Optimized Bandwidth Usage for Real-Time Remote Surveillance System

Evangelos Valtos

Master of Science
School of Informatics
University of Edinburgh
2005
Abstract

Foveation, a non-uniform resolution representation of an image reflecting the sampling in the retina, is used as a technique in low bit-rate video coding. This report discusses the implementation of a surveillance system that applies foveation to the encoded video stream using an eye tracker to find the fixation point. Noise elimination and system accuracy are assured by the use of a Kalman filter. A gaze driven control mechanism for the camera has been implemented to enhance the functionality of the system. A set of experiments demonstrates that foveated encoding is an appealing technique for video transmissions over limited bandwidth; it offers reduced size in the encoded stream without compromising the perceptual video quality.
Acknowledgements

I would like to thank the following people whose contribution to this project was essential: Dr. Sethu Vijayakumar, my supervisor, for his inspiration, guidance, and support throughout this project, Tim Hospedales for his valuable help with the eye tracker and other technical issues, Istvan Siklossy who helped me to get started with the eye tracker setup and calibration, Vasilis Vasaitis for sharing his coding expertise and for his crucial programming advices, Manolis Korakakis for offering his pair of eyes in order to carry out the project experiments.

I would also like to thank Arthur Alexander, Timothy Rost, Hamid Rahim Skeikh for the code fragments regarding the various devices used in the project. Finally thanks to everyone who at times sat on the eye tracker chair to help me test during the implementation of the project.
Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Evangelos Valtos)
# Table of contents

Abstract .......................................................................................................................i

Acknowledgements ................................................................................................ ii

Declaration .............................................................................................................. iii

Table of contents .....................................................................................................iv

1. Introduction ....................................................................................................... 1
   1.1. Introduction ............................................................................................... 1
   1.2. Background .............................................................................................. 1
   1.3. Desirable Outcome ..................................................................................... 3
   1.4. Project Outline ........................................................................................... 3
       1.4.1. System Description ........................................................................... 4
       1.4.2. Methodology ...................................................................................... 4
       1.4.3. Thesis Layout ..................................................................................... 4

2. System Design and Implementation ............................................................ 5
   2.1. System Architecture .................................................................................. 5
       2.1.1. System Setup ...................................................................................... 6
       2.1.2. Development Tools .......................................................................... 7
       2.1.3. Development Environment ............................................................. 8
   2.2. System Modules: Sensory ...................................................................... 11
       2.2.1. Eye Tracking ...................................................................................... 11
       2.2.2. Filtering ............................................................................................ 12
       2.2.3. Video Capture and Display ............................................................... 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.4.</td>
<td>Foveation Model</td>
<td>19</td>
</tr>
<tr>
<td>2.2.5.</td>
<td>Foveated Video Coding</td>
<td>22</td>
</tr>
<tr>
<td>2.3.</td>
<td>System Modules: Motor</td>
<td>23</td>
</tr>
<tr>
<td>2.3.1.</td>
<td>Camera Movement</td>
<td>23</td>
</tr>
<tr>
<td>2.4.</td>
<td>System Modules: User Interface</td>
<td>24</td>
</tr>
<tr>
<td>2.4.1.</td>
<td>Runtime Functionalities</td>
<td>24</td>
</tr>
<tr>
<td>2.4.2.</td>
<td>Command Line Arguments</td>
<td>25</td>
</tr>
<tr>
<td>3.</td>
<td>System evaluation</td>
<td>27</td>
</tr>
<tr>
<td>3.1.</td>
<td>Introduction</td>
<td>27</td>
</tr>
<tr>
<td>3.2.</td>
<td>Calibration Process</td>
<td>27</td>
</tr>
<tr>
<td>3.3.</td>
<td>Experiment 1: Compression Capabilities</td>
<td>29</td>
</tr>
<tr>
<td>3.3.1.</td>
<td>Description and Setup</td>
<td>29</td>
</tr>
<tr>
<td>3.3.2.</td>
<td>Results and Analysis</td>
<td>29</td>
</tr>
<tr>
<td>3.4.</td>
<td>Experiment 2: Reading Task</td>
<td>33</td>
</tr>
<tr>
<td>3.4.1.</td>
<td>Description and Setup</td>
<td>33</td>
</tr>
<tr>
<td>3.4.2.</td>
<td>Results and Analysis</td>
<td>34</td>
</tr>
<tr>
<td>3.5.</td>
<td>Experiment 3: Object Tracking Task</td>
<td>36</td>
</tr>
<tr>
<td>3.5.1.</td>
<td>Description</td>
<td>36</td>
</tr>
<tr>
<td>3.5.2.</td>
<td>Gaze Driven Camera Movement Disabled</td>
<td>36</td>
</tr>
<tr>
<td>3.5.2.1.</td>
<td>Setup</td>
<td>36</td>
</tr>
<tr>
<td>3.5.2.2.</td>
<td>Results and Analysis</td>
<td>37</td>
</tr>
<tr>
<td>3.5.3.</td>
<td>Gaze Driven Camera Movement Enabled</td>
<td>39</td>
</tr>
<tr>
<td>3.5.3.1.</td>
<td>Setup</td>
<td>39</td>
</tr>
<tr>
<td>3.5.3.2.</td>
<td>Results and Analysis</td>
<td>40</td>
</tr>
<tr>
<td>4.</td>
<td>Conclusion</td>
<td>43</td>
</tr>
<tr>
<td>Bibliography</td>
<td>45</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1. Introduction

An interesting method of video compression for real time video streams in order to reduce the size of transmission is priority encoding of selected regions for each visual frame. Such an approach that encodes with priority certain “interesting” regions takes advantage of the neurobiological anatomy of the human eye. Photoreceptors on the retina have a highly nonuniform distribution which has an immediate effect on the area of a visual scene that is captured at high resolution. Only a region of 2-5° of visual angle around the point of gaze is highly processed, while the resolution decays logarithmically with eccentricity. Consequently it would be unnecessary to encode redundant information of a visual scene by using a uniform quality compression algorithm. A human observer would be able to efficiently process and perceive only a small part of an attended scene depending on his current point of fixation.

1.2. Background

Diepen et al proposed a technique [DGR94] that allows graphical masks or windows of arbitrary form, size and content to be moved quickly over a complex graphical stimulus. This moving overlay technique makes use of the ATVista Graphics Adapter, a board with the ability to mix an internally stored and an externally generated image into one composite image. By programming the internal image to be moveable and partly transparent, a
high-performance moving mask or window is created. The technique is implemented on a standard PC interfaced with an eye tracker, thus bringing mask (window) movement under on-line eye-movement control.

Kortum and Geisler have developed a preliminary version of a foveated imaging system [KG96], implemented on a general purpose computer, which greatly reduces the transmission bandwidth of images. Eye movements are recorded so that the high resolution region of the image can be kept aligned with the high resolution region of the human visual system. Their system has demonstrated that significant reductions in bandwidth can be achieved while still maintaining access to high detail at any point in an image.

The work of Lee, Bovik and Kim [LBK99] introduces a prototype for foveated visual communications as one of future human interactive multimedia applications, and demonstrate the benefit of the foveation over fading statistics in the downtown area of Austin, Texas. In order to compare the performance with regular video, they use spatial/temporal resolution and source transmission delay as the evaluation criteria.

