Master of Science in Electrical Engineering
June 2018

Performance Metrics Analysis of GamingAnywhere with GPU accelerated NVIDIA CUDA

Byreddy Sreenibha Reddy
And
Mohammed Zaahid

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Master's in Electrical Engineering with Emphasis on Telecommunication Systems. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author(s):
Sreenibha Reddy Byreddy
E-mail: srby16@student.bth.se

Zaahid Mohammed
E-mail: zamo16@student.bth.se

University advisor:
Dr. Siamak Khatibi
Department of communication systems
E-mail: Siamak.khatibi@bth.se

Dr. Yong Yao
Department of communication systems
E-mail: yong.yao@bth.se

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

The modern world has opened the gates to a lot of advancements in cloud computing, particularly in the field of Cloud Gaming. The most recent development made in this area is the open-source cloud gaming system called GamingAnywhere.

The relationship between the CPU and GPU is what is the main object of our concentration in this thesis paper. The Graphical Processing Unit (GPU) performance plays a vital role in analyzing the playing experience and enhancement of GamingAnywhere. In this paper, the virtualization of the GPU has been concentrated on and is suggested that the acceleration of this unit using NVIDIA CUDA, is the key for better performance while using GamingAnywhere. After vast research, the technique employed for NVIDIA CUDA has been chosen as gVirtuS.

There is an experimental study conducted to evaluate the feasibility and performance of GPU solutions by VMware in cloud gaming scenarios given by GamingAnywhere. Performance is measured in terms of bitrate, packet loss, jitter and frame rate. Different resolutions of the game are considered in our empirical research and our results show that the frame rate and bitrate have increased with different resolutions, and the usage of NVIDIA CUDA enhanced GPU.

Keywords: Cloud Computing, Cloud Gaming, GPU Acceleration, NVIDIA CUDA, GamingAnywhere.

ACKNOWLEDGEMENT

Firstly, we would like to express our heartful gratitude to our Supervisor's Dr Siamak Khatili and Dr Yong Yao for their constant support, encouragement. I had learnt a lot from the meetings with him. They were very good and patient, helped us in the successful completion of our thesis work

We would like to thank Dr Siamak Khatibi, Dr Kurt Tutschku, Dr Markus Fiedler and Patrik Arlos at Blekinge Tekniska Hogskola for their suggestions in the thesis and ATS course.

Finally, we would like to thank god and out parents for their blessings, unconditional love and encouragement throughout our study and work.

B Sreenibha Reddy.
And
Zaahid Mohammed

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

VM	Virtual Machine
OS	Operating System
CPU	Central Processing Unit
GPU	Graphics Processing Unit
GPGPU	General Purpose Graphics Processing Unit
CUDA	Compute Unified Device Architecture
TLP	Thread-Level Parallelism
PD	Processing Delay
OD	Playout Delay
ND	Network Delay
RTT	Round-Trip Time
HTPC	Home Theater PC
HD video	High Definition Video
MVC	Model View Controller
PSNR	Peak Signal to Noise Ratio
API	Application Program Interface
RTP	Real-Time Protocol
RTCP	RTP Control Protocol
RTSP	Real Time Streaming Protocol
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
SDL	Specification and Description Language
LAN	Local Area Network
WLAN	Wireless Local Area Network
NVCC	Nvidia CUDA Compiler

1 INTRODUCTION

Virtualization is a technology which combines or divides computing resources to present one or many operating environments, using methodologies like hardware and software partitioning or aggregation, partial or complete machine simulation, emulation, time-sharing, and others". A virtualization layer is an essential component in virtualization as it provides the capability of using hardware resources to create multiple virtual machines with isolation and performance guarantees. Sometimes, such a virtualization layer is also called hypervisor or Virtual Machine Monitor (VMM). A virtual machine is defined as an emulation of a computer that provides environments for supporting the operating system (OS).

Today, most computer resources, particularly Central Processing Unit (CPU), have been able to be well-virtualized with performance guarantees by using software-based virtualization techniques such as full virtualization and para-virtualization. [17]

Virtualization layer is an essential component in virtualization as it provides the capacity of using hardware resource to create multiple virtual machines with isolation and performance guarantees. virtualization is also called as Hypervisor or Virtual Machine.

Virtual machine is a software implementation of a physical machine - computer - that works and executes analogically to it. Virtual machines are divided in two categories based on their use and correspondence to real machine: system virtual machines and process virtual machines. First category provides a complete system platform that executes complete operating system, second one will run a single program. [12]

The main advantages of virtual machines:

- Multiple OS environments can exist simultaneously on the same machine, isolated from each other;
- Virtual machine can offer an instruction set architecture that differs from real computer's;
- Easy maintenance, application provisioning, availability and convenient recovery. [12]

The main disadvantages:

- When multiple virtual machines are simultaneously running on a host computer, each virtual machine may introduce an unstable performance, which depends on the workload on the system by other running virtual machines;
- Virtual machine is not that efficient as a real one when accessing the hardware. [10]
- Virtualization brings you many advantages – centralizing network management, reducing dependency on additional hardware and software, etc. But as it is always the case, it has certain shortcomings too.

Today, cloud gaming is showing tremendous potential for game developers. In recent years market research studies break current game market growth into three categories: boxed games, games sold online and cloud games. In online games, a player receives updates about other players in the game from the server, a player can send his/her regular game updates to the server. Instead of the server the game logic is processed at the client. “Cloud Gaming renders an interactive gaming application remotely in a cloud and streams scenes as a video sequence back to a thin client over the internet”. [14]

Recently, both industry and the research community have been actively exploring the solutions to resolving this problem through cloud computing techniques. Through utilization of elastic hardware resources and widely deployed data centers, cloud computing has brought new business models for the IT industry. Specifically, it turns the idea of cloud gaming into a reality.

A player burst logs into the system via a portal server, which covers a list of available cloud games. Then the player selects a game and makes a request to play the game. Upon the receipt of a playing request, the portal server executes the selected game on an available gaming server and returns the game server's IP address to the player.

Finally, the player connects to the gaming server and starts to play the selected game. With the help of virtualization, the portal server and gaming server may be able to be deployed on virtual machines, thus significantly improving the utilization of hardware resources. For instance, a high-end server can run hundreds of games concurrently with performance guarantee through virtualization.

Today, cloud gaming is showing tremendous market potential for game developers Virtual Machine is a separate individual operating system installation on your usual operating system. It is implemented by software emulation and hardware virtualization.

GamingAnywhere is an open source cloud gaming platform. It is designed for high extensibility, portability and configurability. It supports various OS like Windows, Linux (Ubuntu, mint, etc), OSX and Android. GamingAnywhere recently became a promising solution approach for the R&D on mobile cloud gaming from both academia and industrial perspectives. General-Purpose computing on Graphics Processing Units (GPGPU) has recently emerged as a powerful computing paradigm because of the massive parallelism provided by several hundreds of processing cores. Under the GPGPU concept, NVIDIA has developed a C-based programming model, Compute Unified Device Architecture (CUDA), which provides greater programmability for high-performance graphics devices. [11]

Although the GPGPU paradigm successfully provides significant computation throughput, its performance could still be improved if we could utilize the idle CPU resource. Indeed, in general, the host CPU is being held while the CUDA kernel executes on the GPU devices; the CPU is not allowed to resume execution until the GPU has completed the kernel code and has provided the computation results. The CUDA technology introduced by NVIDIA in 2006 has become the most effective solution for general-purpose parallel computing on GPUs. It is designed to leverage processing cores (or Streaming Multiprocessors, SMs) in NVIDIA GPUs to execute data-parallel functions, called kernels. To this end, CUDA provides a programming model to deliver an efficient way to express the kernels with a minimal language extension.

