Technical Report

UCAM-CL-TR-977
ISSN 1476-2986

Number 977

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Motion quality models for real-time adaptive rendering

Akshay Jindal

January 2023

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

https://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted October 2022 by the author for the degree of Doctor of Philosophy to the University of Cambridge, St Edmund's College.

Some figures in this document are best viewed in colour. If you received a black-and-white copy, please consult the online version if necessary.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

https://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Abstract

Motion quality models for real-time adaptive rendering

Akshay Jindal

The demand for compute power and transmission bandwidth is growing rapidly as the display technologies progress towards higher spatial resolutions and frame rates, more bits per pixel (HDR), and multiple views required for 3D displays. Advancement in real-time rendering has also made shading incredibly complex. However, GPUs are still limited in processing capabilities and often have to work at a fraction of their available bandwidth due to hardware constraints.

In this dissertation, I build upon the observation that the human visual system has a limited capability to perceive images of high spatial and temporal frequency, and hence it is unnecessary to strive to meet these computational demands. I propose to model the spatio-temporal limitations of the visual system, specifically the perception of image artefacts under motion, and exploit them to improve the quality of rendering.

I present four main contributions: First, I demonstrate the potential of existing motion quality models in improving rendering quality under restricted bandwidths. This validation is done using an eye tracker through psychophysical experiments involving complex motion on a G-Sync display. Second, I note that the current models of motion quality ignore the effect of displayed content and cannot take advantage of recent shading technologies such as variable-rate shading which allows for more flexible control of local shading resolution. To this end, I develop a new content-dependent model of motion quality and calibrate it through psychophysical experiments on a wide range of content, display configurations, and velocities. Third, I propose a new rendering algorithm that utilises such models to calculate the optimal refresh rate and local shading resolution given the allowed bandwidth. Finally, I present a novel high dynamic range multi-focal stereo display that will serve as an experimental apparatus for next-generation of perceptual experiments by enabling us to study the interplay of these factors in achieving perceptual realism.

Acknowledgements

First and foremost, I would like to thank my supervisor Prof. Rafal K. Mantiuk, for his constant support and guidance throughout the past three and a half years. I also owe a debt of gratitude to Prof. Karol Myszkowski, who hosted and mentored me during my secondment at Max-Planck-Institut für Informatik. I am also grateful to my first-year assessors, Prof. Peter Robinson and Prof. John Daugman, and my final examiners, Dr Cengiz Öztireli and Prof. Guillaume Lavoué, for their insightful feedback on my work. Further, I would like to thank Marie Skłodowska-Curie Actions for funding my doctoral education and all the members of RealVision ITN for their guidance and the knowledge shared over several training events.

My appreciation also extends to all my past and present lab colleagues for their immense help with my research projects, erudite discussions over coffee breaks and their relaxing company on our weekend getaways. In particular, I would like to thank Gyorgy Denes, Fangcheng Zhong, Param Hanji, Krzysztof Wolski, Ali Özgür Yöntem, Minjung Kim, Aliaksei Mikhailiuk, Jingyu Liu, Dingcheng Yue, Aamir Mustafa, and Maryam Azimi.

I am deeply indebted to my mentors and supervisors who introduced me to the joys of academia and helped me grow at different stages of my life, Prof. Shrisha Rao, Dr Jaya Sreevalsan-Nair and Mangesh Kulkarni.

Finally, this dissertation would not have been possible without the constant support and boundless patience from my family and all my friends. I am forever grateful to them for always being available and providing an escape in challenging times.

Contents

Glossary

cpd cycles-per-degree.

cpt cycles-per-texel.

CSF contrast sensitivity function.

FoV field of view.

HDR high-dynamic-range.

HMD head mounted display.

HVS human visual system.

JND just-noticeable-difference.

PBR physically-based rendering.

ppd pixels-per-degree.

RSB real scene box.

SPEM smooth pursuit eye motion.

VDP visible difference predictor.

VR virtual reality.

VRR variable refresh rate.

VRS variable-rate shading.

Chapter 1

Introduction

1.1 Motivation

One of the ongoing challenges in physically-based computer graphics rendering is the creation of virtual scenes that would be in every detail indistinguishable from the real world. To this end, we have seen advances in both real-time rendering as well as display technologies. There is a strong trend towards increasing resolution, refresh rates, higher bit-depths and dynamic range in modern displays. The increasing adoption of 3D displays has further pushed these factors to their limits due to their requirements of larger field of view (FoV) and multiple viewpoints. Parallel to the development in display technologies, GPUs have also grown in computational power, enabling previously unachievable effects such as real-time global illumination and accurate motion blur rendering. Shaders have grown in complexity and are capable of achieving photo-realistic shading through their use of physically-based rendering (PBR) materials.

However, increasing the fidelity in one respect usually comes at the cost of another. For example, VR headsets have to use simple shaders to achieve desired frame rate and resolution and mobile devices need to limit their brightness to satisfy power budgets. This trade-off is partially dictated by display design constraints (FoV, mobility, etc.) and partially by available GPU bandwidth. If we were to take an "ideal" VR display that has 8K resolution, 120 Hz refresh rate, 12 bits per colour channel, and 6 focal planes for each eye, we will require a bandwidth of 573 GB/s. In comparison, even the most powerful GPUs today support rendering only up to 17 GB/s. Achieving the required rendering bandwidth is out of reach even for the next few generations of GPUs.

Fortunately, we do not need to strive to achieve such high fidelity because human visual system (HVS) has several limitations. Consider visual information as a spatio-temporal signal. As this signal goes through our visual system, it loses information due to the imperfections in eye's optics. It is then received and sampled by a limited number of rods and cones distributed non-uniformly on our retina. The signal is subsequently encoded by our optical neurons and transmitted to the visual cortex where it is processed at multiple levels with limited attention. Such limitations, if modelled accurately, can be exploited to optimise graphics pipelines and display designs.

In this dissertation, I build upon the idea of *perceptually-motivated rendering* and propose visual models and rendering algorithms derived from spatio-temporal limitations of the visual system. These models are motivated by the introduction of adaptive rendering systems such as variable-rate shading and adaptive-sync displays and strive to optimise the

rendering quality under restricted computational budgets. I also present a novel display that combines multiple realism cues simultaneously at a high fidelity, thereby enabling a new set of perceptual experiments which will study the interplay of these factors on perceived realism.

1.2 Hypothesis

The central thesis of this work is that rendering pipelines can be optimised for quality by exploiting the spatial and temporal limitations of HVS. Thus, it is necessary to develop models of these limitations that parameterise the display, rendering and visual systems to predict perceived quality in actionable units such as JND. It is also necessary to design rendering strategies that utilise such models to reduce the computational complexity or improve rendering quality. Finally, it is crucial to further improve our understanding of the limitations of HVS by building novel displays that enable perceptual experiments on previously unstudied factors.

1.3 Structure

The following chapter briefly discusses models and limitations of the visual system, spatio-temporal artefacts on modern displays, fundamentals of subjective quality assessment and recent adaptive rendering technologies. Chapter 3 describes existing adaptive refresh rate and resolution rendering strategies and validates them in complex motion scenarios through psychophysical experiments. Chapter 4 surveys all existing studies on motion quality, identifies the gaps in the existing literature and introduces a new dataset tailored for variable-rate shading (VRS) artefacts. Chapter 5 focuses on building a content-aware model of motion quality, and Chapter 6 shows how this model can be used to maximise rendering quality under restricted bandwidth. Chapter 7 introduces a novel high dynamic range multi-focal stereo display and compares different image-based rendering strategies on this display.

In Chapters 3, 6 and 7, I also briefly present results obtained by my collaborators to better contextualise my work. In each of these chapters, the introduction will clearly highlight my contribution.

1.4 Publications and talks

Publications

1. Gyorgy Denes, **Akshay Jindal**, Aliaksei Mikhailiuk, and Rafał K. Mantiuk. "A perceptual model of motion quality for rendering with adaptive refresh-rate and resolution." *ACM Transactions on Graphics (Proc. of SIGGRAPH 2020) 39, no. 4 (2020): 133-1.*

   Contribution: Designed and conducted a psychophysical experiment to evaluate the proposed rendering method against existing work in complex motion scenarios.

2. **Akshay Jindal**, Krzysztof Wolski, Rafał K. Mantiuk, and Karol Myszkowski. "Perceptual Model for Adaptive Local Shading and Refresh Rate" *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia 2021) 40, no. 6 (2021): 280.*

   Contribution: Ran a psychophysical study on quality of image artefacts under motion, developed a content-adaptive model of judder, aliasing and blur, and designed and validated a rendering algorithm for determining optimal refresh rate and local shading resolution.

3. Fangcheng Zhong, **Akshay Jindal**, Özgür Yöntem, Param Hanji, Simon Watt and Rafal Mantiuk. "Reproducing Reality with a High-Dynamic-Range Multi-Focal Stereo Display." *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia 2021), Vol. 40, No. 6. (2021) pp.* 241.

   Contribution: Implemented a real-time light field rendering pipeline for the HDR-MFS display and automated the control of the real scene box.

4. **Akshay Jindal**, Fangcheng Zhong, Özgür Yöntem, Marek Wernikowski,Param Hanji, and Rafal Mantiuk."Perceptual Evaluation of Image-based Rendering Methods on High-Dynamic-Range Stereo Display" [*In preparation*].

   Contribution: Implemented 3 image-based rendering methods on our novel display and performed a quantitative and qualitative comparison through a psychophysical experiment.

5. Jingyu Liu, **Akshay Jindal**, Claire Mantel, Søren Forchhammer, and Rafał Mantiuk (2022)."How bright should a virtual object be to appear opaque in optical see-through AR?". *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2022.

   Contribution: Implemented the lumigraph rendering method used in the psychophysical experiment.

Talks

1. "Evolution of foveated rendering techniques". Rainbow Lab Machine Learning Reading Club, 4 June 2021.

2. "Perceptually motivated variable-rate shading". Huawei Computer Graphics and GPU Architecture Seminar., 8 July 2021.

3. "Perceptual model for adaptive local shading and refresh rate". Facebook Reality Labs, 5 November 2021; RealVision Autumn Event, 27 September 2021; Henry Morten Graphics Forum, NVIDIA, 2 February 2022.

Awards

1. Runner-up in the High-Performance Graphics 2022 Student Competition.

2. Best Pitch Award in Entrepreneurship in Technical Science Summer School, Helsingør, Denmark (August, 2019).

Chapter 2

Background

The field of perceptual graphics is concerned with the optimisation of graphics and display systems while considering the final consumer of the output of these systems: the human eye. This requires a thorough understanding of how the human visual system works and how it can be described with tractable computational models. It also requires an understanding of current hardware limitations and experimental methodologies that can map the effect of these limitations on perceived quality. Moreover, it benefits to have the knowledge of the latest display and GPU technologies that can utilise these computational models of HVS to reduce the quality degradation caused by hardware limitations. Since this dissertation heavily relies on these principles, this chapter aims to provide a quick start guide to the readers on the working of the human visual system, spatio-temporal artefacts of modern displays, subjective quality assessment methods, adaptive-sync displays, and variable rate shading technique offered by recent GPUs. It then concludes with some examples of existing literature which have successfully used these principles to reduce the computational bandwidth or to improve the quality of graphics systems.

2.1 Human visual system

Our visual system constitutes our sensory organs (eyes) and parts of the central nervous system (retina, optic nerve, and visual cortex). It gives us the ability to detect light and interpret the world around us. The insight into its working is a crucial tool to have if we were to design optimal displays and rendering algorithms. Though this topic is not usually considered in traditional computer graphics, it is well-studied both physiologically and behaviourally in vision science. While there are many ways to classify and study the visual system, in this section, I follow the taxonomy used by Wandell [1995] to provide a broad overview of different stages of HVS: encoding (transformation of light into neural signals), representation (organisation of neural pathways), and interpretation (inference of scene properties from neural image). This section is meant as a quick introduction to readers unfamiliar with the workings of the human visual system. For more in-depth coverage of the topic, I refer the readers to the seminal textbook on vision science *Foundations of Vision* [Wandell, 1995].

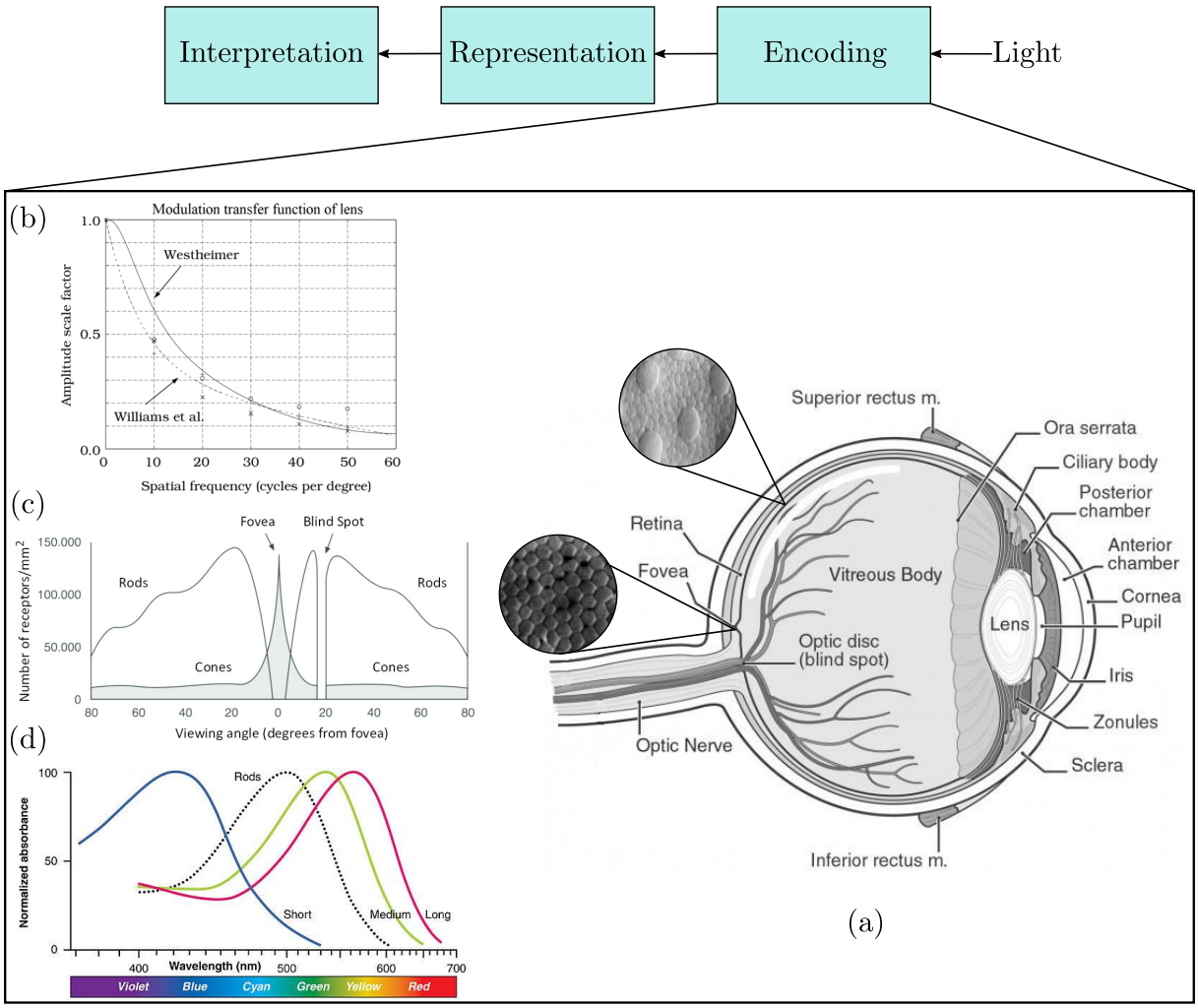


Figure 2.1: Encoding. The incoming light is focused by the eye's optics onto the retina, where it is non-uniformly sampled by rods and cones. Images adapted from [Mikrora, 2019, Schmitz, 2016, Lofgren et al., 2017, Wandell, 1995]

2.1.1 Encoding

The first stage of the visual system involves encoding incoming light by the eye. The process involves focusing the light through optical elements in the eye onto the retina consisting of light-sensitive elements called *photoreceptors* that convert the light into neural signals. The encoding stage imposes fundamental limits on our visual perception, effects of which can be found in the later stages of HVS. The following text will describe the functioning of the eye's optics and the structure of its retina.

Optics Following Figure 2.1 from right to left, light enters the eye through the *cornea*. The cornea bends the light to help the eye focus and is responsible for 2/3 of the eye's optical power [Sridhar, 2018]. Some of this light is blocked by the *pupil* (similar to a camera's aperture), and the pupil's size is controlled by the *iris* (coloured part of the eye), thereby controlling the amount of light entering the eye. The pupil can make a difference of up to 1 log unit of intensity. Though reducing the pupil size can reduce the chance of rays deviating from their path and increase image sharpness, it also reduces the amount of

energy available to activate photoreceptors on the retina and may introduce additional blur due to diffraction. Next, light passes through the *lens*. The lens and the cornea help to focus light correctly on the retina. Objects at different depths get focused at different distances behind the lens. These distances can be calculated as a function of the power of the lens using Lensmaker's equation. When we view faraway objects, the lens + cornea power is roughly 60 diopters. To focus on nearby objects, the *ciliary muscles* attached to the lens change its shape to increase its power and this process is known as *accommodation*. Another thing to note is that different wavelengths of light may also be focused at different distances leading to chromatic aberration. The empty chambers between lens and cornea and between lens and retina are filled with fluid substances called *aqueous humor* and *vitreous humor*, respectively. They help provide nutrients to different eye components and maintain its structure. They have refractive properties similar to that of water.

When the light hits the retina, some of it is reflected back. This escaping light can be captured using an ophthalmoscope to image the retina and thus measure the response of the eye's optical system. Experiments using this procedure [Campbell and Gubisch, 1966] have determined that the eye's optical system can be well modelled as a shift-invariant linear system. This finding has a very practical implication: if we measure an eye's response to a point source of light, say, a single pixel, we can calculate the complete 2D image projected on the retina through a simple convolution operation. The measured response is called the *point spread function* (PSF) of the eye. It is usually a circularly symmetric function but defects in eyes such as astigmatism can change that. When represented in terms of spatial frequency, the same function is called *optical transfer function* (OTF). OTF is a complex value function which describes the scale and phase shift the optical system induces to each spatial frequency. A plot of the magnitude of our eye's OTF (called *modulation transfer function*) is provided in Figure 2.1(b). As seen in the plot, our eye's optics does not respond equally to all frequencies and act as a low pass filter with a cut-off frequency of around 60 cycles-per-degree (cpd).

Retina The light entering through the eye is projected onto the photosensitive layer on the inner wall of the eye called the *retina*. This layer is covered with special neurons called photoreceptors that convert light into electro-chemical neural signals. There are two fundamentally different types of photoreceptors: *rods* and *cones*. Rods are activated at low illumination (*scotopic* light levels), while cones require higher light levels (*photopic* light level). Intensities at which both are active are called *mesopic* light levels. The cones are responsible for colour vision and can be further divided into three sub-types of L-, M-, and S-cones depending on the range of wavelengths they are sensitive to (Figure 2.1(c)). The spatial arrangement and the density of rods and cones vary widely over the retina and is an important factor to study as it determines how the retinal image is sampled and passed on to the subsequent stages of the visual system. There are approximately 5 million cones and 120 million rods in each eye. Rods are present in high density, but many rods converge onto a single neuron and thus have very poor visual acuity but high sensitivity. The region of highest visual acuity on the retina lies in its centre and is called the *fovea*. The fovea has no rods but a high concentration of cones. The cones are tightly packed in the fovea and form a regular pattern that allows them to sample image signals up to 60 cpd accurately. The cone density (and the visual acuity) drops rapidly as we move away from the fovea and intermix with rods in a random manner. This irregularity helps replace aliasing resulting from a low sampling rate with visually insignificant noise

[Yellott Jr, 1982].

2.1.2 Representation

When light hits the photoreceptors, it starts a series of chain reaction that carries information from the retina through a collection of different neurons called *visual streams* to the *visual cortex*, the part of the human brain dedicated to visual processing. Visual stream operates in parallel, and each stream specialises in analysing different aspects of the retinal image. The segregation of streams starts with rods and cones in the retina and continues into the cortical areas. Hundreds of rods connect directly to a single *bipolar* cell, which connects to retinal ganglion cells in the outermost layer of the retina. In contrast, very few cones converge onto one retinal ganglion cell. This organisation of neurons allows the rod pathway to be more sensitive to light at the cost of reduced acuity and the cone pathway to have higher acuity but lower sensitivity. Information is also passed laterally between retinal neurons through *horizontal* and *amacrine* cells. This complicated communication structure between retinal neurons is responsible for a significant portion of visual processing. It leads to the formation of complex receptive fields such as centre-surround or on-off fields, which allows the cells to adjust their sensitivity to edges, colour, and motion.

Signals from retinal ganglion cells are collected and carried by the *optic nerve* to the *lateral geniculate nucleus*(LGN) in the thalamus (Figure 2.2). Within this pathway, two well-studied streams include parvocellular and magnocellular pathways. The parvocellular pathway connects ganglion cells to the upper layers of the LGN (P layers), containing neurons with small cell bodies. They respond to colour, fine details, and still or slow-moving objects. Magnocellular pathways connect ganglion cells to two deeper layers of the LGN (M layers) comprising of neurons with larger cell bodies which respond to objects in motion. These layers connects to different areas of the *primary visual cortex*, also called ventral area *V1*. Neurons in the visual cortex are more complex than retinal neurons and respond asymmetrically to different edge orientations, motion directions, and binocular disparity. Roles of different neurons are also non-uniformly distributed, with 30% of the primary visual cortex responsible for the central 5° of the visual field. At the same time, the periphery is under-represented [Horton and Hoyt, 1991]. Neural representation becomes increasingly more complex as the information flows from the primary visual cortex through a visual hierarchy in V2, V3, V4, V5 and the visual association cortex via the *dorsal* and *ventral streams*. The neurons in the dorsal stream are involved in interpreting spatial information such as the location and motion of the objects. In contrast, neurons in the ventral stream help determine object characteristics such as colour and shape. It should be noted that the visual system does not create new streams as the visual task or imaging condition changes. The visual neurons instead change their sensitivity to light stimulation in response to imaging conditions by a process called *visual adaptation*. One of the most salient example of this phenomenon is *luminance adaptation* which decreases (or increases) the light sensitivity of individual neurons in the visual pathway in response to increasing (or decreasing) illumination levels. This extends the sensing abilities of our eyes to nine orders of magnitude, allowing us to clearly see both a dark starry sky and a bright sun. Change in sensitivity of the visual pathway aims to achieve a constant representation of image *contrast* instead of absolute light levels. Image contrast is the ratio of local intensity and the mean image intensity. Preserving image contrast improves our ability to distinguish objects in both very bright and dark environments.

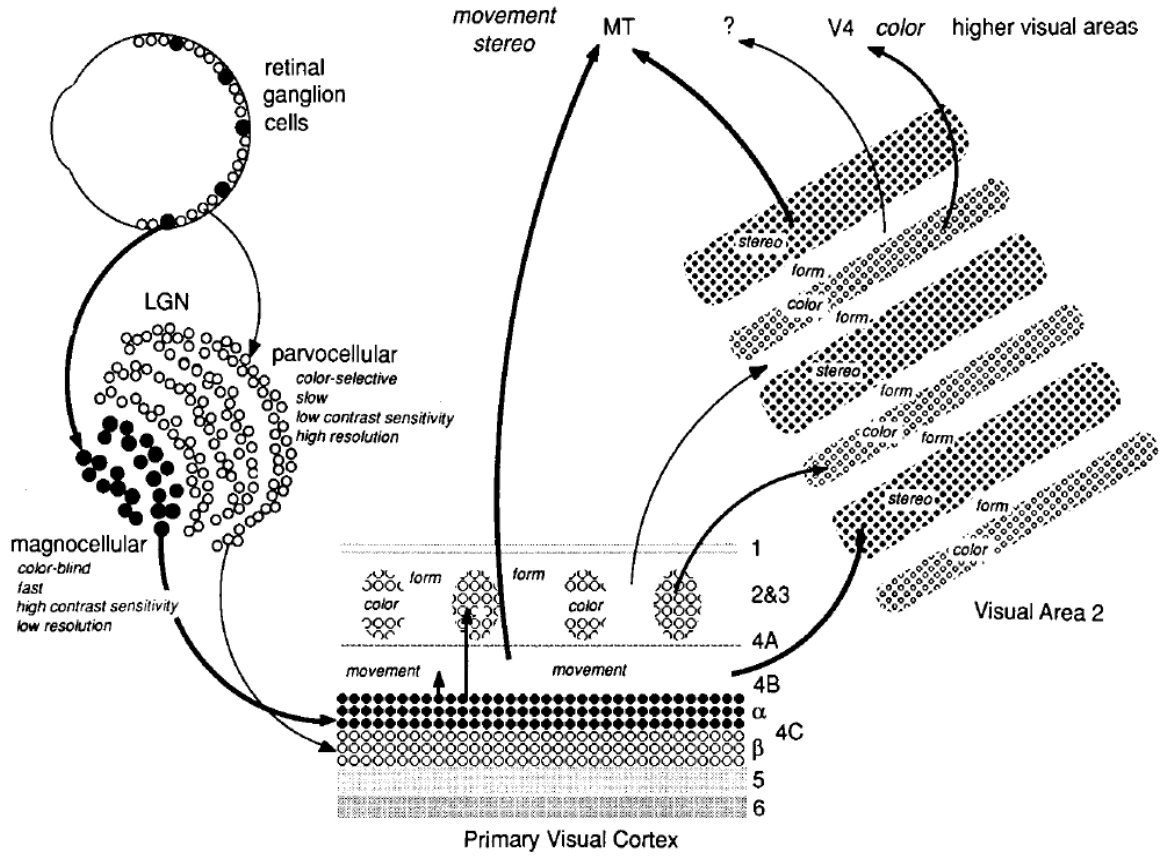


Figure 2.2: Representation. Information from the retina is carried to different parts of the visual cortex by various visual streams. Image from [Livingstone and Hubel, 1988]

Contrast sensitivity function The activity of all the neurons in the HVS can be summarised by an abstract psychophysical construct called a *neural image*. A neural image contains all the information available to an observer after all the neural transformations have been applied to the real image. Knowledge of the neural transformations and the corresponding neural image can help us optimise how we store and display visual media. One way to determine these neural transformations is to measure the responses of all individual neurons and the connections between them. This is obviously an extremely invasive and impractical procedure. An alternate, more practical approach is to study the behaviour of the visual system when presented with detection and discrimination tasks on spatio-temporal contrast patterns. The results from these experiments are used to develop models of contrast sensitivity called *contrast sensitivity functions* (CSFs) that predict the magnitude of contrast necessary to detect a pattern described by the parameters of the CSF. A CSF is a high dimensional function because our perception depends on several factors such as the pattern's spatial and temporal frequency, size, colour, eccentricity, background luminance, and orientation. Figure 2.3 plots contrast sensitivity (reciprocal of contrast threshold) w.r.t. to some of these parameters. Notice how the predictions of CSF closely match the anatomy of the visual system. Our sensitivity to high spatial frequencies is drastically reduced (Figure 2.3(a)), similar to the MTF of our lens (Figure 2.1(b)). Similarly, our sensitivity rapidly goes down with increasing eccentricity (Figure 2.3(c)) matching the distribution of cones on our retina (Figure 2.1(c)). This decrease in sensitivity

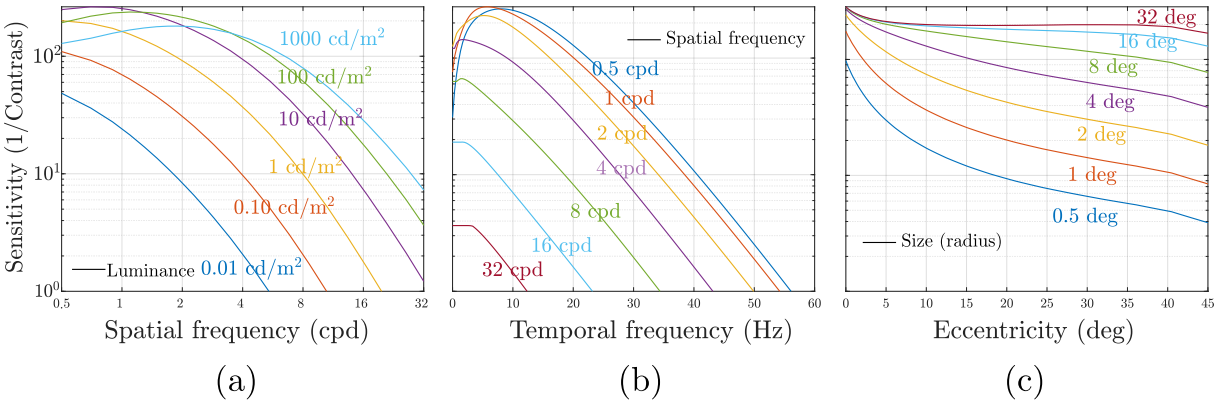


Figure 2.3: Five dimensions of a contrast sensitivity function [Mantiuk et al., 2021]. When not varying, the parameters were set to $\rho = 3\,\text{cpd}$, $\omega = 3\,\text{Hz}$, $L = 100\,\text{cd/m}^2$, eccentricity = 0 deg, and stimulus radius = 1 deg.

with eccentricity is reversed by increasing the stimulus's size, similar to how a higher number of neurons in the visual cortex results in improved image quality (called cortical magnification [Horton and Hoyt, 1991]). CSF also predicts our sensitivity to temporally-varying (flickering) stimuli (Figure 2.3(b)). The points at which the lines in the plot intersect with the x-axis (temporal frequency) are known as *critical flicker frequency*, and the flickering light source starts appearing to be completely steady at those points. Such measurements from the CSF are instrumental in choosing the spatial resolution and refresh rate while designing displays.

Given any real image, CSF can be used to find the corresponding neural image assuming a shift-invariant zero phase shift mapping between the two. Note that we do not have any mechanism to measure the neural image, but only the input threshold stimulus and observer's detection threshold. Since detection is a result of increasing or decreasing neurons' firing rate pooled over the entire neural image, we need to apply a static non-linearity to the neural image that ignores the sign of neural responses and incorporates responses from different neurons. The exact nature of this non-linearity is a contested topic, but popular choices include sum of squares, sum of absolute values, and Minkowski norm. Due to their relative simplicity, CSFs have found many applications in image and video quality metrics [Mantiuk et al., 2021, Andersson et al., 2020], tonemapping [Mantiuk et al., 2008], video compression [Zeng et al., 2002], foveated rendering [Tursun et al., 2019], 3D level-of-detail [Luebke and Hallen, 2001], and other areas of computer graphics.

2.1.3 Interpretation

Once the visual system has built its neural representation, the next step is to use the limited information available to infer/interpret the properties of real-world objects, such as their colour, shape, motion, and depth. The secondary visual cortex and visual association cortex are the parts of the brain believed to be responsible for this inference process. The exact algorithms behind these inferences are still largely unknown and an active area of study. I will briefly describe some leading theories on colour and motion perception as an example.

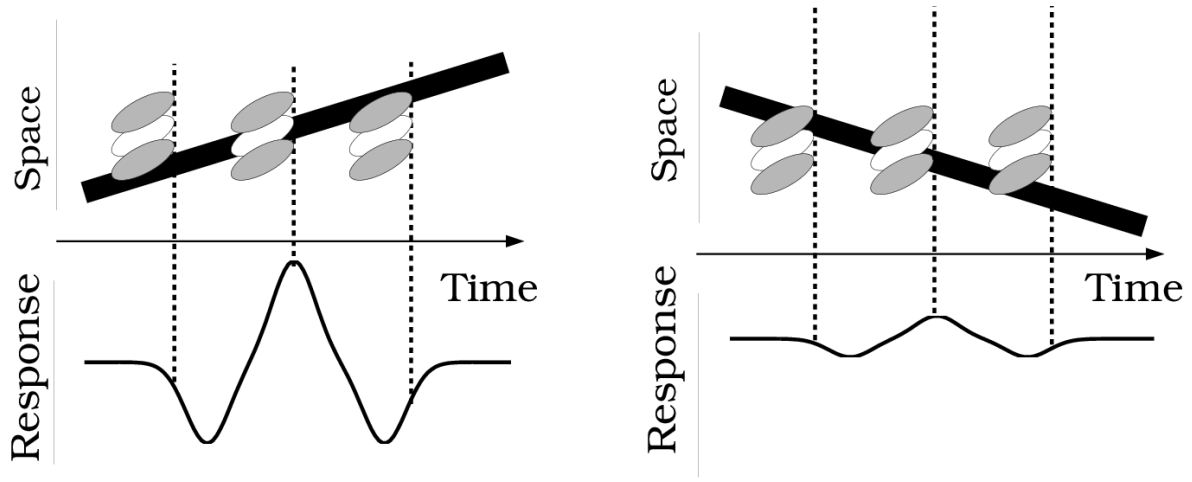


Figure 2.4: Space-time-oriented receptive fields facilitate motion detection by responding asymmetrically to different object velocities and directions. Image source: [Wandell, 1995, Chapter 10]

Colour Object colour is a psychological construct. We see colours by comparing the relative absorption rates of L, M, and S cones in response to light reflected from the object. The reflected light $L(\lambda)$ depends on both the illumination spectrum ($I(\lambda)$) and object's surface reflectance ($R(\lambda)$):

$$L(\lambda) = I(\lambda)\, R(\lambda) \tag{2.1}$$

Hence, colour perception is a process of estimating $R(\lambda)$. However, this problem is underdetermined as infinitely many combinations of $I(\lambda)$ and $R(\lambda)$ can generate the same $L(\lambda)$. Furthermore, the spectrum $L(\lambda)$ is subsequently sampled with only three types of cones, each with a narrow wavelength response. This adds to the problem because different $L(\lambda)$ can result in the same cone responses (also called *metamerism*). Fortunately, not all solutions to this inference problem are equally likely, and hence our visual system can develop priors about this probability distribution to constrain the solution set. For instance, the spectrum of daylight, the most common light source in most of the earth's history, does not change widely and unpredictably with time and location [Judd et al., 1964]. This makes it possible to draw accurate inferences about surface reflectance using low dimensional linear models.

Motion Just like colour, motion perception is also an inference problem. Our visual system needs to estimate the relative velocity of an object w.r.t. its surroundings from the two time-varying retinal images. Neurons in the cortical areas with oriented space-time receptive fields are shown to be particularly useful for this task [McLean et al., 1994]. Figure 2.4 demonstrates this idea where space-time-oriented receptive fields respond differently to the velocity and direction of a moving edge. A centre-surround receptive field is represented using three ellipses where the light area represents the excitatory region, and the dark area represents the inhibitory region. In the first case, the moving line and the receptive field share a common orientation generating a large response. In the second case, they are not well-aligned, leading to a reduced response. An array of such neurons can be used to infer the motion field of a dynamic image, in fact, the same ideas form the founding principles of early optical flow methods [Horn and Schunck, 1981].

The motion in the retinal image is not just generated by object movement; it can also arise from eye movements. Once the interpretation is complete, our brain sends feedback signals to move our eye (or head). The type of eye movements can be broadly divided into the following categories [Purves et al., 2001]:

- *Saccades* are rapid, ballistic eye movements used to change the point of fixation. They can reach velocities of up to 900 $^{\text{deg}}/_{\text{s}}$. Our sensitivity to low-frequency light-dark contrast patterns is strongly reduced during saccades due to neural factors called *saccadic suppression*.

- *Smooth pursuit eye movements* (SPEM) are much slower voluntary eye movements ($0.15\,^{\text{deg}}/_{\text{s}}$ to $80\,^{\text{deg}}/_{\text{s}}$) designed to track moving stimulus and keep it stabilised on the fovea. Inconsistencies between SPEM and the motion of the tracked object can create unstable retinal images and lead to a blurry image. The inconsistencies are higher for unpredictable (non-smooth) motion paths and higher velocities.

- *Vestibulo-ocular* movements compensate for head movements by stabilising our eyes w.r.t. external world.

- *Vergence* movements adjust the rotation of each eye to align their fovea with stimuli located at different depths. Unlike other eye movements, vergence induces motion in the opposite direction in both eyes.

The visual system combines the information about eye movements with the estimated motion field from the retinal images to improve its prediction accuracy.

While the HVS works well in perceiving real motion, we can also perceive motion even when none occurs. One important example of such illusory motion is *apparent motion*: a sensation of continuous motion when presented with a sequence of still images. All modern display systems use apparent motion to recreate a sense of motion while displaying videos and animation. But to create an illusion of smooth motion, we need to ensure that the temporal sampling rate of these displays exceeds the HVS detection capabilities [Watson, 2013] or else the observers may end up seeing objectionable motion artefacts (more in Section 2.2).

Colour and motion perception discussed above are just two examples of many object characteristics our visual system aims to infer from the neural image: shape, size, depth, shadows, occlusions, and textures. All these separate inferences are integrated into a single holistic explanation for the perceived scene. Developing a computational theory for such integration mechanisms continues to be an active challenge in vision science.

2.2 Motion artefacts

Real-time rendering of animated content is prone to various artefacts due to software and hardware limitations. Working with frame rates lower than display refresh rates can give rise to screen *tearing* or *stuttering* artefacts. Asymmetric pixel transition time of LCD panels leads to *trailing* artefacts. Response time compensation used to mitigate trailing may lead to *coronas (glowing edges)* [Jokinen and Nivala, 2017]. *Blur* is perceived whenever a tracked object has a non-zero retinal velocity. This can happen either due to the sample and hold nature of some display technologies (hold-type blur) or imperfect

eye motion when tracking an object [Denes et al., 2020]. One way to reduce blur is to reduce the amount of time a signal is displayed every frame (aka persistence), but this can lead to *flicker* if the display's refresh rate is lower than the critical fusion frequency of the HVS and also from a lower accuracy of saccadic eye motion [Goettker et al., 2020]. Low refresh rates also lead to *judder*/non-continuous motion perception. On low persistence displays, such as HMDs or projectors with butterfly shutters, *ghosting/false-edges* is a common artefact that occurs when the image of a moving object is displayed multiple times at the same location [Scott Murdison et al., 2019]. Finally, the visibility of these artefacts also depends on the displayed content. When the display's spatial and temporal sampling frequency is lower than that of the displayed signal, we see *aliasing* artefacts which are a common occurrence in real-time graphics. Since it is difficult to demonstrate spatio-temporal artefacts with images, I have prepared a video simulation[1].

The artefact visibility can be suppressed by actively modifying image content. For example, increasing motion blur by means of rendering [Sung et al., 2002, Navarro et al., 2011] or longer camera exposure [Fuchs et al., 2010] can reduce aliasing, ghosting and flicker [Stengel et al., 2015, Hoffman et al., 2011]. However, the methods that alter the content are not considered in this work. Artefact suppression can also be achieved by hardware adjustments such as syncing display refresh rate with the frame rate of an application (adaptive-sync), or over-driving liquid crystals to reduce their response time [Feng, 2006]. Unfortunately, achieving an artefact-free motion rendering is often an impossible task due to the current display limitations and usually involves trading off between different artefacts. One way to predict the visibility of these artefacts is through window of visibility analysis [Watson, 2013]. Figure 2.5 provides an example of how displaying a continuously moving checkerboard on a display can give rise to aliases in the frequency domain due to the limited spatial and temporal sampling rate of a display. If the aliases lie inside the spatial and temporal limits of human vision called *window of visibility* (denoted by a diamond), it can lead to visible artefacts such as blur, judder, and flicker. In Chapter 4, I will discuss several other factors that also affect the perception of these artefacts and motion quality.

2.3 Subjective quality assessment

The presence of artefacts in images or videos degrades the quality of the user experience. Since these artefacts are often unavoidable due to hardware or software restrictions, it is helpful to quantify the quality degradation caused by different artefacts. Such data can help us choose between different devices (such as 4K display vs 240 Hz display) or different algorithms (such as JPEG quality level). It can also help us better understand the limitations of such devices/algorithms and facilitate the development of better systems. One way to quantify the effect of these artefacts on perceived quality is to run *subjective quality assessment* experiments where human observers provide their opinion about the perceived quality of a collection of distorted images or videos. Two common protocols for such experiments are direct rating and pairwise comparisons.

Directing rating methods, such as mean opinion score, tasks observers with assigning a score to each condition (e.g. images) on a Likert or a cardinal scale. Though direct rating methods might appear to be a good measurement of perceptual attributes, they suffer

[1] `https://www.cl.cam.ac.uk/research/rainbow/projects/alsarr/motion_quality/motion_artifacts.mp4`

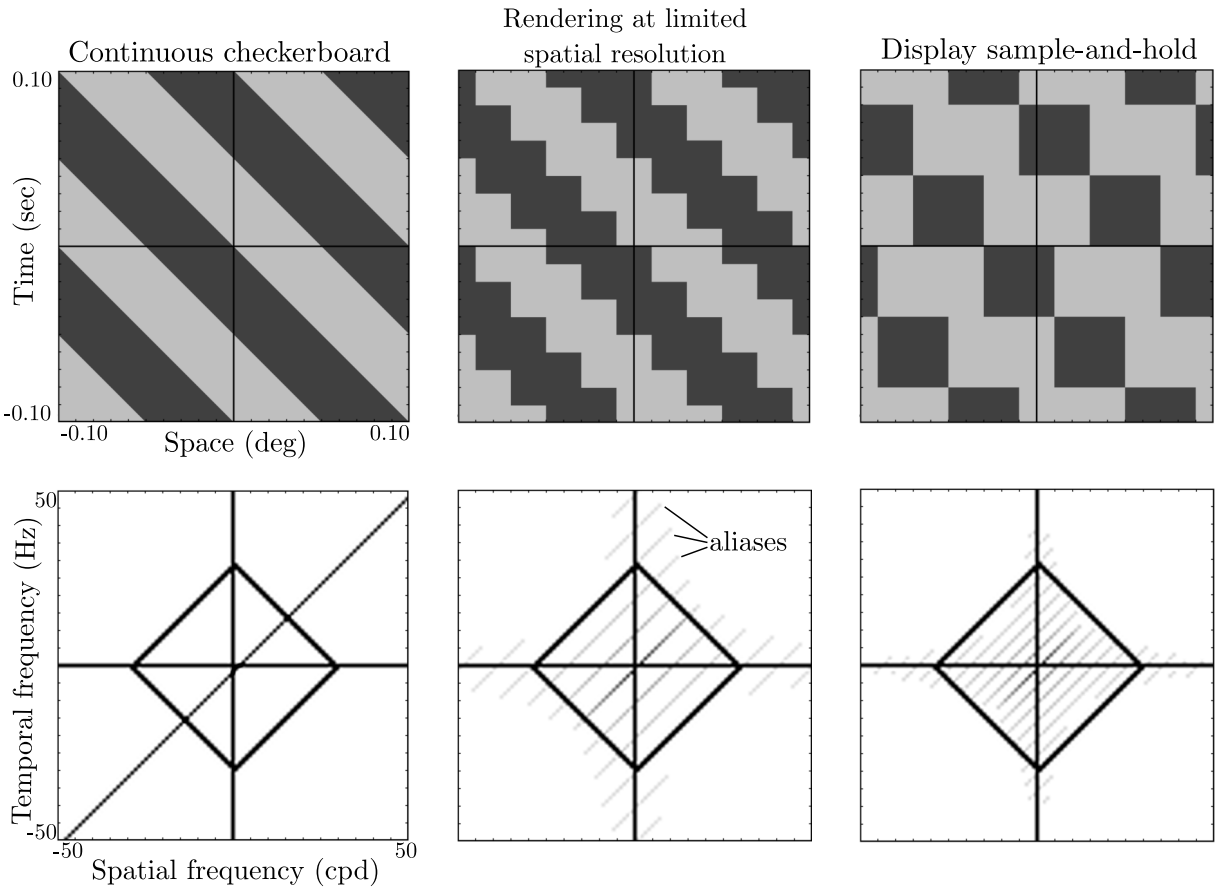


Figure 2.5: Analysing a moving 1D checkerboard signal rendered on a 25 Hz, 50 ppd display. Top row is in spatio-temporal domain. Bottom row is corresponding frequency spectrum. Diamond shape represents the window of visibility. Both rendering with a limited spatial resolution buffer and displaying the image on display with a finite refresh rate leads to creation of aliases in the frequency domain (perceived as judder and flicker) and loss of high-frequency information (perceived as blur). The above figures do not consider SPEM.

from a major limitation. Even after carefully training the participants, the scale used by each participant can substantially vary between the participants or sometimes even within the same participant if participating on different days.

