluated through a user study where the participants are shown the same scene with and without GC DOF tessellation and letting them rate the visual quality of the scenes. It would also be interesting to find the smallest value of blur that is acceptable since previous research indicates that using too much blur is distracting. This study indicated that a blur size of 1.25 – 1.75 decreases the noticeability of pops but whether this is an acceptable level of blur remains to be seen.

The user study was conducted on a system with relatively high latency, using an eye tracker and screen that ran at 60 Hz, where the application ran at 45 frames per second. For other foveated rendering techniques this has been shown to be too slow of an update frequency. Evaluating the technique in a system with lower latency would therefore be interesting.

Evaluating the effect of gaze contingent DOF in combination with other foveated rendering techniques would also be interesting. Multi-resolution foveated rendering has very high and consistent performance gains and the use of anti-aliasing techniques on the low resolution areas of the screen already result in blurring. It is therefore possible that gaze contingent DOF could help to hide pops there as well.

Bibliography

- [1] BARSKY, B. A., AND KOSLOFF, T. J. Algorithms for rendering depth of field effects in computer graphics. *Proceedings of the 12th WSEAS International Conference on Computers* (2008), 999–1010.
- [2] BOUBEKEUR, T., AND ALEXA, M. Phong tessellation. *ACM Transactions on Graphics* 27, 5 (2008), 1–5.
- [3] CANTLAY, I. DirectX 11 terrain tessellation. Tech. rep., NVIDIA, 2011.
- [4] CATMULL, E., AND CLARK, J. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design* 10, 6 (1978), 350–355.
- [5] CATMULL, E. E. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, University of Utah, 1974.
- [6] CLARK, J. H. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM* 19, 10 (1976), 547–554.
- [7] COGAIN. Eye data quality (EDQ) standardisation project. <http://www.cogain.org/info/eye-data-quality>, 2014. [Online; accessed 2016-03-18].
- [8] COOK, R. L. Shade trees. *ACM SIGGRAPH Computer Graphics* 18, 3 (1984), 223–231.
- [9] CRANE, H. D., AND STEELE, C. M. Generation-V dual-Purkinje-image eyetracker. *Applied optics* 24, 4 (1985), 527.
- [10] DECORO, C., AND TATARCHUK, N. Real-time mesh simplification using the GPU. *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games* (2007), 161–166.
- [11] DOO, D., AND SEBIN, M. Behaviour of recursive division surfaces near extraordinary points. *Computer Aided Design* 10, 6 (1978), 356–362.
- [12] DUCHOWSKI, A. T. *Eye Tracking Methodology: Theory and Practice*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

BIBLIOGRAPHY

- [13] DUCHOWSKI, A. T., AND ÇÖLTEKIN, A. Foveated gaze-contingent displays for peripheral LOD management, 3D visualization, and stereo imaging. *ACM Transactions on Multimedia Computing, Communications, and Applications* 3, 4 (2007), 1–18.
- [14] DUCHOWSKI, A. T., HOUSE, D. H., GESTRING, J., WANG, R. I., KREJTZ, K., KREJTZ, I., MANTIUK, R., AND BAZYLUK, B. Reducing visual discomfort of 3D stereoscopic displays with gaze-contingent depth-of-field. *Proceedings of the ACM Symposium on Applied Perception* (2014), 39–46.
- [15] ENDERLE, J. *Models of Horizontal Eye Movements: Early models of saccades and smooth pursuit*. Morgan & Claypool, 2010.
- [16] EPIC GAMES. Depth of field. <https://docs.unrealengine.com/latest/INT/Engine/Rendering/PostProcessEffects/DepthOfField/>, 2015. [Online; accessed 2016-05-16].
- [17] EPIC GAMES. Infinity blade: Grass lands. <https://www.unrealengine.com/marketplace/infinity-blade-plain-lands>, 2015. [Online; accessed 2016-05-19].
- [18] EPIC GAMES. Material parameter collections. <https://docs.unrealengine.com/latest/INT/Engine/Rendering/Materials/ParameterCollections/>, 2015. [Online; accessed 2016-05-16].
- [19] EPIC GAMES. Tessellation. https://docs.unrealengine.com/latest/INT/Resources/ContentExamples/MaterialProperties/1_8/, 2015. [Online; accessed 2016-05-16].
- [20] FUJITA, M., AND HARADA, T. Foveated real-time ray tracing for virtual reality headset. Tech. rep., Light Transport Entertainment Research, 2013.
- [21] GUENTER, B., FINCH, M., DRUCKER, S., TAN, D., AND SNYDER, J. Foveated 3D graphics. *ACM Transactions on Graphics* 31, 6 (2012), 1–10.
- [22] HARRIS, M. NVIDIA GeForce GTX 980 featuring Maxwell, the most advanced GPU ever made. Tech. rep., NVIDIA, 2014.
- [23] HILLAIRES, S., LÉCUYER, A., COZOT, R., AND CASIEZ, G. Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments. *Proceedings - IEEE Virtual Reality Conference* (2008), 47–50.
- [24] HOLMQVIST, K., NYSTROM, M., ANDERSSON, R., DEWHURST, R., JARODZKA, H., AND VAN DE WEIJER, J. *Eye Tracking. A comprehensive guide to methods and measures*. Oxford University Press, 2011.