The CUDA programming model provides a straightforward means of describing inherently parallel computations, and NVIDIA’s GPU architecture delivers high computational throughput on massively parallel problems. There are various experiences gained in applying CUDA to a diverse set of problems and the parallel speedups over sequential codes running on traditional CPU architectures attained by executing key computations on the GPU. The CUDA programming model consists of three abstractions, including a thread hierarchy, a memory hierarchy, and barrier synchronization. A CUDA thread is the smallest unit of the GPU processing, and therefore can provide fine-grained data parallelism and thread-level parallelism (TLP). [2]

1.1 Motivation

From the past five years, mobile gaming has been advanced a lot. The computational load of a mobile device when playing a game, should be offloaded to server to save energy and time on terminal side. As a result, the cloud gaming came into existence. As many cloud gaming systems are proprietary and closed, GamingAnywhere is the only open-source cloud gaming system available for gamer, developer or service provider.

When the users are playing cloud games, the computing, communication and display on mobile devices will consume more energy which can lead to the draining of battery quickly. This avoids the users to make phone calls even. Hence, measuring the power consumption of mobile cloud gaming using GamingAnywhere is quite challenging.

These measurements can be done by using Monsoon Power monitoring tool by which we can get the consumed energy, average power, average current and average voltage of all connected devices on PC or laptop [16][14]. This thesis is motivated by the question: What is the impact on energy consumption of mobile clients when using GamingAnywhere??

1.2 Aims

The main aim of the thesis is to study different performance metric such as image quality (decibels), frame rate (fps) and Response Delay (ms) for mobile cloud gaming using Gaming Anywhere while using a gaming application on a device, after GPU acceleration has been implemented. The GPU acceleration is going to be performed using NVIDIA CUDA, and the main rendering technique used here will be gVirtuS.

The task to perform this virtualization and subsequent acceleration is also one of the main tasks in this research and experimentation work. To analyze the metrics mentioned above we will be using GPU performance in each tested configuration, several graphics card benchmarks, including two synthetic tests and a real-world gaming test, are used in the experiments. Those benchmarks are PassMark Performance Test. To complement these benchmark tests, by monitoring the hardware resource utilization of each graphics card benchmark, we will be using performance monitoring tools such as MSI Afterburner to help us analyze the main bottleneck during our experiments. The gaming application in consideration here will be AssaultCube, a cloud gaming application compatible with GamingAnywhere and mobile devices.

1.3 Objectives

1. To study and understand the system of GamingAnywhere, and to install it for usage of gaming applications in mobile devices.
2. To implement GPU acceleration in the GamingAnywhere using NVIDIA CUDA, and virtualization using gVirtuS.
3. To identify and analyze the advantages of using a mobile device to which GamingAnywhere server is streaming to and infer whether it is a good option for the use of GA.
4. Research thoroughly on Graphics Card benchmarks (PassMark Performance Test and Doom and Performance Monitoring Tools (MSI Afterburner) and understand their workings. This is important as these are the tools that will help us analyze the workings of GamingAnywhere and how it affects the CPU and GPU. This is in turn done by playing the computer game Assault Cube which helps to fulfill the benchmark tests and evaluate the performance meters.

1.4 Research Questions

The following questions can be addressed by Mohammad Zaahid:

- What is the possibility of using GPU Acceleration in GamingAnywhere in Windows or Unix platform?
- Can we use gVirtuS virtualization technique in implementing NVIDIA CUDA GPU acceleration in GamingAnywhere?

The following questions can be addressed by Byreddy Sreenibha Reddy:

- How are the CPU and GPU of any device affected when running GamingAnywhere, in matter of performance meters, such as image quality and frame rate per second?

1.5 Experimentation

1. The first stage of the research is the study of GamingAnywhere and its workings.
2. In the next stage, installation, configuration and implementation of GamingAnywhere testbed is studied and done where GA server and GA client are installed in our PC and mobile respectively.
3. Implementation of GPU acceleration in GamingAnywhere with NVIDIA CUDA.
4. The tools used for assessing the performance metrics (MSI Afterburner, PassMark Performance Test) are studied and analysis of how to configure them is done.
5. Installing a gaming application, AssaultCube with GamingAnywhere on the mobile device.
6. Results are noted, observed and later analyzed. Based on results, conclusions and thoughts are given.

1.6 Thesis Outline

Chapter 1: This chapter starts off by explaining the concepts of Virtualization, followed by Cloud Gaming in the modern world. The unique selling point of the thesis, which is GamingAnywhere, an open source platform is mentioned in detail. The foundations of the project which the hypothesis is, experimentation strategy, along with research questions and hypothesis are mentioned. The tools that are used to perform the experimentation are also listed in the first chapter.

Chapter 2: Background describes about the need for integrating Cloud Computing and Video Gaming applications and its origins. Cloud Gaming is introspected in this section. The options for it, other than GamingAnywhere are deeply explained. GamingAnywhere is diagnosed and high amount of literature is provided.

Chapter 3: Related Work, this chapter talks continues to infer on the options of Cloud Gaming, with previous examples. The new concept of NVIDIA CUDA is talked about and a Segway is given into the deeper understanding of GamingAnywhere.

Chapter 4: Methodology, this chapter gives the reader a deep understanding of the experimentation, the test bed involved, devices used, software's and applications employed to complete the simulation and obtain appropriate results.

Chapter 5: Results and Analysis, this chapter solely consists of the results obtained after conducting deep experiments and examining the applications used for quantifiable values so that a conclusion might be derived after analysis. It also explains the results, the why, the what they mean part of the results.

Chapter 6: Conclusion and Future work section concludes the thesis. It out-lines the results and gives suggestions for the future work that can be done.

1.7 Split of work

This section illustrates the distribution of work among the thesis partners.

SECTION	TOPIC	CONTRIBUTOR
Chapter 1	Introduction	Mohammed Zaahid
	1.1 Motivation	Mohammed Zaahid
	1.2 Aim	Mohammed Zaahid
	1.3 Objectives	Mohammed Zaahid
	1.3 Research Questions	Mohammed Zaahid
	1.4 Methodology	Mohammed Zaahid
	1.5 Thesis Outline	Mohammed Zaahid
	1.6 Time and Activity plan	Byreddy Sreenibha Reddy
Chapter 2	Background	Mohammed Zaahid
	2.1 Virtualization	Mohammed Zaahid
	2.2 Cloud Computing	Mohammed Zaahid
	2.3 Cloud Gaming	Mohammed Zaahid
	2.4Issues and Challenges in Cloud Gaming	Mohammed Zaahid
Chapter 3	Related Work	Mohammed Zaahid
	2.5 Options in Cloud Gaming	Mohammed Zaahid
	2.5.1 Onlive	Mohammed Zaahid
	2.5.2 Gaikai	Mohammed Zaahid
	2.5.3 Stream MyGame	Mohammed Zaahid
	2.6 NVIDIA CUDA	Byreddy Sreenibha Reddy
	2.7 Virtualization Techniques	Mohammed Zaahid
	2.7.1 g-Virtus	Mohammed Zaahid
	2.8 Architecture of GamingAnywhere	Mohammed Zaahid
	2.8.1 GA Server	Mohammed Zaahid
	2.8.2 GA Client	Mohammed Zaahid
Chapter 4	Related Work	Mohammed Zaahid
	4.1 Research Technique	Byreddy Sreenibha Reddy
	4.2 Evaluation Technique	Byreddy Sreenibha Reddy
	4.3 Experimental Testbed	Byreddy Sreenibha Reddy
	4.3.1 Testbed Design	Byreddy Sreenibha Reddy
	4.3.2 GamingAnywhere	Byreddy Sreenibha Reddy
	4.3.2.1 GamingAnywhere Server	Byreddy Sreenibha Reddy
	4.3.2.2 GamingAnywhere Client	Byreddy Sreenibha Reddy
	4.3.3 AssaultCube	Byreddy Sreenibha Reddy
	4.3.4 Experimental Tools	Byreddy Sreenibha Reddy
	4.3.4.1 Graphics Card Benchmark	Byreddy Sreenibha Reddy
	4.3.4.1.1 PassMark Performance Test	Byreddy Sreenibha Reddy
	4.3.4.1.2 Doom 3	Byreddy Sreenibha Reddy
	4.3.4.2 Performance Monitoring Tool	Byreddy Sreenibha Reddy
	4.3.4.2.1 MSI Afterburner	Byreddy Sreenibha Reddy
Chapter 5	Experimentation and Results	Mohammed Zaahid