Pairwise comparison methods instead ask the observers to rank a set of conditions. This is a much simpler task that non-experts can perform with little training. These methods have been shown to have less variance in the experimental results and also a shorter experiment time than direct rating [Mantiuk et al., 2012]. However, mapping the results from such experiments to a practical scale could be quite complex. Figure 2.6 provides an example of a typical image quality assessment experiment using a pairwise comparison protocol. Participants are shown a pair of images at a time from a large set of distorted images and asked to pick the image with better quality (i.e. the image closer to its respective distortion-free reference image). The participant's answer is then stored in a *comparison matrix* C, where each entry C_{ij} denotes the number of times the participant picked image i over image j. Thus the probability of image i being picked over image j is

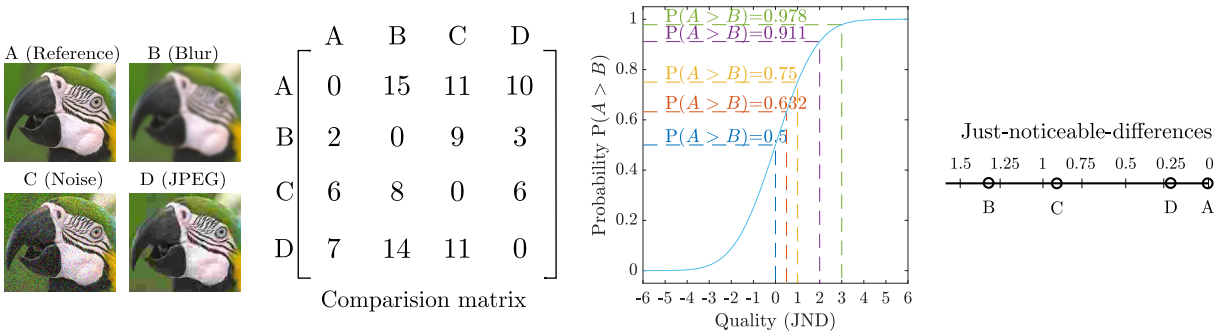


Figure 2.6: Subjective scaling of three image artefacts onto a JND scale

given by:

$$p_{ij} = \frac{C_{ij}}{C_{ij} + C_{ji}} \tag{2.2}$$

Note that due to the subjective nature of the experiment, this data tends to be noisy. To account for this, the protocol uses the Thurstone Case V observer model. This model assumes that participants made their selections by assigning a single quality value to each image, and this quality is a random variable following a normal distribution with same inter- and intra-observer variance. The probability of the population picking image A over image B $P(A > B)$ can then be mapped to a corresponding quality difference $(q_A - q_B)$ using an inverse cumulative normal distribution. The variance of this distribution is set such that the $P(A > B) = 0.75$ is equal to $q_A - q_B = 1\,[\text{JND}]$. JND units are more meaningful than direct ratings because they describe the practical significance of the difference in quality. Finally, the quality of each image is then estimated using maximum likelihood estimation, which picks the quality scores that maximise the probability of observing our data (C). More details on pairwise comparison experiments can be found in [Perez-Ortiz and Mantiuk, 2017].

Objective quality metrics are computational models that aim to predict the quality difference between two images without any user input. The results from subjective experiments provide a means to test such objective metrics. In Chapter 4, I conduct a pairwise comparison experiment to quantify the effect of motion artefacts on quality and in Chapter 5, I present an objective metric of motion quality trained and tested on the results of this experiment.

2.4 Adaptive-sync displays and Variable rate shading

While the objective quality metrics allow us to develop new rendering algorithms that can trade-off between different artefacts to deliver the best quality, the hardware these algorithms run on must have a flexible mechanism to support the adaptive nature of such algorithms. In this section, I will discuss two such technologies which I will use to test the rendering algorithms developed in this dissertation.

Adaptive-sync displays A typical graphics-display pipeline consists of two raster buffers: a front buffer that is shown on the display, and a back buffer that the GPU draws into. When the GPU finishes drawing, the two buffers are swapped. Pixels are copied row-by-row (or column-or-column) from the front buffer to the display. If the swap were

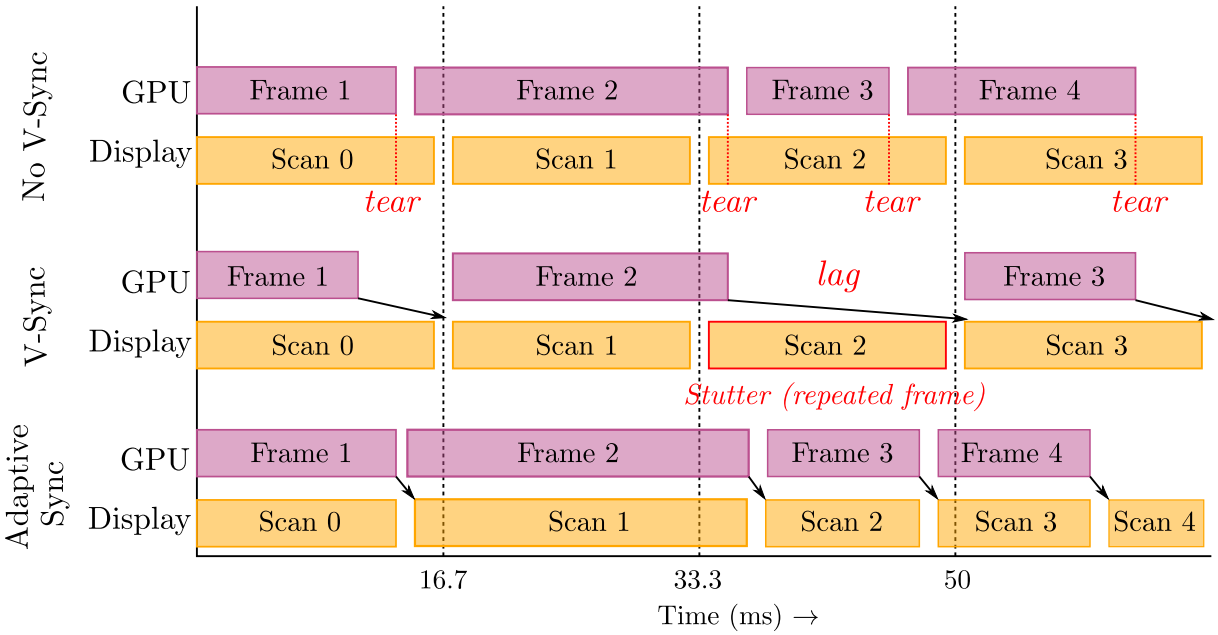


Figure 2.7: Display sync strategies. No synchronisation between rendering and display scan can lead to tearing artefacts. Synchronisation can introduce stuttering artefacts. Adaptive sync technologies eliminate both of these artefacts.

to happen before the display scan is complete, it would result in an image on the display which is made up of more than one frame. To avoid such *tearing* artefacts, it is possible to synchronise buffer swaps with display scan (called *V-Sync*); however, this may pause the GPU till the scan is complete resulting in decreased performance and input lag. Also, if the GPU frame rate is even slightly less than the display's refresh rate, V-Sync will end up repeating frames leading to stuttering or judder artefacts. Adaptive-sync displays such as G-Sync (NVIDIA) or Free-Sync (AMD) eliminate these artefacts by updating their refresh rate to match the rate of frame generation. They can achieve an arbitrary refresh rate (within display capabilities) and thus can help in saving power for static scenes and reducing lag. An illustration of the three techniques can be found in Figure 2.7.

Variable rate shading VRS [Nvidia, 2018] or coarse pixel shading [Vaidyanathan et al., 2014] are recent technologies available on the latest GPUs. VRS decouples shading and visibility calculations to provide flexible control over shading. By rasterizing at full resolution but shading at a lower resolution, the technique reduces pixel shading load while preserving edges and visibility of objects. Unlike their predecessors, Multi-Resolution Shading and Lens-Matched Shading [Kraemer, 2018], which optimise shading workload to match VR optics, VRS allows for much more granular control, where every 16×16 tile of pixels can have a different shading rate. The current specification offers seven different shading rates $(1 \times 1, 1 \times 2, ..., 4 \times 4)$, including non-uniform shading resolution along horizontal and vertical dimensions. Since VRS only executes the fragment shader a few times per multiple raster pixels and broadcasts the same shaded value to the neighbouring pixels, the artefacts generated by VRS on the surface of each object are the same as upsampling with a box filter. In Chapter 4, I will discuss how the visibility of these artefacts depends on several factors such as luminance, refresh rate, velocity, and content, by using VRS to control the resolution of the stimuli.

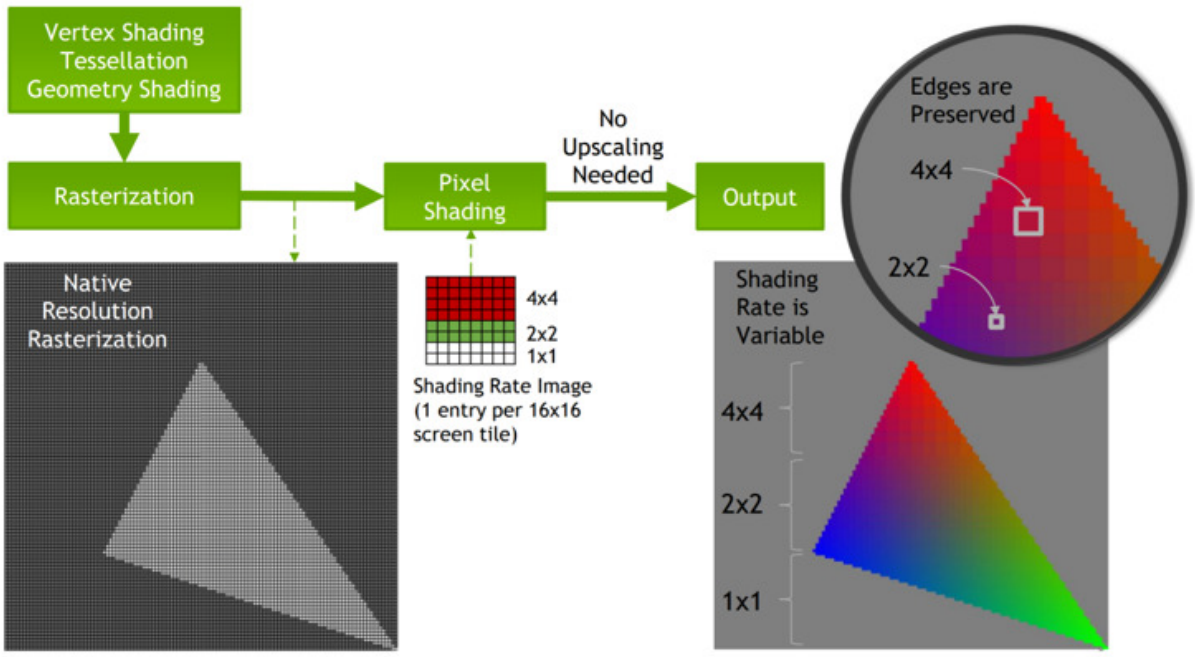


Figure 2.8: Variable rate shading allows visibility calculation at a full resolution and shading calculation at a lower resolution. Image from [Yang et al., 2019].

2.5 Perceptually-optimised rendering

In the above sections, we discussed how modern graphics and display pipelines suffer from certain limitations which can introduce unwanted artefacts in the images. We also discussed how an image presented by the display first goes through certain optical and neural transformations before it is finally perceived and how we can build computational models of vision trained through subjective experiments to predict the visibility of artefacts or quality of images. The field of research that utilises the estimates of the final neural image and adapts the rendering algorithms (or displays) to save computational bandwidth or improve quality is known as *perceptually-optimised rendering*. Though sparsely discussed in traditional graphics research, the idea of using our perceptual limitations to improve rendering is not new. Some of these concepts are so integrated into our daily usage of technology that we barely even think about it. Examples include displays using metamerism to reproduce real-world colours, cinemas presenting animations at a 24 Hz refresh rate to induce apparent motion, and lossy image and video compression assigning low bit-rates to higher spatial frequencies.

Perceptual optimisation has also found success in computer graphics scene simplification. To reduce flickering artefacts that pop up during level-of-detail (LOD) transition, transitions can be restricted to peripheral vision [Luebke et al., 2003] or to when the object is in motion [Clark, 1976]. In both scenarios, the acuity of the visual system is reduced, and observers are less likely to notice the popping artefacts. It is also possible to design LOD algorithms that analyse mesh and textures using higher-order models of vision to identify the most salient mesh features and preserve them during simplification [Lee et al., 2005, Yang et al., 2016]. While these methods attempt to maintain perceptual similarity between full and reduced resolution meshes, a higher saving can be achieved if we instead attempt to maintain the same impression of the scene appearance [Ramanarayanan et al., 2007].

Similar models of perception have also been employed to accelerate rendering. Myszkowski [1998] used perceptually-based visible difference predictor (VDP) to decide the stopping conditions of progressive Monte-carlo rendering. Walter et al. [2002] devised a method to pre-compute the masking ability of textures using aspects of JPEG image compression and used it to adaptively sample light paths, achieving a speed-up of up to 15 times compared to a traditional non-adaptive path tracer. Farrugia and Péroche [2004] gave another perceptually-based progressive rendering solution that accounted for eye adaptation. Saliency models have also served as a good metric for adaptive sampling. Longhurst et al. [2006] came up with a rasterization-ray tracing hybrid where they calculated saliency maps from the output of rasterization and used it to guide the distribution of ray-traced samples. Cater et al. [2003] utilised HVS's property of *inattentional blindness* by combining predetermined task maps with spatio-temporal CSF and used them to guide a hybrid image-based and ray traced progressive renderer. Myszkowski [2002] proposed a temporal extension of the VDP called Animation Quality Metric Algorithm (AQM) to decrease the quality of moving objects in dynamic scenes. Lo et al. [2010] exploited the human binocular fusing ability to significantly reduce the resolution of one of the image pairs while retaining an overall high perceptual quality in a stereo ray-tracer.

It is possible to achieve further gains in performance if we also account for the properties of the display in addition to HVS. For instance, optics in VR headsets "squeezes" the displayed image, and some of the peripheral pixels never make it to the eye. Lens-Matched Shading [Kraemer, 2018] aims to avoid rendering these excess pixels and can achieve up to 50% reduction in shading throughput. VR headsets also suffer from low spatial resolution but have high refresh rates. [Didyk et al., 2010, Lee and Didyk, 2018] provide a method that exploits the retinal integration latency and high refresh rate of the display to improve the apparent display resolution for moving images. With recent adaptive-sync displays, it is also possible to control the trade-off between spatial and temporal resolution [Debattista et al., 2018]. If the displays are fitted with measurement devices such as eye trackers, it is possible to further optimise the rendering by matching the reduced sensitivity in peripheral vision [Tursun et al., 2019, Patney et al., 2016].

Similar to the above approaches, the rendering algorithm I propose in Chapter 6 optimises quality of rendering by accounting for several display, GPU, and perceptual factors.

Chapter 3

Motion adaptive refresh rate and resolution

3.1 Introduction

Growing spatio-temporal resolution of displays, time-shared nature of cloud gaming, and power-constrained mobile devices often give rise to scenarios where a GPU has to work under a limited performance or transmission budget. The current solution is to fix either the refresh rate or the spatial resolution and dynamically vary the other variable [Unreal, 2021, Unity, 2021]. However, this is a sub-optimal strategy for rendering scenes with varying velocities. With the introduction of adaptive-sync displays that dynamically control the display's refresh rate, we can choose the best trade-off between spatial resolution (sharpness of image) and temporal resolution (smoothness of motion) for a given content. Determining this trade-off requires modelling the perceived quality of the image under motion.

To determine this optimal trade-off, Debattista et al. [2018] conducted psychophysical experiments that studied the perceived quality for different resolution and refresh rate ratios across various computational budgets. Then, they fit a cubic polynomial to their data to predict this optimal trade-off for arbitrary budgets. However, their model does not account for any scene parameters such as image content or object velocity and reduces to the traditional constant refresh rate and resolution rendering once the computational budget is fixed. Denes et al. [2020] (my collaborators) extended this idea to also account for scene motion by measuring the perceived quality of motion for a wide range of refresh rates and velocities of motion through extensive subjective experiments. They proposed a white-box visual model that predicts this quality by accounting for two major motion artefacts: judder and blur. This model enabled us to develop a motion-adaptive refresh rate and resolution (MARRR) rendering method that can optimally choose every frame's refresh rate and resolution in real-time.

In this chapter, I will compare MARRR against [Debattista et al., 2018] and traditional constant resolution and refresh rate techniques under complex motion scenarios through a subjective study. This will help us validate the utility of such motion-adaptive techniques in real-time graphics and identify their limitations. I will start by briefly introducing the motion quality model developed by Gyuri Denes in Section 3.2 because Chapter 5 of this dissertation builds upon his work. Following that, the chapter will focus on my contribution: the application of this model to real-time graphics (Section 3.3) and its

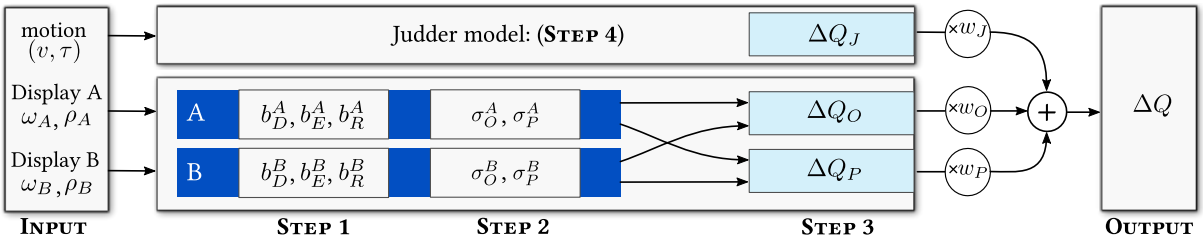


Figure 3.1: Schematic diagram of the visual model of blur and judder.

psychophysical validation (Section 3.4).

This chapter is based on the work published in the SIGGRAPH 2020 article: *A perceptual model of motion quality for adaptive refresh rate and resolution.*

3.2 Background: Visual model of blur and judder

Section 2.2 discussed how modern displays are prone to various artefacts when displaying motion. When rendering on traditional LCDs, two of the most common artefacts are blur and judder. In this section, I present the visual model proposed by Gyuri Denes, which predicts the quality degradation caused by these two artefacts by analysing the display's resolution, refresh rate, tracked object velocity and movement type. Later in Section 3.3 and Section 3.4, I will show how it can be integrated with game engines to adaptively control refresh rate and resolution and how such rendering outperforms traditional constant refresh rate-resolution rendering methods.

Formally, the quality difference between two displays A and B using different refresh rates and resolutions can be defined as:

$$\Delta Q(\ldots) = \Delta Q(\omega_\mathrm{A}, \rho_\mathrm{A}, \omega_\mathrm{B}, \rho_\mathrm{B}, v, \tau), \tag{3.1}$$

where ω_A and ω_B are the refresh rate in Hz of the display A and B, respectively, ρ_A and ρ_B are their respective resolutions in pixels-per-degree (ppd), v is the velocity of the tracked object and τ is the predictability of motion by smooth pursuit eye motion (SPEM) (binary: *predictable* or *unpredictable*) (Section 2.1.3). The quality difference (ΔQ) is defined in *Just-noticeable-difference* (JND) units (Section 2.3).

The quality (ΔQ) can be approximated as a weighted sum of three components:

$$\begin{aligned} \Delta Q(\ldots) = & w_\mathrm{P}\, \Delta Q_\mathrm{P}(\omega_\mathrm{A}, \rho_\mathrm{A}, \omega_\mathrm{B}, \rho_\mathrm{B}, v, \tau) + \\ & w_\mathrm{O}\, \Delta Q_\mathrm{O}(\rho_\mathrm{A}, \rho_\mathrm{B}) + \\ & w_\mathrm{J}\, \Delta Q_\mathrm{J}(\omega_\mathrm{A}, \omega_\mathrm{B}, v, \tau), \end{aligned} \tag{3.2}$$

quality degradation caused by blur parallel to motion direction (Q_P), by blur orthogonal to motion caused by reduction of display resolution (Q_O) and by judder/non-smooth motion (Q_J). The pipeline of the model can be found in Figure 3.1. Next, I will explain how each of these components can be estimated.

Hold-type and eye-motion blur When our eyes undergo SPEM to track an object, it moves in a continuous trajectory. However, an LCD presents motion in a discrete manner, i.e. a sequence of images, each shown for a finite duration of time. This leads to smearing

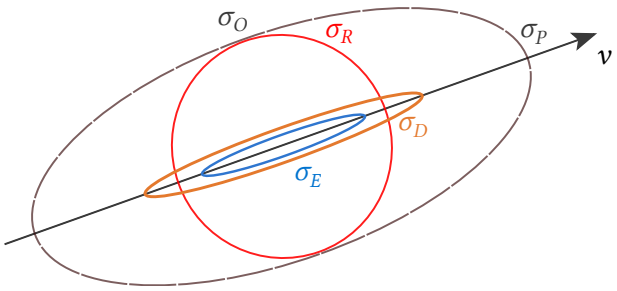


Figure 3.2: Motion parallel and orthogonal blur

of the image on the retina and is perceived as blur. The amount of blur can be modelled as a box filter of width:

$$b_{\mathrm{D}} = \frac{v}{\omega} \quad [\mathrm{deg}]. \tag{3.3}$$

Furthermore, the tracking of our eyes is imperfect, which leads to additional blurring. The error in tracking depends on the predictability of motion. The eye-motion blur can also be modelled with a box filter of width

$$b_{\mathrm{E}} = p_{\mathrm{a}}\, v + p_{\mathrm{b}} \quad [\mathrm{deg}], \tag{3.4}$$

where p_{a} and p_{b} are constant coefficients fitted on psychophysical data and their values are different for predictable and unpredictable motion.

Spatial resolution blur Assuming the use of a bilinear filter to upsample images in real-time graphics, the blur due to reduced spatial resolution can be modelled with a triangle filter of average width :

$$b_{\mathrm{R}} = \frac{1}{\rho} \quad [\mathrm{deg}]. \tag{3.5}$$

Motion parallel and orthogonal blur The combination of the three types of blurs can be simplified by approximating them with Gaussian filters of the following standard deviations:

$$\sigma_{\mathrm{D}} = \frac{v}{\pi f}, \quad \sigma_{\mathrm{E}} = \frac{p_{\mathrm{a}} v + p_{\mathrm{b}}}{\pi}, \quad \text{and} \quad \sigma_{\mathrm{R}} = \frac{\sqrt{2} b_{\mathrm{R}}}{\pi}. \tag{3.6}$$

Since eye-motion blur and hold-type blur only blur the image in the direction of motion, while lowering spatial resolution blurs the image in all directions (Figure 3.2), they need to be combined into motion parallel blur (σ_{P}) and orthogonal blur(σ_{O})

$$\sigma_P = \sqrt{\sigma_{\mathrm{E}}^2 + \sigma_{\mathrm{D}}^2 + \sigma_{\mathrm{R}}^2} \quad \text{and} \quad \sigma_{\mathrm{O}} = \sigma_{\mathrm{R}}\,. \tag{3.7}$$

Energy to quality The reduction in perceived quality due to blur can be calculated in JND units by calculating the loss of energy caused by blur. Let's assume the worst-case scenario of a moving infinitely thin line (unit impulse), which contains uniform energy across all spatial frequencies. When this line is convolved with Gaussian blur kernel of standard deviation σ, it results in a Gaussian signal given by $\exp\left(-2\pi^2\phi_i^2\sigma^2\right)$ in the Fourier domain. This blur signal can then be transformed into the final perceived signal by modulating the Fourier coefficients with the spatial contrast sensitivity function (CSF) (Section 2.1.2). The overall energy of the blur signal (called blur energy) is calculated by sampling a range of frequencies as

$$E_{\mathrm{b}}(\sigma) = \sum_i \left(\frac{\mathrm{CSF}(\phi_i) \exp\left(-2\pi^2\phi_i^2\sigma^2\right)}{\tilde{m}_{t,b}} \right)^{\beta_b}, \tag{3.8}$$

where CSF is Barten's CSF model with the recommended standard observer parameters and the background luminance of $100\,\mathrm{cd/m^2}$ [Barten, 2003], $\phi_i = \{1, 2, \ldots, 64\}$ [cpd] are a range of spatial frequencies to be sampled, $\tilde{m}_{t,b}$ is the threshold parameter, and β_b is the power parameter of the model. They both are obtained by fitting the model to psychophysical data.

The difference in blur energies between display A and B can then be interpreted as quality differences

$$\begin{aligned} \Delta Q_{\mathrm{P}} &= E_{\mathrm{P}}(\sigma_{\mathrm{P}}^{\mathrm{A}}) - E_{\mathrm{b}}(\sigma_{\mathrm{P}}^{\mathrm{B}}), \\ \Delta Q_{\mathrm{O}} &= E_{\mathrm{P}}(\sigma_{\mathrm{O}}^{\mathrm{A}}) - E_{\mathrm{b}}(\sigma_{\mathrm{O}}^{\mathrm{B}}), \end{aligned} \tag{3.9}$$

Judder When continuous motion is reproduced on low refresh rate displays, it appears non-smooth or juddery. The visibility of the judder can be estimated by examining the aliasing copies of the signal in the frequency domain. As shown in Figure 3.3, for a signal of velocity v, the first aliasing copy is located at temporal frequency equal to refresh rate (ω) and spatial frequency

$$\phi = \frac{\omega}{v}. \tag{3.10}$$

Similar to blur, the energy of the alias can be calculated as

$$E_{\mathrm{j}}(\omega, v) = \left(\frac{\mathrm{stCSF}(\phi, \omega)}{\tilde{m}_{t,j}} \right)^{\beta_J}, \tag{3.11}$$

where stCSF is truncated low-pass Kelly's spatio-temporal CSF [Kelly, 1979], β_J is the power parameter for judder, and $\tilde{m}_{t,j}$ is the threshold for judder. Finally, the quality difference due to judder can be expressed as

$$\Delta Q_{\mathrm{J}} = E_{\mathrm{j}}(\omega_{\mathrm{A}}, v) - E_{\mathrm{j}}(\omega_{\mathrm{B}}, v). \tag{3.12}$$

Model calibration The model parameters were fitted on data captured through psychophysical experiments that measured the combined effect of blur and judder on quality as well as the effect of both artefacts individually. The value of each free parameter can be found in Table 3.1.

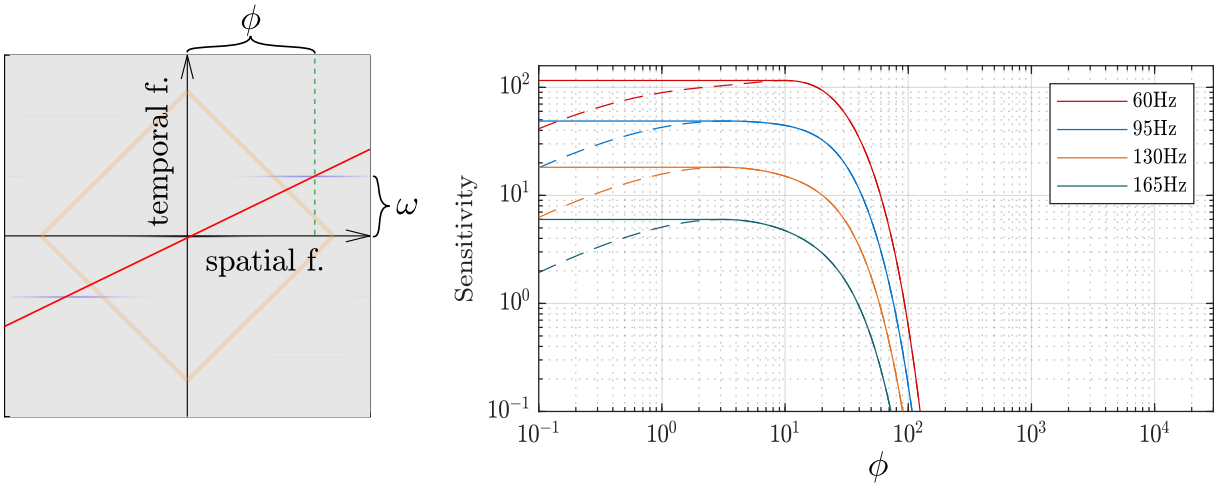


Figure 3.3: Spatia-temporal sampling of a continuous signal leads to creation of aliasing copies in frequency domain (left). Our sensitivity to these replicas can be captured by truncated low-pass Kelly's spatio-temporal CSF (right).

w_J	w_P	w_O	$\tilde{m}_{t,b}$	β_B	β_J
2.218677	1.472819	1.472819	383.5854	1.83564	2.5747

	p_a	p_b	$\tilde{m}_{t,j}$
Predictable	0.001648	0.079818	218.712
Unpredictable	0.004978	0.176820	165.779

Table 3.1: Fitted parameters of MARRR

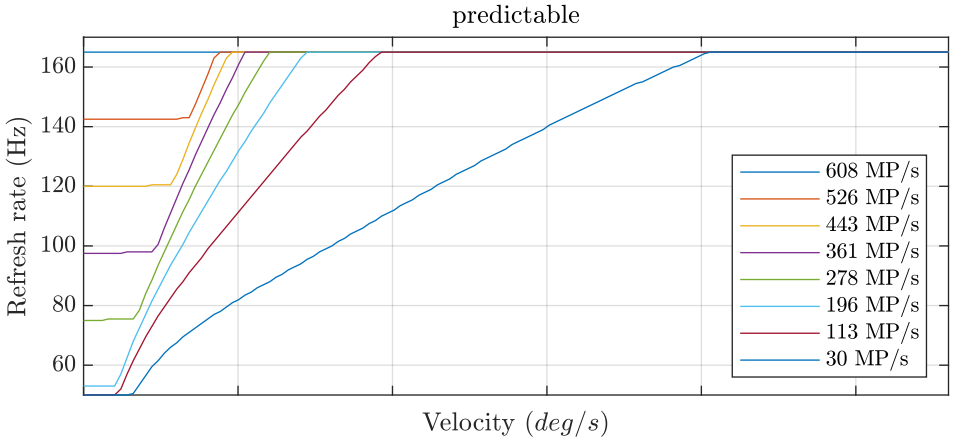


Figure 3.4: Precomputed LUT of optimal refresh rate and resolution for different GPU bandwidths and velocities.

3.3 Rendering with adaptive refresh rate and resolution

Bottlenecks in the graphics pipeline, such as limited transmission bandwidth or power constraints, often require an application to compromise spatial resolution or refresh rate. The traditional solution to this problem of constrained rendering is to either fix the resolution and vary the refresh rate in accordance with the allowed budget or vice versa. However, in these solutions, the resolution-refresh rate trade-off is determined once during initialisation for a given budget and stays the same throughout the application regardless of scene content. With the introduction of adaptive-sync displays such as G-Sync and Free-Sync technologies, which are capable of driving a display at an arbitrary refresh rate without introducing tearing artefacts, the constant refresh rate and resolution rendering become a sub-optimal strategy. We can use the visual model introduced in the previous section to find the refresh rate and resolution configuration that maximises the perceived quality for a fixed rendering budget. This can be expressed as the following optimisation problem:

$$\arg\max_{\omega,\rho} \Delta Q(\omega, \rho, \omega_{\mathrm{R}}, \rho_{\mathrm{R}}, v, \tau) \;\; \text{s.t.} \;\; \omega\,\rho\,\psi_{\mathrm{x}}\,\psi_{\mathrm{y}} \leq \mathrm{B} \wedge \omega \leq \omega_{\mathrm{R}} \wedge \rho \leq \rho_{\mathrm{R}}, \tag{3.13}$$

where B is the rendering budget in pixels-per-second and ψ_{x} and ψ_{y} are the display's horizontal and vertical viewing angles, respectively. All quality calculations are done relative to quality at the maximum refresh rate (ω_{R}) and maximum resolution (ρ_{R}) of the display. The rest of the parameters are the same as Eq. (3.1). Since the quality depends on object's current velocity, the optimal refresh rate and resolution can vary for every frame.

This optimisation problem can be solved with an exhaustive search as the allowed refresh rates are a limited number of integer values. To minimise the overhead of solving this optimisation in real-time rendering, we can precompute the optimal configuration for a range of eye velocities and bandwidths and store them as a LUT (Figure 3.4).

This method can now be easily integrated with any game engine. Before rendering a frame, we can estimate the user's gaze velocity using an eye-tracker and scene motion.

This velocity, along with our allowed bandwidth, can be used to sample the LUT and calculate the refresh rate and resolution for the current frame. The desired resolution is achieved by resizing the viewport using bilinear interpolation. The refresh rate of the display, unfortunately, cannot be set directly because adaptive-sync displays work by predicting desired refresh rate. The adaptive-sync chip on the display analyses the frequency of incoming data packets to guess the application's frame rate and then matches the display refresh rate to it. Hence to set the display refresh rate to the desired value, we need to precisely control the application's frame rate. This can be done by using coroutines that wait for the rendering to finish and then throttle the main thread to free the CPU resources until the desired frame duration is achieved. The entire process is summarised in Figure 3.5.

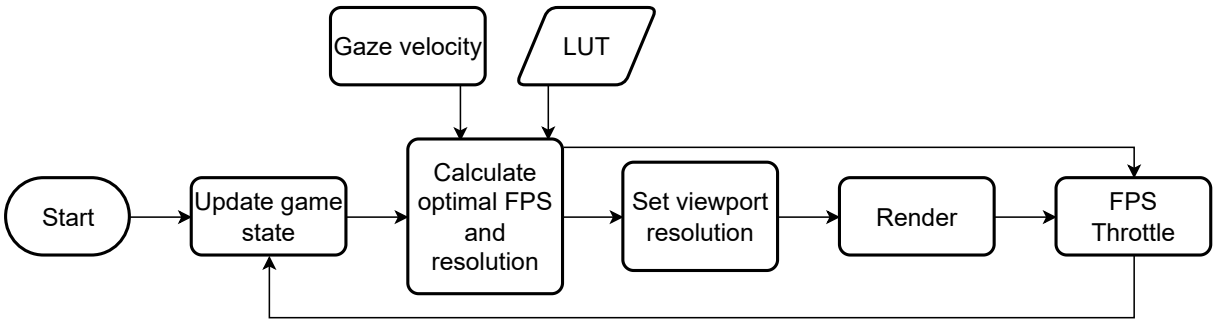


Figure 3.5: MARRR pipeline in Unity3D game engine.

3.4 Psychophysical validation

To validate the applicability of our model in complex motion scenarios, I built a sandbox video game running the MARRR algorithm and ran a preference experiment to compare with fixed refresh rate and resolution rendering, including [Debattista et al., 2018]'s model for selecting refresh rate.

Setup This experiment used a pairwise comparison design where the conditions were played on a single G-sync capable ASUS display (PG279Q). Users had the freedom to toggle between the two conditions but did not know which condition was which. I built the video game in Unity3D, which ran on an ASUS monitor with an Eyelink II eye-tracker sampling the gaze location at 500 Hz (pupil-only mode). In each trial, the two conditions showed the same 3D scene and the same fixed bandwidth (1/3-rd of maximum available bandwidth: 1/3 × maximum resolution (2560×1440 pixels) × maximum refresh rate (165 Hz) = 203 MP/s) but one with MARRR and the other with a constant resolution and refresh rate. MARRR assumed predictable motion and controlled the optimal refresh rate (from 50 Hz to 165 Hz) frame-by-frame by querying a pre-computed lookup table. Refresh rates chosen for constant resolution and refresh rate were 30 Hz, 60 Hz and 120 Hz to mimic conventional gaming hardware and 75 Hz and 90 Hz as predicted by Debattista et al. [2018] experimental data and their model, respectively. Two different values were picked from [Debattista et al., 2018] because their model predictions did not match with their experimental data for the chosen bandwidth.

Figure 3.6: Viking village stimulus used in the experiment with FPS camera (top) and RTS camera (bottom). © Unity Technologies

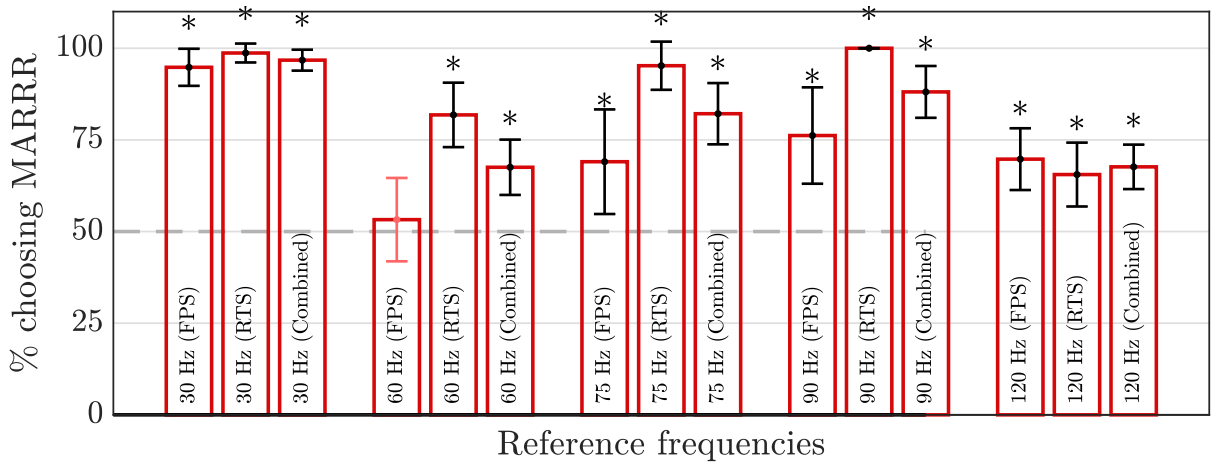


Figure 3.7: Results of validation experiment in a video game setting showing the percentage of participants picking the proposed adaptive MARRR algorithm over standard *constant*-resolution-and-*constant*-refresh-rate rendering. Error bars denote 95% confidence interval. * denotes statistical significance (p-value < 0.05 with the null hypothesis of random choice). MARRR consistently performed better across all conditions except at FPS-60Hz, where the results are not statistically significant. I believe the latter could be due to sample size and participants' habituation to FPS gaming at 60Hz.

Stimuli For video game content, I used the Viking Village scene from Unity. I added two camera modes to this 3D environment: (1) FPS mode, where players could explore the scene and interact with objects, such as collecting coins and shooting enemies, similar to a typical first-person shooter-style video game (Figure 3.6-top). (2) RTS mode, where the players could explore the scene from a bird's eye view, similar to a simulator and real-time strategy style video games (Figure 3.6-bottom). The trials were randomised with an equal number of FPS and RTS conditions.

Estimating velocity To estimate target object velocity for our model in real-time, I used the gaze velocity averaged over 17 samples as reported by the EyeLink SDK. When there was no camera movement, this velocity was prone to be noisy (likely due to ocular drifts or microsaccades [Alekseevsky, 2022]). Hence, when there was no mouse or keyboard interaction, I instead reverted to screen-space object velocities at the gaze location as reported by the game engine.

Participants and procedure 17 people (aged 20-40) volunteered to participate in the experiment. All had a normal or corrected-to-normal vision. In each trial, participants were asked to explore the 3D scene in both rendering conditions and choose the one with overall better visual quality. The order of the comparisons was randomised. Each observer completed 40 comparisons. All participants reported only casual gaming experience (playing only a few times a month) with little-to-no exposure to high-refresh-rate monitors.

Results The results of this experiment indicate an overall preference for MARRR compared to the fixed refresh rate and resolution rendering (Figure 3.7). At the chosen bandwidth, the difference in preference is particularly large for 30 Hz. The difference is significant but less pronounced for comparisons with 60 and 120 Hz. I believe this could

have been caused by scene's complexity with many factors such as aliasing, participants' habituation to certain refresh rates, and eye-tracking noise affecting the results. Also, there is a higher preference for MARRR in the RTS setting than FPS in all configurations except 120 Hz. The results show a clear preference over [Debattista et al., 2018] predictions (75 and 90 Hz), suggesting the importance of accounting for content velocity during rendering.

3.5 Limitations

The results of the above methods demonstrate the potential of adaptive refresh rate rendering algorithms. However, this method has a few limitations. The underlying visual model is too conservative as it disregards scene content and thus cannot be combined with content-aware methods [Pellacini, 2005]. The model, though designed to generalise across displays, was only calibrated on high-persistence LCDs of standard brightness. This limits its applicability to VR headsets and HDR displays. Being a content-independent model, it does not account for spatio-temporal aliasing artefacts arising from dynamic resolution. The lack of content awareness also excludes the use of recent adaptive shading frameworks such as VRS. Another limitation of the proposed rendering method is that it ideally requires fine control of the display's refresh rate, which is not possible with current Adaptive-Sync displays as they rely on a prediction model as explained in Section 2.4. A direct interface to communicate the desired refresh rate to display could reduce the latency in our method.

