TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JOOSE SAINIO

APPLICATIONS OF EYE TRACKING FOR REGION OF INTEREST HEVC ENCODING

Master of Science Thesis

Examiner: Ass. Prof. Jarno Vanne
Examiner and topic approved
on 8th August 2018

ABSTRACT

Joose Sainio: Applications of Eye Tracking for Region of Interest HEVC Encoding
Tampere University of Technology
Master of Science Thesis, 50 pages, 1 Appendix page
September 2018
Master's Degree Programme in Information Technology
Major: Embedded Systems
Examiner: Ass. Prof. Jarno Vanne
Keywords: video coding, High Efficiency Video Coding (HEVC), eye tracking, region of interest (ROI), Kvazaar HEVC encoder

The increase in video streaming services and video resolutions has exploded the volume of Internet video traffic. New video coding standards, such as *High Efficiency Video Coding* (HEVC) have been developed to mitigate this inevitable video data explosion with better compression. The aim of video coding is to reduce the video size while maintaining the best possible perceived quality. *Region of Interest* (ROI) encoding particularly addresses this objective by focusing on the areas that humans would pay the most attention at and encode them with higher quality than the non-ROI areas.

Methods for finding the ROI, and video encoding in general, take advantage of the *Human Visual System* (HVS). Computational HVS models can be used for the ROI detection but all current state-of-the-art models are designed for still images. Eye tracking data can be used for creating and verifying these models, including models suitable for video, which in turn calls for a reliable way to collect eye tracking data. Eye tracking glasses allow the widest range of possible scenarios out of all eye tracking equipment. Therefore, the glasses are used in this work to collect eye tracking data from 41 different videos.

The main contribution of this work is to present a real-time system using eye tracking data to enhance the perceived quality of the video. The proposed system makes use of video recorded from the scene camera of the eye tracking glasses and Kvazaar open-source HEVC encoder for video compression. The system was shown to provide better subjective quality over the native rate control algorithm of Kvazaar. The obtained results were evaluated with *Eye tracking Weighted PSNR* (EWPSNR) that represents the HVS better than traditional PSNR. The system is shown to achieve up to 33% bit rate reduction for the same EWPSNR and on average 5-10% reduction depending on the parameter set. Additionally, the encoding time is improved by 8-20%.

TIIVISTELMÄ

Joose Sainio: Katseenseurannan sovellukset mielenkiintoisen alueen HEVC-pakkaukselle
Tampereen teknillinen yliopisto
Diplomityö, 50 sivua, 1 liitesivu
Syyskuu 2018
Tietotekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Sulautetut järjestelmät
Tarkastaja: Ass. Prof. Jarno Vanne
Avainsanat: videonpakkaus, katseen seuranta, mielenkiintoinen alue, Kvazaar HEVC videokooderi

Videoliikenteen määrä Internetissä on räjähtänyt viime vuosina kasvavien videoresoluutioiden ja suoratoistopalvelujen takia. Tähän haasteeseen on vastattu kehittämällä uusia videopakkausstandardeja, kuten *High Efficiency Video Coding* (HEVC). Videonpakkauksen tarkoituksena on pienentää videon kokoa ja samalla pitää videon laatu mahdollisimman hyvänä. Erityisesti mielenkiintoisen alueen pakkaus pyrkii tähän pakkaamalla paremmalla laadulla alueet, joihin ihmiset kiinnittävät eniten huomiota.

Mielenkiintoisen alueen etsintään ja pakkaukseen tarkoitetut menetelmät hyödyntävät ihmisen näköaistimallia (HVS). Laskennallisia HVS-malleja voidaan käyttää mielenkiintoisen alueen etsimiseen, mutta olemassa mallit on suunniteltu pääasiassa still-kuville. Katseenseurannasta saatua dataa voidaan hyödyntää näiden mallien rakentamiseen ja varmentamiseen, mukaan lukien mallit, jotka ovat tarkoitettu videolle. Siksi tarvitaan luotettava tapa kerätä tätä dataa. Katseenseurantalasit mahdollistavat kyseisen datan keräämisen monipuolisemmin ja niitä onkin tässä työssä käytetty katsedatan keräämiseen 41:stä videosta.

Tämän työn pääasiallinen tavoite on esitellä reaaliaikainen järjestelmä, joka käyttää katseenseurantaa reaaliaikavideon pakkaamiseen paremmalla koetulla laadulla. Pakkaamiseen käytetään avoimen lähdekoodin HEVC videokooderia nimeltään Kvazaar. Subjektiivisella käyttäjätestillä todettiin järjestelmän parantavan laatua verrattuna normaaliin Kvazaarin. Myös objektiivisten testien mukaan järjestelmä käyttää jopa 33% vähemmän bittejä ja asetetuista parametreista riippuen säästöä saadaan keskimäärin 5-10%, kun objektiivisena metriikkana käytetään katseenseurannalla painotettua PSNR-metriikkaa (EWPSNR). EWPNSR vastaa paremmin ihmisen havaintokykyä kuin normaali PSNR. Lisäksi järjestelmä on 8-20% nopeampi kuin normaali Kvazaar.

PREFACE

This Thesis was written as a part of research conducted in the Laboratory of Pervasive Computing at Tampere University of Technology during the years 2017–2018.

I would like to thank my advisor Jarno Vanne for his input. I would also like to thank all members of the Ultra Video Group who helped me during the research, especially Kari, who helped with the eye tracking experiment and gave his comments to this Thesis.

Finally, I would like to thank my parents and siblings who supported me during my studies.

Tampere, September 2018

Joose Sainio

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

ACR	Absolute Category Rating
AVC	Advanced Video Coding
BD-BR	Bjøntegaard-delta bit rate
CTU	Coding Tree Unit
CU	Coding Unit
DC	Degradation Coefficient
DQP	Delta Quantization Parameter
DCT	Discrete Cosine Transform
EOG	Electro-OculoGraphy
EWPSNR	Eye tracking Weighted PSNR
EWSSIM	Eye tracking Weighted SSIM
GOP	Group of Pictures
HEVC	High Efficiency Video Coding
HMD	Head Mounted Display
HVS	Human Visual System
JM	Joint Model
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
MPEG-TS	MPEG Transport Stream
MSE	Mean Square Error
PC	Pair Comparison
POG	Photo-OculoGraphy
PSNR	Peak Signal to Noise Ratio
PUB-SUB	Publish-Subscribe
QP	Quantization Parameter
ROI	Region of Interest
SAD	Sum of Absolute Differences
SEI	Supplemental Enhancement Information
SSIM	Structural Similarity
TUT	Tampere University of Technology
UHD	Ultra High Definition
VCEG	Video Coding Experts Group
VOG	Video-OculoGraphy
VVC	Versatile Video Coding

1. INTRODUCTION

Currently, 73 percent of all consumer network traffic is reported to be video traffic and it is forecasted to raise up to 82 percent by 2021. In the same time frame, all Internet traffic is forecasted to grow threefold, causing total monthly video traffic to be approximately 228 EB (228 000 000 terabytes) [1]. Most of the video traffic is compressed since, e.g., ten minutes of 1080p30 raw RGB-video takes over 110 GB of storage space. Considering the volume of existing video traffic, being able to compress video further can reduce total network load significantly.

High Efficiency Video Coding (HEVC/H.265) [2] is the state-of-the-art video coding standard following the current mainstream standard *Adaptive Video Coding* (AVC/H.264) [3]. HEVC is designed to reduce the bit rate of video by 40 percent over AVC for the same visual quality but with a complexity overhead of 40 percent. HEVC was primarily introduced to cope with increasing video resolutions, especially 4K and UHD [4]. Video encoding is, in general, lossy compression that removes details from the video but tries to maintain good perceived quality. For example, human perception is skewed towards lower frequency details, meaning that high frequency details can be removed more freely without affecting the perceived quality [5].

Region of Interest (ROI) encoding methods are developed to reduce perceptual redundancy of the video [6]. ROI encoding has especially drawn interest recently since the tools for removing statistical redundancy have been improved significantly during the past two decades, making ROI encoding an easier way to increase encoding efficiency. The main purpose of ROI encoding is to produce video that is perceived well by the *Human Visual System* (HVS), with less bits than traditional encoding.

Although video encoding tries to maintain highest possible perceived quality within given bit rate constraint, the current tools do not always allow for it. Traditionally used quality metrics, which the encoder uses for optimizing the output, such as *Peak Signal to Noise Ratio* (PSNR) and *Structural Similarity index* (SSIM) [7] do not match with human perception [8]. To produce better-perceived quality, better models that resemble human perception are needed for encoders. One subdomain of computer vision, which is heavily dominated by neutral networks, is saliency [9]. Saliency maps could be used for detecting the areas that are most interesting to humans, however, they are mostly for two-dimensional data, computationally complex, and do not necessarily model human behavior accurately [10].

The most common fields of eye tracking are usability and human-computer interaction studies but it can also be used to model human vision [11]. Eye tracking can be used to generate the saliency maps, which dictate the interesting areas for the encoder. Eye tracking data gathered from twenty different persons can generate a saliency map, which is reasonably close to universally applicable model [10]. However, encoding can produce artifacts that shift the viewers' attention towards them, invalidating the saliency map. Therefore, any system trying to produce maximal perceived quality should pay attention to it.

The remainder of this Thesis is structured as follows: In Chapter 2, the concepts behind video encoding, subjective video encoding, and eye tracking are discussed. Chapter 3 goes over existing systems that make use of eye tracking as a part of a video codec. Chapter 4 describes the research methodologies used in this work. In Chapter 5, the proposed system using live eye tracking data to improve perceived video quality is described and the performance of said system is evaluated in Chapter 6. In Chapter 7, future possibilities of the system and the usage of eye tracking data are discussed. Finally, Chapter 8 concludes the work.

2. CONCEPTS OF VIDEO ENCODING AND EYE TRACKING

The work presented in this Thesis is based on three key concepts: 1) video encoding; 2) *region of interest* (ROI) video encoding; and 3) eye tracking. In Section 2.1, the most interesting video coding standards and the encoding flow of the current state-of-the-art HEVC standard are considered briefly. In addition, a comparison of open-source video encoders is made to choose the best one for this work. The main concepts of ROI encoding are explained in Section 2.2. Finally, the basics of eye tracking and the tools used for eye tracking are looked at in Section 2.3.

2.1 Video Encoding

The main purpose of video coding is to reduce the file size by removing redundancy. In traditional coding, first encoding and decoding data must preserve the contents, e.g., encoding the content of this Thesis and then decoding it only to find half of the words missing would be highly inconvenient. By default, all analogue data lose detail compared with real world, since analogue signals are on a continuous scale but they have to be clamped to discrete values when digitalized. For videos, each pixel is represented by a limited number of bits, i.e., they have a certain *bit depth*. Data such as sound, images, and video that are mainly consumed by human senses can lose further detail without humans noticing it, e.g., color shades that are very close to each other might be impossible to distinguish from each other and can be encoded to same value saving bits. In order to minimize the video bit rate as much as possible, it is not uncommon for video encoder to produce noticeable visual artifacts, particularly when targeting lower bit rates [6].

For most people, the RGB color space is the most familiar one but in video and image compression, the most commonly used color space is YUV. Like RGB, YUV consists of three different components, the Y-component is called luma and U and V-components are called chroma. The luma component represents brightness and chroma components color information. YUV was introduced as a natural move from black and white television to color television, since the luminance layer is already present in black and white image. Additionally, since humans are more sensitive to changes in brightness than color, the chroma layers can be subsampled. The most and second-most common subsampling schemes are 4:2:0 and 4:2:2, respectively. In 4:2:0 format, the resolutions of chroma layers are halved both vertically and horizontally whereas only the horizontal resolution is halved in 4:2:2 format.

Video coding standards are developed to produce the highest possible quality video for the given bit rate. Currently, the most well-known video formats are AVC, HEVC,

Versatile Video Coding (VVC/H.266) [12], VP9 [13], and AV1 [14]. AVC, HEVC, and VVC are all developed as a joint effort of ITU-T *Video Coding Experts Group* (VCEG) and the ISO/IEC *Moving Picture Experts Group* (MPEG) standardization organizations [15]. Since its finalization in 2013, HEVC has slowly started to replace AVC. For example, most modern smartphones from late 2014 onwards have had HEVC decoding hardware support [16], [17]. Existence of mobile hardware decoders fosters the adoption of a standard since they allow better energy efficiency, which is extremely important factor on popular mobile platforms. However, the largest hurdle for largescale adoption of HEVC has been the licensing issue that is caused by multiple patent pools in the technologies used by HEVC [18]. VP9 and its successor AV1, developed originally by Google and Alliance for Open Media thereafter, were introduced as royalty free alternatives to HEVC [1]. VVC is a foreseen successor to HEVC but its standardization process has just begun. Its most likely competitor AV1 is yet to have any use outside of tech-demos.

In this work, the rest of the discussion is focused on HEVC because it is the current state-of-the-art standard. In addition, some discussion about differences between HEVC and the preceding AVC standard is included to get familiar with the background of the field. One should note that this is not complete coverage of the standards but a quick overview of the parts that are relevant to this Thesis.

2.1.1 HEVC Overview

All modern video coding standards, including AVC and HEVC, are block based [4], i.e., the image is split into smaller blocks that act as encoding units. These blocks are called macroblocks and C*oding Tree Units* (CTU), respectively. Moving from static sized macroblock to variable size CTUs is one of the most noticeable changes between AVC and HEVC [4]. The most common size for CTUs is 64×64 pixels, though standard allows them to be smaller. CTUs can contain one *Coding Unit* (CU) or they can be split into multiple CUs with the smallest allowed size being 8×8, conversely the macroblocks are always 16×16. Figure 2.1 depicts how images are split into CTUs in HEVC and Figure 2.2 how CTUs can be further partitioned into CUs.