		Byreddy Sreenibha Reddy
	Experimental Procedure	**Byreddy Sreenibha Reddy**
	Experiment Metrics	**Byreddy Sreenibha Reddy**
	Experimental Scenarios	**Byreddy Sreenibha Reddy**
Chapter 6	**Analysis and Discussion**	**Mohammed Zaahid**
	PassMark Performance Test calculations and Graphs	**Mohammed Zaahid** **Byreddy Sreenibha Reddy**
	MSI Afterburner calculations and Graphs	**Mohammed Zaahid** **Byreddy Sreenibha Reddy**
Chapter 7	**Conclusion and Future Work**	**Mohammed zaahid** **Byreddy Sreenibha Reddy**
	Conclusion	**Mohammed Zaahid** **Byreddy Sreenibha Reddy**
	Answer to the research questions	**Mohammed Zaahid** **Byreddy Sreenibha Reddy**
	Future Work	**Mohammed Zaahid** **Byreddy Sreenibha Reddy**
Chapter 8	**References**	**Mohammed Zaahid**

2 BACKGROUND

2.1 Virtualization

Virtualization typically refers to the creation of virtual machine that can virtualize all the hardware resources, including processors, memory, storage, and network connectivity. With the virtualization, physical hardware resources can be shared by one or more virtual machines. According to the requirements from Popek and Goldberg, there are three aspects to satisfy the virtualization. To start with, the virtualization ought to give an equal domain to run a program contrasted with a local framework. If the program shows a different behavior under the virtualization, it may not be eligible as a virtualized environment. Having a full control of assets is critical to ensure information and assets on each virtual condition from any dangers or execution obstruction in sharing physical assets. Virtualization regularly anticipates that execution debasement due to the extra errands for virtualization, however great execution ought to be accomplished with a product or equipment bolster in taking care of special directions. With these necessities, proficient virtualization is ensured. In the accompanying area, diverse sorts of hypervisors are clarified with the execution level of virtualization. [17]

2.2 Cloud Computing

Mobile Cloud Computing research aims to bring rich computing applications to energy constrained hand-held devices, such as mobile phones and tablets. While most of the hand-held devices are quite capable nowadays, the battery technology has not improved at the pace of processor and memory technology. According to some estimations doubling the clock speed of the processor consumes eight times more energy, which makes running computationally intensive applications, such as image or voice recognition, very challenging at the mobile end. As the processor is a large constraint for mobile devices, approaches to offload computation from the devices to servers (with fixed power supply) have emerged.

A classic and a very coarse-grained approach to offload computation is to use the mobile as a “thin client”, i.e., use the mobile client as a dumb display and control device, and handle all the remaining computation at the server side. More recent approaches allow offloading on-demand and using more fine-grained granularity (e.g. process, method or class). Such approaches are referred as mobile computation offloading, cyber foraging or surrogate computing in the literature. [16] All the mobile computation offloading systems either aim to save energy of the mobile device or make it possible to accomplish tasks that are not normally possible solely using the mobile device. The offloaded tasks should be carefully chosen because moving the computation unit and related data over wireless networks consumes also energy.

Mobile Cloud Computing inquire about plans to convey rich figuring applications to vitality compelled hand-held gadgets, for example, cell phones and tablets. While a large portion of the hand-held gadgets are very fit these days, the battery innovation has not enhanced at the pace of processor and memory innovation. As per a few estimations multiplying the clock speed of the processor devours eight times more vitality, which makes running computationally serious applications, for example, picture or voice acknowledgement, extremely difficult at the versatile end. As the processor is a huge limitation for cell phones, ways to deal with offload calculation from the gadgets to servers (with settled power supply) have risen. [16] A work of art and an extremely coarse-grained way to deal with offload calculation is to utilize the portable as a "thin customer", i.e., utilize the versatile customer as a stupid show and control gadget, and handle all the rest of the calculation at the server side. Later methodologies permit offloading on-request and utilizing more fine-grained granularity (e.g. process, technique or class). Such methodologies are

alluded as portable calculation offloading, digital scavenging or surrogate processing in the writing. All the versatile calculation offloading frameworks either plan to spare vitality of the cell phone or make it conceivable to achieve errands that are not ordinarily conceivable exclusively utilizing the cell phone.

2.3 Cloud Gaming

Recent advances in cloud technology have turned the idea of Cloud Gaming into a reality. Cloud Gaming, in its simplest form, renders an interactive gaming application remotely in the cloud and streams the scenes as a video sequence back to the player over the Internet. This is an advantage for less powerful computational devices that are otherwise incapable of running high quality games. Such industrial pioneers as OnLive and GaiKai have seen success in the market with large user bases. Through the utilization of elastic resources and widely deployed data-centers, cloud computing has provided countless new opportunities for both new and existing applications. Existing applications, from file sharing and document synchronization to media streaming, have experienced a great leap forward in terms of system efficiency and usability through leveraging cloud computing platforms. Much of these advances have come from exploring the cloud's massive resources with computational offloading and reducing user access latencies with strategically placed cloud data-centers. Recently, advances in cloud technology have expanded to allow offloading not only of traditional computations but also of such more complex tasks as high definition 3D rendering, which turns the idea of Cloud Gaming into a reality.

Cloud gaming, in its simplest form, renders an interactive gaming application remotely in the cloud and streams the scenes as a video sequence back to the player over the Internet. A cloud gaming player interacts with the application through a thin client, which is responsible for displaying the video from the cloud rendering server as well as collecting the player's commands and sending the interactions back to the cloud. Figure 1 shows a high level architectural view of such a cloud gaming system with thin clients and cloud-based rendering.

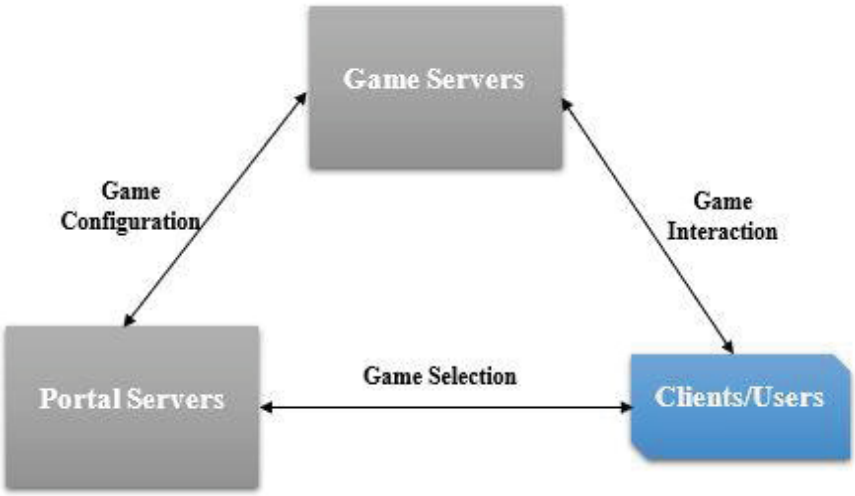

Figure 2.1: Cloud Gaming

2.4 Issues and Challenges in Cloud Gaming

Although cloud gaming shows great advantages for both game developers and players, it is still in the early stage as some significant challenges remain regarding the widespread deployment. As low-latency live video streaming and high-performance 3D rendering are

key factors to ensure the success of cloud gaming, two performance characteristics, low response delay and high video quality must be ensured in cloud gaming.

