

REMOTE GAMING ON RESOURCE CONSTRAINED DEVICES

by

Waazim Reza

A Thesis Submitted to the Faculty of
The College of Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Florida Atlantic University

Boca Raton, Florida

December 2010

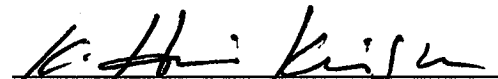
REMOTE GAMING ON RESOURCE CONSTRAINED DEVICES

by

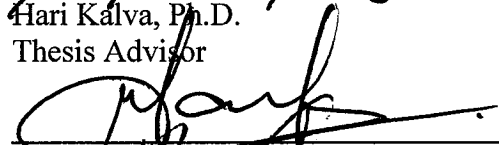
Waazim Reza

This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Hari Kalva, Department of Computer and Electrical Engineering and Computer Science, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

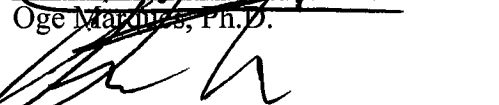
SUPERVISORY COMMITTEE:



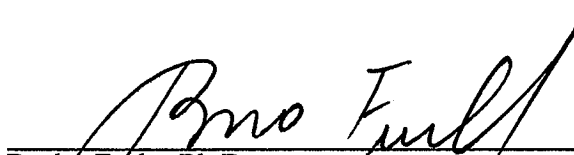
Hari Kalva, Ph.D.
Thesis Advisor



Oge Marques, Ph.D.

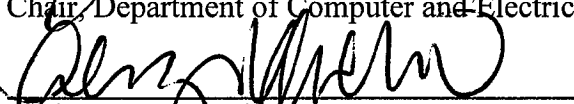


Daniel Raviv, Ph.D.



Borko Furht, Ph.D.

Chair, Department of Computer and Electrical Engineering and Computer Science



Karl K. Stevens, Ph.D., P.E.

Dean, College of Engineering and Computer Science



Barry T. Rosson, Ph.D.

Dean, Graduate College

December 7, 2010
Date

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Dr. Hari Kalva, for his support and direction. He has been patient, encouraging and a great source of motivation throughout my tenure here at FAU. He has inspired me from his calm and serene demeanor even in toughest of situations. His technical and editorial advice was essential to the completion of this master's thesis and has taught me innumerable lessons and insights.

I would like to sincerely thank my committee members Dr. Oge Marques and Dr. Daniel Raviv for their valuable comments, suggestions and input to the thesis. Thanks a lot for your patience and time.

I would like to thank Dr. Thanh Tran from Texas Instruments, for the initial idea and the motivation for the project. Also, Richard Kaufman helped a lot in the implementation of the game. I would like to thank all my colleagues - Sagar, Velibor, Rafael and others in Multimedia lab at FAU with whom I had technical discussions and collaborated on multiple projects. Thanks to Lakitosh, Aneesh and Baishali for being like a family and supporting me.

I express my deepest love and gratitude to my parents U Abdul Salam and Zainaba, for their unparalleled and unconditional support throughout my life. My brothers Reza and Zaqie, have been a great support. Finally, I am extremely grateful for the support from my beautiful wife, Jasna. Jasna, your understanding and patience made this effort possible.

ABSTRACT

Author: Waazim Reza
Title: Remote Gaming on Resource Constrained Devices
Institution: Florida Atlantic University
Thesis Advisor: Dr. Hari Kalva
Degree: Master of Science
Year: 2010

Games have become important applications on mobile devices. A mobile gaming approach known as remote gaming is being developed to support games on low cost mobile devices. In the remote gaming approach, the responsibility of rendering a game and advancing the game play is put on remote servers instead of the resource constrained mobile devices. The games rendered on the servers are encoded as video and streamed to mobile devices. Mobile devices gather user input and stream the commands back to the servers to advance game play. With this solution, mobile devices with video playback and network connectivity can become game consoles. In this thesis, we present the design and development of such a system and evaluate the performance and design considerations to maximize the end user gaming experience. A gaming user experience model capable of predicting the user experience for a given gaming session is developed and verified.

DEDICATION

This thesis is dedicated to my wonderful family.

“You have been with me every step of my way, through good and bad times. Thank you for all the unconditional love, guidance, and support that you have always given me, helping me to succeed and instilling in me the confidence that I am capable of doing anything I put my mind to. Thank you for everything. I love you all!”

REMOTE GAMING ON RESOURCE CONSTRAINED DEVICES

LIST OF FIGURES	ix
INTRODUCTION	1
1.1 Motivation	3
1.2 Background	5
1.3 Scope of the Thesis	8
PROBLEM DESCRIPTION.....	10
2.1 Delays.....	12
2.2 Response Time	13
2.3 Video Quality	13
2.4 Proposed Solution	15
SYSTEM DESIGN AND ARCHITECTURE.....	16
3.1 Game Control Unit.....	17
3.2 Live Game.....	17
3.3 Video Encoder.....	19
3.3.1 MPlayer	20
3.3.2 FFMPEG.....	21
3.3.3 Intel Media SDK.....	21
3.3.4 Integrated Performance Primitive.....	22
3.4 Live video streamer.....	23
3.5 Video player	25
3.6 Gaming commands.....	25
3.7 Server Implementation	26
PERFORMANCE EVALUATION	27
4.1 MOS Formulation	29
4.2 Gaming user experience evaluation case study.....	32
4.2.1 Initial Study	32
4.2.2 Experimental Setup.....	34
4.3 Influence of encoder parameters on MOS.....	39

4.4 Influence of game parameters on MOS.....	44
4.5 Gaming user experience model validation	46
CONCLUSION.....	49
FUTURE WORK.....	51
BIBLIOGRAPHY.....	53

LIST OF FIGURES

Figure 1.1 Cisco Visual Network Index: Forecast and Methodology, 2009-2014	4
Figure 1.2 OnLive Architecture	6
Figure 2.1 Round Trip Response Time.....	11
Figure 3.1 System design and architecture	16
Figure 3.2 Live game snap shot	19
Figure 3.3 Remote Gaming Thread Flow	26
Figure 4.1 Parameters effecting Gaming User Experience.....	29
Figure 4.2 Relation between MOS and R	30
Figure 4.3 Influence of GOP size on average bandwidth used.....	33
Figure 4.4 Influence of IP distance on average bandwidth used	33
Figure 4.5 Experimental setup	34
Figure 4.6 Influence of bit rate on MOS.....	40
Figure 4.7 Influence of bit rate on I_{BR}	40
Figure 4.8 Influence of GOP size on MOS.....	42
Figure 4.9 Influence of resolution on MOS	43
Figure 4.10 Influence of enemy speed on MOS	44
Figure 4.11 Influence of enemy shooting speed on MOS	45
Figure 4.12 Predicted Vs Subjective gaming user experience using average method.....	47
Figure 4.13 Predicted Vs Subjective gaming user experience using proportion method .	48

CHAPTER 1

INTRODUCTION

Games constitute one of the most dynamic area, both with respect to evolution of technology and market. Games have evolved over time from simple two dimensional graphics games like Pong to intriguing three dimensional games like James Cameron's Avatar 3D which approaches real world actions. With this advancement, there have also been high demands on the hardware requirements for the computers and mobile devices in which the games can be played. High end games like James Cameron's Avatar 3D require dedicated graphics processing unit (GPU) which would accelerate 3D graphics rendering, so that end user has a high gaming experience. Mobile devices like mobile phones and other handheld devices have inherent limitations of limited processing resources and storage capabilities. This disadvantage limits mobile devices from playing any advanced game on such platforms. Thus, considering these limitations, a gaming solution has to be explored that would serve gaming in mobile devices flawlessly consuming minimum processing power of the device.

With the emergence of cheaper and faster broadband networks, playing games over the internet has been extremely popular. Comscore reports the total worldwide online gaming community to be 217 million people [1]. With one in four internet users visiting a gaming

site [1], games are more accessible and have become a part of everyday life. As the market for smart phones like Iphone and Android is booming up, the mobile gaming arena is advancing into a higher user experience. However there has been a considerable restriction of the type of the games that a user can play on a mobile device inherently because of the device limitations such as memory, processing, and storage capabilities.

The main goal of this thesis is to present the design and development of a remote gaming solution to deliver games to low cost mobile devices. This thesis firstly introduces challenges in the design and implementation of a remote gaming system and then proposes solution for the same. Design alternatives have to be studied and a remote gaming platform should be developed to study the operation of the games in a wireless network. The impact of video encoding and game configuration on user experience has to be observed. Quality of the video has to be quantified. The video encoder adapts itself to the state of the environment. The tradeoff between the response time and the quality of the video has to be studied. After every session of the game, stats like frame information (frame type and frame size), encoder information (encoder type and encoding time), game statistics (points information and game delays) has to be logged. This information can be archived and used for gaining information about the wireless network and the gaming user.

1.1 MOTIVATION

The advent of high end mobile devices, like smart phones and net books, together with cheaper and faster wireless networks, is making mobile access to Internet a reality. A wide variety of mobile devices like mobile phones, music players and note books are introduced to decode and play these media contents. Apart from the primary use as a communication device, mobile phones are started being equipped with a variety of multimedia capabilities including media players despite their limited system resources.

Also, there has been a boom in the number of digital video content users in recent years. Advances in the storage and compression technologies and improvements in the speed of internet connection have enabled widespread use of multimedia content. These advancements open up promising possibilities for mobile users to play high graphic content, rich games through wireless networks, from their mobile devices.

According to a forecast of IP traffic by Cisco [2], annual global IP traffic will reach two-thirds of a zettabyte. The report which is an initiative to determine the Cisco Visual Networking Index also forecasts that by 2013, video will be 90 percent of all consumer IP traffic and 64 percent of mobile traffic. This tremendous growth in the video traffic is represented in the figure 1.1. Due to this growth it is quite common to see huge servers hosting dramatically increased video contents. Users prefer to watch these contents instantly, rather than waiting for the content to be downloaded. Therefore, enabling mobile remote gaming will significantly change the face of the gaming. The gaming user experience of mobile users who used thin gaming will be much lesser than the same user having a remote gaming experience. It will be difficult to achieve the above goal using

the current client-server gaming architecture because most of the storage and computational complexity of the games still lies on the client device and the client devices are limited by available resources.

