POLITECNICO DI TORINO
Repository ISTITUZIONALE

Cloud Gaming with Foveated Video Encoding

Original
Cloud Gaming with Foveated Video Encoding / KARAM ILLAHI, Gazi; VAN GEMERT, Thomas; Siekkinen, Matti; Masala, Enrico; Oulasvirta, Antti; YLÄ-JÄÄSK, Antti. - In: ACM TRANSACTIONS ON MULTIMEDIA COMPUTING, COMMUNICATIONS AND APPLICATIONS. - ISSN 1551-6857. - STAMPA. - 16:1(2020), pp. 1-24. [10.1145/3369110]

Availability:
This version is available at: 11583/2841252 since: 2022-10-09T17:00:11Z

Publisher:
ACM

Published
DOI:10.1145/3369110

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
ACM postprint/Author's Accepted Manuscript, con Copyr. autore

© KARAM ILLAHI, Gazi; VAN GEMERT, Thomas; Siekkinen, Matti; Masala, Enrico; Oulasvirta, Antti; YLÄ-JÄÄSK, Antti 2020. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in ACM TRANSACTIONS ON MULTIMEDIA COMPUTING, COMMUNICATIONS AND APPLICATIONS, http://dx.doi.org/10.1145/3369110.

(Article begins on next page)

17 April 2024

Cloud Gaming with Foveated Video Encoding

GAZI KARAM ILLAHI, Aalto University, Finland
THOMAS VAN GEMERT, Aalto University, Finland
MATTI SIEKKINEN, Aalto University and University of Helsinki, Finland
ENRICO MASALA, Politecnico di Torino, Italy
ANTTI OULASVIRTA, Aalto University, Finland
ANTTI YLÄ -JÄÄSKI, Aalto University, Finland

Cloud gaming enables playing high-end games, originally designed for PC or game console setups, on low end devices such as netbooks and smartphones, by offloading graphics rendering to GPU powered cloud servers. However, transmitting the high resolution video requires a large amount of network bandwidth, even though it is a compressed video stream. Foveated video encoding (FVE) reduces the bandwidth requirement by taking advantage of the non-uniform acuity of human visual system and by knowing where the user is looking. We have designed and implemented a system for cloud gaming with foveated encoding using a consumer grade real-time eye tracker and an open source cloud gaming platform. In this article, we describe the system and its evaluation through measurements with representative games from different genres to understand the effect of parameterization of the FVE scheme on bandwidth requirements and to understand its feasibility from the latency perspective. We also present results from a user study. The results suggest that it is possible to find a "sweet spot" for the encoding parameters so that the users hardly notice the presence of foveated encoding but at the same time the scheme yields most of the bandwidth savings achievable.

CCS Concepts: • **Information systems** → **Multimedia streaming**; • **Applied computing** → *Computer games*; • **Computing methodologies** → *Image compression*; • **Human-centered computing** → User studies.

Additional Key Words and Phrases: Cloud Gaming, Foveated Video Encoding, Adaptive Bitrate Encoding, Game Streaming, Gaze-Contingent Encoding

ACM Reference Format:
Gazi Karam Illahi, Thomas Van Gemert, Matti Siekkinen, Enrico Masala, Antti Oulasvirta, and Antti Ylä -Jääski. 2019. Cloud Gaming with Foveated Video Encoding. *ACM Trans. Multimedia Comput. Commun. Appl.* 1, 1, Article 1 (January 2019), 24 pages. https://doi.org/10.1145/3369110

1 INTRODUCTION

High-end gaming involves complex rendering of graphics, which is performed by dedicated GPU cards on PC and game console setups. The graphics processing power of low-end PCs as well as netbooks and smartphones is typically insufficient for high-end gaming. Cloud gaming makes high-end gaming possible on low-end devices by offloading graphics rendering to GPU powered

Authors' addresses: Gazi Karam Illahi, Aalto University, Espoo, 02150, Finland, gazi.illahi@aalto.fi; Thomas Van Gemert, Aalto University, Espoo, 02150, Finland, thomas.vangemert@aalto.fi; Matti Siekkinen, Aalto University and University of Helsinki, Espoo, Finland, matti.siekkinen@aalto.fi; Enrico Masala, Politecnico di Torino, Turin, Italy, enrico.masala@polito.it; Antti Oulasvirta, Aalto University, Espoo, Finland, antti.oulasvirta@aalto.fi; Antti Ylä -Jääski, Aalto University, Espoo, Finland, antti.yla-jaaski@aalto.fi.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
1551-6857/2019/1-ART1 $15.00
https://doi.org/10.1145/3369110

cloud servers. The server intercepts rendered scenes, encodes them into a video, and streams the video to a thin client, which decodes and plays the received video. The client also intercepts user input and relays it to the cloud server where it is replayed locally. This kind of approach allows remote game execution without having to modify the game or its underlying engine in any way.

Cloud gaming infrastructure must satisfy the following constraints that are critical to the Quality of Experience (QoE) provided by the service: 1) short enough end-to-end latency between user input and corresponding change in video frame, and 2) large enough amount of available bandwidth to stream high quality video from the remote server. The first one stems from the fact that user perceived latency between control input and visible action on display reduces the quality of user experience in thin client computing generally [43] as well as in cloud gaming specifically [11]. The second one arises from the need to stream sufficiently high quality video to the client device. The bitrate of a full HD video compressed using H.264 with typical settings ranges from 5 to 10 Mbps, but it can go up to tens of Mbps depending on the encoder settings and framerate. Increasing the resolution to 4K would generally boost the bitrate up by at least a factor of three, which is, at the time of writing, reaching the limit for most of the Internet users [5].

In this paper, we focus on the bandwidth challenge and propose to apply so called *foveated video encoding* (FVE) to reduce the bandwidth requirement in cloud gaming. The method takes the non-uniform acuity of the human visual system (HVS) into account when encoding video: the fovea is the region of the retina directly behind the eye lens and visual acuity of the eye is the highest in the fovea, dropping sharply with angular distance from the fovea [44]. Our approach is to encode game video rendered in the remote server with a quality gradient that spatially matches the acuity of HVS by tracking the gaze of the user and using the information during encoding. Encoding video in such a fashion can result in a significantly lower video bitrate compared to non-foveated encoding, hence reducing the bandwidth requirement of cloud gaming.

Foveated video encoding is a well researched concept, which has drawn renewed interest in recent years because of affordable, high quality gaze trackers on the market and low latency networking technologies that have made new application scenarios possible. The primary difference of our approach compared to recent related work on cloud gaming (e.g., [1, 28]) is that we apply real-time gaze tracking and foveated video coding to off-the-shelf games without need for game engine customization.

The main contribution of this paper is to demonstrate feasibility of cloud gaming with real-time foveated encoding, particularly for off-the-shelf games and using commodity solutions available to consumers today. We provide a prototype that is game agnostic and does not require any modifications to the underlying game engine. We also demonstrate that our approach of FVE can reduce the need for bandwidth in cloud gaming by several tens of percent without the user perceiving any degradation in video quality. To that end, we have conducted a user study to examine the effect of foveated video on the user experience using the prototype system. While adjusting parameters in the foveated video encoding, we asked the users to rate video quality and their engagement. The study reveals that a "sweet spot" exists and may be found with careful parametrization of FVE, which yields large bandwidth savings with hardly any degradation in user experience. We also evaluate the prototype system with different parameter values and different games considering gaze data and video bitrate.

The paper is structured as follows. In Section 2 we discuss the background and related work of this work including cloud gaming, foveation and foveated video coding. In Section 3, we describe the implementation of the prototype. In Section 4, we evaluate the system from bandwidth and latency perspectives and in Section 5 we describe our user study and its results. Finally we discuss the possible limitations and planned future work and we conclude the work in Section 6.

2 BACKGROUND AND RELATED WORK

2.1 Cloud Gaming

Cloud gaming applies the cloud computing paradigm to gaming, dividing the implementation of a gaming system between a remote server and a local thin client, on which a player plays the game [22, 39]. Typically, the cloud gaming server is deployed in a GPU equipped cloud or edge server such as a virtual machine (VM) or a container. The cloud gaming server runs the game engine, captures game play video rendered by the engine, encodes it and transmits it to the cloud gaming client. The cloud gaming client receives the game play video, decodes and renders it on screen. The client also captures user input, such as key presses or mouse/joystick movements and transmits them to the server, which relays the user input to the game engine. The user inputs appear local to the game engine. This architecture, shown in Figure 1, allows any off the shelf game to be played in a cloud gaming environment, allowing high quality gaming on even low-end devices and mobile phones with low compute and energy resources. Other approaches of cloud gaming exist as well: for example, rendering load may be dynamically shared between the thin client and the server [7]. Another approach is to render at the server, but instead of a conventional video stream, send video objects in BiFS to the client [28]. These approaches, however, require modifications to the game engine, which has to be done on a game by game basis and consequently cannot be used with off-the-shelf games. Further, support for object representation coding and decoding is sparse as compared to the ubiquitous software and hardware support that conventional video enjoys. A survey on cloud gaming and cloud gaming architectures can be found in [6]. For immersive QoE,

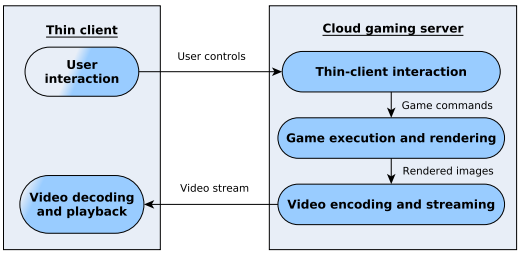


Fig. 1. Cloud gaming architecture [22],[15].

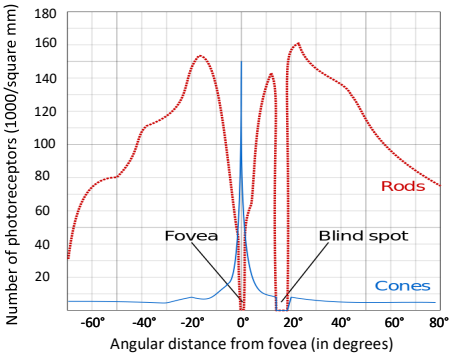


Fig. 2. Density of photoreceptors in the human eye [44].

the entire scheme has to be abstracted from the user, which in effect means two design constraints: there should be no observable delay (end to end) for the user and the video displayed should be of sufficiently high quality which translates to significant downstream bandwidth.