A formal model [RS99] which describes consistently levels of detail and regions of interest for the progressive transmission of raster images is introduced by Rauschenbach and Schumann. This model serves as the foundation of a wavelet-based image communication framework which supports the progressive, redundancy-free transmission of multiple, possibly overlapping regions of interest in an image. To support demand-driven transmission, a method for interleaving control commands with the embedded bit stream is introduced.

Reingold and Loschky [RL02] report three experiments that document a slowing of peripheral target acquisition associated with the presence of a gaze-contingent window. This window effect was shown for displays using either moving video or still images. The window effect was similar across a resolution-defined window condition and a luminance-defined window condition, suggesting that peripheral image degradation is not a prerequisite of this effect. The window effect was also unaffected by the type of window boundary used (sharp or blended). These results are interpreted in terms of an attentional bias resulting in a reduced saliency of peripheral targets due to increased competition from items within the window.

The theoretical benefits, implementational issues, and behavioral consequences of variable-resolution displays are reviewed by Pakhurst and Niebur [PN02]. A mathematical analysis of computational efficiency for a
two region variable-resolution display is conducted. The results are discussed in relation to applications that are limited by computational resources, such as virtual reality, and applications that are limited by bandwidth, such as internet image transmission.

1.3. Desirable Outcome

This project attempts to apply existing approaches in interactive gaze-contingent foveation for video transmission to develop a real-time surveillance system. We hope that the designed model will provide a useful scheme for real-time video transmission over limited bandwidth lines, e.g. an internet connection.

The system provides a prototype which includes a sample hardware setup for the proposed surveillance system and the required software to facilitate the data exchange between the two sites. The goal is to develop a biologically motivated algorithm which apart from its apparent scientific value would result in a useful application with possible commercial use. The resulting algorithm could eventually become a successor of the current compression algorithms and it would be more efficient and suitable for use in video transmissions in the future where the need for real-time and interactive applications is growing. Possible future uses could include entertainment and medical applications, or even the facilitation of the communications between the Earth and distant space crafts in space missions, a domain ruled by the limitation in the available transmission bandwidth.

1.4. Project Outline

The purpose of this project is to develop a real-time surveillance system, where a camera could be guided remotely, by the point of gaze of a human observer, while a smart compression algorithm will be applied to the attended video stream to address the available transmission bandwidth. The algorithm that will be used will apply priority encoding of the attended regions of a visual scene, based on the captured point of gaze, to achieve a tradeoff between the overall compressed size of the video stream and the
Chapter 1. Introduction

degradation of the video quality. A sort description of the proposed system architecture follows.

1.4.1. System Description

A human observer can attend a location on a monitor fed by a remote camera. By capturing the human’s point of gaze using an eye tracking device, the remote camera could be manipulated in order to turn towards the attended location of the scene. In addition, a clever compression algorithm is used on the transmitted video stream so that most of the communication bandwidth is allocated to high-fidelity transmission of a small region around the viewer’s current point of regard, while peripheral image regions are highly degraded and transmitted over little remaining bandwidth.

1.4.2. Methodology

The encoding scheme for the real time video stream in the implemented system is based on work of Sheikh et al [SLEB01]. According to their technique the video stream is preprocessed in order to eliminate the undetectable by the human visual system (HVS) frequencies of a visual scene. An open source video compression library is used to apply standard MPEG1 encoding to the preprocessed stream for further compression. Finally the system utilizes Kalman filtering on the data acquired by the eye tracking device to eliminate noise and overcome issues such as imperfect calibration and limited accuracy of the eye tracking system.

1.4.3. Thesis Layout

The rest of the report is organized as follows. In chapter 2 the design and implementation of the system are covered; there is a full description of the architecture, the design tools, and the modules of the system. Chapter 3 presents the system evaluation process and discusses the system performance and the recorded results on the various tests that have been conducted. The final assessment of the system is made in chapter 4, where a conclusion regarding the system usefulness and performance is reached.
Chapter 2

System Design and Implementation

2.1. System Architecture

A steerable camera is used to record in real-time the scene of a distant area under surveillance. The camera is connected to a PC responsible for the transmission of the recorded data and for the manipulation of the camera movement. The footage captured by the camera is encoded using a compression algorithm dependent on the current point of gaze of a human observer. The compressed video stream is transmitted to a distant PC situated in the same physical location as that of the human observer.

A person can view the area under surveillance on a monitor of a PC receiving the video stream from the remote camera. The observer is free to look at any point of the monitor based on his personal intuition of what is interesting in the attended scene. An eye tracking device is used to capture the point of gaze of the human observer and feed the coordinates to the serial port of the PC. These coordinates determine the center of the attended region, which define the area of the transmitted frames that will receive higher encoding. The extracted coordinates are then sent back to the remote PC as input to the compression algorithm, in order to encode the compressed video stream according to the attended location. Thus, a better usage of the available bandwidth will be achieved by transmitting in high quality only the attended and hence the important part of a scene.

In addition to the compression algorithm, the recorded coordinates form the eye tracking device are used to turn the camera towards a specific direction so that the human observer can have a better view of the desired
region of the remote location. So for example when the recorded coordinates indicate that the attended region is near one of the four edges of the scene, the camera will turn so that more details around that point can be viewed.

Consequently the resulting video stream is always dependent on the real-time point of attention of a human observer, in terms of the region of the area the camera will be turned at, and the allocation of the compression quality for each region of the transmitted video frames.

Since the proposed system should be a real-time application the data from the eye tracking system are properly filtered, so that the outcome of the compression algorithm and the movement of the camera will reflect the current point of gaze as good as possible.

2.1.1. System Setup

The actual system setup used to demonstrate the implemented system differs from the one described in the system architecture in the transmission of the encoded stream. Video transmission over the internet is a know issue and several protocols and video compression techniques exist for that purpose. In our system video transmission could be implemented according to the H.323 teleconferencing protocol that defines the protocols to provide audio-visual communication sessions on any packet network.

Nevertheless, considering that the implemented system had to be deployed and operate in a confined lab room, the architecture of the system had to be altered. Since the camera had to be in the same physical place with the eye tracker and the machine running the video encoding algorithm, it is connected to the same machine responsible for the video compression. The video stream is captured, encoded and decoded in order to be displayed to the same machine instead of being transmitted to a distant location.

The above alteration to the system architecture does not affect the quality or characteristics of the resulting video stream but makes the processes more computationally demanding. Even though the computational complexity is increased by having the encoding and decoding taking place on the same machine, the performance of the system is not noticeably affected. Furthermore, if the encoding and decoding phases were separated and two different machines were used, we could use the additional computational power to increase the performance of the system.
Chapter 2. System Design and Implementation

The setup of the system is shown in Figure 1. The compressed video stream is projected on screen where the user of the system can view the attended area while and eye tracker device captures the point of fixation. The position of that point is sent to the PC receiving the captured video scene from the camera. The coordinates are used to instruct the camera to pan/tilt as needed and to apply the foveation routine on the encoded video stream that the user sees.

![Diagram of system setup](image)

**Figure 1: Overview of the setup for the real-time surveillance system.**

2.1.2. Development Tools

The software was implemented in C/C++ using the GNU Emacs editor and the GNU GCC 3.3.3 and GNU Make 3.80 tools for compilation and build control of the source files.

The Open Source Computer Vision Library from Intel (OpenCV) [URL1] was used to display and manipulate the captured video.