While running cloud games, a cloud gaming system must collect commands from players, process them, then encode, compress and stream results (game scenes) back to players. To ensure the interactivity, these operations must be finished within hundreds of milliseconds. The latency caused by these operations are called response delay, which can be separated into three components:

Processing Delay (PD): the time required for the game server to receive a player's command, process it, and send corresponding encoding scenes back to the player.

Playout delay (OD): The time required for a client to receive, decode and render a frame on the screen.

Network Delay (ND): The time required for a round of data exchange between the server and client. It is usually referred to as the network round-trip time (RTT). Studies of traditional gaming systems have concluded that different game genres have different thresholds of response delay.

3 RELATED WORK

3.1 Options in Cloud Gaming

The smartphone has changed everything. Since the advent of the iPhone in 2007, smart phones and tablets have outpaced PC and Laptop purchases. The smartphone and tablet have become de facto gaming platforms, with device performance standards today that rival early generation consoles. Also, upgrades to cellular and broadband networks globally have stabilized data transfer to those mobile devices, and that stability is only going to get faster and more stable.

Various options emerging and that have been established over the years are

3.1.1 OnLive

The idea of cloud gaming over the Web was first introduced to the public in the 2009 Game Developers conference. OnLive demonstrated cloud gaming by streaming the game, Crysis over the Internet from a mile away to be played on a low-end laptop. Users can stream games available in the OnLive store using OnLive's own client. They were off to a good start gaining the attention of testers, game developers and investors. [18]

Figure 3.1 OnLive

In about three years later, OnLive started letting go of their employees. They can't seem to make money from cloud gaming. They have thousands of subscribers but none of them are the paying kind since players can try out game demos on the OnLive store for 30 minutes, for free.

3.1.2 GaiKai

Then, there is GaiKai, founded in 2008, a company which focuses more on selling their platforms to game manufacturers who wish to provide cloud gaming options to their players. The GaiKai platform is similar to how you stream Netflix on your Smart TV. GaiKai allows white-label businesses with basically any manufacturer. [14]

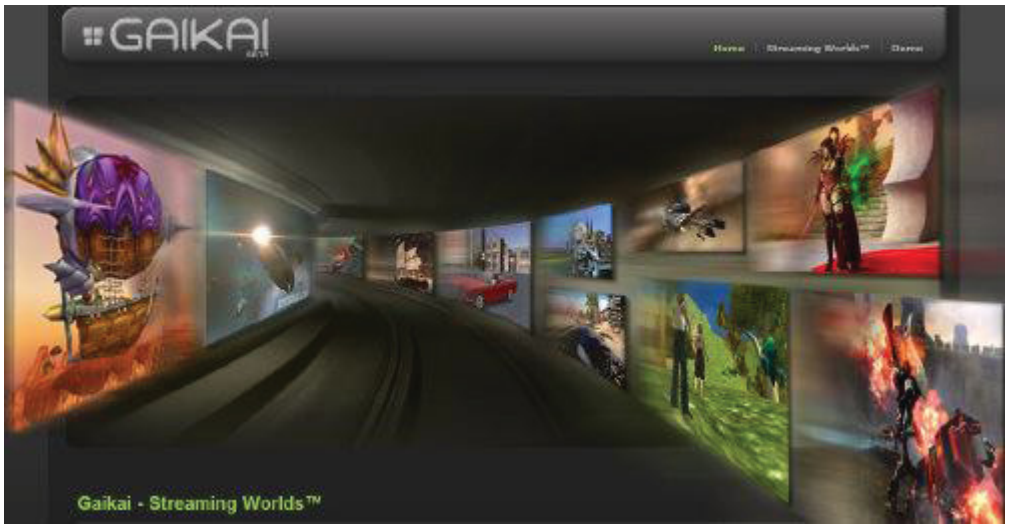

Figure 3.2 GaiKai

So, if for instance, Samsung wants to have their own cloud gaming platform on their range of TVs, GaiKai will build it for them. They had a partnership with Electronic Arts and Ubisoft and was in the midst of making more mainstream games available on their platform. Then, Sony showed up and bought them. For $380 million. [14]

3.1.3 StreamMyGame

This is a complete software only solution that enables games and applications to be played remotely. Our FREE to join community website enables you to both stream games and record games to video. It is revolutionizing the core fundamentals of the computer games industry because members can play high end games on low end devices. [14]

Figure 3.3 StreamMyGame

Features for StreamMyGame are:

- Access and play games remotely via their local/home network. Play high end PC games on a netbook, old PC, notebook, HTPC or certain mobile devices.
- Access and play games remotely via their broadband network. Please note you need a fast broadband uplink to use this service.
- Record game-play to HD video files. Upload your recorded HD video files to YouTube, Facebook and other online video sites.
- Broadcast games so anyone on their local network can watch.
- Members also get access to our community services to meet other gamers and form groups with forums, chat, search, who's online, private messaging, profiles, avatars and news.
- It is compatible with Windows XP, Vista and 7 and with Linux including Ubuntu, Red Hat, Fedora, Debian and Xandros.

3.2 NVIDIA CUDA

CUDA (Compute Unified Device Architecture) is a parallel computing platform and API model developed by NVIDIA, that enables computing performance by coupling the power of the graphics processing unit. Software developers and software engineers can use CUDA-enabled GPU for general purpose processing which is termed GPGPU.

Cuda Image processing, computer visualization, scientific computing etc are the compute intensive tasks exhibited by CUDA. CUDA programming is based on the data parallel processing model. C, C++, Fortran, Java and Python programming languages are used in CUDA platform. C, C++ Programmers use CUDA C/C++ compiled with nvcc, Nvidia's LLVM-based C/C++ compiler. The prior API's like Direct3D and OpenGL requires advanced skills in graphics programming, but this makes it easier for specialists in parallel programming to use GPU resources. OpenACC, OpenCL are CUDA supported programming frameworks. [2]

According to few papers in my research CUDA exposes a fast memory region which is shared amongst threads and allows threads in the same block to coordinate their activities using a barrier synchronization function.

Figure 3.4 NVIDIA CUDA

3.3 Architecture of GamingAnywhere

GamingAnywhere is the first open source cloud gaming system, that has been the primary focus of our thesis. GamingAnywhere has three main advantages over other existing systems.

GamingAnywhere is an open system, in the sense that a component of the video streaming pipeline can be easily replaced by another component implementing a different algorithm, standard, or protocol. For example, GamingAnywhere by default, uses x2647 and vpxenc(WebM) to encode captured raw videos. To expand GamingAnywhere for stereoscopic games, an H.264/MVC encoder may be plugged into it without significant changes. More generally, various algorithms, standards, protocols, and system parameters can be rigorously evaluated using real experiments, which is impossible on proprietary cloud gaming systems. [19]

GamingAnywhere is cross-platform, and is currently available on Windows, Linux, OS X, and Android. This is made possible largely due to its modularized design. GamingAnywhere has been designed to be efficient, as can be seen, for example, in its minimizing of time and space overhead by using shared circular buffers to reduce the number of memory copy operations. These optimizations allow GamingAnywhere to provide a high-quality gaming experience with short response delay. On a commodity Intel i7 server, GamingAnywhere delivers real-time 720p video at ≥35 fps, which is equivalent to less than 28.6 ms of processing time for each video frame, with a video quality significantly higher than that of existing cloud gaming systems. GamingAnywhere outperforms OnLive by 5 dB in Peak Signal-to-Noise Ratio (PSNR). [18] [19]