3.6 Summary

With displays demand for transmission bandwidth surpassing GPU capabilities, it is necessary to adaptively control the quality of rendering. In this chapter, I introduced a visual model proposed by my colleague that predicts the perceived quality by analysing judder and blur motion artefacts. I showed how this model could be integrated with 3D game engines to select every frame's optimal refresh rate and resolution adaptively. I then compared the proposed motion-adaptive rendering method to an empirical model and the traditional constant resolution-refresh rate method in a subjective study. The results showed a significant preference for the proposed method in all tested scenarios making a strong case for including quality of motion considerations in rendering methods. Finally, I identified several limitations of the proposed work that need to be solved to fully realise the potential of adaptive rendering pipelines.

Chapter 4

Motion quality dataset

4.1 Introduction

In Chapter 3, we discussed an application of a motion quality model to real-time rendering, which significantly improved visual quality. Though this model indicates the promise of motion-adaptive rendering, it has certain limitations. It assumes no content information and does not account for aliasing artefacts commonly found in real-time rendering. To develop a new visual model that overcomes these limitations, it is necessary to understand what and how display and content parameters affect the perception of these artefacts. This knowledge can be built through psychophysical experiments, and the resulting datasets can be used to develop, calibrate and validate visual models.

In this chapter, I first survey the psychophysical studies on the perception of motion artefacts conducted in the last two decades (Section 4.2). Through this survey, I identify the major factors that affect motion quality and areas that require more study. I then conduct a subjective quality experiment to fill these gaps in research. The experiment measures the perceived loss of quality due to shading rate reduction under a large range of display refresh rates, resolutions, display persistence, luminance, contrast, and content velocity (Section 4.3). A subset of these identified datasets and the new dataset will be used to calibrate a new visual model in Chapter 5.

This chapter is based on the work published in the SIGGRAPH Asia 2021 article: *Perceptual Model for Adaptive Local Shading and Refresh Rate*.

4.2 A survey of factors affecting perception of motion artefacts

Section 2.2 discussed a range of artefacts that arise when we display a motion signal due to hardware and software limitations. They include judder, blur, aliasing, flicker, ghosting and more. The degree of visibility of these artefacts and their subsequent effect on perceived visual quality depends on multiple display and content-related factors and has been extensively studied by various works. I will now summarise some of the most well-studied factors.

Refresh rate and velocity: There is a consensus among multiple studies examining a wide range of refresh rates and velocities (refer to Table 4.1) that higher refresh rates are

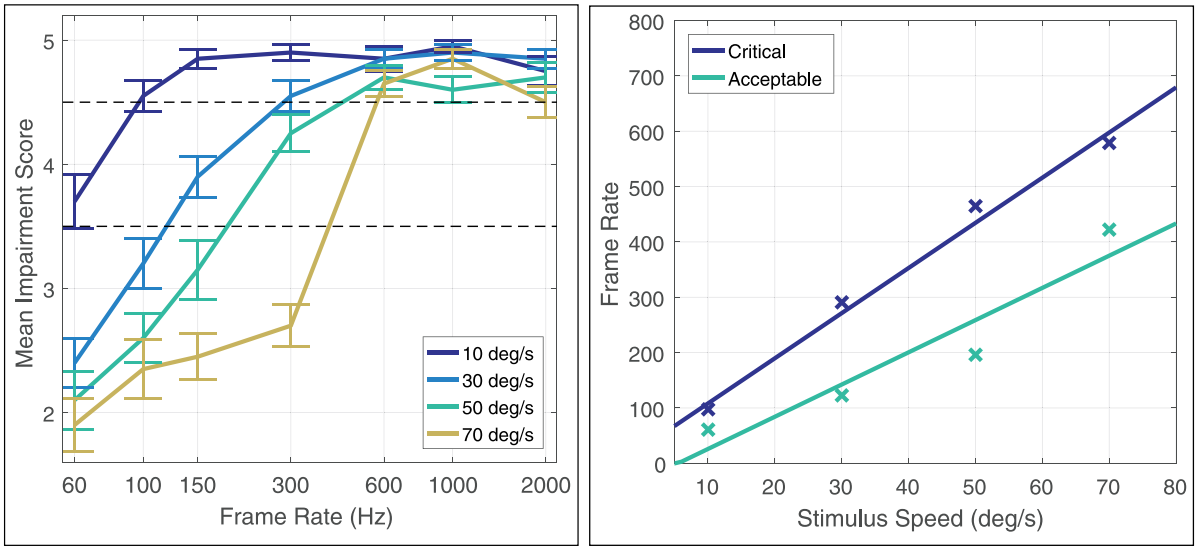


Figure 4.1: Results from Mackin et al. [2016] experiment demonstrating that increase in quality from increasing refresh rate plateaus after a point which depends on object velocity (left) and refresh rate required to achieve artefact free results (critical frame rate) grows linearly with stimulus velocity (right).

required to maintain the quality at higher velocities of motion (Figure 4.1). Mackin et al. [2016] report that even for a simple stimulus, it can take up to 600 Hz (depending upon the velocity) to achieve an artefact-free motion. However, observers found the motion artefacts beyond 300 Hz to be tolerable. They also noticed that temporal aliasing artefacts (judder and ghosting) contribute more to motion quality impairment than hold-type blur. In another study, Kuroki et al. [2007] measured motion quality up to a refresh rate of 480 Hz and found that the quality improves rapidly with increasing refresh rate but saturates after 240 Hz for natural images. Similar trends were also reported by Wilcox et al. [2015], Chapiro et al. [2019], Denes et al. [2020]. Similar to temporal aliasing, the visibility of flicker decreases with increasing refresh rate. These trends stay mostly the same for stereoscopic presentation [Hoffman et al., 2011]. However, unlike temporal aliasing, flicker can be detectable up to 500 Hz during saccadic eye motion [Davis et al., 2015].

Persistence: Also known as duty-ratio or shutter angle, is the fraction of the frame duration the image is displayed. A substantial difference in judderness can be perceived between shutter angles that differ by a large amount [Daly et al., 2015]. Furthermore, it is necessary to lower persistence to significantly reduce the hold-type blur inherent to LCDs as simply increasing the refresh rate is not effective [Sluyterman, 2006]. In fact, LCD panels with strobing backlights [Feng, 2006] are becoming increasingly common in head mounted displays (HMDs) and gaming monitors [Rejhon, 2017]. Unfortunately, psychophysical studies on the range of persistence used in these devices are quite limited (refer to Table 4.1).

Luminance and contrast: The models of contrast sensitivity [Barten, 2003] predict that the HVS is more sensitive to higher luminance and contrast, and thus, one would expect a similar trend for visibility of motion artefacts. Larimer et al. [2001] verified this in their study where they measured the effect of luminance, contrast, and refresh rate

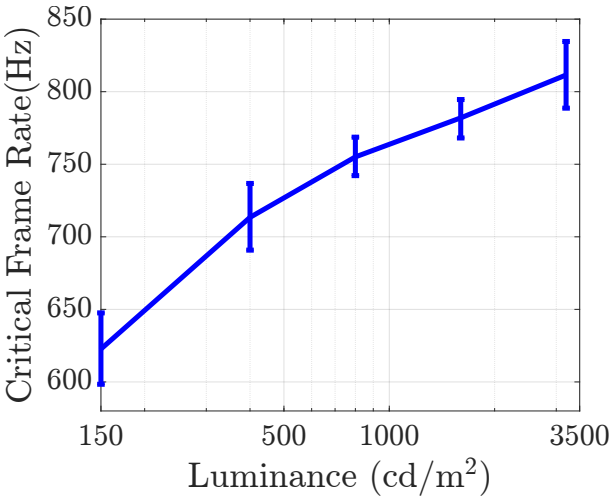


Figure 4.2: Mackin et al. [2016] experiment result confirms that critical frame rate grows non-linearly with stimulus luminance.

on judder and flicker up to 100 cd/m^2 and found luminance to be the dominant variable. Roberts and Wilkins [2013] noted that the visibility of flicker on a square-wave signal increases with increasing Michelson contrast, and Daly et al. [2015] reported similar degradation of overall motion quality with increasing contrast. Since most of these studies were done on displays with standard dynamic range, Mackin et al. [2016] measured the visibility of judder and ghosting as a function of luminance for up to 3200 cd/m^2. They found this relationship to be non-linear (in agreement with CSFs), with displays of 3200 cd/m^2 requiring a 30% higher refresh rate than displays at 150 cd/m^2 (Figure 4.2).

Resolution and spatial frequency: Visibility of blur is linearly dependent on the logarithm of the averaged spatial frequency of an image [Kuroki et al., 2007], with observers showing a slightly higher preference for higher spatial frequency content [Navarro et al., 2011, Daly et al., 2015]. The content resolution has also shown to be an important factor for rendering on variable refresh rate displays, with viewers picking higher refresh rates for low computational budgets and higher resolution for higher budgets [Debattista et al., 2018]. Reducing the resolution in rendering results in blur and aliasing artefacts, but how our sensitivity to these artefacts changes with velocity is still not well studied (Table 4.1). This gap in research is particularly important to address because most adaptive rendering algorithms involve dynamically varying screen resolution.

Other factors: While there is also some evidence of effect of motion direction [Daly et al., 2015], motion predictability [Denes et al., 2020], colour [Daly et al., 2015] and the type of eye motion (saccades and SPEM) [Roberts and Wilkins, 2013] on motion quality, I do not explore them in this dissertation. This is done partially to reduce the dimensionality of the problem and partially because it is difficult to accurately measure some of these dimensions with sufficiently low latency required for real-time rendering.

I summarise the above discussion in Table 4.1 and report the range of each parameter measured in the existing work and the types of motion artefacts studied. I find the studies of persistence and resolution to be particularly lacking as they do not cover the ranges commonly found in VR and mobile devices. Furthermore, spatio-temporal aliasing artefacts

due to low-resolution rendering are also not well considered in these works. Lastly, most of these studies used oversimplified stimuli such as lines and boxes that may not generalise well to complex image content. I attempt to fill these gaps in knowledge by conducting a psychophysical study on multiple display setups and more realistic texture content, as described in Section 4.3. In the first column of Table 4.1. I mark all datasets, including mine, that I will use to calibrate and validate my motion quality model in Chapter 5.

Table 4.1: Existing motion quality studies and the range of motion artefacts and display/content parameters explored by them. My experiments focused on understanding the effect of resolution and display persistence on visibility of motion artefacts, particularly blurring and aliasing artefacts caused by VRS.

Motion Quality Datasets	Motion Artifacts						Range of Display/Content Parameters Studied						
	Judder	**Flicker**	**Blur**	**Ghosting**	**Aliasing (Rendering)**	**Depth Distortion**	**Refresh Rate (Hz)**	**Persistence* (Duty Cycle)**	**Resolution (PPD)**	**Velocity (deg/s)**	**Adaptation Luminance (cd/m2)**	**Contrast (Michelson)**	**Stimuli**
Larimer et al. [2001]	✓	✓		✓			15 : 30	0.01 : 1	×	0.46 : 5.46	7 : 110	0.5 : 1	Vertical Bars
Kuroki et al. [2007]	✓		✓				60 : 480	CRT	64	0 : 80	40	Images	CG+Natural Images
Navarro et al. [2011]			✓		✓		60	1	110	5 : 15	300	×	CG Images
Hoffman et al. [2011]	✓	✓	✓	✓		✓	10 : 50	CRT	42	1 : 16	30 : 60	×	Moving Box
Roberts and Wilkins [2013]		✓					1000 : 5000	0.01	×	0	0.02 : 310	0.05 : 0.4	Stationary Lines
Johnson et al. [2014]	✓	✓	✓	✓		✓	30 : 240	0.25 : 0.5	60	0 : 25	10 : 50	×	Moving Box
Davis et al. [2015]		✓					25 : 1000	×	87	0	20 : 2700	0.09 : 0.95	Stationary Edge
Wilcox et al. [2015]	✓		✓				24 : 60	DLP	118	Videos	16	Videos	Natural Videos
Daly et al. [2015]	✓	✓	✓	✓			12 : 60	DLP	30	0.8 : 1.3	5 : 65	0.125 : 1	Gabor + Natural Images
Mackin et al. [2016]§	✓		✓	✓			60 : 2000	0.01	Printed Stimulus	10 : 70	150 : 3200	×	Moving Edge
Debattista et al. [2018]	✓		✓		✓		15 : 120	1	640 : 2560 px	3D Animation	350	Images	CG Videos
Denes et al. [2020]§	✓		✓				15 : 165	1	62	15 : 45	36	Images	CG+Natural Images
ITU-R Series [2020]§	✓	✓	✓	✓			60 : 240	0.3 : 0.9	7 : 60	8 : 32	169 : 480	Videos	Natural Videos
Ours§	✓		✓		✓		60 : 144	0.05 : 1	13 : 90	0 : 75	2.5 : 150	0.7 : 0.9	CG+Natural Images

* Persistence is the display's duty cycle and should not be confused with the camera's shutter angle. It is not well-defined for CRT and DLP displays due to asymmetric phosphorus decay time and use of colour wheel, respectively.
× Not reported in their publication.
§ Datasets that were used to calibrate/validate CaMoJAB.

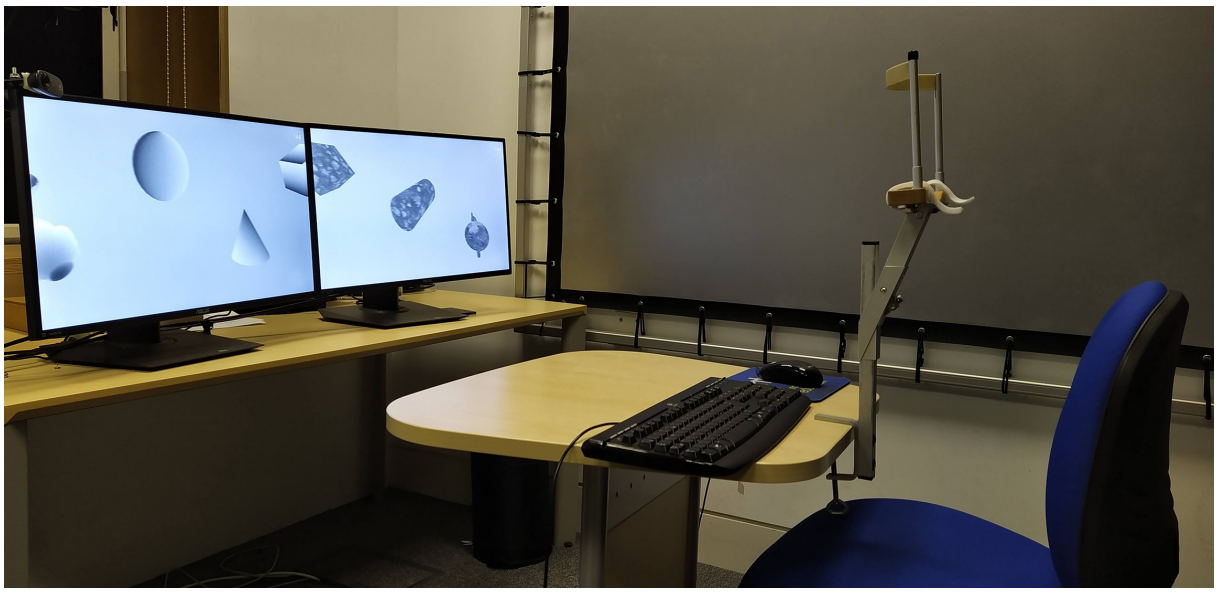


Figure 4.3: Experiment Setup. A chinrest was used to keep viewer's head at a fixed position.

4.3 VRS motion quality experiment

As discussed in the previous section, the impact of resolution, persistence and spatio-temporal aliasing on motion quality is not well-studied. To fill these gaps in understanding, I conduct a quality assessment experiment on a high refresh-rate display. In this experiment, I focus on shading resolution rather than raster resolution to explore how the quality degradation caused by VRS is affected by the above factors. Results are later used to calibrate my metric of judder, aliasing, and blur (Chapter 5).

Setup To study the effect of shading rate reduction on motion quality, observers were shown side-by-side two animations at different shading rates and textures on two separate displays (Figure 4.3). The experiment was divided into three blocks, each with a different configuration, simulating different display devices (Table 4.2). For the first two blocks, I used a pair of ASUS ROG Swift PG279Q 27" WQHD displays as they were capable of showing a wide range of refresh rates and luminance. For the last block, which studied low persistence, I simulated two virtual displays in a VR headset (Valve Index). I was unable to use the low persistence mode (ULMB) provided by the ASUS display as it led to excessive ghosting caused by strobe cross-talk [Rejhon, 2017]. The ASUS displays were calibrated with JETI Specbos 1211-2 spectroradiometer to achieve an accurate luminance reproduction and placed side-by-side to facilitate the comparison of horizontal motion. The experiment was designed in Unity3D, and I used VRS offered by Nvidia RTX 2080Ti for the shading rate reduction. I ensured that the experiment time did not exceed 30 minutes to prevent observer fatigue.

Stimuli Based on previous work detailed in Section 4.2, I identified the following factors to have the highest effect on the quality of motion: refresh rate, persistence, resolution, luminance, contrast, object velocity, and shading rate. Due to the high dimensionality of this problem, I restricted the experiment to three blocks (regions of the space) according

Table 4.2: Range of conditions tested in each block. The same 4 textures (Checkerboard, Noise, Grass, and Gradient) processed to the correct luminance and contrast were used for all blocks.

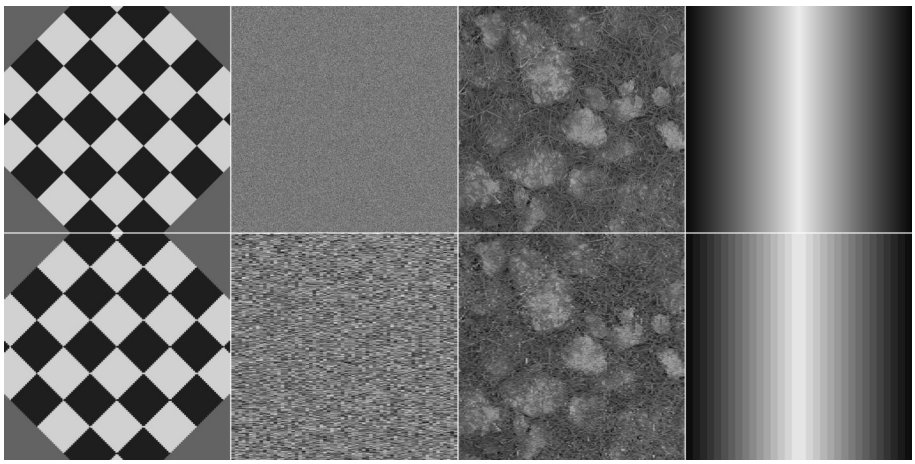
	Refresh Rate (Hz)	**Persistence**	**Velocity (deg/s)**	**Resolution (ppd)**	**Mean Luminance (cd/m^2)**	**Michelson Contrast**	**Device**
Block-PC	144	1	0, 10, 30	50	150	.7, .8, .9	ASUS
Block-Mobile	60	1	3, 10, 20	90	75	.7, .8, .9	ASUS
Block-VR	144	0.05[2]	10, 45, 75	13	2.5	.7, .8, .9	Valve Index

to what I believe to be the typical rendering configurations of our target application, namely, PC, mobile, and VR real-time rendering. The exact details of the range of each tested dimension can be found in Table 4.2. The resolution was varied by changing the viewing distance or scaling the viewport. For content, I used textured 3D models of simple shapes (cube, cone, sphere, cylinder, and capsule) and two popular, more complex models (Stanford bunny and Utah teapot). I used 4 textures: checkerboard, linear-gradient, salt-and-pepper noise, and grass, as shown in Figure 4.4. The textures and contrast were chosen to bring out the most common spatial and temporal artefacts associated with shading rate reduction and to include a wide range of variations in spatial frequency content. The textures were pre-processed to produce the selected luminance and contrast levels for each block. 2× supersampling anti-aliasing (SSAA) was used only in VR (Block-VR) to compensate for low PPD and to reduce artefacts' visibility. All the models moved horizontally at the same velocity. The velocities in each block were selected to be representative of common velocities in real-time graphics on respective displays. A small rotational velocity ($\leq 2\,^{deg}/_{s}$) was added to each model randomly so that all parts of the model were visible over the experiment. Since there was no vertical motion in our stimulus, we used the only possible combination of available shading rates that would keep the vertical rate constant while varying the horizontal shading rate between $\{1 \times 2, 2 \times 2, 4 \times 2\}$[1]. The experiment was divided into 3 separate sessions, one for each block.

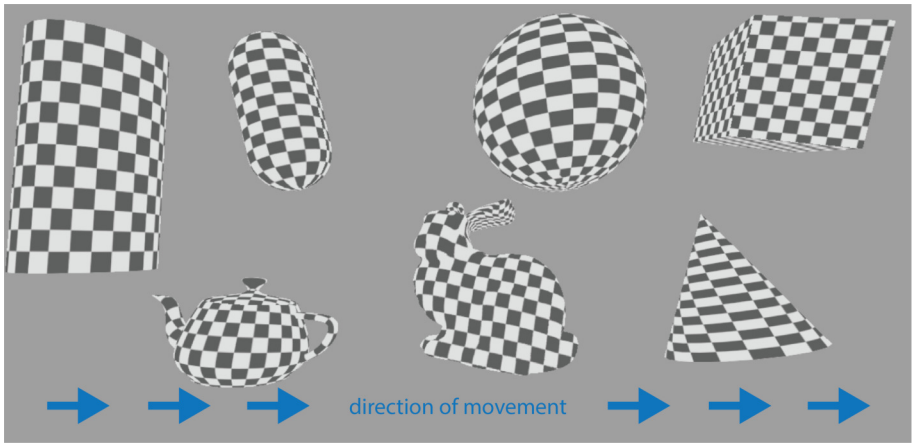
Procedure 13 participants aged 21–40, 4 females and 9 males with normal or corrected-to-normal vision, participated in the experiment. The experiment used a pairwise comparison protocol for quality assessment. In each trial, the participants were presented with two moving stimuli at different shading rates and potentially different textures. They had an option to see the reference stimuli rendered at the original resolution (VRS 1×1) by pressing and holding the space bar. A short blank of 0.5 s was shown before switching between test and reference animations so that the participants could not use temporal flicker to detect the presence of artefacts. Participants were then asked to pick the stimulus closer to its respective reference (stimulus with fewer artefacts). Before the experiment, all participants were briefed in a short training session where I highlighted the motion artefacts caused by VRS. Since we were limited to a relatively small number of participants due to COVID-19 restrictions, I collected more data from each participant to reduce confidence intervals. Each participant performed 120 comparisons for each of the three blocks (4680 comparisons across all participants). The order of comparisons was determined using

[1] A shading rate of ×N means $^{1}/_{N}$ shading resolution

[2] From manufacturer's website.



(a) Textures used in the experiment (top row) and their counterparts rendered using a lower shading rate (bottom row). The VRS was simulated and exaggerated using nearest neighbour downsampling in the above illustration.



(b) Horizontally moving and slowly rotating 3D models, rendered with the checkerboard texture, that were shown in the experiment.

Figure 4.4: Textures and objects used in the experiment.

ASAP, an active sampling method [Mikhailiuk et al., 2020] to maximise the information gained through each trial.

Results The pairwise comparison results from this experiment were mapped to a Just-Noticeable-Difference (JND) scale based on Thurstone Case V assumptions [Perez-Ortiz and Mantiuk, 2017]. A quality difference of 1 JND means that 75% of the population will pick one condition over another. Since the original formulation of the Thurstone Case V assumption does not allow for the computation of confidence intervals, bootstrapping was used to compute them. The results are reported in Figure 4.5. The shading rate of 1×2 was assigned as the 0 JND condition, and other data points are presented relative to it. The shapes of the plots are consistent with previous studies with quality impairment (ΔQ) due to reduced shading resolution decreasing with increasing velocity. However, this effect is not as prominent in VR (Block-VR). This can be explained by hold-type blur increasing with velocity and masking the spatio-temporal aliasing caused by reduced shading rate. Since VR has minimal hold-type blur because of its low persistence, the VRS artefacts

continue to be visible even at high velocities. Another important observation is that quality may degrade non-linearly with shading resolution and the shape of this curve varies widely between different textures, with artefacts in gradient rarely visible and artefacts in noise almost always visible. From these results, I confirm that the motion quality depends on a wide range of display and content-related factors. In the next chapter, I will describe how these results can be explained by a novel visual model of motion quality.

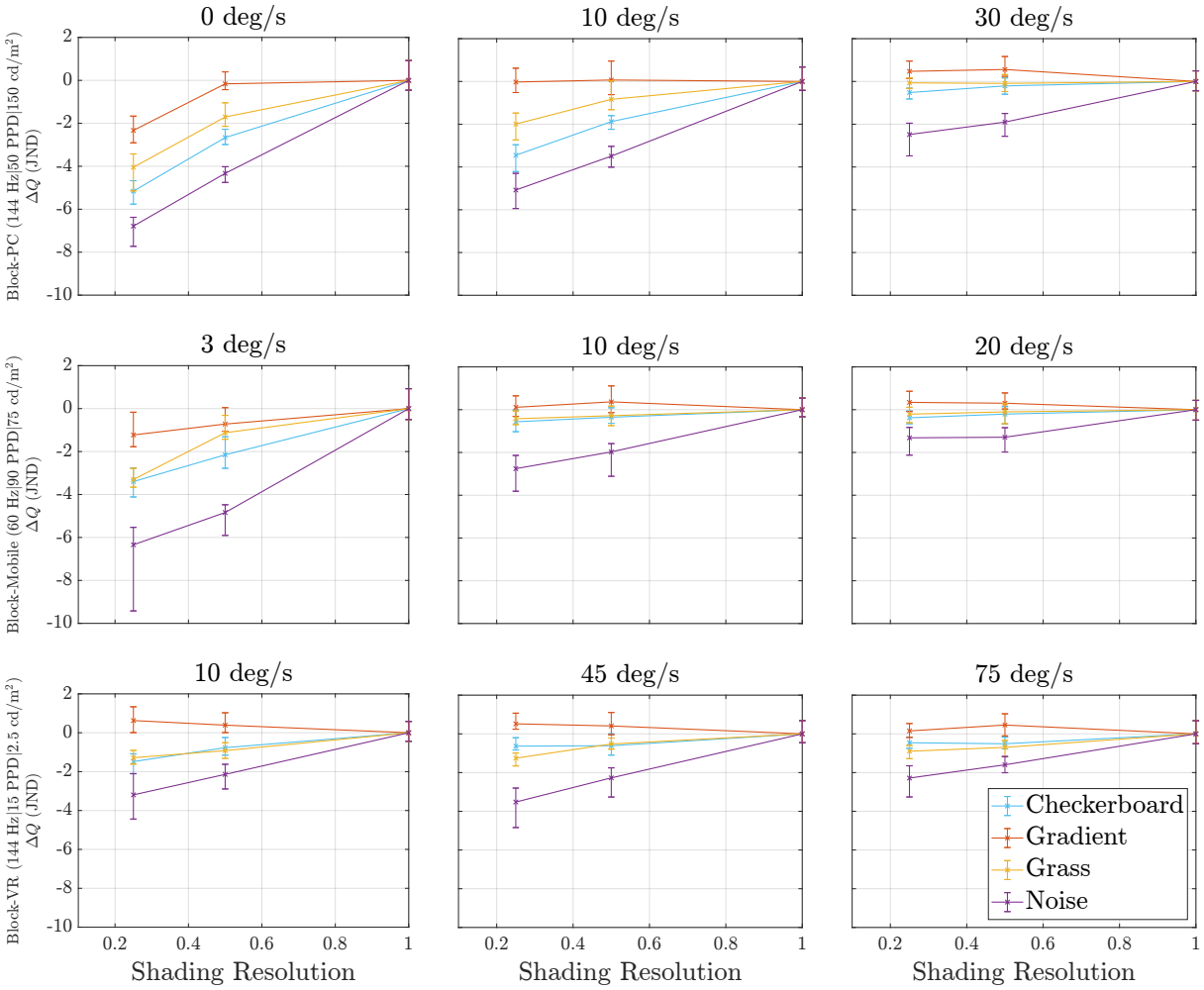


Figure 4.5: Results of VRS motion quality experiment 1 scaled to JND units.

4.4 Summary

To develop better models of motion quality, it is essential to understand how the perception of motion artefacts inherent to modern displays is affected by the different display and content parameters. In this chapter, I surveyed studies on the perception of motion artefacts and identified the following factors to have a significant effect on quality: display refresh rate, object velocity, display persistence, adaptation luminance, image contrast, display resolution and image spatial frequency content. I found the current studies on aliasing artefacts and the effect of persistence and resolution to be lacking. This was addressed by conducting a new subjective study that measured the loss in perceived quality due to shading rate reduction on a wide range of factors identified previously.

The experiment showed that this loss in quality decreases with velocity and varies widely between different textures and displays. These results imply an opportunity to save bandwidth in VRS rendering systems where the shading rate of regions with high velocities can be reduced without any change in quality.

Chapter 5

CaMoJAB: Content-adaptive model of judder, aliasing and blur

5.1 Introduction

When a continuous motion signal is recreated on a display, it can lead to different kinds of motion artefacts (Section 2.2). In traditional rendering, judder, aliasing, and blur are the most prominent of these motion artefacts. Understanding how these artefacts affect the visual quality of rendering is necessary. In Chapter 4, we discussed how the perception of these artefacts depends on several content and display factors; however, all existing metrics of motion quality only model a subset of these factors. In this chapter, I propose a new visual model that builds upon MARRR (explained in Section 3.2) and accounts for all relevant factors affecting motion quality in a content-adaptive manner. The model is tuned to account for distortions resulting from reduced shading rate to facilitate optimisation of VRS rendering but is designed in a white-box fashion to be easily extendable to different graphics pipelines. It isolates various spatial and temporal distortions resulting from the discrete nature of rendering and display systems and measures their impact on perceived quality using spatio-temporal contrast sensitivity functions of the visual system. The model is shown to explain the experimental data from the previous chapter and several other datasets with only a few fitted parameters.

I begin by discussing the state-of-the-art in modelling motion quality in Section 5.2. Then I introduce my model in Section 5.3, followed by its calibration, validation, predictions and ablation in the rest of the chapter.

This chapter is based on the work published in the SIGGRAPH Asia 2021 article:*Perceptual Model for Adaptive Local Shading and Refresh Rate.*

5.2 Related Work: Models of motion quality

Motion quality metrics aim to predict the visibility of motion artefacts given content and display information as input. Such models can then be used for various motion-adaptive rendering applications. Primarily, there are two main approaches to modelling quality: black-box and white-box metrics. Black-box metrics [Chapiro et al., 2019, Debattista et al., 2018] explain the data by fitting an arbitrary function and typically perform well within the dataset's domain but cannot extrapolate beyond it. White-box metrics, on the other hand, rely on psychophysical models and are better at extrapolating the predictions.

I focus on white-box metrics as I aim to build a metric that generalises to a wide range of display technologies and content.

Denes et al. [2020] proposed a white-box metric MARRR (Section 3.2) that models motion quality as a weighted sum of the quality of individual motion artefacts. The metric makes a simplifying assumption that motion artefacts are independent of each other which fails for content-adaptive scenarios where blur and spatio-temporal aliasing from rendering are often inter-dependent. Hoffman et al. [2011] analyse the frequency spectra of a moving line to predict the visibility of motion artefacts; however, it is unknown if such predictions will extend well to supra-threshold appearance. Both the metrics ignore the impact of content on motion quality.

Yang et al. [2019] derive a metric for full, half and quarter shading resolution using an l^2-norm of each tile with the previous frame in image-space modulated with a function of velocity. Though they account for the effect of content, they fail to account for several important perceptual and display factors that affect motion quality. Perceptual video quality metrics such as FovVideoVDP [Mantiuk et al., 2021] are also designed to be content and motion-aware, but their complexity precludes real-time rendering.

Furthermore, most existing models only consider a small subset of motion artefacts. One artefact that is rarely considered is temporal and spatial aliasing due to rendering (Table 4.1). Motion artefacts are often inseparable, and studying them in isolation does not translate well to practical applications. Aliasing due to rendering is also particularly important for our model as it is one of the most common artefacts resulting from the shading rate reduction. The summary of the discussed metrics can be found in Table 5.1. My proposed metric accounts for all the important factors described in Section 4.2 and outperforms the above models, as demonstrated in Section 5.6.

Table 5.1: Models of motion quality and the parameters they account for.

Motion Quality Metrics	Refresh Rate	Resolution	Persistence	Velocity	Luminance	Content-aware
Hoffman et al. [2011]	Yes	No	Yes	Yes	No	No
Debattista et al. [2018]	Yes	Yes	No	No	No	No
NAS Yang et al. [2019]	No	Yes	No	Yes	Yes	Yes
Chapiro et al. [2019]	Yes	No	No	Yes	Yes	No
MARRR Denes et al. [2020]	Yes	Yes	Yes	Yes	No	No
CaMoJAB (ours)	Yes	Yes	Yes	Yes	Yes	Yes

5.3 A content-adaptive model of motion quality

My goal is to design a motion quality model that, given a display configuration and an image, can predict how the quality will change with display refresh rate, shading resolution, display persistence, and image velocity. I aim to make this model as simple as possible for real-time usage while covering a wide range of display parameters and motion artefacts. The texture content is a major factor influencing quality degradation due to VRS (Section 4.3); however, most existing perceptual models ignore the effect of content on motion quality. Therefore, I also aim to make this model content-adaptive.

In this section, I develop a motion quality model by studying the evolution of a continuous motion signal and how it goes through various spatial and temporal distortions

due to the discrete nature of rendering and display systems. I focus on the distortion that VRS introduces to the mapped texture and assume that the distortion due to a lower resolution of shading (e.g. specular reflections) is negligible. The following analysis assumes SPEM and tracking on object, and does not consider changes in image content over time (e.g. moving transparent object). I define the motion quality q as a negate of the weighted sum of spatial distortions (d_s) and temporal distortions (d_t) and show that such a definition correlates well with the experimental results:

$$\begin{aligned} q(t(u,v),\nu,\boldsymbol{D}) = &- w_\mathrm{s} \cdot d_\mathrm{s}\left(t(u,v),\nu,\boldsymbol{D}\right) + \\ &- w_\mathrm{t} \cdot d_\mathrm{t}\left(t(u,v),\nu,\boldsymbol{D}\right), \end{aligned} \tag{5.1}$$

where $t(u,v)$ is the mean luminance (in $\mathrm{cd/m^2}$) of the image at u, v coordinates, ν is the velocity of the texture movement on the screen in $^{\mathrm{deg}}/_{\mathrm{s}}$, and $\boldsymbol{D}$ is a display configuration parameterised as:

$$\boldsymbol{D} = \{\widehat{\omega}, p, \widehat{\rho}, r_\mathrm{x}, r_\mathrm{y}\}, \tag{5.2}$$

where $\widehat{\omega}$ is the temporal Nyquist frequency of the display in Hz (half of display's refresh rate), p is the persistence (or duty-ratio) of the display ($0 < p \leq 1$), $\widehat{\rho}$ is the spatial Nyquist frequency of the display in cpd (half of display's spatial resolution in ppd), and $r_\mathrm{x}, r_\mathrm{y} \in \mathbb{Q}^+ \mid (0 < r_\mathrm{x}, r_\mathrm{y} \leq 1)$ are the shading resolution in x and y directions, respectively. The unit of q is JND.

This model is partially inspired by Watson's analysis of capture and display systems [Watson, 2013] and Denes et al.'s content-independent motion quality model (Section 3.2). However, I focus on understanding the effect of texture content on motion quality to enhance and guide VRS in real-time graphics. For the rest of this dissertation, I will refer to my model as ***CaMoJAB***, a content-adaptive model of judder, aliasing, and blur.

5.3.1 Spatial distortion model

In a typical VRS rendering pipeline, a texture goes through several stages before it is displayed (Figure 5.1). The texture is first quantised and stored as a mipmap. Next, during the texture look up, a mipmap level is selected according to the texel-to-pixel projection [Segal and Akeley, 2022, Section 8.14] and the shading resolution specified by VRS. The image is then displayed on the screen for a portion of the frame duration, which depends on the display persistence. If the object is in motion and is tracked by our gaze (SPEM), it may introduce hold-type or eye motion blur. All the above stages add different kinds of spatial distortions to the signal, resulting in a loss of quality. We will now analyse each of these steps in detail in the frequency domain (Figure 5.2).

Frequency domain analysis Analysing a visual signal in the frequency domain is a useful technique as it facilitates understanding sampling and filtering operations as well as allows for the use of various psychophysical models that are based on spatial frequency theory. Hence, I transform a given 2D texture $t(u,v)$ moving with a velocity ν in a horizontal direction into its frequency domain representation $T(\tau_\mathrm{u}, \tau_\mathrm{v})$ using *Discrete Fourier transform.*

$$T(\tau_\mathrm{u}, \tau_\mathrm{v}) = \mathcal{F}\{t(u,v)\} \quad \left[^{\mathrm{cycles}}/_{\mathrm{texel}}\right], \tag{5.3}$$

where τ_u and τ_v are the spatial frequencies in cycles-per-texel (cpt) corresponding to u and v direction, respectively. Though we will assume horizontal velocity in our model, I explain how it can be extended to arbitrary velocity in Section 6.3.3.

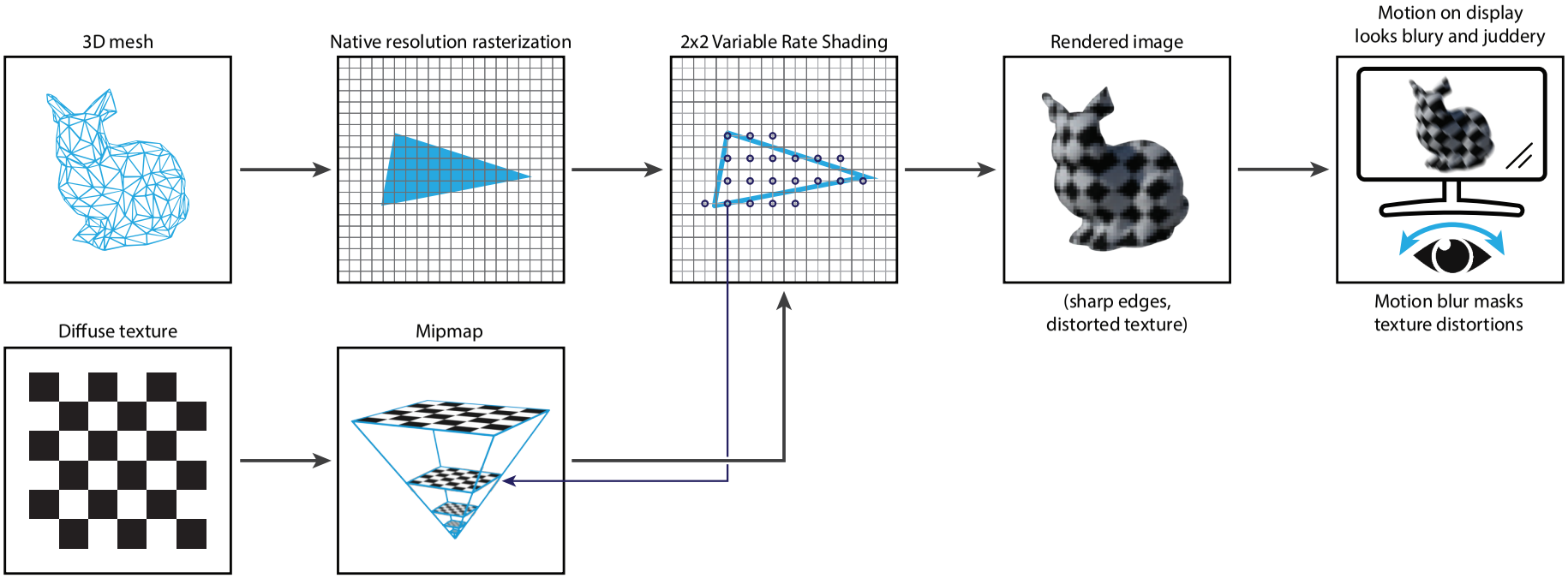


Figure 5.1: Spatial and temporal distortions in a VRS pipeline. Rasterization happens at native resolution but pixel shader is called once per multiple pixels. Larger pixels size leads to selection of coarser mipmap levels yielding distorted texture in the rendered image (artefacts exaggerated for illustration). When the rendered image is displayed at discrete time intervals, the motion appears blurry and juddery. The motion blur may mask the texture distortions caused by VRS.

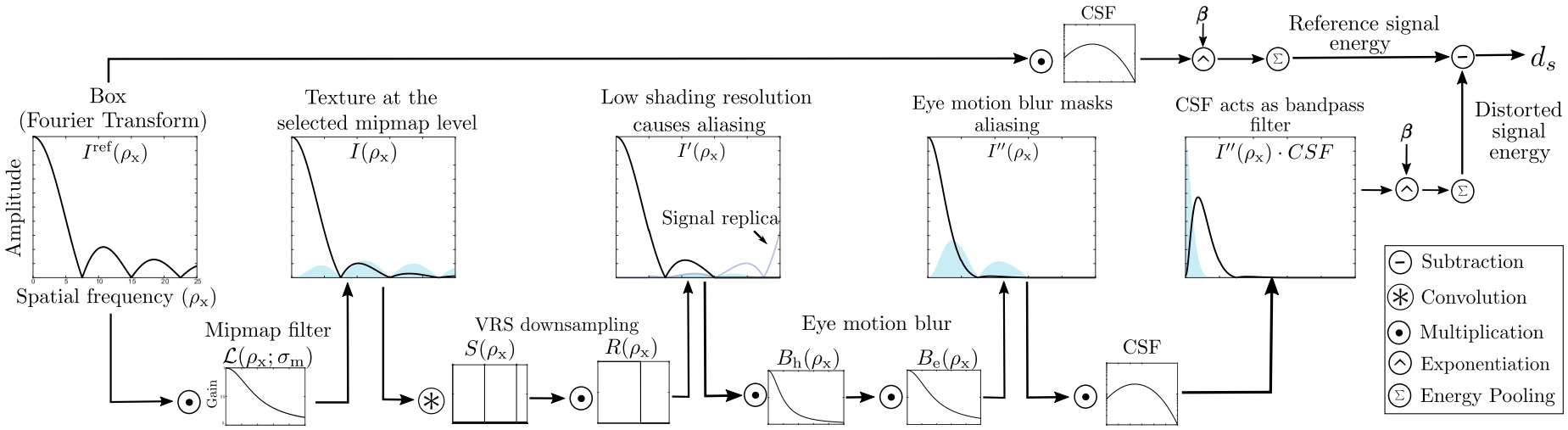


Figure 5.2: Illustration of spatial distortions introduced when an image with a sharp edge (box) is moving on the screen. The black line shows the signal in the frequency domain and the blue shaded region denotes the distortion (difference between the current and previous signal). Picking a low shading resolution leads to a selection of coarser mipmap levels resulting in a loss of high frequencies. Low shading resolution also results in the formation of aliases that distorts the lower frequencies. When the rendered image moves on a screen, our eyes undergo SPEM to follow it. This adds further blurring of the signal and may mask aliasing. The above distortions can be mapped to a JND scale by calculating the difference of CSF-normalised visual energy of the distorted and the reference signal.

