ENTITY MOTION MANAGEMENT IN COMPLEX SIMULATION ENVIRONMENTS USING IMAGE GENERATORS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BURAK ÇİFLİKLİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2008

Approval of the thesis:

ENTITY MOTION MANAGEMENT IN COMPLEX SIMULATION ENVIRONMENTS USING IMAGE GENERATORS

submitted by **BURAK ÇİFLİKLİ** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen ________________
Dean, Gradute School of **Natural and Applied Sciences**

Prof. Dr. Volkan Atalay ________________
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Veysi İşler ________________
Supervisor, **Computer Engineering Dept., METU**

Examining Committee Members:

Assoc. Prof. Dr. Nihan Kesim Çiçekli
Computer Engineering Dept., METU ________________

Assoc. Prof. Dr. Veysi İşler
Computer Engineering Dept., METU ________________

Dr. Atilla Özgit
Computer Engineering Dept., METU ________________

Asst. Prof. Dr. Kayhan İmre
Computer Engineering Dept., Hacettepe University ________________

Assoc. Prof. Dr. Kürşat Çağıltay
Computer Education and Instructional Technology Dept., METU ________________

Date: ________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: BURAK ÇİFLİKLİ

Signature :

ABSTRACT

ENTITY MOTION MANAGEMENT IN COMPLEX SIMULATION ENVIRONMENTS
USING IMAGE GENERATORS

Çiflikli, Burak
M.S., Department of Computer Engineering
Supervisor : Assoc. Prof. Dr. Veysi İşler

September 2008, 75 pages

Image generator host is the interface of the host computer system of a flight simulator to its image generator. Image generator host, updates positions of the entities by sending operational codes to the image generator. Positional data of the entities is pipelined by tactic interface of the simulator at host update rate. A network jitter, latency, packet loss or inadequate bandwidth may disturb the smoothness of this pipelined entity information packets. This study presents an algorithm for the host system of a flight simulator, intending to minimize model flickering in the image generator display output.

Keywords: Image Generators, Host Controllers, Flight Simulation, Perception, Opcode

ÖZ

GÖRÜNTÜ ÜRETECİ BİLEŞENİ KULLANILAN KARMAŞIK SİMÜLASYON ORTAMLARINDA MODEL HAREKET YÖNETİMİ

Çiflikli, Burak
Yüksek Lisans, Bilgisayar Mühendisliği
Tez Yöneticisi : Doç. Dr. Veysi İşler

Eylül 2008, 75 sayfa

Görüntü üreteci arayüz bilgisayarı, simülatörün görüntü üreteci ile olan arayüzüdür. Görüntü üreteci arayüz bilgisayarı, simülasyon ortamında bulunan hedeflerin konumlarını görüntü üretecine işlemsel kodlar göndererek bildirmekle yükümlüdür. Hedef konum bilgileri taktik arayüz sisteminden alınır. Ağ gecikmeleri, paket kaybı ya da yetersiz bant genişliği gibi sebeplerle hedef hareketlerinde bozulmalar gözlemlenebilir. Bu çalışmada, görüntü üreteci bileşeni kullanılan simülasyonlarda, üç boyutlu model hareketindeki bozulmaları en aza indirgeyecek ve host sisteminde çalışacak bir algoritma geliştirilmesi amaçlanmıştır.

Anahtar Kelimeler: Görüntü Üreteci, Anasistem Denetleyici, Uçuş Simülasyonu, Algı, İşlemsel Kod

To my family

ACKNOWLEDGMENTS

I am honored to present my special thanks and deepest gratitude to my supervisor Assoc. Prof. Dr. Veysi İşler for all his guidance and support during this study.

I would like to thank to HAVELSAN for providing me time and resources whenever I needed.

Finally, I would like to thank my family for all their life-long support.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Rationale of the Study

Flight simulation systems have been developed over the last few years especially in defense industry. A flight simulator is a system that simulates the experience of flight with a real aircraft.

In a flight simulation, 3D moving models (like planes or ships) are also used to make the simulation environment look more realistic. Presence sensed by the simulation users can not be directly linked to a specific type of technology; it is a product of the mind [1]. Models with high-resolution textures and more polygons make a simulation session more realistic but smooth movement of 3D models should also be concerned. A flickering model decreases presence in the virtual environment.

In recent years, separate image generator components have been used for visual systems in flight simulators. Image generator host system bridges the image generator component to the rest of the simulator system (Figure 1.1). Host controls the image generator via interface instructions called operational codes (opcodes). Rendering is the task of image generators; host only makes information updates like positional updates for 3D models or weather condition changes.

During 3D model management of the simulation environment, host does not deal with model geometry, textures, level of detail management, etc. These are the tasks of the image generator. Host only makes updates on the model's information like its position and orientation, switch numbers, sub model orientation, etc. Host may be seen like a simple interface system just using some get and set functions. To provide smooth movement of the entities, just pipelining positional data to image generator is generally not enough. In this study, the problems causing 3D model flickering are stated and an image generator host system design for effective entity motion management to minimize model flickering is presented.

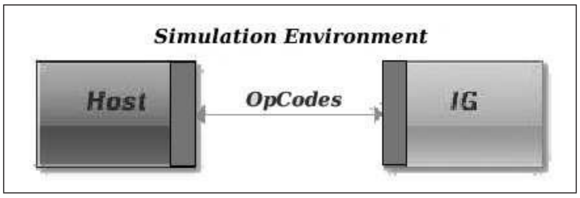

Figure 1.1: Communication between Host and Image Generator

1.2 System Overview

A flight simulator is illustrated in Figure 1.2. Simulators are usually concerned with following subsystems; cockpit, host, image generator and displays.

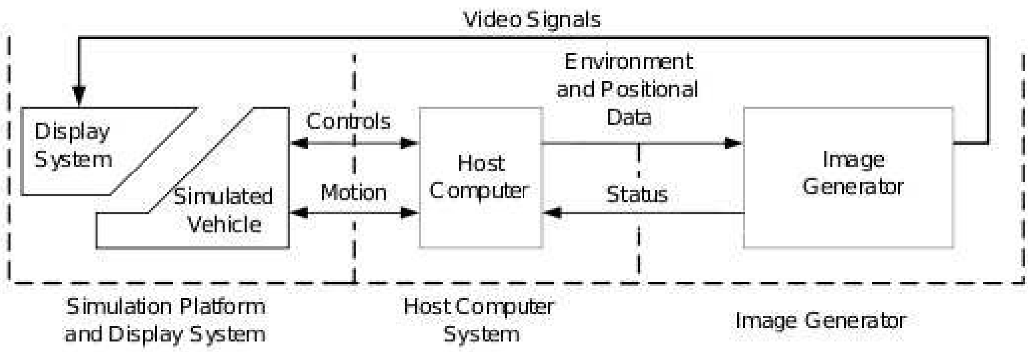

Figure 1.2: Simulator Block Diagram

1.2.1 Cockpit

Cockpit subsystem is one-to-one replica of the actual simulated aircraft. It can have a motion base or vibration platform used to simulate aircraft motion.

1.2.2 Image Generator

The image generator component receives information updates from the host computer and uses these updates to render the simulation environment. The output image is updated at a rate known as the image generator update rate. These images are then sent to the display system as video signals or digital image data. Image generator sends back status data to the host computer over the host interface. This information primarily consists of collision detection (CD) status, results of inquiries for range information, line of sight (LOS), and height above terrain (HAT) of a moving model.

1.2.3 Displays

The display system receives the imagery that is rendered by the image generator and displays the data on a display device. The display device can be a CRT display, a projector, or other similar display device. The display refresh rate is not necessarily the same as the image generator update rate, although it often is. If the refresh rate is faster than the update rate, a type of visual image artifact consisting of unwanted multiple images might occur.

1.2.4 Host

Host receives control and motion input from cockpit and sends environmental and positional data to the image generator. Image generator sends status information back to host. Host subsystem is not a part of the image generator. All communication between the host and the image generator passes through the host interface of the image generator.

Primarily, the image generator receives positional commands from the host at a constant rate known as the host update rate. This rate is often specified at 30 or 60 Hz but could be other values. The position and orientation of the own ship, along with any other companion vehicles or moving models, is sent to the image generator at the host update rate.The host system of a flight simulator is illustrated in Figure 1.3. Image generator host subsystem is the interface of the host computer system to the image generator.