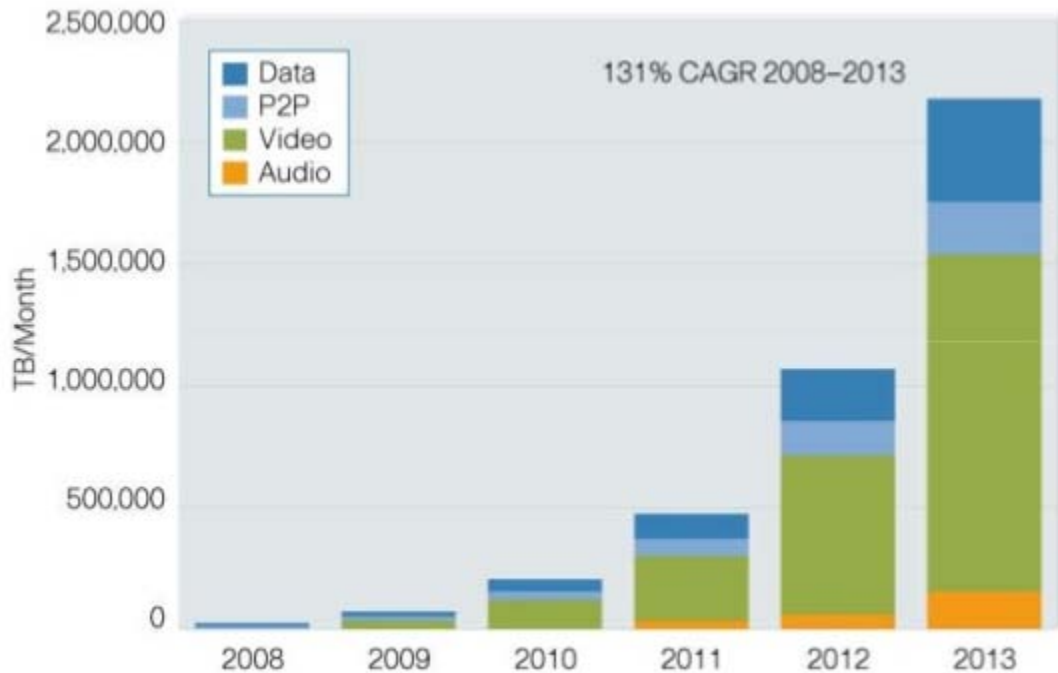


Figure 1.1 Cisco Visual Network Index: Forecast and Methodology, 2009-2014
Source: Cisco

Development of a game platform that would help to study delays associated in every stage of a server client gaming architecture, as pursued in this thesis, is thus motivated. In particular, the development focuses on achieving the following key features:

- No downloading of the game on a client device is required
- No waiting time for physical media to spin up required. The game is instantly available.
- Any game can be played in a cheap mobile device without any GPU requirements.

- A user can play any game irrespective of the platform of the client device.

1.2 BACKGROUND

Many ubiquitous gaming approaches were proposed in the past; one such example is GamePod[3]. Issues of managing multiple gaming environments are addressed by GamePod, which would reduce the hassles of carrying bulky laptops and PCs for gaming purposes. GamePod operates by encapsulating a user's gaming session in a virtualized execution environment and storing all the states associated with the session on a portable storage device. This portable storage device has to be carried with the user to continue the game from the saved state of the game. The idea of remote gaming on a mobile device is not considered, which could provide better gaming user experience if one could play a game on the move.

OnLive[4] is a new gaming service, released in the end of 2009. It works in a server-client fashion. The games are hosted in OnLive data centers on custom high-performance game servers. A high speed broadband internet connects the Onlive MicroConsole, PC or Mac to OnLive servers. On receiving an input from the controllers, a low latency video is sent to the client as a response from the server. OnLive claims to use a unique and powerful video compression algorithm using multiple GPUs to accomplish this.



Figure 1.2 OnLive Architecture

Source: www.OnLive.com

Figure 1.2 represents architecture of the OnLive system. At present, all OnLive games are at HDTV resolution. The resolution of the streamed video is not adjusted dynamically according to the bandwidth availability and network traffic. For best video quality, at least a 5 Mbps wired Internet connection is recommended, but a minimum of 3 Mbps is required. The HD videos make it mandatory for the system to have high speed internet connection. The most important drawback of OnLive is that games cannot be played on a mobile device on a wireless network. Moreover, it has restrictions on the screen resolutions of the client computers which suggest that OnLive is not a completely portable gaming solution.

Research were also done on topics like application level QoS for Games on Demand [5], using an adaptation solution in a client-server implementation. The solution employs network monitoring and rate adaptive video encoding. Optimization of Round Trip Time (RTT) by tuning the encoder parameters like bit rate (changing encoder quantization parameter) is focused. The client programs are run in a set top box and the results with and without adaptation solution were studied. Codec selection of H264 and MPEG2 are not focused, subjective evaluation of the gaming user experience and adaptation of the game delays from inside the game is also not explored.

Previous work on hybrid thin-client protocol for multimedia streaming and interactive gaming application [6] provides a solution that would help a system with thin clients which has minimum graphic processing capability. The operation of the system is classified into two scenarios - in regime scenario and distributed scenarios. 'In regime' scenario is the normal case of operation where equal gaps are placed between GOPs to decode and display. The distributed scenario is based on a feedback system in which the frame rates are adaptively controlled.

Recent papers on Remote Server Based Mobile Gaming (RSBMG) approach discusses about a remote gaming model where screen shots of games are streamed to the user [7] [8].The authors studied the impact of the wireless network conditions such as network congestion and outages. Since users typically do not have control over network conditions, game experience can be improved by appropriately adapting the other components of the system. RSBMG does not consider the effects of video codec selection

and the broader question of maximizing gaming experience using the right configuration of video codecs and game parameters. Another important problem is the selection of the right game for a set of conditions (network and device).

To evaluate the performance of the system we need to have a way of measuring the quality of user experience in the remote gaming setup. Many approaches were proposed to model the quality of experience [9] [10] [11] of the streamed video [12]. Studies were also conducted to model and understand the effect of wireless networks on video delivery [13] [14]. Since gaming is a highly interactive application, video quality metrics and tools, like PSNR and VQM [15], cannot be directly applied to measure the quality of user experience. Previous work on evaluating user experience in gaming [16] [17] [18] [19] has focused only on conventional PC games and not on remote gaming.

1.3 SCOPE OF THE THESIS

The tasks and scope of the thesis are as follows:

- Development of a gaming platform that will enable a user to run a high graphics game which require heavy computing and processing capabilities in a low cost remote device with adaption to the state of the environment..
- Determination of principal parameters, like game parameters and encoder parameters, affecting the performance of the system.
- Develop and validate a model that would quantify gaming user experience.

The thesis has resulted in a computational tool that can be used for determining the optimal operating environment for a particular game.

CHAPTER 2

PROBLEM DESCRIPTION

The proposed remote gaming system runs the game at a server. It starts streaming the game on receiving a request from a client to initiate the game service. The game video is received, decoded and displayed at the client mobile device. The user sends appropriate response back to the server in the form of gaming commands. Time interval between client's gaming command input and the moment the video response is displayed at the client is called as the round trip response time. Remote gaming system faces a lot of challenges as it has to deal with sending continuous coded video data. The challenges are as described below:

- 1) Bandwidth: The remote gaming application demands a high bandwidth to achieve a high gaming user experience. However, the current internet does not provide bandwidth reservation to support this requirement. Also, network congestion could produce a low gaming user experience. Appropriate mechanisms to face congestion are an advantage for the system.
- 2) Delay: If the delay is not bounded, the playback is paused and annoys the human eyes as a result of which the user experience is degraded. A video packet that arrives late can

be assumed to be as a lost packet. However, the delay in wireless network is not constant, thus a buffer at the receiver is used before decoding.

3) Loss: Packet loss is inevitable in a wireless video transmission system. However, a game video decoded at very bad quality due to packet losses produces a low gaming user experience. Thus, it is desirable to have a system that is robust to packet loss. For example, if the video is encoded at a higher GOP size, it would possibly lose more data than that of lower GOP size. Therefore, reducing the GOP size is one way of controlling packet loss.

4) Decoding complexity: Some mobile devices such as mobile phones and personal digital assistants (PDAs) require low power consumption. Therefore, the applications running on these mobile devices must be simple. In particular, low decoding complexity is desirable.

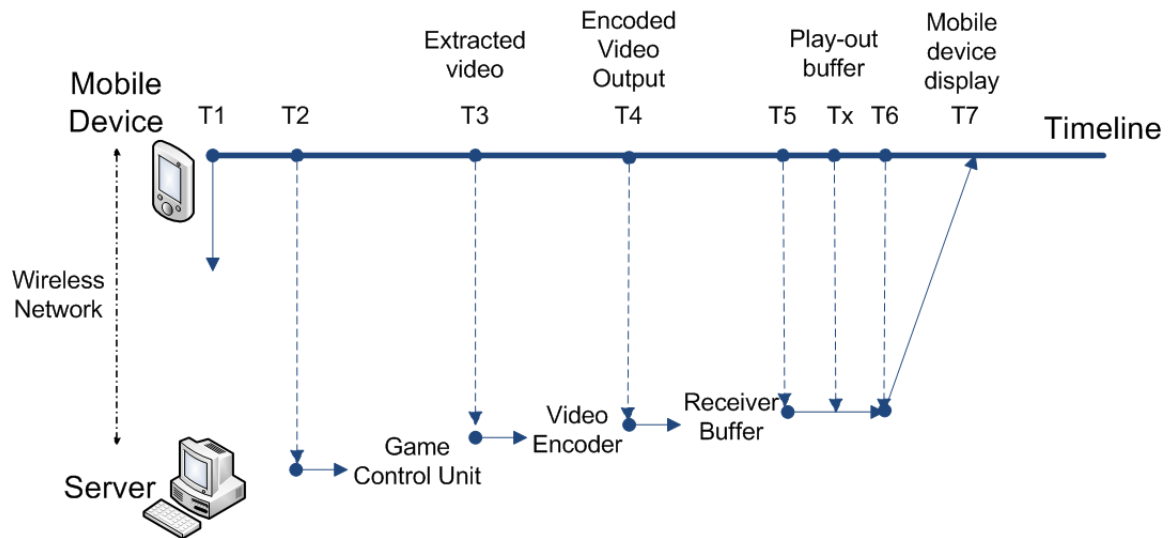


Figure 2.1 Round Trip Response Time

2.1 DELAYS

As the gaming commands are sent by the mobile device to the server, the response is sent from server to the client. The response from the server is an encoded video. The client device receives the encoded video; it decodes the video and displays it. Figure 2.1 represents a round trip response time; from the moment a gaming command on the mobile device is sent from the client device to the moment resulting video frame is received at the mobile device. At time T_1 , mobile device sends a gaming command to the server. The command is received by the server at time T_2 . Server processes the command and renders the game scene at time T_3 . The encoded and packetized scene is sent to the client at time T_4 . All the packets of the scene are expected to completely arrive at T_5 . However, due to the downlink delay variation, the whole scene can arrive earlier or later than T_5 . The packets that arrive early can be cached in a data buffer. The received frames of the late arriving packets are displayed after a short duration called play-out buffer. This removes the impairments of late coming packets. At time T_6 , packets are removed from the buffers de-packetized. The response video is displayed at user's device.