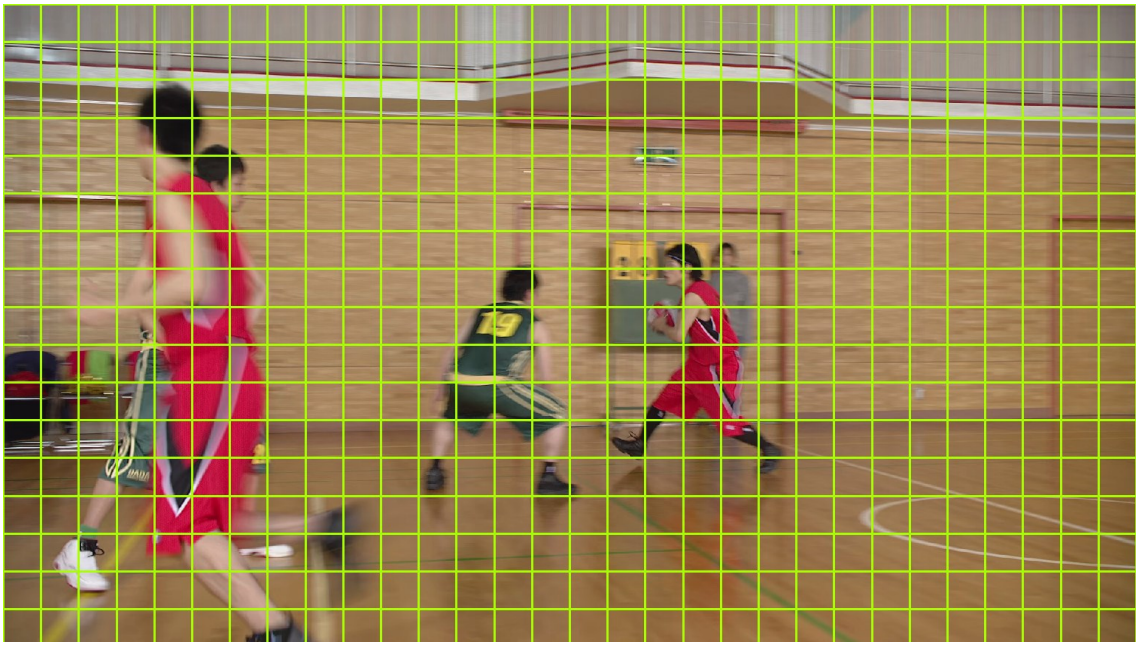

Figure 2.1. *Partition of a video image into coding tree units.*

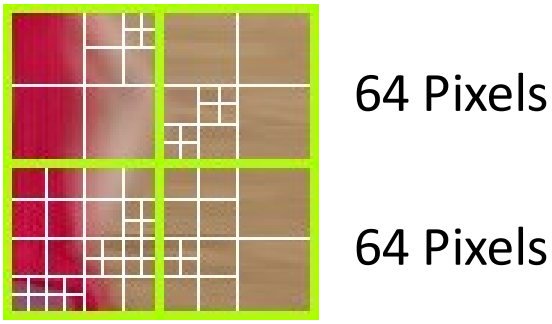

Figure 2.2. *Coding tree unit split into multiple coding units.*

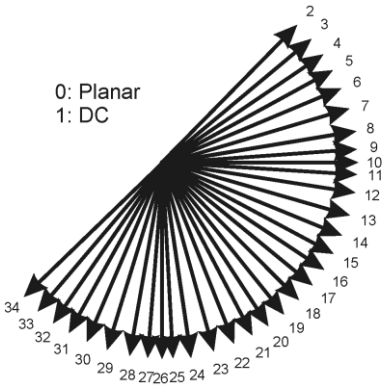

Figure 2.3. *Angles of HEVC intra prediction modes [4].*

HEVC and its predecessors have two different prediction modes: intra and inter. Intra prediction works within a single video frame by copying pixel values from surrounding CUs. In HEVC, intra prediction has 35 different modes [4]: planar, DC, and 33 angular modes. The different angular modes are visualized in Figure 2.3. In planar mode, a gradient of the edge pixels is used whereas average of all edge values is used for prediction in DC mode. In angular mode, the pixel values are copied according to the angle.

The main difference between a still image and video is a temporal dimension, which adds a lot of redundancy in video and favors using inter coding as a prediction type. With inter prediction, the currently encoded CU is compared with co-located areas in previous

frames to find an optimal motion vector for the CU. After the prediction mode is chosen, the prediction is subtracted from the original frame to create a residual.

The residual is encoded using transform coding. In HEVC, the residual is transformed using *Discrete Cosine Transformation* (DCT) or in the case of 4 × 4 luma blocks with *Discrete Sine Transform* (DST). The transformed coefficients are quantized to reduce their size. A *Quantization Parameter* (QP), having a range from zero to 51, is used to determine the quantization level. A higher QP means that the coefficients are quantized to fewer values, resulting in more distortion to the encoded video. Generally speaking, QP has the highest factor to the video bit rate out of all other tools and steps of the encoding.

Both AVC and HEVC allow varying QP at macroblock and CU granularity, respectively [20], [21]. The varying QP can be used for rate control or ROI encoding. Rate control belongs to the non-normative part of the encoding process but it is necessary in real-life use cases. Rate control ensures that the bit rate of the video stays nearly constant by allocating the bits of the video, either at *Group of Pictures* (GOP) level or at picture level. Most live Internet videos are delivered at a constant bit rate because otherwise situations like scene-cuts, which require many bits to maintain the quality, would cause buffering.

Both AVC and HEVC support *Supplemental Enhancement Information* (SEI) messages. The purpose of SEI messages is to allow embedding additional information into the video bit stream, such as color space information and timing information for the frames. At the beginning of the SEI message, there is a sixteen bytes long unique identifier that is used to differentiate between different kinds of SEI messages, followed by the payload of the message.

2.1.2 Comparison of Open-Source Video Encoders

The encoders compared in this Section include *Joint Model* (JM) [22], x264 [23], *HEVC Test Model* (HM) [24], Kvazaar [25], Turing [26], and x265 [27]. A summary of all these projects is given in Table 2.1.

The AVC encoders, JM and x264, are also included because the previous work explored in Section 3 is mainly based on them. JM is a test model of AVC, developed by the standardization group. It implements all features specified in the standard and little attention is paid to its complexity suppressing any real-life use. x264 is probably the most well-known practical open-source AVC encoder. It is developed by the non-profit organization VideoLAN and it is often used as baseline for encoder comparisons [28]. Practically all research on AVC is based on these two encoders. Theoretical proposals are often implemented in JM whereas more practical ones address x264. Especially, when complexity is concerned JM is a poor choice, since its acceleration might have a poor correlation with real applications.

Table 2.1. *Overview of the open-source video encoders.*

Encoder	JM	x264	HM	x265	Turing	Kvazaar
Standard	AVC	AVC	HEVC	HEVC	HEVC	HEVC
License	BSD	GPL2/ Commercial	BSD	GPL2/ Commercial	GPL2.0	LGPLv2.1
Coordinator	JCT-VC	VideoLAN	JCT-VC	MulticoreWare	BBC	TUT
Language	C	C	C++	C++	C++	C
Commits	N/A	2901	4959	12368	89	2592
Last Commit	06/2015	01/2018	04/2018	Active	11/2017	Active

Similarly to JM, HM is a test model implementing all features of HEVC. x265 is developed by MulticoreWare and is currently probably the best-known practical open-source HEVC encoder [27]. Turing codec is a newcomer to open-source HEVC encoders, developed by BBC [26], and unlike x265 it is a complete codec instead of an encoder only. The reported commit count in Table 2.1 is most likely undervalued since the development seems to be mostly internal. Kvazaar is an award-winning open-source HEVC encoder developed by our Ultra Video Group at Tampere University of Technology [29].

Because this work particularly addresses real-time systems, the real-time performance of the encoders is evaluated. In Table 2.2, the version and command line arguments of the applied versions of Kvazaar, Turing, x264, and x265 are tabulated. HM and JM are not included in the evaluation because they are far from real-time. x264 is included as a baseline to display a concrete difference between the performances of the HEVC and AVC standards. For all the chosen encoders, the fastest preset is used, with a low-delay or equivalent configuration if available.

Table 2.2. *Used encoder versions and command line arguments.*

Encoder	Commit	Command line arguments
Kvazaar	`4fb1c16c6198`	--preset ultrafast --owf 0
Turing	`5d44bd79b3be`	--speed fast --concurrent-frames 1
x264	`7d0ff22e8c96`	--preset ultrafast --tune psnr --non-deterministic --no-scenecut --bframes 0 --b-adapt 0 --sliced-threads --no-mbtree --rc-lookahead 0 --sync-lookahead 0 --force-cfr 0
x265	`538f3ad860a5`	--preset ultrafast --tune psnr --bframes 0 --b-adapt 0 --rc-lookahead 0 --no-scenecut --no-cutree --frame-threads 1

The performance figures of the encoders are tabulated in Table 2.3. A more thorough explanation on how these values were obtained is given in Chapter 4. The speedup is reported over the frame rate of x264. Bit rate tells how many bits the encoder requires to produce similar quality. All the benchmarked HEVC encoders are clearly slower than x264, as expected, with x265 being nearly three times, Kvazaar four times, and Turing

over 25 times slower. Still, both Kvazaar and x265 are able to encode 1080p30 video in real time on the test machine. Although Turing is able to provide better quality than Kvazaar and x265, it is simply too slow for real-time coding. It should be noted that the use of low-delay settings reduces the amount of parallelism available to the encoder, so increasing the delay improves encoding speed but not enough to make Turing competitive. Since the quality of Kvazaar and x265 is practically the same x265 would be a better choice if the selection was made on the basis of performance measurements.

Table 2.3. *Performance of the HEVC encoders compared with x264 on the standard HEVC test sequences with Intel i7-5960x processor.*

	Kvazaar		Turing		x265	
Class	**Bit rate**	**Speedup**	**Bit rate**	**Speedup**	**Bit rate**	**Speedup**
hevc-A	-54.8 %	0.26×	-76.8 %	0.04×	-51.3 %	0.34×
hevc-B	-56.0 %	0.24×	-81.9 %	0.04×	-56.4 %	0.36×
hevc-C	-44.2 %	0.22×	-72.8 %	0.03×	-43.1 %	0.35×
hevc-D	-35.3 %	0.28×	-72.1 %	0.05×	-36.5 %	0.51×
hevc-E	-58.5 %	0.24×	-81.6 %	0.04×	-63.6 %	0.36×
hevc-F	-13.6 %	0.20×	-54.8 %	0.03×	-12.2 %	0.34×
Total	**-42.6 %**	**0.24×**	**-73.0 %**	**0.04×**	**-42.9 %**	**0.38×**

The other important aspect for the encoder selection is their compatibility to ROI encoding. x265 supports multiple different rate control schemes but none of them is targeted at ROI encoding. The closest tool of x265 is adaptive quantization that tries to work around the flaws of the traditional rate control. Despite that the rate control scheme of Kvazaar is not as sophisticated, Kvazaar supports *Delta QP* (DQP) matrixes that allow setting the QP values in a way that the ROI areas are encoded with higher quality. This feature clearly advocates implementing the proposed ROI scheme in Kvazaar.

2.2 Region of Interest (ROI) Video Encoding

Since both VVC and AV1 [30], [28] increase the coding complexity massively over their predecessors, it is obvious that most traditional coding techniques have been exhausted. Further bit rate savings can be achieved by either introducing more and more complex coding tools or by changing the approach. In the former case, the complexity grows unsustainably, which is evident from the exponential growth in complexity with each standard. ROI encoding is a prime example of the latter, since video encoding tries to produce the best perceived quality at a given bit rate, the simplest way to achieve it is to allocate the available bits to the areas that humans find salient.

ROI encoding seeks to select the interesting areas and encode them with higher quality. ROI encoding does not specify how the quality of the video should be controlled regarding the ROI. The methods for changing the quality can be divided into two distinct

categories: 1) preprocessing the image; and 2) embedding the quality change into the encoding process [6]. Preprocessing methods most commonly apply low-pass filter or Gaussian blur to the image. They are not standard dependent and can be applied to any encoder. However, their effect is usually limited, since for example, applying a Gaussian blur to an image is inelegant method thus becomes noticeable very fast. Thus, preprocessing is not a commonly used method but it can be used to augment other methods or as an easy to implement method to test ROI detection algorithms.

The more common and effective method for implementing the varying quality is embedding it into the encoding process. The two most common approaches are to use either non-uniform DQP matrixes or a custom rate control system [6]. With DQP matrixes, a lower QP value is set to the areas inside the ROI. Rate control implementations replace the native rate control algorithm of the encoder with a custom one that allocates more bits to the ROI area at the expense of the non-ROI areas. A rate control system is more complex to implement than DQP matrixes, but it allows finer control over the result. In addition, it is the only option for applications that require the bit rate to be limited to a certain level.

While quite a lot of research effort has been put into ROI encoding, surprisingly little research exist on how the quality should change around the ROI. Arndt and Antons conducted a test using eye tracker to recognize where the viewers were looking and displayed higher quality video around the gaze center [31]. The purpose of the test was to find out how the radius of the higher quality area around the ROI affected perceived quality of the video. The degradation in quality was a sharp drop around the ROI and they noted that all test subjects noticed the sharp drop at least for some of the test conditions. Thus, a real system should not use a sharp drop but linear or logarithmic rate of degradation. Changing the degradation method would most likely mean that the radius could be smaller than what was found in their experiment. In addition, the test was conducted with only one 10 minute sequence. It would also be useful to examine differences between low and high foreground/background movement. For example, watching a newscast with a static background versus a background with action where the viewers gaze is more likely to drift around. Finally, the higher quality area was placed at the area where the person is watching. This approach requires that a watcher is using an eye tracker, but a model that tries to find out where the user is watching cannot accurately predict gazes of every single person. Some comparative research has been done between linear and logarithmic rate of quality degradation [32]. The results suggest that logarithmic degradation is better than linear, but no in-depth results are available. However, together with the fact that HVS is logarithmic [6] using logarithmic degradation over linear should be justifiable until further studies on the topic are conducted.

In ROI encoding, one can find it challenging to assess the quality of the result [6]. The best way to assess the quality would be to conduct subjective user tests but those are time consuming, expensive, and require extra care to make sure that the results are credible.

However, objective metrics such as PSNR and SSIM reflect HVS poorly [8], but they are used in encoders due to their computational simplicity. For example, when comparing bit rates of HEVC and AVC for the same quality HEVC requires more bits with PSNR than with subjective metrics [33]. Whereas evaluating different objective quality metrics over the subjective ones is out of the scope of this work, care should be taken when evaluating any ROI encoding system.

A major challenge of ROI encoding is the actual selection of the ROI. Handpicking can be used if the only goal is to test a rate control scheme or when comparing a model for picking the ROI. However, there is a caveat with handpicking the ROI areas, since the handpicked areas may not represent HVS. Eye tracking can be used to detect human gaze points for a certain sequence. About twenty people are required [10] to achieve gaze maps that do not change noticeably even if the number of viewers is increased. However, generating the ROI with eye tracking may not always be possible. A common feature for all the ROI generation methods is that they try to imitate the HVS.

2.2.1 Human Visual System (HVS) and Foveation

Human Visual System (HVS) is the backbone of any perceptual video encoding effort [6]. The 2-5 degree area at the center of human vision, called fovea, is the main concept of the HVS [11]. Outside of the fovea, the perception reduces gradually towards the edge of the vision. This particularly holds with stationary objects while moving objects can grab a person's attention even from the edges of the vision. Foveation is a concept where parts of image are expressed at lower resolution, similarly as the HVS perceives it. The most common example of foveated images are map applications where the image resolution gets gradually better as the user zooms in. Foveated pictures are created by foveation filters.

The HVS can make use of two different patterns: bottom-up and top-down [11]. The bottom-up pattern is mostly the subconsciousness guiding the gaze, e.g., bright colors draw attention. On the other hand, top-down is mostly conscious decision: prior knowledge or looking for something specific in the scene. Most of the time the HVS operates in the bottom-up mode, which makes humans often look at moving objects.