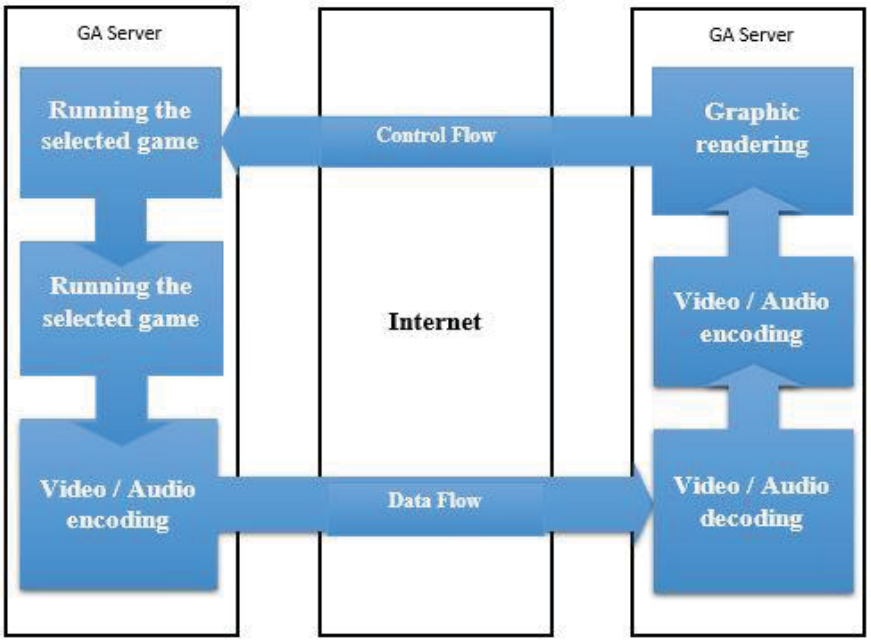

Figure 3.5: GamingAnywhere System Architecture

3.3.1 GA server

The selected game runs on the game server along with an agent. The agent is a thread injected into the selected game. The first requirement of the agent is to capture the video and audio frames that are produced by the chosen game and encode the frames using chosen codecs [18]. These frames are delivered to the client via the data flow. The second requirement is to communicate with the game dynamically. As the user sends actions from

the client, the server must behave, as though it was the user and must re-play the received input events [19]. The server operation involves two main mechanisms to capture game screens.

The first mechanism performs a screen capture of the entire desktop periodically, then extracts the region associated with the game screen. After this, the second mechanism is that the video frame data is obtained from the graphic rendering buffer [19]. This method is executed by using the hook function of DirectX APIs. By using the Window Audio Session API, the audio frames were captured for the game sound. After getting the audio and video frames on server side, the encoding is done by utilizing the libavcodec library. Then the encoded information is streamed to client from the server. Fig 3.5 shows the architecture of GamingAnywhere.

In GamingAnywhere, the control flow and data flow establish the interaction between the GamingAnywhere server and client. For handling the control flow and data flow, the libraries such as libavformat and live555 are used on both sides [19]. The audio and video frames were streamed to the client from the server by using two ways. One is by transmitting RTCP/RTP packets through the RTSP over TCP and the other way is by transmitting the RTP packets over UDP. The GamingAnywhere client handles protocols such as RTSP/RTP corresponding to the method used on the server side.

3.3.2 GA client

The GamingAnywhere client decodes the information which is streamed from the server and render the video and audio frames. For decoding, the GA client also utilizes libavcodec library. Zero-buffering mechanism is used by the decoder. This mechanism is used to reduce the effect of delay while playing the game [19]. SDL library is designed to provide the real time rendering of the decoded video frames through Direct3D and OpenGL. The SDL library precisely read and render the encoded audio frames. After that, SDL library also take the user inputs such as the tapping of the touchscreen on mobile and the pressing of keys on keyboard and moving the mouse in case of Windows and Linux. After capturing the user inputs, they are sent to server side from the client side through control flow.

4 METHODOLOGY

In this chapter, we discuss the research methodology of the thesis. The techniques to measure the GPU Performance of a system while playing the game and without playing the game. To reach the goals, the research technique must be on par with the wanted outcomes.

4.1 Research Technique

In this research techniques, measurements and statistical analysis of data are mainly more useful. From many sources, the data has been collected with a specific end goal. Statistical analysis is done with the gathered data.

The final outcomes and analysis are exclusively based on numerical results from experimental tests. The method of research was utilized to achieve the GPU performance of a system using GamingAnywhere.

4.2 Evaluation Technique

For the study of a system, experiments are performed according to the model of the system. Physical and Mathematical models are available for the model of a system. The physical model is the replica of the actual system, carries all properties and characteristics of the actual system.

This research is the experimentation of the physical model with the involvement of the GamingAnywhere performance for mobile cloud gaming. A mathematical simulation model is considered because the main goal is to implement GA on a physical system to measure the GPU performance.

4.3 Experimental Testbed

4.3.1 Testbed Design

The choice of environment test bed was made on a personal computer, which consists of a server and a client on mobile. GA server is recognized on a system consisting of ubuntu 12.04 having intel core i7 dual-core processor, 8GB of RAM, hard disk of 1TB and NVIDIA graphics card (920M series) with 4GB of memory. The GA mobile client was installed on Vivo v5s having Octa Core 1.5GHz cortex-A53 and good capacitive touchscreen type with 1080 x1920 pixels and Android OS v6.0 Marshmallow.

The GA server is a outside test enclosure stream which streams the game to the GA client. GA client is a inside the test enclosure. GA server and GA client are linked to the same wireless LAN. The inputs from the GA client will be streamed back to the GA server. Both GA server and GA mobile client were connected to the WLAN AP via 802.11 wireless LAN and Gigabit Ethernet LAN, both GA server and GA client are in the same LAN. AssaultCube game, GPU acceleration NVIDIA CUDA, Virtualization technique gVirtuS are installed in the GA server to perform the experimentation. The experimentation will be explained in the next section.

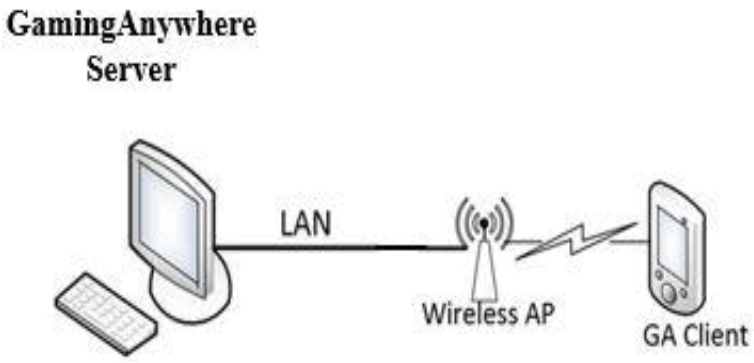

Figure 4.1: Experimental test-bed

Devices	Technical Specification	Description
Server	Intel core i7 dual-core processor, 8GB RAM, Hard disk 1TB, NVIDIA graphics card 920M and 4GB Memory.	GA Version 0.8.0
Client	Vivo v5s, Octa Core 1.5GHz cortex-A53, good capacitive touchscreen type, 1080 x1920 pixels and Android OS v6.0 Marshmallow	GA Client- v22
WLAN	D-Link DAP-1522, Firmware 1.21	Wireless Access point.