2.2 RESPONSE TIME

According to the analysis in 2.1, gaming response time in our approach mainly includes network uplink delay D_{UL} (T1-T2); server delay D_S , including game engine delay (T2-T3) and video encoding delay (T3-T4); network downlink delay D_{DL} (T4-T5); and client delay D_C (T5- T7), including client play-out delay D_{PL} (T5-T6) and decoding delay (T6-T7). Therefore, Response Time (RT) can be described as:

$$\text{Network Uplink Delay} = D_{UL} = (T1-T2) \quad (2.1)$$

$$\text{Game Engine Delay} = D_{GE} = (T2-T3) \quad (2.2)$$

$$\text{Video Encoder Delay} = D_{VE} = (T3-T4) \quad (2.3)$$

$$\text{Network Downlink Delay} = D_{DL} = (T4-T5) \quad (2.4)$$

$$\text{Client Caching Delay} = D_{CC} = (T5-T6) \quad (2.5)$$

$$\text{Decoding Delay} = D_D = (T6-T7) \quad (2.6)$$

$$RT = D_{UL} + D_{GE} + D_{VE} + D_{DL} + D_{CC} + D_D \quad (2.7)$$

2.3 VIDEO QUALITY

Video quality is as important as the response time of the gaming session. A video that is streamed with highest possible quality and sent to a user with minimum response time would have maximum gaming user experience. However, minimum response time is achieved at high compression of the video and since compression is “lossy”, minimum

response time is achieved at the expense of the video quality. Hence there has to be a tradeoff between response time and video quality. Thus video quality plays an important role in determining gaming user experience. The quality of the video can be rated with a score on the basis of the subjective or objective determination. Subjective measurements [20][21] can provide a realistic guide to the video quality perceived by a user. This subjective video quality score is named Mean Opinion Score (MOS) which is generated by averaging the results of a set of standard, subjective tests where a number of users rate the quality on a five point scale from 1 (Bad / Very Annoying) to 5 (Excellent / Imperceptible impairments). The range of the MOS also can be mapped to be another style from 0 to 100. The diverse form of MOS is Difference MOS (DMOS), which has the same principle like the MOS but measures the difference between the original and distorted video frames. MOS and DMOS can reflect the quality of the video signals correctly because they are obtained by using HVS directly. But actually, the subjective video quality measurement methods are complex and not always available. Objective measurements are automatic and readily repeatable but the complex nature of human visual perception [22] makes it difficult to accurately model the response of a human observer. The widely-used peak-signal-to-noise ratio (PSNR) does not correlate well with subjective responses to visual quality [23]. More sophisticated objective quality models have been proposed [24][25][26] but as yet there is no clear replacement for subjective measurements. Various metrics [27] are used for objective quality evaluation like data metrics, Picture metrics, Packet or bit stream-based metric, hybrid metric, full reference, no reference and reduced reference metrics.

2.4 PROPOSED SOLUTION

We propose a client-server based solution which is ideal for the problem in hand. The proposed solution measures delays as mentioned in section 2.2 at every stage of the game. An expensive and fast server with game engine, live video encoder and video streamer modules designed. Each of the modules is independent enough to be configured with different parameters in order to tweak the operation of the module. A low complexity game which could be completely configurable is desired for the solution. The game should be completely controllable so that one can change the pace of the game and collect the stats of the game after every session of the game. The live video encoder at the server should have capabilities of reading the configuration from a file with features like encoder type, number of I, P, B frames, frame spacing and GOP information. The live video encoder can change the encoding parameters on the fly depending on the bandwidth variations. The solution also tries to adjust if the delays exceed above a threshold by preventive actions on the live encoder and streamer. The server streams the video frames to a client with low processing capability.

CHAPTER 3

SYSTEM DESIGN AND ARCHITECTURE

In this chapter, we present system design and architecture of the proposed remote gaming system. As shown in figure 3.1, the system comprises of a server that streams the game to a device with a video playback and user input control. The components of the system are described in the following sub-sections.

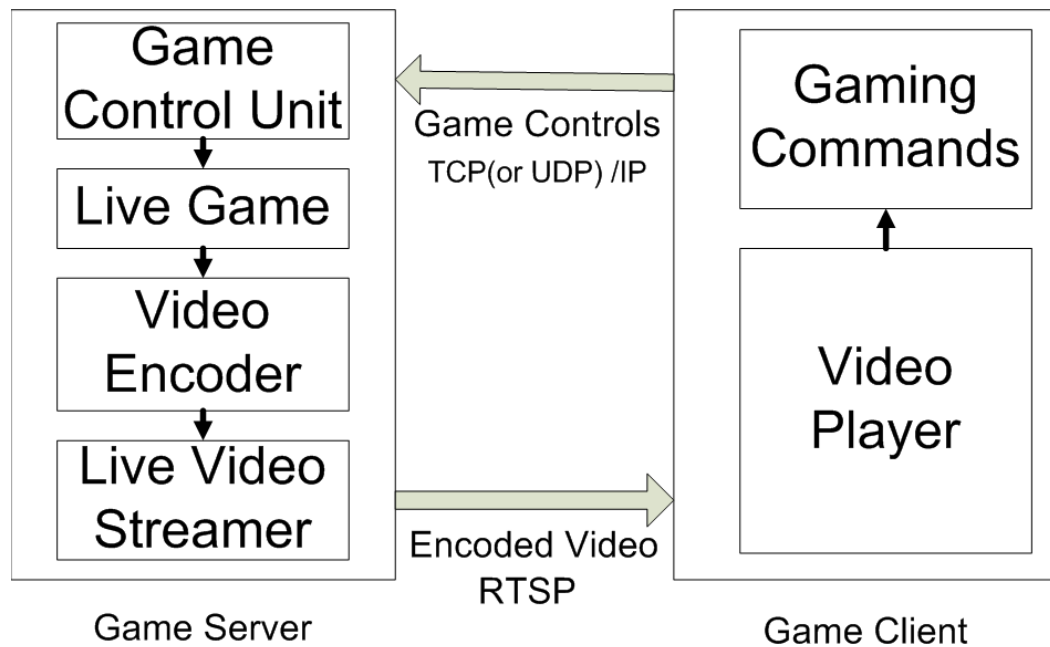


Figure 3.1 System design and architecture

3.1 GAME CONTROL UNIT

Game control unit receives the gaming commands sent from the remote client. The gaming commands are captured from user input, fed to the encoder, and the game streamed to users. The commands are typical controls such as keyboard inputs. The gaming commands are processed by the game control unit at the server and appropriate game parameters are set in the live game module. Any keyboard input which does not change the control of the game is ignored. So the game control unit has the most important task to translate the incoming gaming commands to meaningful game events like firing event and player movement event. The game control unit is implemented using a socket program which continuously checks for game controls in a particular port. The sockets are implemented using Windows socket APIs.

3.2 LIVE GAME

We developed a simple live game using windows DirectX APIs. A shooting game was selected as the theme of the game as it is highly delay sensitive and even milliseconds of delay would reflect in user's gaming experience. The game involves two characters or objects- the player and the enemy, who are associated with different health levels. Both the characters shoot at each other and try to avoid being hit. As the game proceeds, the health of the player and enemy reduces as it gets hit by bullets. If the health of the player reduces below a particular threshold, the color of the background and the player change. So, the motion in the game involves motion due to the player, enemy, motion of bullets and the changes in background.

The game can be tweaked using a configuration file that allows changing the pace of the game, the motion of the enemy and the player, and also controls the shooting speed. These features are used as different test cases to measure and model gaming user experience. YUV frames are extracted from the game and fed to the video encoder. These frames are encoded and streamed at a desired frame rate.

Racing games like need for speed or strategy games like World of Warcraft (WoW) could also be used in our system by capturing the screen at a particular frame rate, encoding it and sending to the client. This approach would have saved time spent on developing a game from scratch. However, such an approach does not provide flexibility in the game and does not allow the game to be tweaked to experiment with delays in the game and encoder.

Figure 3.2 shows a snapshot of the live game at a state where the enemy and the player are shooting against each other.



Figure 3.2 Live game snap shot

3.3 VIDEO ENCODER

An interactive gaming application over a network requires a live encoder which will adapt dynamically according to the changing video content, losses in the channel and bandwidth conditions. Our contributions in the thesis, is the selection of video encoder implementation which would be of great use for any future work in remote gaming. We designed a video encoder module that fits into the gaming system in a modular fashion which makes the system scalable and maintainable. The requirements of an effective encoder implementation were listed and that are as below:

- The encoder module and the live game code should be integrated as a single unit, so that the YUV frames extracted from the live game module is directly fed into the encoder module.
- Since the live game code is based on Windows DirectX API, the encoder implementation should be able to run in a Windows environment.
- The encoder module needs to have a configuration file which has the list of encoding parameters that can be tuned in order to have a varied video encoding experience.

Considering the above mentioned design requirements four open source encoder implementations were studied.

3.3.1 MPlayer

The initial candidate for the live encoder module was MPlayer[28], a GUI based movie player for Linux. We started off with MPlayer because its command line options give a lot of configurability. MPlayer is capable of playing videos of formats - AVI /ASF /OGG/ DVD / VCD / VOB / MPG / MOV. MPlayer code includes an encoder called MEncoder. MEncoder is designed to encode MPlayer-playable movies to other MPlayer-playable formats. MEncoder is only a wrapper which uses codecs such as MPEG-4 (DivX4), libavcodec, PCM/MP3/VBR MP3 audio underneath it. MPlayer is rejected because it is based on a Linux platform.

3.3.2 FFMPEG

FFMPEG [29], an open source solution to record, convert and stream audio and video, was also considered as an option for the live encoder. It supports a significant number of codecs like MPEG2, H.264, Ogg and AAC. It also supports many ARM optimizations. FFMPEG can be compiled in windows, but the code has large amount of dependencies that makes it difficult to integrate with the live game code. As the encoder implementation was intended to use in a Linux environment, the code was not performance optimized for windows. This would produce additional load on the encoder.

3.3.3 Intel Media SDK

In order to optimize the encoder module running on a windows server with Intel processors, Intel encoder implementations were considered. Intel Media SDK and Intel Integrated Performance Primitive (IPP) are the two Intel encoders available for implementing the video encoder. Intel Media Software Development Kit (Intel Media SDK) [30], a software development library that exposes the media acceleration capabilities of Intel platforms for decoding, encoding and video preprocessing. Intel Media SDK enables developers to take advantage of hardware acceleration in Intel platforms. The API library supports a broad selection of hardware platforms including Intel Integrated Graphics chipsets, Intel Architecture Processors and third-party graphics platforms. Intel Media SDK supports encoding of AVC/H.264/MPEG-4 part 10 and MPEG-2 video and decode support for AVC/H.264, MPEG-2 video, and VC-1. The

encoder parameters are initialized before the encoding cycle starts and thus, there is no way to change the encoding parameters in the middle of an encoding cycle. Also, Intel Media SDK does not provide full access to the encoder code base and the outer wrapper API accesses only the library files and DLLs.