The video encoding/decoding was done using the libavcodec library part of the open source FFmpeg project [URL2], a complete solution to record, convert and stream audio and video.
Chapter 2. System Design and Implementation

The Newmat C++ matrix library [URL3] by Robert Davies was used to create and manipulated matrixes needed for the implementation of the Kalman filter.

The video foveation was done using the foveation routine by Hamid Rahim Sheikh [URL4] written in C.

The file vision.cpp contains the main function. The appropriate methods are used to read and filter the data from the eye tracker. The captured video frames are properly processed and converted to apply the foveation routine and the video compression. The methods of OpenCV are used to display the decoded video and a set of run time functionalities are provided which are discussed later on this chapter.

The serial class provides methods to read data from the serial port. It is used to read the coordinates of the fixation point sent to the serial port from the eye tracker.

The class KalmanFilter is used to make an estimation of the fixation position by using a Kalman filter on the eye tracker coordinates.

The camera class is used to read data from the camera as well as to issue pan and tilt commands.

The framegrabber class is used to capture video frames from the camera using the video device.

The class foveate is used to apply the foveation routine to the video frames.

The following classes are written by the corresponding people. The serial and camera classes by Arthur Alexander, the framegrabber class by Timothy Rost, the foveate class by Hamid Rahim Sheikh and the KalmanFilter class is based on the class KalmanCue of Arthur Alexander.

2.1.3. Development Environment

The machine used for the implementation was a Dell Precision 360 workstation with an Intel Pentium 4 3.2Ghz Extreme Edition processor and 1Gb of PC400 DDR RAM running Fedora Core 2 distribution of Linux.

A Sony EVI-D70P high quality color video camera with pan/tilt/zoom operations was used for the video feed.

A Pinnacle PCTV Rave high quality TV Tuner PCI card was used to capture the video frames from the camera for further processing.
An ISCAN RK-464 eye tracking infrared video camera was used to capture the fixation point of the eye of the observer.

A Dell machine running Windows XP was used to run the setup, calibration and operation software for the eye tracker, and send the data to the main machine running the foveated video encoding algorithm.

**The Eye Tracking System**

The ISCAN RK-464 eye tracking system of the SLMC group of the Institute of Perception, Action and Behaviour is used to record and track gaze. The set-up includes a tracking unit mounted on a desk and a workstation computer system running the ISCAN software under Windows XP. The camera is mounted on a motorised precision pan/tilt support and automatically adjusts to small movements of the subject. Additional two television monitors show an image of the eye with the tracked features of the pupil and the corneal reflection marked, and the visual input scene with the tracked gaze overlaid. The Dell workstation provides the video source to the system and a wall-mounted LCD projector. While tracking, the ISCAN workstation streams gaze coordinates over the serial port to the Dell Workstation where the data are processed. Figure 2 summarizes the set-up.
Adjusting the Eye Tracker Setup

Documentation on how to configure the eye tracker for optimal performance and tracking robustness was not readily available since very little information is contained in the manual or on the Internet. Hence the first step was to experiment with the variables involved in the setup of the eye tracker and document an optimal configuration.

In particular, the main reoccurring problem was the loss of the corneal reflection when the subject’s gaze is focused on the upper region of the stimulus. While this problem was temporarily solved by using a regular computer screen, the preferred way of stimuli presentation was through the video-projector. The solution involved lowering the projected image to the maximum, at only 30 centimeters from the ground. In this way, the corneal reflection is not lost when the subject is seated on the eye tracking desk. In order to avoid that the subject’s head gets in the way of the projection, the video projector was mounted further up on the back-wall of the room and placed in a steeper angle. The keystone correction of the video projector compensates this angle to make the projected image a rectangle. The optimal configuration found and used in the experiments can be seen in Figure 3.
A detailed description for each module of the implemented system follows. The design, implementation and role of each module are presented.

**2.2. System Modules: Sensory**

**2.2.1. Eye Tracking**

The foveated video encoding relies on the knowledge of the attended area in a scene by the user of the system. In addition to that the knowledge of the point of attention is used to incorporate the remote camera guidance functionality to the system. When a user observes an area close to the edges of the streamed video frame, the camera is instructed to turn towards that area so that the scene is better attended.

In order to capture the fixation point of the eye an eye tracker is used. The eye tracker uses a specialized infrared video camera system to view the eye and by reading the position of the pupil and the corneal reflection can infer the fixation point. This point can be transmitted to the serial port of a computer as a pair of coordinates.

The coordinates are read constantly during the execution of the main program loop. A separate thread is created, responsible for reading data
from the serial port, to run independently from the video capturing and compression. This is done to avoid timing problems in the execution of the program causing slow video frame rate and loss of packets send to the serial port.

### 2.2.2. Filtering

The raw data returned by the eye tracker are noisy and dependent on the proper calibration and precision of the device. In addition, the human eye even when it fixates at a point performs micro-saccades, which are small but rapid eye movements and result in a constant change of the output returned by the eye tracker. Consequently the use of some kind of filtering of that data, in order to overcome these biological and technical issues, is essential to ensure the proper functionality of the system. A Kalman filter, described next, is used to get an estimation of the fixation point rather than the unreliable raw coordinates.

#### The Discrete Kalman Filter

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. It is a very powerful tool since it can provide estimations of past, present, and even future states even when the precise nature of the modeled system is unknown. A very good introduction and analysis of the Kalman filter is given by Welch and Bishop [WB95].

#### The Process to be Estimated

The Kalman filter tries to estimate the state $x \in \mathbb{R}^n$ of a discrete-time controlled process that is described by the linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

(2.1)

with a measurement $z \in \mathbb{R}^n$ that is

$$z_k = Hx_k + v_k$$

(2.2)
The random variables \( w_i \) and \( v_i \) represent the process and measurement noise respectively. They are assumed to be independent of each other, white, and with probability distributions

\[
p(w) \sim N(0, Q) \quad (2.3)
\]

\[
p(v) \sim N(0, R) \quad (2.4)
\]

The process noise covariance \( Q \) and measurement noise covariance \( R \) matrices are assumed to be constant, however they might change with each time step in practice.

The transition in the state of the system in the difference equation (2.1) from a previous time step \( k-1 \) to a current time step \( k \) is controlled by the \( n \times n \) matrix \( A \), supposing that there is no driving function of process noise. Matrix \( A \) is assumed to be constant but in practice it might change with each time step. The optional control input \( u \in R^l \) is related to the state \( x \) by the \( n \times l \) matrix \( B \). The current system state is related to the measurement \( z_k \) in the measurement equation (2.2) by the \( m \times n \) matrix \( H \). Matrix \( H \) is assumed to be constant but in practice it might change with each time step.

**The Computational Origins of the Filter**

The state estimate at step \( k \) based on the knowledge of the process prior to step \( k \), called a priori state estimate, is defined as \( \hat{x}_k^\sim \in R^n \), and the state estimate at step \( k \) based on the measurement \( z_k \), called a posteriori state estimate, is defined as \( \hat{x}_k \in R^n \).