In a distributed simulation environment, all entities in the environment inform all other entities of their status and actions through the transmission of standard information packets of the environment architecture (DIS/HLA). All events like positional updates or state changes caused by an individual simulator are broadcasted and available to all other simulators. A simulator transmits only changes in the state of the entity it simulate. Continuous activities are transmitted at a certain update rates. It is the receiving simulator's responsibility to determine the effects on its own state caused by the event. The subsystem responsible of these tactical transmissions is called tactic interface of the simulator.

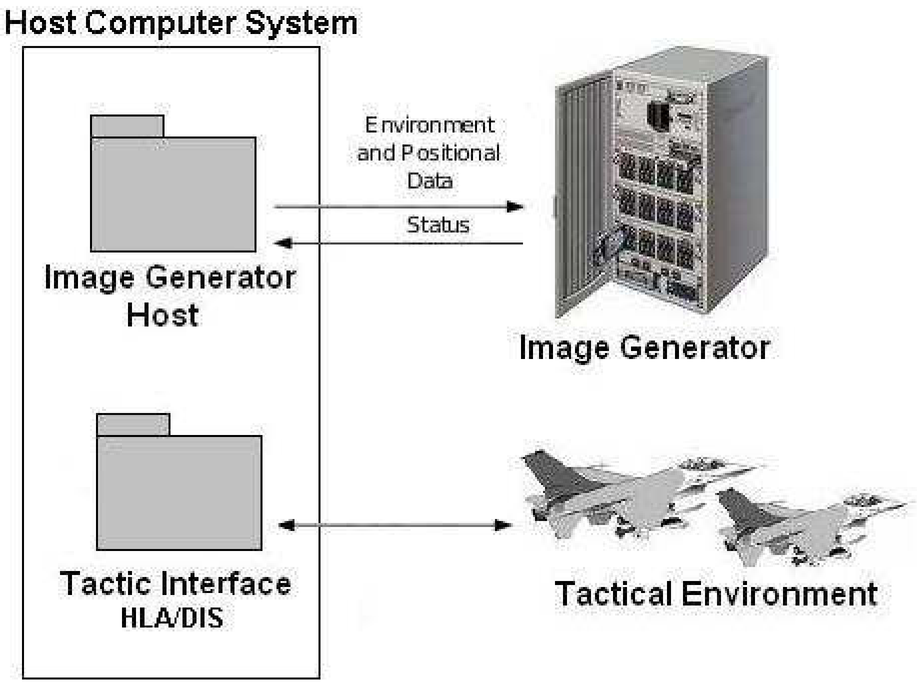

Figure 1.3: Host System

1.3 Image Generator Interface

Host controls the image generator via interface instructions called operational codes (op-codes). Image generators with different interfaces make replacement and integration very difficult. The Common Image Generator Interface (CIGI) is promoted by Simulation Interoperability Standards Organization (SISO) in 2006.CIGI is a standardized interface between a real-time simulator host and an image generator. CIGI is an open interface offered to promote commonality in the visual simulation industry [4].

1.3.1 Message Synchronization

CIGI supports both synchronous and asynchronous operation. Each of these modes is described below.

1.3.1.1 Asynchronous Operation

During asynchronous operation, host can send opcodes to the image generator at any time. Meanwhile, image generator maintains its own frame rate. During each frame, image generator first checks its buffer for incoming CIGI messages. It then updates the scene graph based on the contents of the CIGI messages. Finally, the IG renders the scene.

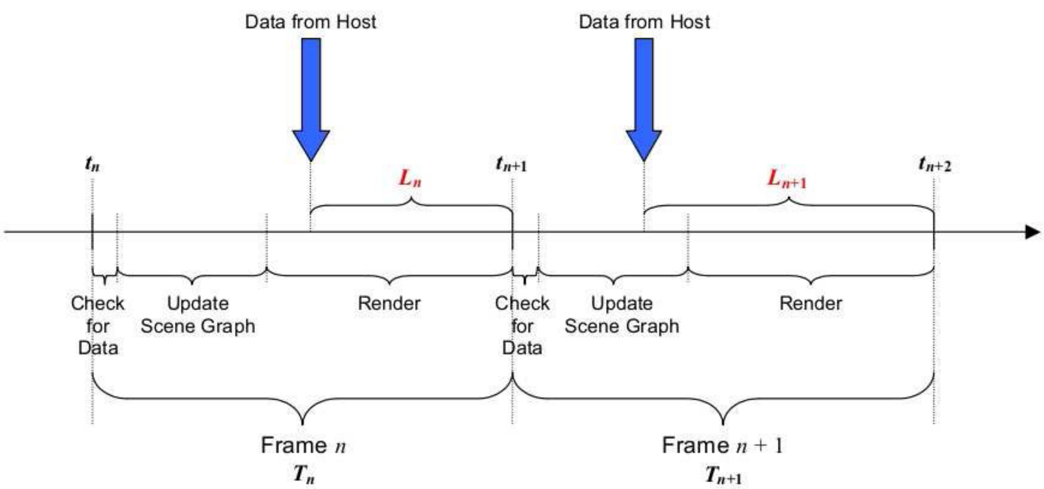

Figure 1.4: Asynchronous Messaging. Opcodes sent by the host are processed with a latency up to one frame

Host might send a message at any point during the image generator's frame, positional and other state changes might not be applied until the beginning of the next frame. This introduces a latency of up to almost one additional frame (Figure 1.4).

1.3.1.2 Synchronous Operation

During synchronous operation, the IG sends a start-of-frame (SOF) message to the host to signal the beginning of each frame. This message dictates the timing of data transfers between the IG and Host (Figure 1.5). Host immediately responds to each SOF message with its own message containing entity positions and orientations, component states, and other data describing changes to the scene during the previous frame. Host then begins its next computational cycle, while the IG updates and renders the scene.

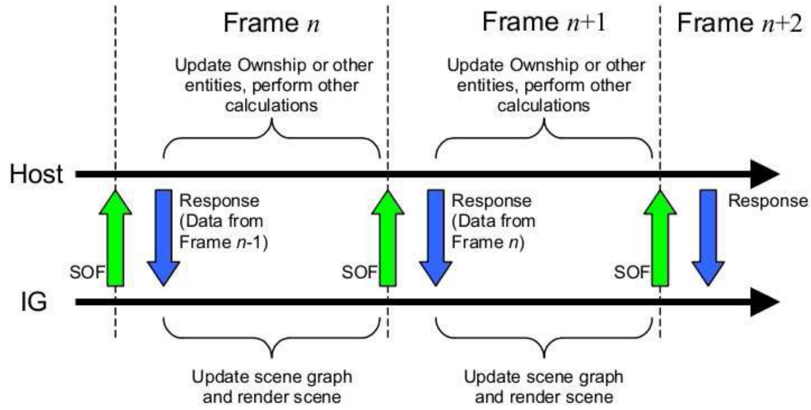

Figure 1.5: Synchronous Messaging

Due to bandwidth limitations or transport delay, IG may not be able to receive a response in time to finish rendering the scene before the start of the next frame. To alleviate this situation, a time offset can be introduced so that the IG sends each SOF message slightly before the beginning of the next IG frame. This technique allows data to arrive from the Host at such a time as to allow the IG its entire frame time for computations and rendering. Because the transport delay may vary from frame to frame, this offset can be adjusted to allow for worst-case network loads (Figure 1.6).