BIBLIOGRAPHY

- [25] HOLMQVIST, K., NYSTRÖM, M., AND MULVEY, F. Eye tracker data quality: What it is and how to measure it. *Proceedings of the Symposium on Eye Tracking Research and Applications* (2012), 45–52.
- [26] HOPPE, H. Progressive meshes. *Proceedings of the 23rd annual conference on Computer Graphics and Interactive Techniques* (1996), 99–108.
- [27] HOPPE, H. View-dependent refinement of progressive meshes. *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (1997), 189–198.
- [28] HU, L., SANDER, P. V., AND HOPPE, H. Parallel view-dependent refinement of progressive meshes. *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games* (2009), 169–176.
- [29] HU, M. C., TSENG, Y. N., WU, J. L., KUO, C. H., HSIAO, Y. M., AND HUA, K. L. Real-time depth of field rendering with bokeh effect. *Proceedings of the International Symposium on Consumer Electronics, ISCE* (2013), 99–100.
- [30] KIMMEL, D. L., MAMMO, D., AND NEWSOME, W. T. Tracking the eye non-invasively: Simultaneous comparison of the scleral search coil and optical tracking techniques in the macaque monkey. *Frontiers in Behavioral Neuroscience* 6, 49 (2012), 1–17.
- [31] LEVOY, M., AND WHITAKER, R. Gaze-directed volume rendering. *ACM SIGGRAPH Computer Graphics* 24, 2 (1990), 217–223.
- [32] LIU, S., AND HUA, H. Spatialchromatic foveation for gaze contingent displays. *Proceedings of the 2008 symposium on Eye tracking research & applications* (2008), 139–142.
- [33] LOOP, C. Smooth subdivision surfaces based on triangles. Ma. thesis, Department of Mathematics, University of Utah, 1987.
- [34] LOPEZ, F., MOLLA, R., AND SUNDSTEDT, V. Exploring peripheral LOD change detections during interactive gaming tasks. *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization* (2010), 73–80.
- [35] LOSCHKY, L. C., LOSCHKY, L. C., WOLVERTON, G. S., AND WOLVERTON, G. S. How late can you update gaze-contingent multiresolutional displays without detection? *ACM Transactions on Multimedia Computing, Communications, and Applications* 3, 4 (2007), 1–10.
- [36] LOSCHKY, L. C., AND MCCONKIE, G. W. User performance with gaze contingent multiresolutional displays. *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications* 6, 10 (2000), 97–103.

