CERTIFICATION OF APPROVAL

Weapon Simulator

by

Sedzurudin Baharudin (2067)

A project dissertation submitted to the
Information System Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION TECHNOLOGY)

Approved by,

[illegible]

(Mr Anang Hudaya)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
November 2004

t
UF
503
.S447
2004

1. Synthetic training devices
2. IT/IS -- Thesis

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

(SEDZURUDIN BAHARUDIN)

Abstract

The Weapons Simulator Project is a project to produce a working weapons simulator for the purpose of preliminary weapons training. The proposed outcome of the project would be a usable weapon simulator, consisting of computer graphics for the environment, physics of the ammunition projectile, visual and sound effects of the application, combined with computer and virtual reality hardware for the navigation and handheld tools. The environment of the simulator will be produced using OpenGL libraries, compiled on Microsoft Visual C++. For the sound, OpenAL libraries will be used. These two libraries were picked for the development of the project as they are easy to use and are widely used for visualization, computer graphics and simulation purposes. Virtual reality hardware such as the virtual reality goggles and tracker are used to enhance the realism aspect of the product. Other than the components stated above, there are also other software and tools used in the development of the simulator, such as 3D Studio Max 5.0 for the modeling of the 3D models, 3D Exploration as the 3D viewer and Adobe Photoshop 7.0 as the texture and picture editor. In all, the simulator will comprise of numerous tools and theories either scientifically or multimedia. And because of the integration of these properties, the Weapon Simulator Project should be a challenging and rewarding project to be accomplished. During the research of the project, physics equations were some of the hard properties. Ways of programming were also stumbled upon such as the usage of Raw Loaders, 3ds Loaders, usage of Heightmaps and the usage of Tga or Targa files in computer graphics. As a conclusion, the project was an adventure and an integration of physics and computer graphics to produce a working simulator.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to the Allah S.W.T for giving the strength, wisdom and patient to complete this project.

I also would like to indicate that this project could not have been completed without the help and generosity of many people. My deepest appreciation goes to Mr Anang Hudaya, Final Year Project (Information Technology) Supervisor for all his guidance, patience, advices and motivation which have inspired me towards completing this project.

A special appreciation and thanks to my beloved parents Mr. Baharudin Bin Mat Jais and Mrs. Zainab Binti Kamarudin, who were there to shower me with their encouragement and love. I would also like to include my siblings Huzaizudin Bin Baharudin, Idzamudin Bin Baharudin, Juliana Liza Bin Baharudin, Idzwarudin Bin Baharudin, Zulhaini Binti Zulkarnain and Puteri Haslinda Binti Megat Supian. There were times when the ties beyond friendship are needed to get us through the day.

I would also like to thank ShapeShifters Incorporated for giving me the chance to work under their wing. The days of my internship at ShapeShifters Incorporated supplied me with the knowledge to undergo this project.

And last but not least, to all my friends all around the world, especially Ms. Nur Alicia Binti Kamarudin, thank you for being there for me. All your patience and commitments were my inspiration throughout the project development. Thank you all for sharing brilliant ideas, knowledge and worthy experiences with me to accomplish this project. May Allah bless you all.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND STUDY

Weapons are the main tools for an infantry in defending the nation and themselves against attacks from enemy forces. However, before qualifying to operate one, a soldier must first undergo training for the specific weapon. There are many aspects need to be taken into account in preliminary weapons training. Firstly, there is the human aspect of getting used to a tool or process, which in this case is getting used to the weapon in use. The learning process for each individual is not the same and it is much harder if it involves motoring skills. Some of the trainees may take much more time to get used to the weapon than others. This takes us to another aspect which is the cost of weapons training. For the time being, every stage of weapons training is done using real weapons and live ammunitions. However, this kind of training method is not suitable for preliminary stages, as we have discussed earlier, different people have different learning speed. So, it would be a waste of cost, which in this case live ammunitions, where a certain trainee needs to practice more on a certain weapon than the others. Seeing this occurrence from another aspect, it could also be seen as a waste as live ammunitions are used but not even a single foe is killed. Another human aspect that is to be taken into is safety and human error. Human error is something that cannot be predicted but can only be prevented. In handling tools which are new to ones knowledge, mishaps are prone to happen. Although safety measures were originally laid for us to follow, accidents still occur. In the future, they will be exposed to the real hazard of the weapon in which they have in their hands. However, is it worth while if it is still in the preliminary phase? As a soldier, would you rather die in combat or at the training field?

1.2 PROBLEM STATEMENT

1.2.1 Problem Identification

The main aspect in which the problems occur within the preliminary phase of weapons training is from human factor. Human factor gives a chain effect which will then trigger other problems. Firstly, the learning process of the human mind is focused. It is well known that different people have different capabilities and this includes how fast they learn new things. However, motoring skills are the hardest among the trades to be trained. The current training curriculum gives access to the trainees to use real weapons during every level of training. So, taking the problem at hand, which is training new cadets in using hi-tech weapons, we can say that it would be a waste if we give them the real thing at the first lesson. This is because, they would take certain time in getting use to the weapon. What if they cannot familiarize themselves with the weapon within the time frame and thus need more training? Would not that incur more cost?