Then the a priori estimate error covariance is

\[
P_k^\sim = E \left[ (x_k - \hat{x}_k^\sim)(x_k - \hat{x}_k^\sim)^T \right] \quad (2.5)
\]

and the a posteriori estimate error covariance is

\[
P_k = E \left[ (x_k - \hat{x}_k)(x_k - \hat{x}_k)^T \right] \quad (2.6)
\]

In order to derive the equations of the Kalman filter, we need to find an equation that connects the a posteriori state estimate \( \hat{x}_k \) with the a priori estimate \( \hat{x}_k^\sim \) and a weighted difference between the actual measurement \( z_k \) and the measurement prediction \( H\hat{x}_k^\sim \) as shown below.

\[
\hat{x}_k = \hat{x}_k^\sim + K \left( z_k - H\hat{x}_k^\sim \right) \quad (2.7)
\]
Chapter 2. System Design and Implementation

The difference \( \left( z_k - H\hat{x}_k^- \right) \) in equation (2.7) is the measurement innovation, or the residual. The \( n \times m \) matrix \( K \) in equation (2.7) is the gain or blending factor and it minimizes the \textit{a posteriori} error covariance in equation (2.6). One form of the gain that minimizes \( P_k \) is given by

\[
K_k = P_k^- H^T \left( H P_k^- H^T + R \right)^{-1}
\]  

(2.8)

For more details on the derivation of equations (2.7) and (2.8) see Welch and Bishop [WB95].

The Discrete Kalman Filter Algorithm

The Kalman filter applies a form of feedback control to the system in order to acquire the estimation of a process. In that context the filter uses the time update equations to estimate the state at some time and then the measurement update equations to obtain feedback in the form of noisy measurements. Using the time update equations the current state and error covariance estimates are projected forward in time to obtain the \textit{a priori} estimates for the next step. Then by applying the measurement update equations a new measurement is taken into account together with the \textit{a priori} estimate to obtain an improved \textit{a posteriori} estimate.

The equations for the time and measurement updates are presented below in Table 1 and Table 2.

**Table 1: Discrete Kalman filter time update equations**

\[
\hat{x}_k^- = A\hat{x}_{k-1} + B u_{k-1}
\]  

(2.9)

\[
P_k^- = A P_{k-1} A^T + Q
\]  

(2.10)

**Table 2: Discrete Kalman filter measurement update equations**

\[
K_k = P_k^- H^T \left( H P_k^- H^T + R \right)^{-1}
\]  

(2.11)

\[
\hat{x}_k = \hat{x}_k^- + K_k \left( z_k - H\hat{x}_k^- \right)
\]  

(2.12)

\[
P_k = (I - K_k H) P_k^-  
\]  

(2.13)

The Kalman filter estimation algorithm can be though as a \textit{predictor-corrector} algorithm for solving numerical problems, where the time update
Chapter 2. System Design and Implementation

equations are predictor equations, while the measurement update equations are corrector equations.

Figure 4 below shows a complete picture of the operation of the filter according to the predictor-corrector algorithm using the equations of the Kalman filter form Table 1 and Table 2.

**Figure 4: A complete picture of the operation of the Kalman filter.**

**Implementation**

The Kalman filter implemented was designed using two different process models: the constant velocity model and the constant acceleration model.

The constant velocity model is described by the following equations:

\[
x(t + \Delta t) = x(t) + \dot{x}(t)\Delta t
\]

\[
y(t + \Delta t) = y(t) + \dot{y}(t)\Delta t
\]

\[
\dot{x}(t + \Delta t) = \dot{x}(t)
\]

\[
\dot{y}(t + \Delta t) = \dot{y}(t)
\]

where \((x, y)\) is the position and \((\dot{x}, \dot{y})\) is the velocity.

The state of the system at a time step is represented by the following vector.

\[
x_k = [x \ y \ \dot{x} \ \dot{y}]^T
\]
Chapter 2. System Design and Implementation

The following $4 \times 4$ transition matrix $A$ is used for the transition of the system to the next state.

\[
A = \begin{bmatrix}
1 & 0 & \Delta t & 0 \\
0 & 1 & 0 & \Delta t \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

The $4 \times 4$ matrix $B$ related to the control input is set to be a $4 \times 4$ identity matrix so that after it is multiplied with the control input $u_k$ the result would still be $u_k$.

The measurement matrix $H$ is

\[
H = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\end{bmatrix}
\]

so that only the position $(x, y)$ is taken into account in the system state.

The equations describing the constant acceleration model are given below.

\[
x(t + \Delta t) = x(t) + \dot{x}(t) \Delta t
\]

\[
y(t + \Delta t) = y(t) + \dot{y}(t) \Delta t
\]

\[
\dot{x}(t + \Delta t) = \dot{x}(t) + \ddot{x}(t) \Delta t
\]

\[
\dot{y}(t + \Delta t) = \dot{y}(t) + \ddot{y}(t) \Delta t
\]

\[
\ddot{x}(t + \Delta t) = \ddot{x}(t)
\]

\[
\ddot{y}(t + \Delta t) = \ddot{y}(t)
\]

The matrix $B$ is a $6 \times 6$ identity matrix.

The $6 \times 6$ matrix $A$ controlling the state transition is

\[
A = \begin{bmatrix}
1 & 0 & \Delta t & 0 & 0 & 0 \\
0 & 1 & 0 & \Delta t & 0 & 0 \\
0 & 0 & 1 & 0 & \Delta t & 0 \\
0 & 0 & 0 & 1 & 0 & \Delta t \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
Chapter 2. System Design and Implementation

The $6 \times 2$ measurement matrix $H$ is

$$
H = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

2.2.3. Video Capture and Display

OpenCV, a set of open source image processing libraries by Intel, is used to provide a basic graphical interface where the produced video can be displayed. The library offers methods for creating an image and displaying it on screen under X-window. There is a function to draw line segments of specific color and thickness between two points in an image, useful to visualize the fixation point at each time. It also provides a simple way to fetch and handle events useful for controlling the runtime behavior of the system.

The data acquired from the camera are captured using the framegrabber class in RGB 24bit format. These data are used to fill the values of an IplImage created with OpenCV. Such an image can then be displayed on a window created with cvNamedWindow using the cvShowImage function. The video encoder as well as the foveation routine requires that the image is in YUV 4:2:0 planar format. A color space conversion from RGB to YUV can be performed using OpenCV’s cvCvtColor function, which yields an YUV image with 4:4:4 chroma subsampling which can be easily downsampled to the required 4:2:0. The size of the captured frames is 288 pixels in height and 352 pixels in width which is the typical size of a CIF frame used in the PAL YUV model.

The line drawing functionality provided by the OpenCV library is used to create and display a crosshair at any point of an IplImage. The function drawCrosshair allows a crosshair of specified color to be displayed on the currently displayed video frame at the point of fixation. When the raw unfiltered data from the eye tracker are used to find the coordinates of the fixation point, a red crosshair is displayed on the video frame. Alternatively if we enable the use of Kalman filtering in the acquired coordinates a green crosshair is displayed on the point of the estimated fixation.
On Screen Display

OpenCV provides a way for drawing text strings on IplImages. The function `cvPutText` renders the text in the image with the specified font and color. This capability offered by the OpenCV library was used to implement an On Screen Display (OSD) feature to present useful information on the video window at run-time. Specifically the user can see on the screen the function mode of the system (tracking or manual), whether the gaze driven movement, the Kalman filter and the foveation are enabled or disabled and finally the value of the viewing distance set for the foveation routine. The OSD feature can be seen on Figure 5.