BIBLIOGRAPHY

- [37] LUEBKE, D., AND ERIKSON, C. View-dependent simplification of arbitrary polygonal environments. *Proceedings of the 24th annual conference on Computer Graphics and Interactive Techniques* (1997), 199–208.
- [38] LUEBKE, D., WATSON, B., COHEN, J. D., REDDY, M., AND VARSHNEY, A. *Level of Detail for 3D Graphics*. Elsevier Science Inc., New York, NY, USA, 2002.
- [39] MANTIUK, R., BAZYLUK, B., AND MANTIUK, R. K. Gaze-driven object tracking for real time rendering. *Computer Graphics Forum* 32, 2 (2013), 163–173.
- [40] MANTIUK, R., BAZYLUK, B., AND TOMASZEWSKA, A. Gaze-dependent depth-of-field effect rendering in virtual environments. *Proceedings of the Second International Conference on Serious Games Development and Applications* (2011), 1–12.
- [41] MANTIUK, R., AND JANUS, S. Gaze-dependent ambient occlusion. *Advances in Visual Computing* (2012), 523–532.
- [42] MARCOS, S., MORENO, E., AND NAVARRO, R. The depth-of-field of the human eye from objective and subjective measurements. *Vision Research* 39, 12 (1999), 2039–2049.
- [43] MATHER, G. The use of image blur as a depth cue. *Perception* 26, 9 (1997), 1147–1158.
- [44] MAUDERER, M., CONTE, S., NACENTA, M. A., AND VISHWANATH, D. Depth perception with gaze-contingent depth of field. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems* (2014), 217–226.
- [45] MICROSOFT. Direct3D 11 graphics.
[https://msdn.microsoft.com/en-us/library/ff476080\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ff476080(v=vs.85).aspx), 2009. [Online; accessed 2016-03-18].
- [46] MICROSOFT. D3D11_QUERY_DATA_PIPELINE_STATISTICS.
[https://msdn.microsoft.com/en-us/library/windows/desktop/ff476192\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ff476192(v=vs.85).aspx), 2016. [Online; accessed 2016-05-19].
- [47] MURPHY, H. A., DUCHOWSKI, A. T., AND TYRRELL, R. A. Hybrid image/model-based gaze-contingent rendering. *ACM Transactions on Applied Perception* 5, 4 (2009), 1–21.
- [48] NGUYEN, H. *GPU Gems 3*. Addison-Wesley Professional, 2007.
- [49] NIESSNER, M., LOOP, C., MEYER, M., AND DEROSE, T. Feature-adaptive GPU rendering of Catmull-Clark subdivision surfaces. *ACM Transactions on Graphics* 31, 1 (2012), 1–11.

BIBLIOGRAPHY

- [50] NOVAK, D. Engineering issues in physiological computing. *Advances in Physiological Computing* (2014), 17–38.
- [51] NUNES, G., BRAGA, R., VALDETARO, A., RAPOSO, A., AND FEIJÓ, B. Analysis and implementation of local subdivision algorithms in the GPU. *2011 Brazilian Symposium on Games and Digital Entertainment* (2011), 101–113.
- [52] OHSHIMA, T., YAMAMOTO, H., AND TAMURA, H. Gaze-directed adaptive rendering for interacting with virtual space. *Proceedings of the IEEE 1996 Virtual Reality Annual International Symposium* (1996), 103–111.
- [53] PETERS, C., AND QURESHI, A. A head movement propensity model for animating gaze shifts and blinks of virtual characters. *Computers and Graphics (Pergamon)* 34, 6 (2010), 677–687.
- [54] POHL, D., XUCONG, Z., AND BULLING, A. Combining eye tracking with optimizations for lens astigmatism in modern wide-angle HMDs. *Proceedings of the IEEE Conference on Virtual Reality* (2016), 1–2.
- [55] SCHERFFIG, L. *It's in Your Eyes – Gaze Based Image Retrieval in Context*. ZKM | Institute for Basic Research, Karlsruhe, 2005.
- [56] SCHÄFER, H., NIESSNER, M., KEINERT, B., STAMMINGER, M., AND LOOP, C. State of the art report on real-time rendering with hardware tessellation. *Eurographics* (2014), 93–117.
- [57] SCHÖN, M. An evaluation of interactors gaze-to-object mapping performance in 3D virtual environments. Ma. thesis, School of Computer Science and Communication, Royal Insititute of Technology, 2016.
- [58] ŠPAKOV, O. Comparison of eye movement filters used in HCI. *Proceedings of the Symposium on Eye Tracking Research and Applications* (2012), 281–284.
- [59] STAM, J. Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (1998), 395–404.
- [60] STENGEL, M., GROGORICK, S., EISEMANN, M., EISEMANN, E., AND MAGNOR, M. A. An affordable solution for binocular eye tracking and calibration in head-mounted displays. *Proceedings of the 23rd ACM International Conference on Multimedia* (2015), 15–24.
- [61] STRASBURGER, H., RENTSCHLER, I., AND JÜTTNER, M. Peripheral vision and pattern recognition: A review. *Journal of vision* 11, 5 (2011), 13.
- [62] SWAFFORD, N. T., COSKER, D., AND MITCHELL, K. Latency aware foveated rendering in Unreal Engine 4. *Proceedings of the 12th European Conference on Visual Media Production* (2015), 1–1.