Table 4.1: Technical specifications

4.3.2 GamingAnywhere

GamingAnywhere is an open-source cloud gaming platform which is used to deploy cloud games. As GamingAnywhere is an open-source, can detailly study about cloud gaming. The figure illustrates a sample cloud gaming service based on GA. A person playing gaming will operate or gives the commands from the keyboard, mouse and touch input to the server. The game server receives the commands and starts using them to play the game and sends the streams encoded frames to the client. Finally, the client decodes and renders the game captured server frames to the local console. In the experiments, however, as VMware Horizon View has already acted as cloud gaming platform, GamingAnywhere is not utilized to deploy cloud games but to capture system times and images, which are then used to calculate response delay and image quality respectively. [20]

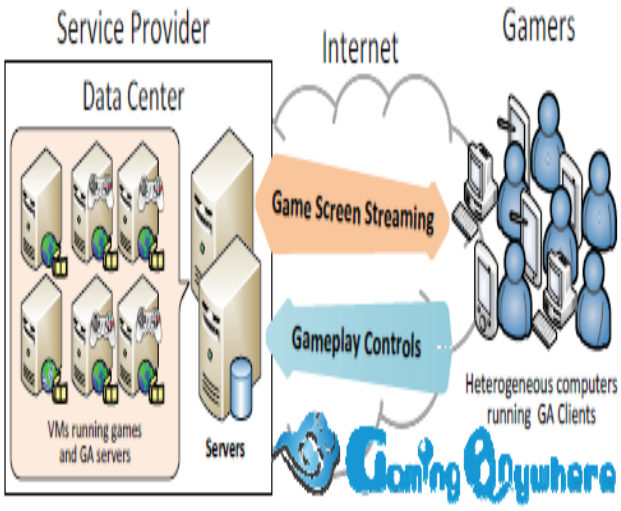

Figure 4.2: A Cloud Gaming Service Using GamingAnywhere

4.3.1 GamingAnywhere server

Audio source, Video source, RTSP server and input replayer were installed while installing GamingAnywhere package. When the process was installed and get initialized the video and audio source will be kept idle. [21] The encoder threads were launched when the client gets connected to the RTSP server. The encoder sends a notification to a module as it is waiting to encode the frames which are captured by the server. The video and audio frames were captured by the source module when more than one encoder get to work. The video and audio encoded frames will be generated all at a time.

Figure 4.3: GA Server

4.3.2 GamingAnywhere client

The encoded video and audio frames were dispatched by GamingAnywhere client which presents real time game screens. To render video and audio frames client uses one string and to handle user inputs it uses another string.

An android application called GA client is installed on mobile which is nothing but GamingAnywhere client. Fig is the GA client software architecture. Fig 2 shows the AssaultCube game screen.

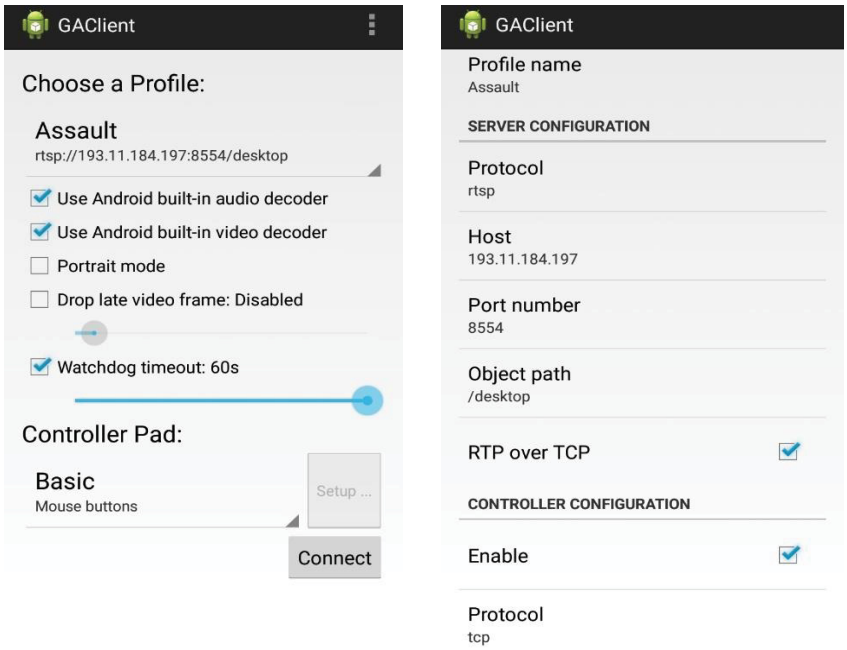

Figure 4.4: GA Client mobile application

Figure 4.5 Assault cube game running on mobile device

4.3.3 AssaultCube

AssaultCube is free and easy to download. It is a first-person shooter game and either single player or multiplayer can play. The game is of a size 40MB, it is a small game and is available for Windows, Mac and Linux.

4.4 Experimental Tools

Experimental tools used in the experiments, including graphics card benchmarks and performance monitoring tools. The graphics card benchmarks run on both virtualization and non-virtualization environments. Performance monitoring tools are responsible for capturing time stamps of hardware events by benchmarks and gaming applications.

4.4.1 Graphics Card Benchmarks

GPU based programs will be running to evaluate the performance of a given GPU. The can be mainly used to measure the GPU Performance such as capability in terms of frames per second. Graphics card testing includes synthetic tests and real-world tests. Synthetic test usually useful in determining the maximum possible performance of a graphics card. Real-world test is based on real games and used to evaluate GPU performance in real-world scenarios. To evaluate GPU Performance in each tested configuration, many Graphic card benchmarks included two synthetic tests and a real-world gaming test are used in the experiments.

4.4.1.1 PassMark PerformanceTest

It is a tool which help us to gain objective benchmark results on a PC using different speed tests, including CPU, memory, hard disk and graphics card. In terms of graphics card tests, it provides a series of basic 2D tests and advanced 3D tests.

4.4.2 Performance Monitoring Tool

Although we can gain performance metrics such as frames per second (fps), from graphics card benchmarks, still it is necessary to gain further resource utilization analysis to determine the main bottleneck. The following two chosen performance monitoring tools are to keep track of each graphics card benchmark resource utilizations.

4.4.2.1 MSI Afterburner

MSI Afterburner is free to use and is compatible with all graphic cards. Critical hardware resource information such as CPU usage, GPU usage and graphics memory usage in real time is monitored to users. Logging functionality is provided where users can record all hardware resource utilization periodically while running graphics card benchmark and game simultaneously.

Benchmarks	Features
PassMark Performance Test	• DirectX benchmark • Comprehensive test for GPU in terms of 2D and 3D capability • Support for DirectX 9 and higher • Has a point-based ranking system
MSI-Afterburner	• Real time testing benchmark

Table 4.1 Experimental Tools

5 EXPERIMENTATION AND RESULTS

5.1 Experimental Procedure

In this experiment, GamingAnywhere server, GamingAnywhere client on mobile were used. AssaultCube game was installed on the server and the game is streamed to the client, the game AssaultCube was chosen as it is the most popular game and easily to analyze the game Frame rates, on GPU system. Once the server streams the values to client, client sends the input to the server. While the game streaming is going on, the GPU performance values are measured and graphically represented by MSI-Afterburner Tools.

CUDA (Compute Unified Device Architecture) is a parallel computing platform and API model developed by NVIDIA, that enables computing performance by coupling the power of the graphics processing unit. Software developers and software engineers can use CUDA-enabled GPU for general purpose processing which is termed GPGPU.

CUDA Image processing, computer visualization, scientific computing etc. are the compute intensive tasks exhibited by CUDA. CUDA programming is based on the data parallel processing model. C, C++, Fortran, Java and Python programming languages are used in CUDA platform. C, C++ Programmers use CUDA C/C++ compiled with nvcc, Nvidia's LLVM-based C/C++ compiler. [2]

The prior API's like Direct3D and OpenGL requires advanced skills in graphics programming, but this makes it easier for specialists in parallel programming to use GPU resources. OpenACC, OpenCL are CUDA supported programming frameworks.

According to few papers in my research CUDA exposes a fast memory region which is shared amongst threads and allows threads in the same block to coordinate their activities using a barrier synchronization function.

The performance metrics obtained in our research are image quality and frame per second streaming rates in our mobile device and computer device. To calculate these metric values, GamingAnywhere has been preassigned with a C++ programming script which will generate a log file after each game session, the game in question being AssaultCube.

We have been vast in our research by using various types of resolutions for our gaming experience and analysis, which would be 1024x768, 800x600, 640x480.