Variable rate shading VRS executes the pixel shader once per multiple pixels and replicates the calculated colour to all those pixels. The shading typically involves a texture look-up operation on the *mipmap* of the underlying albedo/diffuse texture. Mipmapping [Williams, 1983] is the most popular antialiasing method for textures. A *mipmap* is a pyramid structure created by low-pass filtering and downsampling to half the resolution of each fine level to create a coarser level. The common filters used for this purpose are box, Gaussian, Lanczos, and Kaiser [Akenine-Möller et al., 2019, Section 6.2.2]. In the following discussion, I assume a box filter as it is one of the most commonly used filters and yet is also known to be one of the worst possible filters because it unnecessarily blurs low frequencies while retaining some high frequencies that cause aliasing [Akenine-Möller et al., 2019]. For every shaded pixel, the area of the pixel is projected onto the texture. The projected area may include 1 or more texels. The level of detail to sample from is then selected in such a way that the ratio of pixel-to-texel area is close to 1:1. Most modern shading languages provide a function to calculate and query this value (such as *textureQueryLod()* in GLSL). Since VRS enlarges the pixel area, it results in the selection of a coarser mipmap level. Let l be the mipmap level selected when rendering at full shading resolution such that $l = 0$ is the highest (finest) resolution level. The width of the box filter at a lower shading resolution can be approximated as:

$$b_\mathrm{m} = \frac{2^l}{\min(r_\mathrm{x}, r_\mathrm{y})} \quad \left[\frac{\text{texel}}{\text{pixel}}\right] . \tag{5.4}$$

The symbols used in this equation are defined in Eq. (5.2). By taking the minimum of the two shading resolutions, I assume isotropic filtering, but it is trivial to extend it to support anisotropic filtering. I will use upper case symbols to denote functions in the Fourier domain. The Fourier transform of our box filter gives:

$$B_\mathrm{m}(\tau_\mathrm{u}, \tau_\mathrm{v}; b_\mathrm{m}) = \mathrm{sinc}(b_\mathrm{m}\tau_\mathrm{u}) \cdot \mathrm{sinc}(b_\mathrm{m}\tau_\mathrm{v}) . \tag{5.5}$$

However, using a sinc$(\cdot)$ function introduces non-monotonicity in the model predictions making it unsuitable for optimisation problems (example in Appendix B). To mitigate this, I approximate sinc$(\cdot)$ with a Lorentzian function $\mathcal{L}(\cdot)$ of half-width σ_m:

$$\mathcal{L}(\tau; \sigma_\mathrm{m}) = \frac{\sigma_\mathrm{m}^2}{\sigma_\mathrm{m}^2 + 4\tau^2} , \tag{5.6}$$

$$\sigma_\mathrm{m} = \frac{\pi}{2b_\mathrm{m}} . \tag{5.7}$$

I found Lorentzian function to be a good approximation of sinc$(\cdot)$ as it preserves the high frequencies similar to sinc$(\cdot)$, unlike the Gaussian approximation used by Denes et al. [2020] (refer to Figure 5.3).

The texture at a mipmap level l is therefore affected by a low-pass filter as follows:

$$T'(\tau_\mathrm{u}, \tau_\mathrm{v}) = T(\tau_\mathrm{u}, \tau_\mathrm{v}) \cdot B_\mathrm{m}(\tau_\mathrm{u}, \tau_\mathrm{v}; b_\mathrm{m}) , \tag{5.8}$$

$$B_\mathrm{m}(\tau_\mathrm{u}, \tau_\mathrm{v}; b_\mathrm{m}) \approx \mathcal{L}(\tau_\mathrm{u}; \sigma_\mathrm{m}) \cdot \mathcal{L}(\tau_\mathrm{v}; \sigma_\mathrm{m}) . \tag{5.9}$$

Since we will now operate in screen-space, I map the mipmapped texture $T'(\tau_\mathrm{u}, \tau_\mathrm{v})$ to a screen space image $I(\rho_\mathrm{x}, \rho_\mathrm{y})$:

$$I(\rho_\mathrm{x}, \rho_\mathrm{y}) = T'(\tau_\mathrm{u}\, 2\,\widehat{\rho}\, b_\mathrm{m} , \tau_\mathrm{v}\, 2\,\widehat{\rho}\, b_\mathrm{m}) \quad [\text{cycles}/\text{deg}] , \tag{5.10}$$

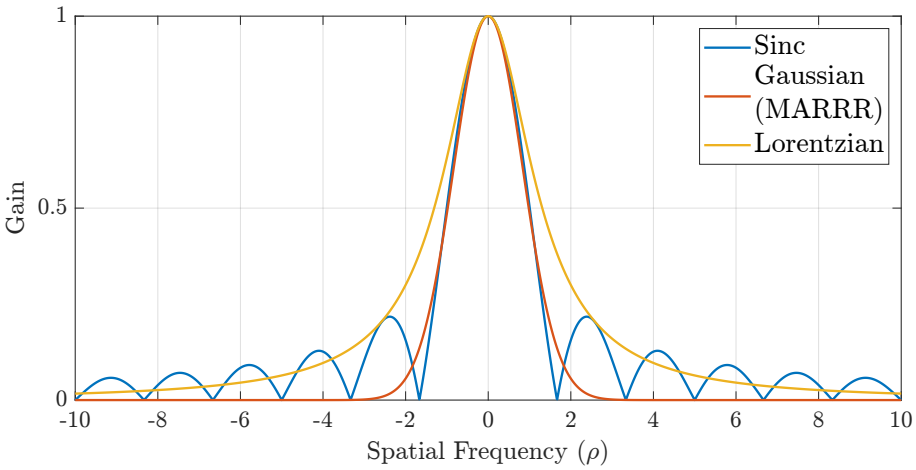


Figure 5.3: Approximating sinc with monotonic functions.

where ρ_x and ρ_y are spatial frequencies in cpd.

VRS repeats the above texture look-up operation for all the "coarse" pixels (large VRS pixels). This is same as downsampling the filtered signal $I(\rho_x, \rho_y)$ by convolving it with a sampling function $S(\rho_x, \rho_y)$. The sampling function is a sequence of impulses at an interval of $2\,\widehat{\rho}\,r_x$ and $2\,\widehat{\rho}\,r_y$ in x and y directions, respectively. It can be represented using a Dirac comb function $\text{Ш}(\cdot)$:

$$S(\rho_x, \rho_y) = \text{Ш}\left(\frac{\rho_x}{2\,\widehat{\rho}\,r_x}\right) \text{Ш}\left(\frac{\rho_y}{2\,\widehat{\rho}\,r_y}\right) . \tag{5.11}$$

In my implementation of $S(\rho_x, \rho_y)$, I only considered the first two impulses to limit the discontinuities introduced by the function (more details in Appendix B).

A display cannot render spatial frequencies higher than its Nyquist frequency. Hence, we can clip the high frequencies by multiplying the resultant signal with a rectangular function $R(\cdot)$:

$$I'(\rho_x, \rho_y) = \left(I(\rho_x, \rho_y) * S(\rho_x, \rho_y)\right) \cdot R(\rho_x, \rho_y) , \tag{5.12}$$

where the rectangular function is defined as:

$$R(\rho_x, \rho_y) = \Pi\left(\frac{\rho_x}{2\,\widehat{\rho}\,r_x}\right) \Pi\left(\frac{\rho_y}{2\,\widehat{\rho}\,r_y}\right) , \tag{5.13}$$

and $\Pi(\cdot)$ is a rectangular function[1].

The above process is illustrated in Figure 5.2 which analyses a moving box in the frequency domain. VRS picks coarse mipmap levels that act as a low-pass filter resulting in loss of energy. The filtered signal is then downsampled through a convolution with $S(\rho_x, \rho_y)$ that results in the creation of replicas (*aliases*) of the spectrum at multiples of the sampling interval. It is followed by multiplication with $R(\rho_x, \rho_y)$ which results in downsampling blur. If the bandwidth of the spectrum I is larger than display's effective Nyquist frequency $(\widehat{\rho}\,r_x, \widehat{\rho}\,r_y)$, higher frequencies *fold* onto lower frequencies and cause spatial aliasing.

[1]The rectangular function is equal to 1 in $(-0.5, 0.5)$, 0.5 at -0.5 and 0.5 and 0 otherwise.

Eye motion blur When the rendered image moves on the screen, our eyes undergo SPEM to track it. The motion of our eyes is continuous, however, an LCD presents the image at discrete moments in time (frames). This leads to smearing of the image on the retina and is known as *hold-type blur*. The hold-type blur can be reduced by reducing the time for which the frame is displayed (persistence). I follow the same practice as Denes et al. and approximate hold-type blur with a box filter of width:

$$b_{\mathrm{h}} = \frac{\nu\, p}{2\,\widehat{\omega}} \quad [\mathrm{deg}]\,, \tag{5.14}$$

where the parameters are the same in Eq. (5.2). The display persistence $p \in [0, 1]$ accounts for shorter integration time on low persistence displays which reduces hold-type blur. Denes et al. showed that when our eye undergoes SPEM, its tracking is imperfect. Our eyes often over- or under-estimate the object's location, which leads to additional blurring. The degree of blurring is proportional to the object velocity. It can also be modelled with a box filter of width:

$$b_{\mathrm{e}} = (c_{\mathrm{a}}\,\nu + c_{\mathrm{b}}) \cdot p \quad [\mathrm{deg}]\,, \tag{5.15}$$

where $c_{\mathrm{a}} = 0.001648$ and $c_{\mathrm{b}} = 0.079818$ are the coefficients reported in their work.

The blur due to the hold-type blur and imperfect eye motion is modelled the same way as the loss of resolution due to VRS and mipmapping, using Lorentzian functions in the frequency domain (approximations of sinc filter):

$$B_{\mathrm{h}}(\rho_{\mathrm{x}}, \rho_{\mathrm{y}}) = \mathcal{L}(\rho_{\mathrm{x}}; \sigma_{\mathrm{h}}), \tag{5.16}$$

$$B_{\mathrm{e}}(\rho_{\mathrm{x}}, \rho_{\mathrm{y}}) = \mathcal{L}(\rho_{\mathrm{x}}; \sigma_{\mathrm{e}}), \tag{5.17}$$

where $\sigma_{\mathrm{h}} = {}^{\pi}/_{2b_{\mathrm{h}}}$ and $\sigma_{\mathrm{e}} = {}^{\pi}/_{2b_{\mathrm{e}}}$.

Note that the blurring due to eye motion is different from the blurring introduced in the previous stages (Eq. (5.9)). It is directional in nature and only happens parallel to object motion, unlike blurring due to mipmapping and downsampling which is omnidirectional.

The final distorted signal reaching our retina can be calculated as

$$I''(\rho_{\mathrm{x}}, \rho_{\mathrm{y}}) = I'(\rho_{\mathrm{x}}, \rho_{\mathrm{y}}) \cdot B_h(\rho_{\mathrm{x}}, \rho_{\mathrm{y}}) \cdot B_e(\rho_{\mathrm{x}}, \rho_{\mathrm{y}})\,. \tag{5.18}$$

Contrast sensitivity and energy pooling To find the sensitivity of HVS to various distortions, we can modulate the signal with the spatio-temporal CSF $S(\rho, \omega, L_a, r)$. For a given spatial frequency ρ, temporal frequency ω, adaptation luminance L_a, and stimulus radius r, the CSF specifies sensitivity, i.e., the reciprocal of minimum contrast threshold visible to an average observer. For my model, I use the CSF defined by Mantiuk et al. [2021] because it accounts for all three required dimensions together and is fitted to data up to 10 000 $\mathrm{cd/m^2}$.

Finally, we can compute the overall visual energy E_s of the distorted signal by modulating it with the CSF and integrating over the entire spectrum:

$$E_{\mathrm{s}}(I'') = \int_0^{\widehat{\rho}} \int_0^{\widehat{\rho}} \left| I''(\rho_{\mathrm{x}}, \rho_{\mathrm{y}})\, S\left(\sqrt{\rho_{\mathrm{x}}^2 + \rho_{\mathrm{y}}^2}, \omega, L_a, r\right)\right|^{\beta} d\rho_{\mathrm{x}} d\rho_{\mathrm{y}}, \tag{5.19}$$

where ω is the temporal frequency, assumed to be 0, β is the power parameter of the model and is fitted to the psychophysical data in Section 5.4. I set the temporal frequency

ω to 0 as we consider the stimulus that is stabilised on the retina and includes blur due to motion. I set the stimulus radius r to $\frac{1}{5}^\circ$ (the reciprocal of peak frequency). The adaptation luminance L_a is approximated as the mean luminance of the considered image part (VRS block). The CSF-normalised energy E_s can be linearly mapped to a JND scale, as shown by Denes et al. [2020].

Given a reference signal with no distortion

$$I^{\text{ref}}(\rho_{\text{x}}, \rho_{\text{y}}) = T(2\widehat{\rho}\,\tau_{\text{u}}\,, 2\widehat{\rho}\,\tau_{\text{v}}) \quad \left[{}^{\text{cycles}}/_{\text{deg}}\right], \tag{5.20}$$

I define the total spatial distortion (d_{s} in Eq. (5.1)) as the visual energy difference[2] between the reference signal and the distorted signal:

$$d_{\text{s}} = E_{\text{s}}(I^{\text{ref}}) - E_{\text{s}}(I''). \tag{5.21}$$

5.3.2 Temporal distortion model

When a continuous motion signal is sampled and displayed at discrete time intervals, the motion appears to be *jerky* or *juddery*. As discussed in Section 4.2, the amount of judder increases with velocity and decreases with increasing refresh rate. This phenomenon can be explained by analysing the signal in the frequency domain [Watson, 2013].

I model the temporal judder (non-smooth motion) similarly to MARRR (Section 3.2) but in a content-adaptive manner. Temporal sampling creates replicas (aliases) of the signal in the frequency domain, each separated by the display's refresh rate, $2\,\widehat{\omega}$ (see Figure 5.4). If the temporal bandwidth of the signal exceeds the Nyquist frequency of the display ($\widehat{\omega}$), the replica will overlap with the original signal and distort it. The temporal frequency of the first replica is $2\,\widehat{\omega}$ [Smith et al., 1997, Chapter 3], and the shift in its spatial frequency can be calculated as:

$$\rho_{\text{A}} = \frac{2\widehat{\omega}}{\nu} \quad \left[\frac{\text{cycles}}{\text{deg}}\right]. \tag{5.22}$$

We can safely ignore the vertical spatial frequency ρ_{y} as we assumed the vertical velocity to be 0, so there is no temporal distortion. The total temporal distortion (d_{t} in Eq. (5.1)) can then be approximated as the CSF-normalised visual energy in the region of the replicated spectrum that is less than the spatial Nyquist frequency of the display ($\widehat{\rho}$):

$$d_{\text{t}} = \int_0^{\widehat{\rho}} \int_0^{\widehat{\rho}} \left| I'(\rho_{\text{x}} - \rho_{\text{A}}, \rho_{\text{y}})\, S\left(\sqrt{\rho_{\text{x}}^2 + \rho_{\text{y}}^2}, 2\widehat{\omega}, L_{\text{a}}, r\right)\right|^{\beta} d\rho_{\text{x}}\, d\rho_{\text{y}}\,, \tag{5.23}$$

Similar to Eq. (5.19), I approximate adaptation luminance L_a as the mean luminance of the considered image part (VRS block) and set the stimulus radius r to $\frac{1}{5}$.

5.4 Model calibration

To determine the free parameter β and the relative weights of the spatial and temporal distortions w_{s} and w_{t}, I use the psychophysical data from my VRS motion quality Experiment

[2]Different from Watson and Ahumada's [2011] visible contrast energy (ViCE). I found the above formulation to be more robust to numerical inaccuracies. More details in Appendix B and Appendix A.

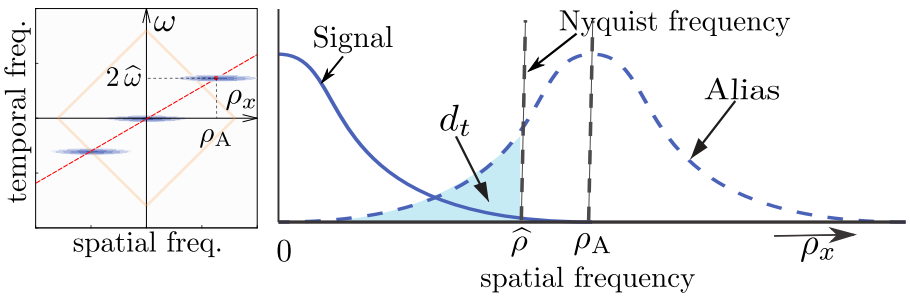


Figure 5.4: Temporal distortion of the signal due to low refresh rate.

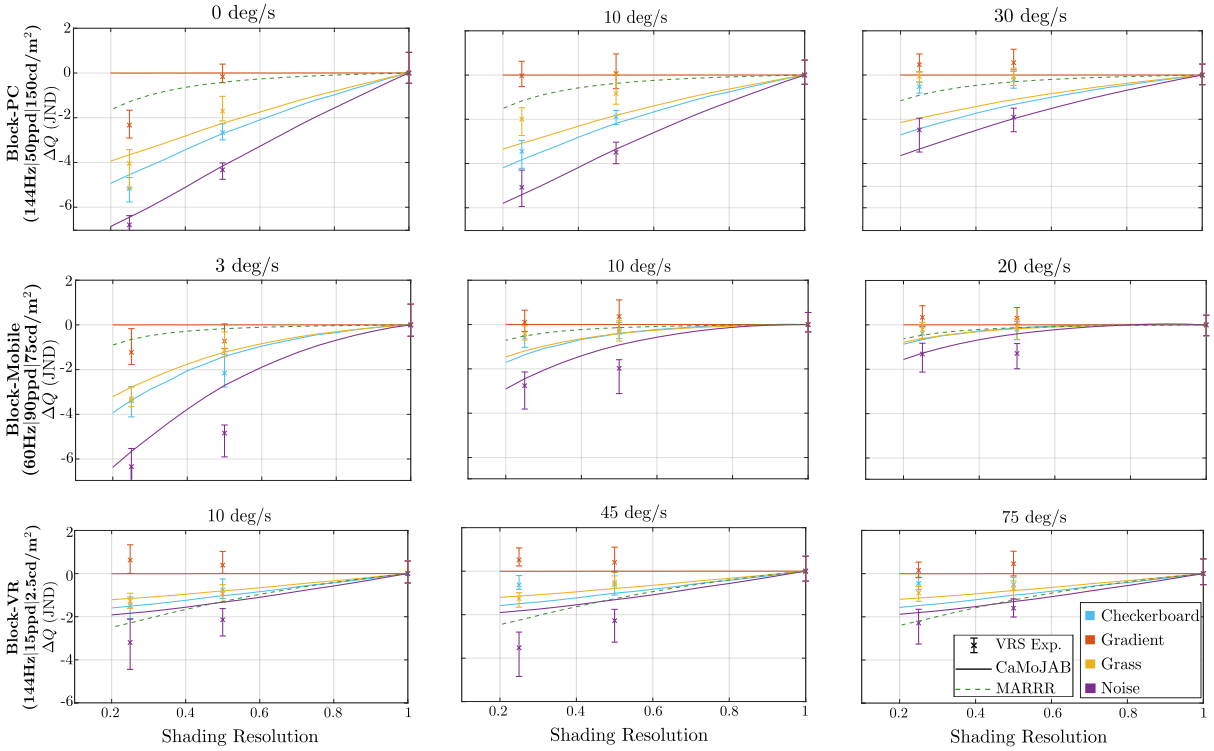
1 (Section 4.3) in addition to Experiment 1 and 3 from Denes et al. [2020]. The experiment by Denes et al. measured supra-threshold motion quality through moving checkerboards at different refresh rates and velocities on a full persistence LCD and no-reference pairwise comparisons. The two datasets use the same JND scale and together include a wide range of display settings and all three artefacts, i.e., motion blur, downsampling artefacts, and judder (refer to Table 4.1 for the exact range of factors tested). The values of the parameters are determined by minimising the root-mean-squared error between the model predictions and the results of the experiments. The fitted parameter values (RMSE = 0.53) are reported in Table 5.2, and the results are plotted in Figures 5.5a and 5.5b. CaMoJAB correctly predicts the reduction in quality differences with increasing velocity and successfully captures the relationship between different textures across all three configurations. An ablation study analysing the contribution of each of the CaMoJAB's component can be found in Section 5.7.

Table 5.2: Model parameters

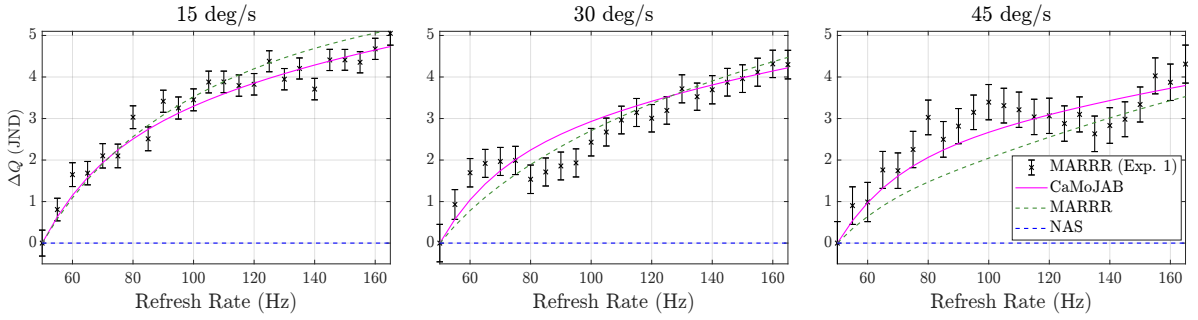
w_s	w_t	β
3.69	2.9	0.36

5.5 Model results

I provide a visualisation of CaMoJAB predictions for the moving grass texture at different velocities, refresh rates, and shading resolutions in Figure 5.6. The first plot predicts that as the velocity (in $^{\text{deg}}/_{\text{s}}$) increases, the difference in quality between high and low shading resolution decreases. This is consistent with our observations in the experiment (Section 4.3). The second plot predicts that at lower velocities, observers prefer higher shading resolution over refresh rate and the trend reverses as the velocity increases. This is consistent with the findings of Denes et al. [2020]. These model predictions hold high practical significance as they can be used to drive real-time rendering on performance adaptive hardware such as variable refresh rate (VRR) displays or VRR compatible cloud gaming [Colenbrander, 2021] and VRS GPUs. I describe and implement one such strategy in a complex motion setting in Chapter 6. Finally, CaMoJAB also provides an opportunity to study the interplay between luminance and persistence (Figure 5.7) for rendering on the upcoming variable persistence-variable refresh rate displays [Verbeure et al., 2017, Hekstra et al., 2008].



(a) VRS Experiment results and CaMoJAB predictions scaled to JND units.



(b) CaMoJAB predictions on Denes et al. [2020]'s Experiment 1.

Figure 5.5: VRS motion quality experiment results and CaMoJAB predictions. CaMoJAB was calibrated together on all the results reported in (a) and (b) and is capable of capturing the correct trends. MARRR being a content-independent model, is unable to capture the difference in quality due to texture in (a). NAS does not account for display factors and thus fails to predict the change in quality due to refresh rate in (b).

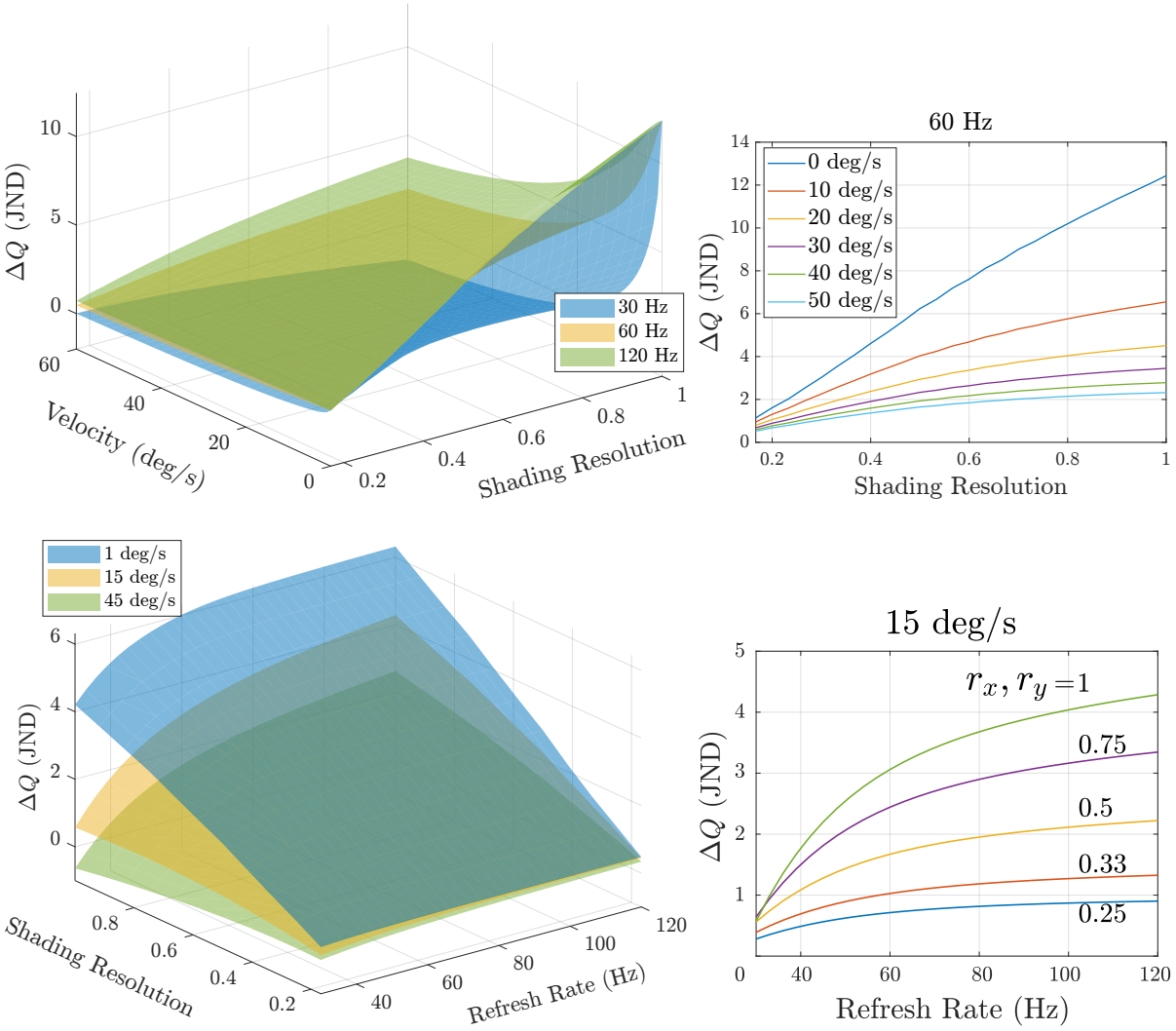


Figure 5.6: CaMoJAB predictions for a moving grass texture on a 150 cd/m^2 and full persistence 60 ppd display. Although shading resolution is plotted as a continuous variable, in practice, only a small set of resolutions are available.

5.6 Comparison and validation

For a comparative study, I picked two state-of-the-art models [Denes et al., 2020] and [Yang et al., 2019]. It should be noted that neither of these models (or any other model to my knowledge) is designed to predict our kind of data. [Denes et al., 2020] is a perceptually based model derived through extensive psychophysical experiments, however, it is content-independent. [Yang et al., 2019] though content dependent, does not take display and viewing conditions into consideration. Furthermore, it is designed for a specific task to give quality scores for only full, half, and quarter shading resolution and not a continuous scale. For a fair comparison, both models were linearly fitted to our data. As can be seen from the quantitative comparison in Table 5.3, CaMoJAB predicts the trends significantly better than the other two models across all datasets. For a more extensive comparison, please refer to Appendix C.

To further validate and verify the generalizability of CaMoJAB, I test its performance on motion quality datasets from [Mackin et al., 2016] and ITU-R recommendations [Series,

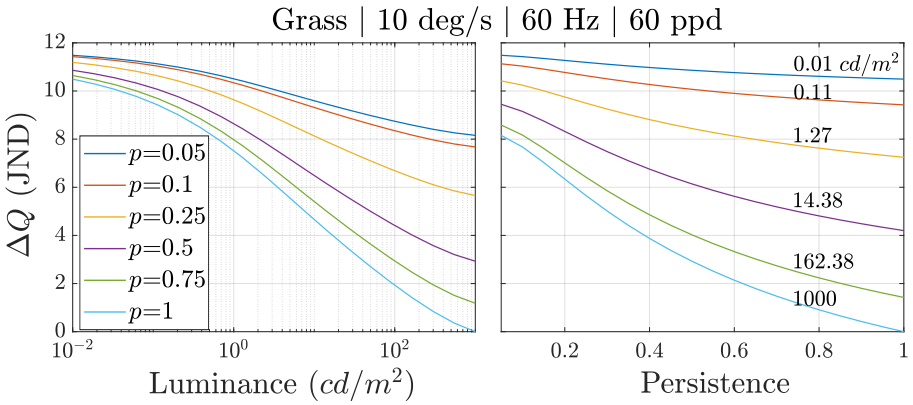


Figure 5.7: Luminance-persistence trade-off for a 60 Hz animation on a 60 ppd display. The motion artifacts in such animation are more noticeable at high luminance and for longer display persistence.

Table 5.3: MARRR, NAS, and CaMoJAB (Ours) performance on motion quality datasets. All models were linearly fitted to each dataset. Numbers denote the RMSE values. I report RMSE corresponding to a moving checkerboard for the ITU dataset as they did not report the stimulus used.

	Block-PC	Block-Mobile	Block-VR	MARRR Exp. 1	MARRR Exp. 3	Mackin et al. 2016	ITU-R FPS Series [2020]	ITU-R PPD Series [2020]
CaMoJAB (ours)	**0.74**	**0.57**	**0.53**	**0.38**	**0.03**	**0.42**	0.28	0.69
MARRR	1.82	1.55	1.14	0.55	0.04	0.55	**0.15**	**0.25**
NAS	1.41	1.1	1.23	3.12	0.21	1.04	1.09	1.19

2020]. The experiment by Mackin et al. used the mean opinion score methodology to study motion quality of real-continuous motion of rotating lines on a low persistence display (simulated via strobing lights on a printed paper). The experiments in ITU-R [Series, 2020] use natural videos as stimuli, and their results form the basis of presently in-force ITU recommendations.

I linearly scaled our model to fit their data and bring the predictions to the same perceptual scale (Figure 5.8). I report the goodness of fit (RMSE) for a moving checkerboard in Table 5.3. The results show that CaMoJAB also generalises well to more complex stimuli.

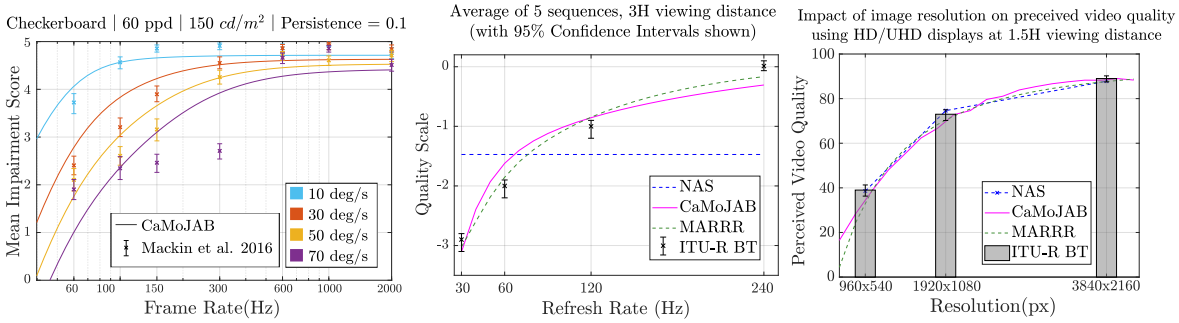


Figure 5.8: CaMoJAB's validation on [Mackin et al., 2016] and ITU-R [Series, 2020]. It generalises well to complex stimuli such as natural videos and real scenes.

Table 5.4: Ablation Study. ✓implies if the component was enabled. Numbers denote the RMSE values for each configuration.

$\{B_m, S, R\}$	$\{B_h, B_e\}$	d_t	Block-PC	Block-Mobile	Block-VR	MARRR Exp. 1	MARRR Exp. 3	**Total**
		✓	2.31	1.75	1.19	0.53	**0.02**	1.42
	✓		2.31	1.75	1.19	0.55	0.21	1.43
✓			1.25	1.22	0.63	3.11	0.21	2.11
✓	✓		**0.73**	**0.56**	**0.53**	0.6	0.21	0.6
✓		✓	1.19	0.98	0.55	0.5	0.05	0.79
	✓	✓	2.31	1.75	1.2	**0.37**	0.05	1.41
✓	✓	✓	0.74	0.57	**0.53**	0.38	0.03	**0.53**

5.7 Ablation study

To understand the contribution of each component of my model, I perform an ablation study. I isolate three main components of CaMoJAB: VRS blur and downsampling (B_m, S, R), eye-motion blur (B_h, B_e), and temporal distortion model (d_t). I disable different combinations of the three components at a time and refit the model to the data. The goodness of fit (RMSE) is reported in Table 5.4. The results indicate that each component provides a poor fit to the data in isolation. Eye-motion blur and temporal distortion model are integral in explaining the motion quality in Denes et al. [2020] data but fail to capture the effect of shading resolution on motion quality, i.e., Block-PC, Block-Mobile, and Block-VR. VRS blur and downsampling, and eye-motion blur are crucial and sufficient in explaining the results of my experiment. This is because the test and the reference conditions only varied in resolution, making the temporal distortion negligible compared to spatial distortion ($d_\mathrm{t}^{ref} - d_\mathrm{t}^{test} \ll d_\mathrm{s}^{ref} - d_\mathrm{s}^{test}$) for this data. Similarly, the temporal distortion model is sufficient to predict Denes et al. [2020] Exp. 3 results because this experiment ensured equal spatial distortion between reference and stimulus conditions in their experiment. As seen from the "Total" column of Table 5.4 that reports the aggregated RMSE values for all datasets together, we get the best fit when all three components are enabled.

5.8 Limitations

The CaMoJAB metric accounts for most relevant factors of motion quality, including texture content, on-screen velocities, luminance, effective resolution, and display persistence. However, it does not model contrast masking [Legge and Foley, 1980, Watson and Solomon, 1997], degradation of colour, foveation [Tursun et al., 2019] or saliency. Those limitations are partially dictated by the scarcity of available data and partially by the real-time requirement of our application. The complexity of contrast masking would make the metric too costly for a real-time application, and foveation would require eye tracking. Also, CaMoJAB does not account for individual variations and instead models the average observer. Modelling individual variations would require much more measurements from each participant. CaMoJAB assumes the observed texture to be constant and thus precludes dynamic effects such as transparency, specular reflections, shadows, and procedural textures. We will see in the next chapter how the model still performs well in complex settings despite these limitations.

5.9 Summary

As our understanding of HVS improves, so do our models of its limitations. In this chapter, I proposed a more complete model of three prominent motion artefacts: judder, aliasing, and blur. The model evaluates how the three artefacts are affected by several content and display parameters such as refresh rate, resolution, persistence, luminance, shading rate, velocity, and spatial frequency, and how they are masked by hold-type blur, eye motion blur, and, limited spatio-temporal sensitivity of the visual system. The model is shown to outperform current state-of-the-art and explain multiple datasets with only a few fitted parameters. The results of the model provide a promising avenue to optimise visual quality in various graphics and display pipelines.

Chapter 6

ALSaRR: Adaptive local shading and refresh rate

6.1 Introduction

Shading is one of the most computationally expensive parts of the rendering pipeline, yet the demand for shading computations is growing significantly with the increase in both the resolution and refresh rate of displays. Even the most powerful GPUs cannot meet these demands and are limited by their computational power and bandwidth. Furthermore, with the growing popularity of mobile gaming, which needs to operate under a limited power budget, and GPU-sharing cloud gaming [Yadav and Annappa, 2017], rendering often needs to operate at a fraction of the maximum GPU capacity. To address this challenge, all popular GPU manufacturers have introduced a more flexible shading mechanism called VRS in their next-gen chipsets (Section 2.4). VRS enables the control of shading resolution within each 16×16 image tile while retaining visibility computation at the native resolution. VRS has been used to exploit the limits of the HVS by intelligently distributing the shading budget based on foveation [Tursun et al., 2019], scene content and motion [Yang et al., 2019], or depth-of-field [Intel, 2019]. All these works propose a dynamic quality control mechanism that allocates the rendering budget to those aspects of an image or animation that have the highest impact on the overall quality. In this chapter, I propose to control both the VRS state map and the refresh rate based on all major factors affecting image quality: texture content, on-screen velocities, luminance, effective resolution, and display persistence. Owing to VRS that allows for local control of resolution, I extend the MARRR rendering method proposed in Chapter 3 using my CaMoJAB metric (from Chapter 5) to further optimise visual quality without the need for eye tracking. Unlike the aforementioned existing works, which control VRS to avoid any visual loss regardless of the per-frame rendering cost, my goal is to find the best trade-off of spatio-temporal resolution under a limited rendering budget.

I start by discussing existing work on motion-adaptive and fixed-bandwidth rendering in Section 6.2. I then introduce my adaptive local shading and refresh rate rendering method in Section 6.3, followed by instructions on how to implement it in real-time game engines (Section 6.4). The method is then validated in a subjective study in Section 6.5, and finally, I discuss the limitations and future work in Section 6.6.

This chapter is based on the work published in SIGGRAPH Asia 2021 article: *Perceptual Model for Adaptive Local Shading and Refresh Rate*. The results presented in this chapter

include both my work and that of my collaborator Krzysztof Wolski. While I developed the proposed rendering method and its prototype in MATLAB, Krzysztof wrote its parallel implementation for GPU and built the validation experiment in Unity3D. I then collected and analysed the data from the experiment.

6.2 Related Work

6.2.1 Motion adaptive rendering

Offline rendering algorithms can make use of visual models for adaptive sampling. Ray tracers can be developed to directly synthesise an image in the frequency domain [Bolin and Meyer, 1995] which makes it easier to modulate them with models of visual masking [Ferwerda et al., 1997]. Visible difference predictors could be used to determine the stopping conditions of progressive Monte-Carlo rendering [Myszkowski, 1998] or further degrade the quality of moving objects [Myszkowski et al., 1999, Yee et al., 2001, Navarro et al., 2011]. Other high-level visual models such as crowding [Jarabo et al., 2012], attention [Cater et al., 2003] and saliency maps [Longhurst et al., 2006] have also been successfully used in decreasing the computational cost of path tracers. An extensive survey on perceptually accelerated sampling can be found in [Weier et al., 2017]. The above techniques do not account for display aspects and involve computationally expensive quality metrics making them unsuitable for real-time rendering.

Locally adaptive real-time shading is a relatively new area as the traditional rasterization frameworks did not allow for a sub-image level of control until recently. The primary focus of these developments has been on rendering for VR with the goal of either matching the non-uniform pixel distribution of HMDs [Pohl et al., 2015] or exploiting the reduction of visual acuity with eccentricity [Tursun et al., 2019]. The recent introduction of VRS allowed for controlling the shading rates of each 16×16 pixel tiles, enabling a plethora of optimisations. Vaidyanathan et al. [2012] analysed motion and defocus blur in the frequency domain to adaptively control the shading rate, and Yang et al. [2019] developed a similar framework better suited for VRS hardware. Drobot [2020] extends this idea to all platforms through their software-based implementation of VRS. Schmid et al. [2019] describe how ray-tracing of reflections can be done in a fashion similar to VRS by varying the number of ray samples for each 8×8 pixel tile. Most of the above techniques are not perceptually motivated, and their main focus has been on reducing the rendering cost while yielding visually equivalent output. I argue that the goal of obtaining no visual loss is an impractical target for real-time rendering, as producing high-quality results may require a much larger computational budget than available. In contrast to those works, my aim is to deliver the minimum degradation of quality under a constrained rendering budget.

6.2.2 Fixed-bandwidth rendering

Fixed-bandwidth rendering involves the optimal distribution of rendering resources without exceeding a fixed rendering budget given in *pixels/sec*. This can be done by either trading off between resolution and refresh rate (Chapter 3, [Debattista et al., 2018, Denes et al., 2020]) or by fixing the refresh rate and dynamically changing the resolution [Unreal, 2021, Unity, 2021]. All of these works control the global resolution of a frame, and extending

them to locally adapt the resolution is non-trivial.

The problem of maximising quality for a given bandwidth is well studied in the rate-distortion theory but the common solutions involve coding tree units and dynamic programming [Ortega and Ramchandran, 1998] not suitable for real-time rendering. My proposed optimisation is similar to Li et al. [2017], who found the optimal trade-off between delay, power, and rate-distortion in dynamic adaptive streaming applications by formulating it as an integer linear programming problem and providing a greedy approximation. I develop a similar quality optimisation scheme more suited for VRS and propose a parallel implementation on GPU that meets the demands of real-time rendering.

6.3 ALSaRR rendering algorithm

Modern graphics engines require dynamic quality control that adapts rendering parameters (resolution, refresh rate, level of detail, and others) to the available resources (GPU frame-budget, power). This becomes especially important for mobile devices with high-resolution and refresh rate displays, but also in time-sharing cloud rendering, used for streaming of games. Current solutions in popular game engines typically involve dynamic resolution scaling of the entire frame to meet these constraints [Unreal, 2021, Unity, 2021]. This is a sub-optimal approach, as better quality can be obtained by (a) adaptively selecting the best combination of resolution and refresh rate (Section 3.3) and (b) adjusting the resolution locally, using VRS. In this section, I describe my Adaptive Local Shading and Refresh Rate (**ALSaRR**) algorithm that uses my motion quality metric to determine the optimal distribution of shading rate and refresh rate under a given bandwidth constraint.

An overview of ALSaRR can be found in Figure 6.1. It takes as input the allowed bandwidth B in pixels per second and several auxiliary buffers rendered at the resolution of the VRS map: motion vectors, texture IDs, mipmap levels and luminance. The pooled motion vectors and the bandwidth B are used to determine the best frame rate for the current frame. Then, the maps are fed to the CaMoJAB to compute for each VRS block the ratios of quality to bandwidth for all possible shading rates. A greedy knapsack solver then uses the ratios to find the distribution of shading rates that maximise the quality at the given budget. In the following sections, I describe each step in detail and how they can be implemented in a real-time game engine.

6.3.1 Auxiliary buffers

Our quality metric requires the knowledge of texture, the mapping of texels to fragments, luminance, and velocity of the pixels rendered on the screen. For that purpose, we render into a G-buffer: texture IDs, MIP map level, motion vectors, and luminance at the resolution of the VRS map (width/16 $\times$ height/16 pixels). As rendering luminance would require additional shading, which we want to avoid, we use the previously rendered frame (without reprojection) to approximate luminance. The previous frame is first subsampled, then converted to grayscale luma, from luma to linear absolute luminance values, and scaled according to the display peak luminance.