2.2 Foveation and Foveated Video Encoding (FVE)

The human visual system (HVS) has a non-uniform sampling response to visual stimuli, due to a phenomenon called foveation. Foveation is caused by the nature of distribution of photoreceptor cells in the human eye. There are two types of photoreceptors in our eye, rods and the cones, that are responsible for vision under low illumination and high illumination levels respectively [44]. The cones are involved in most activities like reading, gaming, etc. The density of photoreceptors in the retina is non-uniform, as shown in Figure 2. The cone density is highest at the fovea, which is

the region of retina directly behind the lens, within a range of 2° of the human visual field [37], and drops off sharply as the distance from the fovea increases. Consequently, the sampling response of a visual scene and, hence, the perceived resolution corresponds to the density of cones. Considering foveation, the practice of encoding a video frame with uniform spatial quality is wasteful.

It should be noted that foveation is just one of the phenomena in human vision and that there are various other pyscho-physical and pyscho-optical phenomena involved [26]. Further, gaze fixation and consequently foveation is directly dependent on head and eye movements, of which vestibulo-ocular reflexes, saccades and smooth pursuit are the most prominent. Vestibulo-ocular reflexes compensate for head movement when focusing on a point, saccades are fast eye movements in between fixations and smooth pursuit occurs when the eye is tracking a moving subject [41]. The interplay of foveation, eye movements and vision is an actively researched topic [38] and is outside of the scope of this work. Here foveation is considered as it occurs during the so called fixations [44]. Fixations occur when the eyes focus on a target after a saccade. Further, there are various eye movements involved in maintaining visual fixation, e.g., micro-saccades, ocular drifts and ocular micro-tremors [36] which may affect eye tracking and hence eye tracking based FVE.

An effective real-time streaming scheme optimizes the trade-off between delivered video quality and the available bandwidth by employing so called adaptive bitrate streaming. In adaptive bitrate streaming the bitrate of the streamed video, and hence its quality, is varied according to the available bandwidth. Typically, the change of quality is temporal, changing based on factors including network conditions. Foveated video streaming differs from traditional adaptive streaming in that the video quality may be changed spatially within a frame as well: encoding at highest quality where the user gaze is fixated or predicted to fixate and at lower quality elsewhere. This spatial rate adaptation, used concurrently with temporal rate adaptation can yield significant improvements in streaming efficiency with respect to available bandwidth and delivered QoE

FVE has been studied for quite some time, as noted earlier. A survey on the field was published by Wang et al. [45] about a dozen years ago. However, FVE has seen limited deployment, primarily due to the requirement of gaze information of each individual viewer for effective FVE. There are two approaches of determining the gaze location of a viewer, either by pre-analyzing the video for salient features where the user is likely to fixate their eyes or by tracking the gaze in real time. Analyzing video for salient features for foveated video coding has been an area of active research, see for example [19, 46]. The latter approach of using real time gaze location for foveated encoding has also drawn more interest with the availability of relatively economical non-invasive gaze tracking solutions. Most of the current approaches to FVE rely on tiling of the video frame. Zare et al. describe a solution for VR applications using HEVC compliant tiles in [47]. They partition 360° video into HEVC compliant tiles with different resolutions and display high quality video tiles within the users viewport. A similar solution is proposed by Qian et al. [33] for streaming panoramic video over wireless networks: streaming only the visible portion of the video. A foveated video streaming solution for video on demand is described by Ryoo et al. in [37], using real-time webcam-based gaze tracking. Their approach is based on partitioning the video into tiles and pre-coding videos into multiple resolution versions, streaming high resolution tiles at the gaze location and lower resolution tiles elsewhere. Similarly, [24] use an eye-tracker in an HMD to determine a user's gaze and send high resolution HEVC compliant tiles at that location, while using a single low resolution HEVC video for the background. The approach uses two separate video decoders at the client and the video is pre-encoded with different quality levels to make it suitable for on-demand 360° video applications. A gaze-aware video streaming solution for mobile devices is proposed in [31], wherein the video stream is stopped or reduced in quality when the user's gaze is away from the device, without altering the audio-stream. The proposed solution is shown to improve power consumption and bandwidth usage.

2.3 QoE

Quality of Experience (QoE) as a parameter is difficult to quantify. Although computational methods for assessing video quality have been around for a long time[2] (with some supporting foveated video), the current system and experiment is focused more on the end-user's subjective experience. Objective measures may prove helpful in assessing the video quality, but we believe that the overall QoE is best reported by the users themselves (similar to[17]). Many works have studied QoE in FVE and imaging. However, since a standard foveated encoding scheme does not exist, the parameters considered are different from the parameters we consider. Lungaro et al.[25] investigate the QoE of a similar FVE system in order to reduce the bandwidth requirements of high-resolution video. Their system defines a circular high quality foveal region,an annular region around the foveal region with transitional quality, and a background region with lower quality. The authors name the Round-Trip Time (RTT, i.e. end-to-end latency) as the main constraint in such systems, and explore different combinations of encoding parameters (size and quality of the foveal, annular and background regions) and network connection properties by employing a user study. The authors conclude that acceptable QoE may be achievable even with the current wireless networks with proper parametrization. Furthermore, they notice that at some point increasing the size of the foveal area i.e. the area with high quality does not further increase QoE, and that instead increasing the background resolution (quality of the peripheral area) is needed. Rai et al.[34, 35] explore the perceived video quality in foveated video systems, with a focus on artifacts in the peripheral area. In [34] experimental results indicate that non-flickering spatial artefacts in the peripheral region are less disruptive for the viewer than temporally flickering artefacts and also that the threshold of an artefact being disruptive is higher in visual periphery than in the foveal region. The authors highlight the need for consideration of supra-threshold effects of distortions in the peripheral areas in order to maintain a high QoE. In [35] the authors note that there may be a correlation between gaze disruptions and perceived video quality. By means of another user study the authors found a strong correlation (0.84) between gaze disruptions and DMOS (opinion scores).

In a cloud gaming system such as this, a user's QoE will be largely determined by the perceived video quality and perceived latency. How much so, especially with regard to latency, remains an area of active research. Some of our participants reported that latency issues were more disturbing than video issues. This is in line with results from other works like [12], but it should be noted that the extent of QoE deterioration due to delays in cloud and online gaming is highly dependent on the game genre and gameplay pace [21]. We refer the reader to [13, 29, 40] for further reading on latency and QoE.

2.4 Cloud Gaming and FVE

Two works that focus on foveated video for cloud gaming are [1] and [28]. Authors in [1] develop a game attention model based on both saliency of the gameplay video and and a game priority model which considers objects in the video frame based on likelihood of game player's attention. The proposed scheme is evaluated using pre-recorded video sequences. A solution to reduce downstream bandwidth required for cloud gaming using foveated encoding of graphics objects based on live gaze data is described in [28]. The solution described therein uses the MPEG-4 BiFS framework instead of conventional video encoding and requires hooks into the game engine to get game object data. The proposed scheme is evaluated with an experimental 2D game. These approaches require either substantial prior knowledge of the game or changes to game engine or both, hence they cannot be used with off the shelf games.

In our previous work [18], we develop a foveated video streaming solution for cloud gaming, wherein we use a gaze tracker to track the gaze of a player in real time and send the gaze information

to the cloud gaming server. At the cloud gaming server, we incorporate the gaze information in the gameplay video encoding scheme for foveated video encoding. The approach works with any off-the-shelf game and requires no modifications to the game engine. In [18], we make optimistic but cautious conclusions about the feasibility of FVE in cloud gaming and the potential of bandwidth savings without affecting QoE. In this work, we extend our previous work by validating our observations on feasibility of FVE and potential bandwidth savings with a user study.

3 CLOUD GAMING WITH FOVEATED VIDEO ENCODING

To implement FVE for cloud gaming, we use GamingAnywhere [15] as our cloud gaming platform. It is an open source, portable and extensible software comprising cloud gaming server and client implementations. Further, to track the gaze locations of a player we use an eye tracker on the client side. The GamingAnywhere server captures gameplay video, encodes it and streams it to

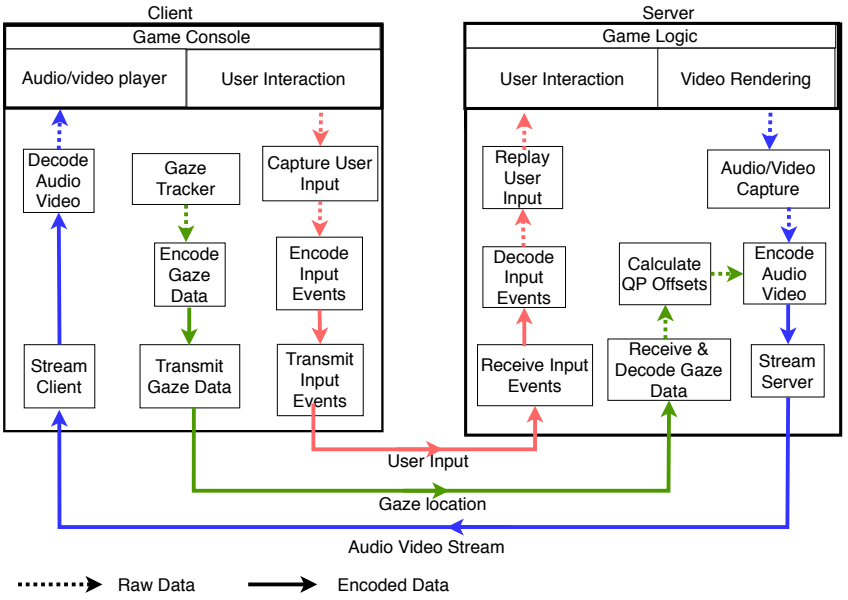


Fig. 3. Architectural overview of the prototype.

the GamingAnywhere client. It also receives user input from the client and replays the input to the game engine. In our prototype, it is modified to accept real-time gaze location data from the client. The gaze location is used by the server to encode video in a foveated fashion in real-time. The GamingAnywhere client receives gameplay video from the server, decodes it and renders it on-screen. It also captures user input actions, like key presses or mouse movements and forwards them to the GamingAnywhere server. In our prototype, a gaze tracker is installed on the client machine, which further tracks the user's gaze. The gaze location data is sent to the server as soon as it is available at the client. We use a Tobii 4C gaze tracker[1] to track the gaze and configure the GamingAnywhere server to use the *x264* encoder in adaptive quantization mode. Adaptive quantization allows us to change the quantization parameter on a per macroblock basis for each frame. The exact method of calculating the QPs is discussed in detail in 3.2. An architectural