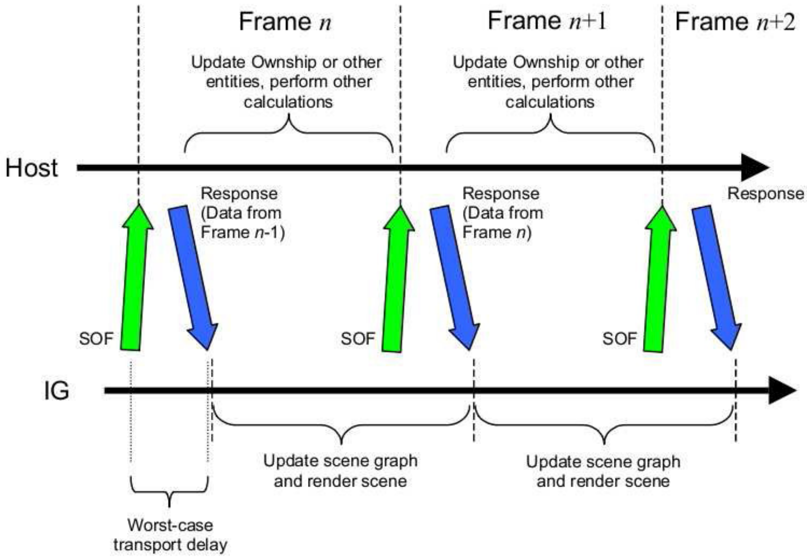

Figure 1.6: Synchronous Messaging

1.3.2 Positional Update

Entity Control opcode is used to update the position of a 3D model in the simulation environment. Packet structure is given in Figure 1.7 and parameter definitions are given in Appendix G. Each entity is identified by a unique identifier called the Entity ID. When the Host sends an Entity Control packet, image generator sets the state of the entity object corresponding to the value of the Entity ID parameter.

In synchronous messaging, the state of the entity is set by the incoming Entity Control packet. In asynchronous messaging, the IG may extrapolate positional data each frame by using the amount of time that has elapsed since the opcode is received. By determining entities' velocities and accelerations from prior frames, the IG may calculate the probable positions of those entities during the current frame.

For smooth entity motion, tactic interface should pipeline positional information of an entity at host update rate (mostly 30 or 60 Hz). In a simulation frame, tactic interface may not able to gather and pipeline information packets of every entity in the tactical environment. A network jitter, latency, packet loss or inadequate bandwidth may result in such a missing entity information situation. Also some tactical environment management systems broadcast positional updates or state changes at lower rates not to cause network traffic. In such cases, the position and orientation of the missing entity should be predicted and an Entity Control opcode should be send to image generator.

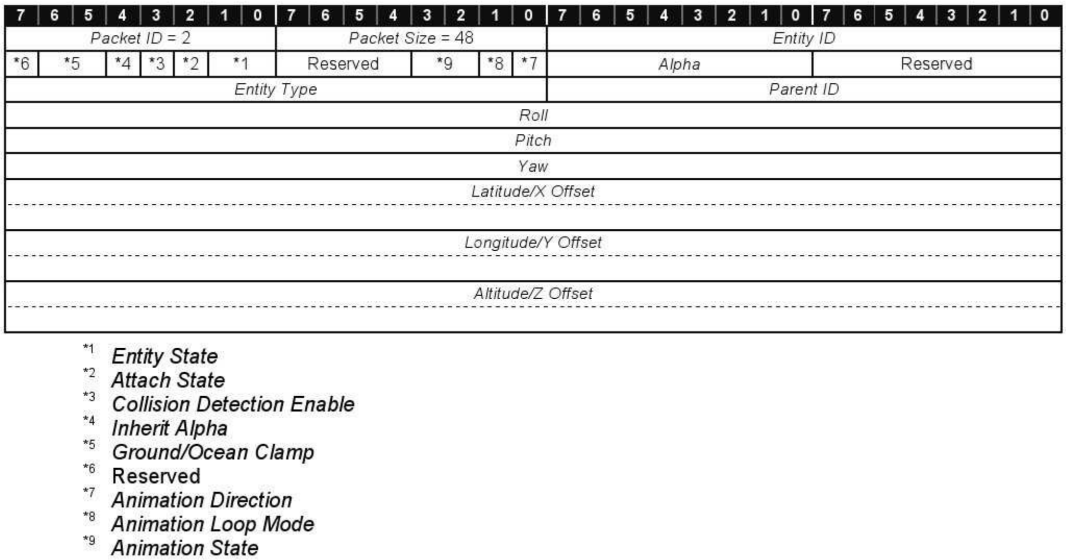

Figure 1.7: Entity Control Packet Structure

Ethernet bandwidth of a standard PC interface is enough to send huge number of Entity Control opcodes in real time. Number of bytes that can be sent in one second considering we have a gigabit Ethernet interface is:

$$\frac{10^9}{8} = 1.25 \times 10^8 bytes \tag{1.1}$$

Number of bytes that can be sent in one frame at 60 Hz frame rate is:

$$\frac{1.25}{60} \times 10^8 \approx 2 \times 10^6 bytes \tag{1.2}$$

Number of models whose position can be updated in one frame at 60Hz frame rate is:

$$\frac{2 \times 10^6}{50} = 40000 \tag{1.3}$$

Where;
Size of Entity Control opcode is $\approx$ 50 bytes

It is possible to forward any realistic number of entity positional update opcodes to image generator. But it is not possible for an image generator to process all of the forwarded opcodes. Today's powerful image generators can handle nearly 50 to 100 entity positional update opcodes if there are no other tasks like mission functions or rendering. A culling algorithm within the Host should be applied to reduce the number of entities whose position will be updated.

1.4 Road Map

Chapter 2 provides background information on dead reckoning & smoothing of entities in the simulation environment, level of detail management of 3D models, entity culling, perception based models and measuring presence.
Chapter 3 presents an algorithm to eliminate flickering from the visual scene.
Chapter 4 presents application of the proposed algorithm to a simple flight simulator.
Chapter 5 presents the performance testing approach and discusses the results of the tests.
Chapter 6 concludes the study by summarizing the overall study and results, and providing the further improvement opportunities.
Lastly the questionnaires, 3D models used in the tested environment and complete results of the questionnaires are provided in the Appendices.

CHAPTER 2

RELATED WORK

This chapter presents foundations of virtual reality related to the following areas in the study: ensuring smooth entity motion, level of detail management, culling, perception based models and measuring presence.

2.1 Smooth Entity Motion

Tactic interface receives standard information packets of the environment architecture (DIS/HLA) which gives position updates or state changes of companion simulators or other entities. These information packets are forwarded to image generator host after being processed by tactic interface. For smooth entity motion, tactic interface should pipeline this information at host update rate (mostly 30 or 60 Hz). In a simulation frame, tactic interface may not able to gather and pipeline information packets of every entity in the tactical environment. A network jitter, latency, packet loss or inadequate bandwidth may result in such a missing entity information situation. Also some tactical environment management systems broadcast position updates or state changes at lower rates not to cause network traffic. In such cases, the position and orientation of the missing entity should be predicted. In 1995, IEEE Standard for Distributed Interactive Simulation Application Protocols [2] is published and in this standard, a physically-based prediction algorithm called dead reckoning is presented. The motion of an entity is estimated as:

$$P = P_0 + V_0\Delta t + \frac{1}{2}A\Delta t^2 \qquad (2.1)$$

Where;

P_0 *is the position vector in world coordinates at initial time*

V_0 *is the velocity vector in world coordinates at initial time*

A *is acceleration vector*

Δt *is time increment for dead reckoning step*

This first order extrapolation is generally used for orientation. For position estimation dead reckoning is expended to second order [3]. Also for more accurate predictions, the two most recent positions are used rather than using just one. The order and step relationship is described in Table 2.1.

Table 2.1: Position Estimation Equations

	One-Step	Two-Step
1^{st} *order*	$x_t = x_{t'} + v_{t'}T$	$x_t = x_{t'} + \frac{x_{t'} - x_{t''}}{t' - t''}T$
2^{nd} *order*	$x_t = x_{t'} + v_{t'}T + \frac{1}{2}a_{t'}T^2$	$x_t = x_{t'} + v_{t'}T + \frac{1}{2}\frac{v_{t'} - v_{t''}}{t' - t''}T^2$

For smooth entity motion, predicting missing information is a very important step but it is not enough. When an information packet is pipelined by tactical interface, it can not be directly used by image generator host. This causes jumps and flickering in the movement of the entity on the rendered output. A correction algorithm should be applied to the incoming data [2]. This correction algorithm is generally called smoothing. The smoothing equation is as follows:

$$x_i = x_0 + (x_f - x_0)\frac{i}{p} \tag{2.2}$$

Where;

x_i *is* i^{th} *smoothing position*

i *is integer from* i *to* p

p *is number of smoothing points*

x_0 *is starting position of smoothing* (*i.e the position before update*)

x_f *is final position*

2.2 Entity Culling

It is possible to forward any realistic number of entity position update opcode to image generator. But it is not possible for image generator to process all of them. Today's powerful image generators can handle nearly 50 to 100 entity position update opcodes if there are no other assigned tasks like mission functions or database and model rendering. We should use a culling algorithm to reduce the number of entities whose position will be updated. As it is not possible to eliminate an entity from the scene, a culling algorithm should adjust the updating frequency of the entities. To develop such an algorithm, level of detail and culling algorithms are surveyed.