5.2 Experiment Metrics

The performance metrics analyzed in this experiment

5.2.1 Delay

It is defined as, the time interval of packet travelling from source to destination, which can also be referred as one-way delay. Round trip delay is referred when the time taken for vise-versa (packet travelling from destination to back source).

5.2.2 Frame Rate

It is defined as the rate of the frequency at which the images frames are displayed. It is measured in frames per second (fps). 50fps (frame rate) are used in this experiment as it is the default value for the GA server.

5.2.3 Image Quality

To measure

5.1. The formula used for bandwidth calculation is

$$\text{MSE} = \frac{\sum_{j=1}^{N}(\sum_{i=1}^{M}(f_{i,j} - F_{i,j})^2)}{M\ X\ N}$$

$$\text{PSNR} = 10\log\frac{255^2}{MSE}$$

5.3 Experimental Scenarios

Different experimental scenarios are carried out in the experimental research for 30 iterations. All the iterations are done for 15minutes each. The tools mentioned beforehand are used to calculate the following values which can help infer better conclusions.

5.3.1 Scenario 1

Resolutions	Bitrate (Kbps)	Packet Loss	Jitter	Frame Rate(fps)
1024x768	2819.37	0	693.62	46.4
800x600	2212.3	0	1222.33	38.88
640x480	1764.25	0	1785.68	32.84

Table 5.1: Performance metric Analysis using NVIDIA GEFORCE 920M processor

Resolution	PSNR MEAN
1024x768	51.2
800x600	40.3
640x480	37.1

Table 5.2: Image Quality (PSNR MEAN) using NVIDIA GEFORCE 920M processor

From first Scenario, in table1: Performance metric were calculated for three different resolutions using Intel Processor. In table2: Image Quality was calculated for different resolutions using Intel Processor.

5.3.2 Scenario 2

Resolutions	Bitrate (Kbps)	Packet Loss	Jitter	Frame Rate(fps)
1024x768	2004.3	0	367.8	29.06
800x600	1886.6	0	286.2	32.47
640x480	1635.8	0	213.8	34.84

Table 5.3: Performance metric Analysis using INTEL PROCESSOR

Resolution	PSNR MEAN
1024x768	45.5
800x600	40.1
640x480	36.8

Table 5.4: Image Quality (PSNR MEAN) using INTEL PROCESSOR

From first Scenario, in Table 5.1 and Table 5.2: Performance metric were calculated for three different resolutions using NVIDIA GEFORCE 920M Processor. In Tables 5.3 and 5.4 Image Quality was calculated for different resolutions using just the Intel Processor.

6 ANALYSIS AND DISCUSSION

6.1 PassMark Performance Test

Figure 6.1 PassMark Performance Results

The above figure shows the results of the PassMark Performance Test 2D benchmark. From the results, we can see that the system when using the Intel Processor behaves poorly compared to the other tested configuration, which is with the NVIDIA GeForce 920M Processor. It achieves a overall 1628 score with the 2D Graphics Mark having a 478, which is pivotal to the research because the game employed in this research is AssaultCube, which is 2D. However, the 3D Graphics Mark is 863 which is irrelevant here.

6.2 MSI Afterburner

During the experimentation procedure, we have launched MSI Afterburner before the execution of benchmarks and all gaming applications, to record all key events, GPU usage and graphics memory usage. The logging functionality is enabled which is pre-programmed into the software.

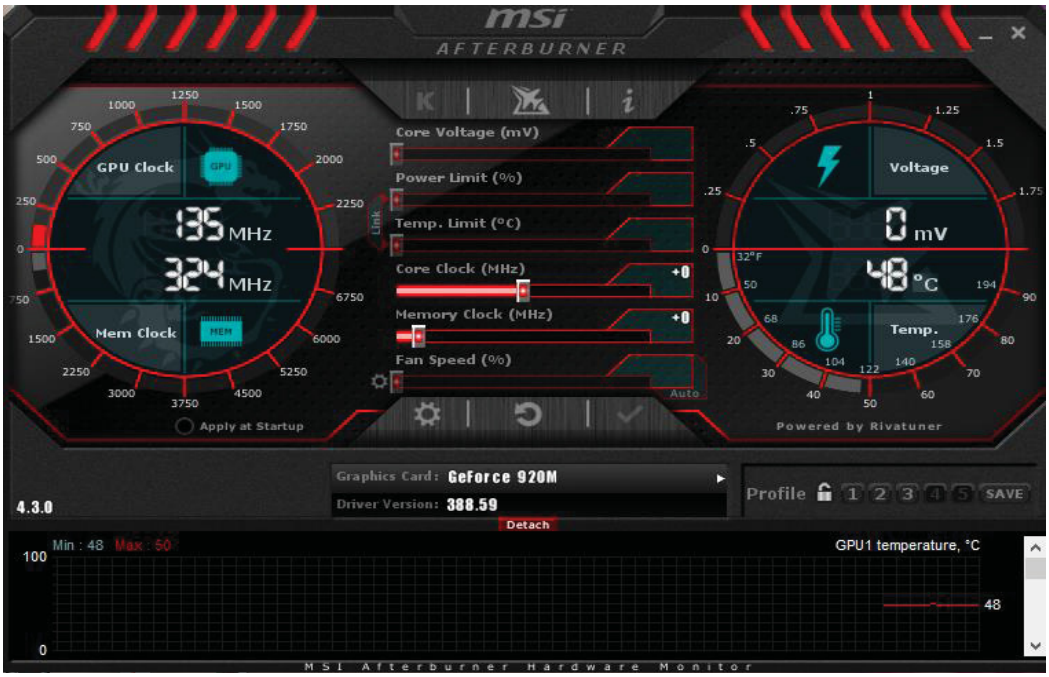

Figure 6.2 MSI Afterburner dock

The aforementioned logging of the MSI Afterburner produced results which gives us an insight on the effects of the gaming application has on the Graphics Processing Unit, and the Central Processing Unit.

Figure 6.3 Logs of MSI Afterburner

7 CONCLUSION AND FUTURE WORK

The last chapter of this document is going to be talking about the inferences and conclusions of all the experimentation and the simulations we have performed over the duration of this project. The research questions will be answered to an extent which will be satisfactory. This area provides a section for the possible future work that can be conducted to contribute to the field of study of mobile cloud gaming and GamingAnywhere, which will also be helpful to raise even better research questions.

7.1 CONCLUSION

During the beginning of this research, we started with the assumption that GamingAnywhere would be the best outlet to analyze the performance of a Graphic Processing Unit (GPU), with the two scenarios of enabling its Graphics Card, as so, or enabled with NVIDIA CUDA acceleration. After performing various experimentations, we have realized that GamingAnywhere was indeed the best way to go to reach these outcomes. We then later wanted to work with a specific type and procedure of NVIDIA CUDA acceleration, known as gVirtuS. We chose this route in our research because we felt that this would be the best way to talk about the advantages, pros and cons to NVIDIA CUDA accelerations in GPUs. At the end of this research, we have analyzed the properties of these elements of our project and hence can state the options we chose were the right options for us.

In this research, we started our experimentation by seeing how to quantify the performance metrics such as Image Quality and Frame per Second streaming of the mobile device which was using GamingAnywhere while playing the AssaultCube.

Along with this performance related question, we looked at various theoretical research queries and questions which would in turn be the turning point in our research. The research questions talked about during our project have been answered in the next question.

As we have observed in our research analysis, the best way to analyze the performance metrics for the mobile cloud gaming device and computer device, is to vary the resolutions of the monitor from 1024px to 640px. When these resolutions varied, we have noticed that the performance metrics, like image quality and frame rate per second has decreased from higher to lower.