[1] https://tobiigaming.com/eye-tracker-4c/

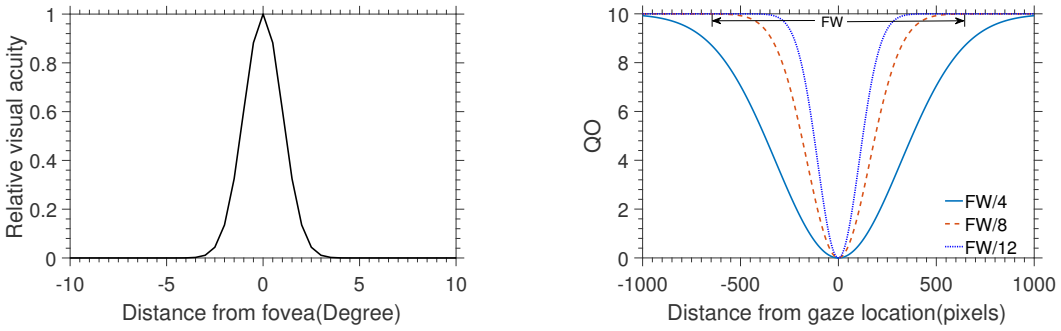
overview of the prototype is illustrated in Figure 3. The solid arrows indicate encoded data, for example, user input, video or gaze coordinates, while dotted arrows indicate decoded or raw data.

3.1 Gaze Tracker

The Tobii 4C Eye Tracker used at the client is an economical eye-tracking device directed towards gaming and human-computer interaction applications. It has an on-board ASIC that can track each eye and provides eye location, gaze location and other related data, invariant to head movements. We use the eye tracker in a "light filtering" mode where the gaze data is adaptively filtered considering both age and velocity of the reported gaze points in an attempt to filter out noise [42]. Noise in gaze data may include, for example, micro-saccades which happen when the eyes are trying to focus on a target. We make a design decision to use lightly filtered gaze data instead of fixation data which is also available from the eye tracker, because the algorithm used by Tobii for fixation calculation is not publicly available. The lightly filtered gaze data as received from the eye tracker is minimally encoded and sent over a TCP link to the server. The TCP link is parallel to, rather than coupled with, the user input channel and the video stream channel to prevent the other data flows from hindering the gaze data flow. The server, as mentioned above, accepts the gaze data, decodes and sanity-checks it to use only the latest gaze updates. The gaze coordinates are then used to develop a quality profile for the current frame in the encoding pipeline.

3.2 FVE with Real Time Gaze Tracking

Foveation occurs because of the non-uniform density of cone cells in the human eye. The relative visual acuity of the human eye is illustrated in Figure 4a. We simplistically model the relative visual acuity of the HVS as a two dimensional Gaussian function centered around the fovea. More complex models of foveation and relative visual acuity of the HVS for foveated encoding have also been developed, for example in [10]. We make a design decision to use a simpler model to minimize the modifications needed in the video processing pipeline of the encoding scheme and to minimize the latency overhead added by foveated encoding.



(a) Visual acuity of human eye

(b) QO at different values of W

Fig. 4. Foveation and QO calculation. FW is the width of the output frame in pixels

Video encoding has multiple steps of compression, which may be lossless or lossy. The primary lossy compression stage in modern encoders is the quantization step. This step of encoding enables realization of the trade-off between quality and video bitrate. The level of quantization determines the quality of reconstructed video: the higher the quantization, the lower the quality and the

resulting bitrate. Encoders may use different rate control strategies to achieve an optimal trade-off between quality and video bitrate by controlling quantization and other encoding parameters. Some strategies are, for example, constant quantization, attempting constant delivered quality or enforcing a constant bit rate.

3.2.1 Encoder. In this work the *x264* encoder [27] for the MPEG-4 Advanced Video Coding (AVC) standard [20] is used. In *x264* the quantization level applied to a macroblock is controlled by a parameter called Quantization Parameter (QP). In *x264* the QPs may be determined automatically by the encoder based on the rate control algorithm used or set by the user. In practice user-defined QPs are implemented at the encoder API level by allowing the user to add an offset (Quantization Offset, QO) to each QP calculated by the rate control algorithm when encoding a frame. One of the rate control strategies available in *x264* is the so called constant rate factor (crf) mode, in which the encoder attempts to maintain a constant perceived quality temporally, determining QP values accordingly. In the *crf* mode, the encoder takes advantage of the fact that the human eye perceives still and moving objects differently and compresses the video according to the motion in the frame. For single pass encoding, this mode is considered the most efficient. To implement FVE, we use *x264* in the *crf* mode, but we add an offset to the QPs calculated by the *crf* algorithm. The QOs are calculated such that the QO and hence the total QP is lowest at the gaze location and increases away from the gaze location. This scheme keeps the quality highest at the gaze location and lowers it away from the gaze. It should be noted that the QOs are sent to the encoder at the API level without modifying the underlying *x264* encoder.

3.2.2 QO calculation. In the prototype, the server is modified to accept gaze data sent by the client over a TCP connection parallel to the gameplay video and user input channels. At the server, the module responsible for inputting gameplay frames to the encoder calculates a QO for each macro block of the video frame. The GamingAnywhere server and client negotiate gameplay video resolution when they connect initially, so the video processing modules know what number of macroblocks to expect. To calculate the QO of each macro block, the gaze location is translated to a macroblock based coordinate system. The macroblock corresponding to the current gaze location is assigned the lowest QO, while the QO of macroblocks away from the gaze location increases progressively with distance from the gaze macroblock. Since we model the HVS acuity as a two dimensional Gaussian function, we calculate the offsets using a two dimensional Gaussian function. For the current video frame to be encoded, the QO, $QO(i, j)$ for a macroblock at i , j, where i and j are indices of the matrix of macroblocks comprising the frame, is calculated as:

$$QO(i,j) = QO_{max}\left(1 - \exp -(\frac{(i-x)^2 + (j-y)^2}{2(W)^2})\right) \tag{1}$$

In Equation (1), QO_{max} is the maximum offset which is configurable by the server administrator (or user), x and y are the indices of the macroblock corresponding to the gaze location, and W is a measure of the size of the foveal region. We define foveal region as the region on the screen of the client machine which corresponds to the gaze location and where the game video quality should be high. QO_{max} and W are user configurable and allow us to investigate the relationship between QO and size of the foveal region and the video bitrate and resulting QoE.

Figure 4b shows the relationship between QO and distance from the gaze location at various values of W (varied in terms of the output frame width FW). In a video frame, the area perceived with the highest visual acuity depends on the viewing distance. A larger viewing distance translates to a larger area sampled at high resolution by the HVS. In our prototype evaluations, we vary W in terms of the frame width FW because users naturally tend to view smaller screens from a smaller distance and larger screens from a larger distance. Defining W in terms of FW thus

makes the foveated encoding scheme scalable with screen dimensions and pixel density: at a given parameterization, the physical dimensions of the foveal region are proportional to the physical dimensions of the screen[2]. In this prototype W represents the diameter of a circle centered at the gaze point, where QO at any point on the circle is about 40% of QO_{max}, following our simple model of the HVS. We believe varying quality according to a Gaussian curve follows the HVS acuity better than step functions of quality variations used in other approaches of foveated encoding (e.g. [37] and [4]) and also reduces block artefacts.

4 SYSTEM EVALUATION

In this section, we investigate the effectiveness of foveated video streaming to reduce bandwidth requirements. We consider three games of different genres for analysis of their video bitrate with different parameterization of foveated encoding. Furthermore, we briefly analyze player gaze patterns with four games to roughly estimate the latency between a detected eye movement and the corresponding change in the received video.

4.1 Measurement Setup

The measurement setup comprises of our cloud gaming system prototype as described in section 3, wherein the client and the server are connected over a campus GbE network. An Ubuntu Linux work station serves as the server and a Windows laptop serves as the client. Three games of different genres with different gameplay styles are considered for video bitrate analysis: AssaultCube, Trine 2 and Little Racers STREET (henceforth abbreviated as Little Racers). AssaultCube is an action game of the First Person Shooter (FPS) genre, wherein the player controls a weapon from a selection of weapons from a first person point of view. Being an FPS game, it has a fast paced gameplay. Trine 2 is a side scrolling puzzle and adventure game, wherein the player assumes one of a selection of in-game avatars and explores the virtual world, solving challenges along the way. Little Racers is a so called top down racing game wherein the player races a car from bird's-eye perspective on different race tracks available in the game.

We conduct a set of measurements for each game by capturing all packets between the cloud gaming server and the client using tcpdump[3]. Raw tcpdump data is then analyzed with Wireshark[4] to extract throughput per second. In a set of measurements for a game, a player familiar with the gameplay controls plays the game for a fixed duration for each set of parameters while making an effort to replicate gameplay over the sessions. To encode the gameplay video, the cloud gaming server is configured to use the *x264* encoder with the following parameters:

```
--profile main --preset ultrafast --tune zerolatency --crf 28 --aq-mode 1 --ref
1 --me_method dia --me_range 16 --keyint 48 --intra-refresh --threads 4
```

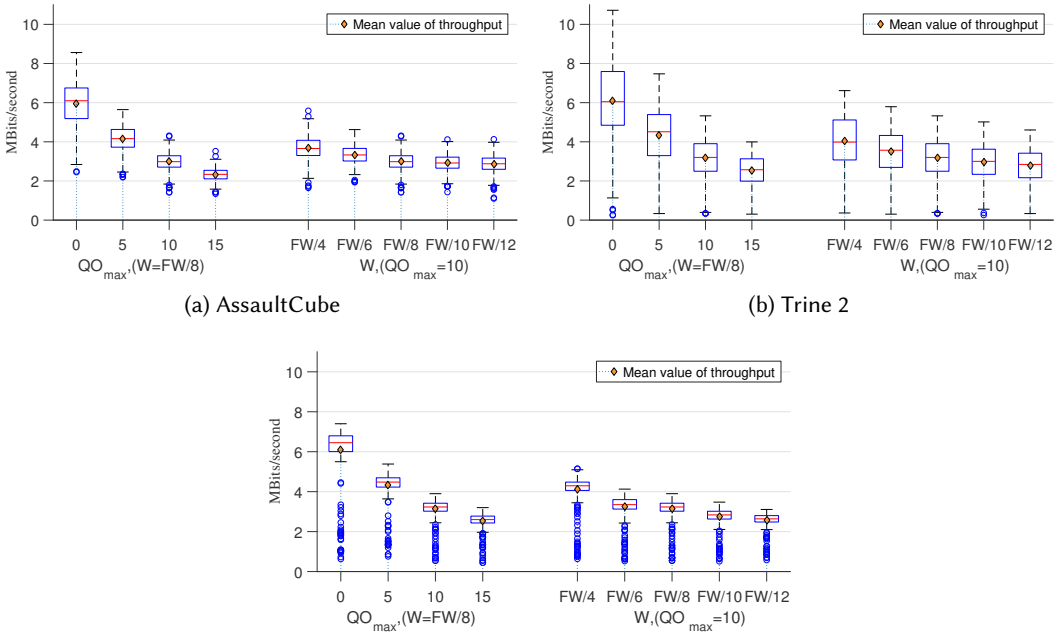
The encoding parameters are partly based on recommendations by developers of GamingAnywhere in [16], with QOs added on top of those calculated by *x264* as discussed in Section 3.2.2.