2.2.1 LOD

A common method for optimizing the rendering of small and distant objects is multi-resolution modeling: the description of geometry and surface attributes such as color and texture at a variety of scales [5]. Appropriate level of details for models within a virtual environment can be chosen with an algorithm based on size. Such a decision mechanism is introduced by James H. Clark [6]. Clark defined each object as a tree hierarchy. The entire environment is itself an object and is represented as a rooted tree. There are two types of arcs; transformations and pointers to more detailed structures. Each non-terminal node represents a sufficient description of the object if it covers no more than some small area of the display; the arcs leading from the node point to more detailed objects which collectively define a more detailed version of the original object if its description is insufficient because it covers a larger area of the screen. The terminal nodes of the tree represent either polygons or surface patches (or other primitives) according to whether they are primitive elements of a faceted or a smooth object. Funkhouser and S'equin [7] formulated LOD switching task as a multiple choice knapsack problem (MCKP), which is known to be NP-hard, and used an approximation method to select the appropriate LOD for each object to be rendered. Mason and Blake [8] utilized a hierarchical level of detail approach and used a variation of the MCKP to select appropriate representations for every frame. The approximation method of their extended MCKP can be seen as a top down greedy traversal of the LOD hierarchy. In this hierarchy, objects have explicit and implicit representations. An object is explicitly represented if its currently selected representation is one of its impostors, and implicitly represented if it is represented by the explicit representations of its descendants. The explicit representations or impostors of an object are ordered by increasing rendering complexity, and are defined as lower LODs than any of its implicit representations.

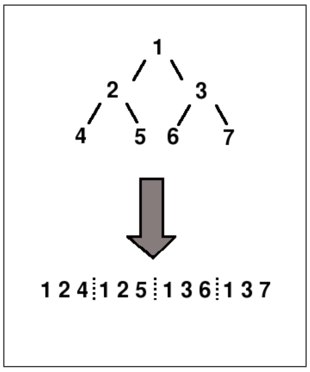

Figure 2.1: Transforming LOD hierarchy to equivalent non-hierarchical description

Their algorithm performs incremental steps until the rendering budget is violated. Within a step, the selection is replaced by the most precious successor node in the level of detail tree hierarchy. An inverse decremental step is also introduced, which replaces least precious successor nodes with their parent. Root node represents the entire scene and it is the initially selected representation. A node with in the tree is the low level representation of its children nodes (Figure 2.1).

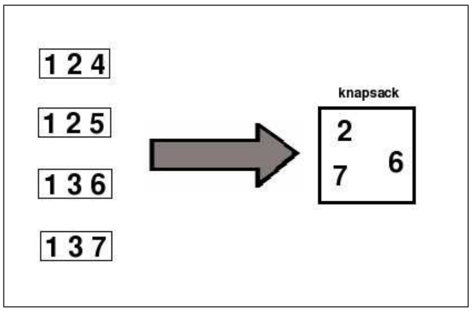

Figure 2.2: Schematic representation of the MCKP

This presents a way to transform a given hierarchical description to an equivalent constrained non-hierarchical one. So, the level of detail optimization problem for the hierarchical description is equivalent to a constrained version of the Multiple Choice Knapsack Problem (Figure 2.2).

In Zach, Mantler and Karners' implementation, LOD selection problem for both discrete and continuous variables are considered [9]. Their discrete LOD selection is based on Mason and Blakes work. For continues LOD selection, a variable x_i is used instead of a set of representations. x_i denotes the chosen resolution for rendering that object.

$$x_i \in [0, 1]$$

In their study, it is stated that the objects that are smooth and closer to each other have mostly identical resolutions. The objects are organized in a quad tree hierarchy. If some threshold Θ is not exceeded, n smooth objects are replaced by one object, namely x. This process continues until Θ is exceeded or the original object representations are still smaller than Θ.

2.2.2 Culling

There are three kinds of visibility culling that are mostly used in computer graphics; view frustum culling, back-face culling and occlusion culling [10]. These techniques avoid processing invisible portions of a scene by discarding polygons that are off-screen, oriented away from the viewer or occluded respectively.

Teller and Sequin [11] present an object space algorithm for fast architectural walk through system which divides a database into cells, roughly corresponding to rooms in a building. Cell-to-cell visibility can be computed in a pre-processing phase.

Grundhofer presented a non-conservative multi-pass approach [12] which uses low level of detail representations of the geometric models to decide the visibility for the higher resolution versions.

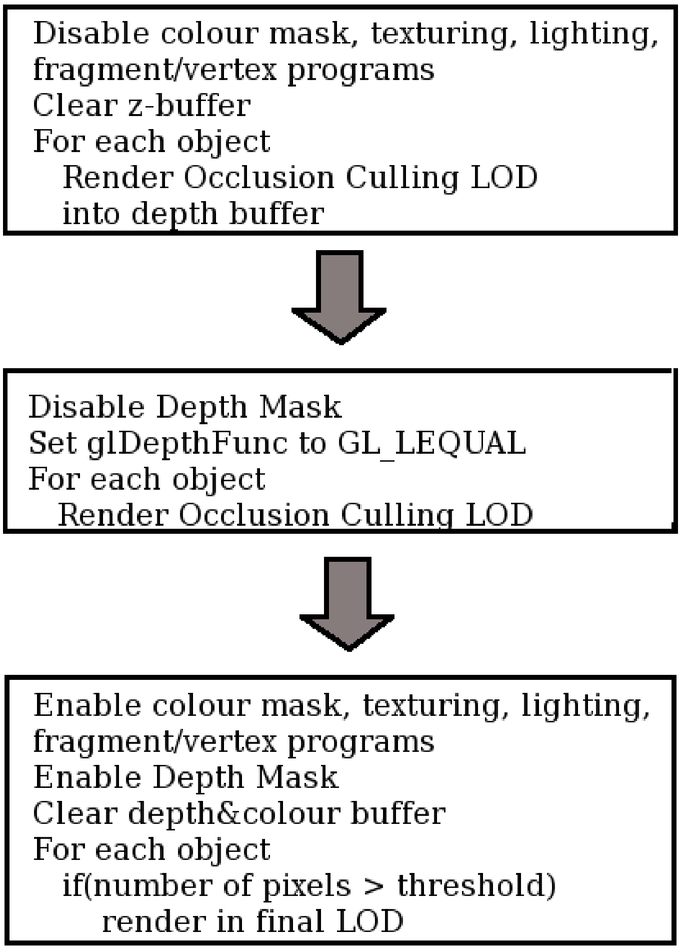

Figure 2.3: LOD-based multi pass occlusion culling

The algorithm consists of three rendering passes (Figure 2.3). An initial view frustum culling pass removes all objects outside the viewing frustum. The LOD pre-processing step computes the required LODs for each object. The next passes make use of the OpenGL occlusion culling extension which counts the number of visible pixels of geometric primitives. Occlusion query objects have to be generated and activated for each geometric object. After the geometry is drawn, the occlusion query has to be disabled and the number of visible pixels can be queried. In the first rendering pass the depth buffer is cleared and the objects are rendered with low levels of detail into the z-buffer with activated depth testing. During this pass, color buffer writes and rendering features like lighting, shading, texturing, and vertex/fragment shaders are disabled. After this pass the depth buffer contains the depth information for the whole scene. In the second pass all the objects are drawn again but now depth buffer writes are disabled as well. With each object now an occlusion query is send. Only finally visible pixels are counted, since the z-buffer of the first pass is used.
In the last pass color and depth buffer writes are enabled and the depth function is turned back to its default state. Before rendering an object with a higher LOD, which is selected based on a distance calculation, the result of the corresponding occlusion query is retrieved. If the query result is zero, no pixels of this object were visible during the second rendering pass and the object can be skipped in any case. Instead of skipping only objects with zero visible pixels, a threshold value can also be used.