3.3.4 Integrated Performance Primitive

Intel Integrated Performance Primitive (IPP) [31] is a software library that provides broad range of functionality in the domain of signal and image processing, computer vision, speech recognition, video coding and 3D data processing delivering a variety of options to choose when designing and optimizing an application. An advantage of IPP is that it does not use complex data structures, which helps to reduce overall execution overhead. IPP provides a rich library of reusable code for implementing video encoders and decoders in the form of simple to use APIs. An encoder instance was created using the IPP APIs and was configured to set the parameters required from a parameter file (.par file). The configuration file provides an easier way of controlled experiments with the encoding parameter like bit rate, resolution, and frame rate. The live game module was properly channeled to the encoder module so that the extracted YUV frames in the live game module are encoded by the live encoder. Enormous amount of time was spent in choosing the right video encoder and integrating it in to the above mentioned remote gaming system. The knowledge that was gained was surely a contribution of this thesis.

3.4 LIVE VIDEO STREAMER

Video streaming is a video service in which a video file you receive is not entirely cached locally. The user does not have to wait for the video to be downloaded entirely to the cache. Any delay in playing subsequent frames will cause user dissatisfaction leading to low gaming user experience. The live game that is being streamed is very time sensitive. Even milliseconds of delay would cause low user experience. So the design of a live video streamer module had to consider the following factors:

- End to end delay has to be minimal and constant.
- In order to reduce the delay, lowest possible cache has to be used at the receiver end.
- As far as gaming user experience is considered, delay is more crucial than the quality of the video

In order to accomplish the above design criteria, network streaming protocols like TCP, UDP and RTP/RTSP were considered. TCP or Transmission Control Protocol [32] is based on a hand shake or packet acknowledgement system. Whenever a packet is lost, a retransmission of the packet is done. The client will not be able to playback the video unless it receives the packet. This retransmission mechanism causes delay which is highly unacceptable. UDP or User Datagram Protocol [33] was also analyzed to incorporate in the streaming system. Unlike TCP, UDP does not rely on a positive acknowledgment from the client on the receipt of a packet. UDP does not have any bounds of losses. The server keeps sending packets at a constant rate without any regard

to the state of the network. Even if the network is congested, the server continues to send at the same rate. However, UDP is a media unaware protocol. A media aware protocol would be aware of the transmission characteristics and the application being used. UDP uses a checksum comparison to check for the packet integrity at nodes in the network. Irrespective of the number of corrupted bits, if a packet is corrupt due to bit errors, the packet is entirely discarded. But the game is less sensitive to a small amount of bit errors than delays. Thus, we need a protocol that understands the media that is being transmitted and tries to reduce the delay as much as possible.

RTSP satisfies the needs of minimal delay for the media transmission. Live video streamer is implemented using LIVE 555 Media Server, an open source media streaming application [35] used for streaming data over a RTSP, RTP and SDP connection. The Media Server uses RTSP protocol for streaming. It can stream several kinds of media files including MPEG2 and H264 videos. These streams can be received and played by any standard RTP/RTSP media clients. VLC media player and Quick time media player are the two examples of RTSP/ RTP compliant media players we have used for evaluation. The open RTSP command-line RTSP client can receive/store the stream data, without playing the received media. The key features of LIVE555 Media Server are:

- The server can transmit multiple streams of same file or multiple streams of different files concurrently.
- The media streams are transmitted as RTP/UDP packets by default or tunneled through HTTP

- The server can also support raw UDP streaming for non-standard clients.

3.5 VIDEO PLAYER

Any RTP/RTSP media client can be used as the video player. We have used VLC media player and Quick time media player for our experiments. VLC is a multimedia player which supports most audio and video formats such as H.264, Ogg, DivX, MKV, TS, MPEG-2, mp3, MPEG-4, AAC from files, physical media such as DVDs, VCD, Audio CD, TV capture cards and also many streaming protocols [36]. We use the VLC media player as a client for receiving the video stream.

3.6 GAMING COMMANDS

Gaming commands user interface lets the user to input the gaming commands and sent it to the server. Gaming commands are captured using socket code for the client. Sockets were implemented using windows sockets API or WinSock API. WinSock API defines a standard interface between a Windows TCP/IP client application (such as an FTP client or a web browser) and the underlying TCP/IP protocol stack. Thus, the gaming commands are sent back to the server using TCP protocol. This is a lightly loaded program which endlessly scans for any user input. Whenever a user input is done, the input is sent to the servers IP address.

3.7 SERVER IMPLEMENTATION

The server was run on a 64-bit Windows 7 operating system machine with an Intel Core 2 Quad CPU Q8200 @2.33 GHz processor running on 8 GB RAM. The server code was neatly divided into different modules and run as different as separate threads. The game control unit, live game and video encoder run independently as separate threads. The live video streamer is an independent application in itself. The encoded video output from the video encoder is fed into the live video streamer using pipes. Standard output of the video encoder is piped in to the standard input of the live video streamer. Figure 3.3 shows the separate threads and the way the data is shared between the threads.

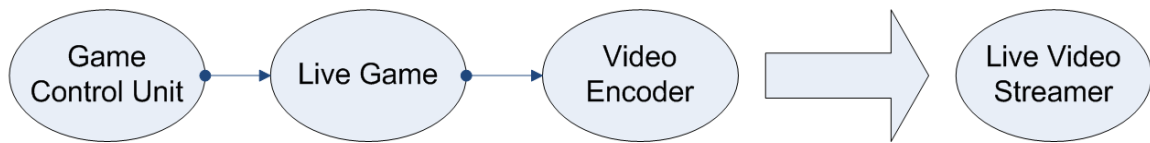


Figure 3.3 Remote Gaming Thread Flow

CHAPTER 4

PERFORMANCE EVALUATION

The performance of the remote gaming system can be improved by tuning the system parameters such that the gaming user experience is improved. This requirement demands for an analytical relationship between the gaming user experience and system parameters. Gaming user experience is a subjective quantity that depends on multiple factors. The precise definition of gaming user experience is very difficult, as it depends on the mood of the user, user's previous experience with similar games, game response time and quality of the rendered game. The exact game response time can be measured by recording the game at the client when the user played and the video quality, by comparing the client side and server side video. However, these measurements are time consuming and costly. Thus, a user case study can be used to measure the gaming user experience in terms of Mean Opinion Score (MOS). MOS is a subjective scoring method in which users score from 1(bad) to 5(excellent). MOS can be used to address the gaming scenario as a measure of delay and quality of the perceived video by the user.

The game parameters that affect the gaming user experience are the enemy speed and enemy shooting speed. As the enemy speed increases, the player is unable to catch up with the enemy. As the network delay increases, the problem worsens.

Player actions are perceived with a delay and by then enemy moves to a different position. This contributes to user dissatisfaction, yielding a low gaming user experience. Thus the impact of game parameter on gaming user experience has to be analyzed for a particular network condition.

Encoder configuration has a big impact on the response time and video quality. Encoder parameters like frame resolution, bit rate, distance between I and P frames and the Group of Pictures (GOP) size affects the gaming user experience. Also, since a lossy compression is used at the encoder before sending the game to the client device, video quality degradation is highly probable in the decoded video when perceived by the user at the client side, which could reduce the gaming user experience. With lower GOP-sizes, video quality degradation will be less in the case of packet loss for streaming video. If game video is encoded at a very high resolution, network congestion would cause problems to send these packets to the destination and thus could cause dissatisfaction among the gaming users due to the late arrival of packets or loss of packets. On the other hand, if a game video is encoded at a smallest resolution, the user would not enjoy the game in a resolution smaller than that of his mobile device itself. Also, if the bit rate of the encoded frame sent to the user is greater than the data rate of the channel, there are high chances of packet loss and delay. Thus, some of the packets may arrive late and the play-out buffer has to be increased to have a smooth playback. Thus, as shown in figure 4.1, the factors affecting the gaming user experience are those due to the encoding parameters and game parameters.

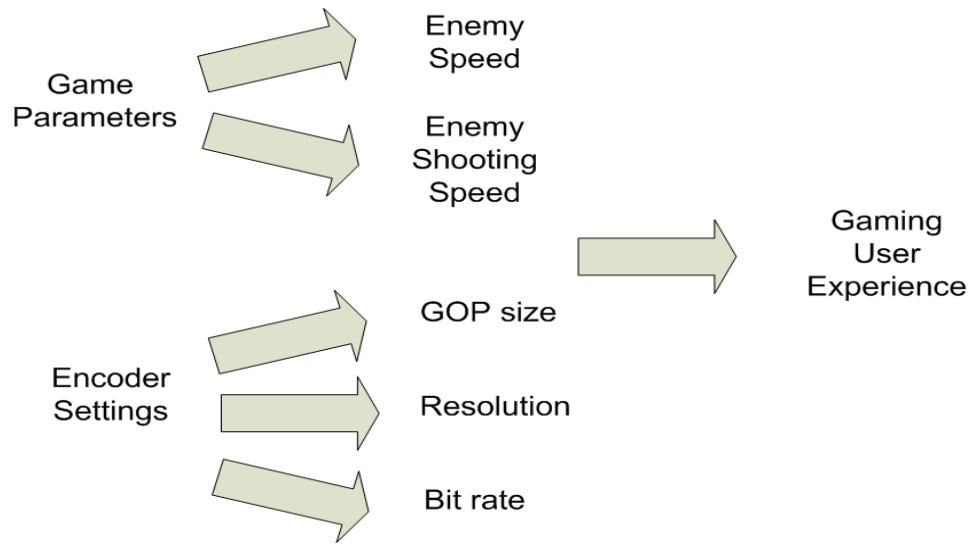


Figure 4.1 Parameters effecting Gaming User Experience

4.1 MOS FORMULATION

Since the game user experience depends on the above mentioned subjective factors, a mean opinion score method can be used to quantify the gaming user experience. In audio and video services, Mean Opinion Score (MOS), a subjective score that ranges from 1 (bad) to 5 (excellent), has been widely used to measure the perceived quality. From the above discussion, we can assume that the MOS is a function of encoding parameters and the gaming parameters. Thus the MOS can be formulated as:

$$\text{MOS} = f(\text{Encoder}, \text{Game}) \quad (4.1)$$

ITU-T E-model for transmission planning [37], which was initially intended for audio transmission, describes a computational model, known as the E-model, which can be used to compute the MOS analytically. The primary output from the model is the "rating

factor" R which can be transformed to give estimates of customer opinion. The transmission rating factor R can lie in the range from 0 to 100, where R = 0 represents an extremely bad quality and R = 100 represents a very high quality. The formulation of MOS as proposed by the model is as follows in equation 4.2 and the relation is as plotted in figure 4.2:

$$\text{MOS} = 1 + 0.035xR + Rx(R-60) \times (100-R) \times 7 \times 10^{-6} \quad (4.2)$$