4.2 Foveation and Video Bitrate

To evaluate the effect of foveation on the video bitrate and consequent savings in the downstream bandwidth requirements, we perform a series of measurements with the above described setup. Over a set of measurements for each game, we vary the maximum Quantization Offset QO_{max} which controls the quality degradation while keeping the W parameter that controls the foveal region constant, followed by varying W while keeping QO_{max} constant. As mentioned earlier,

[2]Some modern gaze trackers can measure the viewing distance, allowing W to be set agnostic of screen specifications.
[3]http://www.tcpdump.org/
[4]https://www.wireshark.org/

we define values of W relative to the frame width (FW) of output video. The results for the three



(a) AssaultCube

(b) Trine 2

(c) Little Racers STREET

Fig. 5. Measured video bitrates with different games and parametrization of foveated encoding. FW is the width of the display in pixels. The box comprises the inter-quartile range, the red line in the middle of the box is the median, and the diamond denotes the mean.

games considered in throughput analysis are shown in Figure 5. Increasing the QO_{max} while keeping W constant results in a significant change in the output bitrate. However, decreasing W beyond 1/8th of the output frame size has no marked effect on output video bitrate. The reason for this is that beyond 1/8 of the screen size on the cloud gaming client of our setup, the number of macroblocks encoded at high quality (with low QOs) is a very small fraction of the total number of macroblocks on the screen. In the measurement setup the output frame width is 1366x768 pixels which corresponds to about 4000 macroblocks of 16x16 pixels. With $W = FW/8$ the number of macroblocks with a QO of 40% or less of QO_{max} is about 100. At higher resolutions of output video we expect the pattern of bandwidth savings to be similar to the results above. However, foveal regions of a smaller size (below $W = FW/8$) should have a more pronounced effect as the number of macroblocks affected increases.

Comparing the results from the three games it can be seen that there is little difference between the average or median bitrates in contrast to the variance, which is significant. This is due to the nature of the gameplay in each game. The least variance is in Little Racers which has a birds-eye view perspective wherein even when the player-controlled car is changing position constantly, the overall map and hence the frame graphics change infrequently. Trine 2, which has complex graphics and where the frame changes almost always with player actions, exhibits the most variance. However, changing the foveation parameters affects all games in a similar fashion.

(a) AssaultCube (b) Little Racers STREET

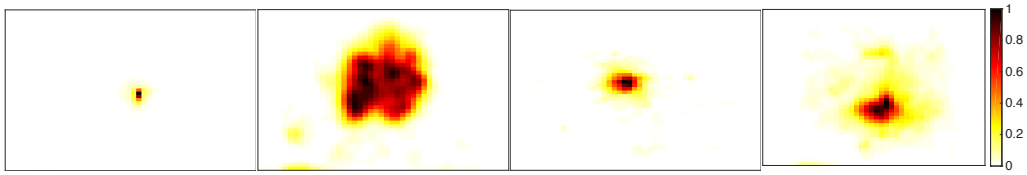
(c) Formula Fusion (d) Trine 2

Fig. 6. Sample screen captures of the games whose gaze patterns are considered

4.3 Gaze Patterns and Latency

To investigate the latency-related feasibility of foveated streaming for cloud gaming we study gaze data from four games. Three of the games are same as considered in section 4.2. An additional game, Formula Fusion, is also considered. Formula Fusion is a "futuristic" racing game with a fast game play. The player's point of view may be configured to be behind the vehicle or inside the vehicle. We configure it in the behind-the-vehicle mode. For the analysis, the Tobii 4C eye tracker is configured to capture gaze data for each game while a player plays the game on a Windows computer for 15 minutes. Sample screen captures of the games are presented in Figure 6. From Figure 6, we can observe where the user is likely to fixate their gaze and where they might glance occasionally.

4.3.1 Gaze Patterns. The gaze data for each game is plotted as heatmaps in Figure 7, using bivariate Gaussian kernel density estimation of gaze coordinates. Inspecting the heatmaps, the highest density of gaze coordinates, for all games, is at the center of the screen. This is expected as most gameplay graphics recenter at or around the center of the frame. In AssaultCube, which is an FPS game, the gaze coordinates are highly localized to the center of the frame. In FPS games the player's attention is directed towards the cross-hairs of their weapon most of the time, which is usually located in the center of the frame. There might be occasional glances to various information icons, like the map and in-game incident reporter (see Figure 6), but they are too few in number to register in the heatmap. The gaze location heatmap of Little Racers shows the widest spread of gaze location around the center. Again this is explained by the nature of the game play; the player's point of



(a) AssaultCube (b) Little Racers STREET (c) Formula Fusion (d) Trine 2

Fig. 7. Gaze tracking heatmaps from 15 minute gameplay sessions. The color scale is normalized.

attention is the car they control around a race track. The race track, although occupying the whole frame, is re-centered as the car is moved (to and around the frame center). Gaze location data for Trine 2, where the player controls an avatar from a two dimensional side view, shows wider spread of gaze locations than AssaultCube or Formula Fusion. The avatar can move at least forward (right), backward (left), and up (jump), explaining the spread. Formula Fusion shows a gaze pattern similar to, but more spread out than AssaultCube, due to the fact that the car is roughly located at the center of the frame most of the time and the player may glance around to explore the upcoming track and other vehicles on the track. Glances to the periphery of the gameplay frame are too low in number to register in the heat map for all games except Little Racers where some gaze locations at the bottom left of the frame register. This is due to the game play race map being located there[5]. It is evident that the seamlessness of foveated encoding in cloud gaming depends on the type of gameplay, but it may be possible to draw genre specific conclusions.

4.3.2 Latency Considerations. We next examine *gaze moments*, which we define as time periods within which the user's gaze lingers within a circular region of a certain radius. We define regions of two radii, $FW/8$ and $FW/4$ whose cumulative distribution functions of gaze moments are plotted in 8 and 9 respectively. From the plots it can be observed that the user's gaze lingers within a region of radius $FW/4$ and even $FW/8$ almost all the time for a time slot longer than the sampling interval of the eye tracker (which is approximately 10ms). The scenarios where gaze moments are shorter than 10ms are the most challenging in terms of providing a seamless gaming experience without the user observing foveation, considering the end-to-end latency. Furthermore, note from the plots in either definition of the gaze region ($FW/8$ or $FW/4$) that a vast majority of the gaze moments, about 80-90%, last longer than 100ms for all the games. Long gaze moments which last longer than 1s comprise about 20-40% of the gaze moments, and as expected these long gaze moments comprise a larger fraction for AssaultCube and Formula Fusion.

To investigate how fast a player's gaze moves while playing the considered games, we compute the rate of gaze changes during play. The rate of gaze changes is calculated by dividing distance of consecutive gaze data samples, in pixels, by the time difference of samples[6]. Figure 10 illustrates the CDF of the results. It is evident that gaze change rates of more than 1000 pixels/s, which indicate across screen glances, are rare. Even at 1000 pixels/second, with 40-45 fps encoding used in our experiments, the per frame change in gaze location is less than 25 pixels, which is well within the range of the high quality (foveal) region. For the vast majority of gaze changes which exhibit slower rate of change of gaze, it follows, the per frame change in gaze is even smaller. Fast gaze

[5]In games with a fixed location of a game map or a heads up display, it is possible in our prototype to apply smaller or no quantization offsets to those locations, keeping the quality of the game-play map or heads-up display high.

[6]It should be noted that the Tobii 4C eye tracker uses some filtering based on previous gaze location and age of the sample and this might affect the results. The filtering algorithm is proprietary and hence we do not know how strong the effect is.

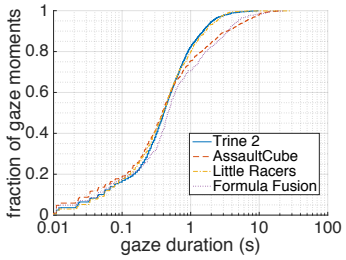


Fig. 8. CDF of gaze moment duration when $W = FW/8$.

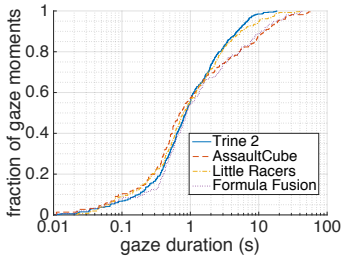


Fig. 9. CDF of gaze moment duration when $W = FW/4$.

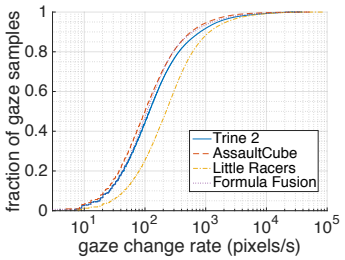


Fig. 10. CDF of rate of gaze shifting.

movements are challenging for seamless foveated video encoding. However,the psycho-physical phenomena involved in such movements such as saccadic omission (which entails loss of visual acuity during a saccade[44]), together with detection of onset of a saccade could be used to improve quality (reduce *QOs*) in the target region of a saccade. For the user study environment described in the next section and assuming an end-to-end system latency of 100ms, we calculate a maximum tolerable rate of gaze change of about 1200 pixels/seconds beyond which a players gaze lands in a region of low video quality before the frame is updated.