2.2.3 Visual Perception

In order to optimize the rendering pipeline and increase image quality, it is essential to obtain an understanding of design constraints imposed by human visual system. Virtual environment design requirements and constraints should be developed by taking into consideration the abilities and limitations of human visual sensory.
The human eye can be modeled as a thin lens system (Figure **??**). Light rays from the object point enter the eye and are refracted by the lens through the image point I. The amount of refraction depends on the power of the lens. The lens of the human eye changes its power to focus the object point of interest at the retina. Object points located closer or further away will be out of focus and create a circle of confusion [13].
In 1999, Gerald Pitts and Daniel Cornell made an experiment to establish thresholds for pe-

ripheral vision based level of detail switching [14]. A reliable benchmark for viewer distraction and perception of unacceptable level of detail switches was tired to be established in this experiment. The participants were placed in an environment in which they were asked to focus on some central task while various objects in the background were changing level of detail. The assumption was that if the user's attention is taken from the focus task when a background object changes LOD, then the change in detail was too drastic. Visually, the viewer appears to be driving down a road with a row of cones on either side used to simulate this forward motion (Figure 2.4). There are some mountains on the horizon, spaced at regular intervals. These mounts were the objects used to make LOD changes because they were fixed in peripheral on the horizon throughout the course of the simulation. To secure the attention of the viewer and attempt to guarantee the orientation of their focus vector, a focus task was placed in the center of the simulation. This task consisted of a square that maintained a constant distance from the viewer, rotating in the center of the viewer's field of view. The viewer's focus task was occasionally changed, either changing rotational direction or changing in color. Two criteria were ultimately selected to determine user distraction: movement of the eye and reaction times to the changes in the task.
The amount of change in object geometry during switching LOD levels was the main focus of the experiment. However, upon further inspection, no link between this criteria and user distraction was found. An interesting side note revealed by the analysis of the experimental data was that slower viewer velocities tended to have greater distraction rates.

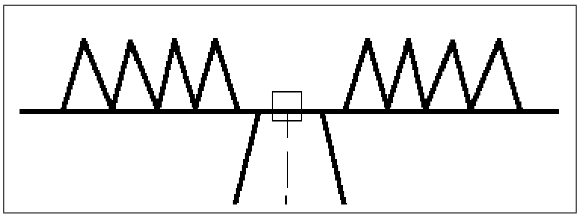

Figure 2.4: Simulation scene of the experiment

In 2003, Ross Brown, Luke Cooper and Binh Pham introduced a new approach to level of detail management based on visual attention [15]. The method for determining the level of detail of each visible object is dependent upon the calculated visual importance of the object.

$$P_{LOD}(i) = LOD(I, P(i)); P_{LOD}(i) \leq P(i) \tag{2.3}$$

$$LOD(I, P(i)) = P(i) + aI + s \tag{2.4}$$

Where;

$P(i)$ *is the number of polygons in object i*

a *is an arbitrary scale value*

I *is the object importance* $\in -1, 1$

s *is the stress adjustment*

The stress adjustment value is calculated from the time taken to update the previous frame. The result of this equation is the number of the polygons of the object. This value is then used to select the corresponding object LOD to render. The importance value I is a simple yes/no value mapped to {-1, 1}. Semantically, it says that either yes, the object is visually important, or no, the object is not visually important. To derive the variable I, five features are used in the model: size, position, rotation motion, speed and luminance.

$$I = \sigma \sum_{i=1}^{n} f_i w_i \tag{2.5}$$

Where;

f_i is the importance of the feature i ∈ {0,1}

w_i is the weight of feature i

σ is a transfer function that converts the continuous value returned by the summation to a discrete {-1,1} value through the use of a step function.

A virtual attention model which exploits the peculiarities of the human visual system is introduced by A.K. Beeharee, A.J. West and R. Hubbold in 2003 [16]. Their model is based on a series of experiments. They prepared a virtual room which contains several sections providing different points of interest and activity (Figure 2.5). There are six regions in the room, each of which has an activity taking place like color changing or rotating boxes. Participants move through a fixed path to carry boxes from the piles to the grid squares. They reported the changes in the environment.

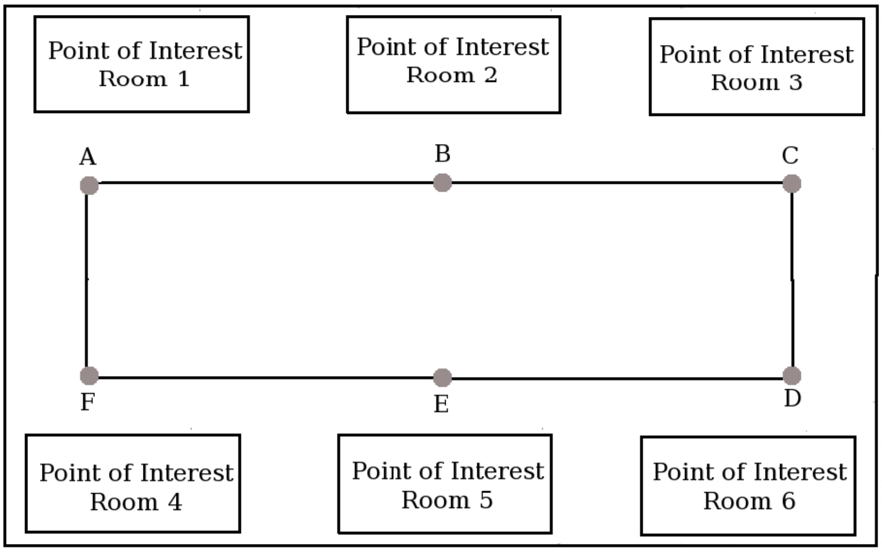

Figure 2.5: Experimental Setup

The results of the experiment showed that human visual system is sensitive to change in color, orientation and speed of the objects. Beeharee, West and Hubbold developed an attention engine based on these criterias. The engine computes a silency map to establish areas of visual interest. Each feature is weighted to build a saliency value for each object.

$$I_{Saliency} = W_{col} \times I_{col} + W_{con} \times I_{con} + W_{ori} \times I_{ori} + W_{spd} \times I_{spd} \tag{2.6}$$

Where;

W_{col} is the weight of color

I_{col} is the value for color

W_{con} is the weight of contrast

I_{con} is the value for contrast

W_{ori} is the weight of orientation

I_{ori} is the value for orientation

W_{spd} is the weight of speed

I_{spd} is the value for speed

2.3 Measuring Presence

The effectiveness of virtual environments are linked to the sense of presence reported by users of those virtual environments. An increase in the effectiveness of a simulation environment also increases the effectiveness of the training session.

A variety of measures of presence have been proposed [21]. Witmer and Singer developed the most comprehensive questionnaire. They presented a presence questionnaire (PQ) in 1998 to measure presence in virtual environments [20]. In addition, an immersive tendencies questionnaire (ITQ) was also introduced to measure differences in the tendencies of individuals to experience presence. These questionnaires are being used to evaluate relationships among reported presence and other research variables. PQ and ITQ questionnaire item stems are given in Appendix A and Appendix B.

The PQ and ITQ use a seven-point scale format. Instructions asked respondents to place an 'X' in the appropriate box of the scale in accordance with the question content and descriptive labels (Figure 2.6).

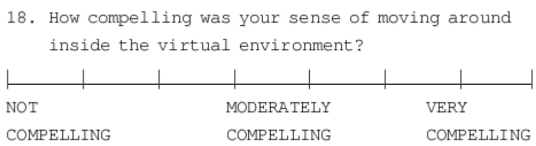

Figure 2.6: An Example Item from the Presence Questionnaire

The total PQ score gives the degree to which individuals experience presence in the tested virtual environment.

CHAPTER 3

ALGORITHM

The approach presented in this study combines computer graphics algorithms to smooth entity motion (Figure 3.1). Firstly smoothing and dead reckoning algorithms presented in IEEE Standard for DIS Application Protocols [2] are applied to the incoming target positional data. Secondly, the targets are culled with an algorithm based on frustum culling and occlusion culling. Finally, the entities are culled with respect to perceptional criteria like size and position. Last sent time and distance are also included in the final step.

```
//PreProcessing step
for each entity
        construct bounding volume

//Main Loop
while simulation continues

        receive entity positions from tactical interface

        //First Pass
        predict missing entity positions by dead reckoning
        smooth incoming positions

        //Second Pass
        view frustrum culling
        occlusion culling with bounding boxes

        //Third Pass
        perception based culling

        construct entity position update opcodes for remaining entities
        send opcodes to image generator

end while
```

Figure 3.1: Pseudo-code of the Algorithm

3.1 Preprocessing

In the second step, the entities are passed through frustum culling and occlusion culling algorithms. The entities are rendered in the image generator so the geometries are stored in the image generator. Image generator host is just responsible of position updates. The only information in the image generator host is the positions and types of the entities. In the preprocessing step, bounding boxes for the entities are constructed to be used in culling algorithms with respect to entity types. Bounding boxes are constructed manually (Figure 3.2).