BIBLIOGRAPHY

- [63] TOBII AB. Developer’s guide Tobii EyeX SDK for C/C++. <http://developer-files.tobii.com/wp-content/uploads/2016/03/Developers-Guide-C-Cpp.pdf>, 2015. [Online; accessed 2016-05-16].
- [64] TOBII AB. Tobii EyeX Controller. http://mb.cision.com/Public/2874/9885313/aeb44c161f2532b2_org.jpg, 2015. [Online; accessed 2016-06-09].
- [65] VLACHOS, A., PETERS, J., BOYD, C., AND MITCHELL, J. L. Curved PN triangles. *Proceedings of the 2001 Symposium on Interactive 3D Graphics* (2001), 159–166.
- [66] WIKIMEDIA COMMONS, *Brion*. Cirles of confusion lens diagram. https://commons.wikimedia.org/wiki/File:Cirles_of_confusion_lens_diagram.png, 2005. [Online; accessed 2016-06-03].
- [67] WIKIMEDIA COMMONS, *PiccoloNamek*. DOF shallow depth of field. <https://commons.wikimedia.org/wiki/File:DOF-ShallowDepthofField.jpg>, 2005. [Online; accessed 2016-06-03].
- [68] WIKIMEDIA COMMONS, *Rhcastilhos*. Schematic diagram of the human eye. https://commons.wikimedia.org/wiki/File:Schematic_diagram_of_the_human_eye_en.svg, 2007. [Online; accessed 2016-05-31].
- [69] WIKIMEDIA COMMONS, *Romainbehar*. Catmull-Clark subdivision of a cube. https://commons.wikimedia.org/wiki/File:Catmull-Clark_subdivision_of_a_cube.svg, 2006. [Online; accessed 2016-05-31].
- [70] WIKIMEDIA COMMONS, *Z22*. Diagram of four Purkinje images. https://commons.wikimedia.org/wiki/File:Diagram_of_four_Purkinje_images.svg, 2015. [Online; accessed 2016-05-31].
- [71] XIA, J. C., EL-SANA, J., AND VARSHNEY, A. Adaptive real-time level-of-detail-based rendering for polygonal models. *IEEE Transactions on Visualization and Computer Graphics* 3, 2 (1997), 171–183.
- [72] YARBUS, A. L. *Eye movements and vision*. Plenum Press, New York, NY, USA, 1967.