Cost is another aspect which is to be discussed. At the present curriculum will use live ammunitions for training. It would seem that the ammunitions are just used for target practice rather than on the enemy. We can accept using small bullets for target practice, but what if highly dangerous mortar or explosives are used? Ammunitions cost thousand and would be a waste of money if they are not used for defense purposes. Live ammunitions should be substituted with a much cost effective method so that other than being effective, the infantry itself can invest in other things such as weapon advancement or research and development.

Safety is also an important aspect that should be taken into account. In preliminary weapons training, the users are in an amateur level. So, the users are prone to make a mistake, and making mistake while using a highly dangerous weapon can cause injury and further more, death. The starting phase of weapon training can be substituted and not using the actual weapon. This is because, the preliminary phase only wants the user to get use to the equipment before advancing to a higher stage. The later stages should use the actual tools, where in this case are the actual weapons.

1.2.2 The significance of the project

The project should solve most problems in preliminary weapons training. The system, rather than using the actual weapons, is using a make belief environment. Although it is a make belief environment, it should be efficient in accommodating the users in his or her stage of learning to use a certain weapon. The system can be accessed by the user and thus, speeding up the learning experience. The user can practice using the system again and again, as many times as the user wants. As the system is using computer graphics and computer hardware to real bullets, waste can be cut to a minimum. No ammunitions are wasted and cost can cut to electrical bills and maintenance of the system hardware. Another aspect about the end product is that it is safe. As there is no live ammunition used, the dangerous aspect is taken away from the training session. The user can safely use the system over and over again, and doing mistakes will not incur to a big massacre. Based on the benefits that the simulator can bring, it would be a suitable project to pursue. However for the time being, the system should only be used in the preliminary stages of weapons training. This is because, the users must get their hands on the real weapon to experience the real thing.

1.3 AIM AND OBJECTIVES

1.3.1 Aim

The aim of this research project is to learn and acquire the knowledge, experience and expertise to develop a simulator in the form of a weapon simulator as well as to study the fundamentals. Not only that, but to be familiarize with other virtual reality aspects such as realism, collision detection and impressiveness.

1.3.2 Objectives of the project

The objective of the project is to produce a usable weapon simulator. The final product will be used for preliminary weapons training. This means that the simulator will only be used in the early stages of weapons training. It is also to manipulate computer graphics programming and hardware for the functionality of the simulator and thus achieving realism.

1.4 SCOPE AND RELEVANCE OF PROJECT

1.4.1 Scope of project

The project would cover on operating certain weapons. The project shall cover three aspects which are the environment, physics and visual and sound effects. The environment would comprise a range of terrain with a complete sky covering it. There will also be proper lighting in the environment to give it more realism effect. The environment must be designed in such a way that the user will be immersed into the scenery of the simulator.

The physics is one of the most important aspects in the system. This is because it will control the trajectory of the ammunitions, movement of the user within the environment and also realism properties.

The visual and sound effects are to add to the immersive experience of the user of the system. The visual effects are different than the view of the terrain or environment. Visual effects here mean the smoke from the burning vehicles, ammunition, change of weather and such. Sound effects are not to be left out as the auditory sense must be included to achieve maximum realism. For the time being, he proposed sound quality for the system should be stereo. However, research can be done to achieve surround sound where the user can estimate the distance in which the sound comes from.

All of these aspects are very important as we are to achieve realism in the environment of the simulator. This is because realism is the main factor on how effective a simulator can be. So, these three main aspects are the challenges that need to be completed to produce the weapon simulator.

1.4.2 Relevance of project

Although a simulator needs to integrate numerous physics laws, realism in its environment and computer graphics, the challenge is quite hard but not impossible. The computer graphics will be developed via OpenGL libraries. OpenGL libraries are one of the most used graphics libraries used around the world for visualization, simulation and 3D-shooter games. As it is frequently used, sources and guides in using and manipulating it would not be a problem. However, like any other programming and library, time must be spent for practice and identifying which and what to be used. For the physics, equations are to be used. Physics formulas are abundant in sources. The challenge is to implement it. C++ programming will be used to integrate both the physics and graphical properties of the system. The programming of the physics laws are just like drawing a graph line using C++ coding. 3D models in the environment can either be modeled or downloaded via the internet. The integration of these properties will produce simulator that is able to help amateur users in using hi-tech and dangerous weapons. Although realism is something that sounds quite complicated, however with the usage of the proper tools, the task can be accomplished easily and efficiently.

CHAPTER 2

LITERATURE REVIEW

Simulators have been used by military armies around the world in training their infantry. Such stated by Boh Choun Kiat (2001)

> Even in the era of dramatically reduced defense spending worldwide, simulation is one of the few areas of military budgeting that has not suffered as greatly as others, largely because simulators are used more and more heavily to replace traditional training with actual equipment. Simulation now enables military planners to prepare and train their forces for the complex engagements of the future. Advanced simulators are used to forecast, analyze and plan potential conflicts with degrees of precision that were impossible with previous generation technologies. Emerging simulation technologies will enable manufacturers of the 21st century to build military and commercial systems faster, better and at lower cost than they can today. With the current speed of technological development, it often seems that what is considered speculative today will be in prototype tomorrow and the hot market items before week's end.