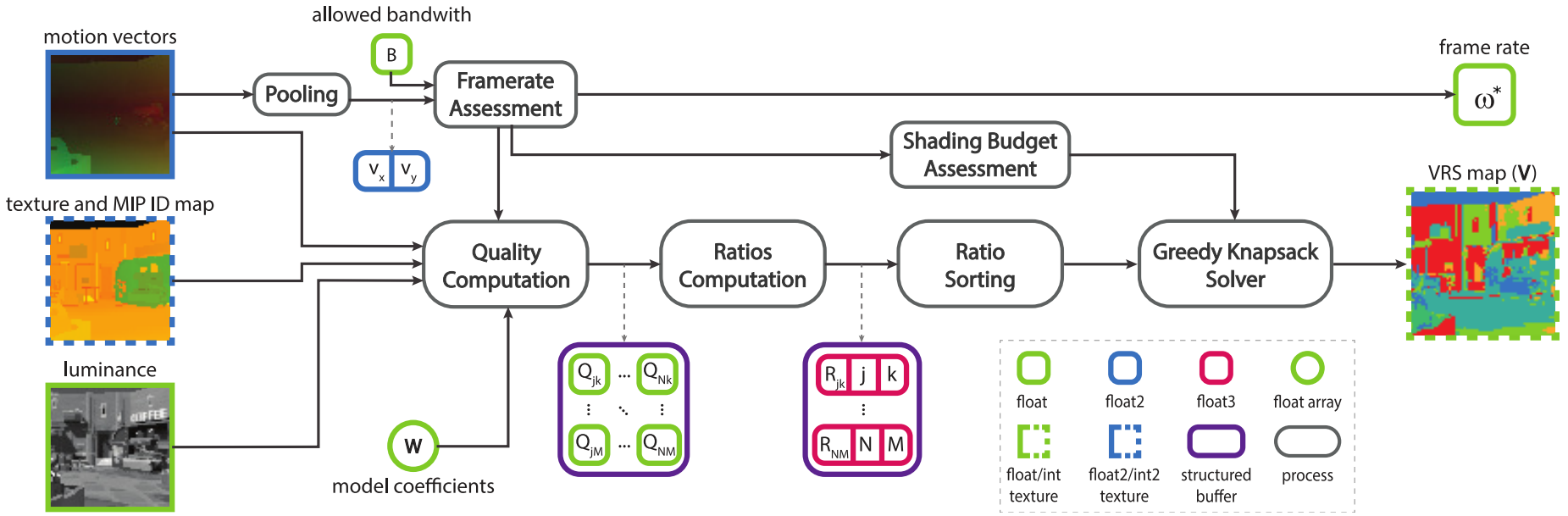


Figure 6.1: The processing diagram for Adaptive Local Shading and Refresh Rate (ALSaRR). The method uses motion vectors, luminance, and precomputed polynomial coefficients to predict the optimal refresh rate and distribution of the shading rates given allowed bandwidth.

6.3.2 Optimal refresh rate

Adaptive-sync displays allow us to choose arbitrary refresh rates, which let us either improve the smoothness of motion or spatial resolution, depending on the velocity of motion. Similar to MARRR, I express this as an optimisation problem:

$$\begin{aligned} &\arg\max_{\omega^*, r_\mathrm{x}, r_\mathrm{y}} q\left(I, v_\mathrm{frame}, \widehat{\omega}, p, \widehat{\rho}, r_\mathrm{x}, r_\mathrm{y}\right) \\ &\text{s.t.} \quad \omega^* R_\mathrm{x} r_\mathrm{x} R_\mathrm{y} r_\mathrm{y} = B \ \wedge r_\mathrm{x} = r_\mathrm{y} \wedge \ \omega^* \in \mathbb{Z}^+ \ \wedge \ \omega^* \leq \omega_\mathrm{max}, \end{aligned} \tag{6.1}$$

where q is the quality function defined in Eq. (5.1), R_x and R_y are the horizontal and vertical display resolution in pixels, ω^* is the optimal refresh rate of the display (optimisation parameter) such that the temporal Nyquist frequency $\widehat{\omega} = \frac{\omega^*}{2}$, ω_max is the maximum refresh rate of the display, and B is the fixed budget in pixels-per-second. As the refresh rate is global for the frame, I take the entire screen-space image (I) as a proxy for the image being tracked and pool the motion buffer to compute the average velocity of the frame (v_frame) in $^{\mathrm{deg}}/_{\mathrm{s}}$. The rest of the parameters are the same as in Eq. (5.2).

Once we have the optimal refresh rate ω^*, we can calculate the maximum allowable shading budget of the given frame as

$$b_\mathrm{frame} = \frac{B}{\omega^*} \quad [pixels]. \tag{6.2}$$

Since solving the optimisation for every frame is too expensive for real-time rendering, I precompute the best combinations using the first frame and store them as a 1D LUT of velocity vs refresh rate (and resolution).

6.3.3 Optimal shading rates

This section describes how to obtain an optimal VRS map under a limited shading budget b_frame.

Problem formulation A screen-space image can be divided into N 16×16 pixel VRS tiles ($\boldsymbol{\mathcal{T}}_j$, $j = 1, \ldots, N$).

We want to assign to each tile $\boldsymbol{\mathcal{T}}_j$ a bandwidth ($\boldsymbol{B}_{jk}$) and a quality ($\boldsymbol{Q}_{jk}$) corresponding to k^{th} shading rate from the set of all available shading rates $\boldsymbol{\mathcal{R}}$ (Figure 6.2).

$$\boldsymbol{\mathcal{R}} = \{4 \times 4, 4 \times 2, \ldots, 1 \times 1\} \tag{6.3}$$

such that for $\boldsymbol{\mathcal{R}}_k = m \times n$. The bandwidth can be calculated as

$$\boldsymbol{B}_{jk} = \frac{256}{m \cdot n} \quad [\text{pixels}]. \tag{6.4}$$

Since our motion quality model assumes unidirectional motion, we calculate the quality in the x and y directions separately using the quality function from Eq. (5.1), so that

$$\begin{aligned} q_x(\boldsymbol{\mathcal{T}}_j, \boldsymbol{D}, \boldsymbol{\mathcal{R}}_k) &= q\left(\boldsymbol{\mathcal{T}}_j, \nu_j^\mathrm{x}, \widehat{\omega}, p, \widehat{\rho}, \frac{1}{m}, \frac{1}{n}\right) \quad [\text{JND}], \\ q_y(\boldsymbol{\mathcal{T}}_j, \boldsymbol{D}, \boldsymbol{\mathcal{R}}_k) &= q\left(\boldsymbol{\mathcal{T}}_j^\top, \nu_j^\mathrm{y}, \widehat{\omega}, p, \widehat{\rho}, \frac{1}{n}, \frac{1}{m}\right) \quad [\text{JND}], \end{aligned} \tag{6.5}$$

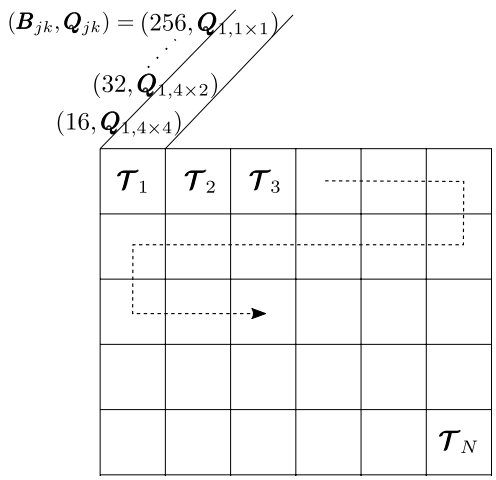


Figure 6.2: Problem formulation for optimal shading rates. Given N VRS tiles each with multiple possible (bandwidth, quality) pairs, we want to select bandwidths that maximises the total quality but does not exceed shading budget (b_{frame}). The problem shares similarities with knapsack problem.

where ν_j^{x} and ν_j^{y} are the minimum velocity in the tile j along x or y (the most conservative assumption). The rest of the parameters are the same as in Eq. (5.2). We define the total quality of the tile as the mean of q_x and q_y

$$\boldsymbol{Q}_{jk} = \left(\frac{q_x(\cdot) + q_y(\cdot)}{2} \right) \quad \text{[JND]}. \tag{6.6}$$

Finally, given the maximum allowable bandwidth per frame b_{frame}, we need to select the shading rate k for each tile $\boldsymbol{\mathcal{T}}_j$ to find

$$\arg\max_{\boldsymbol{B}_{jk}} \sum_{j=1}^{N} \boldsymbol{Q}_{jk} \quad \text{subject to} \quad \sum_{j=1}^{N} \boldsymbol{B}_{jk} < b_{\text{frame}} \tag{6.7}$$

Dynamic programming solution The optimal solution to the above knapsack problem can be found using dynamic programming. For each $j \in 1..N$ and each c between 0 and B I define a *subproblem* as follows: $\mathbb{Q}(j, c)$ is the maximum quality possible when only tiles 1 to j are considered, and the maximum bandwidth is at most c. Our goal is to compute $\mathbb{Q}(N, b_{\text{frame}})$. Note that, unlike the traditional 0-1 knapsack problem, we cannot skip any tile but only vary its bandwidth and quality by selecting its shading rate.

We start with trivial cases and work our way up. The trivial cases are "no tiles" and "total bandwidth 0". In the first case, the maximum quality is 0. The second case is impossible because of our constraint of not skipping any tile. I denote the quality in such cases as $-\infty$.

$$\mathbb{Q}(0, c) = 0 \;\forall\; c \quad \text{and} \quad \mathbb{Q}(j, 0) = -\infty \;\forall\; j \tag{6.8}$$

Consider next the case $j > 0$ and $c > 0$. To find $\mathbb{Q}(j, c)$, we iterate over all possible shading rates. For the k^{th} shading rate of tile j, the maximum achievable quality is $\mathbb{Q}(j-1, c-\boldsymbol{B}_{jk}) + \boldsymbol{Q}_{jk}$, because we obtain a quality of $\boldsymbol{Q}_{jk}$ for the given tile, and must use an optimal solution for the first $j-1$ tiles under the constraint that the total bandwidth

is at most $c - \boldsymbol{B}_{jk}$. Of course, this is only feasible if $c \geq \boldsymbol{B}_{jk}$. I summarise this discussion in the following recurrence for $\mathbb{Q}(j, c)$.

$$\mathbb{Q}(j,c) = \begin{cases} 0, & \text{if } j = 0 \\ -\infty, & \text{if } c = 0 \\ max\{(\boldsymbol{Q}_{jk} + \mathbb{Q}(j-1, c-\boldsymbol{B}_{jk}) \\ \qquad \forall\ k = 1, ..., |\boldsymbol{\mathcal{R}}|\}, & \text{if } j \geq 1 \text{ and } \boldsymbol{B}_{jk} \leq c \\ -\infty & \text{if } j \geq 1 \text{ and } \boldsymbol{B}_{jk} > c \end{cases} \tag{6.9}$$

For ease of implementation of the memoisation cache, I add another shading rate ϕ to our set $\boldsymbol{\mathcal{R}}$ with quality $-\infty$ and bandwidth 0 to ensure we never skip a tile. This algorithm has a *time complexity* of $\Theta(N \cdot b_{frame} \cdot |\boldsymbol{\mathcal{R}}|)$ and *space complexity* of $\Theta(N \cdot b_{frame})$

Though this approach provides the optimal solution, it is too slow for real-time scenarios on standard gaming setups. Hence I provide another approximate but fast greedy solution.

Greedy solution The above optimal dynamic programming algorithm is not suitable for real-time rendering (alone $N = 8100$ and $b_{\text{frame}} = 267\,000$ for a FullHD resolution). Instead, I solve the problem using an approximated greedy solver [Kellerer et al., 2004, Section 2.1], which has the time complexity of $\Theta(N \cdot |\boldsymbol{\mathcal{R}}|)$ and the space complexity of $\Theta(N \cdot |\boldsymbol{\mathcal{R}}|)$. The algorithm finds near-optimal allocation of shading rates based on the quality predictions. Its pseudocode can be found in Algorithm 1.

Let $\boldsymbol{V}$ be a vector of N elements storing the current shading rate for each tile j in $\boldsymbol{V}_j$. We initialise $\boldsymbol{V}$ with the index of the lowest bandwidth shading rate, $\boldsymbol{V}_j = 1\ \forall\ j \in [1, N]$, and subtract the cost of rendering at that rate from the current frame bandwidth budget, b. Next, we calculate all possible ratios of quality to bandwidth for each tile, $\boldsymbol{R}_{jk} = \boldsymbol{Q}_{jk}/\boldsymbol{B}_{jk}$, and sort them in non-increasing order. The first element in the sorted list will give the tile and shading rate that increases the quality at the least cost (bandwidth). Next, we greedily pick (j, k) pairs from the head of the sorted list and update the current frame budget, $b = b - \boldsymbol{B}_{jk} + \boldsymbol{B}_{j\boldsymbol{V}_j}$ and shading rate of j^{th} tile, $\boldsymbol{V}_j = k$. The steps are repeated until the remaining budget becomes 0.

To ensure that the near-optimal solution is sufficient, I compared the results of the greedy algorithm with the accurate solution. Although the theoretical error in the estimated near-optimal solution can be up to 50% of the optimal objective function value, I found in an empirical comparison of the two methods that in practice, the greedy method has the expected error of less than 1% while having orders of magnitude lower time and space complexity. I compared both solutions in a practical video-game scenario by extracting per-frame luminance, motion, and material data from a 6-sec video (360 frames) of gameplay in the *Sci-fi Market* scene[1]. Both the greedy and dynamic programming methods were implemented in MATLAB and were run offline on each frame on a budget of 25% (of FullHD 60 Hz) to calculate their mean performance.

As seen from the results in Table 6.1 and Figure 6.3, the greedy method is a good approximation of the optimal dynamic programming solution and gives real-time performance making it ideal for our application.

[1] `https://assetstore.unity.com/packages/essentials/tutorial-projects/adventure-sample-game-76216`

Algorithm 1: Greedy solution for the modified knapsack problem.

Result: near-optimal shading rate of each tile

$b \longleftarrow b_{\text{frame}}$

for $j = 1$ *to* N ; // For each tile

do

  $\boldsymbol{V}_j \longleftarrow 1$; // Shading rate 4×4

  $b \longleftarrow b$ - $\boldsymbol{B}_{j1}$; // Update remaining bandwidth

  for $k = 2$ *to* $|\boldsymbol{\mathcal{R}}|$; // For each shading rate

  do

    $\boldsymbol{R}_{jk} \longleftarrow \frac{\boldsymbol{Q}_{jk}}{\boldsymbol{B}_{jk}}$; // Calculate ratios

  end

end

sort($\boldsymbol{R}$) ; // From the highest to the lowest

while $b > 0$ **do**

  (j,k) $\longleftarrow \boldsymbol{R}$.pop() ; // Pick the highest ratio

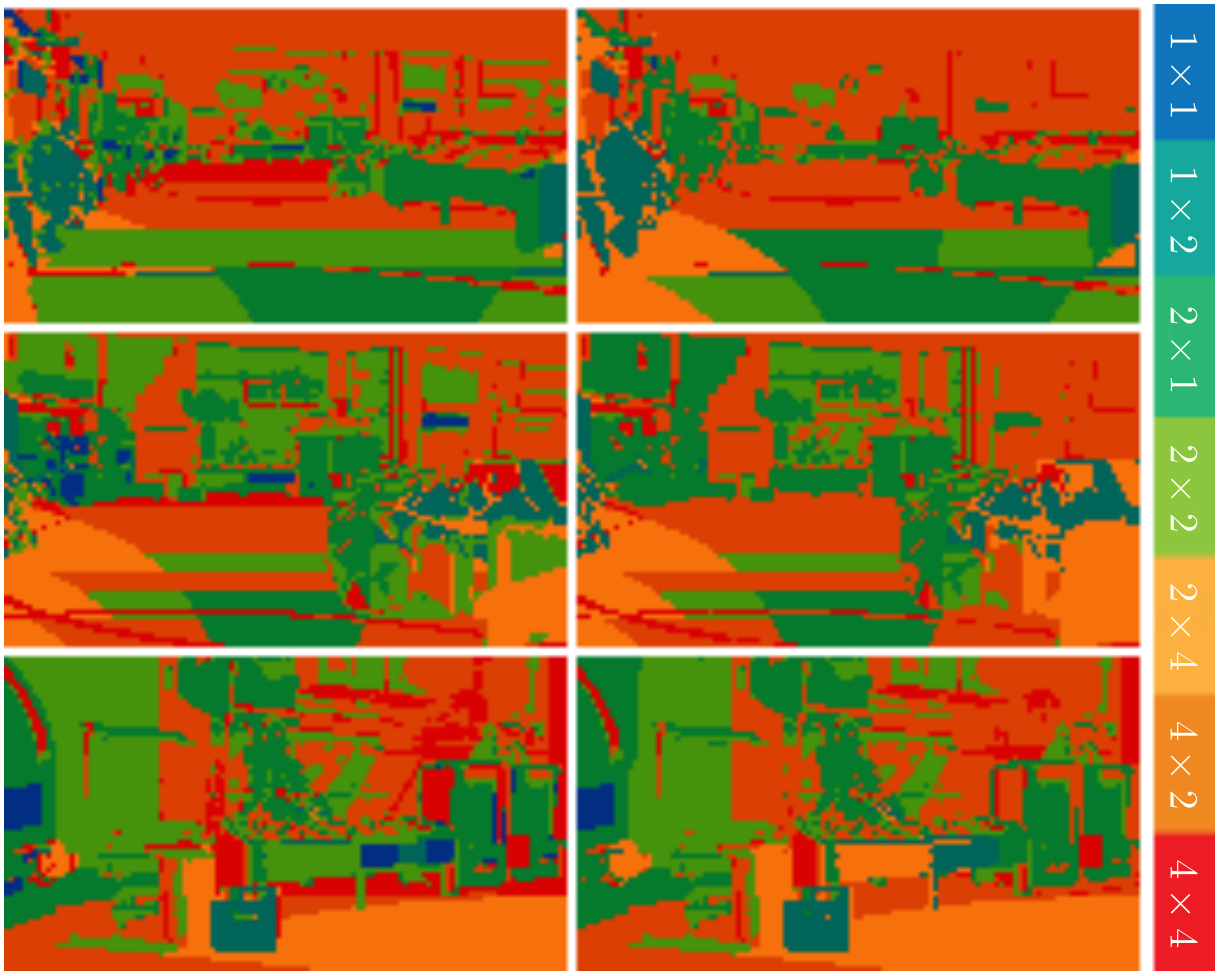
  $b \longleftarrow b$ - $\boldsymbol{B}_{jk} + \boldsymbol{B}_{j\boldsymbol{V}_j}$; // Update the bandwidth

  $\boldsymbol{V}_j \longleftarrow$ k ; // Update the shading rate

end

Table 6.1: Empirical comparison of Knapsack solutions. Error is defined as 100 - percentage of the optimal solution (Solution 1).

	Solution 1 (Dynamic Programming)	Solution 2 (Greedy Approach)
Mean Error (per frame)	0 %	0.95% (±0.26%)
Mean Time (per frame)	19000ms (±900ms)	13ms (±0.8ms)



Greedy method Dynamic programming (optimal)

Figure 6.3: Sample VRS maps produced by greedy and dynamic programming method. Greedy method appears to be a good approximation of the optimal solution.

6.4 Real-time implementation

To reduce the performance overhead of our method, the quality predictions of CaMoJAB (Eq. (5.1)) are precomputed for every texture, mipmap level, and shading rate and stored as polynomials of the form:

$$\begin{aligned} q_{i,j,k}(\nu, f, L) = (W_0 + W_1\nu + W_2 f + W_3\nu^2 + W_4\nu f + W_5 f^2) \\ \cdot (W_6 L^3 + W_7 L^2 + W_8 L + W_9), \end{aligned} \tag{6.10}$$

where i is the texture index, j is the MIP map level, k is the shading rate, ν is velocity, f is the refresh rate of the current frame ($2\,\widehat{\omega}$), L is the logarithmic luminance of the tile, and $W_{0,\ldots,9}$ are the coefficients of the polynomial, stored separately for each (i, j, k) triplet (Figure 6.4).

Our real-time implementation follows Algorithm 1 and computes the ratios of quality to bandwidth for each tile and each possible shading rate. Those ratios are sorted in a non-increasing order (the best quality-to-shading rate first) using a bitonic sort, as this algorithm can utilise the parallelism of the GPU.

The velocity of eye motion without eye-tracking In a typical animation, multiple objects can move on the screen in different directions with different velocities, and the eye could follow any of them. For that reason, the MARRR method required an eye tracker to determine the actual velocity of the eye motion relative to the screen. However, because we control the shading rate locally, we can make a conservative assumption that the eyes could follow each of the 16×16 tiles. This lets us use a different velocity of eye motion for each tile instead of a global velocity per frame, as done in MARRR. Because the distortions are the most visible when the object is followed, we get the worst-case estimate of the quality degradation due to the reduced shading rate. This gives us an important practical advantage over MARRR as eye tracking is still rare in most real-time graphics applications..

Implementation details To test our method on realistic scenes of sufficient complexity, we implemented a native plugin for Unity. The plugin modifies the default forward rendering pipeline by allowing for a custom VRS map. VRS is enabled only for the fragment shader stage and is disabled for the post-processing pass, as that would introduce reduced shading at the pixel level and shade fragments across the edges.

Performance We measured the performance of ALSaRR on a GeForce RTX 2080Ti graphics card using built-in Unity profiler. Our implementation utilises parallelism of GPU and requires, on average, 1.3 ms to process the frame of resolution 1920 × 1080 pixels. It also requires approximately additional 100 MBs of VRAM to store intermediate data. The processing time is independent of the rendered content and does not include preprocessing (fitting of the polynomials, Eq. (6.10)). We did not attempt to implement the method on a mobile GPU, but we expect the efficient implementation on mobile architectures may require further performance optimisations, such as computing VRS maps at a lower resolution and upsampling them.

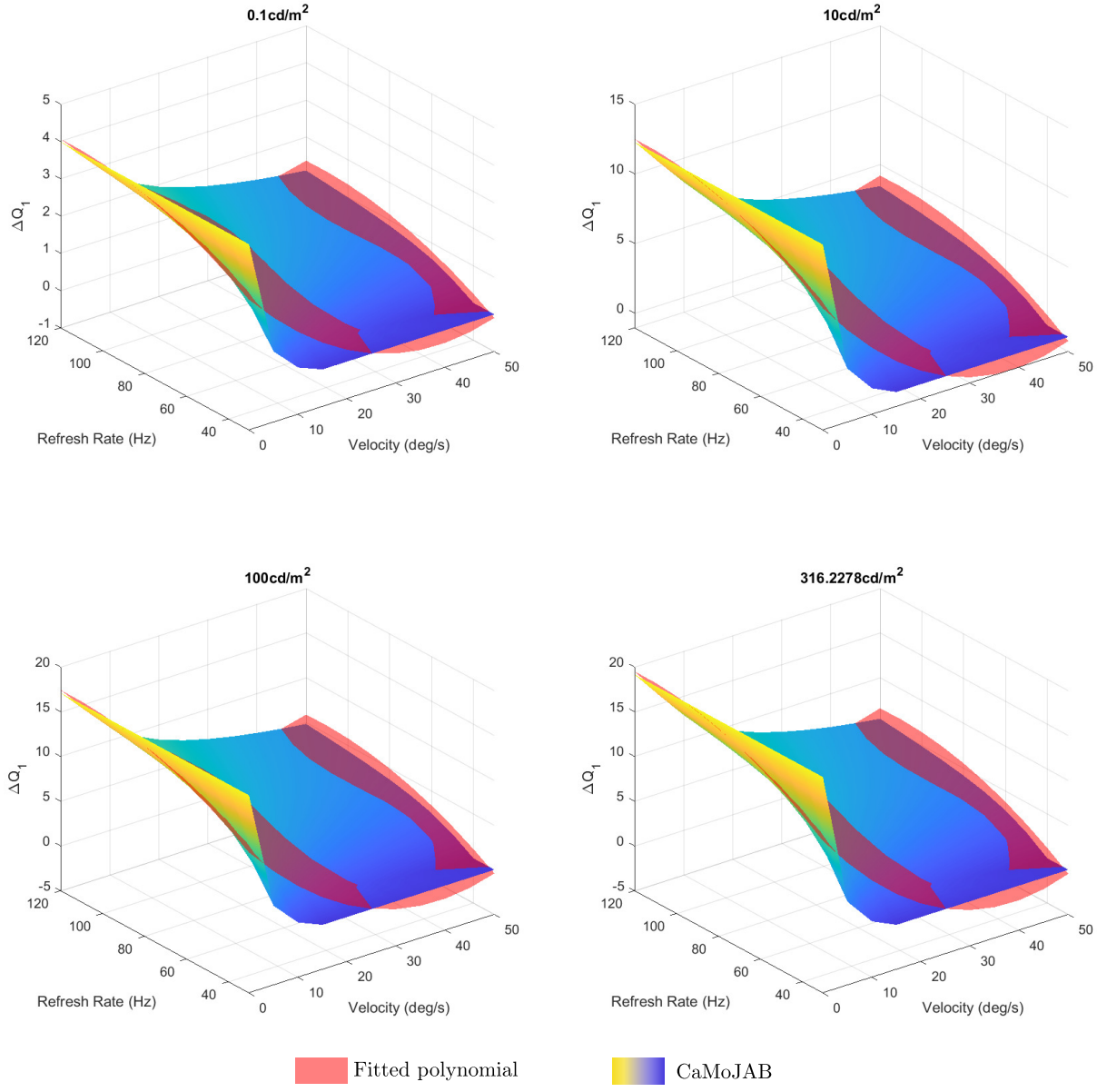


Figure 6.4: A polynomial is fitted for every texture, mipmap level, and shading rate on CaMoJAB predictions. The figure provides a preview of how the chosen polynomial well approximates quality variation of 1×1 shading rate across different velocities, refresh rate, and luminance.

6.5 Psychophysical validation

In this section, I report on a pairwise-comparison preference experiment, which compared our ALSaRR method against constant-refresh-rate-resolution and MARRR. The experiment simulated video game scenarios, all implemented in Unity 3D and rendered with one of the compared methods.

Stimuli For the experiment, we used three 3D scenes acquired from the Unity Asset store (Figure 6.5). We prepared 4 different camera motion scenarios for every scene. One of them emulated a first- or third-person game, giving participants the ability to control the camera with the mouse and keyboard. The remaining scenarios contained predefined motion paths of different camera velocities. Each scene was rendered using one of the three compared methods: ALSaRR, constant-refresh-rate-resolution, and MARRR (Chapter 3). We used several combinations of shading and refresh rates for the constant-refresh-rate-resolution method, each with a fixed budget of 6.25% or 12.5% pixels of the full bandwidth (1920×1080 @ 120 Hz). We used a lower resolution than the native resolution of the display (2560×1440) because we noted in the pilot experiments that participants have difficulty distinguishing between the conditions at higher resolutions (on a 27" monitor seen from 75 cm). MARRR method originally required an eye tracker to determine the velocity of eye motion (relative to the screen). However, for closer comparison, we made the same simplifying assumption as we did for the selection of refresh rate in our method and estimated the eye motion velocity as the average velocity within the entire frame (pooled from the motion buffer).

The rendered scenes were shown on the same monitor used in Experiment 1 (Section 4.3).

Procedure The experiment consisted of pairwise comparisons between the three compared rendering methods. In each trial, one of the conditions was our method, and the other condition was either a constant-resolution-refresh-rate or MARRR. The order of the scenes and the compared methods was randomised. The camera motion scenario was randomly selected for each trial as a full factorial design was infeasible within a reasonable time budget.

The participants could press the *space* bar to toggle between two compared rendering methods. They had to choose the method with overall better visual quality by pressing the *Enter* key while the preferred rendering mode was active. The question shown to the observers was: *"Which of the two methods has higher visual quality (i.e. smooth motion and sharp image)?"*. Each observer performed 108 comparisons (3 scenes × 6 budgets/rendering methods × 6 repetitions). The experiment was split into 2 sessions lasting 30 minutes to prevent observer fatigue.

Participants and procedure 12 participants (aged 24-41) with normal or corrected-to-normal vision participated in the experiment. They were compensated for their time.

[2] `https://assetstore.unity.com/packages/essentials/tutorial-projects/adventure-sample-game-76216`

[3] `https://learn.unity.com/project/john-lemon-s-haunted-jaunt-3d-beginner`

[4] `https://assetstore.unity.com/packages/3d/environments/urban/sally-s-country-home-33123`

Figure 6.5: Preview of the scenes selected for validation experiment. The *SciFi Market* scene[2] (top) and *John Lemon's Haunted Jaunt* scene[3] (bottom) have been released as learning materials by Unity. The *Sally's Country House* scene[4] (middle) created by *Gabro Media* has been acquired from Unity Asset Store.

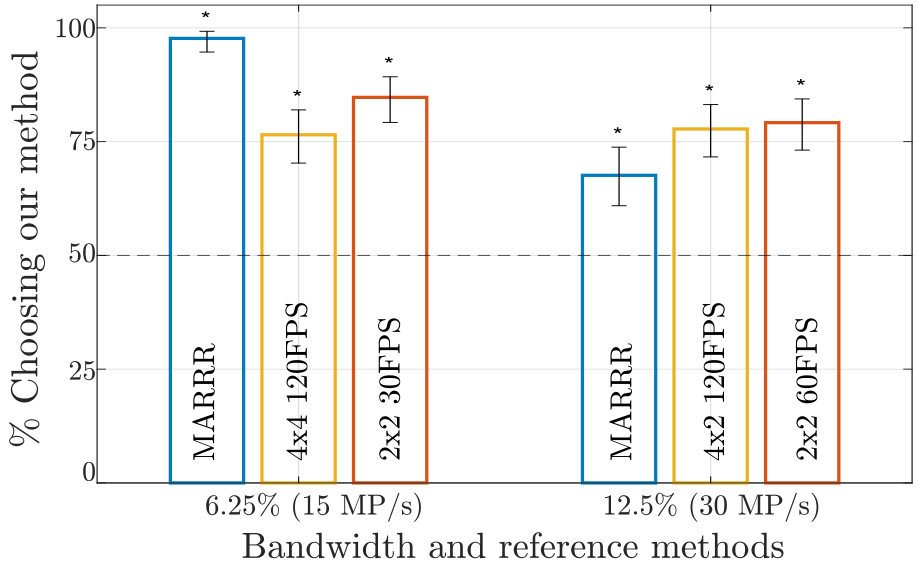
Results The results of the validation experiment, visualised in Figure 6.6, show a clear preference for our ALSaRR method compared to fixed resolution rendering and MARRR. The difference is significant across all tested conditions (binomial test with the null hypothesis of random guess). As expected, the improvement due to adaptive rendering decreases as the bandwidth increases because the differences between the conditions become less noticeable. MARRR performed better than fixed-frame-rate at a higher budget of 12.5%, but worse at 6.25%. It should be noted, however, that our fixed-rate conditions used reduced shading rate instead of bilinear upsampling used in MARRR. This may indicate that reducing the shading rate may produce better visual results than upsampling from a lower resolution when the rendering budget is low. The lack of eye-tracking could also negatively affect the performance of MARRR.

The per-scene results, shown in Figure 6.6(b), suggest the performance gain of our method was the smallest for the "Scifi market" scene. After closer inspection, we noted that the scene contained a large number of hard shadows, which resulted in aliased edges for our method. This is because our current implementation of ALSaRR relies on the information from textures rather than rendered fragments and, therefore, ignores local shading (with the exception of mean luminance). This problem is difficult to mitigate in real-time rendering as the decision on shading rate must be made before local shading is computed.

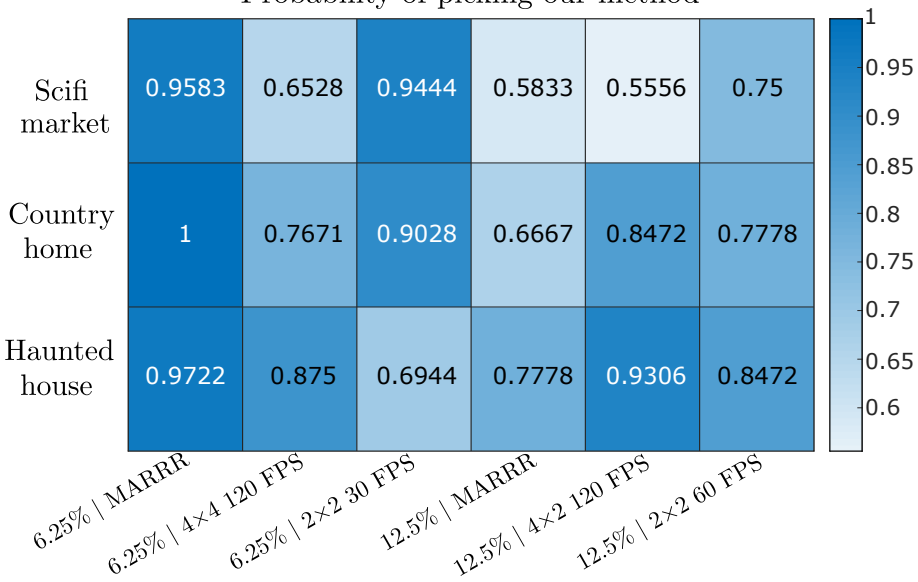
6.6 Limitations

One of the important advantages of ALSaRR over foveated rendering [Tursun et al., 2019] and MARRR is that it does not require eye tracking. However, to eliminate the dependency on eye tracking, we had to make a conservative assumption that the eye can follow movement in every VRS tile. If the gaze location was known to us, we could potentially further improve the predictions and therefore allocation of the rendering budget. ALSaRR could also be combined with foveated rendering for further gains in quality.

Practical considerations for ALSaRR Our ALSaRR method can control real-time rendering with an acceptable overhead because we can precompute the quality predictions for all textures, mipmap levels, and shading rates. The main shortcoming of this approach is that our CaMoJAB quality metric is unaware of the shading in the final image, for example, when a tile contains a shadow boundary. A few such failure cases are demonstrated in Figure 6.7, where ALSaRR assigns non-optimal shading rates due to the lack of correct frequency information. The examples shown in the figure include (1) reflections, (2) procedural particle systems and (3) transparent objects. Some game engines may not support rendering motion vectors for transparent materials, further exacerbating the problem. It should be noted that ALSaRR uses diffuse textures as a proxy for object frequency information. Though this served as a good approximation for our tested scenes, it may not hold true for more complex materials. These issues can be addressed to an extent by precomputing the worst-case scenario shading and then using that for our precomputed quality functions. We could also use the approach from NAS [Yang et al., 2019] and directly analyse the frequency information by reprojecting the previous frame. This approach, however, relies on the assumption that the previous frame has been rendered with sufficient quality, and it also adds a substantial overhead of reading and processing full-resolution frames.



(a) Results of Experiment 2, aggregated across all scenes



(b) Results of Experiment 2 for each scene

Figure 6.6: Results of the validation experiment showing percentage of participants picking our method over MARRR and constant-refresh-rate-resolution rendering. Error bars in (a) denote 95% confidence interval. The numbers in (b) are the probabilities of selecting ALSaRR as better than the alternative method.

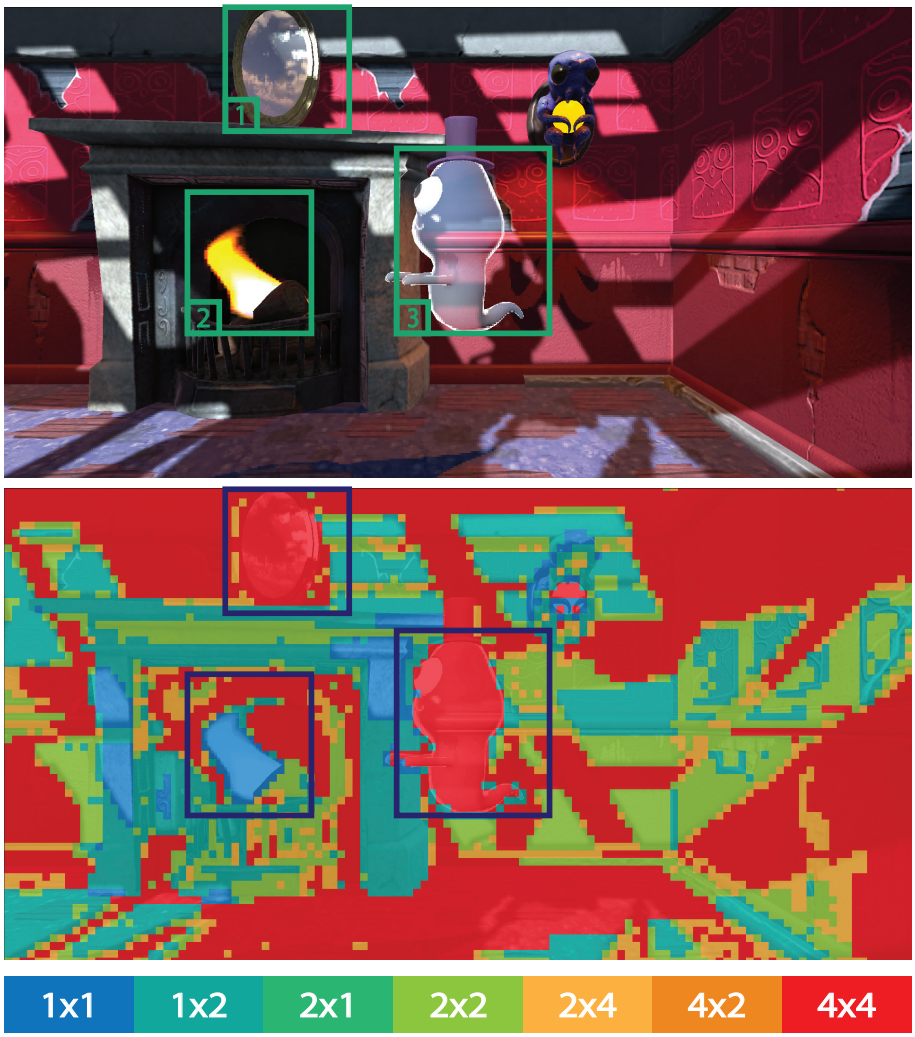


Figure 6.7: Examples of fail cases: because ALSaRR relies on textures rather than shaded pixels, it assigns low shading rates to mirror reflections (1) and transparent ghost (2); high shading rate to fire particles (3).

Our current implementation of ALSaRR operates on the rendering budget expressed in pixels per second, while most applications are limited instead by the power of GPU cycles. This is not a fundamental limitation of our method, as it can work with the budget expressed in any units as long as there exists a model that can predict such units from the VRS rates.

6.7 Summary

There is a shift in real-time graphics from rendering with a fixed resolution and refresh rate to a more adaptive approach, in which we control spatio-temporal resolution in order to maximise the quality under a given rendering budget. VRS and adaptive refresh-rate monitors provide an opportunity for such fine control, which becomes particularly important as the increasing resolution and refresh rate of displays places a high demand on the GPU, especially on mobile devices. In this chapter, I described the ALSaRR method that takes advantage of the limitations of the visual system to reallocate the rendering budget to the most vital part of the spatio-temporal domain. ALSaRR easily integrates into common rasterization pipelines with minimal overhead using GPU parallelism. A subjective study with a wide variety of content and motion validated that ALSaRR provides better visual quality than existing adaptive and traditional constant resolution methods for the same bandwidth.

Chapter 7

HDRMFS: High-Dynamic-Range Multi-Focal Stereo Display

7.1 Introduction

The end goal of computer graphics has always been to render images that are spatially and temporally indistinguishable from real-world 3D scenes. However, the discrete nature of capture, display, and graphics systems is bound to introduce certain image artefacts not present in the real world. As discussed in Chapter 4, several studies have been conducted to discover the bounds for the display and graphics parameters (such as refresh rate and resolution), where these artefacts become invisible. Unfortunately, there are three major problems in all these studies:

1. The range of the parameters studied is often limited and quite lacking when compared to the real world. For e.g., the display used in our motion quality experiment in Section 4.3 has a dynamic range of 3 orders of magnitude, while the real world can have up to 14 orders of magnitude.

2. Only a small subset of realism cues are presented at a given time even though the final image we perceive is a combination of all cues. The results obtained in such cases might vary significantly when the same factors are studied together with other cues. For e.g. motion and banding artefacts not visible on SDR displays can become visible on HDR displays [Chapiro et al., 2019, Kim et al., 2020] or perception of depth from binocular disparity may vary when motion parallax is introduced [Hartle and Wilcox, 2021]. The integrating nature of HVS also raises the possibility that small differences to the real world in a single cue may be masked when other cues are provided at sufficient quality.

3. The reference image/video provided for the experimental task is presented on the same display-graphics system and so is also prone to similar artefacts. Alternatively, no reference is provided, and participants are asked to use their mental model of the scene, which introduces individual variations between their quality scales.

The main reason behind the above problems is the limited capabilities of common consumer displays. These displays are designed to be affordable or serve specific applications, so they optimise their parameters accordingly within cost constraints (For e.g. high spatial resolution of TVs but high refresh rate for VR headsets). Their design constraints make it

challenging to recreate multiple cues at a sufficient fidelity required for psychophysical experiments. Another reason behind these problems is that most experimental setups do not provide any mechanism to compare displayed image side by side to a real 3D scene which could be argued to be the best benchmark for perceptual realism: when displayed content is perceptually indistinguishable from the real-world 3D scene.

In this chapter, I show how these issues can be overcome by building a novel display apparatus that combines multiple realism cues at a high fidelity: spatial resolution (at least 85 ppd at a viewing distance of 490 mm, refresh rate of 60 Hz, correct binocular disparity, peak luminance of up to 3000 $\mathrm{cd/m^2}$ and dark level below 0.01 $\mathrm{cd/m^2}$, a range of focal depth for each eye (1.18 to 2.04 diopters), and a colour gamut of BT.709. It uses special optics similar to see-through AR displays which allows it to compare the presented image with a small scene inside a box of size 370 mm × 260 mm × 360 mm (width × height × depth). Building this display required a complex combination of multiple display technologies, optics, a 3D scene acquisitions pipeline and a practical rendering system that can deliver all these cues at high fidelity in real-time.

I begin this chapter with a short discussion on candidate display designs and rendering methods that could potentially be used to achieve our goal (Section 7.2). I then briefly describe the display and scene capture pipelines (Section 7.3) which were built by my collaborators Fangcheng Zhong, Özgür Yöntem, Param Hanji, and Rafal Mantiuk. Then, I dive into my contribution: the pose estimation (Section 7.4.1) and rendering pipeline (Section 7.5) that offers three image-based rendering methods with widely different scene representations. I then compare these methods in Section 7.6, both in terms of computational performance and objective image quality, to demonstrate their strengths and weaknesses for different scenarios. Finally, in Section 7.7, I briefly describe various (previously impossible) psychophysical experiments that have been conducted on this display.