2.2.2 History of Computational Saliency Models

Whereas the HVS model has been an interesting topic since the late 19th century, [11] the first computational model for it was introduced in 1998. It is called the IKN-model, named after its creators Itti, Koch, and Niebur [34]. The model was originally created to find possible interesting image areas in order to apply more computationally complex algorithms for them. The model is purely bottom-up because the HVS also works in bottom-up mode when scanning the image. The model uses Gaussian pyramids of luminance, color, and orientation to calculate the saliency map. The model works on

biological basis where the scan order of the salient objects is addressed. The first object is considered to be the most salient one. Then, based on the distance to the previous salient object the next most salient is chosen and a scan-path for the image is formed.

The original IKN model was meant for images but in [35] the model was extended to work with video data by adding temporal effect to the luminance, color, and orientation pyramids. The model picks one to five different ROIs based on the parameters given to the model. The actual ROI encoding is done by applying varying Gaussian blur to the raw video frame before encoding. The model was verified by using eye tracking from eight subjects, which may not be enough [10]. Furthermore, the perceived video quality was not verified, which leaves the credibility of the results questionable.

After the IKN model, several other computational models have been created, e.g., [36] and [37]. The work in [36] focuses on detecting the salient objects from an image. The detection is done by using a conditional random field with maximal logarithmic sum of different features as the optimization function. The features are from three categories: local, regional, and global. The local feature is simply an average of different levels of contrast pyramids. The regional feature is calculated by forming a histogram of the colors in the image and then the supposed salient object's histogram is compared with its surroundings. Finally, the global feature is formed by calculating the spatial variance of color in the image. Conversely, [37] focuses on how saliency is distributed on an image. It uses logarithmic representation of Fourier transforms of 64 by 64 blocks in the image. The saliency map is formed by deducting the average Fourier transform of multiple images from the transformed image and then applying an inverse Fourier transform to the remaining values and smoothing the result.

Recently, solutions using Neural Networks have become the state-of-the-art methods for creating saliency models, most prominently [38] and [39]. The drawback with [38] is that it only segments the salient object out of the image similarly to [36]. Assigning only one large area completely salient is not very usable in video coding, since the actual saliency is most likely heterogeneously distributed inside the object. For example, if the salient object is a house, people are more likely to look at specific parts of the house such as doorways and windows. On the other hand, [39] can generate very natural heat maps compared with real human fixations but it is primarily designed for images.

Surprisingly little effort has been put into saliency models meant specifically for videos. There exist no neural networks that would try to generate models for improving the perceived quality or influence the rate control of video encoding. The model presented in [40] is probably the most promising among the models released in recent years. It is heavily based on the IKN model with global motion compensation added for the temporal effect, since the IKN model does not perform well when the camera is moving [40].

2.3 Eye Tracking

The English term “eye tracking” is somewhat confusing, since it commonly refers to gaze tracking rather than tracking eyes only. Throughout this Thesis, eye tracking refers to its broader definition, but tracking eye movements is considered here first.

Eye trackers are devices used for tracking the eye movements. They can be split into four broad categories based on the measurement methodology: 1) Electro-OculoGraphy (EOG); 2) scleral contact lens/search coil; 3) Photo-OculoGraphy (POG) or Video-OculoGraphy (VOG); and 4) video-based combined pupil and corneal reflection [11].

EOG was developed in the 1970s and it relies on measuring electrical differences on the skin when the wearer shifts their gaze, by using electrodes placed on skin near the eyes. The main disadvantage of EOG is that it only tracks the eyes and for actual gaze tracking the user’s head has to be held in place or the movement has to be tracked.

Like EOG, the contact lenses also record eye movement only but with a contact lens and a measuring device placed on the eye [11]. Contact lenses provide the best accuracy, but they are the most intrusive method for eye tracking and cause discomfort for the user.

A difference between POG and VOG methods is the lack of temporal dimension in POG. Otherwise, they group together since both methods use features of the eyes under movement and corneal reflection usually from infrared light [11]. The features can be either detected manually, or automatically from a video, but manual detection can be extremely tedious and error prone. In Figure 2.4, corneal reflections from an infrared light and automatically detected pupil are visualized. Like the previous methods, POG and VOG are only suitable for eye and not gaze tracking on their own.

Video-based combined pupil and corneal reflection is an advancement of the POG and VOG methods. It introduces easy ways to turn the eye tracking into gaze tracking [11]. Cameras and image processing software are used to track the head position relative to eyes and the systems can be worn or table mounted.

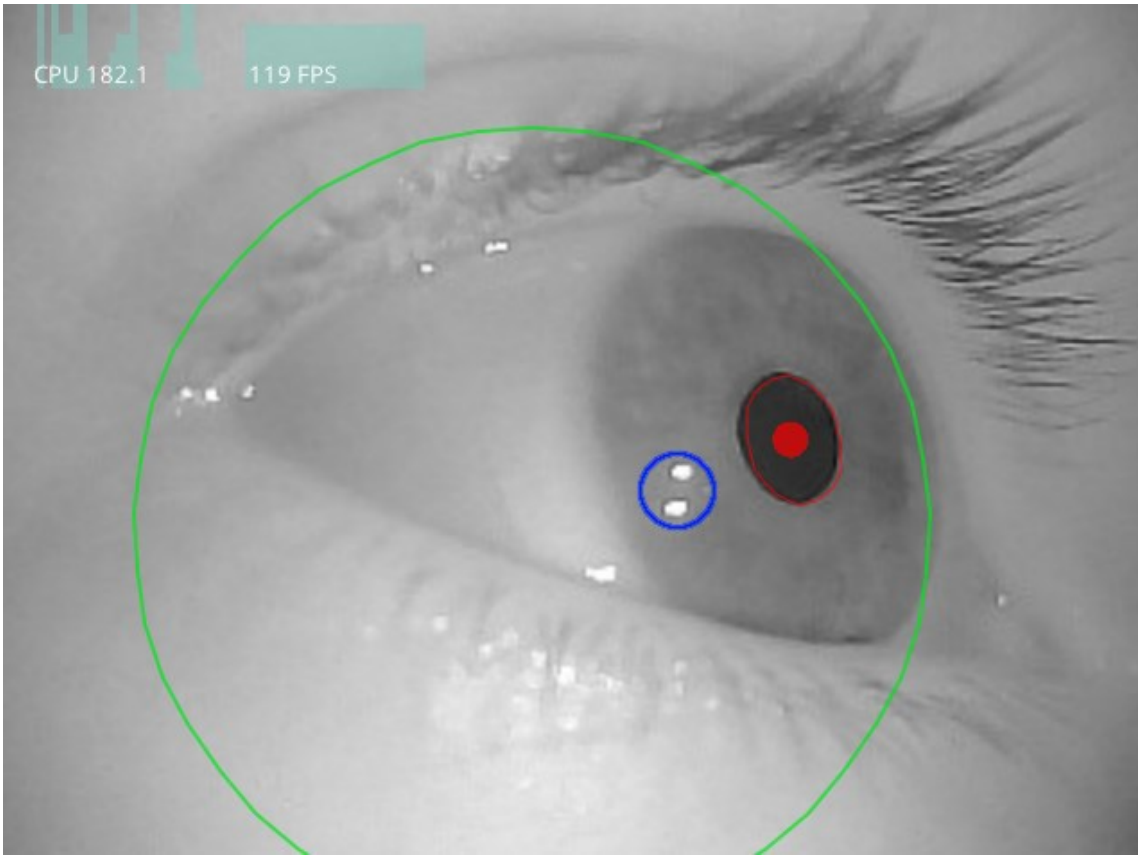

Figure 2.4. *Human eye, under infrared illumination, with corneal reflection highlighted in blue, the pupil in red, and the whole eye based on calculated model in green.*

Both the EOG and corneal contact lenses are still used in some cases but in the last decade all commercial eye trackers have been the pupil and corneal reflection type. These can be split into three distinct classes: 1) embedded to a *head mounted display* (HMD); 2) eye tracking glasses; and 3) screen-based solutions. HMDs are used for virtual reality and while virtual reality is out of the scope of this work, it could be considered in the future. Eye tracking glasses are worn like regular glasses. The glasses have at least infrared cameras for eyes and typically a scene camera. The gaze data is relative to the video from the scene camera. Screen based solutions can be further split into two subclasses, either the sensor is integrated into the display or the sensor is a separate sensor unit that can be used with different displays. Figure 2.5 depicts a distinct sensor on a laptop and a person wearing eye tracking glasses.

Currently, there are multiple companies offering eye tracking equipment for commercial and research use, most notably Tobii [41], SR-Research [42], Pupil Labs [43], and Ergoneers [44]. Basic information about these companies and what type of eye trackers they offer are tabulated in Table 2.4. Tobii is probably the best-known eye tracking equipment manufacturer and it is the only one offering all the different equipment types. SR-Research is the oldest company, which is fairly evident from their eye tracking

Figure 2.5. *A separate eye tracking sensor attached to a laptop [41] and a person wearing eye tracking glasses.*

glasses; they are more intrusive than the ones offered by the other companies. Pupil Labs is a newcomer to the eye tracking field and unlike others it offers completely open-source software stack and even partially open hardware. All the others only offer software development kits and do not allow modifying the core of their software. Ergoneers focuses on bringing eye tracking to automotive and transportation but it also offers eye tracking for more general use cases.

Table 2.4. *Different eye tracking equipment manufacturers and their products.*

Manufacturer	**Tobii**	**SR-Research**	**Pupil Labs**	**Ergoneers**
Based on	Sweden	Canada	Germany	Germany
Since	2001	1991	2014	2005
Screen based	Yes	Yes	No	No
HMD	Yes	No	Yes	Yes
Glasses	Yes	Yes	Yes	Yes

The screen-based solutions, where the sensor is integrated into the display, provide the best tracking quality but they tend to be the most expensive solutions of up to hundred thousand euros. In addition, since they are integrated into the display they are only usable for experiments where the screen is suitable for the test. The biggest weakness of separate sensors is that most of them are designed at most 24-inch displays [45]. The screen size limitation becomes an issue with *Ultra High Definition* (UHD) resolution screens, since those are practically always over 24 inches. Eye tracking glasses allow the greatest flexibility when it comes to test environment. However, their accuracy is a little bit worse than that of the screen-based solutions since the sensor unit is attached to the user and maybe able to move a bit with the user's head movements.

Eye tracking glasses were chosen due to them being possible to use with large enough screens and the extra accuracy provided by integrated screen-based solutions is not

necessary when combining data from multiple people together. The Ergoneers' glasses are discarded because their software stack is completely closed source. Parts of the Tobii's software stack is open source but Pupil Labs' software stack is completely open source, as well as most of the hardware is also open source. In addition, the Pupil Labs' glasses have more competitive price, so they are chosen.

3. EXISTING EYE TRACKING SOLUTION FOR VIDEO CODECS

Currently, there exist a few systems that use eye tracking for either video encoding, decoding, or presentation. The primary purpose of all these systems is to improve the perceived quality of the video over traditional systems with similar bit rate. In this chapter, a couple of most notable prior-art systems in the field are reviewed: a foveation based codec is presented in Section 3.1, Section 3.2 presents a system for delivering higher quality video based on user gaze. Finally in Section 3.3an encoding system that uses offline gaze data.

3.1 Foveated Video Codec

The foveated video codec introduced in [46] is most likely one of the first, if not the first, systems that makes use of eye tracking data in video encoding. It improves video quality during preprocessing and encoding, i.e., unlike most other systems it uses both methods described in Section 2.2. The system is loosely based on the old H.263 standard. Because H.263 does not support non-uniform QP values by default the decoder side of the system had to be also modified for it.

The preprocessing is done by applying a foveation filter to the image before the encoding [46]. The foveation filter consists of multiple low pass filters with varying cutoff frequencies. The strongest filter is only applied to the areas furthest away from the gaze centers. Symmetric and circular-symmetric filters are used to smooth out the areas between different low-pass filters [46].

In the encoder, a conventional motion estimation algorithm is replaced by a hierarchical algorithm where the optimal motion vector is searched using a pyramid of down sampled images [46]. The similarity criterion of the algorithm is *Sum of Absolute Differences* (SAD) that is weighted based on how far it is from the gaze center. To further improve the coding gain, the rate control algorithm is also optimized using the gaze data. The QP values are set lower around the gaze centers and they degrade towards the edges of the frame.

The system is mostly theoretical and all parts are simulated separately, so it is still questionable whether it can be used in a real application. Since there is no real implementation it is impossible to know the total computational complexity of the system, although the complexity of most of the parts is analyzed to be reasonably simple. Additionally, the system is based on pre-obtained gaze centers from an eye tracker, which raises further question about the real-time applicability.

3.2 Gaze Influenced Video Delivery

A couple of prior-art systems use pre-encoded video of different qualities and eye tracking to enhance the perceived quality of a video [31], [47]. Both systems transfer the gaze data to a server that sends back the higher quality video only on the area where the viewer is looking. Figure 3.1 depicts the basic idea behind these systems.

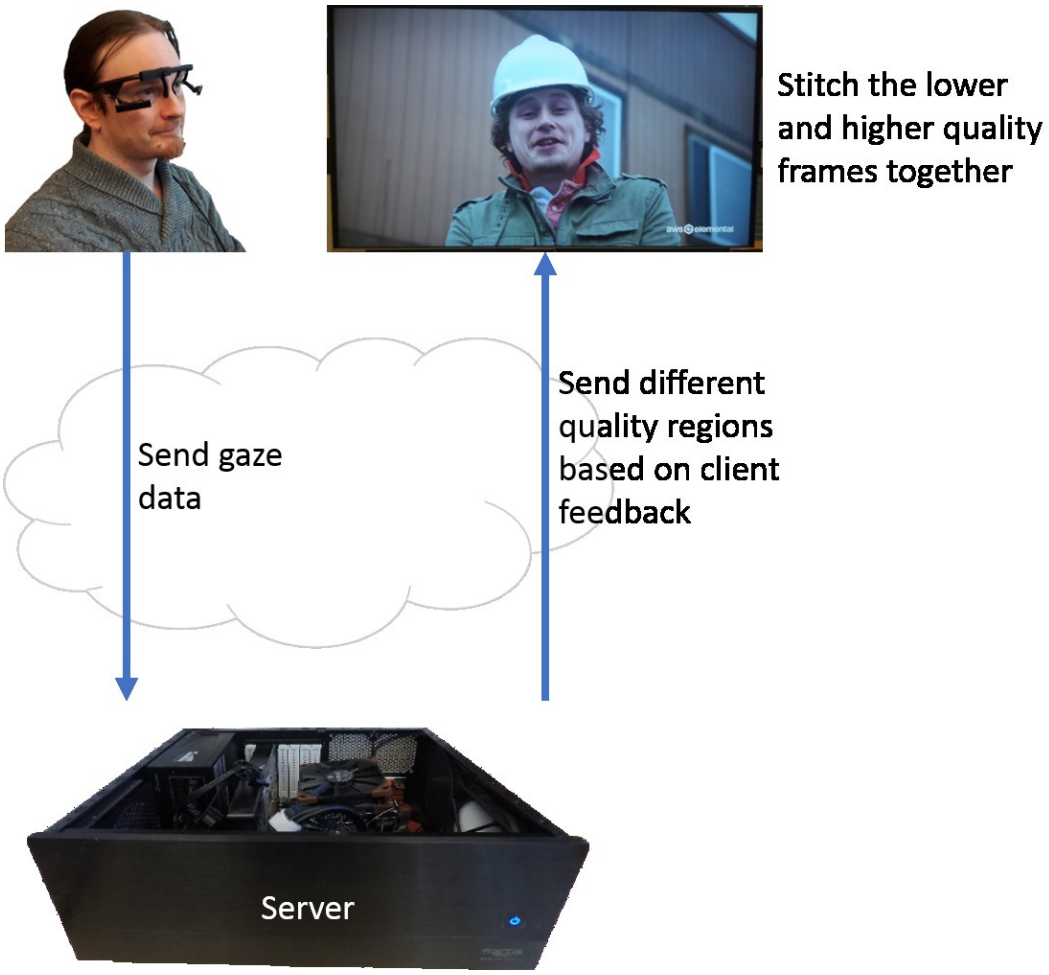

Figure 3.1. *Architecture of the systems.*

The system presented in [31] was designed for two reasons: to test system that serves better quality video around the users gaze and to test how the radius of the higher quality area affects the perceived quality. The server side has both the higher and lower quality videos already encoded. The lower quality frame is always sent together with a cropped out region of the higher quality frame and the client stitches the videos together for playback. In the experiment, both the server and client ends were on the same machine. The arrangement is reasonable for testing but the results cannot be used as is for any real-world system because latencies are not considered. All online video services have at least

some latency due to network delay and buffering the data at the client. Considering that human gaze can move very fast, latency should be accounted for such a system.