So with the advancement of technology and the saving of cost, using such simulators is a must. In addition, the utilization of the technology in hand will not only benefit the military, but also other sectors as studies will be done in growing the knowledge of simulator programming and development.

Dave Eberly (2001) www.harcourt.com

> "Building a real-time collision detection system is by no means a trivial task. A firm understanding is required of the geometry and mathematics for intersection testing, especially when the objects are in motion. The

skilled use of convexity is essential for distance calculations. The system must be designed carefully to support high-performance physical simulations. In particular, spatial partitioning and tight-fitting bounding volumes must play a role in minimizing the computational requirements of the system. The system is sufficiently large that the principles of software engineering apply to its development. Moreover, collision detection is notoriously difficult to implement robustly when using floating-point arithmetic. The challenges of architecting and implementing a collision detection system are formidable!

So, in order to achieve working collision detection, knowledge in programming and computer graphics is not enough. Physics and mathematics are also essential in order to achieve a working collision detection. In fact, collision detection is also one of the most tedious task that lies in this project.

Peter Eberhard and Shoushan Jiang (2000) Mathematical and Computer Modelling of Dynamical Systems Vol.6, No.3, pp. 309-322

An algorithm using interpolation methods for the efficient search of the collision time and state of planar bodies is presented. Using interpolation and directed distances, the algorithm can efficiently obtain information about the collision. Further, a simulation system for multiple bodies is investigated and for some simple examples comparisons are shown of the proposed method and a traditional approach.

Algorithms also play an important role in developing the virtual environment, especially the trajectory aspect. Algorithms join the physics and mathematics with the programming to assimilate the intended output. However, algorithms must not be too complicated or too long as it would waste time during compiling or during program execution.

Chris Hecker (2000) http://delivery.acm.org

> Some game genres generally provide better physics than others. For example, games simulating specific machines, such as flight simulators or car races, have pretty solid physics simulators delivering a convincingly realistic and consistent experience. These domain-specific games have an easier time than games that try to simulate human beings walking around in cluttered rooms. Simulating a speeding race car may seem more complicated than simulating a walking human, but it's actually much easier, because it's a more defined problem. There are well-known physics equations governing the limited number of ways cars behave in motion; these ways of behaving are called the car's "degrees of freedom." Humans, by contrast, have hundreds, if not thousands, of degrees of freedom, and the human brain exerts incredibly subtle control over each one of them.

Although organic movement would look good in the simulator or any virtual environment, it is harder to produce. It is much easier and time saving to assimilate movements of vehicles or non-living things. This is because the physics behind inorganic movement are much easier to apply. Their bodies are much stiff and stiffness on non-living objects are likely to be overlooked. However, stiffness is something that a user would always see and critic if it is detected on a living object.

CHAPTER 3

METHODOLOGY AND PROJECT WORK

3.1 PROCEDURE IDENTIFICATION

Research is defined as the process of findings solutions to a problem after a thorough study and analysis of the situational factors. Although research does not necessary have to solve a problem, it also means of equipping oneself with additional knowledge. In this project, research plays a vital role in understanding the development process and tools to build the application.

There are several steps identified in this research:

3.1.1 To understand what the problem area or issue is

3.1.2 To know where the problem area or issues exists

3.1.3 To identify as clearly and specifically as possible the problems or issues that need to be studied and resolved

3.1.4 To gather information, analyze the data and determine the factors that are associated with the problem or issue

3.1.5 To solve the problem or issue by taking the necessary corrective measures

3.1.6 To check whether the solution for the problem is feasible with constraints such as time and money

3.1.7 To develop the end product with the recommended solutions that meets the constraints

In developing the project, the waterfall and evolutionary development model was combined and thus chosen to be implemented in the project. In this methodology, the system or product which is to be produce is rapidly developed from one phase to another. By doing this, problems and difficulties can be minimized in the later stages as the product must be in full working order before it goes to the next phase. The final product will be achieved after the product has gone through several upgrades and refinement during the development of the product. In other words, several versions of the simulator will be produced, with the next version being more advanced than the one before it. However, only the last version is being used or given to the customer as it is the final product.

By using this approach, certain functionality of the simulator can be focused on during certain stages of development. Only after the stage is passed that the next addition of functionality can be added to the system. Errors and problems are easier to be detected by using this method, thus avoiding the domino effect. At the end, modifications on certain attributes or behavior of the system can also easily be done as the system can be separated into segments, modified and assembled back again.

The Figure 3.1 below is the model used in developing the simulator. Notice that shown below is the well known Waterfall Development Model. However, within each phase, the evolutionary model is used thus enhancing and endowing a much finer product after each phase.