![Figure 5: The On Screen Display feature.](image)

The OSD feature is enabled by pressing the key ‘o’. The user can get a list of all the available software functionalities, described in section 2.4, and the corresponding keys that activate/deactivate them by using pressing the key ‘h’ which will display a help text on the video window, see Figure 6.
2.2.4. **Foveation Model**

The human eye is a very complex receiver where the density of the neurons in the retina decreases non-uniformly with the distance from the center (or fovea) of the retina. The fact that the density is highest at the fovea allows us to apply image foveation as a data compression algorithm for video coding.

When a human observer views an image the point where his eye is fixating, which is projected onto the fovea, is perceived by the HVS with the maximum resolution information of this image, while the perceived resolution at the retina rapidly falls away from the fovea. Using an eye tracker we can find the point of fixation and together with the viewing distance we can apply the foveation model which discards resolution information of the image areas that are projected on the retina away from the fovea. The resulting image will have less resolution information and therefore could be encoded using less data, but the perceived image quality form the observer will not be severely affected.

Foveation is modeled as non-uniform sampling of a 2-D signal. The maximum detectable spatial frequency at each point on an image is proportional to the density of sensor neurons at the projection of that point on the retina. By applying band limiting filters to this maximum detectable
frequency at each area of the image at the encoder the amount of information that needs to be transmitted is reduced.

**Ideal Foveation Model**

The foveation model uses a relation for the maximum detectable spatial frequency at a point of an image as a function of the coordinates of the fixation point and the viewing distance of the observer from the image. Eliminating spatial frequencies greater than the maximum detectable frequency in video coding will not compromise perceptual quality. The normalized maximum detectable frequency \( f_c \) is found using the following empirical model.

\[
f_c(x,y,x_f,y_f,V) = \frac{1}{1 + K \tan^{-1}\left(\frac{\sqrt{(x-x_f)^2 + (y-y_f)^2}}{V}\right)}
\]

(2.14)

In equation (2.14) \((x_f,y_f)\) are the coordinates of the fixation point, \(V\) is the viewing distance from the image (see Figure 7), and \(K=13.75\). Thus the ideal foveation of an image would require to locally bandlimit the image at coordinates \((x,y)\) to \((x_f,y_f)\) but that would have a great computational complexity. In order to be able to develop a practical implementation for video coding a faster alternative is used.

![Figure 7: Human eye fixation on an image.](image)

**Approximations to Ideal Foveation**

In order to reduce the computational complexity of the foveation process approximations are made to the ideal foveation model. First odd-length
even-symmetric 2-D FIR filters are used. Second only eight possible values of the maximum detectable frequency \( f_c(x,y) \) are used so that the image is divided into foveation regions every one having a constant maximum detectable frequency. Third the foveation regions are constrained to be 16 x 16 blocks, which is the size of a macroblock in most video coding standards. Finally lookup tables (LUT) are used so that pre-computed information can used to reduce the run time.

The approximate foveation model for the normalized maximum detectable frequency \( f_c \) at the center coordinates \((x,y)\) of a macroblock is

\[
f_c(x,y,x_f,y_f,V) = \min \left( \frac{i}{8} : d > B[i,V], 1 \leq i \leq 8, i \in Z^+ \right)
\]

(2.15)

\[
d = (x-x_f)^2 + (y-y_f)^2
\]

(2.16)

\[
B[i,V] = \min \left\{ r^2 : f_c(r,V) \times 8 = i, r \in R^+ \right\}
\]

(2.17)

\[
f_c(r,V) = \frac{1}{1 + K \tan^{-1} \left( \frac{r-R}{V} \right)}
\]

(2.18)

where \( K=13.75 \) as before, \( V \) is the viewing distance and its value should be one of the set of viewing distance stored in the LUT \( B \), and \( R \) is the radius of a circle centered at \((x_f,y_f)\) that will be coded with full resolution. Equations (2.17) and (2.18) are precomputed and stored in the LUT \( B \), therefore only the computation of (2.15) and (2.16) is made at run time for each macroblock.

![Figure 8: The ideal (solid) and the approximate (dashed) foveation models as function of distance from the fixation point. (Figure taken from [SLEB01])](image_url)
Foveated Video Coding

For the real-time implementation of a foveated video encoder we use the spatial domain foveation preprocessing technique described by Sheikh et al [SLEB01]. Only the luminance component $Y$ of video frame in YUV4:2:0 format is foveated since foveation of the chrominance components gives little reduction in the bit rate.

In spatial domain foveation preprocessing an initial phase in the video encoding is added where the foveation routine is applied in every frame to filter each foveation region with a low-pass filter that has a cutoff frequency equal to the maximum detectable frequency for that region. The preprocessed video frame is then encoded using a standard MPEG1 video encoder. Since the initial phase eliminates the high frequency information, less bit rate is required to code the video.

The foveation routine uses 7 foveation filters, one for each of the possible 8 foveation regions (The region where $f_c = 1$ is not filtered). The foveation region filtering used are 7-tap, even-symmetric, separable 2-D FIR filters with 16-bit fixed point coefficients. At the picture edges the signal was symmetrically extended. The filters were designed using constrained least squared error minimization in Matlab and the coefficients were scaled to give unity gain at DC. Video foveation filtering is the same as filtering an image except that the used filter is switched each time a different foveation region is found in a video frame. It is noted that foveation preprocessing is
independent of the video coding scheme used making this approach easy to use but on the other hand is slow in execution.

2.3. System Modules: Motor

2.3.1. Camera Movement

A basic feature of the implemented remote surveillance system is to allow the camera to be moved by the person who is monitoring an area. The user of the system could very easily guide the camera using the keyboard to issue specific pan/tilt commands to camera and adjust his field of vision as desired. Such commands have been implemented allowing the camera to turn towards 8 different directions according to the area in a visual frame that the user wants to have a better look at.

It is very natural though that an observer would like to center his visual field at the point of his attention. Since an eye tracking device is used to capture that point, needed as an input to the foveation routine, this point could as well be used to have a gaze driven camera movement without the need of additional commands using the keyboard.

The camera should adjust its direction only when the fixation point is to the periphery of the vision field. For that reason each video frame is divided in 9 areas as shown in Figure 10. When the fixation point is on one of the 8 positions in the periphery the appropriate move command is given to the camera. When the fixation point stays on the center of the visual field the camera remains still.
2.4. System Modules: User Interface

2.4.1. Runtime Functionalities

The simple event handling capability offered by the function `cvWaitKey` of OpenCV is used to create a simple user interface where the user can control the runtime execution of the program by pressing specific keys. This function is called periodically in every step of the main program loop and waits for key event for “delay” milliseconds. The run-time functionalities provided along with the corresponding key that activates/deactivates them are summarized on Table 3.
Table 3: Program functionalities summarized.