The system in [47] uses eye tracking to improve perceived video quality. It is not meant to be a complete system that considers all latencies, but some attention is paid to it. For example, the lowest quality layer is sent like in usual streaming services, i.e., it is buffered on the client side, while the higher quality layer is sent live based on the gaze. The video frames are split into uniform size cells that are used for serving the different quality areas. The cells are stitched on the client side into a full frame. The eye tracker is realized with cheap webcams, whose tracking quality is poor but good enough for the application. Also, the system works with mobile devices, since the front camera of the smartphone can be used for the eye tracking [47]. Overall [47] is superior for a real-world solution but the main contribution of [31] is the effect of ROI size to perceived quality.

3.3 Eye Tracking for Semiautomatic Saliency Model

A system that uses eye tracking as a part of semiautomatic saliency model is introduced in [10] and further refined in [48]. It is most likely the first semiautomatic saliency model. It represents a middle ground between an automatic saliency model and collecting extensive eye tracking data. Using human gaze data enables combining both top-down and bottom-up elements of the HVS into the model rather cheaply. Eye tracking data from a single observer is used as a base for building the saliency model. The gaze data from a single point in time is propagated backwards and forwards using the motion vector field of the next or previous frame. It should be noted that the motion vector field is not the one calculated during the encoding but the saliency model generation is done completely before the encoding process.

The main difference between [10] and [48] is how the actual ROI encoding is implemented. Both of them use x264 as a baseline. In [10], the video is first encoded regularly and the QP map is extracted during the encoding by using the multiple pass feature of x264. The extracted QP map is then modified according to the saliency model so that QP is decreased at salient areas and increased elsewhere. Conversely, in [48] a custom rate control algorithm is implemented. The idea behind the introduced algorithm is simple: assign X percent of the bits to Y percent of the most salient areas in the video. However, the implementation is not straightforward, since without multiple encoding passes it is difficult to estimate bit allocation to each frame. However, a rough estimation can be done based on the QPs chosen by the native rate control, which allows for a concrete calculation for bit allocation.

In both systems, there is a sharp drop in quality outside of the salient areas, so allocating bits too aggressively to the salient areas would cause noticeable difference between the salient and non-salient areas. Furthermore, the saliency model has to be generated before

encoding, especially in [10], because it requires multiple encoding passes. Hence, both of these systems are primarily intended for offline encoding to maximize coding quality.

Even though the semiautomatic model is interesting, maybe even more interesting are the findings from using the model for encoding. In both cases, *Eye tracking Weighted SSIM* (EWSSIM) was used as an objective quality metric whereas subjective tests are additionally performed for [48]. EWSSIM is based on *Eye tracking Weighted PSNR* (EWPSNR) [49]. A clear advantage over the native rate control of x264 is shown. The model loses to two observers when comparing with the ground truth eye tracking data but when used for encoding the model produces better results. More impressively, a significant improvement in bit rate is reported with subjective test [48]. The improvement is higher when the target bit rate is lower because the different methods might produce visually similar results at higher bit rates and only the details that humans do not pay attention differ. However, at least with the rate control parameters chosen for the subjective test the quality seems to be worse for high enough bit rate than with regular x264, questioning the model usability for generic cases.

In general, both systems present valuable aspects but leave many questions in the field of ROI encoding unanswered. The critical question that keeps the model from being used, is how to select the ROI size and how much bits should be allocated to the area, and this problem is acknowledged by the authors [10]. Additionally, the authors do not consider the computational complexity of the model nor the complexity that is added to the encoding process. Although the math seems simple, it would have been good to include complexity analysis for the model. The authors suggested that removing the back propagation component from the model would make it suitable for real-time applications, but no results were shown [48]. Moreover, even if the model works without the back propagation the complexity might prove problematic.

4. RESEARCH METHODOLOGY

Three primary factors are typically measured in video encoder evaluations: bit rate, quality, and complexity. Bit rate is the number of bits the encoder outputs per unit of time, usually a second, quality equals distortion between an original and a coded picture, and complexity refers to the computational complexity, i.e., coding speed of the encoder. In Section 4.1 the methodology for obtaining the objective quality measurements is explained. Conversely, Section 4.2 explains the methodology for subjective quality evaluation. In Section 4.3 the complexity measurement is explained. In Section 4.4 the test material used for the measurements is introduced. Finally, Section 4.5 goes over the eye tracking data gathering process.

4.1 Coding Efficiency

Bit rate and quality are meaningless without each other. For example, if bit rate is not limited one could just pass the uncompressed video without any quality degradation. Thus, these two variables can be combined to a single metric called coding efficiency. If the bit rate is reduced for the same quality or quality improved for the same bit rate, the coding efficiency is better. Any improvements to coding efficiency often come at the cost of complexity, i.e., if the coding efficiency is improved, the encoding speed tends to be lower.

Coding efficiency is computed from bit rate and quality. Bit rate for a video sequence requires counting the total number of bits the encoder uses for the sequence. For objective quality metrics, the quality is given by distortion to the original frame. The two most common metrics used for quality computation are PSNR and SSIM [7]. PSNR is the simplest metric and is given by *Mean Square Error* (MSE) as

$$\mathrm{PSNR} = 10 \cdot \log_{10} \frac{(2^{\mathrm{B}} - 1)^2}{\mathrm{MSE}},$$

where MSE is normalized to the bit depth *B* of the video. PSNR is in logarithmic scale. Lower MSE means less distortion and the converse applies to PSNR, i.e., when distortion approaches zero PSNR approaches infinity.

SSIM is significantly more computationally complex than PSNR but it also matches human perception in many cases better than PSNR [7]. SSIM uses means, variance, and covariance of a window around each pixel. Usually, the window size is 11 by 11 and Gaussian weighting is used for the window [7]. Even though PSNR and SSIM are the most used algorithms they are originally designed for images; thus, they do not consider the temporal dimension at all.

Both PSNR and SSIM can be weighted with eye tracking data for ROI coding assessment [10], [49]. PSNR is weighted by the individual square errors before MSE is calculated whereas SSIM is weighted by each value of SSIM before calculating the average for a single frame. In both cases, the weighting was done using fixations, which are more stable than raw gaze points. However, no fixation is registered when a moving object is being watched. Since moving objects tend to draw watcher's attention, gaze directed towards them should be included. Because of this, the weighting in this work is done using the raw gaze point by convolving a Gaussian kernel with the size equivalent to 5 degrees of vision, i.e., the largest fovea size over the gaze points. In case no gaze points are found for a specific video frame, the MSE or SSIM of the frame is not weighted. Weighting turns both metrics to resemble the HVS more closely and will add slight temporal component, since humans tend to focus on a single object at a time.

Both PSNR and SSIM operate on a single-color plane at the time so the results of luma and chroma planes have to be combined. Since luminance has higher priority in the HVS, the planes are weighted at 6:1:1 ratio. The average PSNR and SSIM for the whole video is an arithmetic mean of all encoded frames [21]. PSNR does not work if the encoder manages to encode any single frame at zero distortion because PSNR would be infinite for that frame causing the PSNR to be infinite for the whole sequence. However, in practice this happens rarely unless the encoder is explicitly set to lossless mode, which is not the case in this work.

The *Bjøntegaard-delta bit rate* (BD-BR) [50] has been developed as a single metric that encapsulates both bit rate and distortion differences of two encoders to a single quantity. First, the test sequences are encoded by both encoders using four different QP values. The HEVC common test conditions [51] define these QP values as 22, 27, 32, and 37. They are also used in this work. The measured bit rate is converted to logarithmic scale to prevent over emphasizing high bit rates [50]. A third-order polynomial is fitted to pass through the four measured distortion points [50]. The BD-BR is computed as a difference of the areas that are given by an integral over the distance that both curves cover [50]. A negative BD-BR means that the compared encoder manages to produce similar quality with smaller number of bits. Conversely, positive bit rate means that more bits are required for similar quality. While BD-BR was originally meant to be used with PSNR, it also works with other metrics as long as they are transformed to similar scale as PSNR.

4.2 Subjective Quality Evaluation

The most reliable way to compare encoder qualities is to perform subjective quality tests. ITU-T has released multiple recommendations on how the subjective tests should be conducted, most recently the ITU-T Recommendation P.910 [52]. Although it is ten years old, it is still mostly relevant. However, some parts such as monitors have changed a lot since then, e.g., CRT monitors have been completely replaced by LCD monitors and monitor resolution and sizes have increased. The recommendation has a rigid set of rules

for the viewing conditions [52] to simplify setting up a new scenario or reproducing a prior scenario. However, the disadvantage is that the viewers might feel uncomfortable in the situation, and it may affect the results, or the conditions do not match the intended use of the system. The most realistic conditions would be obtained by arranging tests at viewers' homes. However, this is not a practical solution. When conducting subjective tests, compromises must be made between controlled laboratory environment and viewers comfort.

The recommendation lists a couple of testing methods including *Absolute Category Rating* (ACR) and *Pair Comparison* (PC) that are the most used [6], [52]. In ACR, the viewer is shown the video sequences one at time and after each sequence the viewer is asked to rate the viewed video, usually on one to five scale [52]. Conversely, in PC the viewer is shown the same video sequence twice in a row and then asked to select the better one [52]. The results of ACR have to be normalized in terms of viewer, sequence, and type of distortion (if many) in order to generate the *Mean Opinion Score* (MOS). Because the results must be normalized to three different factors, there is a possibility they get twisted from the original meaning. However, with eliminating outliers after normalization and using a large enough sample size mostly neutralizes the risk. PC only ranks the perceived qualities but not express the quantity of the difference. As the recommendation states, they are only suggestion on how the test could be conducted and can be adapted to different test purposes [52]. Due to these reasons, the PC method was used in this work because it was only necessary to find out whether the proposed method was better or not.

As is the case with BD-BR, either bit rate or quality has to be set constant for subjective test because interpreting the results would be ambiguous otherwise. Typically, bit rate is set constant since trying to produce video that would be perceived at the same quality is difficult [6]. Technically, MOS could be used as a distortion metric for BD-BR but that would bring a fourth variable to the subjective test. In this work, the bit rate is set constant. First, the videos are encoded with the proposed system that uses constant QP and then the measured bit rate is used to encode the same video again using native rate control to produce video with the same size.

Final aspect that should be considered with subjective tests is the selection of test subjects [52]. Depending on the application, it might be extremely important or practically a side note. The most important factor in most cases is that the viewers have normal or corrected-to-normal vision, unless the intention is to test how vision impairments affect the perceived quality. The rule of thumb is that the viewers should match the intended audience of the application. Thirteen people from our Ultra Video Group were used for the subjective test with one being female and twelve males. The age of the participants ranged from 23 to 33.

The test was conducted simultaneously for all the participants seated comfortably from three to five meters away from the screen. The used screen was a 55-inch Panasonic UHD television. Many of the participants were experts of video encoding but they were instructed to view the videos normally and to evaluate the quality of the whole video, not to look extensively at any coding artifacts. The quality voting was done at the end of each video pair. One drawback with this method is that unsure participants are more likely to vote for the second video. However, the order of videos is random so both methods should gain additional votes this way.

4.3 Complexity

Although complexity is not the main focus of this work it should be evaluated to make sure it is acceptable. The complexity of an encoder is measured by running the encoder multiple times and taking an arithmetic mean of the running times. Because the encoder binary and the sequence are cached by the operating system, the first run tends to be slower and is discarded. To get as fair results as possible the amount of other processes running on the same machine is minimized and only one encoder instance is run at a time.

The QP also affects encoding speed. With low QP values, the residual is quantized to more coefficients than with larger values of QP. The entropy coding of coefficients is relatively slow making encoding time higher with low QPs. In this work, the measurements were done using the same QP values as in the coding efficiency evaluation. Each test sequence was encoded five times with all four QP values and an average of the runs was taken for each QP value. For each sequence, the speedup was computed by averaging the speedups at each QP. A ratio between the average encoding time of the anchor and that of the tested encoder was reported. The details of the computer used for the tests are listed in Table 4.1.

Table 4.1. *Properties of the test computer.*

Processor	Intel i7-5960x
Base clock frequency	3.00 GHz
Boost clock frequency	3.50 GHz
Number of cores	8
Number of threads	16
Processor cache	20 MB
Motherboard	Asus x99-A
Memory	16 GB
Operating system	Windows 10 64-bit

4.4 Test Material

The HEVC common test conditions [51] define a set of 24 test sequences with different characteristics. In this work, 22 of these videos were used. They are tabulated in Table 4.2. The sequences are divided into six different classes enumerated from A to F. Classes A, B, C, and D consist of sequences with varied content and have resolutions of 2560×1600, 1920×1080, 832×480 and 416×240, respectively. Class E features video

Table 4.2. *Details of the HEVC test sequences.*

Class	Sequence	Resolution	Frame rate (Hz)	Length (s)
hevc-A	PeopleOnStreet	2560×1600	30	5
	Traffic	2560×1600	30	5
hevc-B	BasketballDrive	1920×1080	50	10
	BQTerrace	1920×1080	60	10
	Cactus	1920×1080	50	10
	Kimono	1920×1080	24	10
	ParkScene	1920×1080	24	10
hevc-C	BasketballDrill	832×480	50	10
	BQMall	832×480	60	10
	PartyScene	832×480	50	10
	RaceHorses	832×480	30	10
hevc-D	BasketballPass	416×240	50	10
	BlowingBubbles	416×240	50	10
	BQSquare	416×240	60	10
	RaceHorses	416×240	30	10
hevc-E	FourPeople	1280×720	60	10
	Johnny	1280×720	60	10
	KristenAndSara	1280×720	60	10
hevc-F	BasketballDrillText	832×480	50	10
	ChinaSpeed	1024×768	30	16.7
	SlideEditing	1280×720	30	10
	SlideShow	1280×720	20	25

conferencing content with 1280×720 resolution. Finally, class F consist of screen content such as computer-generated graphics with various resolutions.