7.2 ANSWERS TO THE RESEARCH QUESTIONS

1. What is the possibility of using GPU Acceleration in GamingAnywhere?

Ans. While performing our experiments, we wanted to venture into the region of applying the workings of GamingAnywhere and analyzing them in two different scenarios. The scenarios would be one with it being GPU accelerated with the help of NVIDIA GeForce 920 graphic card and one scenario would be one without enabling the graphic card while playing the game. The game in question here is AssaultCube, which is played both on the mobile device, and the computer with the graphics card. This question can be answered by the fact that our experimentation was successfully performed by using the two scenarios, to which we can attest to by pointing out the difference and comparing the values of the performance metrics between when the game was played using the NVIDIA Graphic Card whose Graphic Processing Unit was accelerated.

2. **Can we use gVirtuS virtualization technique in implementing NVIDIA CUDA GPU acceleration in GamingAnywhere?**

Ans. Yes, we can use the gVirtus virtualization technique in implementing GPU acceleration, as it helped to improve the performance stats of the computer device and the mobile device in which GamingAnywhere is being implemented. This can be seen by Tables 5.1, 5.2, 5.3 and Table 5.4 which shows the image quality and frame per seconds, clearly stating that the acceleration of the graphics processing unit is enhanced and made better when implementing it with NVIDIA CUDA GPU acceleration by using the gVirtus virtualization technique.

3. **What is the impact on the CPU, GPU and its performance when GamingAnywhere is used in a client-server mode?**

Ans. In the experiment, we have noticed that the effect of GamingAnywhere on the CPU, and GPU is not that substantial. However, we can see that there is change when the device running GamingAnywhere is subjected to change relating to resolutions of the screen and the processor being used to run GamingAnywhere. From Table 5.1 and Table 5.3, we can see that the frame rate per second while using the NVIDIA GeForce 920M processor to run the GPU, is higher than the frame rate per second while using the Intel Processor. Subsequently, it is noticed that when the resolution of the screen is also decreased, hence is the frame rate per second of the concurrent size.

Inferring from Table 5.2 and Table 5.4, we can say that the calculated Peak Signal to Noise Ratio also behaves the same way, like the frame rate per second, which is to say that the PSNR also increases with the usage of the NVIDIA GeForce 920M Processor rather than Intel Processor. The PSNR represents the quality of image in the screen of the devices using GamingAnywhere.

7 FUTURE WORK

As our research is focused on the smaller and minute changes in the implementation and usage of NVIDIA CUDA implementation, gVirtuS and such, and then realizing and interfering its performance metrics in both the mobile device and computer, it can imply for more future research and works for further projects. Quantifying performance metrics for larger time-scale metrics and with the different strategical and action adventurous games can be extended as a future work.

Using these research techniques, we have performed our project and experimentation in only the Windows OS, it further implies scenarios for experimentations and simulations in Linux Ubuntu and other platforms.

8 REFERENCES

[1] C. Reaño and F. Silla, “A Performance Comparison of CUDA Remote GPU Virtualization Frameworks,” in 2015 IEEE International Conference on Cluster Computing, 2015, pp. 488–489.

[2] “An Easy Introduction to CUDA C and C++,” NVIDIA Developer Blog, 31-Oct-2012. [Online]. Available: https://devblogs.nvidia.com/easy-introduction-cuda-c-and-c/. [Accessed: 16-May-2018].

[3] C.-L. Hsu and H.-P. Lu, “Why do people play on-line games? An extended TAM with social influences and flow experience,” Inf. Manage., vol. 41, no. 7, pp. 853–868, Sep. 2004.

[4] M. Chabbi, K. Murthy, M. Fagan, and J. Mellor-Crummey, “Effective sampling-driven performance tools for GPU-accelerated supercomputers,” in 2013 SC - International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2013, pp. 1–12.

[5] “Game Development - Amazon Web Services.” [Online]. Available: https://aws.amazon.com/gaming/. [Accessed: 16-May-2018].

[6] “GPU acceleration for Windows Server OS.” [Online]. Available: https://docs.citrix.com/en-us/xenapp-and-xendesktop/7-9/hdx/gpu-acceleration-server.html. [Accessed: 16-May-2018].

[7] O. Soliman, A. Rezgui, H. Soliman, and N. Manea, “Mobile Cloud Gaming: Issues and Challenges,” in Mobile Web Information Systems, 2013, pp. 121–128.

[8] “Specs & Features of GRID Cloud Gaming GPUs |NVIDIA.” [Online]. Available: http://www.nvidia.com/object/cloud-gaming-gpu-boards.html. [Accessed: 16-May-2018].

[9] J. Y. Li et al., “The Implementation of a GPU-Accelerated Virtual Desktop Infrastructure Platform,” in 2017 International Conference on Green Informatics (ICGI), 2017, pp. 85–92.

[10] “What is virtual machine | Pros and cons of virtual machines.” [Online]. Available: http://www.serial-server.net/virtual-machine/. [Accessed: 16-May-2018].

[11] Z. Zhuo, “A Performance Comparison of VMware GPU Virtualization Techniques in Cloud Gaming,” p. 93.

[12] “What is virtual machine | Pros and cons of virtual machines.” [Online]. Available: http://www.serial-server.net/virtual-machine/. [Accessed: 16-May-2018].

[13] J. Y. Li et al., “The Implementation of a GPU-Accelerated Virtual Desktop Infrastructure Platform,” in 2017 International Conference on Green Informatics (ICGI), 2017, pp. 85–92.

[14] B. L.-L. U. 08 May’18 2018-04-20T23:53:14+00:00, “The Best Cloud Gaming Services: Welcome Gaming to the Web,” Cloudwards, 20-Apr-2018. [Online]. Available: https://www.cloudwards.net/top-five-cloud-services-for-gamers/. [Accessed: 16-May-2018].

[15] “Specs & Features of GRID Cloud Gaming GPUs |NVIDIA.” [Online]. Available: http://www.nvidia.com/object/cloud-gaming-gpu-boards.html. [Accessed: 16-May-2018].

[16] “Quantifying User Satisfaction in Mobile Cloud Games.” [Online]. Available: http://www.iis.sinica.edu.tw/~swc/pub/user_satisfaction_in_mobile_cloud_gaming.html. [Accessed: 16-May-2018].

[17] “Placing Virtual Machines to Optimize Cloud Gaming Experience.” [Online]. Available: http://www.iis.sinica.edu.tw/~swc/pub/virtual_machine_allocation_for_cloud_gaming.html. [Accessed: 16-May-2018].

[18] “OnLive.”

[19] “GamingAnywhere - An Open Source Cloud Gaming System.” [Online]. Available: http://gaminganywhere.org/index.html. [Accessed: 16-May-2018].

[20] “GamingAnywhere - Performance.” [Online]. Available: http://gaminganywhere.org/perf.html. [Accessed: 16-May-2018].

[21] S. Musinada, On energy consumption of mobile cloud gaming using GamingAnywhere. 2016.

[22] V. V. S. S. G. Grandhi, On the Quality of Service of mobile cloud gaming using GamingAnywhere. 2016.

9 APPENDIX

Appendix

GamingAnywhere Scripts

SERVER SCRIPTS

ga-server-event-driven config/server.assaultcube.win32.conf

CLIENT SCRIPTS

ga-client config/client.rel.conf rtsp:///192.168.186.244:8554/desktop

AssaultCube Configuration File

#configuration for the assaultcube game #work with ga-server-event-driven [core] include=common/server-common.conf include=common/controller.conf include= common/ video-x264.conf include=common/video-x264-param.conf include=common/audio-lame.conf [video] video-fps=50[audio] audio-init-delay=1000 [filter] filter- source-pixelformat=rgba [ga-server-event-driven] game-dir=D: AssaultCubePortable App assaultcube game-exe=bin_win32 ac_client.exe # hook configuration # version: d9, d10, d10.1, d11, dxgi, sdl hook-type=sdl hook-audio=coreaudio enable-audio=true enable-server-rate-control=Y server-token-fill-interval=20000 server-num-token-to-fill=1 server-max-tokens=2