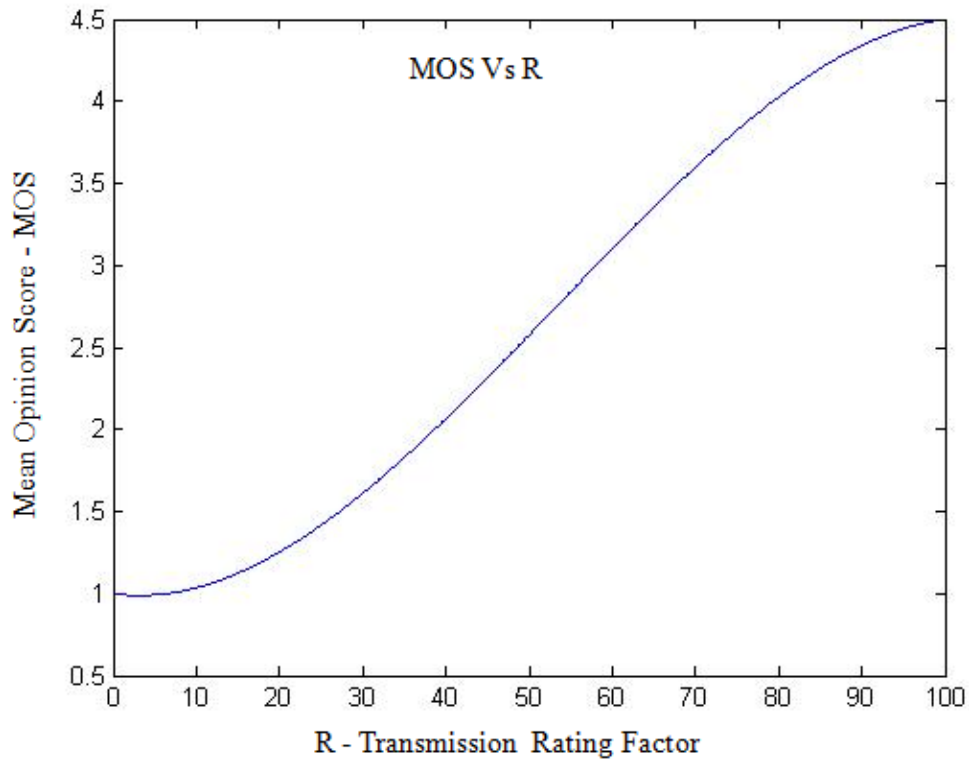


Figure 4.2 Relation between MOS and R

The E-model is based on an "impairment factor principle". According to this principle, all relevant degradations are transformed to the rating factor R. In our system we assume to

have two major impairments. Impairments caused due to the encoding settings (I_E) and those due to gaming parameters (I_G). Thus, R can be formulated as below:

$$R = 100 - I_E - I_G \quad (4.3)$$

The encoder parameters that affect the gaming user experience are the bit rate, GOP size and video resolution. Hence,

$$I_E = \alpha * I_{BR} + \beta * I_{GOP} + \gamma * I_{RES} \quad (4.4)$$

Where,

I_{BR} = impairment caused due to variation in bit rate

I_{GOP} = impairment caused due to variation in GOP size

I_{RES} = impairment caused due to variation in video resolution

I_{BR} , I_{GOP} and I_{RES} affect the MOS in different proportions. α , β and γ are the parameters that represents these proportions.

Also game parameters that affect the gaming user experience are the enemy speed and enemy shooting speed. Hence,

$$I_G = \delta * I_{ES} + \epsilon * I_{ESS} \quad (4.5)$$

Where,

I_{ES} = impairment caused due to variation in enemy speed

I_{ESS} = impairment caused due to variation in enemy shooting speed

δ, ϵ = represents the proportions by which the I_{ES} and I_{ESS} respectively, affect MOS in the presence of multiple imparities.

From equations 4.2, 4.3, 4.4 and 4.5, it is obvious one can determine MOS from the impairments I_{BR} , I_{GOP} , I_{RES} , I_{ES} and I_{ESS} .

4.2 GAMING USER EXPERIENCE EVALUATION CASE STUDY

4.2.1 Initial Study

As an initial study, the game was run with different GOP sizes and the bandwidth was measured using Wireshark [38] tool. As the GOP size increases, the bandwidth used also increases and after a GOP size of 12 the bandwidth used remains constant. Also, with lower GOP- sizes, video quality degradation will be less due to packet loss. So as shown in figure 4.3, optimal GOP size can be selected as 12.

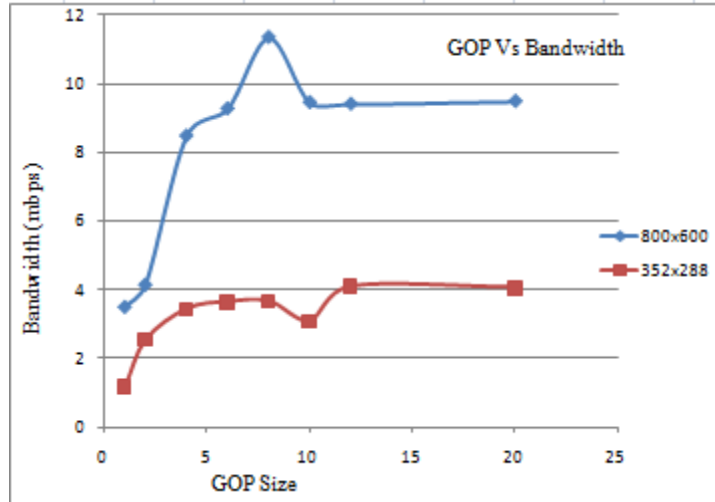


Figure 4.3 Influence of GOP size on average bandwidth used

Similar experiments with IP distance produced less variation in the bandwidth used by the application. The dependency of IP distance versus the bandwidth is as shown in Fig 4.4.

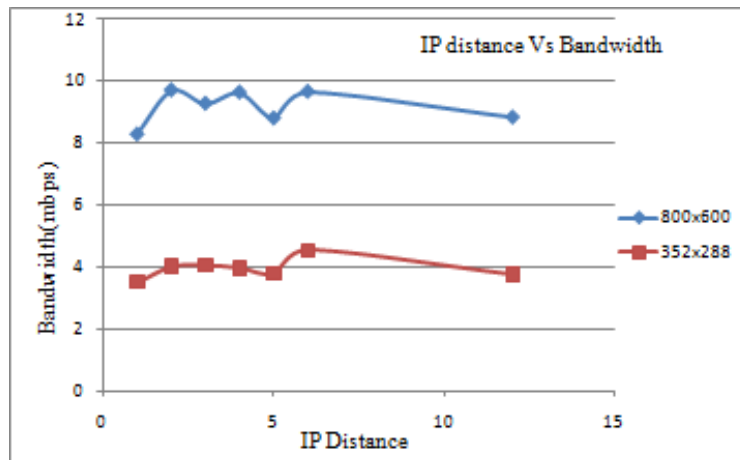


Figure 4.4 Influence of IP distance on average bandwidth used

4.2.2 Experimental Setup

Understanding the above stated relationships of bandwidth versus GOP size and IP distance, a gaming user experience evaluation study was conducted which involved sixteen game sessions conducted for fifteen subjects of the age group 23-30. Since the game being run is a simple arcade shooter game, only a minute is assigned for every session. Each of the game sessions signify a controlled experiment where the game parameters or the encoder parameters are varied. The subject user is given enough demonstrations of the game before the actual session starts, so that the subject gets accustomed with the gaming environment. The games were streamed to client laptops rather than mobile devices, for convenience. However, porting the solution to client devices is an easy task. Since the end user is expected to play in a mobile device, the game was encoded at mobile resolutions of 176x144, 352x288 and 800x600. The game was encoded with MPEG-2 codec. H.264 faced issues with implementation as there was no existing implementation for H.264 live 555 streaming. Also, some generally used MPEG-2 bit rates were used for the experiments. All the gaming sessions are decoded and displayed at the same display resolution of 700x400 at the client, which provides uniform display experience for all the sessions.



Figure 4.5 Experimental setup

A gaming user experience evaluation form with following rating levels were given to the subjects and they were asked to rate the sixteen sessions accordingly.

MOS	Quality	Impairment
4.5 – 5.0	Excellent	Imperceptible
4.0 – 4.5	Good	Perceptible but not annoying, will not quit the game
3.0 – 4.0	Fair	Slightly Annoying, might quit the game
2.0 – 3.0	Poor	Annoying, usually quit the game
1.0 - 2.0	Bad	Very Annoying, definitely quit the game

Table 4.1 MOS scaling used in the gaming user experience evaluation form

From equation 4.2, the appropriate R values for the above MOS scale are:

MOS	R Value
4.5 – 5.0	100
4.0 – 4.5	80-100
3.0 – 4.0	60-80
2.0– 3.0	40-60

1.0- 2.0	0-40
----------	------

Table 4.2 Relationship between MOS and transmission rating factor R

Sessions 1, 2 and 3 are controlled experiments studying the relationship of the gaming user experience versus the bit rate. The game was encoded at a resolution of 352x288 and the bit rates 600 kbps, 2 mbps and 5 mbps were used for sessions 1, 2 and 3 respectively.

Sessions 4, 5 and 6 were conducted with a fixed bit rate of 5 mbps and resolution of 176x144, 352x288 and 800x600 respectively. Sessions 7,8 and 9 were conducted at a fixed resolution of 800x600 encoded at a bit rate of 5 mbps and with a variable GOP size of (1,1) ,(12,3) and (30,12) respectively. Sessions 10, 11, 12 are replicas of sessions 7, 8, 9 but conducted at a lower bit rate of 600 kbps. Sessions 13-16 make use of the game parameters. Session 13 is conducted at a higher enemy speed factor of 3.0 and session 14 with a lower enemy speed factor of 2.0. Session 15 is conducted at a lower enemy shooting speed of 30 and session 16 with a higher shooting speed of 15.

The gaming user experience can also be measured from the above experiments from the game statistics like number of the hits taken by the enemy and number of hits taken by the player. This is an objective way of measuring the user experience in terms of game points. The averaged MOS of the sixteen sessions after user study was noted and they are as shown below:

Session #	Parameter Name	Parameter Value	Average MOS	R
1	Bit rate	0.6	3.0	74
2	Bit rate	2	4.1	84
3	Bit rate	5	4.26	85
4	Resolution	176x144	2	40
5	Resolution	352x288	4.16	86
6	Resolution	800x600	4.26	90
7	GOP(M,N)	(0,1)	3.34	67
8	GOP(M,N)	(12,3)	2.96	59
9	GOP(M,N)	(30,12)	2.9	58
10	GOP(M,N)	(0,1)	2.5	50
11	GOP(M,N)	(12,3)	3.52	70
12	GOP(M,N)	(30,12)	3.8	76
13	Enemy Speed	3	1	0
14	Enemy Speed	2	4.1	84
15	Enemy shooting speed	30	3.9	78
16	Enemy shooting speed	10	3.91	78

Table 4.3 User case study results

As mentioned in (4.4) and (4.5),

$$I_E + I_G = (\alpha * I_{BR} + \beta * I_{GOP} + \gamma * I_{RES}) + (\delta * I_{ES} + \epsilon * I_{ESS}) \quad (4.6)$$

During the bit rate variation, it is assumed that all the other parameters are kept at its best value, so that the impairment caused due to them is zero. Using equation 4.3, since all other impairments are zero, I_{BR} can be calculated as below:

$$R = 100 - I_{BR} \quad (4.7)$$

Thus, as the bit rate varies from 74 to 85, I_{BR} will change from 26 to 15.