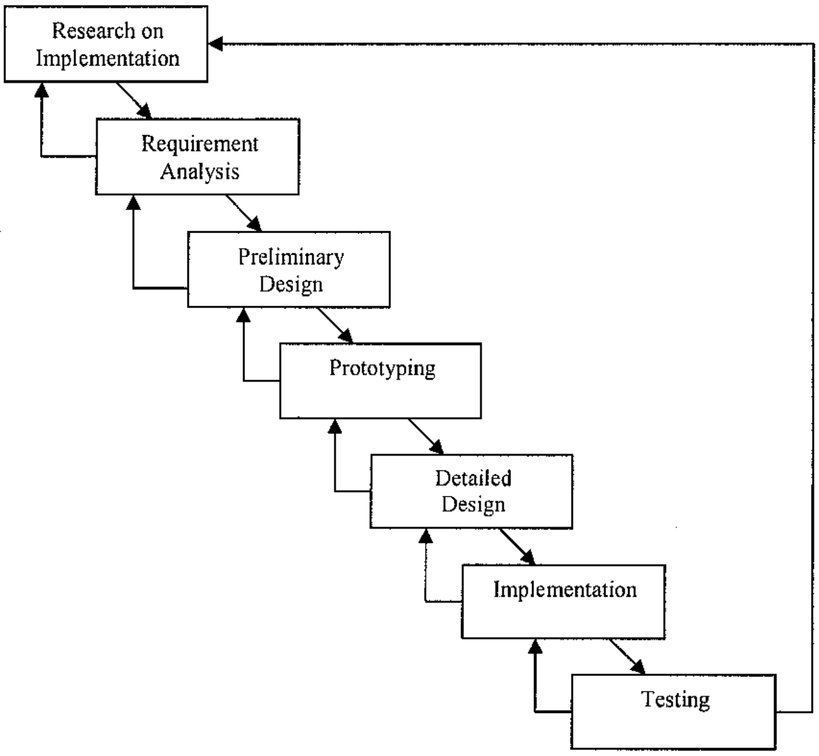


Figure 3.1: The Waterfall Development Model

At each stage, there are specific objectives to be accomplished; where each activity must be deemed successful for work to proceed to the next phase. Below are the activities done during the development of the military simulator.

3.2.1 Research on Implementation

Research on the weapon simulator is done throughout the hardware and software development. However, the main focus was on the software as that will be the main tool in developing the simulator. The hardware entities are only as a platform to run the simulator. So an early assumption was made; the better the hardware, the smoother the simulator will run. The research mainly revolves on these questions:

- How will the simulator be developed?
- What are the tools going to be used for the development?
- What are the minimum requirements needed for the hardware?
- How will the ammunition simulate the projectile motion?
- What are the hardware to be used to substitute the handheld peripherals?

The first stages of research were done via the Internet. The main website referred was www.gametutorials.com. Here, tutorials on using advance OpenGL programming were available. There were also tips on how to troubleshoot programs, forums and programs donated by frequent visitors of the website. The website's purpose is mainly to teach anyone to learn OpenGL programming and mainly to those 3D game enthusiast. This is because most popular 3D shooter games uses OpenGL programming. Examples are Doom, Counter Strike, Quake, Unreal Tournament and such. So most of the questions were answered by going through the forums and tutorials within the www.gametutorials.com website.
For the last question, the university's Virtual Reality Laboratory was referred. At the beginning of the year, a new set of virtual reality peripherals called the Flock of Birds had arrived. However, research was not even started by the university on

the new equipment. So, details on the new equipment were research via the Internet and the manuals.

3.2.2 Requirement Analysis

Once the research of implementing the hardware is done, the next step is to analyze what are the requirements that are needed for this project. This requirement covers on how and what are needed to develop the weapon simulator. Below are the software requirements:

1. Microsoft Visual C++
2. OpenGL Libraries (Glut 3.7)
3. Adobe Photoshop 7.0
4. 3D Studio Max 5.0
5. 3D Exploration
6. Flock of Birds Libraries

Microsoft Visual C++ is used as the compiler for the whole program. This is because the programs will be in C++ format. C++ is chosen as it suits very well with OpenGL programming. The OpenGL libraries will be called and manipulated by C++ command. Although earlier computer graphics were done using C programming, there are some abilities that C cannot do such as the usage of class. As stated in the previous section, OpenGL in one of the most widely used libraries for the usage of visualization and computer graphics. VRML was also considered to be used in the project. However, as OpenGL had more capabilities than VRML, VRML was put aside. As OpenGL programming is mainly based on C++ programming, familiarizing the way of coding and commands to manipulate the OpenGL graphic library was not a hassle. This is added with the abundant source of OpenGL and C++ examples in the Internet. Adobe Photoshop 7.0 is as the primary image editor. Other than the usual coloring and picture editing functions, the ability of creating pictures with alpha channels or Targa (TGA) image formats is very useful. Alpha channels can be manipulated in OpenGL programming to create masking techniques, where surfaces can be seen transparent. These and more abilities of the simulator will be discussed. Vertices and shapes can be created using OpenGL commands solely. However the task is not easy and efficient. And because of that, 3D Studio Max 5.0 is used as the 3D editor. Other softwares that are available in the market for 3D editing are Maya, Milkshape and Lightwave. However I was well-versed in using 3D Studio Max. Using this software, other than making the modeling easy, the texturing task was also simplified. The usage of 3D Studio Max as the 3D editor was also based on the ability of the simulator to use *.3ds files. *.3ds files are models that can be read by the OpenGL compiler thus rendered onto the screen. 3D Studio Max 5.0 is quite heavy to be run simultaneously with other programs and system restart can occur due to the heavy processing load of the CPU. So, another software was used as the viewer for the 3D models. The software used was 3D Exploration. Although being used only as a viewer, the software is also capable to add lighting, reassign textures and export into several formats of 3D models. Another feature that is praised about 3D Exploration is the way it renders a model and its texture. This because, 3D Explorer renders models the same way how a model is rendered in OpenGL. Sometimes after the modeling process is completed, the finished 3D models cannot be seen properly on the render scene, for example missing textures, following the wrong axis and not showing the right surfaces. Based on this theory, every model is checked first using 3D Explorer before it is called in to the render scene of the simulator. The Flock of Birds library is used for the integration with the virtual reality peripherals. However, the utilization of the equipment may not be put to the maximum as there in minimal knowledge in programming them.