<table>
<thead>
<tr>
<th>Key</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>q</td>
<td>Quits the program</td>
</tr>
<tr>
<td>t</td>
<td>Toggle tracking/manual mode</td>
</tr>
<tr>
<td>i</td>
<td>Capture the current video frame</td>
</tr>
<tr>
<td>o</td>
<td>Enable On Screen Display</td>
</tr>
<tr>
<td>h</td>
<td>Display help text</td>
</tr>
<tr>
<td>g</td>
<td>Toggle gaze driven camera movement on/off</td>
</tr>
<tr>
<td>k</td>
<td>Toggle Kalman filter on/off</td>
</tr>
<tr>
<td>c</td>
<td>Toggle crosshair display</td>
</tr>
<tr>
<td>f</td>
<td>Toggle video foveation on/off</td>
</tr>
<tr>
<td>,</td>
<td>Decrease viewing distance</td>
</tr>
<tr>
<td>.</td>
<td>Increase viewing distance</td>
</tr>
<tr>
<td>w</td>
<td>Move camera towards upper left position</td>
</tr>
<tr>
<td>e</td>
<td>Move camera towards upper position</td>
</tr>
<tr>
<td>r</td>
<td>Move camera towards upper right position</td>
</tr>
<tr>
<td>s</td>
<td>Move camera towards left position</td>
</tr>
<tr>
<td>d</td>
<td>Reset camera position to center</td>
</tr>
<tr>
<td>f</td>
<td>Move camera towards right position</td>
</tr>
<tr>
<td>x</td>
<td>Move camera towards lower left position</td>
</tr>
<tr>
<td>c</td>
<td>Move camera towards lower position</td>
</tr>
<tr>
<td>v</td>
<td>Move camera towards lower right position</td>
</tr>
</tbody>
</table>

2.4.2. Command Line Arguments

Apart from the function keys that control the runtime behavior of the system a number of arguments can be passed at the command line and change the default values and initial behavior of the system.

The syntax for passing the argument on the command line is:

```bash
./vision [options]
```

The available options are summarized on Table 4.
Table 4: Command line arguments.

<table>
<thead>
<tr>
<th>Option</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>-v &lt;filename&gt;</td>
<td>Specify the filename of the resulting compressed stream. Default value is encodedvid.mpg</td>
</tr>
<tr>
<td>-r &lt;filename&gt;</td>
<td>Specify the filename where the encoded frame sizes will be saved. Default name is framesizes.txt</td>
</tr>
<tr>
<td>-f &lt;value&gt;</td>
<td>Enable foveation of the encoded stream and set the viewing distance to the specified value.</td>
</tr>
</tbody>
</table>
Chapter 3

System evaluation

3.1. Introduction

The system evaluation process aims to test whether the implemented system satisfies its design specifications and assess the usability and plausibility of the proposed hardware and software setup as a remote surveillance system. Furthermore, the performance of the foveated video encoding is evaluated as a useful coding scheme for real-time video transmission over channels with limited bandwidth.

A set of tests were conducted where a person was asked to perform certain tasks using the implemented system and his ability to accomplish them was recorded using both the spatial prioritization and a constant quality compression algorithm. The performance to the various tasks is judged based on the time needed to be completed along with the gain in bandwidth using the foveated video encoding. Various foveation degrees and thus compression degrees are tested.

3.2. Calibration Process

Figure 3 illustrates the details of the experimental setup. The following calibration sequence is then used in order to achieve the most accurate mapping from the camera referential to the stimulus referential:

- First the height of the chair and the position of the head-support are adapted to the subject.
Chapter 3. System evaluation

- Then the eye tracker camera is shifted through the controls in the ISCAN software so that the subject’s pupil is in the centre of the recorded image (see Figure 11).
- The focus of the camera and the threshold values for the pupil and corneal reflection are adjusted.
- Finally it is verified that the corneal reflection is not lost by instructing the subject to shift the gaze to all four corners of the projected image.

Now the actual calibration procedure is performed using the nine points calibration option from the ISCAN software rather than five. A single white dot, on which the subject is instructed to focus, is presented to the subject over a black background at nine different positions (see Figure 12). Finally the success of the calibration is double-checked by instructing the subject to shift his gaze over the calibration points.

![Figure 11: Close-up of subject’s eye with crosses indicating the tracked pupil and corneal reflection.](image-url)
3.3. Experiment 1: Compression Capabilities

3.3.1. Description and Setup

The first set of experiments that were performed test the compression rate achieved with the use of foveation. The percentage of the reduced frame size was recorded for different amounts of foveation.

For the purpose of this experiment, the camera was placed at a fixed position with the gaze driven movement disabled. That way the same frame was encoded at all runs. The viewing distance, affecting the foveated area and thus the level of compression was different at each run. In total 20 values of the viewing distance where tested (from 1 to 20 with a step of 1). For this purpose 100 frames of the same visual scene were encoded recording for each frame the resulting size of the foveated and non foveated encoded frame.

3.3.2. Results and Analysis

The non-foveated encoding as well as foveated encodings (for various values of the viewing distance) of the reference frame are shown in Figure 13.
Chapter 3. System evaluation

Figure 13: Non foveated encoding (upper left) and foveated encodings of the same frame for various values of the viewing distance. The values of viewing distance for the foveation of the various frames are: upper right (V=6), lower left (V=11), and lower right (V=16).

The average sizes of the encoded frames of the non-foveated and foveated compressions for the various values of the viewing distance are presented on Table 5. In Figure 14 a graph of the average frame size of non-foveated and foveated encoding is shown while in Figure 15 a graph with the percentage of additional compression achieved in foveated encodings is presented for different values of the viewing distance.
Table 5: Level of compression for various amounts of foveation. Column 1 lists the viewing distance \( V \), column 2 and 3 the sizes in bytes of non-foveated and foveated encoding of the same frame, and column 4 the gain in size achieved using foveation.

<table>
<thead>
<tr>
<th>( V )</th>
<th>Non-foveated encoding</th>
<th>Foveated encoding</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3905</td>
<td>2885</td>
<td>0.26</td>
</tr>
<tr>
<td>2</td>
<td>4058</td>
<td>3186</td>
<td>0.21</td>
</tr>
<tr>
<td>3</td>
<td>4011</td>
<td>3210</td>
<td>0.20</td>
</tr>
<tr>
<td>4</td>
<td>3975</td>
<td>2993</td>
<td>0.25</td>
</tr>
<tr>
<td>5</td>
<td>4005</td>
<td>3101</td>
<td>0.23</td>
</tr>
<tr>
<td>6</td>
<td>4055</td>
<td>3116</td>
<td>0.23</td>
</tr>
<tr>
<td>7</td>
<td>3988</td>
<td>3129</td>
<td>0.22</td>
</tr>
<tr>
<td>8</td>
<td>3896</td>
<td>3168</td>
<td>0.19</td>
</tr>
<tr>
<td>9</td>
<td>3991</td>
<td>3394</td>
<td>0.15</td>
</tr>
<tr>
<td>10</td>
<td>3820</td>
<td>3193</td>
<td>0.16</td>
</tr>
<tr>
<td>11</td>
<td>3886</td>
<td>3271</td>
<td>0.16</td>
</tr>
<tr>
<td>12</td>
<td>3919</td>
<td>3490</td>
<td>0.11</td>
</tr>
<tr>
<td>13</td>
<td>4018</td>
<td>3416</td>
<td>0.15</td>
</tr>
<tr>
<td>14</td>
<td>3988</td>
<td>3490</td>
<td>0.12</td>
</tr>
<tr>
<td>15</td>
<td>3922</td>
<td>3369</td>
<td>0.14</td>
</tr>
<tr>
<td>16</td>
<td>3885</td>
<td>3588</td>
<td>0.08</td>
</tr>
<tr>
<td>17</td>
<td>3991</td>
<td>3576</td>
<td>0.10</td>
</tr>
<tr>
<td>18</td>
<td>3878</td>
<td>3677</td>
<td>0.05</td>
</tr>
<tr>
<td>19</td>
<td>3854</td>
<td>3424</td>
<td>0.11</td>
</tr>
<tr>
<td>20</td>
<td>4074</td>
<td>3802</td>
<td>0.07</td>
</tr>
</tbody>
</table>

![Average frame size for non-foveated and foveated encoding](image)

Figure 14: Average frame size of non-foveated and foveated encoding versus the value of the viewing distance \( V \).
Chapter 3. System evaluation

Figure 15: Percentage of additional compression when using foveation versus the value of the viewing distance $V$.