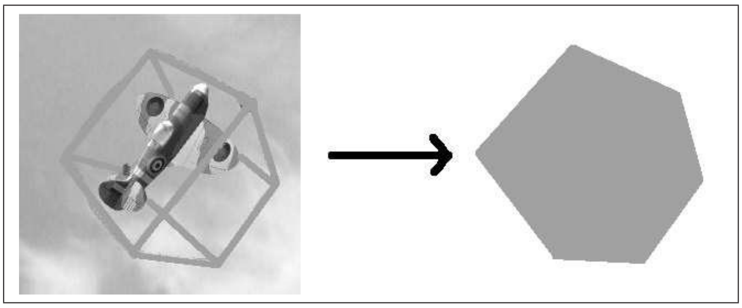

Figure 3.2: Bounding Box Construction

3.2 Smoothing Step

In this step, the entity positions pipelined form tactical interface are first processed through dead reckoning and smoothing. Position and orientation of each entity is extrapolated. Second order extrapolation is used at this step. For the entities which are not gathered from tactical interface, the output of second order extrapolation is used directly. For the entities which are successfully gathered from tactical interface, smoothing is applied. Incoming data is not directly used. The difference between incoming data and calculated result in extrapolation step, is divided to smoothing step number. This value is then added to extrapolation result. Which means; the entity is not directly positioned to incoming data, it is positioned in steps.

3.3 View Frustum and Occlusion Culling Step

In this step, the entities that are out of view frustum are eliminated first. For view frustum culling, only the positions of the entities are used. No geometry is used in view frustum culling algorithm. The entities that are within view frustum are then passed to occlusion culling. In occlusion culling step, bounding box of each remaining entity is rendered with a different color. Each pixel of the output image is then examined. The entities with bounding boxes having colors that are not in the final image are then eliminated. Remaining entities are then passed to next step. Also the number of pixels in the final image is stored for each entity. This value will be used in the final pass.

3.4 Perception Based Culling Step

This is the final elimination step before sending the entity positions to image generator. In this step, an importance value is calculated for each entity. The most important entities are then selected and rest is eliminated. The number of entities that will survive is image generator dependent.

A visual perception based elimination model is developed based on the LOD management algorithm based on visual attention introduced by Brown, Cooper and Pham [15] and virtual attention model introduced by Beeharee, West and Hubbold [16]. An importance value is calculated for each entity and the most important entities are chosen for the next step. The importance value is calculated by summation of weighted importance features. To decide the importance features of the elimination model, a questionnaire is prepared. The questionnaire is given in Appendix C. The participants are simulator pilots who are retired from Turkish Air Forces (TAF).

3.4.1 Questionnaire Analysis

All of the participants' answers to question 10 include terrain flight in order to avoid enemy detection. Terrain flight is flying close to the earth's surface during which altitude is adapted to the contours and cover of the ground. During such a flight, aircraft is masked by terrain.

Most of the participants focus on the aircrafts closer to own ship and bigger in size. Also an angular movement which causes an increase in the percentage of the visible surface of an aircraft, attracts the pilots attention.

3.4.2 Visual Importance Features

Four features are decided to be used in the algorithm; remaining (not occulted in the second pass) geometry, distance from eye point, altitude and time passed from last update. Each feature has weight coefficients and the importance value is calculated by summation of weighted features.

$$I = W_{size} \times I_{size} + W_{dist} \times I_{dist} + W_{alt} \times I_{alt} + W_{lst} \times I_{lst} \tag{3.1}$$

Where;
W_{size} is the weight of size
I_{size} is the value for size
W_{dist} is the weight of distance
I_{dist} is the value for distance
W_{alt} is the weight of altitude
I_{alt} is the value for altitude
W_{lst} is the weight of last update time
I_{lst} is the value for last update time

Color is also an important factor in visual perception. But all of the entities used in the simulation environment are aircrafts with camouflage. Color attribute of the entity is excluded from the importance value calculation.

3.4.3 Size and Distance from Eye point

The entities that are bigger in size and closer to eye point are more likely to attract the user. The fist two weighted features are size and the distance to eye point. The remaining geometry from the second pass is used for this calculation as occulted geometry is not expected to attract the user.

3.4.4 Altitude

The entities that are flying in low altitudes with respect to own ship are more likely to be masked by terrain in the final image produced in the image generator (Figure 3.3). Entities with high altitudes are more likely to attract the user. The importance value for altitude criteria is maximized if the target has an altitude higher or equal to own ship. Importance value starts to decrease when the target altitude decreases.

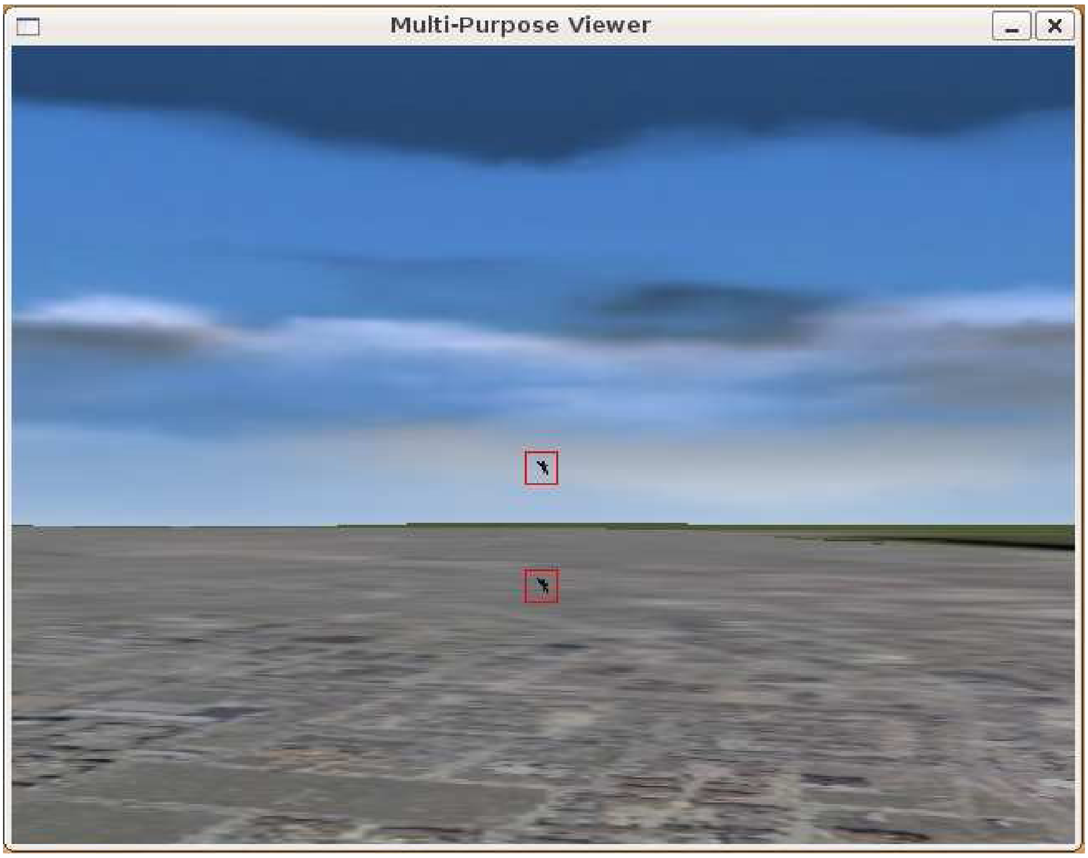

Figure 3.3: Targets at Different Altitudes

3.4.5 Last Sent Time

In the final pass, entities with high weighted features are selected. Without including last sent time to the final pass, always same entities will be selected for position update. Entities with low-weight features will remain non-updated. To provide position updates for low-weight features, last sent time is also included in the final step. Time passed form last position update is counted as a weighted feature.