Rate of change of R and I_{BR} for bit rate variation is

$$\Delta(R_{BR}) = \Delta(I_{BR}) = \text{abs}|85-74| = 11$$

Similarly, Rate of change of R and I_{GOP} for GOP size variation is

$$\Delta(R_{GOP}) = \Delta(I_{GOP}) = \text{abs}|58-67| = 9$$

Rate of change of R and I_{RES} for resolution variation is

$$\Delta(R_{RES}) = \Delta(I_{RES}) = \text{abs}|90-40| = 50$$

Rate of change of R and I_{ES} for enemy speed variation is

$$\Delta(R_{ES}) = \Delta(I_{ES}) = \text{abs}|0-84| = 84$$

Rate of change of I_{ESS} for enemy shooting speed variation is

$$\Delta(R_{ESS}) = \Delta(I_{ESS}) = \text{abs}|78-84| = 6$$

Since $\Delta(R_{ESS})$ is negligible, ϵ can be neglected. This implies that variation in enemy shooting speed does not produce major variation in the gaming user experience. Hence the term I_{ESS} in 4.6 can be neglected.

$$I_E + I_G = \alpha * I_{BR} + \beta * I_{GOP} + \gamma * I_{RES} + \delta * I_{ES} \quad (4.8)$$

Assuming that the proportions α , β , γ and δ are more or less equal.

$$I_E + I_G = (I_{BR} + I_{GOP} + I_{RES} + I_{ES}) / 4 \quad (4.9)$$

However, the encoding and gaming parameters affect the gaming user experience in different proportions. Thus the averaging done in equation 4.9 would cause masking of the MOS values and smoothen out the MOS curve. More accurate values for these proportions can be determined by computing the rate of change in the impairments when the game and encoder parameters are varied within their extreme parameter values.

From the above results, α can be calculated as below:

$$\begin{aligned} \alpha &= \Delta(I_{BR}) / (\Delta(I_{BR}) + \Delta(I_{GOP}) + \Delta(I_{ES})) \\ &= 11/154 \\ &= 0.07 \end{aligned}$$

Similarly,

$$\beta = 0.06, \gamma = 0.3 \text{ and } \delta = 0.54$$

Thus, substituting the values $\alpha = 0.07$, $\beta = 0.06$, $\gamma = 0.3$ and $\delta = 0.54$ in (4.8)

$$I_E + I_G = 0.07 * I_{BR} + 0.06 * I_{GOP} + 0.3 * I_{RES} + 0.54 * I_{ES} \quad (4.10)$$

Equation 4.10 clearly states that the video resolution and the game parameter enemy speed have a high influence on the MOS.

4.3 INFLUENCE OF ENCODER PARAMETERS ON MOS

From game sessions 1, 2 and 3, influence of bit rate on MOS was studied. As shown in table 4.3, as the bit rate increases from 600 kbps to 5 mbps, the MOS improves from 3.8 to 4.26. When bit rate increases, the delay and the quality of the rendered game video

increases. The delay as perceived by the user was minimal and user was satisfied with the video quality improvement. Thus the user experience improved with bit rate increase.

The relationship between the bit rate and MOS is as shown in figure 4.5.

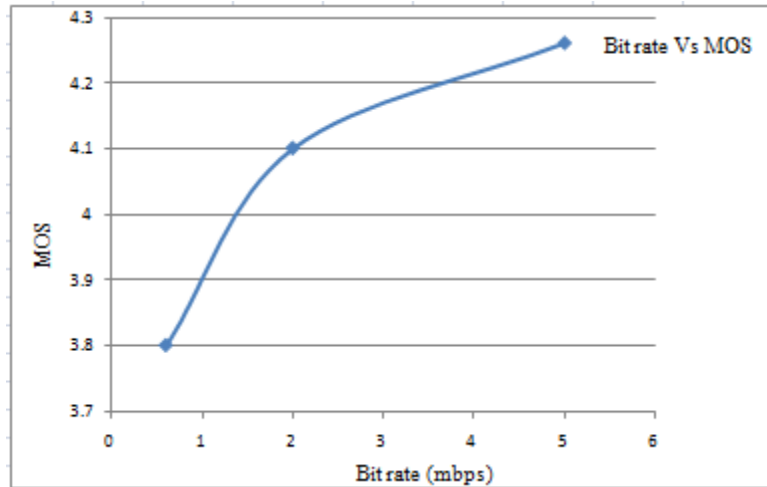


Figure 4.6 Influence of bit rate on MOS

The relationship between I_{BR} and bit rate is as shown in figure 4.6.

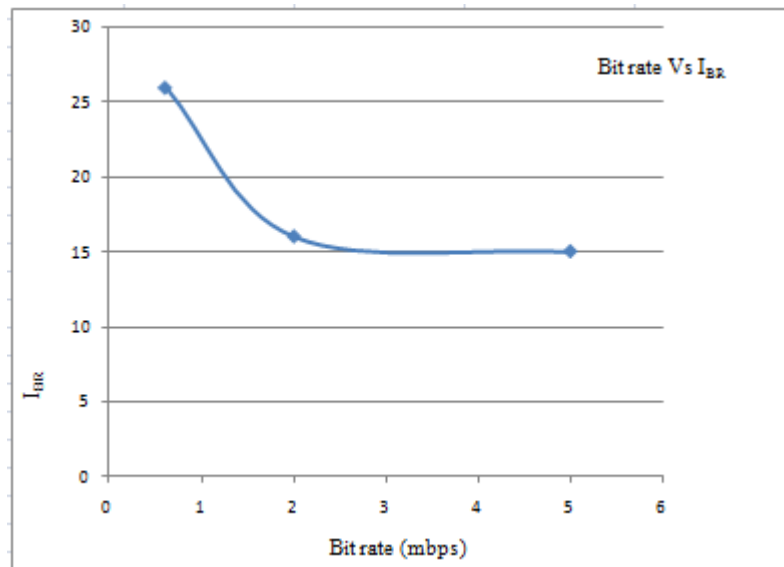


Figure 4.7 Influence of bit rate on I_{BR}

From the figure 4.6, using linear regression analysis I_{BR} is formulated as below.

$$I_{BR} = \begin{cases} 10 * \frac{BR - BR_2}{BR_1 - BR_2} + 16 & \text{If } (BR_1 < BR \leq BR_2) \\ 15 & \text{If } (BR > BR_2) \end{cases} \quad (4.11)$$

Where,

$BR_1 = 0.6$ (600Kbps)

$BR_2 = 2$ (2mbps)

The values of BR_1 and BR_2 would vary for different genres of games, which can be experimentally calculated using user case studies.

Similarly from the sessions 7, 8 and 9, the influence of GOP size was studied and plotted as shown in figure 4.7. A similar linear regression analysis was performed to find the relation between the impairment caused by the GOP size (I_{GOP}).

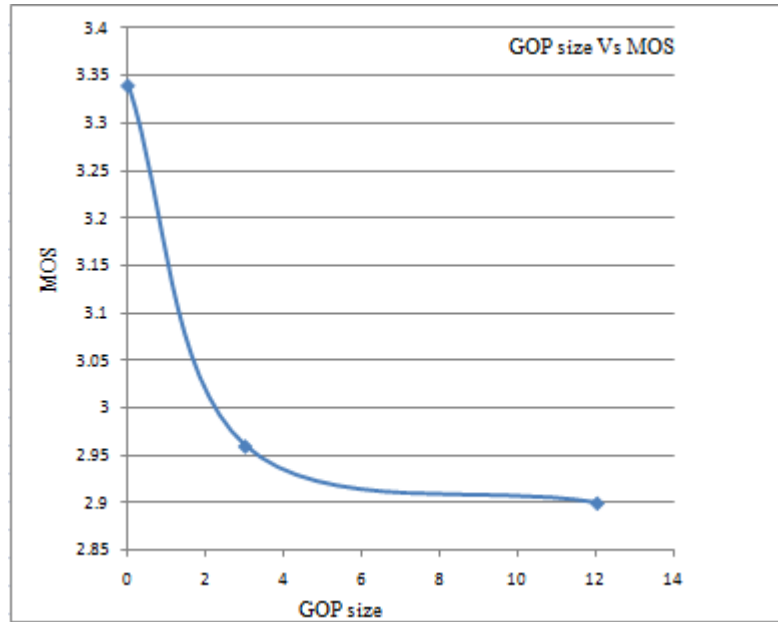


Figure 4.8 Influence of GOP size on MOS

$$I_{\text{GOP}} = \begin{cases} 9 * \frac{\text{GOP} - \text{GOP}_1}{\text{GOP}_2 - \text{GOP}_1} + 33 & \text{If } (\text{GOP}_1 < \text{GOP} \leq \text{GOP}_2) \\ 42 & \text{If } (\text{GOP} > \text{GOP}_2) \end{cases} \quad (4.12)$$

Where,

$$\text{GOP}_1 = 0$$

$$\text{GOP}_2 = 3$$

Similarly from the sessions 4, 5 and 6, the influence of resolution was studied and plotted as shown in figure 4.8. In the figure, x-axis is in kilo pixels (width*height/1000). As the resolution increased, the user experience of user improved. A similar linear regression

analysis was performed to find the relation between the impairment caused by the resolution (I_{RES}).