This chapter is based on the work published in SIGGRAPH Asia 2021 article: *Reproducing Reality with a High-Dynamic-Range Multi-Focal Stereo Display*

7.2 Related Work

7.2.1 Computational 3D Displays

To display an image that is perceptually similar to the real scene, it is necessary to reproduce several realism cues on a display: high dynamic range, spatial and temporal resolution, binocular disparity, accommodation, motion parallax, field-of-view, and colour. Computational displays are an emerging class of displays that attempt to reproduce a subset of these through a combination of optics, display technology, and computation. Examples include high dynamic range displays which achieve a large range of contrast and luminance by replacing the backlights in conventional LCDs with programmable LEDs or a projector [Seetzen et al., 2004b] , or light field displays [Wetzstein et al., 2012, Lanman et al., 2011] which generalizes parallax barriers to recreate the light field of the real scene (though often at the cost of spatial resolution). Wearable displays with stereo vision are also growing in popularity. They provide a strong depth cue by showing two different images to both eyes, either by using separate displays for the two eyes or using shutter glasses and polarisation filters [Hainich and Bimber, 2016]. Such displays often suffer from vergence-accommodation conflict as they use a single focal plane and thus cannot

reproduce focal (or accommodation) cues. This can be resolved by adding multiple focal planes [Akeley et al., 2004b] or using varifocal displays [Oculus, 2020].

For our display, we aim to maximise the visual quality of displayed images by delivering the following dimensions at a high fidelity: physical luminance, dynamic range, colour gamut, and binocular and focal depth cues. To enable the study of the interplay of these factors, we want to deliver these capabilities all together with a sufficient quality instead of maximising a single one. To achieve this, we combine ideas from several display architectures (Section 7.3) and provide an end-to-end pipeline consisting of 3D scene acquisition, display, and rendering. Our display pipeline is shown to achieve such a high fidelity rendering in psychophysical experiments that the presented images are often confused with physical 3D objects.

7.2.2 Photorealistic rendering

Over the last few decades, rendering has made large strides in achieving photorealistic results. It has been successfully adopted in several areas, including entertainment [Christensen and Jarosz, 2016], simulation [NIVIDA, 2021], and tourism [Snavely et al., 2006]. This success can be primarily attributed to three major rendering paradigms: path tracing (IBR), rasterization with physically-based shading and image-based rendering. Path tracing works by tracing millions of light paths, which gives accurate results but is often very slow to compute (minutes to hours per frame). Rasterization with physically-based shading is quite fast and is widely adopted in modern video games, but it requires a lot of artist hours modelling scene materials and ad-hoc approximation of physical effects such as global illumination, which may not generalise well to different scenes. In contrast, IBR combines 3D reconstruction techniques from computer vision with computer graphics rendering techniques to generate novel views from a set of captured scene images.

Since its first appearance in the early 90s [Chen and Williams, 1993, McMillan and Bishop, 1995], IBR remains the dominant technique for reproducing novel views of real scenes. Its high-quality results, ease of integration, good performance and generalizability across complex scenes make it an ideal candidate for 3D displays aiming for perceptual realism. Many IBR methods have been proposed in the last three decades and can be roughly arranged on a scale of how much 3D information they require. They may assume no 3D information and only work with pure sampled ray-based representations such as light fields [Levoy and Hanrahan, 1996, Isaksen et al., 2000]. Or have a discrete scene representation such as layered depth images or sprites with depth [Shade et al., 1998, Szeliski and Golland, 1998], which sidesteps rasterization and uses efficient image warping. Other approaches require more explicit geometry of the 3D scene, such as Lumigraph [Gortler et al., 1996], surface light fields [Wood et al., 2000] and view-dependent texture mapping [Debevec et al., 1996] which maps the images to a single global 3D mesh of the scene. The parts of images to be mapped to different parts of the mesh are chosen based on the distance between images and the target viewpoint. The scene geometry can also be varied for each view which may help in better capturing local non-rigid effects such as specular motion [Overbeck et al., 2018, Szeliski, 2021].

More recently, the introduction of deep neural networks into both 3D reconstruction and image-based rendering pipelines has achieved previously unachievable results such as reconstruction of fine geometry and complex reflections. DNNs can be used to learn parts of traditional IBR pipelines such as learning per-pixel blending weights [Hedman et al.,

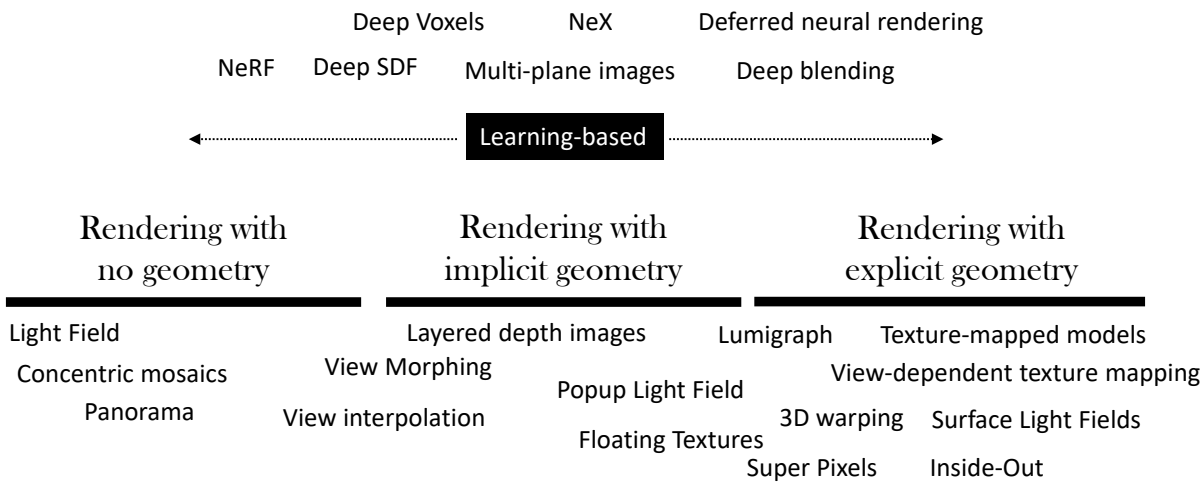


Figure 7.1: A spectrum of image-based rendering methods based on how much geometric information is assumed by the method. Methods/representations on the left only work with images while representations on the right require detailed geometry. Deep learning based methods span across the entire spectrum ranging from learning weights of traditional representations or a completely novel representation.

2018, Flynn et al., 2016], or learn an intermediate scene representation like multi-plane images [Zhou et al., 2018, Li et al., 2020, Wizadwongsa et al., 2021] or voxel grids [Sitzmann et al., 2019a, Lombardi et al., 2019] which can be used to generate novel views using traditional rendering techniques, or learn a 3D scene parameterised as a continuous function using multi-layered perceptrons (commonly know as implicit neural representations or coordinate-based representations)[Sitzmann et al., 2019b, Mildenhall et al., 2020, Niemeyer et al., 2020]. A comprehensive survey of all such neural rendering methods can be found in [Tewari et al., 2020].

In this work, I will implement and analyse 3 different IBR methods on our display: Dynamically reparameterised lightfield [Isaksen et al., 2000], Lumigraph implemented as view-dependent texture mapping on global scene geometry [Debevec et al., 1996, Gortler et al., 1996], and NeX, a neural multi-plane images method [Wizadwongsa et al., 2021]. The methods were chosen to be representative of broader categories on the IBR spectrum (Figure 7.1) and thus cover a wide range of usage scenarios. We also required our chosen methods to support HDR images and run in real-time, which excluded NeRF and its variants from our available options.

7.3 Display apparatus

A photograph of our display apparatus can be found in Figure 7.2. The apparatus has three main components:

- a Wheatstone stereoscope with four high-dynamic-range displays (two for each eye) at two focal depths
- a Real Scene Box (RSB) in front of the observer containing a small-scale lightable physical scene that is seen through a pair of beam-splitters

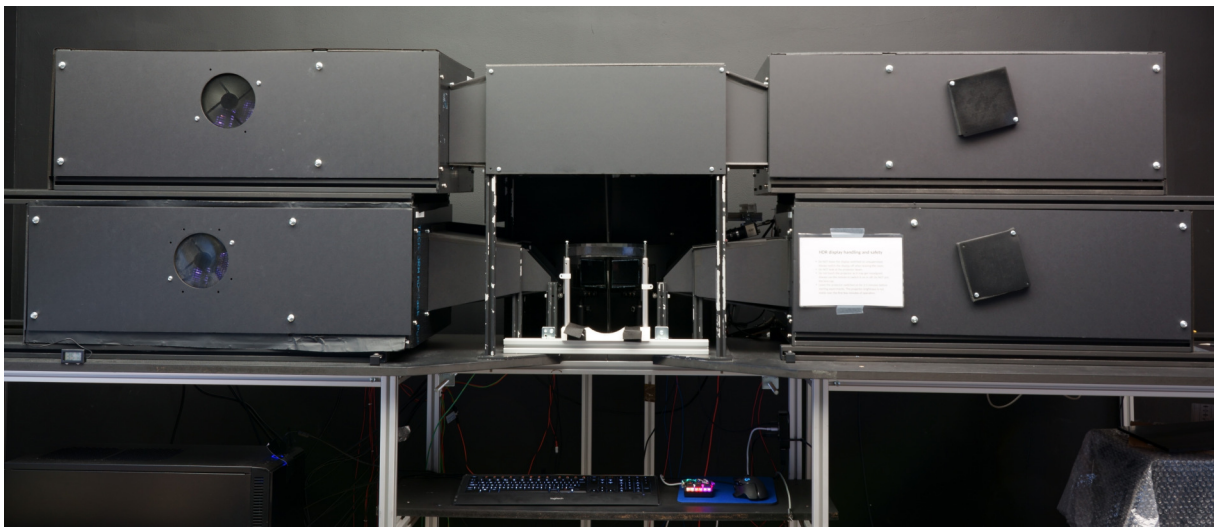


Figure 7.2: The front view of the display.

- a 1D horizontal camera gantry capable of capturing dense light fields of the RSB

The RSB can be rotated to face the camera rig to capture its light field. The light field is rendered on the display and is spatially superimposed on the real scene when the RSB is facing the observer. The lights in RSB are software controlled and can be turned on and off to instantly switch between the real or the displayed scene. Figure 7.3 provides the schema of the display apparatus.

I describe each component in more detail below.

HDR displays Our HDR display is a projector-based dual modulation display following a design similar to [Seetzen et al., 2004a]. All four HDR displays are built using an IPS LCD panel from iPad3 (9.7" 2048×1536) with its backlight replaced with a DLP projector (Acer P1276) with its colour wheel removed. The display can reach a peak luminance of at least 3000 $\mathrm{cd/m^2}$ and a black level of less than 0.01 $\mathrm{cd/m^2}$. The LCD and projector were geometrically aligned by capturing images of a point grid displayed separately on each display and then using homography to align the two grids. The same grid of points was also used to estimate the point spread function of the projector. The HDR display was colour calibrated using a spectroradiometer (Specbos 1211). The effective bit-depth of both displays was elevated to 10 bits by spatio-temporal dithering and bit-stealing (only DLP).

Focal planes and optics To deliver correct focal cues to our eyes, we chose a multi-focal display design [Akeley et al., 2004a] with two images shown at the same distance as near and far planes of the RSB (490 mm (2.04 D) and 850 mm (1.18 D), respectively). The two HDR displays for each eye are stacked vertically along with an arrangement of beam-splitters and first surface mirrors that allow for a horizontal light path with a virtual image forming in the RSB (Figure 7.3). We chose this display design over refractive [MacKenzie et al., 2010] or varifocal [Chang et al., 2018] optics to minimise aberrations and retain the dynamic range. However, multi-focal displays are very sensitive to misalignment in head position. To mitigate this, we used a chinrest to reduce head movement and a head tracker based on Purkinje image tracking [Guestrin and Eizenman, 2006]. Note that

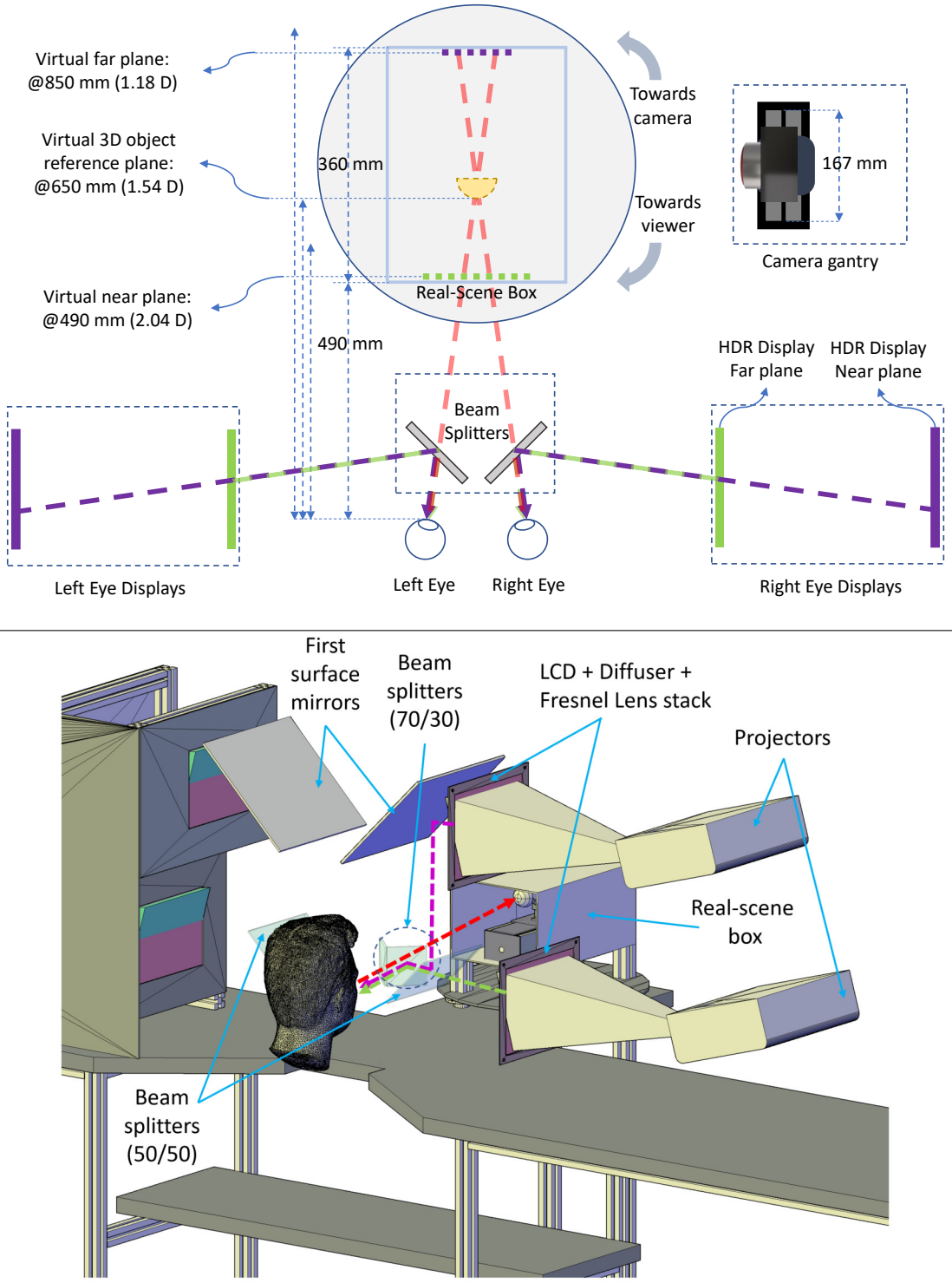


Figure 7.3: Schematic of the high-dynamic-range multi-focal stereo display apparatus. The beam splitters creates two virtual images per eye (green and blue dashed lines inside the real-scene box) overlapping with the physical scene in the real-scene box. The red, green, and purple lines depict the light paths from the displays and the physical scene to the eyes. The box can be rotated to face the camera gantry to facilitate the light field capture of the physical scene.

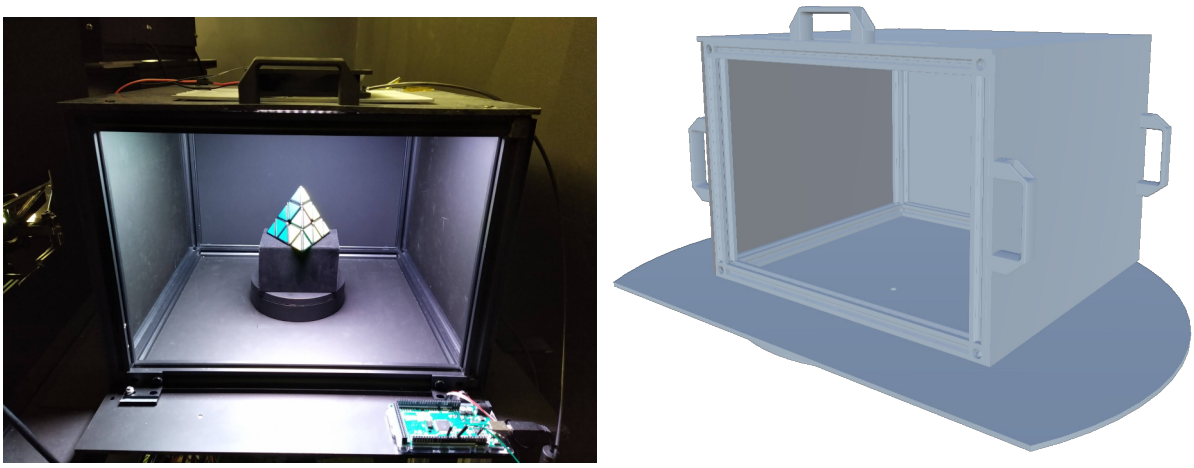


Figure 7.4: A photo and schematic of the real-scene box.

the HDR displays are placed on aluminium strut profiles and can be moved to adjust the distance between the two virtual planes.

Real scene box The real-scene box has inner dimensions of 370 mm × 260 mm × 360 mm (width × height × depth), exceeding the field-of-view of LCDs such that the side walls of the box are never visible to the observer. The box is made from black acrylic with the back wall coated with Vantablack. Having such high light absorption materials surrounding the real scene ensures that only the object of interest is visible to the observer. This is a necessary condition for experiments testing perceptual realism, where we want observers to only use cues from object appearance to distinguish between real and virtual scenes (Section 7.7). The ceiling of the box was fitted with a 16 × 16 RGB LED array and colour calibrated using a spectroradiometer (Specbos 1211) and fitted with a gain-gamma-offset model. The back wall was also fitted with four point-shaped calibration LEDs at known locations to help us register the lightfield camera's and observer's eyes' pose and locations (more on this in Section 7.4.1). The box sits on a rotatable platform which allows it to either face the observer for viewing or face the camera for lightfield capture. A CAD drawing and a photo of the RSB are shown in Figure 7.4.

A comprehensive discussion of the display apparatus can be found in our paper [Zhong et al., 2021].

7.4 HDR Lightfield acquisition

Before we start rendering on our display, we first need to capture a real scene at a sufficient quality in a format that can be used to render the scene from the observer's viewpoint. We capture a horizontal light field of the RSB consisting of 7360 × 4912 pixels resolution HDR exposure stack using a Sony α7R3 mirror-less camera with a Sony SEL prime lens (focal length 55 mm) mounted on a motorised camera gantry at a distance similar to viewpoint (550 mm from the far plane of real-scene box). The gantry has a baseline of 167 mm with an accuracy of 5 µm. Each exposure stack is demosaiced [Menon et al., 2006], and merged into a 16-bit HDR image [Hanji et al., 2020]. The merged HDR images are then colour calibrated to the Rec.709 space as used by our display.

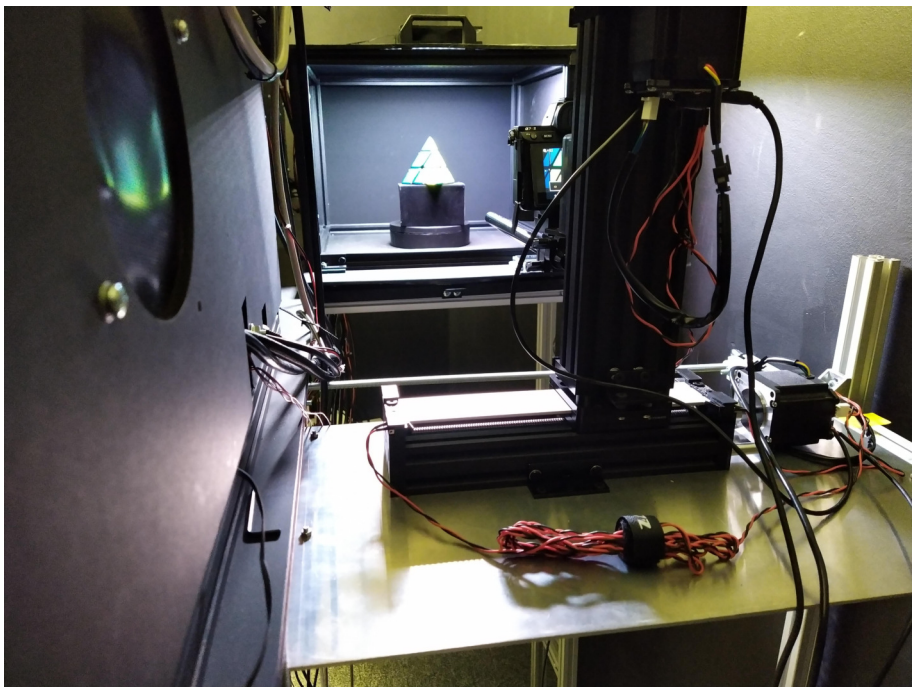


Figure 7.5: A gantry mounted with a DSLR camera is used to capture lightfield of the real scene

7.4.1 Camera pose estimation

To render the light field of a scene or learn some representation of it, it is necessary to determine certain characteristics of the camera for captured image. I assume a pinhole camera model and describe how I calculated the intrinsic and extrinsic characteristics of the camera.

Intrinsic camera matrix estimation The intrinsic camera matrix describes the geometric properties of a camera, i.e., its focal length, principal point offset and axis skew. It maps the 3D camera coordinate to 2D homogeneous image coordinates. To estimate this matrix, we can first print a checkerboard pattern of known dimensions and take photos of this pattern from different views (Figure 7.6). Then we can detect the checkerboard points in the captured images, solve for camera intrinsic and extrinsic matrices in closed form [Zhang, 2000], and adjust the matrices to account for non-linear lens distortion using numerical optimisation [Heikkila and Silvén, 1997]. I used the implementation available in the MATLAB CV toolbox for all three steps. Since we will use the same camera to capture all the subsequent light fields, the 3×3 intrinsic matrix of the data camera (K_D) will remain constant.

Extrinsic camera matrix estimation The extrinsic camera matrix describes the camera's pose, i.e. its location and what direction it is pointing. It has two components (rotation and translation) and can be represented by a 3x4 matrix. An extrinsic matrix can be easily calculated in closed form if we know the intrinsic matrix and have at least 4 points in world space and their corresponding pixel coordinates. For this, at each light field camera position, we capture an additional image of 4 LEDs placed on the far wall of the RSB (Figure 7.7). I again used the MATLAB CV toolbox to calculate the extrinsic

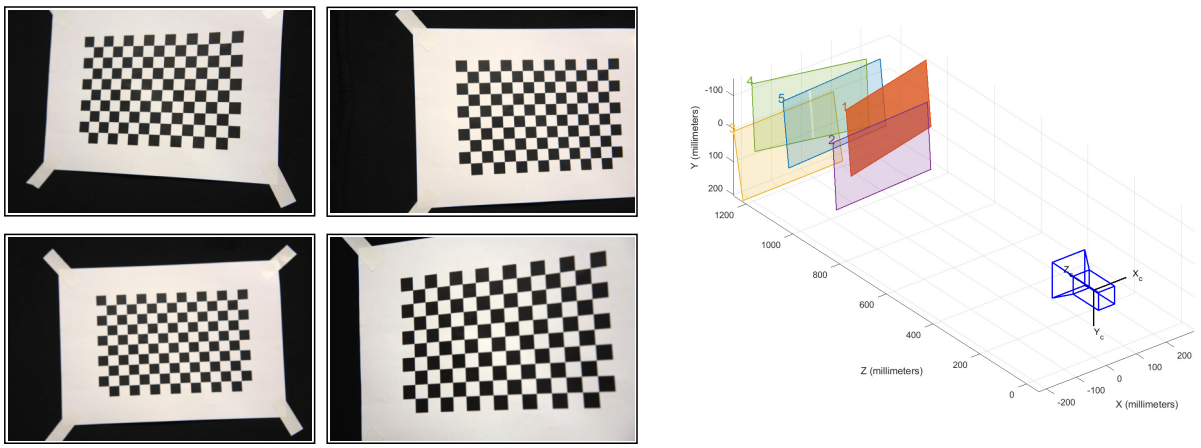


Figure 7.6: A printed checkerboard with known dimensions is used to calculate the camera's intrinsic matrix.

camera matrix (E_D^n) for each light field image. It should be noted that in practice, known patterns are usually not available, and camera pose is determined through more error-prone methods such as structure-from-motion or odometry. These errors propagate to rendering and can degrade the visual quality of the displayed image. Hence it is crucial to only proceed further once the reprojection error is sufficiently low.

Image cropping With the current configuration of the capture setup, a large part of each captured image consists of just black (background) pixels. Hence cropping each image to its respective region of interest can help us save memory. I determine this region of interest by finding the smallest axis-aligned bounding box (AABB) containing all the non-zero pixels for each image and then fitting another AABB over AABBs of all the images together. All images are cropped according to this AABB. Cropping an image changes the intrinsic matrix of the camera. This change is captured by a simple translation of the camera's principal point offset by the pixel coordinate of the top-left corner of the AABB in the original image. Note that further savings in memory can be achieved by having a separate AABB for each image (and thus a separate intrinsic matrix), but I chose to have a single AABB for simplicity.

7.5 Rendering for HDRMFS display

Since one of our goals for this display is to enable the study of perceptual realism, we need to render the scene on each HDR display with the highest quality possible. We discussed a range of photorealistic image-based rendering methods in Section 7.2.2 that can deliver such quality. However, our display design imposes certain challenges on the rendering method:

1. The rendering should be real-time (preferably higher than the LCD refresh rate of 60 Hz).

2. The renderer should be able to process and generate high resolution and HDR images for novel viewpoints without introducing noticeable artefacts (such as blur, noise, and distortion).

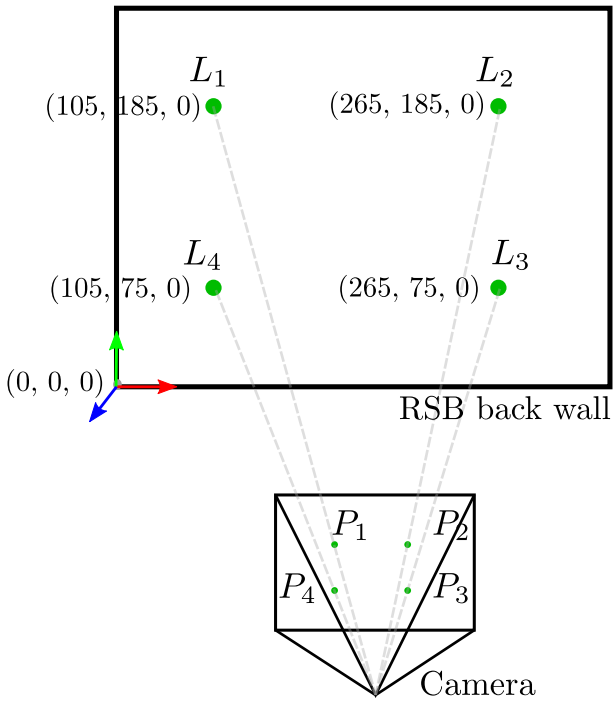


Figure 7.7: RSB coordinate system and camera pose estimation. We define a coordinate system with the bottom-left corner of RSB's back wall as origin, XYZ-axes corresponding to the RGB arrows in the illustration, and units in millimetre. Four LEDs ($L_1 \dots L_4$) are place on the back wall at known positions and are used to calculate the camera pose by solving a system of linear equations formed with their corresponding pixel positions ($P_1 \dots P_4$). The size of the LED's are exaggerated for illustration, in practice, they are $<$1 mm in diameter

3. The rendering method should either take or automatically infer scene depth information to deliver correct depth cues.

I picked three rendering methods satisfying the above requirements with widely different scene representations, allowing them to capture many usage scenarios. They include:

1. Dynamically reparamterised light field rendering: A homography-based light field rendering method with minimal pre-processing requirements and a very low computational overhead [Isaksen et al., 2000]. However, it can only deliver correct depth cues for planar objects as it does not contain depth information.

2. Lumigraph: A lumigraph implemented as view-dependent texture mapping on a single global 3D mesh [Debevec et al., 1996, Gortler et al., 1996]. This method utilises the fast rasterization pipeline of modern GPUs and can deliver correct depth cues. However, the 3D scene must be composed of simple shapes or known geometry.

3. NeX rendering: A learning-based multi-plane image with neural basis expansion [Wizadwongsa et al., 2021]. The planes in its representation can be aligned with the RSB to deliver correct depth cues and the method is capable of learning scenes with arbitrary geometry and shading. However, it requires hours of training to learn a suitable scene representation.

Figure 7.8 summarises the pre-processing steps required by each method to transform the lightfield into a format suitable for rendering. In the following sections, I will describe each method, its pre-processing steps, and its implementation in more detail.

7.5.1 Rendering method 1: Dynamically reparamterised light field rendering

IBR has many applications ranging from free-viewpoint rendering which can be used for view interpolation or generating stereo pairs, relighting, and focus adjustment. Depending on the target application and device, there could be restrictions on available computing budget and latency. These restrictions often exclude the use of IBR methods requiring expensive pre-processing such as 3D reconstruction or training a neural network. In such cases, it is necessary to generate new views without any information about the scene. Light fields are a generalisation of all such rendering methods which parameterise the scene as a 4D function of camera position and orientation (we can drop one dimension from the plenoptic function assuming the radiance of a light ray remains constant in free space). The scene images can be treated as samples of this function, and rendering from new viewpoints can be generated by interpolating between these samples. Isaksen et al. [2000] proposed an alternative reparameterisation of light fields called *"Dynamically reparamterised light fields (DRLF)"* that allows for focus readjustment and achieves real-time rates by utilising multi-texturing and homography.

Given images from N co-planar data cameras C_{D}^{n}, DRLF aims to render the light field from a virtual camera C_{V}. It assumes all cameras to be pin-hole cameras and a focal plane F for the virtual camera. To reduce aliasing, contributions of each data camera are modulated with an aperture function A^{n}. Every pixel $p_{\mathrm{V}}^{(i,j)}$ on the virtual camera can then

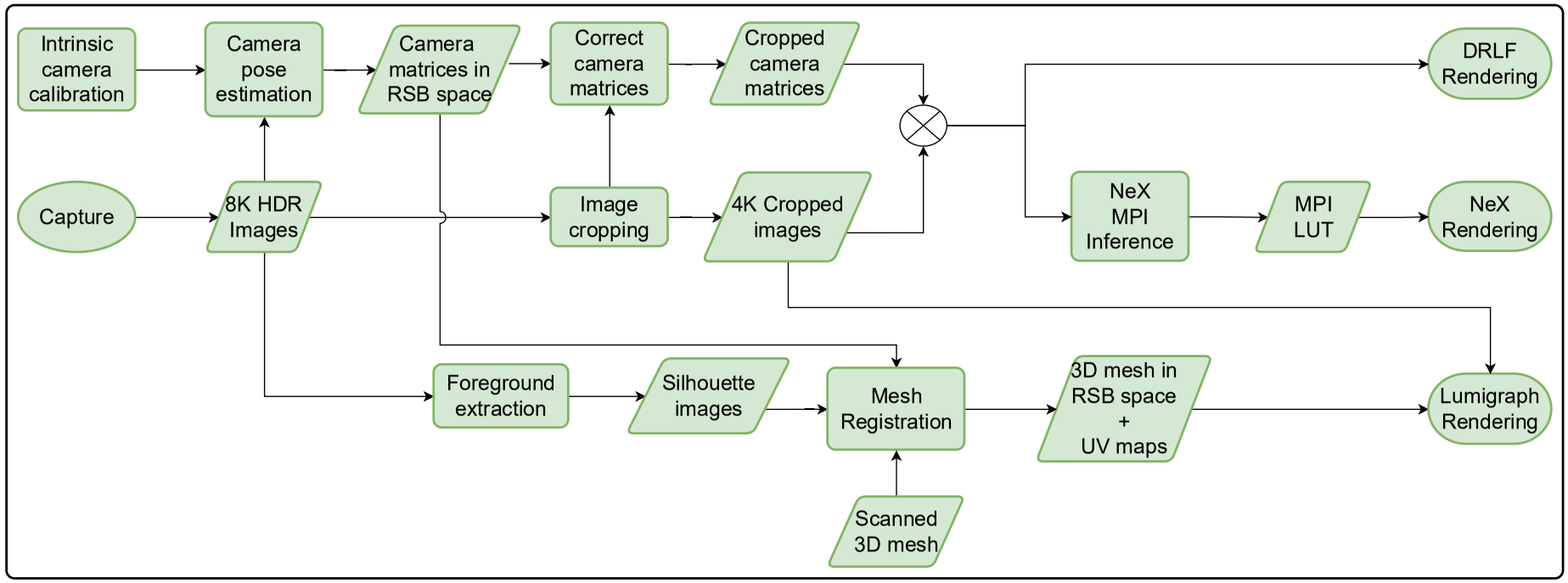


Figure 7.8: 3D reconstruction pipeline. The captured light field images need to be processed into suitable representations before they can be used by the three rendering algorithms. First, the camera poses are calculated for each image, then the images are cropped to save memory, and finally they are used with the camera poses to run DRLF, or, to learn an multi-plane image for NeX, or, to fit a 3D mesh for lumigraph. $\otimes$ represents packing of data together.

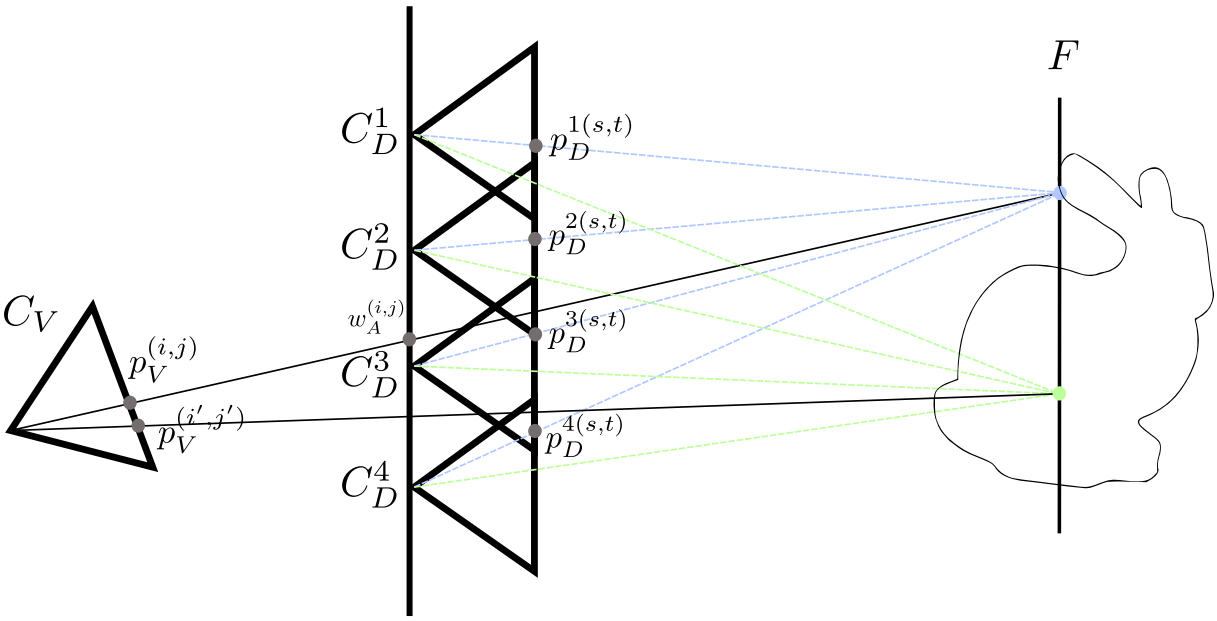


Figure 7.9: DRLF rendering method. Pixel $p_{\mathrm{V}}^{(i,j)}$ on C_{V} is projected on the focal plane F using the inverse of its projection matrix. The point on the focal plane is projected to all data cameras C_{D}^{n} to derive its colour. If the focal plane lies on the object surface (blue dot), all data cameras will have the same value ($p_{\mathrm{D}}^{n(s,t)}$) and the point will appear sharp, otherwise the image will appear blurry (green lines).

be mapped to a corresponding pixel $p_{\mathrm{D}}^{n(s,t)}$ on each data camera using a transformation matrix.

$$p_{\mathrm{V}}^{(i,j)} = \sum_{n=1}^{N} A^{n}\left(w_{\mathrm{A}}^{(i,j)}\right) \cdot p_{\mathrm{D}}^{n(s,t)}, \tag{7.1}$$

$$p_{\mathrm{D}}^{n(s,t)} = K_{\mathrm{D}} P_{\mathrm{D}} E_{\mathrm{D}}^{n} (K_{\mathrm{V}} P_{\mathrm{V}} E_{\mathrm{V}})^{-1} p_{\mathrm{V}}^{(i,j)}, \tag{7.2}$$

where K is the intrinsic matrix of the camera, P is the projection matrix, E is the extrinsic matrix and subscripts V and D represent virtual and data camera, respectively. $w_{\mathrm{A}}^{(i,j)}$ is the projection of $p_{\mathrm{V}}^{(i,j)}$ onto the aperture plane in world space. The intrinsic and extrinsic matrices of the data cameras are the same as described in Section 7.4.1. To make the intrinsic and projection matrices of the virtual camera invertible and handle arbitrary focal planes, they are modified to be of the form

$$K_{\mathrm{V}} P_{\mathrm{V}} = \begin{bmatrix} f_x & s & 0 & x_0 \\ 0 & f_y & 0 & y_0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ n_x^c & n_y^c & n_z^c & -N^c \cdot w_F^c \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{7.3}$$

where, (f_x, f_y) is the focal length, s is the axis skew, (x_0, y_0) is the principal point offset, $[n_x^c, n_y^c, n_z^c]$ is the focal plane normal N, w_F is the position of the focal plane in 3D world space, and the superscript c represent that the focal plane is defined in camera coordinate system (Figure 7.9).

Implementation I implemented this rendering method in OpenGL and GLSL. The method only takes the cropped images and cropping-adjusted camera matrices as input and does not require any pre-processing. The images from all the data cameras are stored in an Array Texture. The transformation matrices are calculated once during initialisation and passed to the shaders. For each fragment, I calculated its colour by solving Eq. (7.1). I used a box filter as the aperture function (nearest-neighbour strategy) for all cameras to prevent ghosting artefacts in the results. If we iterate through all data cameras for every pixel (for e.g., to create depth of field effect), the complexity of this algorithm grows linearly with the number of data cameras for a given display.

7.5.2 Rendering method 2: Lumigraph implemented as view-dependent texture mapping

Using a single focal plane to represent our scene might work if the scene is mostly planar; however, it will be error-prone if the scene has a large disparity. To handle such cases, it is necessary to have information about the scene's geometry. Lumigraph is an extension of light field that projects the light rays on a geometry proxy of the scene. The more accurate this proxy is, the fewer errors in the rendered image. However, generating this geometry proxy from images (also called 3D reconstruction) is still an open problem. Since 3D reconstruction is not a focus of this work, we either use objects of known shapes (like spheres or cubes), or generate high-quality proxy meshes of the scene using laser scanning and derive their correct coordinate transformations using differentiable rasterization as described below.

Proxy mesh registration The proxy mesh of the scene needs to be translated, rotated, and scaled in the RSB coordinate system to perfectly overlap with the real object inside the box. To find this transformation, we first estimate the object's silhouette in each light field image. Since the object's background is uniformly black in our setup, a simple threshold-based method is sufficient to extract the silhouette.The grab cut algorithm [Rother et al., 2004] might be more suitable for complex objects such as refractive material or a very dark surface. The silhouettes along with the proxy mesh and camera poses are then passed to a differentiable rasterizer (*SoftRas* [Liu et al., 2019]) implemented in Pytorch3D [Ravi et al., 2020] to determine the spatial transformation for the mesh that minimises the error between rendered mesh silhouettes and extracted silhouettes. I used full resolution images for this step, however, the process can be accelerated by using cropped images and camera poses.Once the mesh is transformed into the correct position, I generate its UV coordinates by reprojecting mesh vertices onto cropped lightfield images using cropping-corrected camera matrices.

Implementation We now have a single mesh with multiple UV maps corresponding to each view. We can render this mesh from the virtual camera C_{V} using the traditional OpenGL pipeline and calculate the colour of each pixel through simple texture look operations in the fragment shader. The mesh is stored in OpenGL Array Buffer with multi-texturing enabled. The method is implemented as a 2-pass OpenGL rendering. In the first pass, I render the mesh N times using UV coordinates of a different data camera each time and store the rendered results in a 2D Array Texture. In the second pass, I calculate the final pixel colour by modulating each rendered image with the aperture

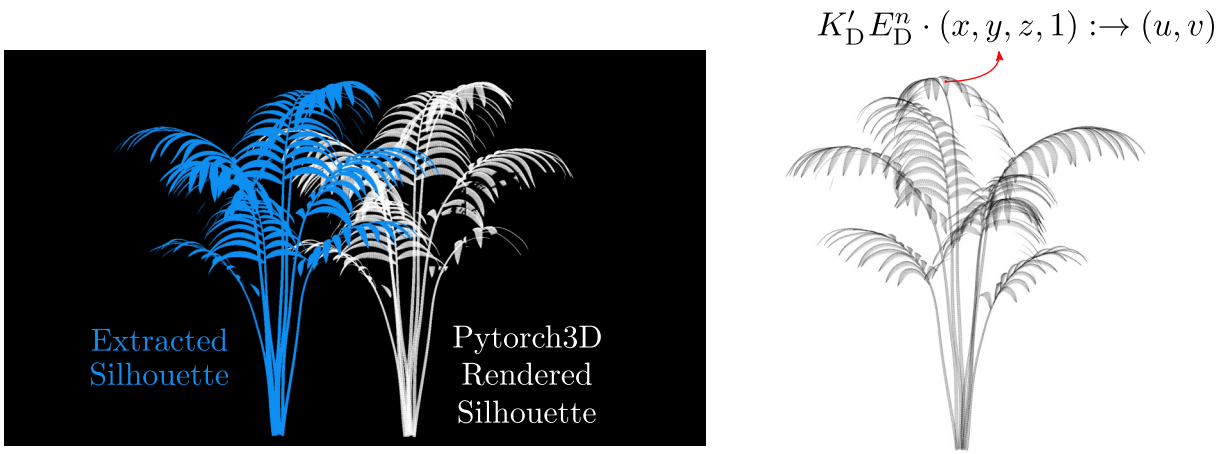


Figure 7.10: The object silhouette is extracted from each light field image (blue plant) and a proxy mesh is provided to Pytorch3D's differentiable rasterizer. An optimal 3D transform for the mesh is found such that the rendered silhouette (white plant) perfectly overlaps with extracted silhouette. Once we know the correct mesh coordinates, we can texture map them to each light field image using its extrinsic matrix (E^n_D) and cropping-corrected intrinsic matrix (K'_D).

function and summing them together (similar to Eq. (7.1)). In addition to the number of data cameras, the time complexity of this method also grows with the number of triangles in the proxy mesh.