In addition, sequences from several other sources were used to collect eye tracking data: seven videos from Ultra Video Group [53], one from AWS Elemental [54], and twelve videos from Xiph.org [55]. These videos are tabulated in Table 4.3. The sequences from [53] are of various content and have UHD resolution. The sequences were originally 120 Hz, but they were down sampled to 60 Hz because no 120 Hz 4K resolution monitor was available at the time. The down sampling was done by removing every other frame. The single sequence from AWS Elemental is an UHD remake of the famous Foreman sequence. The Xiph.org sequences represent various content such as a distinct object of

interest on the foreground or no clear areas of interest and a lot of background movement. The sequences are either of 1920×1080 or 1280×720 resolution.

Table 4.3. *Additional eye tracking test sequences.*

Source	Sequence	Resolution	Frame rate (Hz)	Length (s)
Ultra Video Group	Beauty	3840×2160	60	5
	Bosphorus	3840×2160	60	5
	HoneyBee	3840×2160	60	5
	Jockey	3840×2160	60	5
	ReadySteadyGo	3840×2160	60	5
	ShakeNDry	3840×2160	60	2.5
	YachtRide	3840×2160	60	5
AWS Elemental	Foreman 4k	3840×2160	24	10
Xiph.org	CrowdRun	1920×1080	50	10
	OldTownCross	1920×1080	50	10
	Parkrun	1280×720	50	10
	PedestrianArea	1920×1080	25	15
	RushHour	1920×1080	25	20
	Shields	1280×720	50	10
	SpeedBag	1920×1080	30	19
	Station2	1920×1080	25	12.5
	Stockholm	1280×720	60	10
	Vidyo1	1280×720	60	10
	Vidyo3	1280×720	60	10
	Vidyo4	1280×720	60	10

For the subjective tests, six videos were chosen: BasketballDrive, Johnny, Kimono, OldTownCross, PeopleOnStreet, and Shields. The selection includes many kinds of content. BasketballDrive has a moving camera and a lot of movement, Johnny has a static camera with little movement, Kimono has a scene cut, OldTownCross has a moving camera but not much other movement, PeopleOnStreet has a static camera but a lot of movement otherwise, and Shields has a zoom.

4.5 Eye Tracking Data Collection

In this work, the gaze data was collected with eye tracking glasses rather than a screen based sensor that is used by most of the previous works. In the screen based solutions, the gaze recorded by the device is automatically tied to the relative screen location whereas the gaze is relative to the screen cameras view of the eye tracking glasses. The Pupil Labs' software has a feature that allows using specific tags to map out an area of the scene cameras view. In Figure 4.1, a screen with the tags is depicted, the area highlighted in blue is where the screen area is mapped. It should be noted that the lightning conditions were brighter than that of Figure 4.1 and they are better illustrated in Figure 4.2.

In this work, the experiments were performed with Pupil Labs Eye Tracking glasses, a 27-inch Lenovo ThinkVision X1 UHD display, and the same computer as the complexity tests. The videos were displayed using *Media Player Classic – Home Cinema* (MPC-HC). The sequences below UHD resolution were displayed uncompressed. The UHD sequences were compressed since the test computer did not have a fast enough SSD to

Figure 4.1. *The screen with the tags used for tracking.*

play them back at 60 frames per second. The sequences were compressed as little as possible to a degree that would not affect people gaze points.

The subjects were obtained by advertising the experiment in the student and staff intranets of *Tampere University of Technology* (TUT), thus most of the subjects were students or staff of TUT. The subjects were offered a chocolate bar as a reward for participating in the experiment. Because the eye tracking glasses do not allow using regular glasses at the same time, the subject were required to have a normal vision at one-meter distance or use contact lenses for correction. A total of thirty-seven subjects were gathered with seventeen being female and twenty male. The minimum, maximum, mean, and median age of the participants were 13, 43, 27.4, and 26, respectively. Majority of the subjects were of Finnish background but a couple of them were from Middle East, East Asia, Southern Europe, or Eastern Europe. The subjects were not specifically screened for normal vision and were trusted considering the reward was no significant enough to warrant lying just to obtain the reward. Most of the subjects had normal vision, couple of them were near-sighted and completed the experiment without glasses, and two used contact lenses.

At the beginning of the experiment, the subjects were given a written instruction on how the experiment will proceed. The instructions can be found in Appendix A. After it was confirmed that the subject had understood the instructions (s)he was seated about one meter from the screen. The subjects were allowed to move within the seat. The monitor was adjusted so that the subject was looking straight at the center of the screen. The subject put on the eye tracking glasses and the experiment personnel adjusted the eye

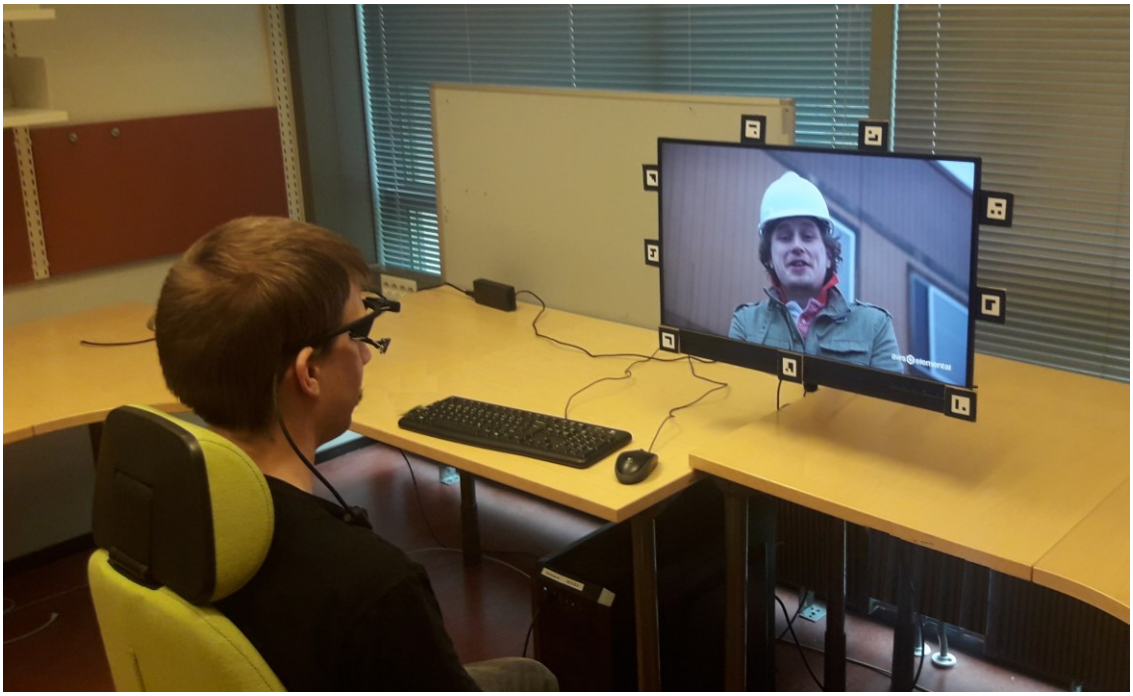

Figure 4.2. *The eye tracking experiment test environment.*

cameras so that the subject's eyes were completely within the frame of the eye camera. Figure.2. depicts a snapshot of the test environment where a subject is sitting in the chair with the eye tracking glasses on. Next, the eye tracking glasses were calibrated, and the successfulness of the calibration was confirmed by the personnel. In most cases, the calibration was successful with a single attempt, but in about quarter of the experiments a second calibration was required and once a third attempt was necessary. After the calibration routine was finished it was made sure that the subject had no questions and was left alone to minimize any outside interference for the remainder of the experiment.

The experiment consisted of 41 videos listed in the previous section. The total duration of the actual video viewing was about twelve minutes. The order of the videos was randomized for each subject to remove any statistical errors that could come from a static order. For example, an interesting object in a corner at the end of a video would make it more likely that at the beginning of the next the subject would still be watching the same position. After every five videos, the video playback was paused and a sequence of five calibration symbols was shown. The subject was informed that the experiment will be

over once the MPC-HC player is closed or that they can also wait for the personnel after the experiment ended.

After the experiment, each subject was questioned verbally if they had any problems, findings, or other comments concerning the experiment. Most subjects had nothing to say. However, some found the experiment fairly exhausting and that they could not relax because the videos were so short. However, the risk was already known beforehand and considered a normal reaction. It was one of the reasons why the experiment was made as short as possible. For example, the Xiph.org collection included more videos than were used. A notable observation from the questioning was that none of the subjects found the eye tracking glasses uncomfortable so wearing the glasses should not affect the viewing experience.

The Pupil Capture software is written in Python which has some limitations in multithreading. Therefore, the capture software uses *ZeroMQ* (ZMQ) [56] sockets for communicating between the different parts of the program. ZMQ is specifically meant for distributed computing and parallelism [56]. The socket interface allows communicating with the program, e.g., external programs can start the recording session. In addition, a custom Python script was used to control the whole experiment by communicating with the other components. A separate script was used because embedding all necessary features into MPC-HC would have been too cumbersome. MPC-HC has a HTTP interface that allows controlling it, which enabled using the external control script. MPC-HC was also slightly modified to support the ZMQ interface of the capture software by initializing a ZMQ context and sockets at the startup of MPC-HC. Additionally, the results of the calibration check were made accessible through the socket interface.

A diagram of the control flow between the components is depicted in Figure 4.3. At the beginning of the experiment, the control script was used to launch the calibration. This was not compulsory but it allowed an easy way to log and record the calibration attempts for further study. For example, to find out why multiple calibrations were necessary. The control script required confirmation of the calibrations success from the personnel. If the calibration was accepted the control software signaled MPC-HC to start the recording. MPC-HC signaled the name of the video a bit before the video was played. MPC-HC stopped the recording after the video was finished. A three second gap also made separating successive recordings easier. After five recordings, the capture software signaled the control script to pause the video playback in the middle of a gray screen and start the calibration check. Once the calibration check was finished the control script again signaled to start the video playback. This was repeated until the final video sequence was played. Once the final recording finished the capture software signaled it to the control script and MPC-HC closed and the experiment was over.

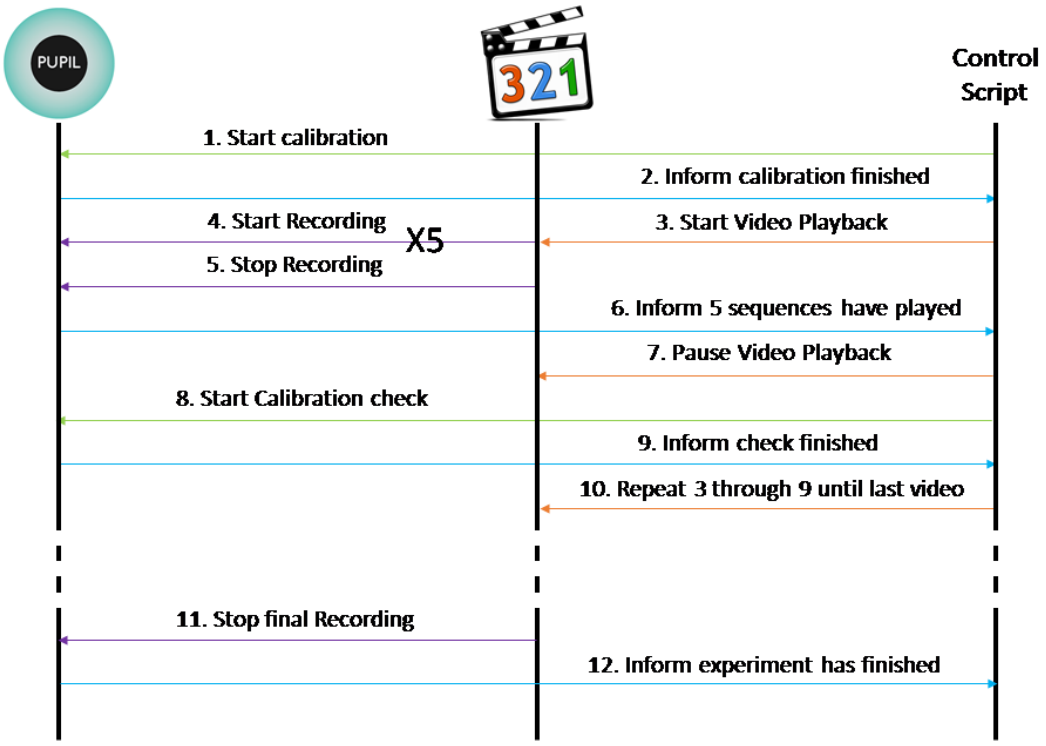

Figure 4.3. *Dataflow between the different software components in the eye tracking experiment.*

5. PROPOSED SYSTEM

As shown in Chapter 3, there are multitude of different ways to make use of eye tracking data in video codecs. Whereas collecting eye tracking data is usually done with screen-based solutions, this work shows that it can also be collected with eye tracking glasses but some additional processing is required. The process is presented in Section 5.1.

Section 5.2 presents a system that uses eye tracking data to enhance the perceived quality of the live encoded video. The system is explained in detail with a proof-of-concept demonstration scenario. Several different systems using eye tracking data for encoding have been introduced but to the best of the author's knowledge this is the first solution using live eye tracking data.

5.1 Eye Tracking Data Processing

During the recording the capture software saves all of the messages transported through the ZMQ interface. The Pupil player software allowed exporting the gaze data to a csv format afterwards, however, only a single recording at a time. The player had to be modified so that it would automatically open a new recording and exports its data once the previous recording was exported.

The purpose of the mid-experiment calibrations were to ensure that the device was still properly calibrated or afterwards correct the results if there was a problem during the experiment. Originally, it was considered that the eye tracker would be recalibrated by the user if the check result was too poor. However, it was found that some people would require multiple attempts to get the device calibrated and as they got more and more exhausted the calibration became even more difficult, so the calibration was verified manually by the experiment personnel. The calibration process is also exhausting since the contrast between the calibration symbols and the background is extreme, so performing the calibration multiple times would definitely affect the results as the subject gets increasingly exhausted during the experiment. The calibration checks were used to form a timeline of how the projected screen surface had moved in between the calibration points. Then, a correction function was derived for each recording on the timeline and the associated gaze points were corrected. The correction process is covered in more detail in [57][1].