The resulting encoded frame sizes after the use of foveation presented on Table 5 reveal the usefulness of this technique in video compression. We can also see how the size of the foveated area affects the level of compression. The ideal parameters for foveation that would result in video streams with the higher compression possible without compromising the video quality could be found by comparing the various video streams. These parameters though, would be different in each case based on the conditions of the experiment (distance of the eyes of the observer from the video fed, type of video stream etc.).

The additional compression level achieved with the use of foveation is also dependant of the captured scene. The values for the gain in size of the compressed stream shown on Table 5 are only relevant to the specific test. Higher or lower values could be achieved for different type of encoded frames where more or less information could be discarded.

This experiment demonstrates the effect of foveation on the resulting video stream size; the smaller the central foveation region is (where the maximum spatial frequency of the frame is retained), the higher the amount of compression of the video stream is. Furthermore there is an amount of foveation where the tradeoff between the reduction in video size and the degradation of video quality is acceptable and the foveated video coding is preferable than any other uniform quality compression algorithm, in cases of video transmission over limited bandwidth.
Chapter 3. System evaluation

3.4. Experiment 2: Reading Task

3.4.1. Description and Setup

This task tests the quality of the encoded video using various amounts of foveation based on the ability of an individual to read with ease a small passage. A person was asked to read a small text of about 70 words and the time needed was recorded.

The camera was turned to the text and the encoded video stream was projected to a screen on the wall, where the person could read it while the eye tracker was capturing the point of fixation. The camera was fixed during the test and the gaze driven movement was disabled since the whole text was inside the vision field. The task was repeated for various values of the viewing distance $V$ determining the amount of the foveation effect. For lower values of the viewing distance the video frames are more heavily foveated and the encoded frame size smaller.

The task was repeated for 3 different values of the viewing distance and the person was asked to read the text using foveated and non foveated video encoding in turns. In total the reading task was repeated 6 times using a different text each time. The texts were different to avoid having known content affect the reading time. They were all chosen though from the sports news section of the BBC website in order to have the same degree of difficulty as far as it concerns the required knowledge of English.

A non-foveated encoding together with foveated encodings (for various values of the viewing distance) of the texts used for the reading task are shown in Figure 16.
Figure 16: Non foveated encoding (upper left) and foveated encodings for various values of the viewing distance of the texts used for the reading task. The values of viewing distance for the foveation of the various texts are: upper right (V=7), lower left (V=13), and lower right (V=19).

3.4.2. Results and Analysis

The reading times as well as the compression gain for each repetition of the task are summarized on Table 6.
**Chapter 3. System evaluation**

**Table 6: Summarized results for reading task. (The dash in the viewing distance column means non foveated video stream. The reading time is in ms.)**

<table>
<thead>
<tr>
<th>Text</th>
<th>Words</th>
<th>V</th>
<th>Time</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arsenal</td>
<td>73</td>
<td>7</td>
<td>42877</td>
<td>0.28</td>
</tr>
<tr>
<td>Aston Villa</td>
<td>79</td>
<td>-</td>
<td>26118</td>
<td>-</td>
</tr>
<tr>
<td>Birmingham</td>
<td>71</td>
<td>13</td>
<td>37916</td>
<td>0.26</td>
</tr>
<tr>
<td>Blackburn Rovers</td>
<td>73</td>
<td>-</td>
<td>24758</td>
<td>-</td>
</tr>
<tr>
<td>Bolton</td>
<td>74</td>
<td>19</td>
<td>26399</td>
<td>0.09</td>
</tr>
<tr>
<td>Charlton</td>
<td>72</td>
<td>-</td>
<td>20918</td>
<td>-</td>
</tr>
</tbody>
</table>

In general the person was able to read the text with ease using both foveated and non foveated video streams with a constant flow of his speech in most cases. The reading times were slower for non foveated video streams though. A small delay was observed on the flow of speech when the reader had to change line and consequently the region which was encoded with the higher spatial frequency had to be moved across the video frame.

Even though the reading times where shorter for non foveated video encodings the ability of a person to read the text viewing the foveated video stream was found adequate. Furthermore excellence in such reading tasks is not a part of the design specifications of the implemented system, which is designed to be a surveillance system. The current performance, given the gain in size using the foveated encoding, is found to be satisfactory.

Further analysis in order to find the ideal amount of foveation which would result in a quality of the encoded video stream that would not cause any disruptions to a human reader could be found by additional testing. Nevertheless that would require the execution of psychophysical experiments beyond the scope of this project.

Consequently the quality of foveated video stream is proven to be efficient even for tasks requiring high detail video encoding. Thus using foveated encoding in a surveillance system would offer the required lever of video quality without requiring an extreme amount of bandwidth in order to be transmitted.
3.5. Experiment 3: Object Tracking Task

3.5.1. Description

The following sets of experiments examine the object tracking and identification capabilities of the system testing its performance as a surveillance system. A person was asked to find and name a number of objects in a foveated and non foveated video stream and the time to complete this task was recorded.

3.5.2. Gaze Driven Camera Movement Disabled

3.5.2.1. Setup

The first set of object tracking tests was performed with the gaze driven camera movement disabled. The camera was fixed across two shelves on which the objects needed to be identified were randomly placed. Objects of different shapes and colors were chosen. Namely the person was asked to identify inside the visual scene a pair of scissors, a lock, a roll of black tape, a small black box, a transparent plastic cup, and a yellow and a red round plastic object (see Figure 17). More specifically the task was to fixate at each object, with no particular order, and identify it by its name.

![Figure 17: The setting of the tracking task and the objects used.](image.png)
Chapter 3. System evaluation

There were overall 10 repetitions of this task were 5 different values for the viewing distance were selected resulting in various amounts of foveation. The objects were repositioned in every run without the observer being able to see the setting prior to the beginning of the experiment. Non foveated and foveated video streams were projected one at a turn trying to have similar setting in the compared foveated and non foveated version of the identification task. The time needed to name all of the objects was recorded for each run.

In Figure 18 foveated encodings of the video stream during the object tracking task are shown for various values of the viewing distance.

![Figure 18: Foveated encodings from the object tracking task with the gaze driven camera movement disabled. The values of viewing distance for the foveation of the various frames are: upper left (V=2) upper right (V=3), lower left (V=4), and lower right (V=5).](image)

3.5.2.2. Results and Analysis

The identification times as well as the compression gain for each repetition of the task are summarized on Table 7.
Chapter 3. System evaluation

Table 7: Summarized results for object tracking task with gaze driven camera movement disabled. (The dash in the viewing distance column means non foveated video stream. The reading time is in ms.)