3.5 Sending Step

In this final step, the positions of the remaining entities are converted to moving model control opcodes. Finally, prepared opcodes are sent to image generator.

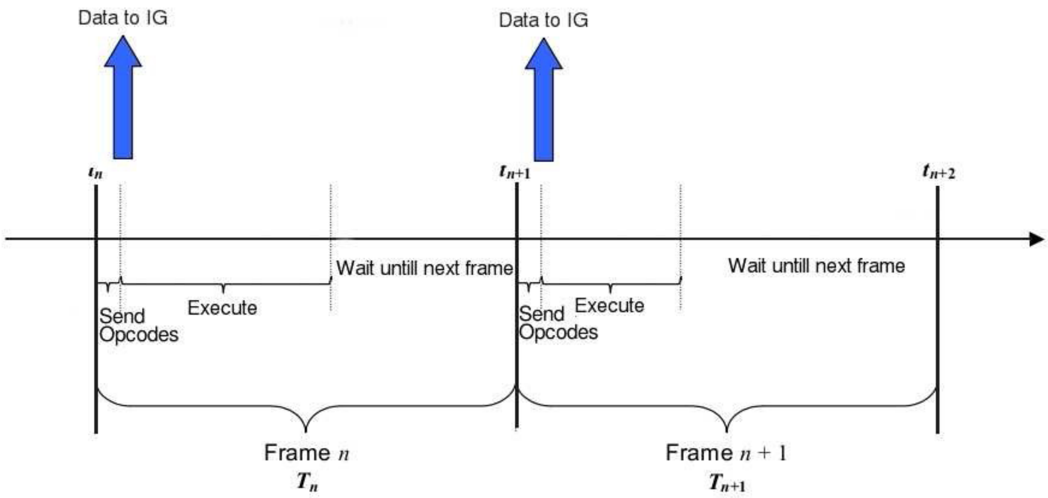

Figure 3.4: Host Execution

3.6 Execution and Messaging

The first three steps of the algorithm are executed serially. The sending step is not executed immediately. Host waits until the end of the simulation frame. The opcodes prepared in the first three steps are sent to the image generator at the begining of next frame (Figure 3.4). Messaging with a constant frame rate is achieved even if the execution period of the first three steps alter between frames.

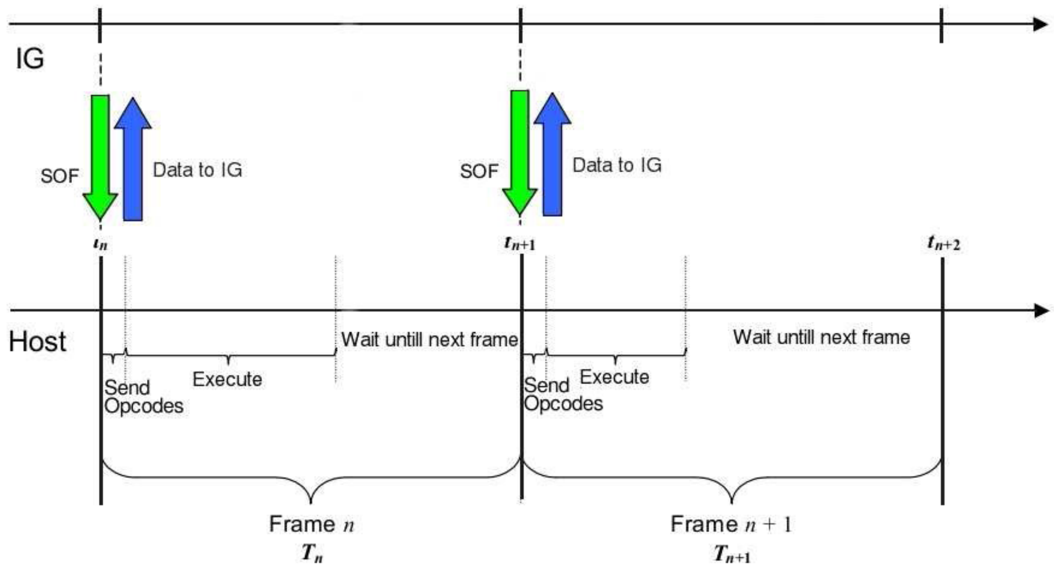

Figure 3.5: Synchronous Messaging

During synchronous messaging, image generator sends a start-of-frame (SOF) message to the host to signal the beginning of each frame. As the opcodes are ready at the start of the frames, host can immediately respond to each SOF message (Figure 3.5).

During asynchronous messaging, host sends opcodes to the image generator at a constant frame rate (Figure 3.6). In this case, IG will not receive either zero or two messages during a frame, causing frame jitter. So in both cases (synchronous and asynchronous messaging), host executes synchronously with the image generator. Only a latency of one frame is observed during asynchronous messaging.

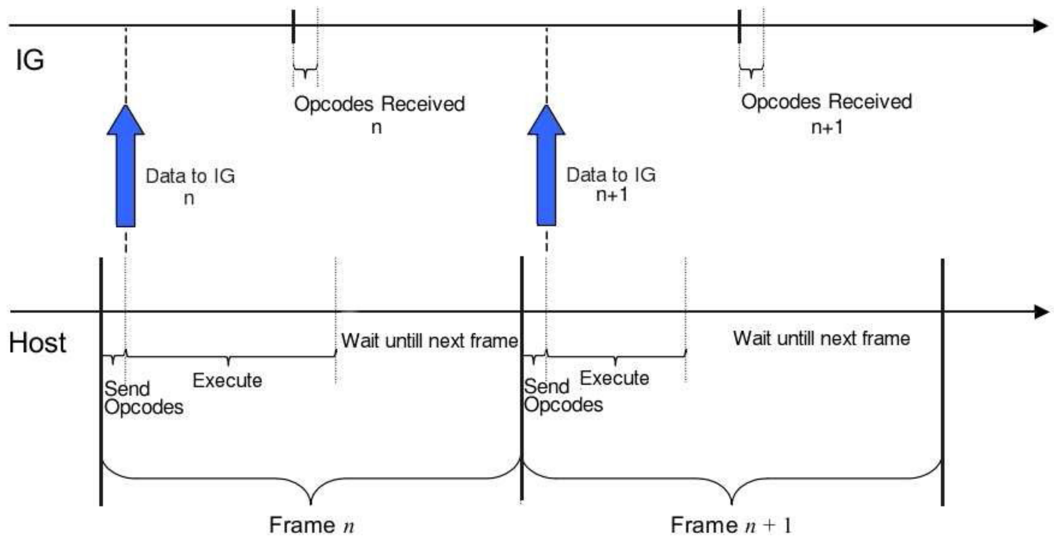

Figure 3.6: Asynchronous Messaging

CHAPTER 4

IMPLEMENTATION

In this chapter, the application of the proposed algorithm to a simple flight simulator is described. The chapter is organized as follows: Section 4.1 gives the high level overview of the overall system and a scenario which illustrates the implementation, Section 4.2 describes the details of cross user authentication process and discussions, Section 4.3 provides details and discussions on access control mechanism, and finally Section 4.4 provides the details of audit mechanism.

4.1 Overall Infrastructure

A simple flight simulator is developed to apply the purposed entity management algorithm. The simulator is composed of two main components: Host and Image Generator (Figure 4.1). The messaging between host and the image genarotor is asynchronous. User interacts with the simulator via a joystick plugged into the host component. The simulator is not a member of a simulated federation. Tactical environment is also simulated. The outer world is rendered in the image generator component.