For visualization reasons, heat maps were also generated for the sequences. Additionally, a heat map of all gaze points is created for a single subject in each sequence. During the extraction, the player software creates the heat map based on the resolution of the screen area. The maps were generated by applying Gaussian filter with radius equivalent to three degrees of sight radius on a 27-inch screen from one meter away. However, setting the

resolution to 1080p or higher causes instability in the software so the heat maps for them were created afterwards.

5.2 Gaze Controlled Real-time ROI Encoding

The main idea behind the proposal is to use the gaze data from the eye tracking glasses to encode the video with higher perceived quality. The video is obtained from the scene camera of the eye tracking glasses.

Kvazaar supports DQP matrixes, but by default the support is only for a single matrix over the whole sequence. As an input, Kvazaar accepts arbitrary sized matrixes but they are sampled to CTU level using nearest neighbor method. Thus, it does not make sense to use matrixes of finer granularity. Originally, the DQP matrix was stored in the configuration structure of Kvazaar. In order to allow a different matrix for each input frame the DQP matrix is moved to the picture data structure.

Like in the eye tracking experiment, the ZMQ interface of the Pupil capture software was used for communication. Kvazaar was also modified to support ZMQ sockets. The messages sent by the capture software are packed by MessagePack [58] that uses a format similar to *JavaScript Object Notation* (JSON). However, it packs the data so that it requires less space than JSON making the transfers faster without any significant complexity overhead. MessagePack suffers from lackluster documentation, but it is simple to use. The ZMQ sockets work on a *publish-subscribe* (PUB-SUB) principle [56]. The PUB socket is on the capture software whereas the socket added into Kvazaar acts as a SUB socket.

The capture software sends many messages but only the gaze data and the raw video frame messages are of interest here. By default, the capture software does not publish the frames from either the scene camera or eye cameras but has a plugin that allows the publishing. All messages have a timestamp. For the gaze messages the timestamp is relative to the frame of the eye camera from which the gaze data was detected from. The interesting information in the gaze message are gaze location normalized to the view of the scene camera and a confidence value. The confidence value ranges from 0.0 to 1.0 and it is based on how well the capture program detected the pupil from the frame of the eye camera. In addition to the frame data, the frame message contains the width, height, and the data format of the frame.

The scene camera of the eye tracking glasses is capable of multiple different resolutions and frame rates, most notably 1080p30 and 720p60. For this work, the 1080p30 option is the main consideration because the higher resolution video contains more CTUs thus allowing more granularity for the degradation of quality around the gaze center. The feed from scene cameras is compressed using *Motion JPEG* (MJPEG) on the camera hardware due to USB transfer limits. The frame publisher plugin allows multiple different formats:

JPEG, RGB, and YUV of which Kvazaar can take YUV as input. The plugin (version 1.7) does not identify the chroma subsampling of the YUV format but it was found out to be YUV422 by testing. Since Kvazaar at the time of writing only supported YUV420, the YUV422 has to be down sampled before encoding. The down sampling is done using linear filtering, i.e., an average of every two vertical samples is taken.

The eye tracking cameras operated at a higher frequency than the world camera, so multiple gaze points were detected per scene camera frame. The gaze data is stored by Kvazaar, unless the confidence value is too low, to wait for the video frame where the gaze data will be mapped. Time stamps could be used to map each gaze point to the video frame that is temporally closest to it but using next frame instead lowers the overall latency of the system and is simpler to implement. The timeline of different messages and mapping gaze points to frames are depicted in Figure 5.1. An average is calculated from all gaze points in a single frame. If no valid gaze points are found for a specific frame the average of previous frames is used. In addition, the gaze center is written to a custom SEI message. The SEI message can be used, e.g., as a part of video editing pipeline. A filter can read the gaze center from the SEI message and use it, e.g., to highlight the gaze area automatically.

Figure 5.1. *Timeline of frames and gaze points arriving with coloring on which frame each gaze point will be mapped.*

Since the purpose of the system is to provide better quality around the gaze points of the user, a slope of the quality degradation has to be solved. It is more likely that logarithmic change of quality is better for human perception so it is used over linear. For each CTU, the logarithmic distance in CTUs from the center of the CTU to the gaze center is calculated. The ROI is chosen as a single point defined by the gaze center, instead of a larger area, because the DQP matrix is formed in a way that CTUs around the gaze center tend to get zero DQP values. The distance is multiplied by *Degradation Coefficient* (DC) and the multiplied values form the DQP matrix. The larger the DC value the faster the QP drops around the gaze center.

Figures 5.2 and 5.3 depict the effect of different DC values on the DQP matrix for two different resolutions. The green areas have the lowest DQP values starting from zero whereas the areas with the darkest shade of red have DQP values in the low twenties.

Figure 5.4 depicts how the DC affects PSNR, SSIM, and EWPSNR BD-BRs with four different types of sequences. Eye tracking data of two persons (one female and one male) were selected randomly among the people who participated in the eye tracking experiment.

PeopleOnStreet has a static camera with a lot of moving people. BasketballDrive is a short clip from basketball game where the camera follows player passing for another player who then scores. Johnny has a frontal shot of a man in a suit talking and FourPeople has four people holding a panel and passing some fliers to each other. Expectedly, the PSNR-BD-BR and SSIM-BD-BR curves show consistently decreasing quality as the DC increased and there were no significant differences between the two different viewers. The EWPSNR-BD-BR values were higher for the female watcher than male. Considering that EWPSNR is calculated based on the combined eye tracking results most likely the female watcher looked more at areas that most other people did not.

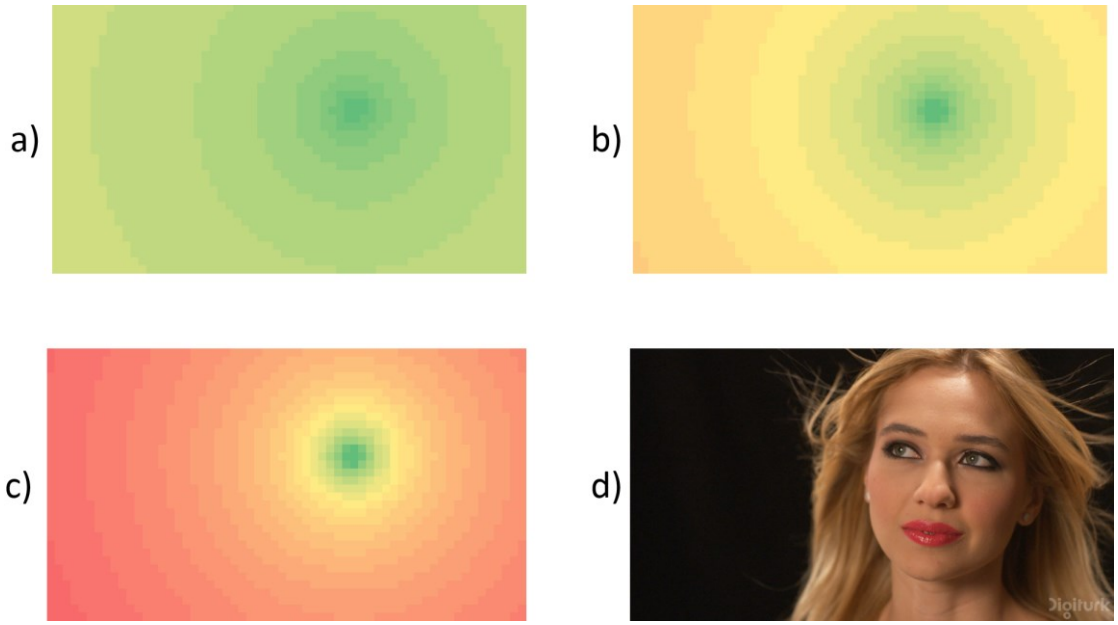

Figure 5.2. *DQP matrixes visualized as heat maps. a) DC = 2, b) 3.5, c) 6, and d) the original frame of the 4K sequence Beauty.*

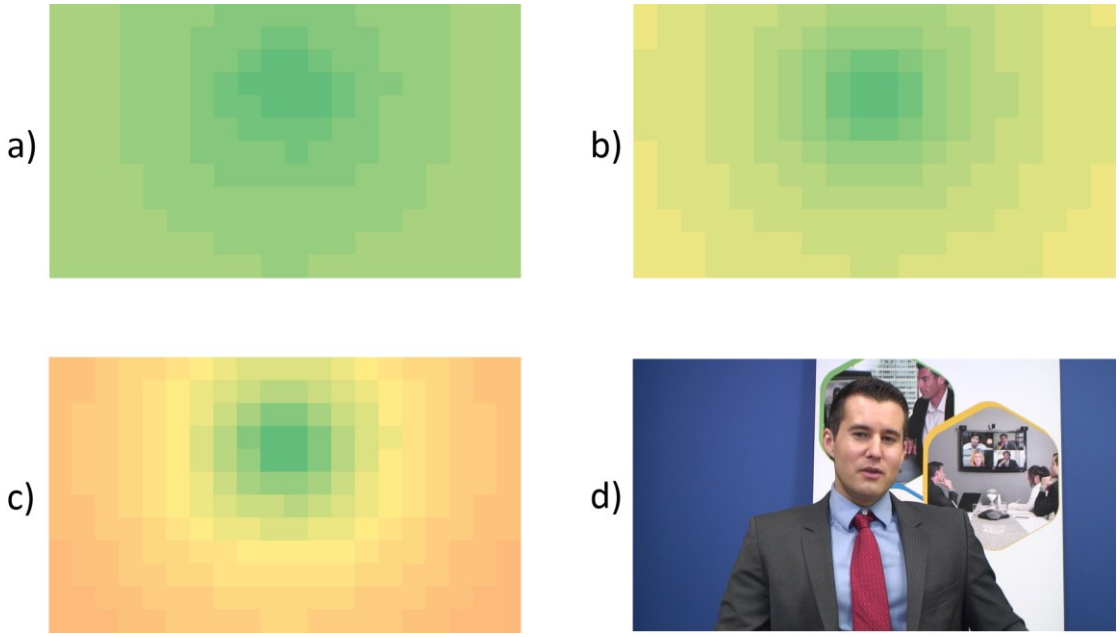

Figure 5.3. *Three DQP matrixes visualized as heat maps for a) DC = 2, b) 3.5, c) 6, and d) the original frame of the 720p sequence Johnny.*

Overall, the EWPSNR values were negative at least for some DC values in all sequences other than FourPeople. Considering that the eye tracking data was collected without including audio in the sequences, the gaze for each person is expected to wander a lot differently in the sequence because the person speaking will not draw as much attention. The EWPSNR values were better than expected for the PeopleOnStreet sequence since there are no objects that would grab attention immediately. Most likely people just looked at the center of the video since there is no objects of interest. The results for BasketballDrive are expected, since most people looked at the ball. Neither of the chosen persons looked purely at the face of the person in the Johnny sequence, which explains the fairly poor results especially for the female watcher. If a user who looked purely at the face of the person was chosen, the results would be significantly better. At least 75 percent of the people looked at the face at all times.

Based on these results, the optimal value for the DC is between 2.0 and 3.5 in a general case. However, if it is known that the gaze points of a test person are similar to a majority of the people watching the encoded video, DC values as high as 6.0 or even 7.0 could be used.

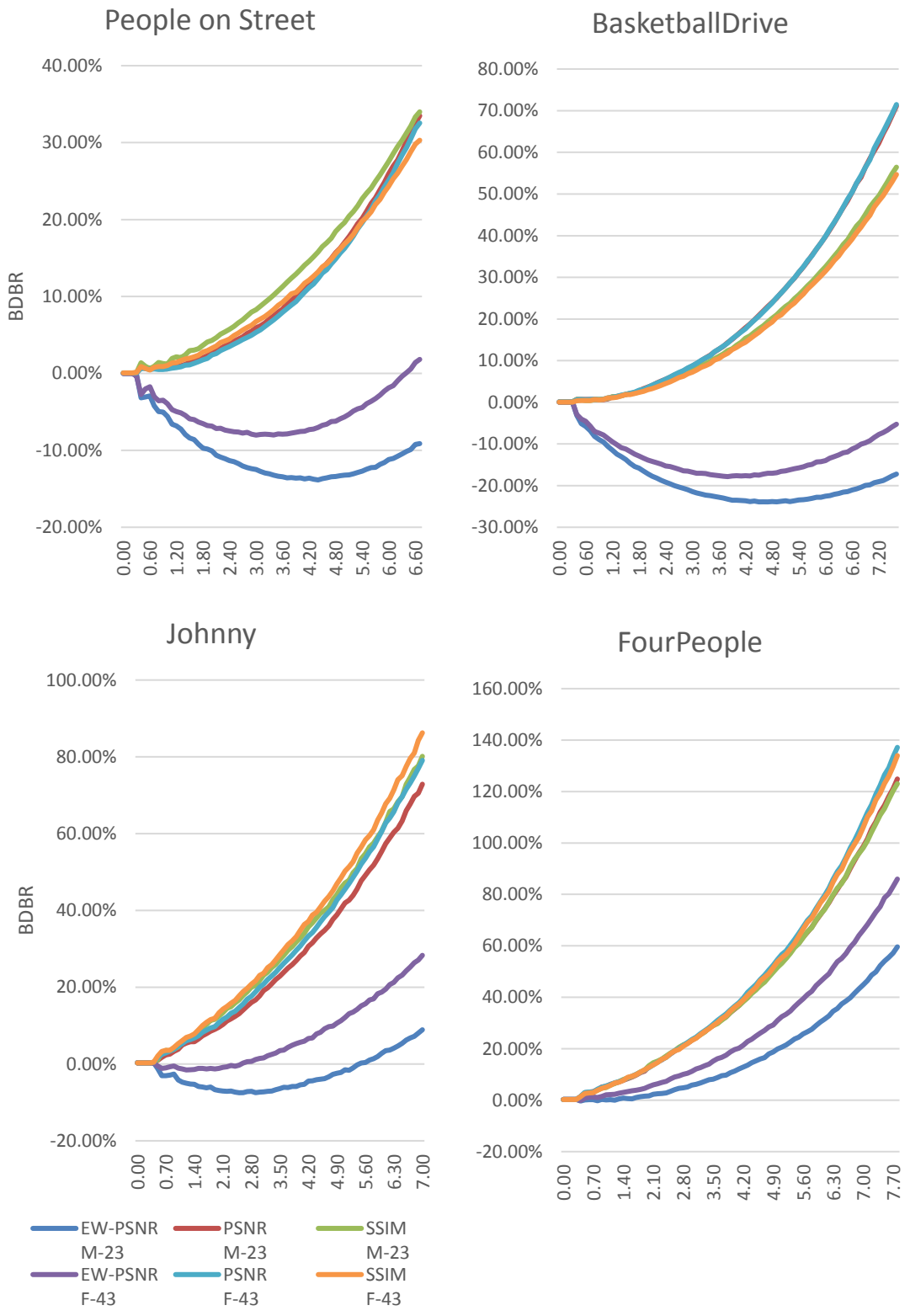

Figure 5.4. *The effect of the Degradation Coefficient compared with regular encoding.*

The whole system can be demonstrated with the setup depicted in Figure 5.5. In addition to Kvazaar and the capture software, the demonstrator includes FFmpeg [59] which is a

popular open-source multimedia framework that can be used for practically any video editing related tasks.