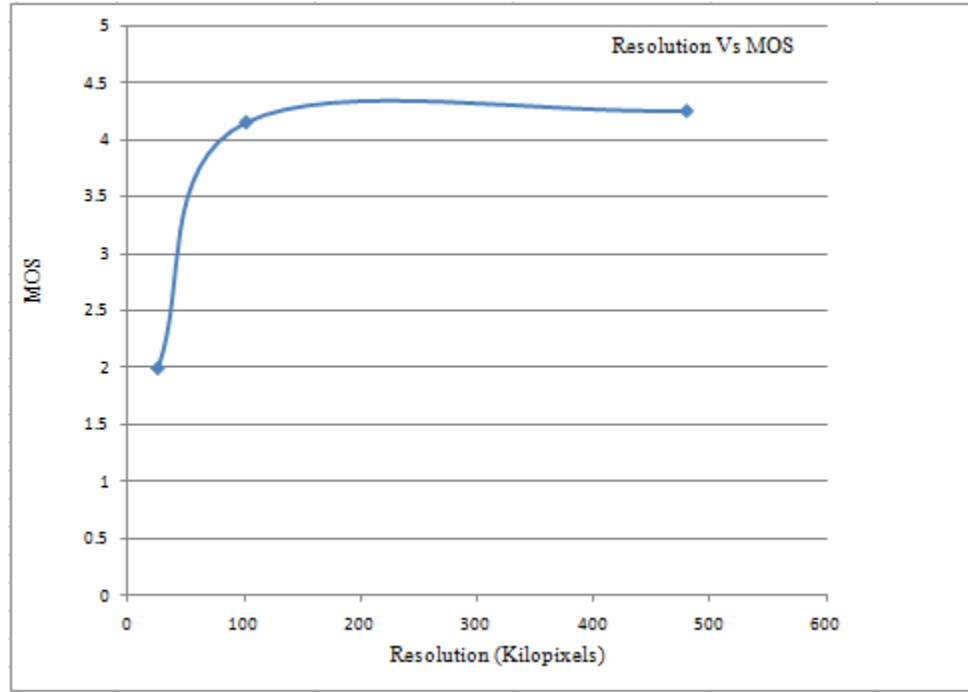


Figure 4.9 Influence of resolution on MOS

$$I_{RES} = \begin{cases} 46 * \frac{RES - RES_2}{RES_1 - RES_2} + 14 & \text{If } (RES_1 < RES \leq RES_2) \\ 10 & \text{If } (RES > RES_2) \end{cases} \quad (4.13)$$

Where,

$$RES_1 = 176 * 144 / 1000$$

$$RES_2 = 352 * 288 / 1000$$

4.4 INFLUENCE OF GAME PARAMETERS ON MOS

From sessions 2, 13 and 14, influence of enemy speed on MOS was studied. As shown in table 4.3, as the enemy speed increases to three times the initial speed, the player cannot follow the enemy motion, essentially due to the delay in the network, and eventually the game becomes unplayable. The relationship between MOS and the enemy speed is as shown in the figure 4.9.

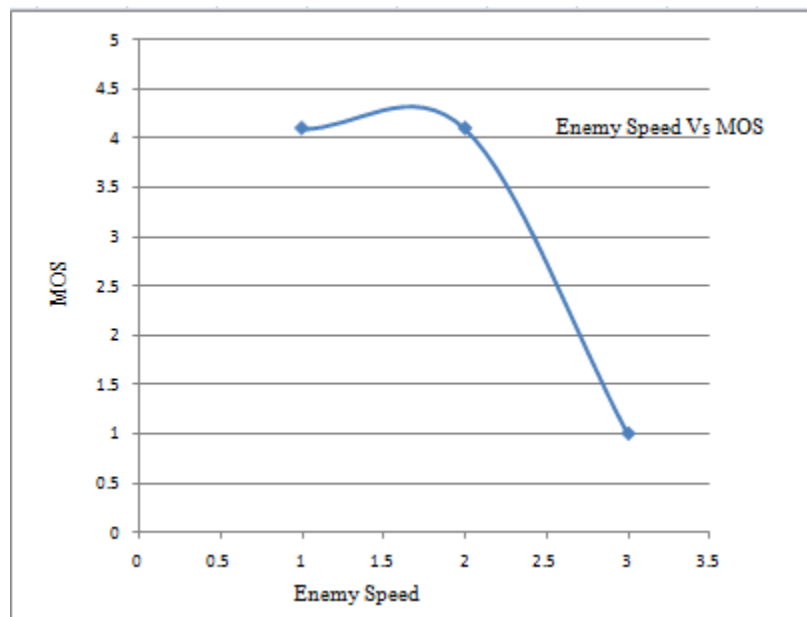


Figure 4.10 Influence of enemy speed on MOS

As discussed in the section 4.3, the impairment due to enemy speed (I_{ES}) was calculated using linear regression analysis. The analytical formulation for I_{ES} and I_{ESS} are as below:

$$I_{ES} = \begin{cases} 16 & \text{If } (ES_1 < ES \leq ES_2) \\ 84 * \frac{ES - ES_2}{ES_3 - ES_2} + 16 & \text{If } (ES_2 < ES \leq ES_3) \end{cases} \quad (4.14)$$

The value of the enemy speeds ES_1 is 1, ES_2 is 2 and ES_3 is 3.

From sessions 2, 15 and 16, influence of enemy shooting speed on MOS was studied. The relationship between MOS and the enemy shooting speed is as shown in the figure 4.10.

The MOS was not affected highly by the enemy shooting speed mainly because the user enjoyed the game as the enemy shooting speed increased. Hence, impairment due to the enemy motion speed can be neglected in equation 4.3.

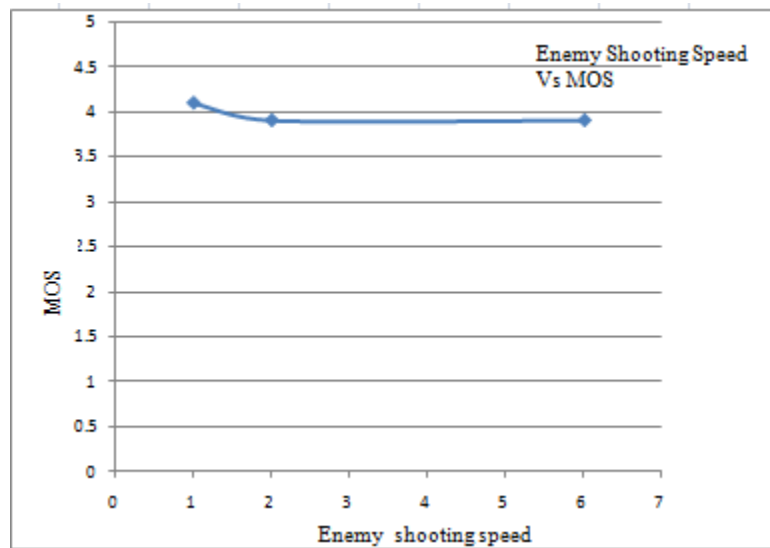


Figure 4.11 Influence of enemy shooting speed on MOS

4.5 GAMING USER EXPERIENCE MODEL VALIDATION

The model developed in section 4.3 and 4.4 was verified using another set of controlled experiments. We use the same experimental setup described earlier in section 4.2. Sixteen game sessions were conducted and all the factors were simultaneously varied. However, sessions were not repeated for multiple users. Also, intermediate parameters that fall inside the operable range were also used to verify the model. For a given encoder and game setting, the game user experience was predicted using equations 4.9 (average method) and 4.10 (proportion method). While equation 4.9 gives the average values of the imparities, equation 4.10 gives a more accurate gaming user experience. Also, the predicted gaming user experience values are rounded to integers as we are interested only in classifying user experience into five categories as mentioned in table 4.1.

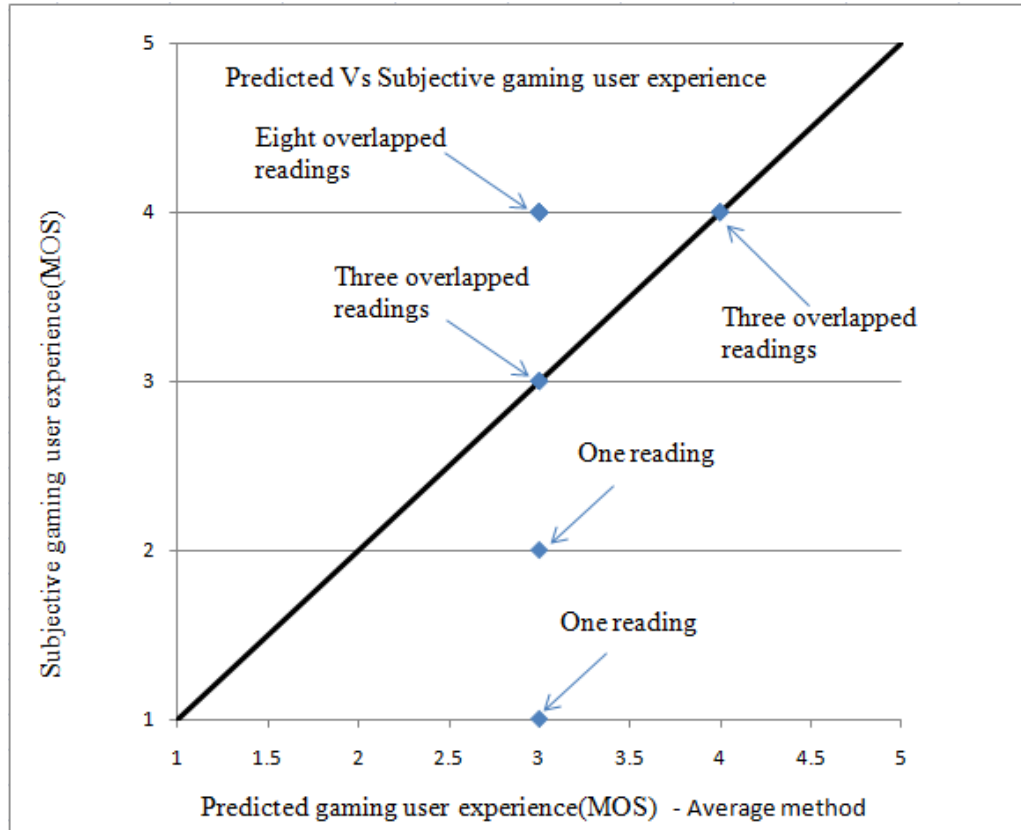


Figure 4.12 Predicted Vs Subjective gaming user experience using average method

The figure 4.11 shows the relationship between the gaming user experience predicted by average method (x-axis) and the subjective gaming user experience. The correlation between the predicted and subjective gaming user experience was calculated to be 0.2774. Out of the sixteen experimental game sessions, user experience of thirteen sessions was predicted as 3, which is the mid-value of the mean opinion score scale. Hence, using an averaging technique does not really describe the actual gaming user experience.