Appendix A

Further details of the user study

The user study was performed using a computer with the following specifications:

General:

OS : Windows 10.0.10586 Build 10586

Game Engine : Unreal Engine 4.11

Eye tracker hardware and software:

Eye tracker model : Tobii EyeX Controller

Firmware version : 2.0.2-33638

Tobii EyeX Controller Core Version : 2.0.9

Tobii EyeX Controller Driver Version : 2.0.4

Tobii Service Version : 1.9.0.6164

Tobii EyeX Engine Version : 1.9.0.6164

Tobii EyeX Config Version : 3.2.7.366

Tobii EyeX Interaction Version : 2.0.4.2420

Hardware:

Graphics Card : NVIDIA GeForce GTX 970

CPU : Intel Xeon E5-1650 v3 @ 3.50GHz

Memory : 16 GB DDR4 2133 MHz

Hard Drive : Samsung MZHPV256HDGL SSD

Monitor : Phillips 272S4L 27" 2560x1440 60 Hz

APPENDIX A. FURTHER DETAILS OF THE USER STUDY



Figure A.1: The experiment environment.

The user study took place in a private office at KTH which can be seen in figure A.1.

Appendix B

User study information sheet and consent form

This appendix contains the information sheet and consent form used in the user study. These were based on the information sheet and consent form used in [53] with only slight modifications to make them more relevant for the user study, see section 3.2.2.

Participant Information Sheet

Studying gaze-to-object mapping and foveated rendering

1. Aims and objectives of the studies

The aim of these studies is to evaluate some gaze-to-object mapping techniques and to determine the effects of gaze contingent depth of field on foveated rendering techniques. These experiments will be useful as part of related research with these aims.

2. Why have I been chosen?

For the purposes of these studies it is needed the recruitment of some participants with good vision. They must be able to watch some virtual scenes and be able follow the instructions of each experiment. These are the only criteria needed.

3. Do I have to take part?

No, the participation is voluntary. If you change your mind about taking part in the study you can withdraw at any point during the session. If you decide to withdraw all your data will be destroyed and will not be used in the study. There are no consequences if you no longer wish to participate in the study.

4. What do I have to do?

The experiments will last approximately thirty minutes. You will complete a series of trials, consisting of watching some scenes and performing some task. There will be an eye-tracker to capture where you are looking at in the screen, but you will not be recorded whilst participating in the experiment.

5. What are the risks associated with this project?

There are no risks associated with these experiments.

6. What are the benefits of taking part?

As a student, by taking part in this study you will gain an insight into how a perceptual experiment is conducted and what it is like to be a participant in such a study. You will also gain some insight into the area of eye tracking and visual perception.

7. Withdrawal options

As mentioned before, if you change your mind about taking part in the study you can withdraw at any point during the session. If you decide to withdraw all your data will be destroyed and will not be used in the study. You can also withdraw up to four weeks after participating by contacting Martin Schön or Tim Lindeberg (see contact details below) with your participant number.

8. Data protection & confidentiality

The data will be confidential. Only the researchers will have access to the raw data. All the consent forms will be stored in a separate, secure (locked) location from the raw data itself. You will only be identified by your participant code number. The raw data from these experiments will be retained until final data analyses are completed. They will then be destroyed. Your data will only be associated with your code number and access to the file will be password protected.

9. What if things go wrong? Who to complain to?

If anything goes wrong or you wish to complain about any aspect of the studies, please contact Martin Schön, Tim Lindeberg or Dr. Christopher Peters explaining the nature of your complaint. (see contact details below).

10. What will happen with the results of the study?

The results will be used in research projects.

11. Who has reviewed this study?

The studies has been reviewed by Dr. Christopher Peters.

12. Further information/Key contact details

Martin Schön (e-mail: mscho@kth.se).

Tim Lindeberg (e-mail: timlin@kth.se).

Dr. Christopher Peters (e-mail: chpeters@kth.se).

Consent Form

Studying gaze-to-object mapping and foveated rendering

The aim of these studies is to evaluate some gaze-to-object mapping techniques and to determine the effects of gaze contingent depth of field on foveated rendering techniques. These experiments will be useful as part of related research with these aims.

Please tick

1. I confirm that I have read and understood the participant information sheet for the studies and have had the opportunity to ask questions. ☐
2. I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason. ☐
3. I understand that all the information I provide will be treated in confidence. ☐
4. I agree to take part in these experiments. ☐

Name of the participant: _____

Signature of the participant: _____

Date: _____

Name of the researcher: _____

Signature of the researcher: _____

Date: _____