Latency in cloud gaming has been the focus of research by many researchers [9], [8], [11], including us[22]. Previous work on the quality of experience of cloud gaming suggests a latency threshold of 100ms [21], beyond which the user's QoE begins to degrade. In [22], achievable end-to-end latency (e2e) between a user control action to the corresponding change in a video frame at the client is investigated. It is observed that with a low enough network latency of 20-30ms, and well-provisioned compute and render resources, a sub 100ms e2e latency is achievable. Recent work on latency requirements in Virtual Reality (VR) [3] reports an eye-to-image latency between 50ms to 70ms for seamless foveated rendering[7]. It is noted in[3] that the results are conservative as the test subjects were specifically asked to look for artifacts in the peripheral regions. In a natural gaming or video watching environment, the latency requirements may be less stringent. Furthermore, with an increase in the size of the foveal region (which the authors parameterize as a combination of eccentricity and blur radius), higher values of latency may be usable.

Characterizing the latency of the Tobii Eye Tracker 4C is beyond the scope of this work. However, the latency of Tobii EyeX, the predecessor to Eye Tracker 4C has been found to be about 50ms [14]. Tobii Eye Tracker 4C has a higher sampling rate than Tobii EyeX and uses purportedly improved hardware and algorithms, so it may be assumed that the latency of the 4C is less than 50ms. Considering this latency in the end-to-end latency of a modern (mobile) device in a cloud gaming setup by replacing the device-to-kernel latency of the device with the eye tracker latency [22], a sub 100ms end-to-end latency is still possible. In Figures 8 and 9 we see that a majority of gaze moments last longer than 100ms and hence longer than the time it takes to update the foveal region. Furthermore, from Figure 10, we see that for all games except Little Racers, for 50% of the time, the gaze changes by less than 10 pixels from the time the eye gaze is located to the time the foveated region is updated on the screen. With suitably defined values for W parameter, foveated encoding should be transparent to the user. However, since QoE is a highly subjective experience, we conduct a user study to validate these postulations and inferences.

[7]Note that in VR applications eye-to-image latency constraints are more stringent due to certain physiological effects of VR environments collectively called VR sickness.

Table 1. Dependent variables recorded in the experiment per session, using 100-point Likert scales. Before the experiment started the participant was explained what was meant by each question.

Variable	Explanation
Video quality	Quality of the video image (pixelation, artefacts, etc. as opposed to detail of in-game models)
Video adequacy	How adequate the video was for their task, i.e. did the video hinder their performance or not?
Enjoyment	How enjoyable was the experience?
Satisfaction	How satisfied are you with your performance (score)?
Effort	How much effort did you put into completing the task?
Concentration	How well were you able to keep your concentration?

5 USER STUDY

5.1 Method

In order to determine what parameterization of the foveated video encoding (FVE) are optimal with regards to QoE and bandwidth usage, we set up a controlled laboratory experiment to gather data on the perceived video quality and a player's experience in using our foveated cloud gaming system. In the experiment participants played a game on the prototype with different FVE parameters. The goal of the experiment was twofold: Firstly, to determine whether users notice if foveated video encoding is being used, and, if so, how strongly FVE influences their experience in playing a game. Secondly, to compare the subjective assessments of the video quality to the bandwidth usage for several different parameter values to try and determine a relationship between them. Using the latter comparison we show support for the hypothesis that there is an optimal parameterization with regards to bandwidth usage and video quality.

5.2 Participants

We recruited 12 participants from the Computer Science building at Aalto University. In order to make sure that their ability to play the game itself was not a confounding factor, we invited only participants that had some experience in playing First Person Shooter (FPS) games on PC (i.e. using a keyboard and mouse as the controller). The participants were aware only of their task, the data we collected and that the system was based on cloud gaming. Only after the experiment was finished were the participants told the purpose of the experiment and the particular kind of foveated encoding technology that was used.

5.3 Experiment Design

The experiment was a within-subjects design with the maximum quantization parameter (QO_{max}) as the independent variable. The independent variable was divided into 5 levels with equal intervals from highest to lowest QO_{max}, where a QO_{max} of 16 corresponds to the maximum level of quantization (i.e. worst quality) and QO_{max} of 0 corresponds to no foveated encoding. The dependent variables were video quality, video adequacy, enjoyment, performance satisfaction, effort and concentration, which were measured through respective 100-point Likert scales (see Table 1. The remaining dependent variables were gaze data (output from the client software), bandwidth (measured through Wireshark on the client), participant comments (taken after each session), and the participant's score.

The rating scale we used is adapted from Mullin et al.[30] and is further inspired by Pauliks et al.'s work[32]. Pauliks et al. argue that for short video presentations any method of assessing video quality is equally well-suited. Mullin et al. discuss the problems with ITU recommended video quality assessment scales, and propose the use of a 0 - 100 rating scale without labels. In this study, we asked the participants to rate the video quality and how adequate the video was for the task on a 0-100 rating scale. The other questions related to the participant's enjoyment, performance, concentration and effort, respectively. For an overview of the rating scale questions and their explanation as given to the participant, see Table 1.

The client and server of the prototype system as described in 3 are deployed respectively on a Lenovo IdeaPad Y580 laptop with a 15.6" full-HD screen (1920x1080) running Windows 10 64-bit and an Ubuntu Linux 16.04 LTS 64-bit host with an Nvidia GeForce GTX 1050Ti graphics card. The game used is Assault Cube v1.1.0.4 and the map used is 'Desert' with 7 bot players in a Free-for-all game mode ('Bot Deathmatch'). The Tobii 4C eye-tracker was attached to the lower part of the laptop screen, aimed towards the user's face and calibrated for each participant. The participant was seated on a standard-issue chair at a fixed distance (50cm) from the laptop screen and the laptop and participant are aligned on the same axis. The participant used a standard issue wireless optical mouse to control the game, together with the laptop's keyboard. The bandwidth data was recorded using Wireshark 2.6.0, while the client device and server were connected on the campus intranet on a 1 Gbps wired connection. The set-up can be seen in Figure 11.

Fig. 11. *Experimental setup showing the Tobii 4C eye-tracker (bottom of the screen), peripherals and the Lenovo Y580 client device rendering AssaultCube on the Desert map.*

5.4 Materials & Apparatus

The foveated area was set as a circle, with a radius of 1/8th of the screen size: recall from previous sections that the foveated area of the human visual system is about 2 degrees, and that the effective width depends on the distance between the participant's eye and the screen. Given the distance to the screen of about 50 cm and a screen width of 40 cm, we calculate the foveated area to be about 2 cm, which is about 1/20th of the screen width. The W parameter was set conservatively to $W = FW/8$ to ameliorate sudden gaze movements and any inaccuracies in eye tracker data.

5.5 Procedure

For each participant the experiment consisted of five sessions of 5 minutes each, where each session used a different QO_{max} setting. The QO_{max} settings we used were $QO_{max} = 0, 4, 8, 12, 16$. In order to control for learning and fatigue effects, we used counterbalancing with rotation, where the starting condition was rotated and the order of the different conditions was fixed in the following sequence: $QO_{max} = 8, 4, 12, 0, 16$ for the respective sessions. The task, for each session, was to get at least 40 kills (amount of deaths was mentioned to be irrelevant). Before the experiment the participant played a warm-up game (at $QO_{max} = 0$) for 2 minutes, in order to get comfortable with the game controls and to ensure the functioning of the software. In order to prevent hypothesis guessing and to prevent the participants from focusing on video quality exclusively, we provided each participant with a challenging main task. We assigned each participant an in-game difficulty level based on their skill level to ensure that the main task was challenging enough for each participant. In a pilot study, some of the more experienced users reported that the game was too easy on the 'Worse' or 'Medium' settings, which caused them not not focus on the task. In order to control this we set the difficulty to one of three increasingly difficult levels so each user has a challenging task. Using the self-reported skill level of the participant and their performance in the warm-up game, the difficulty level to use was determined to be 'Worse', 'Medium' or 'Good'.

(1) The participant was invited in, welcomed and the general idea of the experiment and the procedure was explained to them. Before starting the experiment the participant was also briefed on what constitutes video quality (as opposed to in-game graphics quality). The participant was not told about foveation or encoding used in the system, to avoid hypothesis guessing. We explained the questions on the rating scales and asked the participant to read and sign the consent form.

(2) The participant was then placed on the chair in front of the laptop, and asked to sit comfortably in front of the computer considering the distance and alignment. The eye-tracker was calibrated and the warm-up game started, during which the controls of the game and the task were explained.

(3) After the warm-up game, and after answering any questions from the participant and confirming that the participant was ready, the first condition of QO_{max} was started. When the participant commenced the task, we started a timer for 5 minutes, after which we determined and recorded the participants score in the game and asked the participant to rate enjoyment, performance satisfaction, video quality, video adequacy, effort and concentration on respective 100-point Likert scales. We also asked the participant to comment in general on their thoughts about playing that session. This step was repeated for subsequent conditions with the different respective settings of QO_{max}.

(4) After the last session was finished as per step 3, we explained to the participant what technology we were using (particularly the foveated video encoding) en the purpose of the experiment. We then asked them to, with this new knowledge, comment on their experience.

5.6 Results: Foveation and QoE

In this section we present and discuss the results of the user study. We first study the Mean Opinion Scores (MOS) for video quality and objective evaluation of video quality, followed by the results of the bandwidth usage logging and the reported scores for the other questions on the questionnaire that served as a control and relate more to the task overall. We aggregated the scores per participant and grouped them by maximum quantization offset (QO_{max} parameter). Recall that the QO_{max} setting was varied per condition, so the 5 different levels correspond directly to the five different sessions and their respective conditions. The score range on all rating scales was 0 - 100, where 0 is the lowest score, and 100 the highest. Participants were free to give any rating on this scale. We calculated the Mean Opinion Score as the mean of the responses per QO_{max}. We used a Student's t-distribution to estimate the mean and calculate the 95% confidence intervals, due to the relatively

Table 2. (Parameter) settings used in the experimental setup.