Two computers are needed in the demonstration. The first computer is used for the eye tracking and video encoding. The eye tracking glasses are connected to this computer. The video encoded by Kvazaar is piped to an FFmpeg instance that encapsulates the HEVC bit stream into *MPEG Transport Stream* (MPEG-TS) and streams it to the second computer through Ethernet. MPEG-TS is a standardized transmission container for audio and video. The purpose of the second computer is to play the stream. The stream is decoded and played back by another instance of FFmpeg that also parses the SEI message containing the gaze center. FFmpeg uses a custom filter to draw a red dot over the gaze center for visualizing the user's gaze point.

The first computer has to be more powerful since it has to be able to run the capture software and encode the video; a desktop with at least eight-core processor is necessary. Conversely, the second computer can be a laptop since it only has to receive and decode the stream and display the decoded video.

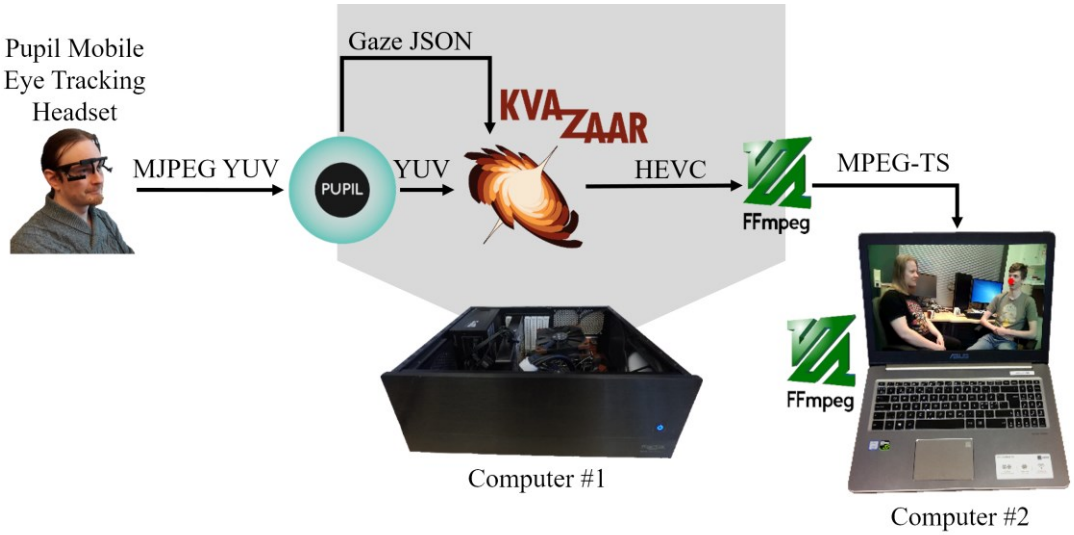

Figure 5.5. *The demonstration setup and the data formats between the components.*

At the beginning of the demonstration session, one person puts the eye tracking glasses on. First, the glasses are calibrated using the capture software as in the data collection experiment. For the demonstration, it is beneficial to set the DC to such a high value that the effect is visible even though the perceived video quality might not be optimal, or alternate between the optimal DC value and a high one, such as 2.0 and 7.0. After the calibration is successful Kvazaar and the FFmpeg instance on the encoding computer can be started. Finally, FFmpeg on the receiving laptop is started and the rest of the viewers can corroborate the quality change around the gaze. In the demonstration session, it is

better if the first person wearing the glasses is one of the personnel presenting the demonstration, so that all spectators can see how the systems works.

6. PERFORMANCE EVALUATION

This chapter contains coding efficiency and complexity evaluations of the proposed systems. This proposal is primarily intended for live usage so the evaluations are based on the Kvazaar ultrafast preset and low-latency setup. All the experiments were done using two test persons, same as used for DC exploration in Section 5.2, as the "live" watchers to know how much the results change between persons. They male person is labeled M-23 and female F-43 for the tables.

Tables 6.1 and 6.2 tabulate the overhead in coding efficiency for HEVC common test sequences when SSIM and PSNR are used as the distortion metric, respectively. Traditional metric show that the quality degraded because they resemble the HVS poorly, e.g., video may have low quality at the edges but high at the center where people look, thus having high perceived quality but low quality when measured with PSNR or SSIM. The performance is evaluated using three different values of DC: 2.0, 3.5, and 6.0. The degradation is higher with large resolutions since by default the DQP values are larger at their edges than with smaller resolutions. The results between the two persons are practically the same excluding the class A with high DC and the class C for PSNR when DC = 6.0. Most likely, the person with higher bit rate looked more at the edges of the video causing the average value of the DQP to be lower.

Table 6.1. *SSIM BD-BR for different DC values compared with regular Kvazaar.*

DC	**2.0**		**3.5**		**6.0**	
Person	**M-23**	**F-43**	**M-23**	**F-43**	**M-23**	**F-43**
hevc-A	5,53 %	6,27 %	15,30 %	20,25 %	46,53 %	40,56 %
hevc-B	5,69 %	5,73 %	16,63 %	16,72 %	50,06 %	50,21 %
hevc-C	2,42 %	2,85 %	7,24 %	7,54 %	22,05 %	21,64 %
hevc-D	1,45 %	1,50 %	4,43 %	4,72 %	12,96 %	13,03 %
hevc-E	11,53 %	11,09 %	25,76 %	25,05 %	63,85 %	59,87 %
hevc-F	2,43 %	2,54 %	7,94 %	8,06 %	24,01 %	21,49 %
Total	4,61 %	4,74 %	12,45 %	12,97 %	35,57 %	34,04 %

Table 6.2. *PSNR BD-BR for different DC values compared with regular Kvazaar.*

DC	2.0		3.5		6.0	
Person	**M-23**	**F-43**	**M-23**	**F-43**	**M-23**	**F-43**
hevc-A	5,52 %	5,15 %	16,02 %	17,89 %	50,22 %	40,93 %
hevc-B	5,69 %	5,81 %	17,22 %	17,34 %	52,98 %	53,40 %
hevc-C	2,88 %	1,89 %	8,50 %	6,41 %	25,12 %	20,07 %
hevc-D	1,15 %	0,94 %	3,72 %	3,03 %	12,96 %	10,92 %
hevc-E	10,03 %	9,45 %	23,93 %	22,89 %	61,16 %	57,10 %
hevc-F	3,33 %	3,42 %	10,37 %	10,04 %	29,43 %	27,37 %
Total	4,50 %	4,21 %	12,74 %	12,23 %	37,22 %	34,26 %

Table 6.3 lists the BD-BR values for each sequence when using EWPSNR as the distortion metric. For DC = 2.0, the quality improved for a majority of the sequences. The major exception was FourPeople with which the quality degraded for both persons. The case was the same with Cactus and BasketballDrillText for the male person. In general, the sequences with no clear objects of interest have mostly reduced quality whereas sequences with a clear objects have improved quality. Also, the sequences with either multiple or somewhat clear objects of interest tend to improve quality with the higher DC values.

The male person looked more at the areas that most of the people did not but nonetheless, the differences between the two persons are overall fairly small. Generally, the system seems to provide better quality when the video has a clear object of interest that draws most people’s attention.

Table 6.3. *EWPSNR BD-BR for different DC values compared with regular Kvazaar.*

DC	**2.0**		**3.5**		**6.0**		**Clear**
Person	**M-23**	**F-43**	**M-23**	**F-43**	**M-23**	**F-43**	**ROI**
PeopleOnStreet	-6,0 %	0,0 %	-5,7 %	-13,4 %	4,9 %	0,0 %	No
Traffic	-3,1 %	-9,5 %	3,5 %	-6,8 %	32,7 %	14,8 %	No
BasketballDrive	-14,1 %	-17,2 %	-17,8 %	-22,6 %	-14,5 %	-22,6 %	Yes
BQTerrace	-7,6 %	-11,5 %	-2,7 %	-9,1 %	25,0 %	12,9 %	No
Cactus	0,3 %	-0,9 %	7,2 %	4,8 %	32,1 %	27,7 %	No
Kimono	-7,5 %	-13,2 %	-7,7 %	-16,3 %	1,9 %	-12,0 %	Yes
ParkScene	-14,5 %	-16,1 %	-16,9 %	-19,7 %	-12,3 %	-16,0 %	Yes
BasketballDrill	-4,8 %	-6,8 %	-5,8 %	-9,4 %	-2,6 %	-8,4 %	Yes
BQMall	-8,2 %	-9,9 %	-11,6 %	-15,1 %	-11,3 %	-17,3 %	No
PartyScene	-11,6 %	-13,4 %	-17,9 %	-22,7 %	-22,2 %	-30,1 %	Yes
RaceHorses	-8,0 %	-12,4 %	-11,9 %	-19,6 %	-13,4 %	-25,9 %	Maybe
BasketballPass	-7,3 %	-5,4 %	-12,3 %	-9,8 %	-18,0 %	-14,5 %	Yes
BlowingBubbles	-6,2 %	-9,8 %	-11,8 %	-17,2 %	-15,4 %	-25,4 %	Yes
BQSquare	-6,5 %	-6,9 %	-9,5 %	-10,6 %	-9,2 %	-13,2 %	Maybe
RaceHorses	-4,8 %	-6,3 %	-8,3 %	-11,2 %	-10,3 %	-16,5 %	Maybe
FourPeople	5,3 %	1,3 %	15,2 %	7,9 %	46,1 %	29,0 %	No
Johnny	-1,2 %	-6,9 %	3,5 %	-6,4 %	18,7 %	2,8 %	Yes
KristenAndSara	-0,6 %	-2,7 %	4,2 %	0,2 %	22,9 %	14,2 %	Maybe
BasketballDrillText	1,7 %	-5,7 %	7,0 %	-7,3 %	25,1 %	-3,4 %	Maybe
ChinaSpeed	-12,7 %	-16,7 %	-19,9 %	-25,9 %	-24,8 %	-33,7 %	Yes
SlideEditing	-5,2 %	-5,2 %	-6,4 %	-7,5 %	-4,4 %	-11,6 %	Maybe
SlideShow	-6,7 %	-5,9 %	-9,6 %	-8,5 %	-12,7 %	-10,7 %	Maybe
Total	-5.9 %	-8.2 %	-6.1 %	-11.2 %	1.7 %	-7.3 %	

In Table 6.4, the results of the subjective quality session are listed. The same male whose eye tracking data was used for the objective quality was used as the “live” watcher. Eye tracking data was used from only one person to reduce the length of the test and since the objective results were similar between the two persons it was not necessary to include both of them. The DC and QP values were selected to generate low bit rates based on trial and error. For higher DC values, lower base QP values were used because higher DC saves more bits. Overall, the results seem promising despite that Kimono and PeopleOnStreet should be ignored since the rate control algorithm of Kvazaar does not work optimally [60], and caused noticeable jitter. Hence, Johnny with DC = 6 is the only scenario where the perceived quality was worse, whereas most of the scenarios benefit from the proposed system (similar quality with OldTownCross). The quality with Johnny is not a surprise considering that the person looked around the scene, whereas most people looked exclusively at the character in the video. For the lower DC value, the proposed system clearly produces superior quality, at least compared with the rate control of Kvazaar.

Table 6.4. *Vote counts for preferring the proposed system out of 13 participants.*

	DC 2; QP 27	DC 2; QP 32	DC 6; QP 22	DC 6; QP 27
Johnny	10	13	7	2
BasketballDrive	13	13	11	13
OldTownCross	11	12	5	7
Shields	11	13	9	13
Kimono	13	13	13	13
PeopleOnStreet	13	13	13	13

Table 6.5 lists the speedups in encoding time for each sequence class with DC values of 2.0, 3.5, and 6.0. Again, the same two persons are used as "live" watchers. The encoding time is reduced for every sequence other than SlideEditing. The speedup is larger with higher DC values because the DQP gets larger. For the same reason, larger resolutions tends to have a higher speedup than smaller ones. For class A, the speedup is not as large even though it contains the largest resolutions because the sequences have large areas with no movement and changing the QP does not affect how long encoding those areas takes. The encoding time does not vary much between the persons.

Table 6.5. *Speedup in encoding times for different values of DC.*

DC	**2.0**		**3.5**		**6.0**	
Person	**M-23**	**F-43**	**M-23**	**F-43**	**M-23**	**F-43**
hevc-A	1.06×	1.12×	1.18×	1.19×	1.11×	1.27×
hevc-B	1.14×	1.14×	1.22×	1.23×	1.33×	1.33×
hevc-C	1.10×	1.10×	1.16×	1.16×	1.27×	1.26×
hevc-D	1.04×	1.04×	1.07×	1.08×	1.13×	1.14×
hevc-E	1.04×	1.04×	1.06×	1.06×	1.08×	1.07×
hevc-F	1.05×	1.05×	1.08×	1.09×	1.13×	1.15×
Total	**1.08×**	**1.08×**	**1.13×**	**1.14×**	**1.19×**	**1.21×**

7. FUTURE WORK

Both the eye tracking data and the gaze controlled ROI encoding system have many potential use cases. Especially, the gaze controlled encoding system has a lot of different options considering it is currently a proof-of-concept system.

7.1 Eye Tracking Data

As noted multiple times in this Thesis, little effort is put into studies how steeply the quality of the video can deteriorate around the ROI for maximal perceived quality. The eye tracking data was originally collected solely for this reason, but other uses were also discovered, e.g., using it for weighting PSNR to EWPSNR. Where objective testing methods are simple, subjective tests are more important and difficult to conduct. Additionally, it is important to find out how the objective quality metric relates to the actual perceived quality. If some objective quality metric behaves similarly to the subjective results, it can be used for this type of distortion reliably without the need for performing subjective tests.

The eye tracking data is used to detect ROI in a video. The parameters that could be considered for the experiment are 1) the base QP of the video; 2) radius of the ROI; 3) degradation method; and 4) the rate of the degradation. The parameters have to be limited since the amount of testing data increases exponentially. The base QP can be limited to two different choices: a lower value representing high quality video and a higher one for lower quality video, with the most probable being 22 and 32, respectively. The effect of the radius is not studied much [31] so it can be ignored in the experiment. Additionally, since the ROI is formed from multiple gaze points, the radius can be formed based on how many gaze points are clustered together. For the quality degradation method, linear and logarithmic methods are the most commonly used [6], [32], thus they should be tested. Finally, the rate of degradation is the parameter with the most variability allowed, since it has the largest impact out of the parameters when considering ROI encoding. The video resolution is a factor when considering the rate of the degradation, whether the rate should be equal when considering the distance in pixel domain or the actual width when displayed on the screen. Since in the experiment the videos will be shown stretched to the screen size it makes more sense to normalize the rate of the degradation to the screen size. Four to five different rates of degradation should be enough but actual trials should be done first to determine how large of an affect the different rates have.