<table>
<thead>
<tr>
<th>Run</th>
<th>Time</th>
<th>V</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10080</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>15517</td>
<td>1</td>
<td>0.11</td>
</tr>
<tr>
<td>3</td>
<td>10198</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>10078</td>
<td>2</td>
<td>0.10</td>
</tr>
<tr>
<td>5</td>
<td>9718</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>15998</td>
<td>3</td>
<td>0.12</td>
</tr>
<tr>
<td>7</td>
<td>8040</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>8960</td>
<td>4</td>
<td>0.21</td>
</tr>
<tr>
<td>9</td>
<td>9599</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>12319</td>
<td>5</td>
<td>0.15</td>
</tr>
</tbody>
</table>

The user of the system had no problem to identify all of the objects within a certain time period for both non-foveated and foveated video streams. The time needed for the completion of the task was somewhat less for non foveated streams. The identification time for non foveated streams was around 10 sec while for foveated streams was 10 to 16 sec.

The difference in time for non-foveated and foveated encoding is not considered to be significant and does not affect negatively the performance of the system. Furthermore the additional amount of compression added thanks to the spatial prioritization encoding scheme favors the use of foveated streams for the purpose of the system.

During the execution of the above tests it was noted that the user had no problem to track objects of various colors and shapes and further to distinguish between of objects of the same color and similar shape (the black tape and the black box). The viewer could even identify in the foveated streams objects which had low contrast with their background (transparent plastic cup).

The excellent performance on this object tracking task using foveated video stream makes evident that the use of such a technique is perfectly suitable for the purposes of the designed surveillance system; it serves both the need for low-bandwidth transmission and high enough video quality in order to identify various objects in a visual scene.
3.5.3. Gaze Driven Camera Movement Enabled

3.5.3.1. Setup

For the second set of object tracking tests, the gaze driven camera movement was enabled and the camera was free to pan/tilt according to the point of fixation of the observer. Like before, the camera was placed across the two shelves but now a zoom factor was used to have the camera look at only a part of the search space. The objects used were a pair of scissors, a lock, a roll of black tape, a small black plastic cover, and a transparent plastic cup. Again the task was to fixate at each object and name it but now the person would have to search the whole area of the two shelves using the moving capability of the camera.

The task was repeated 10 times using 5 different viewing distance values for the foveation routine. The conditions of the experiment were the same as before, that is the objects were repositioned after each turn with the observer having no knowledge of their initial position. Furthermore the boundaries of the search area were known to the observer so that he wouldn’t have to turn the camera away from a specified region.

In Figure 19 foveated encodings of the video stream during the search task are shown for various values of the viewing distance.
Figure 19: Foveated encodings from the object tracking task with the gaze driven camera movement enabled. The values of viewing distance for the foveation of the various frames are: upper left (V=1) upper right (V=2), lower left (V=3), and lower right (V=5).

3.5.3.2. Results and Analysis

The identification times as well as the compression gain for each repetition of the task are summarized on Table 8.
Table 8: Summarized results for object tracking task with gaze driven camera movement enabled. (The dash in the viewing distance column means non-foveated video stream. The reading time is in ms.)

<table>
<thead>
<tr>
<th>Run</th>
<th>Time</th>
<th>V</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>36477</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>120551</td>
<td>1</td>
<td>0.16</td>
</tr>
<tr>
<td>3</td>
<td>40916</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>100595</td>
<td>2</td>
<td>0.14</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>53475</td>
<td>3</td>
<td>0.11</td>
</tr>
<tr>
<td>7</td>
<td>81114</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>4</td>
<td>0.15</td>
</tr>
<tr>
<td>9</td>
<td>29397</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>45435</td>
<td>5</td>
<td>0.18</td>
</tr>
</tbody>
</table>

The summarized results of Table 8 reveal a different performance of the system in this search task. The time the user needed to identify all of the objects was notably less for the non-foveated version of the video encoding. Interestingly, there were 2 cases where the viewer couldn’t track all of the objects (Noted with a dash on the time column of Table 8). The time needed for non-foveated streams was 30-53 sec while for the foveated streams was 45-120 sec.

This more complex search task, where the whole area was not revealed to the viewer, made the tracking of the objects more difficult and resulted in bigger search times. The viewer now had to search each area exhaustively before moving to a different region.

As noted before there were 2 cases where the viewer couldn’t identify all of the objects. This did not happen only for foveated encoded streams though. The object not identified in both cases was the transparent plastic cup due to its low contrast with its background (see Figure 20). Consequently foveation was not the reason to blame for the failure of the observer to identify the object; it was due to its placement (area with bright background) and the quality of the compression algorithm applied after the foveation that made the object indiscernible.
Judging by the performance of the system in this task one might think that the use of foveated video streams is inappropriate. Nevertheless considering the reduction in frame sizes and the fact that the viewer eventually does identify all of the objects (except the cases already discussed) the use of foveation seems to be an attractive technique for better utilization of the available bandwidth. Furthermore we could achieve better results by fine tuning the parameters of the foveation and compression.

This experiment presented the gaze driven camera movement capability of the system and demonstrated how it can assist a search and tracking task. The designed surveillance system provides a complete framework where a limited bandwidth line can be utilized to both guide a camera and transfer a video stream of high enough perceptual quality.
Chapter 4

Conclusion

Foveated encoding is an attractive approach for real-time video stream transmission. It’s a biological based technique that like all lossy video compression standards makes use of some aspects of HVS modeling. The resulting size of the encoded stream is lower than that with a use of a standard encoder while the perceptual quality is not sacrificed.

We have implemented a system that shows how an eye tracker can be used to find the fixation point and apply the foveation model. Furthermore a real-time application of the foveated encoding for a surveillance system is demonstrated. The issues of system precision and noise elimination were overcome using Kalman filtering while a gaze driven camera movement was implemented to facilitate the use of the system.

A set of evaluation tests have been conducted in order to assess the performance of the implemented system. The experiments have shown that a person could successfully perform reading and object tracking tasks viewing the foveated video stream. The times needed to complete such tasks were in most cases competitive to those that if a standard encoder was used for the same task. The compression achieved by the use of foveation was 10-30% more than that of standard encoder for the various cases. Consequently foveated encoding is an appealing method to use for the designed surveillance system where the video stream has to be transmitted over a limited bandwidth internet line.

The only missing component in the implemented system is to have the camera in a distant site and actually transmit the encoded stream to the PC that will decode the stream rather than perform the encoding/decoding at the
Chapter 4. Conclusion

same machine. That could be easily done by incorporating the foveation routine to an H.263 video encoder part of the H.323 protocol suitable for internet real-time applications, thanks to the fact that foveation preprocessing is independent of the video encoder.
Bibliography


[LBK99] Sanghoon Lee, Alan C. Bovik, Young Yong Kim: Low Delay Foveated Visual Communications over Wireless Channels. ICIP (3) 1999: 90-94


[RS99] Uwe Rauschenbach and Heidrun Schumann, "Demand-driven image transmission with levels of detail and regions of interest", Computer & Graphics 23, pp857-866, 1999

[RL02] Reingold E.M.; Loschky L.C., Saliency of peripheral targets in gaze-contingent multiresolutional displays, Behavior Research Methods, Instruments, & Computers, 1 November 2002, vol. 34, no. 4, pp. 491-499(9)