Parameter	Setting	Note
QO_{max}	8, 4, 12, 0, 16	Varied per subsequent session, in this order. Starting condition rotates.
W	8	Fixed, used to control the size of the foveated area.
Render resolution	1920 x 1080	Resolution set in the game and fed to the encoder in pixels, fixed.
Video resolution	1920 x 1080	Resolution of the video stream the client receives in pixels, fixed.
Video FPS	50	Frames-per-second set in the encoder.
Graphics quality	Highest	Relates to the video quality settings built into AssaultCube.
Goal score	40 kills	Target score in kills the participant needed to reach, deaths was irrelevant.
Difficulty setting	W(1), M(6), G(5)	Bot difficulty: Worse, Medium, Good with number of assignments.
Wireshark filter		*host [Server IP] and not port 22 and not port 5900*
Eye-tracker		Calibrated at the start of the experiment for each participant.

low number of samples ($n = 12$). After a participant had finished all their sessions, we asked for free form comments and also informed them about the purpose of the experiment asking them whether they had noticed FVE.

In order to understand how the results compare to those obtainable with objective video quality metrics, we also calculate PSNR and Eye Weighted PSNR (EWPSNR) [23] of a foveated encoded gameplay video sequence at various QO_{max} values. The foveated encoding considered the gaze to be fixated at the center of the frame.

5.6.1 Video Quality. In Figure 12 QO_{max} is plotted together with the Mean Opinion Score of the video quality. In Figure 13, the video quality MOS and bandwidth usage are plotted against QO_{max}. Note that the left y-axis (average megabits per second) relates to the bandwidth usage and the right y-axis (score 0 - 100) to the video quality MOS. In Figure 12 we can see a drop-off in perceived quality when $QO_{max} > 8$. We can see that $QO_{max} = 4$ is rated equal to using no foveated encoding ($QO_{max} = 0$), and that at $QO_{max} >= 12$ users consistently rate the quality to be low. This corresponds to the free-form comments we received during the experiment, where 7 out of 12 participants commented on the video quality being much worse suddenly for $QO_{max} = 16$, and 3 out of 12 for $QO_{max} = 12$. A two sample t-test assuming unequal variances confirms this: the means are not significantly different between $QO_{max} = 0$ and $QO_{max} = 4, 8$, or 12 ($alpha = 0.05, p = 0.92, 0.53, 0.14; t = 0.106, 0.65, 1.53 < tCritical = 2.1$), although $QO_{max} = 12$ is close. The means for $QO_{max} = 0$ and $QO_{max} = 16$ are significantly different ($alpha = 0.05, p = 0.009; t = 2.88 > tCritical = 2.88$). The bandwidth usage as depicted in Figure 13 follows that of results in Section 4.2, showing a logarithmic decrease in bandwidth usage with QO_{max}. The video quality is best at $QO_{max} = 0$, and consequently requires the most bandwidth. At about $QO_{max} = 8$ the bandwidth usage starts to plateau. Considering both video quality and bandwidth usage, it is clear that between $QO_{max} = 8$ and $QO_{max} = 12$ there is potential for finding a sweet spot where the bandwidth savings are significant while QoE is minimally affected.

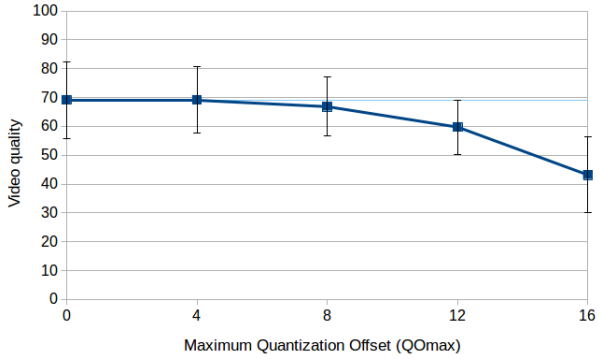


Fig. 12. *MOS for the video quality (scale 0 - 100), by QO_{max} with 95% Confidence Interval. The light blue line at video quality = 69 represents the MOS for $QO_{max} = 0$.*

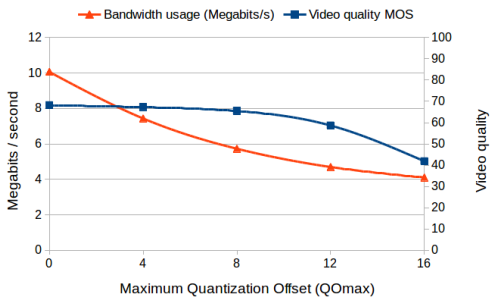


Fig. 13. *MOS for the video quality (scale 0 - 100) plotted against the bandwidth usage in average megabits per second, by QO_{max}.*

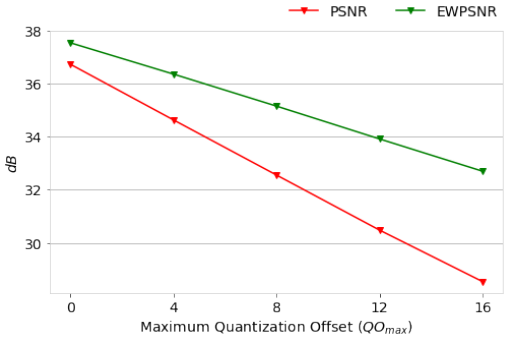


Fig. 14. *Average PSNR and EWPSNR values of foveated gameplay video sequence at different QO_{max} values. The gaze was fixated at the center of the frame during foveated encoding*

The average PSNR and EWPSNR for different values of QO_{max} are shown in Figure 14. It is clear that with increase in QO_{max} both PSNR and EWPSNR become smaller, but the drop in PSNR values is steeper, which suggests that EWPSNR indeed accounts for the foveated encoding to some extent. At $QO_{max} = 8$, the drop in EWPSNR is just 2dB and at $QO_{max} = 12$ the drop is less than 4 dB indicating a presumably lower loss of quality, which is corroborated by the MOS scores in Figure 12. Interestingly, MOS scores decrease nonlinearly as a function of QO_{max}, while PSNR and EWPSNR values decrease linearly. It is illustrative of the fact that objective metrics, even the ones that adapt to gaze location like EWPSNR, may not precisely reflect the viewer perceived video quality when foveated video encoding is applied. Further work is needed to understand the root cause of this discrepancy and hopefully develop more appropriate metrics.

We also plot individual MOS difference scores with respect to QO_{max} in Figure 15. The difference is calculated with respect to the condition $QO_{max} = 0$, at which the MOS difference for a user is zero. If we take a look at the individual difference scores in Figure 15, we see an interesting picture: two participants rate the video quality to be almost linearly worse for higher values of QO_{max} (4 and 10). Upon further inspection of the comments and their data we see that one of these

participants had a high level of experience, and reported lag during the sessions. The participant reported that their experience overall was not up to their expectations, and as a result their ratings clearly diverge from the average. The second participant may have suspected that the experiment was about the video quality in an FVE context, and thus payed more attention to it. Looking at the other responses we see that there are rather large differences in how participants rate video quality: at $QO_{max} = 4$ in Figure 15 we can see that no participant gave the same rating in this condition. On the other hand, at $QO_{max} = 12$ there seems to be a point of convergence, where participants rate the quality as being nearly equal to the condition without foveation ($QO_{max} = 0$). Combined with the MOS scores from Figure 12, the data seems to suggest that users do not notice significant differences in video quality until $QO_{max} > 12$. Recall however, that the starting condition was rotated and that the order of QO_{max} is different from the order on the X-axis in Figure 15.

5.6.2 Video Adequacy and Game Enjoyment. We also asked participants to rate the adequacy of the video after each session and plotted their responses against their video quality ratings in Figure 16. Here we explained to participants that by video adequacy we mean how much they felt that the video quality allowed them do to their task (getting 40 kills) well. Hence, if encoding artefacts are distorting a user's vision, we expect the adequacy to be rated low, while if there are no obvious hindrances, we expect the adequacy to be rated high, regardless of other aspects related to performance and quality. Comparing the MOS for video quality and adequacy of the video in Figure 16 shows no big surprises, with most participants finding the quality overall to be quite adequate for the task. Only at $QO_{max} = 16$ participants reported that the lower quality was disturbing and less adequate (e.g. they failed to see items on the ground, such as ammunition boxes or players in a dark corner).

Finally, as a control, we asked the participants to rate how satisfied they were with their performance, how much they enjoyed the task, how much effort they put in and how well they were able to hold their concentration. Their scores are plotted in Figure 17. The results are consistent over the different QO_{max} settings, but performance satisfaction and enjoyment vary more: most likely due to different expectations and latency issues (participants with less experience reported an overall higher level of enjoyment and satisfaction regardless of score and latency issues, while more experienced players reported that their performance and possible latency issues were not up to their standards, thus lowering their respective scores on the rating scale). The consistent results may be interpreted as the players being engaged with the game notwithstanding the video quality, which augments our results for video quality and adequacy.

Free form comments are roughly grouped into comment classes. In response to whether FVE was noticeable, one participant reported that the degradation in quality (at $QO_{max} = 16$) was especially visible when he made fast gaze movements, but otherwise did not notice this having anything to do with the use of the eye-tracker or the foveated encoding. All other participants reported to have been fully unaware of the purpose of the eye-tracker and the use of foveated video encoding. All the participant's comments have been generalized and depicted in Figure 18. Note that the comment 'Good quality' was mainly uttered with regard to a previous session, i.e. meaning 'better quality than before'.

6 LIMITATIONS, CONCLUSIONS AND FUTURE WORK

6.1 Limitations

A limitation of the prototype is limited mitigation (by increasing W) for quick gaze movements (i.e. high amplitude saccades). One possible solution, which we plan to implement in future work is synergistic use of real time gaze information with saliency of video game-play. A rather simple saliency map is the location of a game map or a so-called heads-up display in the video frame.