For the hardware requirements, the simulator needs a high speed processor, efficient amount of RAM and most importantly, a high speed graphics card. I have made the assumption that the machine that created the simulator is the minimal requirement for the simulator to operate. Below are the hardware requirements for the simulator:

1. AMD AthlonXP 2.0 GHz processor
2. MSI Motherboard with onboard sound
3. 256 DDR RAM

4. 64MB GeForce3 graphics card
5. Gyro Mouse
6. Joystick
7. Keyboard

The first four hardware used for the project is very important for the operation of the simulator. As the simulator and other similar C++ and graphics programs have to make so many calculations in few nanoseconds, fast processors and graphic cards are indeed needed. A processor with the speed of over 1.6 GHz is quite adequate to run the simulator. The choice of graphic cards is based on the onboard RAM of the graphics card. The graphics card that I am using, GeForce3 is manufactured by NVIDIA Corporation. Having a RAM range of 64 megabytes, the rendering of models and polygons within the render scene of the simulator are quite smooth. The final three hardware listed are used as the navigation tool to manipulate movement and action within the 3D environment. An ordinary mouse points to the screen based on its position on a flat plane. Being a flat plane, there are only two axis, x-axis and y-axis. However, a Gyro Mouse is a mouse that has another additional axis, the z-axis. The Gyro Mouse movement is not based on a flat plane, but based on the elevation of its position. This means that we can move the mouse around in the air just point the Gyro Mouse to a screen and the cursor will follow. Having this ability, the Gyro Mouse is used as the point of ammunition. Users will use the Gyro Mouse to target enemies on the render scene. Other than using the Gyro Mouse, a joystick can also be used to navigate and move the corsair of the simulator. The angle of the gun is changed by tilting the joystick either to the front or back. The ability of several joysticks in the market to support vibrating is another added bonus to add to the realism of the simulator. Lastly, the keyboard is the last resort for the navigation of the simulator. Although the realism effect is not that outstanding by using this peripheral, however it can also be used in early usage of the system. Meaning that it can be used to make the users familiar with the system, before they use it with the complete settings.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Physics laws and Kinematics equations

- $v = v_0 + at$
- $x = x_0 + v_0t + ½ at^2$
- $v^2 = v_0^2 + 2a(x - x_0)$
- $v = (v+v0)/2$
- $F = ma$
- $a = (v-v_0)/t$

These are the basic physics laws and kinematics equations that were studied earlier in the preliminary design phase. The objective then was to familiarize with the basics of physics laws. With these formulas, objects were animated in the virtual environment based on calculations.

However, with more research done within the subject, another formula was found. The formula was based on the motion of an object with the presence of gravity.

- $x = (v \cos \alpha)\, t$
- $y = (v \sin \alpha)\, t - gt^2/2$

x and y are the coordinate of the ammunition within the environment and α is the heading or angle of weapon based on the ground. The above formula is actually simplified from the earlier formulas. Being simplified, it will also save time for the computer to calculate and execute the trajectory. Using the above formula, most of the values can be set, rather than the earlier formulas where the values are progressively changing.

4.2 The Usage of *.3ds Files and Loaders

During the earlier experience with OpenGL, the objects such as the polygons and surfaces were drawn manually. Meaning that if a car is to be drawn, shapes such as cubes and icosahedrons had to be used for the body and tires. Using this method, details in the shape such as curves cannot be modified freely as the shapes are taken as it is and just assembled them together via rotation and translation commands. Another method is by drawing each vertex one by one. This is also a hassle as each coordinate has to plot one by one. In other words, it is like connecting the dots. There are other 3D model formats such as *.max, *.3ds, *.wrl that are used widely. Building the 3D models is quite easy, but the real challenge is to draw or simulate the 3D objects onto the environment. As the research is done mainly through the internet, a method called a 3ds Loader was found. A 3ds Loader is actually lines of coding that can read polygons stored in a 3ds file. 3ds files are actually 3 dimensional files developed by the developers of 3D Studio Max. So rather than saving the modeled objects to *.max files, the object is then exported as a *.3ds file. To put the object onto the environment, the *.3ds file is called by the 3ds Loader and thus the object appears.