7.5.3 Rendering method 3: NeX

In cases where a good quality geometry proxy of the scene is not readily available, the 3D information of the scene can be learned using neural networks. Such neural rendering methods can learn an intermediate representation of the scene during training and use this representation for rendering novel views in real-time. One such successful representation is multi-plane images consisting of D planar images of dimensions $H \times W \times 4$ where the last channel is the RGB and alpha transparency value. The images are scaled and placed equidistantly along a reference camera frustum. New views ($\widehat{I}$) can be rendered from this RGBα MPI by transforming each plane to the target camera view via homography and composting them together:

$$\widehat{I} = \sum_{d=1}^{D} \mathbb{W}\left(c_d\right) \cdot \mathbb{W}\left(\alpha_d\right) \prod_{i=d+1}^{D} \left(1 - \mathbb{W}\left(\alpha_i\right)\right), \tag{7.4}$$

where, c_d and α_d are the colour and alpha image of the d'th layer, and $\mathbb{W}$ is the homography warping function that warps each image to the target view. The above rendering equation is differentiable, allowing the MPI to be learned through a neural network.

Though this approach gives good results, it assumes Lambertian surfaces and cannot handle view-dependent effects. To overcome this, NeX proposed by Wizadwongsa et al. [2021], models each pixel as a function of viewing direction:

$$\mathbb{C}^p\left(v\right) = k^p_0 + \sum_{n=1}^{N} k^p_n \, H_n\left(v\right), \tag{7.5}$$

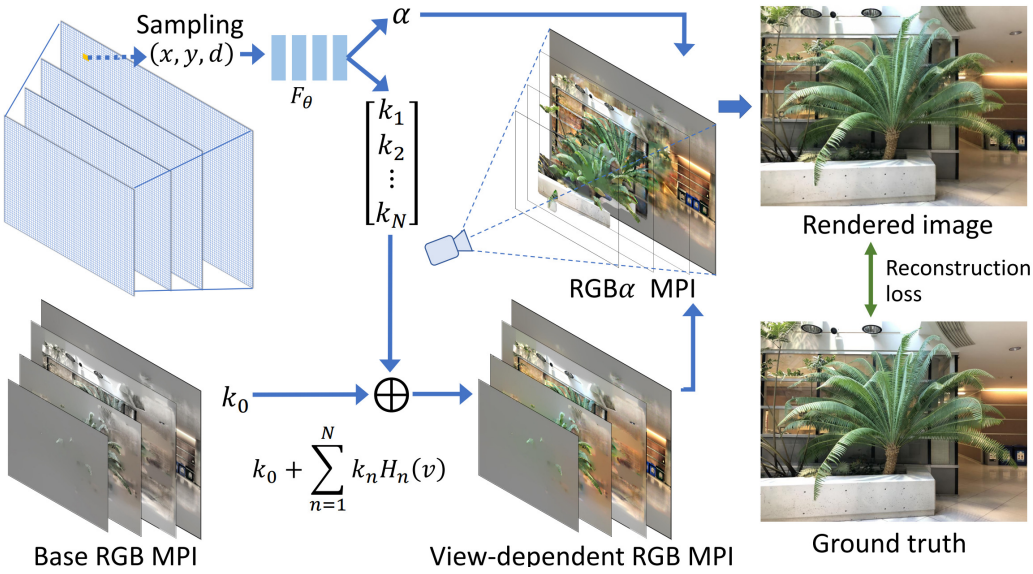


Figure 7.11: NeX overview as provided by [Wizadwongsa et al., 2021]. The colour of each pixel on each plane is a combination of base RGB k_0, transparency α, and view-dependent neural basis coefficients $k_n H_n(v)$.

where p is the pixel, $v = (v_x, v_y, v_z)$ is the viewing direction, and $\mathbb{C} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is 3D mapping function from v to pixel colour. Unlike traditional MPI which only stores the RGB value for each pixel in each planar image, NeX stores the base RGB value (k_0) and N additional reflectance parameters $(k_1, k_2, \cdots, k_N)$ such that $k_n^p \in \mathbb{R}^3$. Each reflectance parameter has a corresponding basis function $H_n(v) : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, which is shared across all pixels.

The parameters $\alpha, k_1, \cdots, k_n$ are learned using the MLP F_θ, the global basis functions $H_1, H_2, \cdots, H_N$ are learned using another MLP G_ϕ, and k_0 is optimized explicitly as a learnable parameter with a total variation regularizer ($\mathbb{T}$) in the image reconstruction loss ($\mathbb{L}(\cdots)$):

$$F_\theta : (\mathtt{x}) \rightarrow (\alpha, k_1, k_2, \cdots, k_N)\,, \tag{7.6}$$

$$G_\phi : (v) \rightarrow (H_1, H_2, \cdots, H_N)\,, \tag{7.7}$$

$$\mathbb{L}\left(\widehat{I}_i, I_i\right) = \left\|\widehat{I}_i - I_i\right\|^2 + \omega \left\|\nabla\widehat{I}_i - \nabla I_i\right\|_1 + \gamma\mathbb{T}(k_0), \tag{7.8}$$

where θ and ϕ are the network parameters, $\mathtt{x} = (x, y, d)$ is the pixel location (x,y) at plane d, $\widehat{I}_i$ and I_i are the reconstructed image-reference image pairs, and ω and γ are model hyperparameters. In my implementation, I used the same architecture and hyper-parameters as described in their original publication (Figure 7.11).

Real-time implementation For learning Eq. (7.6) and Eq. (7.7), I use the Pytorch implementation provided by the authors. Using the loss function in Eq. (7.8) directly on linear HDR colour values does not give good results because such values are non-linearly related to our perception of visible differences. I adapt the code to support our HDR images by encoding them with PU21 [Azimi et al., 2021] to convert absolute HDR linear

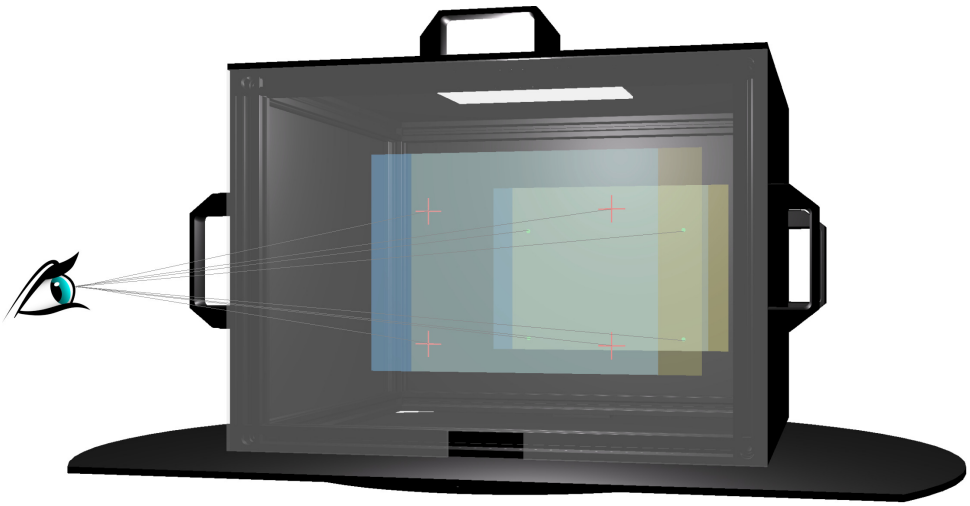


Figure 7.12: RSB with a detachable transparent plane with red markers. Users are asked to mark the pixels on all 4 displays (blue and yellow planes) corresponding to the 4 crosses and 4 LEDs on the back wall. This correspondence is used to calculate the transformation matrices for each display.

colour values into approximately perceptually uniform (PU) values. The encoded cropped HDR images and cropping-corrected camera matrices from Section 7.4.1 are passed to the Pytorch implementation. Once the training is complete, F_θ and G_ϕ are densely sampled and stored as a LUT in the form of 16-bit floating-point RGBA HDR textures. I then procedurally generate a triangle mesh made up of D planes with size and location corresponding to the MPI planes. The planes in this mesh are then texture mapped to the HDR LUT, and the mesh is rendered back to front from the virtual camera C_V. The final colour of the pixel is calculated in the fragment shader by looking up the values of k_0^p, k_n^p, and H_n in the HDR LUT/texture and solving Eq. (7.5). The time complexity of this method only grows with the number of MPI layers and number of basis.

7.5.4 Eye calibration and multi-focal rendering

The rendering methods discussed above provide a way to render the RSB scene from an arbitrary viewpoint. In our setup, we need to render from the viewpoint of the user's eyes. Hence, for both eyes, we need to calculate the intrinsic and extrinsic transformation matrices corresponding to near and far displays. We ask every user to perform a calibration step. A detachable transparent plane with 4 red crosses at known positions is attached to the near plane of the RSB, and the 4 LEDs on the back wall are switched on (Figure 7.12). The users are then shown virtual crosses on each display and are asked to align them with the real crosses/LEDs. Using the positions of the 8 real points and their corresponding pixels on each display, we determine a 3×4 projection matrix for each display using the direct linear transformation (DLT) approach [Hartley and Zisserman, 2003]. The projection matrix is further decomposed into an intrinsic and extrinsic matrix using RQ decomposition [Simek, 2012]. The two matrices define the virtual camera in the above rendering methods and generate physically correct views for all displays. To deliver correct focal cues, the rendering on the near and far displays are filtered using linear depth filtering in the diopter space [Akeley et al., 2004a].

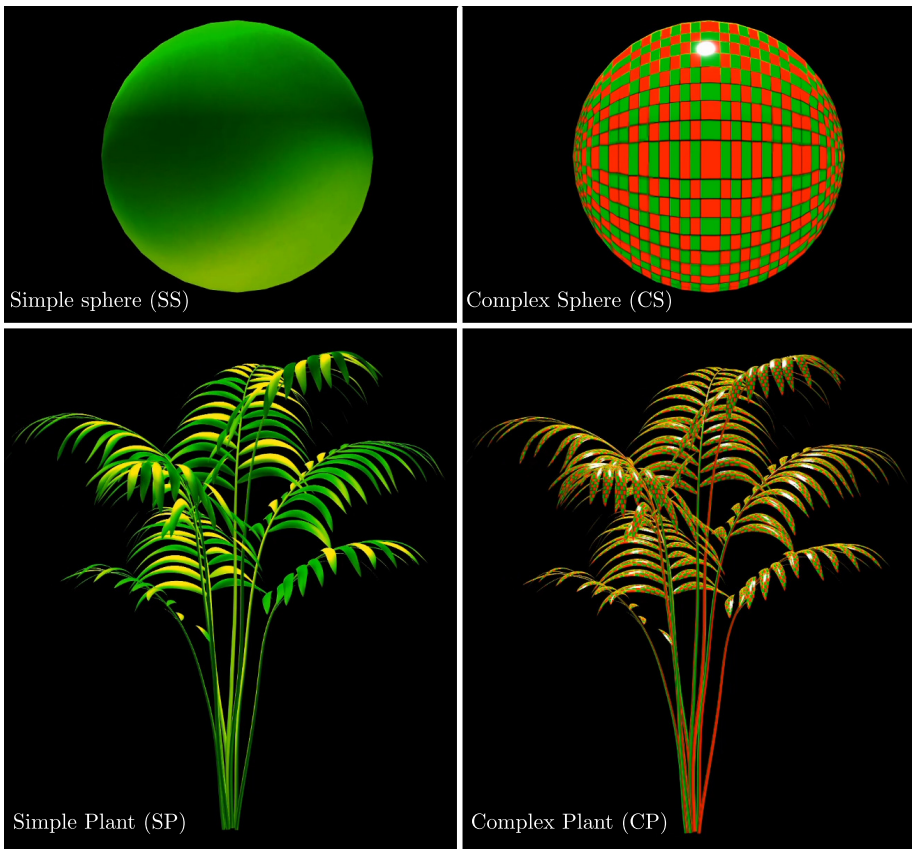


Figure 7.13: 3D scenes used for comparing different IBR methods

7.6 Benchmarking rendering methods

The rendering methods I have implemented for our display have very different representations. It is crucial to understand their strengths and weakness to decide which method to use for a given experiment. This section compares the three methods on synthetic scenes of varying geometry and material complexity. I choose to use synthetic images instead of real images to discard the effect of inaccuracies from pose estimation and 3D reconstruction.

Synthetic scenes A total of 4 scenes (2 geometries $\times$ 2 materials) were used for comparison (Figure 7.13). The geometries used were a sphere mesh (2 800 vertices and 960 triangles) and a plant mesh (254 244 vertices and 84 748 triangles). The sphere mesh was chosen to represent a simple scene with smoothly changing disparity and no occlusions. In contrast, the plant mesh was chosen to create a challenging scene with a high triangle count, thin edges, disocclusions and discontinuous disparity. The meshes were mapped to two materials: a Lambertian material with a low-frequency gradient image as its diffuse component and a specular material (Phong shading) with a high-frequency checkerboard as its diffuse component. Most IBR methods assume Lambertian surfaces in their scene representation and specular surfaces continue to be an ongoing challenge in this field of research. I created a scale-accurate digital twin of the RSB and light field capture setup in the Unity3D game engine and used it to generate light field images and accurate camera poses.

Experiment setup I rendered 20 images of 2160×1440 resolution and 16-bit per channel (linear RGB) with a total baseline of 100 mm for all 4 scenes. The images were divided into two sets of 10 images: the even number images were fed to the rendering methods for novel view synthesis and the odd images were kept aside for validation. Accurate object silhouettes and camera matrices for each image were extracted from unity and were used to generate lumigraph (Section 7.5.2) and NeX MPI LUTs (Section 7.5.3). The MATLAB/OpenGL implementations of the three methods was used to generate new views from the perspective of validation images to facilitate objective quality evaluation. The computational performance of each method was also measured over 1000 frames on a i7-8700 CPU @ 3.20GHz, 32GB RAM and an RTX 2080Ti GPU with V-Sync disabled. Since existing quality metrics only support the evaluation of a single view, I restricted this experiment to a single display, but I expect the computational cost to grow linearly with each display. In addition to the validation poses, 60 Hz videos of the rendering results were also generated and are available on the project web page[1].

Results The three methods have different computational requirements. The DRLF method is computationally the simplest method with the highest average FPS and lowest VRAM requirements (Figure 7.16). The lumigraph method is close second both in terms of performance and memory requirements. The memory required by its mesh representation is negligible compared to HDR images. There is also no noticeable effect of the number of triangles between sphere and plant scene, indicating that mesh size is not a bottleneck on modern GPUs. NeX is the slowest method and also the most memory intensive (20 times more memory). However, it should be noted that my implementation used 72 layers and 6 sub-layers in the NeX MPI representation (as suggested by the authors). A hyper-parameter sweep for a trade-off between quality and performance could prove helpful here but may not be practical as every training session required approximately 8 hours to converge. Also, I found one of the biggest bottlenecks in NeX's performance to be the texture-lookup operation on MPI LUTs (15360×3680 16-bit RGBA textures). An implementation that makes better use of the GPU memory architecture might improve performance.

The artefacts induced by the three methods are also quite different. DRLF works well on the simple sphere but blurs the regions away from the focal plane in other scenes. This is visible as blurred textures, or missing thin triangles (Figure 7.14). Lumigraph renders sharp textures in all scenes but maps incorrect textures near thin edges (disocclusions) (Figure 7.15a). Since lumigraph uses the rasterization of a mesh, the geometry edges can also suffer from aliasing artefacts. In both methods, these artefacts become more prominent in videos and appear as distracting flickering near thin edges or juddery specular highlight. NeX works well on both thin edges and specular highlights but gravely affects the texture appearance of checkerboard material (the width of the checkerboard squares appears to change in the video). Also, NeX induces unnatural halos around the object when the rendering viewpoint is far from MPI's reference pose (Figure 7.15b).

To quantify the quality of each method, I compared their renderings for validation poses against the validation images using 3 popular objective quality metrics: PSNR, SSIM, and HDR-VDP-3. Since PSNR and SSIM were designed for SDR images, the HDR images were first encoded using PU21 transform Azimi et al. [2021]. HDR-VDP-3 assumed a linear RGB colour space and a 45 ppd angular resolution. The plots in (Figure 7.16)

[1] `https://www.cl.cam.ac.uk/research/rainbow/projects/hdrmfs/rendering_methods/`

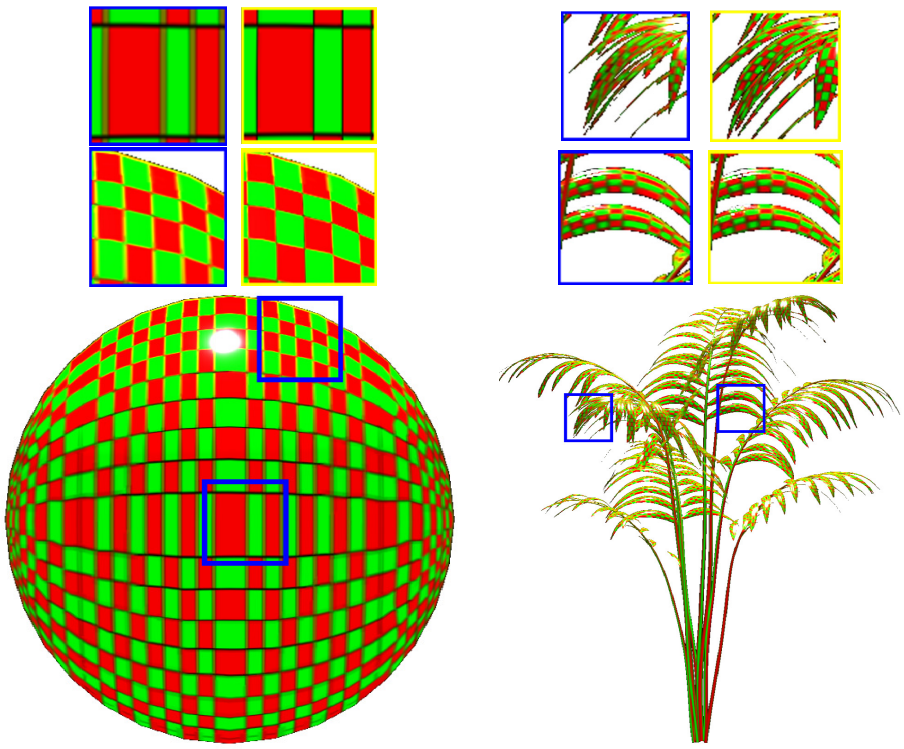
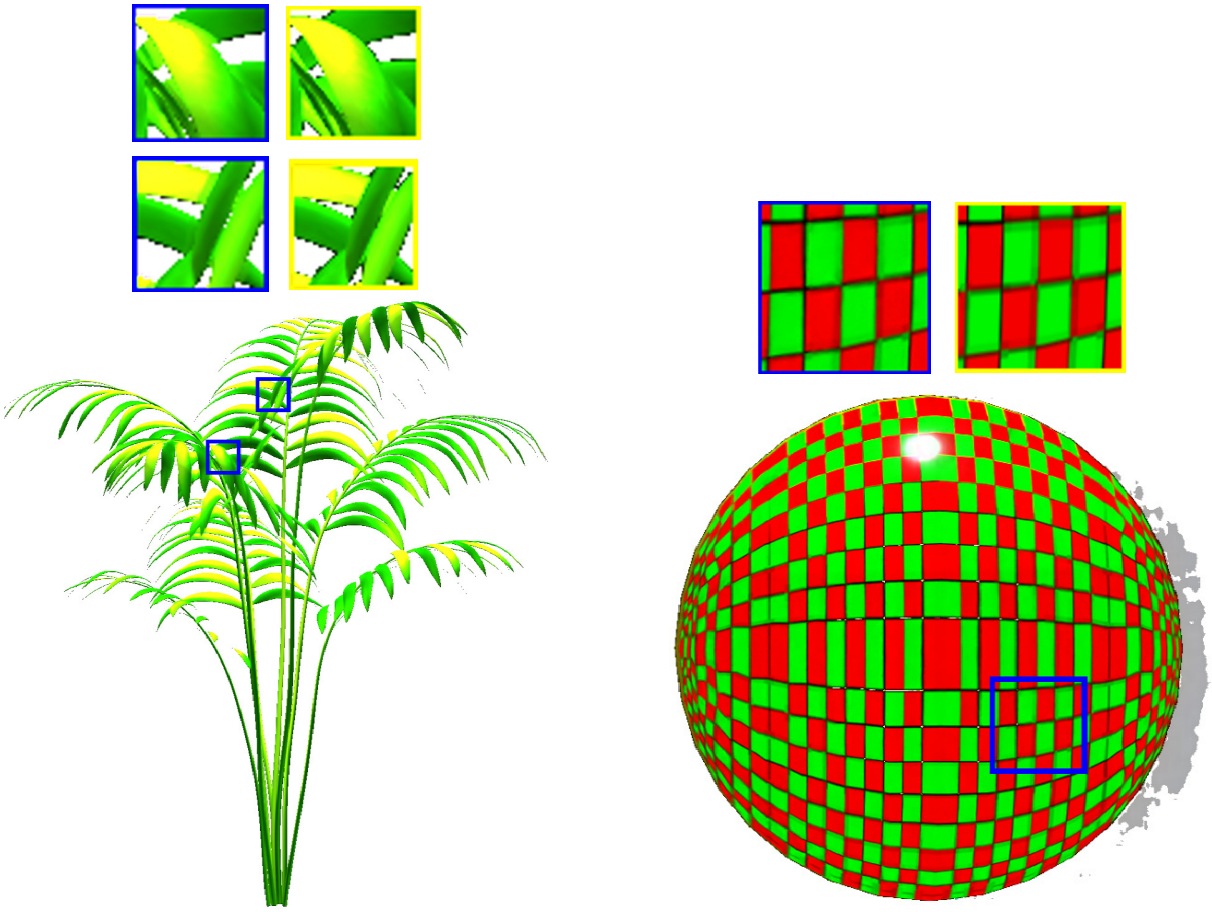


Figure 7.14: DRLF artefacts. The focal plane was set to the center of the object. Points near the focal plane appear sharper while points further away appear distorted. (Rendered results in blue boxes and ground truth in yellow boxes)

show the quality results averaged over 10 validation images, and the error bars show the standard deviation. Additionally, I also run a blind video quality assessment metric MDTVSFA [Li et al., 2021] on the videos recorded for each method. MDTVSFA is one of the leading metrics on MSU Video Quality Metrics Benchmark 2022 [Vatolin et al., 2022]. Overall, the lumigraph rendering method is consistently rated to be of the highest quality in all 4 scenes. However, there is not much consensus on the rating and ranking of the methods across 4 metrics. Lumigraph and DRLF have similar quality (low std. deviation) across all 10 validation images (except DRLF for SP) (Figure 7.13), but NeX have a higher variance due to artefacts in images far from the reference pose. Since the metrics I used are some of the most popularly used metrics in IBR evaluation, it is crucial to understand which of these metrics best relate to human ratings. Note that none of these metrics were designed for IBR artefacts, and most of them do not account for HDR or temporal artefacts such as judder and flicker. Though not part of this dissertation, I'm preparing a subjective study to compare the 3 methods on our HDR-MFS display in terms of perceptual realism and evaluate the performance of existing objective quality metrics.

7.7 Experiments conducted on HDRMFS display

Since its completion, our display pipeline has facilitated two (previously impossible) psychophysical studies and will continue to be a platform for the next generation of perceptual realism experiments. I will briefly describe them next.



(a) Lumigraph artefacts

(b) NeX artefacts

Figure 7.15: Lumigraph suffers from texture ghosting in areas of disocclusion. NeX causes changes in texture appearance and adds unnatural halos around object when the camera is far away from the reference pose. (Rendered results in blue boxes and ground truth in yellow boxes).

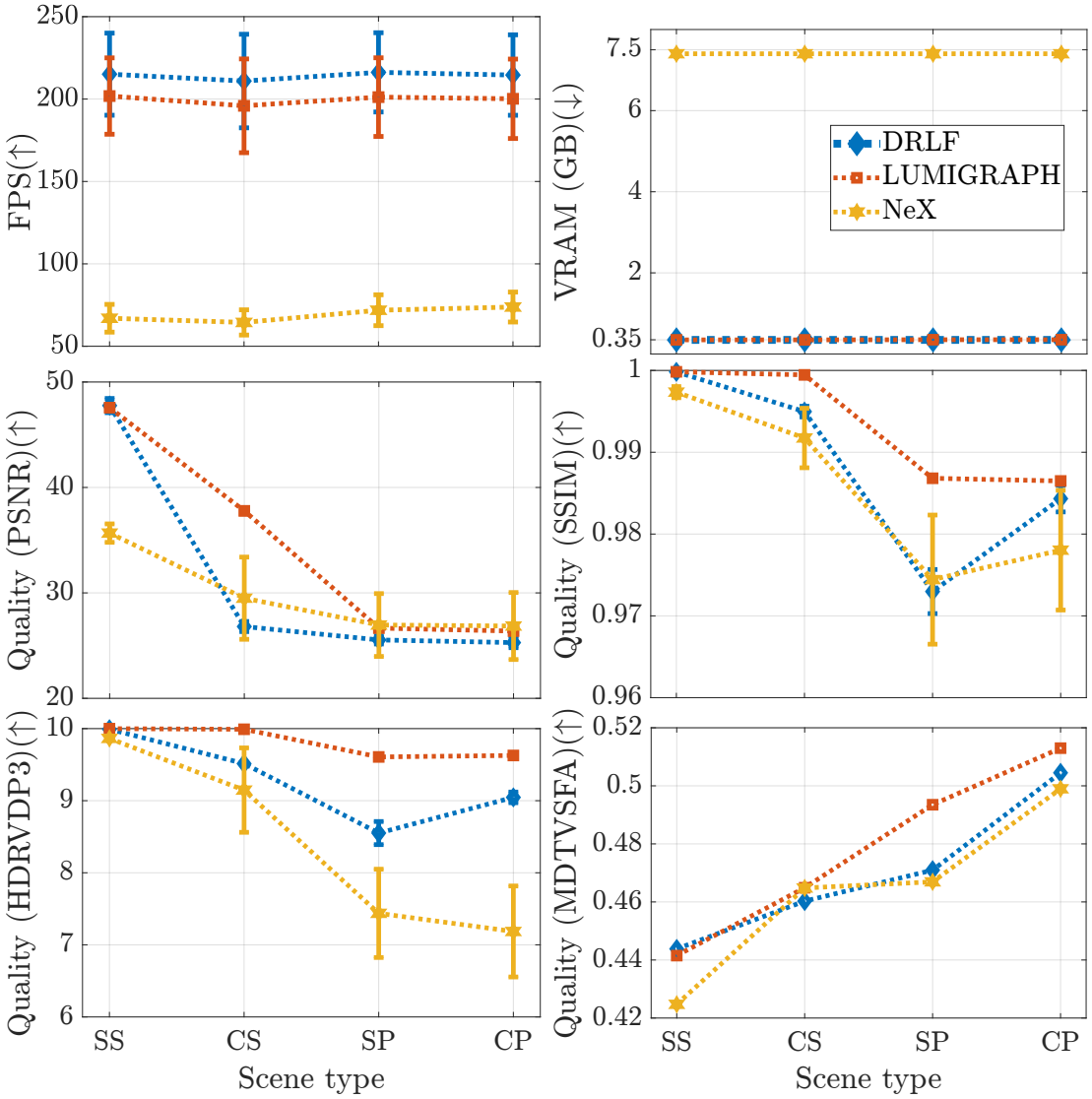


Figure 7.16: Performance and qualitative comparison of the three rendering methods. NeX is the most computationally expensive method among the three tested methods. There is not much agreement in the results of the four tested quality metrics.

Visual Turing test My collaborators Zhong et al. [2021] conducted a study to determine if the participants could distinguish between real and virtual objects displayed by our system (also known as the visual Turing test). Another goal of this study was to test the sensitivity of HVS to degradation of contrast in the presence of all other realism cues. A wooden sphere was used as the stimuli and Lumigraph as the rendering method. The experiment followed a three-interval-forced-choice (3IFC) methodology where in each trial, participants were shown 3 intervals in which two were real and one virtual or two virtual and one real. Participants were asked to pick the interval that appeared different from the other two. Results indicated the probability of detecting the difference between real and virtual objects to be 0.44. The degradation of contrast by 20% increased the probability of detection to 0.56. The experiment shows a promising avenue for scaling different distortions (in colour, disparity, and focus) in terms of their effect on perceptual realism.

Transparency in AR My collaborator Liu et al. [2022] conducted a study on our display to find the minimum brightness required for a virtual image to appear completely opaque in an optical see through augmented reality display. In the experiment, the back wall of the real scene box was covered with a checkerboard pattern and participants were shown a virtual square with noise texture on either near or far display using lumigraph rendering. The frequency and contrast of noise, focal depth (near or far plane), and luminance of the RSB were varied in each trial. The participants were asked to adjust the luminance of the square to the smallest brightness at which the square appears opaque and the background does not show up on the square. The results of the experiment show that it takes up to 2 orders of magnitude more brightness than background luminance to render the foreground completely opaque. The contrast of the virtual object and the amount of defocus blur also impacted this value significantly. These results can help facilitate the development of new AR displays.

7.8 Summary

Continually improving our understanding of the limits of the human visual system is one of the core ideas behind perceptual graphics research. One path to this is to build new display architectures that enable the study of previously unstudied factors. In this chapter, I presented a novel end-to-end capture-display system that combines several realism cues such as high dynamic range, binocular disparity, colour, focal depth, and resolution at high fidelity. I implemented three different image-based rendering methods adapted to the requirements of our display. A qualitative assessment of their results revealed that the performance of the state-of-the-art image quality metrics on reconstruction artefacts remains to be improved. Since its completion, the display has facilitated two previously impossible psychophysical studies while several more are under preparation.

Chapter 8

Conclusion

The main goal of this work was to demonstrate how the knowledge of our visual system's spatial and temporal limitations can significantly improve the quality and performance of image synthesis. It is becoming increasingly necessary to exploit these limitations as displays and consumers' computational demands grow and GPUs transmission and power bandwidth struggle to keep up. In this dissertation, I proposed a visual model that can predict how the perceived quality of rendered content changes under motion w.r.t. content and display parameters. I developed new rendering algorithms that can utilise such models to optimise quality under constrained bandwidth. I also presented a novel display that has and will continue to enable a new set of psychophysical experiments on previously unstudied factors.

8.1 Contributions

The main contributions of my dissertation are as follows:

Review of motion-adaptive rendering and motion psychophysics In Chapter 3, I compared a state-of-the-art motion-adaptive rendering strategy with traditional constant resolution methods in a preference study involving realistic motion scenarios. This study demonstrated how adding motion quality consideration to rendering can significantly boost the quality of experience. The study also identified several limitations in existing approaches. In Chapter 4, I conducted an extensive literature review on psychophysical studies on motion quality to identify its relationship to different content and display factors. I also conducted a psychophysical experiment to fill in some research gaps I identified through this review, specifically, the effect of display persistence, resolution, and aliasing on quality.

CaMoJAB: Content-adaptive model of judder, aliasing and blur In Chapter 5, I showed how frequency domain analysis of different steps of rendering and display pipeline can be combined with spatio-temporal contrast sensitivity functions to estimate the quality of motion for three prominent motion artefacts: judder, aliasing, and blur. The model accounts for several display and content parameters such as refresh rate, resolution, persistence, luminance, velocity, shading rate, and spatial frequency. It was trained and tested on several datasets with only a few fitted parameters and significantly outperforms

state-of-the-art metrics. This work has received interest from several industry partners who develop VRS-enabled GPUs.

ALSaRR: Adaptive local shading and refresh rate While perceptual models like CaMoJAB readily lend themselves to optimising rendering performance and quality, it is not always straightforward to integrate them with real-time rendering methods. In Chapter 6, I proposed a novel rendering method that uses a visual model to distribute the shading budget to the most important parts of the spatio-temporal domain. It formulates the distribution of shading samples and refresh rate as an optimisation problem of maximising quality for a fixed computational bandwidth and solves it using an optimal dynamic programming solution as well as a fast greedy approximation. The method has been integrated with the Unity3D game engine as a plugin and has been shown to out-perform existing adaptive and traditional constant resolution methods on various scenes and multiple bandwidths.

HDRMFS: High-Dynamic-Range Multi-Focal Stereo Display Psychophysical experiments are usually not conducted using real objects but rather images presented on a display device. Displays have inherent limitations, such as a limited range of parameters to be studied or the inability to present multiple perceptual cues together. In Chapter 7, I helped build a novel display that combines multiple realism cues such as high dynamic range, binocular disparity, focal depth, colour, and resolution at a high fidelity to enable previously impossible perceptual experiments. I built a light field rendering system adapted to this display which offers three rendering algorithms depending on the performance requirements and the amount of depth information available of the 3D scene. A qualitative analysis of the three rendering methods found little agreement between different objective quality metrics, indicating a need for further studies in this direction. Since its conception, the display and rendering system has already enabled two experiments, and more are in progress.

8.2 Future work

Building computational models of the human visual system is an incredibly complex problem. While CaMoJAB metric made strides toward modelling most relevant factors of motion quality, it does not model contrast masking [Legge and Foley, 1980, Watson and Solomon, 1997], degradation of colour, foveation [Tursun et al., 2019] or saliency. Those limitations are partially dictated by the scarcity of available data, and partially by the real-time requirement of applications. The complexity of contrast masking would make the metric too costly for a real-time application, and foveation would require eye tracking. However, motion in peripheral vision is known to strongly mask changes in object appearance [Suchow and Alvarez, 2011] and can potentially increase the computational savings of existing foveated rendering methods. This work also does not account for individual variations but models the average observer. Modelling individual variations would require much more measurements from each participant. In Chapter 7, we saw how computationally demanding neural rendering methods such as NeX are. It remains an open question: What is the best way to incorporate perceptual priors into neural representations and algorithms? Another exciting direction is using perceptual models to accelerate path tracing as dedicated chips for tracing rays become increasingly common in

modern GPUs. While the flexible nature of path tracing greatly facilitates perceptually uniform distribution of ray samples in screen-space, it remains to be seen how the use of perceptual models can be moved to path space for faster variance reduction. Finally, I would like to point out that perceptually optimising rendering would be impossible if the underlying hardware were not flexible enough to accommodate it. Giving developers more control of the hardware, such as spatially varying refresh rate or persistence, will provide an opportunity to further improve pixel quality and reduce computational cost.

References

Kurt Akeley, Simon J. Watt, Ahna Reza Girshick, and Martin S. Banks. A stereo display prototype with multiple focal distances. *ACM Trans. Graph.*, 23(3):804–813, August 2004a. ISSN 0730-0301. doi: 10.1145/1015706.1015804. URL `https://doi.org/10.1145/1015706.1015804`.

Kurt Akeley, Simon J Watt, Ahna Reza Girshick, and Martin S Banks. A stereo display prototype with multiple focal distances. *ACM transactions on graphics (TOG)*, 23(3): 804–813, 2004b.

Tomas Akenine-Möller, Eric Haines, and Naty Hoffman. *Real-time rendering*. Crc Press, 2019.

Dmitri Alekseevsky. Microsaccades, drifts, hopf bundle and neurogeometry. *Journal of Imaging*, 8(3):76, 2022.

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D Fairchild. Flip: A difference evaluator for alternating images. *Proc. ACM Comput. Graph. Interact. Tech.*, 3(2):15–1, 2020.

Maryam Azimi et al. Pu21: A novel perceptually uniform encoding for adapting existing quality metrics for hdr. In *2021 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2021.

Peter G. J. Barten. Formula for the contrast sensitivity of the human eye. In Yoichi Miyake and D. Rene Rasmussen, editors, *Image Quality and System Performance*, volume 5294, pages 231 – 238. International Society for Optics and Photonics, SPIE, 2003. doi: 10.1117/12.537476. URL `https://doi.org/10.1117/12.537476`.

Mark R Bolin and Gary W Meyer. A frequency based ray tracer. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 409–418, 1995.

FW Campbell and RW Gubisch. Optical quality of the human eye. *The Journal of physiology*, 186(3):558–578, 1966.

Kirsten Cater, Alan Chalmers, and Greg Ward. Detail to attention: exploiting visual tasks for selective rendering. In *ACM International Conference Proceeding Series*, volume 44, pages 270–280, 2003.

Jen-Hao Rick Chang, B. V. K. Vijaya Kumar, and Aswin C. Sankaranarayanan. Towards multifocal displays with dense focal stacks. *ACM Trans. Graph.*, 37(6), December 2018.

ISSN 0730-0301. doi: 10.1145/3272127.3275015. URL https://doi.org/10.1145/3272127.3275015.

Alexandre Chapiro, Robin Atkins, and Scott Daly. A luminance-aware model of judder perception. *ACM Transactions on Graphics (TOG)*, 38(5):1–10, 2019.

Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, 1993.

Per H. Christensen and Wojciech Jarosz. The path to path-traced movies. *Found. Trends. Comput. Graph. Vis.*, 10(2):103–175, October 2016. ISSN 1572-2740. doi: 10.1561/0600000073. URL https://doi.org/10.1561/0600000073.

James H Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 19(10):547–554, 1976.

Roelof Roderick Colenbrander. Adaptive graphics for cloud gaming, May 20 2021. US Patent App. 16/688,369.

Scott Daly, Ning Xu, James Crenshaw, and Vikrant J Zunjarrao. A psychophysical study exploring judder using fundamental signals and complex imagery. *SMPTE Motion Imaging Journal*, 124(7):62–70, 2015.

James Davis, Yi-Hsuan Hsieh, and Hung-Chi Lee. Humans perceive flicker artifacts at 500 hz. *Scientific reports*, 5:7861, 2015.

Kurt Debattista, Keith Bugeja, Sandro Spina, Thomas Bashford-Rogers, and Vedad Hulusic. Frame rate vs resolution: A subjective evaluation of spatiotemporal perceived quality under varying computational budgets. In *Computer Graphics Forum*, volume 37, pages 363–374. Wiley Online Library, 2018.

Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996.

Gyorgy Denes, Akshay Jindal, Aliaksei Mikhailiuk, and Rafał K. Mantiuk. A perceptual model of motion quality for rendering with adaptive refresh-rate and resolution. *ACM Transactions on Graphics*, 39(4):133, jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392411. URL https://dl.acm.org/doi/10.1145/3386569.3392411.

Piotr Didyk, Elmar Eisemann, Tobias Ritschel, Karol Myszkowski, and Hans-Peter Seidel. Apparent display resolution enhancement for moving images. In *ACM SIGGRAPH 2010 papers*, pages 1–8. 2010.

Michal Drobot. Software-based variable rate shading in call of duty: Modern warfare. In *ACM SIGGRAPH 2020 Courses*. 2020.

Jean-Philippe Farrugia and Bernard Péroche. A progressive rendering algorithm using an adaptive perceptually based image metric. In *Computer Graphics Forum*, volume 23, pages 605–614. Wiley Online Library, 2004.

Xiao-Fan Feng. LCD motion blur analysis, perception, and reduction using synchronized backlight flashing. In *Human Vision and Electronic Imaging XI*, pages M1–14. SPIE Vol. 6057, 2006.

James A Ferwerda, Peter Shirley, Sumanta N Pattanaik, and Donald P Greenberg. A model of visual masking for computer graphics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 143–152, 1997.

John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016.

Martin Fuchs, Tongbo Chen, Oliver Wang, Ramesh Raskar, Hans-Peter Seidel, and Hendrik P.A. Lensch. Real-time temporal shaping of high-speed video streams. *Computers & Graphics*, 34:575–584, 2010.

Alexander Goettker, Kevin J. MacKenzie, and T. Scott Murdison. Differences between oculomotor and perceptual artifacts for temporally limited head mounted displays. *Journal of the Society for Information Display*, 28(6):509–519, jun 2020. ISSN 1071-0922. doi: 10.1002/jsid.912. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/jsid.912`.

Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996.

E.D. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on Biomedical Engineering*, 53 (6):1124–1133, 2006. doi: 10.1109/TBME.2005.863952.

Rolf R Hainich and Oliver Bimber. *Displays: fundamentals & applications*. AK Peters/CRC Press, 2016.

Param Hanji, Fangcheng Zhong, and Rafał K. Mantiuk. Noise-aware merging of high dynamic range image stacks without camera calibration. In *Advances in Image Manipulation (ECCV workshop)*, pages 376–391. Springer, 2020. URL `http://www.cl.cam.ac.uk/research/rainbow/projects/noise-aware-merging/`.

Brittney Hartle and Laurie M Wilcox. Cue vetoing in depth estimation: Physical and virtual stimuli. *Vision Research*, 188:51–64, 2021.

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.

Janne Heikkila and Olli Silvén. A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pages 1106–1112. IEEE, 1997.

Gerben Johan Hekstra, Leo Jan Velthoven, and Michiel Adriaanszoon Klompenhouwer. Motion blur decrease in varying duty cycle, January 8 2008. US Patent 7,317,445.

David M Hoffman, Vasiliy I Karasev, and Martin S Banks. Temporal presentation protocols in stereoscopic displays: Flicker visibility, perceived motion, and perceived depth. *Journal of the Society for Information Display*, 19(3):271–297, 2011.

Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

Jonathan C Horton and William F Hoyt. The representation of the visual field in human striate cortex: a revision of the classic holmes map. *Archives of ophthalmology*, 109(6): 816–824, 1991.

Intel. Use variable rate shading (vrs) to improve the user experience in real-time game engines — siggraph 2019 technical sessions, 2019. URL `https://www.slideshare.net/IntelSoftware/use-variable-rate-shading-vrs-to-improve-the-user-experience-in-real-time-game-eng`

Aaron Isaksen, Leonard McMillan, and Steven J Gortler. Dynamically reparameterized light fields. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 297–306, 2000.

Adrian Jarabo, Tom Van Eyck, Veronica Sundstedt, Kavita Bala, Diego Gutierrez, and Carol O'Sullivan. Crowd light: Evaluating the perceived fidelity of illuminated dynamic scenes. In *Computer Graphics Forum*, volume 31, pages 565–574. Wiley Online Library, 2012.