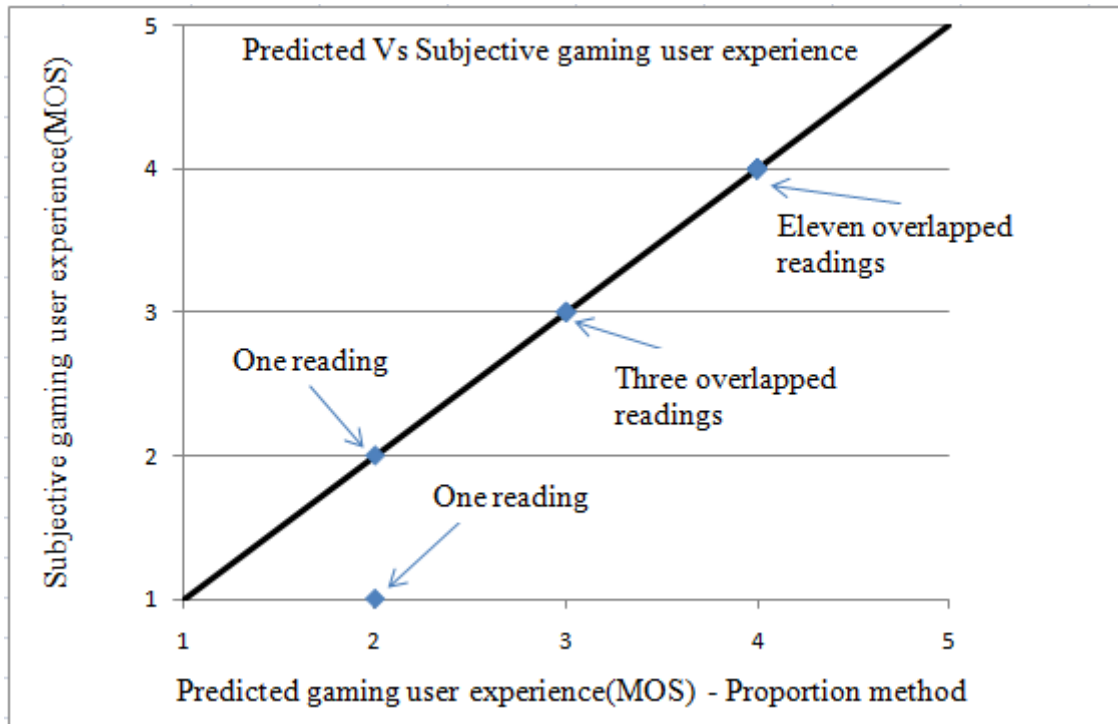


Figure 4.13 Predicted Vs Subjective gaming user experience using proportion method

The figure 4.12 shows the relationship between the gaming user experiences predicted by the proportion method (x-axis) and the subjective gaming user experience (y-axis). The correlation between the predicted and subjective gaming user experience was calculated to be 0.9734.

CHAPTER 5

CONCLUSION

In this thesis, we presented the design and implementation of a remote gaming system which is built over existing video encoding and streaming solutions. Modules like live game, video encoders and video streamer were studied, modified and integrated to develop the remote gaming system. We have successfully built a remote gaming system that is capable of logging tons of useful information related to the server settings during a game session.

The factors affecting the gaming user experience were studied and an analytical model to describe the gaming user experience was developed. The model is capable of accurately describing mean opinion score of a gaming session at a particular game and encoder setting. Also, this model can be used in a form of reverse engineering that adjusts the remote gaming system parameters automatically to optimize the gaming user experience.

The validation of the model was done using proportion method and average method. The results prove that the correlation between the predicted and the subjective gaming user experience was higher for the proportion method. This proves that different impairments have different effect on the gaming user experience.

The user is not very concerned about the delays existing in the system. As the user progresses in the game play, the user starts playing ahead of time. For example, a user fire at the enemy even before it is aligned to player's firing level. However, if the rendered video quality at the client is reduced, the user experience is low. Thus quality of video decoded is more critical than the delay as far as gaming user experience is considered.

We can also conclude that the game parameters play an important role in characterizing the gaming user experience. The game parameters can be used to hide the delays in the network.

CHAPTER 6

FUTURE WORK

There are many tasks that have to be implemented in the future modifications of the system. A live adaptation module which modifies the parameters of the system to achieve a consistent and high level of user experience is desired. Since the proposed remote gaming setup is highly extensible, addition of live adaptation module would be an easy task. Live adaptation module would be intelligent enough to learn the network characteristics and tune the system parameters in cases of congestion or losses. Live adaptation module can learn about the network from a ton of performance statistics for every encoded frame. Also, it can quantify the gaming user experience using the proposed model. Thus a consistent gaming user experience can be delivered.

The gaming user experience model for a multi-user gaming system has to be studied. The interactions between multiple users will be more complicated than a single user. Also, the live streaming module has to handle more than one user, which could be an overhead considering the bandwidth availability. Synchronization between the users is also critical to have maximum gaming user experience.

Also, the audio-video synchronization for the remote gaming system has to be explored. Introducing audio into the existing game would be a challenge considering the fact that the delays associated with audio and video are different. The player could get easily distracted and irritated if the audio plays at a different instance. Also, the decoder in the client mobile device will have to handle responsibility of decoding the audio, which could potentially overload the mobile device.

BIBLIOGRAPHY

- [1] http://www.comscore.com/Press_Events/Press_Releases/2007/07/Worldwide_Online_Gaming_Grows (last accessed on 11/27/2010).
- [2] Cisco Visual Networking Index: Forecast and Methodology, 2009-2014 : http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html
- [3] Potter, S. Baratto, R. Laadan, O. Nieh, J. R.: 'GamePod: Persistent Gaming Sessions on Pocketable Storage Devices'. Mobile Ubiquitous Computing, Systems, Services and Technologies, 2009. UBICOMM '09. Third International Conference , Sliema, pp.269-276
- [4] <http://www.onlive.com> (last accessed on 11/27/2010).
- [5] Jarvinen, S. Laulajainen, J.-P. Sutinen, T. Sallinen, S. 'QoS-Aware real-time video encoding How to Improve the User Experience of a Gaming-on-Demand Service'. Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, pp. 994 - 997
- [6] D. De Winter, P. Simoens, L. Deboosere, F. De Turck, J. Moreau, B. Dhoedt, P. Demeester: 'A hybrid thin-client protocol for multimedia streaming and interactive gaming applications'. 16th Annual International Workshop on Network and Operating Systems Support for Digital Audio and Video ,2006
- [7] Shaoxuan Wang Dey, S.: 'Addressing Response Time and Video Quality in Remote Server Based Internet Mobile Gaming'. Wireless Communications and Networking Conference (WCNC), 2010 IEEE
- [8] Shaoxuan Wang Dey, S.: 'Modeling and Characterizing User Experience in aCloud Server Based Mobile Gaming Approach,' in Proc. IEEE GLOBECOM 2009, Honolulu, USA, Nov. 2009.
- [9] D. S. Hands, 'A Basic Multimedia Quality Model' IEEE Transactions on Multimedia, vol. 6, no. 6, pp. 806-816, Dec. 2004.

- [10] S. Tasaka and Y. Watanabe, 'Real-Time Estimation of User-Level QoS in Audio-Video IP Transmission by Using Temporal and Spatial Quality' IEEE GLOBECOM, Nov. 2007.
- [11] K. Yamagishi and T. Hayashi, 'Opinion Model for Estimating Video Quality of Videophone Services' IEEE GLOBECOM, Nov. 2006.
- [12] D. Wu et al., 'Streaming Video over the Internet: Approaches and Directions' IEEE Trans. Circuits and Systems for Video Technology, Mar. 2001, pp. 282-300.
- [13] S. Winkler, F. Dufaux, 'Video Quality Evaluation for Mobile Applications'. Proc. of SPIE Conference on Visual Communications and Image Processing, Lugano, Switzerland, Jul. 2003.
- [14] M. Ries, O. Nemethova, M. Rupp, 'Video Quality Estimation for Mobile H.264/AVC Video Streaming'. Journal of Communications, Vol. 3, pp. 41 - 50, 2008.
- [15] <http://www.its.bldrdoc.gov/vqm/> (last accessed on 11/27/2010).
- [16] M. Claypool and K. Claypool, 'Latency and Player Actions in Online Games'. Communications of the ACM, 49(11), 2006.
- [17] Dick, M., O. Wellnitz, and L. Wolf, 'Analysis of factors affecting players' performance and perception in multiplayer games'. ACM SIGCOMM workshop on Network and System Support for Games, (NetGames), 2005.
- [18] W.-C. Feng, F. Chang, W.-C.Feng, and J. Walpole, 'A traffic characterization of popular on-line games'. IEEE/ACM Transactions on Networking, vol 13, no.3, pp.588-500, Jun. 2005.
- [19] T. Yasui, et al. 'Influences of network latency and packet loss on consistency in networked racing games' ACM SIGCOMM workshop on Network and System Support for Games, (NetGames), 2005.
- [20] ITU-T Recommendation BT.500-11: 'Methodology for the subjective assessment of the quality of television pictures', 2002
- [21] ITU-T Recommendation P.910: 'Subjective video quality assessment methods for multimedia applications', 1999
- [22] Wade, N., and Swanston, M.: 'Visual perception' (Psychology Press, London, 2001, 2nd edn.)

- [23] Zhong, Y., Richardson, I., Sahraie, A., and McGeorge, P.: ‘Qualitative and quantitative assessment in video compression’. 12th European Conf. on Eye Movements, Dundee, Scotland, August 2003
- [24] van den Branden Lambrecht, C.J., and Verscheure, O.: ‘Perceptual quality measure using a spatio-temporal model of the human visual system’. Proc. SPIE-Int. Soc. Opt. Eng., San Jose, CA, USA, 1996, Vol. 2668, pp. 450–461
- [25] Wu, H., Yu, Z., Winkler, S., and Chen, T.: ‘Impairment metrics for MC=DPCM=DCT encoded digital video’. Proc. Picture Coding Symp., Seoul, Korea, 2001
- [26] Tan, K., and Ghanbari, M.: ‘A multi-metric objective picture quality measurement model for MPEG video’, IEEE Trans. Circuits Syst. Video Technol., 2000, 10, (7), pp. 1208–1213
- [27] Winkler, S.: ‘Video quality measurement standards – current status and trends’, Information, Communications and Signal Processing, 2009. ICICS 2009. 7th International Conference., 10, (7), pp. 1208–1213
- [28] MPlayer Organization, “The online documentation of MPlayer,” <http://www.mplayerhq.hu/DOCS/HTML-single/en/MPlayer.html> (last accessed on 11/27/2010).
- [29] FFMPEG General Encoder Implementation, “The online documentation of FFMPEG”. <http://www.ffmpeg.org/general.html> (last accessed on 11/27/2010).
- [30] Intel Media SDK, <http://software.intel.com/en-us/articles/intel-media-sdk/> (last accessed on 11/27/2010).
- [31] Integrated Performance Primitives (IPP) Documentation, “The Online documentation for Intel Performance Primitives (IPP)”. http://software.intel.com/sites/products/documentation/hpc/ipp/pdf/userguide_win_i a32.pdf (last accessed on 11/27/2010).
- [32] Transmission Control Protocol RFC, <http://www.faqs.org/rfcs/rfc793.html> (last accessed on 11/27/2010).
- [33] User Datagram protocol RFC, <http://www.faqs.org/rfcs/rfc768.html> (last accessed on 11/27/2010).
- [34] Real Time Streaming Protocol RFC, <http://www.ietf.org/rfc/rfc2326.txt> (last accessed on 11/27/2010).
- [35] Live555 video streamer FAQ, <http://www.live555.com/liveMedia/faq.html> (last accessed on 11/27/2010).

- [36] VideoLAN Documentation, <http://wiki.videolan.org/Documentation:Documentation> (last accessed on 11/27/2010).
- [37] ITU-T Rec. G.107, “The E-model, a computational model for use in transmission planning,” Mar. 2005.
- [38] Wireshark Documentation, “The online documentation for Wireshark”, <http://www.wireshark.org/docs/> (last accessed on 11/27/2010).