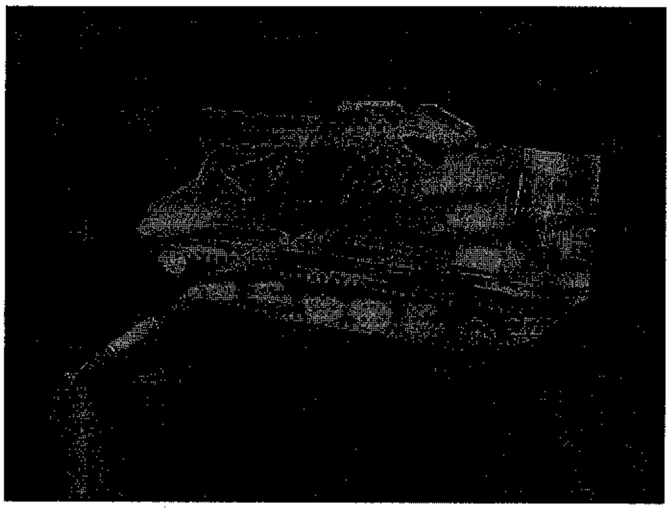


Figure 4.2(a): Sample of a *.3ds file

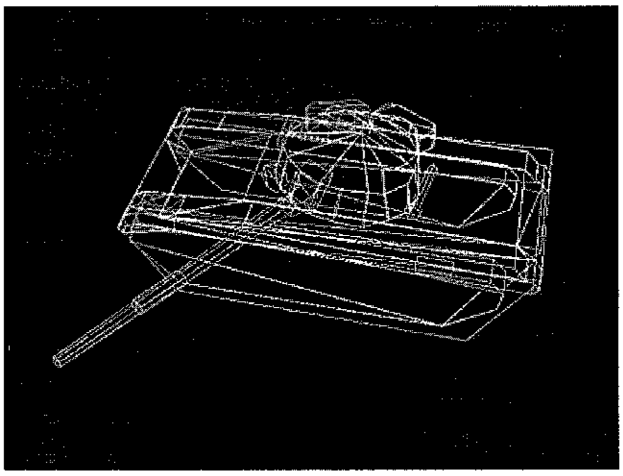


Figure 4.2(b): Sample of *.3ds file in wireframe mode

4.3 From 3D objects to C++ coding and Display List

Other than using 3ds Loaders, there is also another method of drawing a 3D object onto the environment. By using a software called 3D Exploration, the 3D objects are saved as C++ coding. The software actually scans the 3D object and reads every vertex, line and polygon. It also includes the texture of the object. Then every aspect of the object is translated as lines of coding. The end product of the transformation are lines of coding in C++. However, work still need to be done before the object can be seen in the virtual environment. Some lines of coding need to be edited such the directory for the texture, the name of the object within the coding itself must be redefined and also lighting properties of the virtual environment. Usually, this method is used with the Display List function. By using the Display List function, the commands and arguments are stored for subsequent execution. Numbers or names are assigned to each object or in case, each C++ file. The calling of the objects within the Display List is then done by calling pointers or arrays.

4.4 3ds Loader versus Display List

As there were two methods of drawing the 3D objects, an experiment was done to choose the perfect method for developing this project. The target of this experiment is to find which method is faster and does not waste frames per second speed based on the number of polygons drawn. The objects that were tested on were buildings that complete a whole city. Each building is saved in both in *.3ds format and in C++ format. Then, both of them are loaded onto the virtual environment separately, and the frame per second of the program is taken. Below are the results:

No of Objects	Polygons/Object	Accumulated Polygons	FPS(Display List)	FPS(Loader)
1	none	0	546	543
2	10	10	545	542
3	10	20	543	539
4	10	30	540	538
5	10	40	534	537
6	10	50	533	535
7	10	60	531	528
8	10	70	528	526
9	10	80	526	525
10	10	90	523	524
11	10	100	521	521
12	10	110	520	520
13	10	120	519	520
14	10	130	517	520
15	10	140	517	519
16	10	150	514	517
17	10	160	512	516
18	10	170	510	514
19	22	192	505	507
20	22	214	503	503
21	22	236	500	500
22	22	258	496	498
23	22	280	493	496
24	22	302	488	495
25	28	330	485	491
26	28	358	481	489
27	32	390	475	484
28	32	422	470	480
29	32	454	465	476
30	34	488	461	473
31	34	522	457	468
32	34	556	453	466

33	34	590	450	457
34	120	710	438	447
35	120	830	430	437
36	160	990	414	432
37	160	1150	405	420
38	176	1326	394	406
39	176	1502	382	395
40	768	2270	354	349
41	768	3038	329	313
42	768	3806	308	283
43	768	4574	289	260
44	768	5343	272	239
45	768	6110	257	221
46	910	7020	244	203
47	1528	8548	227	182
48	1620	10168	203	163
49	2528	12696	182	140
50	2626	15322	164	122
51	2892	18214	151	106
52	3520	21734	135	91
53	3748	25482	124	80
54	3900	29380	120	71
55	3900	33282	109	63
56	5139	38421	91	56
57	6163	44584	81	49
58	6195	50779	73	44
59	6216	56995	67	39
60	9412	66407	64	34
61	9936	76343	54	30
62	10942	87285	47	27
63	11824	99109	42	24
64	12608	111717	38	20
65	12812	124529	33	19
66	49206	173735	Unable to draw	14