The video material used for the experiment should also be considered. Sequences can be roughly categorized into a 2 by 2 matrix based on how much background and foreground movement they have. Videos from all these categories should be considered since it is

likely that the optimal rate of degradation can be different for different categories. For example, a video with a lot of foreground movement and little background movement is likely to have a good perceived quality even if the QP drops sharply outside of the foreground objects. On the other hand, if there is a lot of background movement the degradation is likely more noticeable.

Even with the limitations if at least two videos from each category per different test condition is wanted to be rated by each participant, they would have to watch total of 160 sequences. Whereas watching 160 sequences is feasible during one experiment split into two sessions, with a break in-between, there are not enough unique sequences and using duplicates is not recommended for performing an ACR test [52]. An alternative is to split the test into two different experiments where the first will determine whether logarithmic or linear degradation is better by performing PC test for each test condition. Then the second experiment can be conducted with reduced test conditions with ACR, to determine the effect of different rates of degradation to the perceived quality.

7.2 Gaze Controlled ROI Encoding

The simplest improvement to the system would be to use Kvazaar as a library in the Pupil capture software instead of transferring the data to Kvazaar as an external program. This would be also beneficial for the capture software because the video encoding is currently done by a basic video interface of OpenCV [61] library, which is not meant for video compression. Implementing the change would require a Python interface for Kvazaar.

Changing the scene camera of the eye tracking glasses would improve the overall video quality of the system. The pupil glasses and software are modular and allow changing the parts. However, since the pupil software supports multiple platforms, it uses libusbK USB drivers instead of the default ones. Two cameras were tested with the system: a Sony 4K action camera using external capture card and a Logitech Brio 4K webcam. The libusbK drivers for the capture card did not work out of the box and the webcam failed with no apparent reason for inputs higher than 720p30. However, getting them to work should be possible. If the libusbK turns out to be too difficult to use it is always possible to write a Windows specific capture plugin for the capture software.

Mounting the alternative cameras should also be solved. For the webcam, it should be possible to disassemble the camera mount and substitute it for the default scene camera, since it is also a repurposed webcam. However, the action camera is larger so it requires an alternative mounting option. For example, using a helmet mount could work, although the capture software might require some tweaking considering that the scene camera is further away from the actual position of the eyes. More advanced system could use a camera that is not attached to the person’s head. In that case, some method would be necessary to track the head movement so that the gaze can be accurately mapped to the video. For example, in a car, the eye tracking sensor and a camera that is used to track the

head movement can be embedded into the dashboard. For the video, a dashcam could be used. The system can work as sort of a black box for the car. In such a system, it would probably make more sense to encode the area at the gaze center with worse quality since most likely the interesting area for the accident investigation is outside of the driver's gaze.

The ROI generation of the system can also be improved. The model in [48] could be embedded into the encoder with only the forward propagation part. Also, instead of using the DQP matrixes, a rate control algorithm similar to [48] could be used. One weakness the system currently has is that if the wearer's gaze moves from the original area, the frame that will be used as a reference will have a worse quality at the gaze area, meaning the frames while moving the gaze will have worse quality. This can be alleviated by building a custom coding structure where frames that are more likely to be used as reference have larger high-quality areas. Similarly, the high quality area is larger when the gaze is moving .

8. CONCLUSIONS

This work presented different ways to use eye tracking data to improve video encoding. First, some existing systems using eye tracking to enhance the perceptual quality of video were presented. Common feature for all these systems was that they either used pre-encoded video or were not real-time capable. Second, the collection of eye tracking data from different videos using eye tracking glasses was depicted. Eye tracking data from videos is usually collected using screen based eye trackers but in this work, eye tracking glasses were validated as a working method. Furthermore, eye tracking glasses have the benefit of allowing more flexible viewing conditions, which in turn allows gathering more diverse data.

Finally, a real-time system using eye tracking data to improve the perceived video quality was presented. The system uses eye tracking glasses to obtain the gaze data, which in turn is used to encode the video from the scene camera with higher perceived quality. Traditional quality metrics, such as PSNR and SSIM, do not show quality improvements. However, EWPSNR represents the HVS better and shows up to 33.7% and on average 5-10% bit rate reduction over traditional video encoding. A conducted subjective test also justified that the system produces better quality. Additionally, the system improves the encoding speed by 8-20% depending on how aggressively the quality is degraded. The system has many potential uses, e.g., livestreaming events from a point of view perspective with higher perceived quality.

BIBLIOGRAPHY

[1] Cisco, “Cisco Visual Networking Index: Forecast and Methodology, 2016-2021,” Sept. 2017.

[2] ITU-T and ISO/IEC, “High efficiency video coding,” Recommendation ITU-T H.265 and ISO/IEC 23008-2, Apr. 2013.

[3] ITU-T and ISO/IEC, “Advanced video coding for generic audiovisual services,” Recommendation ITU-T H.264 and ISO/IEC 14496-10, May 2003.

[4] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[5] N. Jayant, J. Johnston, and R. Safranek, “Signal compression based on models of human perception,” *Proc. IEEE*, vol. 81, no. 10, pp. 1385-1422, Oct. 1993.

[6] J. Lee and T. Ebrahimi, “Perceptual video compression: A survey,” *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 6, pp. 684-697, Oct. 2012.

[7] Z. Wang and A. C. Bovik, “Mean squared error: Love it or leave it? A new look at signal fidelity measures,” *IEEE Signal Process. Mag.*, vol. 26, no. 1, pp. 98-117, Jan. 2009.

[8] S. Tourancheau, F. Autrusseau, Z. M. P. Sazzad, and Y. Horita, “Impact of subjective dataset on the performance of image quality metrics,” *Int. Conf. on Image Process.*, San Diego, CA, USA, Oct. 2008.

[9] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” *Int. Conf. on Mach. Learning*, Lille, France, 2015.

[10] Y. Gitman, M. Erofeev, D. Vatolin, B. Andrey and F. Alexey, “Semiautomatic visual-attention modeling and its application to video compression,” *Int. Conf. on Image Process.*, Paris, France, Oct. 2014.

[11] A. Duchowski, *Eye Tracking Methodology: Theory and Practice.* (Second ed.) 2007.

[12] JVET, “Working draft 1 of versatile video coding,” [Online]. Available: https://mpeg.chiariglione.org/standards/mpeg-i/versatilevideo-coding/n17669-working-draft-1-versatile-video-coding

[13] A. Grange, P.De Rivaz, and J.Hunt, *VP9 Bitstream & Decoding Process Specification* [Online]. Available: http://www.webmproject.org/vp9/

[14] *AV1* [Online]. Available: https://aomedia.googlesource.com/aom

[15] B. Bross, W. J. Han, G. J. Sullivan, J R. Ohm, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 9," document JCTVC-K1003, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Oct. 2012.

[16] *Maliv550 GPU announcement and specification* [Online]. Available: https://www.anandtech.com/show/8650/arm-announces-maliv550-video-processor-malidp550-display-processor

[17] *Snapdragon 610 & 615 announcement and specification* [Online]. Available: https://www.anandtech.com/show/7784/snapdragon-610-615-qualcomm-continues-down-its-64bit-warpath-with-48core-cortex-a53-designs

[18] *A crisis, the causes and a solution* [Online]. Available: http://blog.chiariglione.org/a-crisis-the-causes-and-a-solution

[19] *Alliance for Open Media Established to Deliver Next-Generation Open Media Formats* [Online]. Available: https://aomedia.org/alliance-to-deliver-next-generation-open-media-formats/

[20] M. Budagavi, A. Fuldseth, G. Bjøntegaard, V. Sze, M. Budagavi, and G. J. Sullivan, "HEVC transform and quantization," *in High Efficiency Video Coding (HEVC): Algorithms and Architectures*, Cham, Switzerland: Springer, 2014, pp. 141-169.

[21] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[22] *The H.264/AVC JM Reference Software ver. 19.0* [Online] Available: http://iphome.hhi.de/suehring/tml/

[23] *x264* [Online]. Available: http://www.videolan.org/developers/x264.html

[24] *Joint Collaborative Team on Video Coding Reference Software, ver. HM 16.0* [Online]. Available: http://hevc.hhi.fraunhofer.de/

[25] *Kvazaar HEVC encoder* [Online]. Available: https://github.com/ultravideo/kvazaar

[26] *Turing Codec* [Online]. Available: http://turingcodec.org/

[27] *x265* [Online]. Available: http://x265.org/

[28] D. Vatolin, D. Kulikov, and M. Arsaev, "HEVC video codecs comparison 2017," [Online]. Available: http://www.compression.ru/video/codec_comparison/hevc_2017/

[29] M. Viitanen, A. Koivula, A. Lemmetti, A. Ylä-Outinen, J. Vanne, and T. D. Hämäläinen, "Kvazaar: open-source HEVC/H.265 encoder," *in Proc. ACM Int. Conf. Multimedia*, Amsterdam, The Netherlands, Oct. 2016.

[30] D. Grois, T. Nguyen, and D. Marpe, "Performance comparison of AV1, JEM, VP9, and HEVC encoders," *in Proc. Appl. of Digital Image Process. XL,* San Diego, CA, USA. Aug. 2017.

[31] S. Arndt and J. N. Antons, "Enhancing video streaming using real-time gaze tracking," *in Proc. ISCA/DEGA Workshop on Perceptual Quality of Syst.*, Berlin, Germany, Aug. 2016

[32] B. Ciubotaru, G. Ghinea, and G. M. Muntean, "Subjective assessment of region of interest-aware adaptive multimedia streaming quality," *IEEE Trans. Broadcast.*, vol. 60, no. 1, pp. 50-60, Mar. 2014.

[33] T. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J. Ohm, and G. Sullivan, "Video quality evaluation methodology and verification testing of HEVC compression performance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 76-90, Jan 2016.

[34] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Patt. Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254-1259, Nov. 1998.

[35] L. Itti, "Automatic foveation for video compression using a neurobiological model of visual attention," *IEEE Trans. Image Process.*, vol.13, no. 10, pp. 1304-1318, Oct. 2004.

[36] T. Liu, J. Sun, N. N. Zheng, X. Tang and H. Y. Shum, "Learning to detect a salient object," *in Proc. Conf. on Comput. Vision and Pattern Recognition*, Minneapolis, MN, USA, 2007.

[37] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," *in Proc. Conf. on Comput. Vision and Pattern Recognition*, Minneapolis, MN, USA, 2007.

[38] X. Li, L. Zhao, L. Wei, M. H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang, "DeepSaliency: Multi-task deep neural network model for salient object

detection," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3919-3930, Aug. 2016.

[39] S. S. S. Kruthiventi, K. Ayush, and R. V. Babu, "DeepFix: A fully convolutional neural network for predicting human eye fixations," *IEEE Trans. Image Process.,* vol. 26, no. 9, pp. 4446-4456, Sept. 2017.

[40] H. Hadizadeh and I. V. Bajić, "Saliency-aware video compression," *IEEE Trans. Image Process.,* vol. 23, no. 1, pp. 19-33, Jan. 2014.

[41] *Tobii* [Online]. Available: https://www.tobii.com/

[42] *SR-Research* [Online]. Available: https://www.sr-research.com/

[43] *Pupil Labs* [Online]. Available: https://pupil-labs.com/

[44] *Ergoneers* [Online]. Available: https://www.ergoneers.com/en/

[45] *Tobii Pro x3-120* [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-x3-120/

[46] S. Lee and A. C. Bovik, "Fast algorithms for foveated video processing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 2, pp. 149-162, Feb. 2003.

[47] J. Ryoo, K. Yun, D. Samaras, S. R. Das, and G. Zelinsky, "Design and evaluation of a foveated video streaming service for commodity client devices," *in Proc. ACM Multimedia Syst. Conf.*, Klagenfurt, Austria, Jun. 2016.

[48] V. Lyudvichenko, M. Erofeev, Y. Gitman, and D. Vatolin, "A semiautomatic saliency model and its application to video compression," *in Proc. Int. Conf. on Intelligent Comput. Comm. and Process.,* Cluj-Napoca, Romania, Sep, 2017.

[49] Zhicheng Li, Shiyin Qin, and Laurent Itti, "Visual attention guided bit allocation in video compression," *Image and Vision Computing*, vol. 29, no. 1, pp. 1-14, Jan. 2011.

[50] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," VCEG-M33*, 13th VCEG meeting, Texas*, Apr. 2001.

[51] F. Bossen, "Common HM test conditions and software reference configurations," JCTVC-L1100, *12th JCT-VC meeting*, *Geneva*, Jan. 2013.

[52] ITU, "Subjective video quality assessment methods for multimedia applications," *ITU-T Recommendation P.910, Geneva*, Switzerland, Apr. 2008.

[53] *Ultra Video Group* [Online]. Available: http://ultravideo.cs.tut.fi/

[54] *AWS Elemental* [Online]. Available: https://www.elemental.com/resources/4k-test-sequences

[55] *Xiph.org, Derf's Collection* [Online]. Available: https://media.xiph.org/video/derf/

[56] *ZeroMQ* [Online]. Available: http://zeromq.org/

[57] K. Siivonen, J. Sainio, M. Viitanen, J. Vanne, and T. D. Hämäläinen, "Open framework for error-compensated gaze data collection with eye tracking glasses," submitted to ISM2018.

[58] *MessagePack* [Online]. Available: https://msgpack.org/

[59] *FFmpeg* [Online]. Available: https://ffmpeg.org/

[60] *Kvazaar issue tracker* [Online]. Available: https://github.com/ultravideo/kvazaar/issues/211

[61] *OpenCV* [Online]. Available: https://opencv.org/

APPENDIX A. THE INSTRUCTION GIVEN FOR PARTICIPANTS

Instructions

1. Putting the headset on

Once you have the headset on, try closing your eyes. If you feel that your eyebrows touched the headset or that they moved, try moving them a bit further out and try again. After you are comfortable with the glasses, the attendant will set the cameras properly.

2. Calibration

NOTE: Please keep your head as still as possible during the calibration

During calibration, you will be shown symbols as described below. Gaze directly into the middle of the symbol and the red dot will turn green. Please keep your gaze on the dot until the symbol disappears and then move your gaze to the middle of the next symbol. This process will repeat until the calibration is done.

Calibration symbol

3. Actual test

After calibration, the test begins and the assistant will leave the room. The actual test consist of 41 videos that are all about 10 seconds long. After each video, you will be shown gray screen for three seconds. Every five videos you will be shown five symbols similar to the ones used during calibration to ensure that the calibration is valid. Afterwards the test resumes normally. The test ends when the video player shuts down and the computer returns to the desktop. At this point, you may remove the headset. Inform the assistant that the test is finished.

If you have any problems during the test, you can pause the test by pressing spacebar. Call in the attendant to figure out the problem.

NOTE: During this phase, you may move your head freely but try to avoid drastic head movement. Most importantly try not to squeeze your eyes because it can move the glasses. Blinking normally is fine.