Paul V Johnson, Joohwan Kim, David M Hoffman, Andy D Vargas, and Martin S Banks. Motion artifacts on 240-hz oled stereoscopic 3d displays. *Journal of the Society for Information Display*, 22(8):393–403, 2014.

Kimmo P Jokinen and Wen Nivala. 65-4: novel methods for measuring vr/ar performance factors from oled/lcd. In *SID Symposium Digest of Technical Papers*, volume 48, pages 961–964. Wiley Online Library, 2017.

Deane B Judd, David L MacAdam, Günter Wyszecki, HW Budde, HR Condit, ST Henderson, and JL Simonds. Spectral distribution of typical daylight as a function of correlated color temperature. *Josa*, 54(8):1031–1040, 1964.

Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Multidimensional Knapsack Problems*, pages 235–283. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-24777-7. doi: 10.1007/978-3-540-24777-7_9. URL `https://doi.org/10.1007/978-3-540-24777-7_9`.

D. H. Kelly. Motion and vision II Stabilized spatio-temporal threshold surface. *Journal of the Optical Society of America*, 69(10):1340, oct 1979. ISSN 0030-3941. doi: 10.1364/JOSA.69.001340. URL `https://www.osapublishing.org/abstract.cfm?URI=josa-69-10-1340`.

Minjung Kim, Maryam Azimi, and Rafał K Mantiuk. Perceptually motivated model for predicting banding artefacts in high-dynamic range images. In *Color and Imaging Conference*, volume 2020, pages 42–48. Society for Imaging Science and Technology, 2020.

Manuel Kraemer. Accelerating your vr games with vrworks. In *NVIDIAs GPU Technology Conference (GTC)*, 2018.

Yoshihiko Kuroki, Tomohiro Nishi, Seiji Kobayashi, Hideki Oyaizu, and Shinichi Yoshimura. A psychophysical study of improvements in motion-image quality by using high frame rates. *Journal of the Society for Information Display*, 15(1):61–68, 2007.

Douglas Lanman, Gordon Wetzstein, Matthew Hirsch, Wolfgang Heidrich, and Ramesh Raskar. Polarization fields: dynamic light field display using multi-layer lcds. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, pages 1–10, 2011.

James Larimer, Jennifer Gille, and James Wong. 41.2: Judder-induced edge flicker in moving objects. In *SID Symposium Digest of Technical Papers*, volume 32, pages 1094–1097. Wiley Online Library, 2001.

Chang Ha Lee, Amitabh Varshney, and David W Jacobs. Mesh saliency. *ACM transactions on graphics (TOG)*, 24(3):659–666, 2005.

Haebom Lee and Piotr Didyk. Real-time apparent resolution enhancement for head-mounted displays. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):1–15, 2018.

Gordon E Legge and John M Foley. Contrast masking in human vision. *Josa*, 70(12): 1458–1471, 1980.

Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996.

Chenglin Li, Laura Toni, Junni Zou, Hongkai Xiong, and Pascal Frossard. Delay-power-rate-distortion optimization of video representations for dynamic adaptive streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(7):1648–1664, 2017.

Dingquan Li, Tingting Jiang, and Ming Jiang. Unified quality assessment of in-the-wild videos with mixed datasets training. *International Journal of Computer Vision*, 129(4): 1238–1257, 2021.

Zhengqi Li, Wenqi Xian, Abe Davis, and Noah Snavely. Crowdsampling the plenoptic function. In *European Conference on Computer Vision*, pages 178–196. Springer, 2020.

Jingyu Liu, Akshay Jindal, Claire Mantel, Søren Forchhammer, and Rafał K Mantiuk. How bright should a virtual object be to appear opaque in optical see-through ar? In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2022.

Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. URL `https://arxiv.org/abs/1904.01786`.

Margaret Livingstone and David Hubel. Segregation of form, color, movement, and depth: anatomy, physiology, and perception. *Science*, 240(4853):740–749, 1988.

Cheng-Hung Lo, Chih-Hsing Chu, Kurt Debattista, and Alan Chalmers. Selective rendering for efficient ray traced stereoscopic images. *The Visual Computer*, 26(2):97–107, 2010.

S Lofgren, J Thaung, and C Lopes. Laser pointers and eye injuries: An analysis of reported cases, 2013 swedish radiation safety authority report number: 30, issn: 2000-0456, 2017.

Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019.

Peter Longhurst, Kurt Debattista, and Alan Chalmers. A gpu based saliency map for high-fidelity selective rendering. In *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 21–29. ACM, 2006.

David Luebke and Benjamin Hallen. Perceptually driven simplification for interactive rendering. In *Eurographics Workshop on Rendering Techniques*, pages 223–234. Springer, 2001.

David Luebke, Martin Reddy, Jonathan D Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. *Level of detail for 3D graphics*. Morgan Kaufmann, 2003.

Kevin J MacKenzie, David M Hoffman, and Simon J Watt. Accommodation to multiple-focal-plane displays: Implications for improving stereoscopic displays and for accommodation control. *Journal of vision*, 10(8):22, jan 2010. ISSN 1534-7362. doi: 10.1167/10.8.22. URL `http://www.ncbi.nlm.nih.gov/pubmed/20884597`.

Alex Mackin, Katy C Noland, and David R Bull. The visibility of motion artifacts and their effect on motion quality. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2435–2439. IEEE, 2016.

Rafał Mantiuk, Scott Daly, and Louis Kerofsky. Display adaptive tone mapping. In *ACM SIGGRAPH 2008 papers*, pages 1–10. 2008.

Rafał K Mantiuk, Anna Tomaszewska, and Radosław Mantiuk. Comparison of four subjective methods for image quality assessment. In *Computer graphics forum*, volume 31, pages 2478–2491. Wiley Online Library, 2012.

Rafał K. Mantiuk, Gyorgy Denes, Alexandre Chapiro, Anton Kaplanyan, Gizem Rufo, Romain Bachy, Trisha Lian, and Anjul Patney. FovVideoVDP : A visible difference predictor for wide field-of-view video. *ACM Transaction on Graphics*, 40(4):49, 2021. doi: 10.1145/3450626.3459831.

Judith McLean, S Raab, and LA Palmer. Contribution of linear mechanisms to the specification of local motion by simple cells in areas 17 and 18 of the cat. *Visual neuroscience*, 11(2):271–294, 1994.

Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 39–46, 1995.

Daniele Menon, Stefano Andriani, and Giancarlo Calvagno. Demosaicing with directional filtering and a posteriori decision. *IEEE Transactions on Image Processing*, 16(1): 132–141, 2006. URL https://doi.org/10.1109/TIP.2006.884928.

Aliaksei Mikhailiuk, Clifford Wilmot, Maria Perez-Ortiz, Dingcheng Yue, and Rafal Mantiuk. Active sampling for pairwise comparisons via approximate message passing and information gain maximization. In *2020 IEEE International Conference on Pattern Recognition (ICPR)*, Jan 2020.

Mikrora. Labeled eye diagram, Jun 2019. URL https://www.mikrora.com/labeled-eye-diagram/.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020.

Karol Myszkowski. The visible differences predictor: Applications to global illumination problems. In *Eurographics Workshop on Rendering Techniques*, pages 223–236. Springer, 1998.

Karol Myszkowski. Perception-based global illumination, rendering, and animation techniques. In *Proceedings of the 18th spring conference on Computer graphics*, pages 13–24, 2002.

Karol Myszkowski, Przemyslaw Rokita, and Takehiro Tawara. Perceptually-informed accelerated rendering of high quality walkthrough sequences. In *Rendering Symposium*, page 5–18, 1999.

Fernando Navarro, Susana Castillo, Francisco J Serón, and Diego Gutierrez. Perceptual considerations for motion blur rendering. *ACM Transactions on Applied Perception (TAP)*, 8(3):1–15, 2011.

Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020.

NIVIDA. Nvidia omniverse, 2021.

Nvidia. Nvidia turing gpu architecture. 2018. URL https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf.

Oculus. Half dome 3. https://www.oculus.com/blog/half-dome-updates-frl-explores-more-comfortable-compact-vr-prototypes-for-work/, 2020.

A. Ortega and K. Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*, 15(6):23–50, 1998. doi: 10.1109/79.733495.

Ryan S Overbeck, Daniel Erickson, Daniel Evangelakos, Matt Pharr, and Paul Debevec. A system for acquiring, processing, and rendering panoramic light field stills for virtual reality. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.

Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):1–12, 2016.

Fabio Pellacini. User-configurable automatic shader simplification. *ACM Transactions on Graphics (TOG)*, 24(3):445–452, 2005.

Maria Perez-Ortiz and Rafal K. Mantiuk. A practical guide and software for analysing pairwise comparison experiments. *arXiv preprint*, dec 2017. URL `http://arxiv.org/abs/1712.03686`.

Daniel Pohl, Timo Bolkart, Stefan Nickels, and Oliver Grau. Using astigmatism in wide angle hmds to improve rendering. In *2015 IEEE Virtual Reality (VR)*, pages 263–264, 2015. doi: 10.1109/VR.2015.7223396.

Dale Purves, G Augustine, D Fitzpatrick, L Katz, A LaMantia, J McNamara, and S Williams. Types of eye movements and their functions. *Neuroscience 2nd edition. Sunderland (MA) Sinauer Associates*, 2001.

Ganesh Ramanarayanan, James Ferwerda, Bruce Walter, and Kavita Bala. Visual equivalence: towards a new standard for image fidelity. In *ACM Transactions on Graphics (TOG)*, volume 26, page 76. ACM, 2007.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

Mark Rejhon. Strobe crosstalk: Blur reduction double-images, 2017. URL `https://blurbusters.com/faq/advanced-strobe-crosstalk-faq/`.

JE Roberts and AJ Wilkins. Flicker can be perceived during saccades at frequencies in excess of 1 khz. *Lighting Research & Technology*, 45(1):124–132, 2013.

Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. " grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23 (3):309–314, 2004.

Jan Schmid, Y ULUDAG, and J DELIGIANNIS. It just works: Raytraced reflections in" battlefield v". In *GPU Technology Conference*, 2019.

Jim Schmitz. Color blindness simulation research, Aug 2016. URL `https://ixora.io/projects/colorblindness/color-blindness-simulation-research/`.

T. Scott Murdison, Christopher McIntosh, James Hillis, and Kevin J. MacKenzie. 3-1: Psychophysical evaluation of persistence- and frequency-limited displays for virtual and augmented reality. *SID Symposium Digest of Technical Papers*, 50(1):1–4, 2019. doi: https://doi.org/10.1002/sdtp.12840. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/sdtp.12840`.

Helge Seetzen, Wolfgang Heidrich, Wolfgang Stuerzlinger, Greg Ward, Lorne Whitehead, Matthew Trentacoste, Abhijeet Ghosh, and Andrejs Vorozcovs. High dynamic range display systems. *ACM Trans. Graph.*, 23(3):760–768, August 2004a. ISSN 0730-0301. doi: 10.1145/1015706.1015797. URL `https://doi.org/10.1145/1015706.1015797`.

Helge Seetzen, Wolfgang Heidrich, Wolfgang Stuerzlinger, Greg Ward, Lorne Whitehead, Matthew Trentacoste, Abhijeet Ghosh, and Andrejs Vorozcovs. High dynamic range display systems. In *ACM SIGGRAPH 2004 Papers*, pages 760–768. 2004b.

Mark Segal and Kurt Akeley. *The OpenGL Graphics System: A Specification*. The Khronos Group, 2022.

BT Series. The present state of ultra-high definition television. 2020.

Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 231–242, 1998.

Kyle Simek. Dissecting the camera matrix, part 1: Extrinsic/intrinsic decomposition, 2012. URL `https://ksimek.github.io/2012/08/14/decompose/`.

Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019a.

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019b.

AAS Sluyterman. What is needed in lcd panels to achieve crt-like motion portrayal? *Journal of the Society for Information Display*, 14(8):681–686, 2006.

Steven W Smith et al. The scientist and engineer's guide to digital signal processing, 1997.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006.

Mittanamalli S Sridhar. Anatomy of cornea and ocular surface. *Indian journal of ophthalmology*, 66(2):190, 2018.

M. Stengel, P. Bauszat, M. Eisemann, E. Eisemann, and M. Magnor. Temporal video filtering and exposure control for perceptual motion blur. *Visualization and Computer Graphics, IEEE Transactions on*, 21(2):1–11, 2015. doi: 10.1109/TVCG.2014.2377753.

Jordan W Suchow and George A Alvarez. Motion silences awareness of visual change. *Current Biology*, 21(2):140–143, 2011.

K. Sung, A. Pearce, and C. Wang. Spatial-Temporal Antialiasing. *IEEE Transactions on Visualization and Computer Graphics*, 8(2):144–153, 2002.

Richard Szeliski. *Computer vision: algorithms and applications 2nd Edition*. Springer Science & Business Media, 2021.

Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 517–524. IEEE, 1998.

Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020.

Okan Tarhan Tursun, Elena Arabadzhiyska, Marek Wernikowski, Radosław Mantiuk, Hans-Peter Seidel, Karol Myszkowski, and Piotr Didyk. Luminance-contrast-aware foveated rendering. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH)*, 38(4), 2019.

Unity. Unity - manual: Dynamic resolution, 2021. URL `https://docs.unity3d.com/Manual/DynamicResolution.html`.

Unreal, 2021. URL `https://docs.unrealengine.com/en-US/RenderingAndGraphics/DynamicResolution/index.html`.

Karthik Vaidyanathan, Robert Toth, Marco Salvi, Solomon Boulos, and Aaron E Lefohn. Adaptive image space shading for motion and defocus blur. In *High Performance Graphics*, pages 13–21, 2012.

Karthik Vaidyanathan, Marco Salvi, Robert Toth, Tim Foley, Tomas Akenine-Möller, Jim Nilsson, Jacob Munkberg, Jon Hasselgren, Masamichi Sugihara, Petrik Clarberg, et al. Coarse pixel shading. In *Proceedings of High Performance Graphics*, pages 9–18. 2014.

Dmitriy Vatolin, Aleksandr Gushchin, Maxim Smirnov, Anastasia Antsiferova, and Eugene Lyapustin. Msu video quality metrics benchmark 2021 methodology, 2022. URL `https://videoprocessing.ai/benchmarks/video-quality-metrics_both.html`.

Tom Verbeure, Gerrit A Slavenburg, Thomas F Fox, Robert Jan Schutten, Luis Mariano Lucas, and Marcel Dominicus Janssens. System, method, and computer program product for combining low motion blur and variable refresh rate in a display, September 26 2017. US Patent 9,773,460.

Bruce Walter, Sumanta N Pattanaik, and Donald P Greenberg. Using perceptual texture masking for efficient image synthesis. In *Computer Graphics Forum*, volume 21, pages 393–399. Wiley Online Library, 2002.

Brian A Wandell. *Foundations of vision*. Sinauer Associates, 1995.

Andrew B Watson. Visual detection of spatial contrast patterns: Evaluation of five simple models. *Optics Express*, 6(1):12–33, 2000.

Andrew B Watson. High frame rates and human vision: A view through the window of visibility. *SMPTE Motion Imaging Journal*, 122(2):18–32, 2013.

Andrew B Watson and Albert J Ahumada. Blur clarified: A review and synthesis of blur discrimination. *Journal of Vision*, 11(5):10–10, 2011.

Andrew B Watson and Joshua A Solomon. Model of visual contrast gain control and pattern masking. *JOSA A*, 14(9):2379–2391, 1997.

Martin Weier, Michael Stengel, Thorsten Roth, Piotr Didyk, Elmar Eisemann, Martin Eisemann, Steve Grogorick, André Hinkenjann, Ernst Kruijff, Marcus Magnor, et al. Perception-driven accelerated rendering. In *Computer Graphics Forum*, volume 36, pages 611–643. Wiley Online Library, 2017.

Gordon Wetzstein, Douglas R Lanman, Matthew Waggener Hirsch, and Ramesh Raskar. Tensor displays: compressive light field synthesis using multilayer displays with directional backlighting. 2012.

Laurie M Wilcox, Robert S Allison, John Helliker, Bert Dunk, and Roy C Anthony. Evidence that viewers prefer higher frame-rate film. *ACM Transactions on Applied Perception (TAP)*, 12(4):1–12, 2015.

Lance Williams. Pyramidal parametrics. In *Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, pages 1–11, 1983.

Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021.

Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296, 2000.

Himanshu Yadav and B Annappa. Adaptive gpu resource scheduling on virtualized servers in cloud gaming. In *2017 Conference on Information and Communication Technology (CICT)*, pages 1–6. IEEE, 2017.

Bailin Yang, Frederick WB Li, Xun Wang, Mingliang Xu, Xiaohui Liang, Zhaoyi Jiang, and Yanhui Jiang. Visual saliency guided textured model simplification. *The Visual Computer*, 32(11):1415–1432, 2016.

Lei Yang, Dmitry Zhdan, Emmett Kilgariff, Eric B Lum, Yubo Zhang, Matthew Johnson, and Henrik Rydgård. Visually lossless content and motion adaptive shading in games. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(1):1–19, 2019.

Hector Yee, Sumanta Pattanaik, and Donald P. Greenberg. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Trans. Graph.*, 20(1):39–65, 2001.

John I Yellott Jr. Spectral analysis of spatial sampling by photoreceptors: topological disorder prevents aliasing. *Vision research*, 22(9):1205–1210, 1982.

Wenjun Zeng, Scott Daly, and Shawmin Lei. An overview of the visual optimization tools in jpeg 2000. *Signal Processing: Image Communication*, 17(1):85–104, 2002.

Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. doi: 10.1109/34.888718.

Fangcheng Zhong, Akshay Jindal, Ali Özgür Yöntem, Param Hanji, Simon J. Watt, and Rafał K. Mantiuk. Reproducing reality with a high-dynamic-range multi-focal stereo display. *ACM Trans. Graph.*, 40(6), dec 2021. ISSN 0730-0301. doi: 10.1145/3478513.3480513. URL https://doi.org/10.1145/3478513.3480513.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

Appendix A

Downsampling distortions

One of the most common use cases of image quality metrics in computer graphics is as a loss function in an optimisation problem such as image enhancement or restoration algorithms. When the loss function satisfies certain mathematical properties like monotonicity and differentiability, it enables the optimisation to converge onto a good solution. However, I found that when it comes to measuring the quality of images resampled using nearest neighbour interpolation (a common operation in computer graphics), existing quality metrics do not satisfy the aforementioned properties. An example can be found in Figure A.1, which plots quality predictions from several popular quality metrics for two images that have been downsampled to a lower resolution for a range of scaling factors. All metrics except CaMoJAB give non-smooth and non-monotonic predictions, making it hard for most optimisation methods to converge to a globally optimal solution. While CaMoJAB cannot guarantee differentiability due to the use of Dirac delta function in Eq. (5.11), it can be proven that CaMoJAB predictions monotonically decrease with decreasing image resolution.

Proposition: *CaMoJAB's predicted quality always decreases with decreasing image resolution*

Proof. We can safely ignore the temporal distortion component (d_t) of CaMoJAB because we are only considering stationary images. The spatial distortion ($d_\mathrm{s} = E_\mathrm{s}(I^\mathrm{ref}) - E_\mathrm{s}(I'')$) component, as defined in Eq. (5.21), is the difference in visual energy (powered summation) of the reference image and the low resolution distorted image. Since a reference image can be arbitrarily defined as any higher resolution version of the distorted image, we can prove the proposition by showing that d_s is always ≥ 0.

Let us consider the frequency spectrum of a 1D reference image after a discrete Fourier transform

$$A_\mathrm{ref} = \{a_0, a_1, \cdots, a_N\} \tag{A.1}$$

where a_i represents the amplitude of the i^th frequency, N corresponds to the Nyquist frequency, and $\{\cdots\}$ denote a finite sequence. I only consider the positive frequencies but the following analysis can be trivially extended to negative frequencies as the signal is

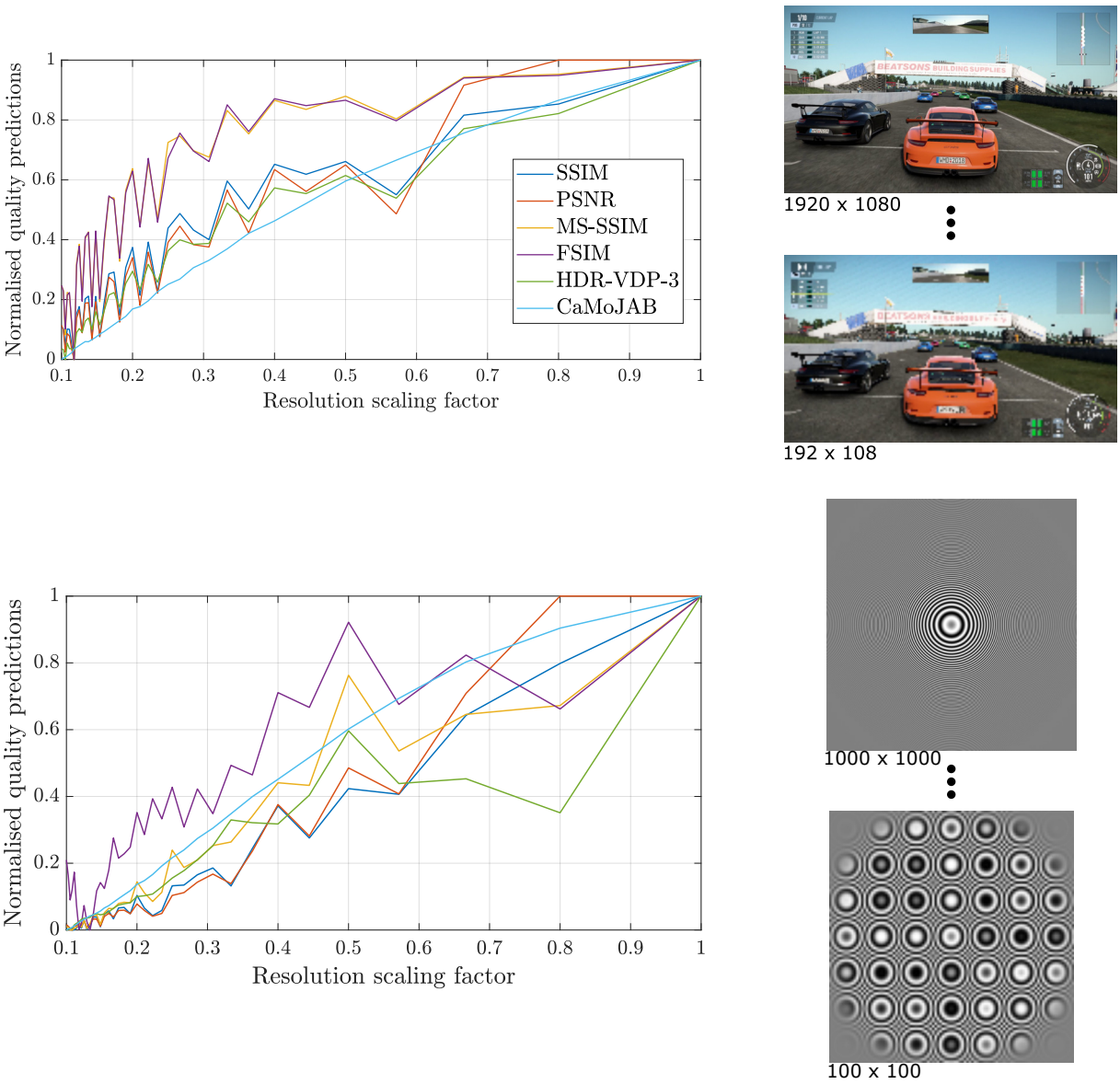


Figure A.1: Image quality predictions on downsampling image artefacts. All metrics except CaMoJAB behaves in a non-smooth and non-monotonic manner

symmetric in both directions. The visual energy of this signal is defined as

$$E(A_{\text{ref}}) = \sum_{i=0}^{N} |a_i|^{\beta} \tag{A.2}$$

where β is the power parameter fitted to experimental data (Section 5.4). The distorted image is derived from the reference image using only two types of operations: blurring (Eq. (5.8), Eq. (5.18)) and downsampling (Eq. (5.12)). Let us see how both of these operations result in a loss of energy.

Since a blur operation is a multiplication with a normalised filter in the frequency domain, the amplitude spectrum of a blurred signal will be

$$A_{\text{blur}} = \{a'_0, a'_1, \cdots, a'_N\} \mid a'_i \leq a_i. \tag{A.3}$$

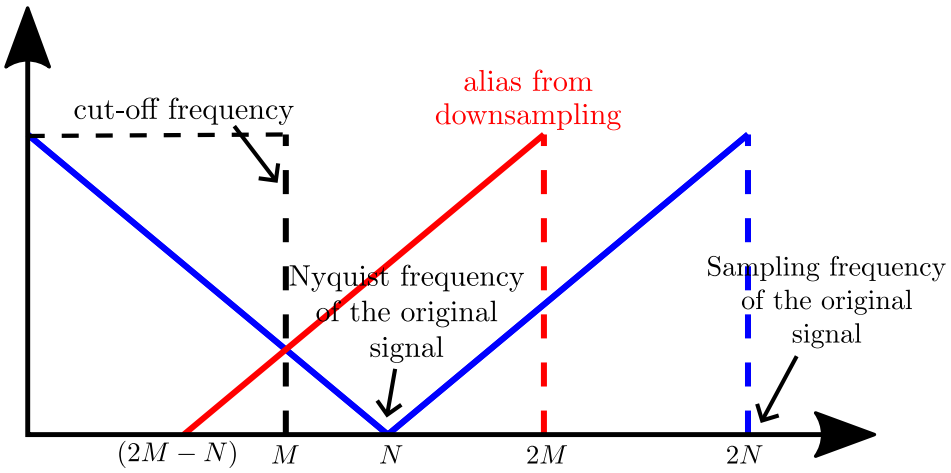


Figure A.2: Consider for example a 30 cpd ($= N$) signal displayed on a 40 ppd display. The maximum frequency the display can present (called the cut-off frequency) is 20 cpd ($=M$).

The energy of the blurred signal is

$$E(A_{\text{blur}}) = \sum_{i=0}^{N} |a_i|'^{\beta} \tag{A.4}$$

It is trivial to see that $E(A_{\text{ref}}) - E(A_{\text{blur}}) \geq 0$.

Now, let's look at downsampling. Sampling at a frequency less than the two times the Nyquist frequency of A_{ref} i.e. $2N$ will lead to a replica of the signal to fold onto itself as illustrated in Figure A.2.

Since the folding is symmetric, the downsampled signal will be

$$A_{\text{down}} = \{a_0, a_1, \cdots, a_{2M-N-1}, (a_{2M-N} + a_N), (a_{2M-N+1} + a_{N-1}), \cdots, a_M\} \tag{A.5}$$

The energy of the downsampled and reference signal is given by

$$E(A_{\text{down}}) = |a_0|^{\beta} + |a_1|^{\beta} + \cdots + |a_{2M-N-1}|^{\beta} + (|a_{2M-N} + a_N|)^{\beta} + \cdots + |a_M|^{\beta} \tag{A.6}$$

$$E(A_{\text{ref}}) = |a_0|^{\beta} + |a_1|^{\beta} + \cdots + |a_{2M-N-1}|^{\beta} + |a_{2M-N}|^{\beta} + \cdots + |a_N|^{\beta} \tag{A.7}$$

Subtracting and rearranging,

$$\begin{aligned} E(A_{\text{ref}}) - E(A_{\text{down}}) = {} & \left(\left(|a_{2M-N}|^{\beta} + |a_N|^{\beta}\right) - \left(|a_{2M-N} + a_N|\right)^{\beta}\right) + \\ & \left(\left(|a_{2M-N+1}|^{\beta} + |a_{N-1}|^{\beta}\right) - \left(|a_{2M-N+1} + a_{N-1}|\right)^{\beta}\right) + \cdots \end{aligned} \tag{A.8}$$

It can be shown that $\left(\sum |x_i|\right)^{\gamma} \leq \sum |x_i|^{\gamma} \;\; \forall\, \gamma \in [0, 1]$ (Proof in the following paragraph.) Hence, $E(A_{\text{ref}}) - E(A_{\text{down}}) \geq 0$.

Since the distorted image is derived by downsampling or blurring the reference image, and both of these operations result in a loss of energy, the energy of the reference image is always greater than the energy of the distorted image. Therefore, CaMoJAB's predicted quality monotonically decreases with decreasing image resolution.

Lemma: $(\sum_{i=0}^{n} |x|)^{\gamma} \leq \sum_{i=0}^{n} |x|^{\gamma} \quad \forall \quad \gamma \in [0,1]$

Proof. Let $\lambda = \sum_{i=0}^{n} |x_i|$. When $\lambda = 0$, all $x_i = 0$ and the inequality trivially holds. For $\lambda > 0$, let's define

$y_i = \frac{|x_i|}{\lambda}$.

$\because y_i \in [0,1]$ and $\gamma \in [0,1] \quad \Rightarrow \quad y_i^{\gamma} \geq y_i$

Together with $\sum_{i=0}^{n} y_i = 1$, we get

$\sum_{i=0}^{n} |x_i|^{\gamma} = \lambda^{\gamma} \sum_{i=0}^{n} y_i^{\gamma} \geq \lambda^{\gamma} \sum_{i=0}^{n} y_i = \lambda^{\gamma} = (\sum_{i=0}^{n} |x_i|)^{\gamma}$

Appendix B

Approximations in CaMoJAB

While developing CaMoJAB in Chapter 5, I made a few approximations in different stages considering its application to real-time rendering. In this chapter, I provide some additional results to motivate those choices.

Using Lorentzian to approximate sinc blur filter In Eq. (5.8) and Eq. (5.16), we model mipmap and eye-motion blur using box filters in spatial domain. The Fourier transform of a box filter is a sinc function. I proposed to instead model blur with Lorentzian function which is monotonic and well approximates sinc function (Figure 5.3). The motivation behind this approximation was the application of CaMoJAB to solve the optimisation problem defined in Eq. (6.1) which determines the optimal refresh rate for a given velocity and image. Figure B.1 provides an example optimisation solution on John Lemon's Haunted Jaunt scene (Figure 6.5) with the blur filter modelled both as a sinc and Lorentzian function. Sinc gives unstable non-monotonic predictions which can result in unpleasant motion artefacts in practice. In comparison, using Lorentzian always gives monotonic results. The sharp transitions in refresh rate are usually not noticeable in practice because of the predictive nature of G-Sync (Section 2.4) which act as a smoothing operation.

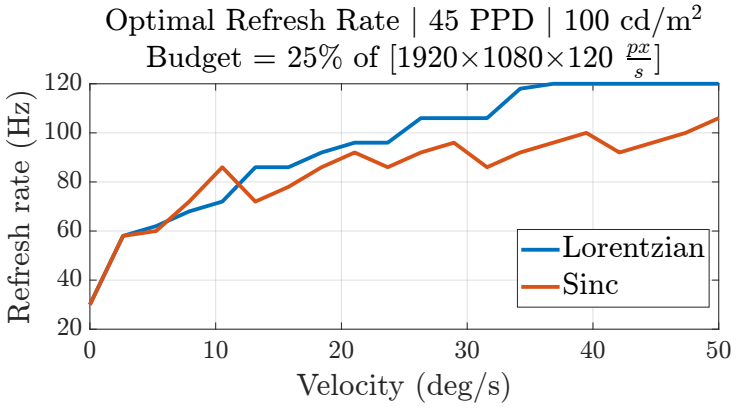


Figure B.1: Optimal refresh rate for haunted house scene with blur filter modelled as either Lorentzian or sinc. Using Lorentzian leads to more stable optimisation. The sharp transitions are due to integer refresh rates and low number of velocity sample points.

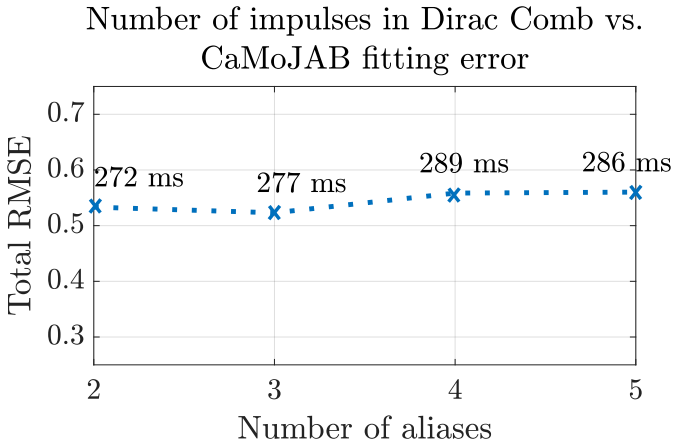


Figure B.2: Number of impulses in Dirac comb (downsampling operation) does not have a large effect on CaMoJAB performance.

Number of modelled aliases vs fitting error VRS downsamples the texture images which can be represented by a Dirac comb function (impulse train) in Fourier domain. A Dirac comb function is an infinite series of Dirac delta functions spaced at intervals of sampling frequency. Convolution with Dirac comb creates replicas of the signal (called aliases) at each impulse location. In practice, we can only use finite number of impulses while modelling Dirac comb. I found the number of replicas to not have much effect on CaMoJAB's prediction error (Figure B.2) on the calibration dataset (Section 5.4) and hence chose to use only the first two replicas to reduce the performance overhead and discontinuities introduced by this function.

Visual difference energy vs difference of visual energy A common practice to determine the perceived quality of a distorted signal is to calculate the difference between the CSF-normalised reference and distorted signal and take its energy as measure quality [Watson, 2000]. Another approach was proposed by Denes et al. [2020] where they show that the difference in visual energies of reference and distorted signals also make for a good measure of perceived quality. In CaMoJAB, I used the latter as I noticed in practice it gives stable and smoother results and satisfies some useful properties as described in Appendix A. Figure B.3 provides some example predictions of the model when trained with visual difference energy (VDE: $d_\mathrm{s} = E_\mathrm{s}(I^\mathrm{ref} - I'')$) vs. visual energy difference (CaMoJAB: $d_\mathrm{s} = E_\mathrm{s}(I^\mathrm{ref}) - E_\mathrm{s}(I'')$). While the results from both approaches follow the same general trends, VDE results are non-smooth and non-monotonic.

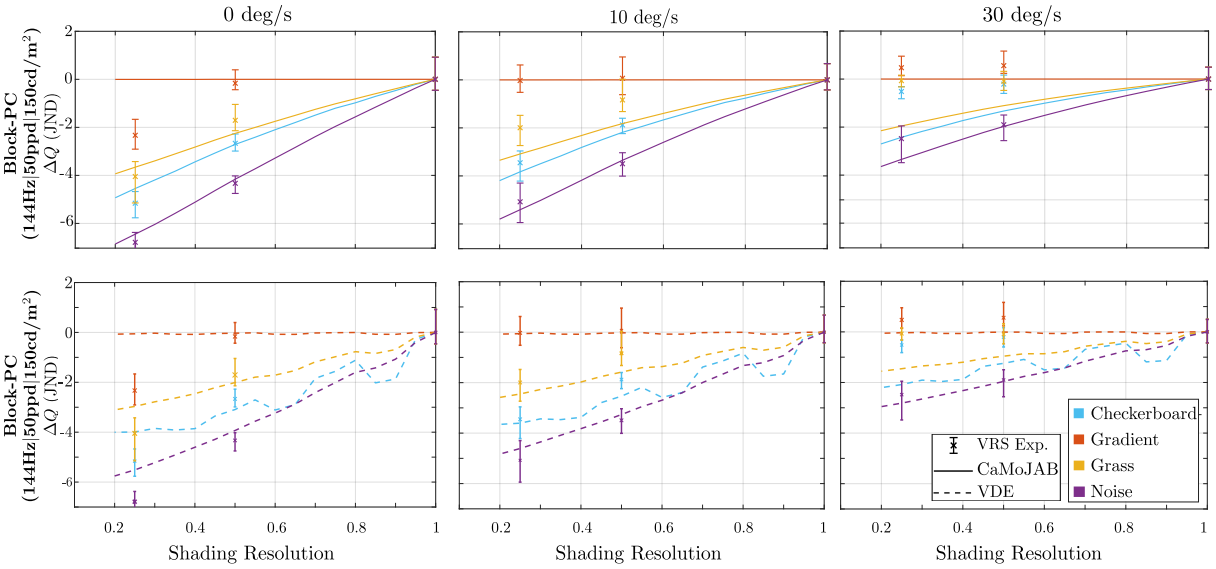


Figure B.3: Comparison of visual difference energy (VDE, bottom row) and visual energy difference (CaMoJAB, top row). The latter leads to more stable predictions.

Appendix C

Motion quality models: Additional Comparison

Here I provide additional plots for NAS and MARRR performance on our Experiment 1 data (Section 4.3) in Figure C.2 and Mackin et al. [2016] dataset in Figure C.1. MARRR, a perceptually motivated but content-independent model, fails to predict the change in quality due to texture in Experiment 1, but performs well on Mackin et al.'s dataset. While NAS predicts different quality for different textures in Experiment 1 and correctly captures the reduction in artefacts visibility with velocity, it does not predict the correct trend between the textures. It also cannot capture changes in quality due to the refresh rate in Mackin et al.'s dataset.

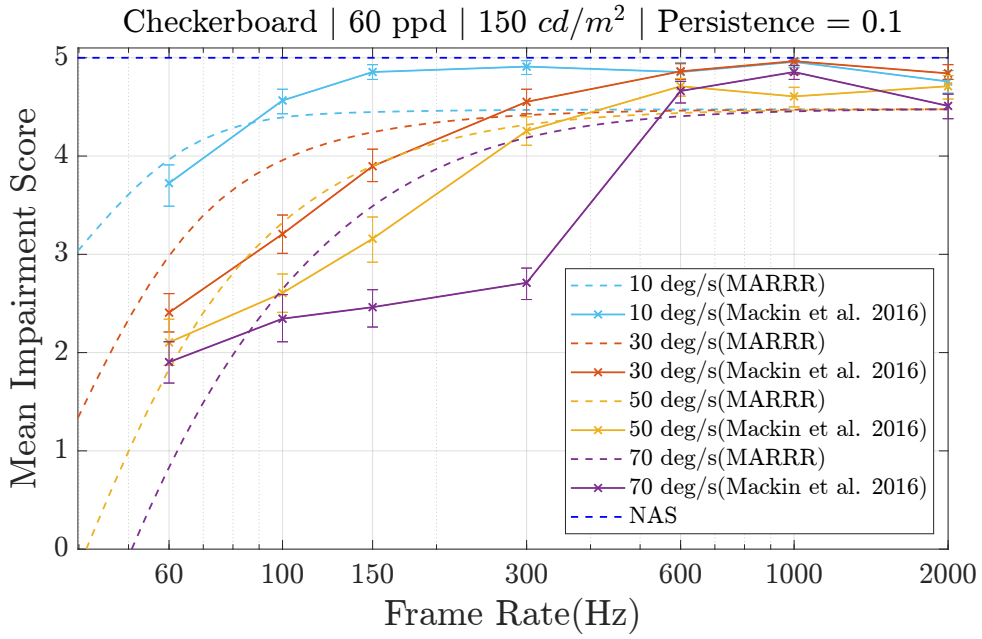


Figure C.1: NAS and MARRRR predictions on Mackin et al. [2016] data

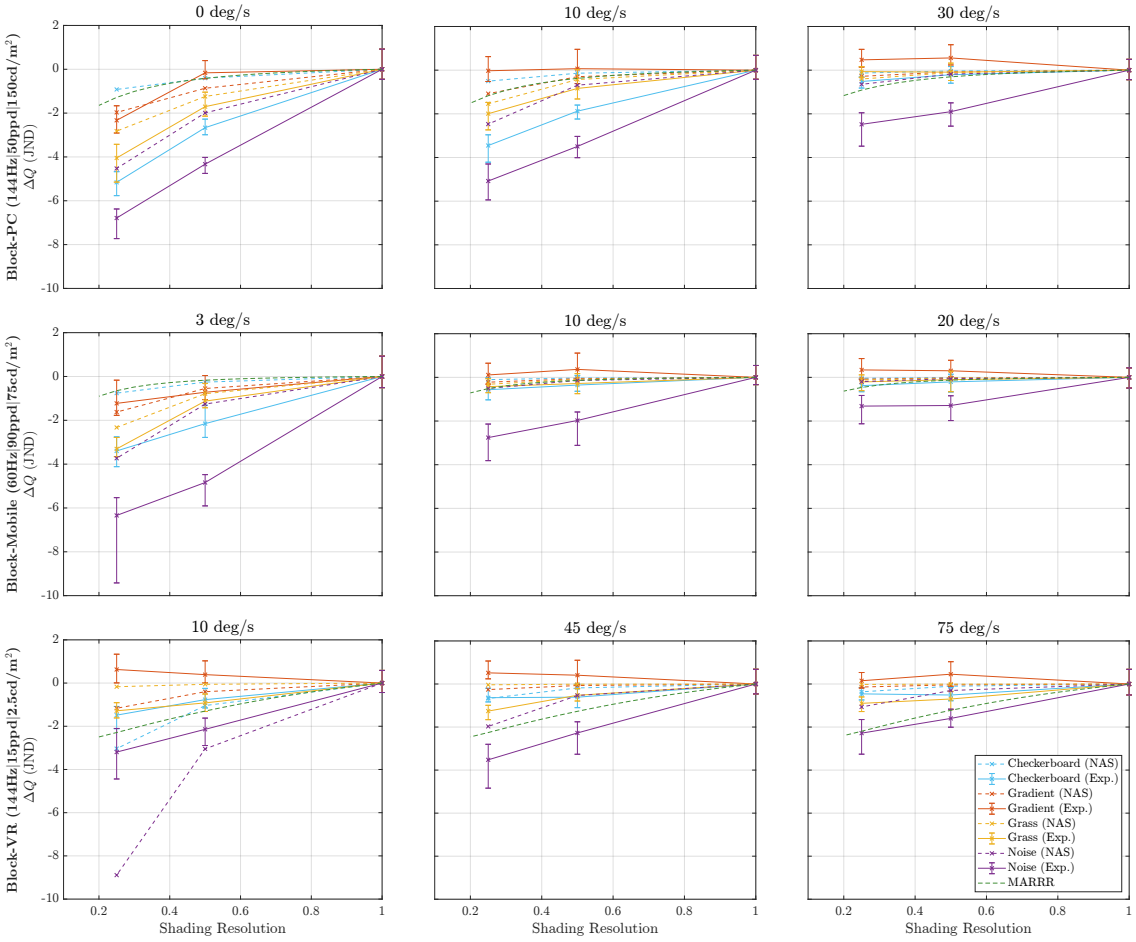


Figure C.2: NAS and MARRR predictions on our Experiment 1 results. Both models were linearly fitted to the data for fair comparison. MARRR is content independent and hence fails to capture change in quality dues to texture. NAS doesn't capture the correct trend between textures. Also, it can only predict quality for half and quarter shading resolution.