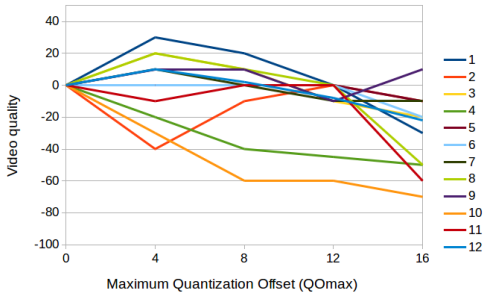


Fig. 15. *Individual difference scores for the video quality per QO_{max}, with reference $QO_{max} = 0$.*

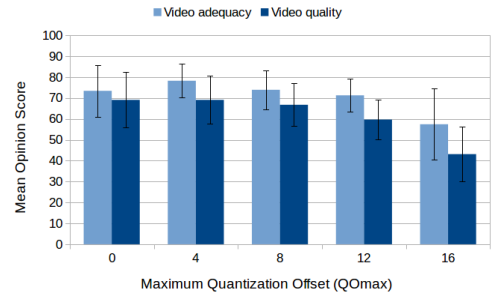


Fig. 16. *Video adequacy MOS (scale 0 - 100= and Video Quality MOS (scale 0 - 100) QO_{max} with 95% confidence intervals.*

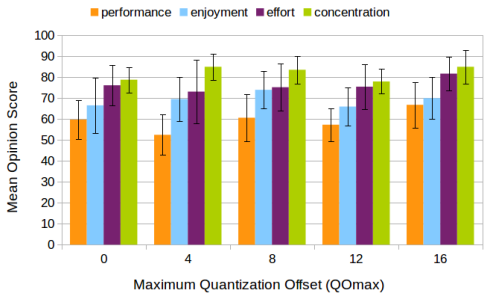


Fig. 17. *Mean Opinion Scores for the task-related rating scales: Enjoyment, Performance satisfaction, Effort and Concentration.*

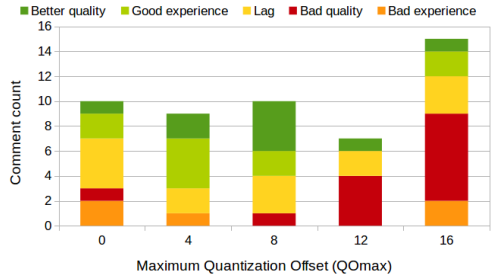


Fig. 18. *Generalized comments from the participants grouped by QO_{max}. Y-axis represents how many times a particular comment was given.*

Another possible solution is detecting onset of a saccade and leveraging the pyshco-physical phenomenon of saccadic omission to pre-emptively update the spatial quality profile of the frames being encoded.

The main issue we encountered during the user study was user-reported "lag": during several sessions, users reported more or less noticeable amounts of lag in the game. In some sessions the connection dropped completely, and the session had to be restarted. Two participants reported seeing pixelated areas, consistent with the extra-foveal region, which may have been due to a delay in updating the foveal region. Furthermore, what constitutes "lag" is subjective. An experienced player may have a lower tolerance for acceptable lag, while a casual player may have a higher tolerance. Some of our participants referred to the lag as "input lag" while others suggested it was network latency. The reported delays may be due to the cloud gaming software or network issues, but despite our best efforts we have not been able to determine the exact cause of the reported delays. Although care should be taken in future work to prevent unwanted delays, we believe the influence on our results is limited, given the data and participant comments.

In the user study of the current work we studied a single game, which was a First-Person Shooter. This type of game invites users to mostly look at the centre of the screen, as observed in our gaze data heat maps, which is not representative of all games. However, with relatively stable gaze patterns it is easier to gauge whether users notice degraded quality outside of the foveated area,

whether it causes distractions or limits their ability to e.g. observe information at the edges of the screen, during intense gameplay or in conditions with delays.

Considering the large confidence intervals in Figure 12 and the high variability we observed in the data, it appears that participants grade video quality differently. This shows in the different initial scores participants gave for a condition and the fact that the maximum quality condition ($QO_{max} = 0$ is rated at 70/100. Although a capable and fast open-source cloud gaming platform, GamingAnywhere might affect perceived gameplay video quality (on the client), especially when compared to native rendering or cutting-edge commercial cloud gaming services like GeForce Now [8], where the server-side hardware is optimized for cloud gaming. However, the higher quality in such systems would likely be due to higher baseline available bandwidth which may remove the need for foveated encoding altogether. In this work we investigated the influence of different QO's on Quality of Experience and bandwidth usage, taking GamingAnywhere and AssaultCube's maximum quality settings as the baseline. It may be that a different system provides a higher baseline quality, and this may influence the noticeability of foveated encoding as well. However, this question is outside of the scope of the current work. The participant's characteristics are another possible influencing factor. We recruited participants that have some experience with video games, and explained how video quality differs from in-game graphics quality. However, our data suggests that more experienced players have higher standards for video quality, and we cannot assert that none of our participants confounded video quality and graphics quality.

6.2 Conclusion and Future Work

In this work, we proposed to combine cloud gaming with foveated graphics. We developed a prototype system that integrates foveated streaming with off-the-shelf gaze tracker device into state-of-the-art cloud gaming software. Our evaluation results suggest that its potential to reduce bandwidth consumption is significant, as expected. We also demonstrate the impact of different parameter values on the bandwidth consumption with different games and provide some pointers on how to select parameter values. Back of the envelope latency estimations based on related work and gaze tracker specifications combined with gaze data analysis give us reason to be relatively optimistic about the impact on user experience. A user study establishes the feasibility of FVE for FPS games. The user study underlines the significant bandwidth savings that can be accrued with suitable parameterization of FVE without sacrificing QoE. Since FPS games have significantly tighter latency constraints, we are optimistic that foveated graphics for games of other genres will show similar results. As future work, we are planning to examine the QoE dimension in more depth through more subjective studies considering different genres of games and impact of network latency. Another direction we want to explore is synergistic use of saliency maps (of video gameplay) and gaze tracking for FVE. We see the current work as a stepping stone towards a broader investigation into how to properly apply foveated encoding in cloud gaming. Furthermore, we intend to attempt eliminating specialized hardware for eye tracking by employing web-cameras for the purpose. Using web cameras, which are ubiquitous in modern consumer computing devices like netbooks and mobile devices, would enable widespread adoption of foveated streaming for cloud gaming. Lastly, we also intend to investigate the feasibility of extending the work towards Virtual Reality.

REFERENCES

[1] Hamed Ahmadi, Saman Zad Tootaghaj, Mahmoud Reza Hashemi, and Shervin Shirmohammadi. 2014. A Game Attention Model for Efficient Bit Rate Allocation in Cloud Gaming. *Multimedia Syst.* 20, 5 (Oct. 2014), 485–501.

[8] https://www.nvidia.com/en-us/geforce/products/geforce-now/

[2] Zahid Akhtar and Tiago H. Falk. 2017. Audio-Visual Multimedia Quality Assessment: A Comprehensive Survey. *IEEE Access* 5 (2017), 21090–21117. https://doi.org/10.1109/ACCESS.2017.2750918
[3] Rachel Albert, Anjul Patney, David Luebke, and Joohwan Kim. 2017. Latency Requirements for Foveated Rendering in Virtual Reality. *ACM Trans. Appl. Percept.* 14, 4, Article 25 (Sept. 2017), 13 pages. https://doi.org/10.1145/3127589
[4] Ayub Bokani. 2014. Empirical evaluation of real-time video foveation. In *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*. ACM, 45–46.
[5] Speedtest by Ookla. 2019. *Monthly comparisons of internet speeds from around the world.* Retrieved August 15, 2019 from https://www.speedtest.net/global-index
[6] Wei Cai, Min Chen, and Victor C. M. Leung. 2014. Toward Gaming as a Service. *IEEE Internet Computing* 18, 3 (May 2014), 12–18. https://doi.org/10.1109/MIC.2014.22
[7] Wei Cai, Conghui Zhou, Victor C. M. Leung, and Min Chen. 2013. A Cognitive Platform for Mobile Cloud Gaming. In *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, Vol. 1. 72–79. https://doi.org/10.1109/CloudCom.2013.17
[8] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. 2014. On the quality of service of cloud gaming systems. *IEEE Transactions on Multimedia* 16, 2 (2014), 480–495.
[9] Kuan-Ta Chen, Yu-Chun Chang, Po-Han Tseng, Chun-Ying Huang, and Chin-Laung Lei. 2011. Measuring the Latency of Cloud Gaming Systems. In *Proceedings of the 19th ACM International Conference on Multimedia (MM '11)*. ACM, New York, NY, USA, 1269–1272. https://doi.org/10.1145/2072298.2071991
[10] Zhenzhong Chen and Christine Guillemot. 2010. Perceptually-Friendly H.264/AVC Video Coding Based on Foveated Just-Noticeable-Distortion Model. *IEEE Transactions on Circuits and Systems for Video Technology* 20, 6 (June 2010), 806–819. https://doi.org/10.1109/TCSVT.2010.2045912
[11] Sharon Choy, Bernard Wong, Gwendal Simon, and Catherine Rosenberg. 2012. The Brewing Storm in Cloud Gaming: A Measurement Study on Cloud to End-user Latency. In *Proceedings of the 11th Annual Workshop on Network and Systems Support for Games (NetGames '12)*. IEEE Press, Piscataway, NJ, USA, Article 2, 6 pages. http://dl.acm.org/citation.cfm?id=2501560.2501563
[12] Mark Claypool and Kajal Claypool. 2006. Latency and Player Actions in Online Games. *Commun. ACM* 49, 11 (Nov. 2006), 40–45. https://doi.org/10.1145/1167838.1167860
[13] Jonathan Deber, Ricardo Jota, Clifton Forlines, and Daniel Wigdor. 2015. How Much Faster is Fast Enough? User Perception of Latency & Latency Improvements in Direct and Indirect Touch. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* 1, 1 (2015), 1827–1836. https://doi.org/10.1145/2702123.2702300
[14] Agostino Gibaldi, Mauricio Vanegas, Peter J. Bex, and Guido Maiello. 2017. Evaluation of the Tobii EyeX Eye tracking controller and Matlab toolkit for research. *Behavior Research Methods* 49, 3 (01 Jun 2017), 923–946. https://doi.org/10.3758/s13428-016-0762-9
[15] Chun-Ying Huang, Cheng-Hsin Hsu, Yu-Chun Chang, and Kuan-Ta Chen. 2013. GamingAnywhere: An Open Cloud Gaming System. In *Proceedings of the 4th ACM Multimedia Systems Conference (MMSys '13)*. ACM, New York, NY, USA, 36–47. https://doi.org/10.1145/2483977.2483981
[16] Chun-Ying Huang, Cheng-Hsin Hsu, Yu-Chun Chang, and Kuan-Ta Chen. 2013. GamingAnywhere: An Open Cloud Gaming System. In *Proceedings of the 4th ACM Multimedia Systems Conference (MMSys '13)*. ACM, New York, NY, USA, 36–47. https://doi.org/10.1145/2483977.2483981
[17] Wijnand Ijsselsteijn, Yvonne de Kort, Karolien Poels, Audrius Jurgelionis, and Francesco Bellotti. 2007. Characterising and Measuring User Experiences in Digital Games. *International Conference on Advances in Computer Entertainment* (2007).
[18] Gazi Illahi, Matti Siekkinen, and Enrico Masala. 2017. Foveated Video streaming for cloud gaming. In *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. 1–6. https://doi.org/10.1109/MMSP.2017.8122235
[19] Laurent Itti. 2004. Automatic foveation for video compression using a neurobiological model of visual attention. *IEEE Transactions on Image Processing* 13, 10 (Oct 2004), 1304–1318. https://doi.org/10.1109/TIP.2004.834657
[20] ITU-T Rec. H.264 & ISO/IEC 14496-10 AVC. 2003. Advanced video coding for generic audiovisual services. *ITU-T (2003)* (May 2003).
[21] M. Jarschel, D. Schlosser, S. Scheuring, and T. Hoßfeld. 2011. An Evaluation of QoE in Cloud Gaming Based on Subjective Tests. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. 330–335. https://doi.org/10.1109/IMIS.2011.92
[22] Teemu Kämäräinen, Matti Siekkinen, Antti Ylä-Jääski, Wenxiao Zhang, and Pan Hui. 2017. A Measurement Study on Achieving Imperceptible Latency in Mobile Cloud Gaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, New York, NY, USA, 88–99. https://doi.org/10.1145/3083187.3083191
[23] Zhicheng Li, Shiyin Qin, and Laurent Itti. 2011. Visual attention guided bit allocation in video compression. *Image and Vision Computing* 29, 1 (2011), 1 – 14. https://doi.org/10.1016/j.imavis.2010.07.001