Table 4.4: Table of performance between 3ds Loader and Display List

During the later stage of the experiment when the number of polygons was 173735, the Display List method could not be used. This is because the lines of coding were too long for the compiler to handle. So, the experiment was limited to 66 objects with 173735 polygons. At the beginning, the Display List method is slightly faster than the 3ds Loader method. However, when the number of polygons were beyond 120, the 3ds Loader method starts to take the lead. The drop in frames per second was due to the time it took to process each C++ file. As for the 3ds Loader, only a single C++ file is needed and the drawing of the *.3ds object is also faster. A chart is produced to observe the performance of the two methods.

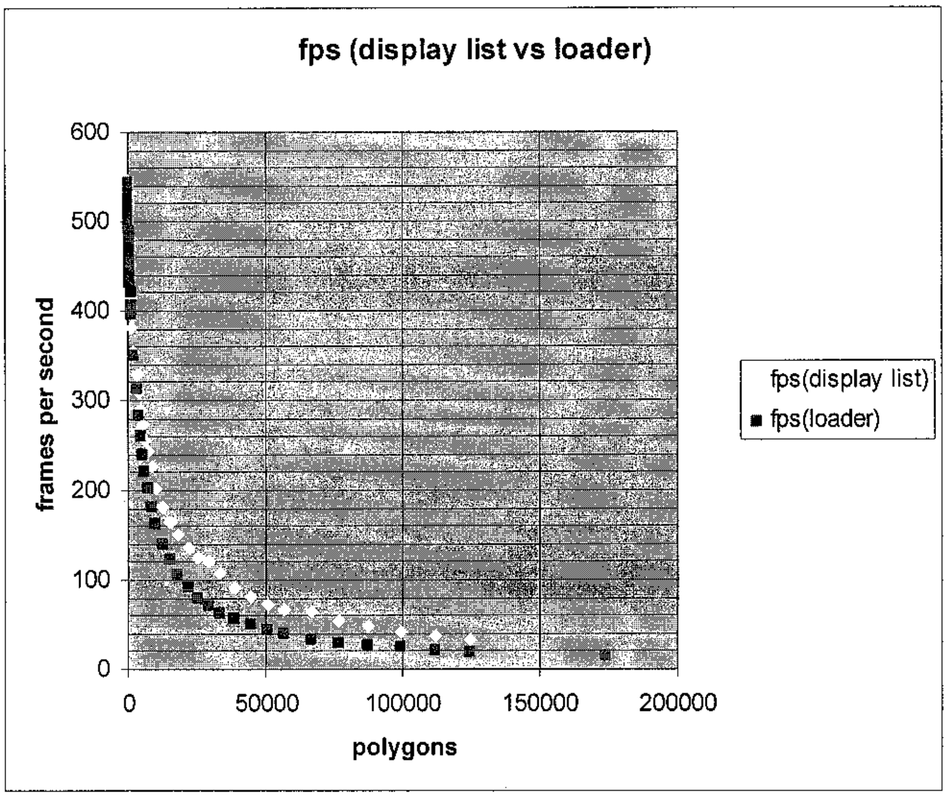


Figure 4.4: Performance between 3ds Loader and Display List

4.5 The Usage of Heightmaps

Heightmaps are vertices or surfaces that are rendered based on picture format named Raw. Raw picture format are in gray scale, where the colors ranges from the mixture of white and black colors. Every grayscale color has its own value, where black is 0 and white is 255. Based on these values, the heightmap algorithm will read the values of the image. Each value is then plotted and rendered onto the render scene. Heightmaps are usually used for landscapes and terrain rendering purposes as the gray scale can depict the contours of hills and lands quite easily and smoothly.

Figure 4.5(a): Sample of Raw file

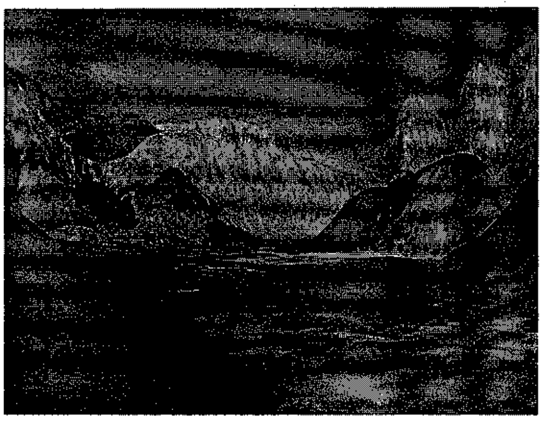


Figure 4.5(b): Terrain based on sample Raw file

4.6 Separating the Calculations from the Render Scene

Rendering a scene or picture itself is a calculating process. Each vertices and points are calculated and then rendered to the scene. This is known as the drawing or rendering process. However, there are situations where movements or changes are to be seen on the render scene. These movements are usually based on physics, physical movements and such. These movements add more calculations to the program. However, the calculations are more to the mathematical side. These calculations take the process time in the rendering process. This is because every time the scene is to be drawn, the calculation needs to be completed. A way of making the program go faster is by separating the calculations with the rendering programming. The calculations should be placed in a separate function that only handles mathematical or algorithms. The function should then only pass the value of the final calculation to the function that handles the rendering.