[24] Pietro Lungaro, Rickard Sjöberg, Alfredo J.F. Valero, Ashutosh Mittal, and Konrad Tollmar. 2018. Gaze-Aware Streaming Solutions for the Next Generation of Mobile VR Experiences. *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (April 2018), 1535–1544. https://doi.org/10.1109/TVCG.2018.2794119

[25] Pietro Lungaro and Konrad Tollmar. 2017. QoE design tradeoffs for foveated content provision. *2017 9th International Conference on Quality of Multimedia Experience, QoMEX 2017* (2017). https://doi.org/10.1109/QoMEX.2017.7965669

[26] Twan Maintz. 2005. Digital and medical image processing. *Universiteit Utrecht* (2005).

[27] Loren Merritt and Rahul Vanam. 2006. x264: A high performance H. 264/AVC encoder. *online] http://neuron2. net/library/avc/overview_x264_v8_5. pdf* (2006).

[28] Iman S. Mohammadi, Mahmoud-Reza Hashemi, and Mohammad Ghanbari. 2015. An object-based framework for cloud gaming using player's visual attention. In *2015 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*. 1–6. https://doi.org/10.1109/ICMEW.2015.7169781

[29] Sebastian Möller, Dennis Pommer, Justus Beyer, and Jannis Rake-Revelant. 2013. Factors influencing gaming qoe: Lessons learned from the evaluation of cloud gaming services. In *Proceedings of the 4th International Workshop on Perceptual Quality of Systems (PQS 2013)*. 1–5.

[30] Jim Mullin, Lucy Smallwood, Anna Watson, and Gillian Wilson. 2001. New techniques for assessing audio and video quality in real-time interactive communications. February (2001), 1–63.

[31] Shruti Patil, Yu Chen, and Tajana Simunic Rosing. 2015. GazeTube: Gaze-Based Adaptive Video Playback for Bandwidth and Power Optimizations. In *2015 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.

[32] Romass Pauliks, Konstantins Tretjaks, Kirils Belahs, and Romass Pauliks. 2013. A survey on some measurement methods for subjective video quality assessment. *2013 World Congress on Computer and Information Technology, WCCIT 2013* (2013). https://doi.org/10.1109/WCCIT.2013.6618758

[33] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. 2016. Optimizing 360 video delivery over cellular networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. ACM, 1–6.

[34] Yashas Rai, Marcus Barkowsky, and Patrick Le Callet. 2016. Role of spatio-temporal distortions in the visual periphery in disrupting natural attention deployment. *Human Vision and Electronic Imaging (HVEI)* (2016), 1–6. https://doi.org/10.2352/ISSN.2470-1173.2016.16HVEI-117

[35] Yashas Rai and Patrick Le Callet. 2017. Do gaze disruptions indicate the perceived quality of non-uniformly coded natural scenes? *Electronic Imaging* 2017, 14 (2017), 104–109. https://doi.org/10.2352/ISSN.2470-1173.2017.14.HVEI-124

[36] Michele Rucci, Paul V McGraw, and Richard J Krauzlis. 2016. Fixational eye movements and perception. *Vision research* 118 (2016), 1–4.

[37] Jihoon Ryoo, Kiwon Yun, Dimitris Samaras, Samir R. Das, and Gregory Zelinsky. 2016. Design and Evaluation of a Foveated Video Streaming Service for Commodity Client Devices. In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. ACM, New York, NY, USA, Article 6, 11 pages.

[38] Natela Shanidze, Saeideh Ghahghaei, and Preeti Verghese. 2016. Accuracy of eye position for saccades and smooth pursuit. *Journal of vision* 16, 15 (2016), 23–23.

[39] Ryan Shea, Jiangchuan Liu, Edith C-H Ngai, and Yong Cui. 2013. Cloud gaming: architecture and performance. *IEEE Network* 27, 4 (2013), 16–21.

[40] Ivan Slivar, Mirko Suznjevic, Lea Skorin-Kapov, and Maja Matijasevic. 2014. Empirical QoE study of in-home streaming of online games. (2014), 1–6.

[41] Rand S. Swenson. 2006. *Review of Clinical and Functional Neuroscience.* https://www.dartmouth.edu/~rswenson/NeuroSci/index.html

[42] Tobii. 2015. *Developer's Guide Tobii EyeX SDK for C/C++*. http://developer-files.tobii.com/wp-content/uploads/2016/03/Developers-Guide-C-Cpp.pdf

[43] Niraj Tolia, Mahadev Satyanarayanan, and David G. Andersen. 2006. Quantifying Interactive User Experience on Thin Clients. *Computer* 39, 3 (2006), 46–52. https://doi.org/10.1109/MC.2006.101

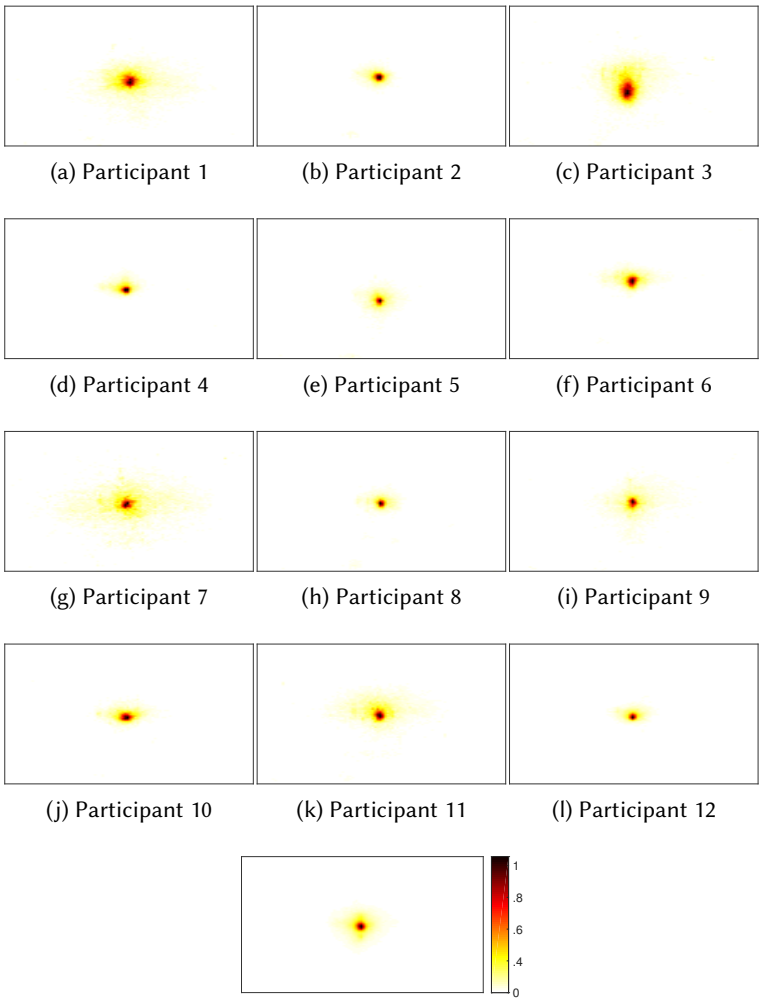
[44] Brian A Wandell. 1995. *Foundations of vision*. Vol. 8. sinauer Associates Sunderland, MA.

[45] Zhou Wang and Alan C Bovik. 2006. Foveated image and video coding. In *Digital Video, Image Quality and Perceptual Coding*, Hong Ren Wu and Kamisetty Ramamohan Rao (Eds.). CRC Press, Boca Raton, 431–457.

[46] Zhou Wang, Ligang Lu, and Alan. C. Bovik. 2003. Foveation scalable video coding with automatic fixation selection. *IEEE Transactions on Image Processing* 12, 2 (Feb 2003), 243–254. https://doi.org/10.1109/TIP.2003.809015

[47] Alireza Zare, Alireza Aminlou, Miska M. Hannuksela, and Moncef Gabbouj. 2016. HEVC-compliant Tile-based Streaming of Panoramic Video for Virtual Reality Applications. In *Proceedings of the 2016 ACM on Multimedia Conference (MM '16)*. ACM, New York, NY, USA, 601–605.

A GAZE HEATMAPS OF STUDY PARTICIPANTS



(a) Participant 1 (b) Participant 2 (c) Participant 3

(d) Participant 4 (e) Participant 5 (f) Participant 6

(g) Participant 7 (h) Participant 8 (i) Participant 9

(j) Participant 10 (k) Participant 11 (l) Participant 12

(m) Combined all participants

Fig. 19. Gaze tracking heatmaps of the user study participants.