4.7 The Usage of Alpha Channels

Targa picture formats or TGA have in them a fourth channel which is the alpha channel. Usually, picture formats have 3 channels which are the red, green and blue channels (RGB). The fourth channel in the TGA image format is called the Alpha channel. This channel if we see it in any ordinary picture viewer would seem visible. However, the alpha channel is actually transparent when it used with the blending function in OpenGL. Other than viewing it in the OpenGL environment, the effect of the alpha channel can also be seen by using Adobe Photoshop7.0. This software is also able to edit the alpha channel in the TGA picture. Based on the ability of the TGA image, the target of the simulator was made based on that concept.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 CHALLENGES

The main challenge that was encountered during the project was actually getting use to the glut3.7 libraries. In order to get the desired output, research and tutorials were done again and again. Troubleshooting the coding was also a big task as mistakes need to be detected line by line.

The author also needed to brush up on his physics as it was the main ingredient in the trajectory program. Physics books and the internet were the main reference for the author. Calculus was also being taken account as it helped the author to understand formulas clearly.

The last challenge of this project was to cope up with the time. There were a lot of iteration processes that occurred along the development phase. Sometimes, the author had to struggle in order to maintain the work / report that need to be submitted in time. The cause of this problem was because of rapid iteration process during the development phase. Although sometimes the time required was not enough, the author had managed to submit all the works / reports on time.

5.2 RECOMMENDATIONS

The main recommendation that would be a very big help to the project would be in the hardware department. The usage of the joystick, keyboard and mouse lacks the realism aspect of a simulator. It is recommended that a mouse called a Gyro-Mouse to be used. The regular mouse that we use operates on a flat surface, meaning that it interacts within a two-dimensional environment. However, the Gyro-Mouse operates in a three-dimensional environment. The mouse can actually sense its elevation level based on the nearest floor surface.

On the software side, the targets should be substituted with other objects. As the objects currently used are quite simple, other objects such as vehicles, organic objects or even human models can be used as the target. Movement of the targets can also add more realism in the simulator. Other than that, it is also to help the user to train on moving target, thus adding more value to the simulator.

5.3 CONCLUSION

The Weapon Simulator Project is an adventure into computer graphics and physics. With the integration between the two, it is hoped that another environment similar to what we are in can be produced. Within the project's scope, the environment shall be used for weapons training. The system would bring many benefits to the present curriculum of teaching new soldiers in using new weapons. Other than the safety and cost factor, the project can also be enhanced into other simulators such as a flight simulator, tank simulator and such. However, more research and work are needed to achieve those. In all, this project should benefit the nation's infantry where it can be used in the preliminary phase of weapons training.

REFERENCES

- Dave Eberly. 15 December 2001 < http:/www.gametutorials.com>

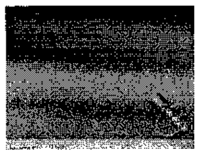
- Chris Hecker. 6 April 2000 <http://delivery.acm.org>

- Peter Eberhard and Shoushan Jiang, 2000 *Mathematical and Computer Modeling of Dynamics Systems Vol.6*, No.3, pp. 309-322

- Douglas C. Giancoli 1998, *Physics - 5th Edition*, New Jersey, Prentice Hall

- James Stewart 1997, *Calculus – 4th Edition*, New York, Brooks/Cole Publishing

APPENDICES



Let us consider the motion of free body in the presence of the gravitational forces. If the cannon is located at the point with coordinates (0, 0, 0), then the shell fired will move by the trajectory, which can be described by the following equations

$$X = (v \cos\varphi)\, t$$
$$Y = (v \sin\varphi)\, t - gt^2/2,$$

where v is the initial velocity of the shell along the gun tube, φ is the angle between the gun tube and the horizon (X-axis), t is time, g is the acceleration of the free fall in the gravitational field of the Earth. Substituting t from the first equation into the second one we can find the expression for the trajectory of the shell:

$$Y = X\, \mathrm{tg}\varphi - (g/2v^2)(1 + \mathrm{tg}^2\varphi)\, X^2$$

This means that the trajectory is of parabolic form. We can find from this equation the maximal range of a shot X_{max} (when Y=0) and the maximal height of the trajectory Y_{max} (when the first derivative of Y equals to 0):

$$X_{max} = v^2\sin(2\varphi)/g$$
$$Y_{max} = v^2\sin^2\varphi/2g$$

We can see from the first of these two equations that the maximal range of the shot is achieved when the angle φ equals to 45 degrees. The maximal height of the flight is achieved when we shoot vertically. Video-animation shows the cases of the shooting at angles 30, 45, and 70 degrees.