4.2 Host

The host component is responsible of three main tasks: aerodynamics of own ship, management of tactical environment and visualization of these two tasks. The aerodynamics model calculates the position of own ship. Tactical component calculates the positions of the entities in the simulation environment. After the positions of own ship and other entities are

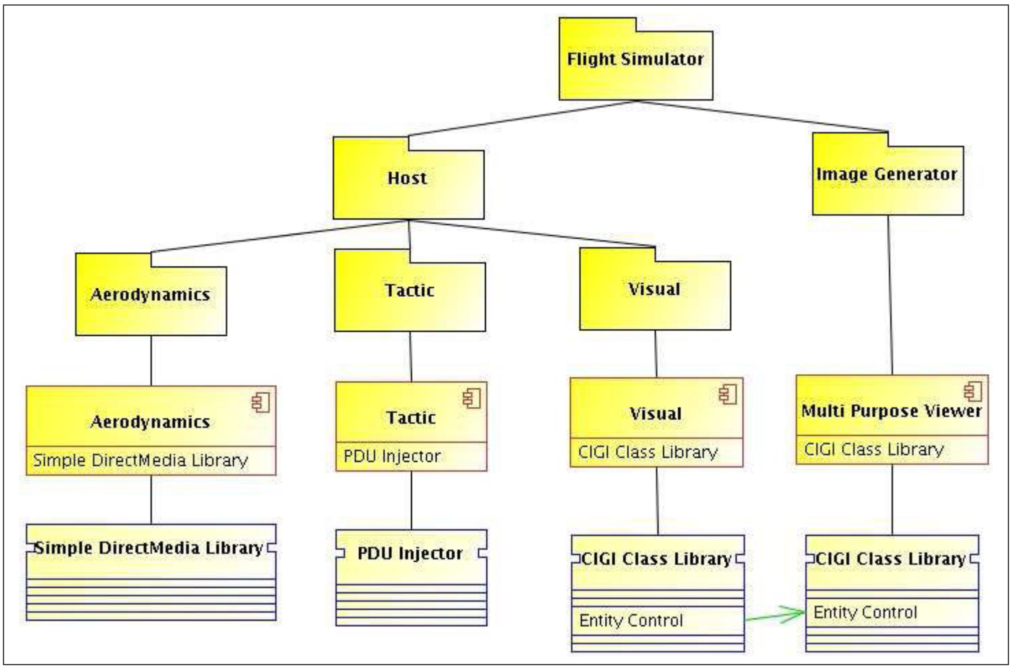

Figure 4.1: Flight Simulator Components

calculated, this positional data is wrapped and sent to the image generator component. This sequence loops in real time till the end of the simulation (Figure 4.2).

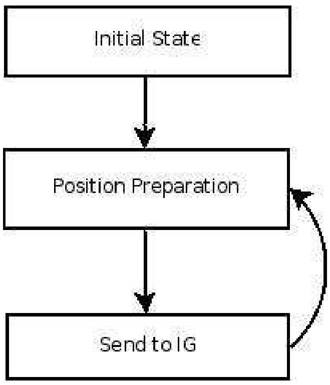

Figure 4.2: Flowchart of Host Component

4.2.1 Aerodynamics Component

The aerodynamics component calculates the position of the own ship at every simulation frame. The user interacts with the simulator via a joystick plugged into the host to control own ship. The simulator has a primitive aerodynamics model. The model decides the position of the own ship in six degree of freedom according to pitch, roll and throttle value captured from a joystick plugged into the host. Simple Direct Media Library (SDL) is used for interfacing the joystick. Further information on joystick interfacing can be found in SDL website [18].

4.2.2 Tactical Component

Tactical component manages the positions of the entities in the simulation environment. The purposed multi-pass entity management algorithm is applied in this component.


```
− <IncomingTargetData>
  − <Unit>
      <IGID>2</IGID>
      <Count>50</Count>
      <X>-17400.121896</X>
      <Y>-4989000.936928</Y>
      <Z>3960950.0</Z>
      <Psi>0.0</Psi>
      <Theta>0.0</Theta>
      <Phi>0</Phi>
      <LvX>5</LvX>
      <LvY>50</LvY>
      <LvZ>50</LvZ>
      <LaX>0.0</LaX>
      <LaY>0.0</LaY>
      <LaZ>0.00</LaZ>
      <AvX>0.0</AvX>
      <AvY>0.0</AvY>
      <AvZ>0.0</AvZ>
    </Unit>
  − <Unit>
      <IGID>1</IGID>
      <Count>150</Count>
      <X>-17410.121896</X>
      <Y>-4989020.936928</Y>
      <Z>3960850.0</Z>
      <Psi>181.567306</Psi>
      <Theta>1.006238</Theta>
      <Phi>150.031930</Phi>
      <LvX>-2</LvX>
      <LvY>50</LvY>
      <LvZ>50</LvZ>
      <LaX>0.0</LaX>
      <LaY>0.2</LaY>
      <LaZ>0.01</LaZ>
      <AvX>0.0</AvX>
      <AvY>0.0</AvY>
      <AvZ>0.0001</AvZ>
    </Unit>
```


Figure 4.3: Protocol Data Units in XML Format

4.2.2.1 First Pass: Position Calculation

A distributed simulation environment contains other simulators and computer generated vehicles. The positional information of these entities are received over an inter network in the form of protocol data units (PDUs). In this study, the distributed simulation environment is also simulated. The protocol data units are prepared initially and stored in XML format (Figure 4.3). The simulated PDUs are received by the simulator within the simulation frame defined in the Count tag of the units.

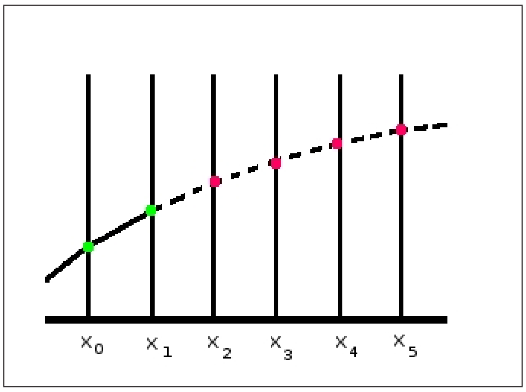

Figure 4.4: Dead Reckoning for movement on x-axis. (a) Positions marked as green are received data units (b) Positions marked as red are predicted positions

In the first pass of the algorithm, the positions of the entities are calculated. These positional data is calculated at every simulation frame. If no PDU is received for an entity during the simulation session, the position of the entity is predicted each frame (Figure 4.4).
When a PDU for an entity is received within a frame, neither its position is only predicted nor incoming position is directly used. The difference between predicted position and incoming position is divided into smoothing steps. These steps are added to the predicted position (Figure 4.5). The entity is not directly positioned to the incoming position, it follows a path which intersects the real path after smoothing steps are applied. This intersection is observed if no other PDUs are received till smoothing is finished. If a new PDU is received, the smoothing step is restarted again.

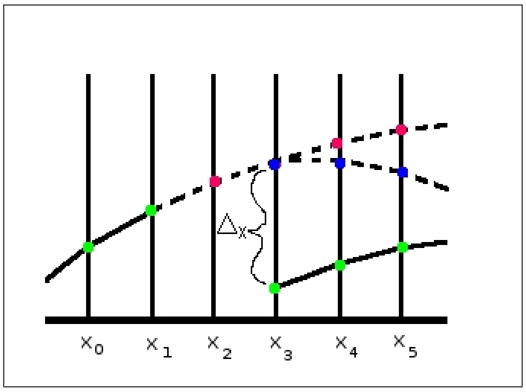

Figure 4.5: Smoothing for movement on x-axis. (a) Positions marked as green are received data units (b) Positions marked as red are predicted positions (c) Positions marked as blue are positions after smoothing step

4.2.2.2 Second Pass: View Frustum and Occlusion Culling

In the second pass, the entities are started to be eliminated if total count exceeds the image generator limit. Image generator limit is defined in the start up as a static integer. This value is image generator dependent.

The first applied elimination method is view frustum culling. The entities which are not inside the view frustum are eliminated and their positions are not updated to the image generator. Model geometry is not included in inside/outside test. The vertical and horizontal angles of the frustum are extended by ten degrees (five degrees at each side). The center of model geometry is passed to point in frustum test (Figure 4.6). The model with center positions outside the extended view frustum are eliminated. The positions calculated in the first pass are used in this test.

Entities are eliminated after their positions at that frame are calculated. The calculations for the entities that will be eliminated may seen unnecessary. But the position of the entity at that frame affects the elimination algorithms. An entity outside the view frustum may enter the frustum in the next step, but this can not be predicted if the positional data is calculated even the entity is eliminated.

The next applied elimination method is occlusion culling. In a full flight simulator architecture, rendering is the task of the image generator. 3D models and databases are stored in the IG. In the algorithm purposed in this study, the entities are pre-rendered in the host sys-

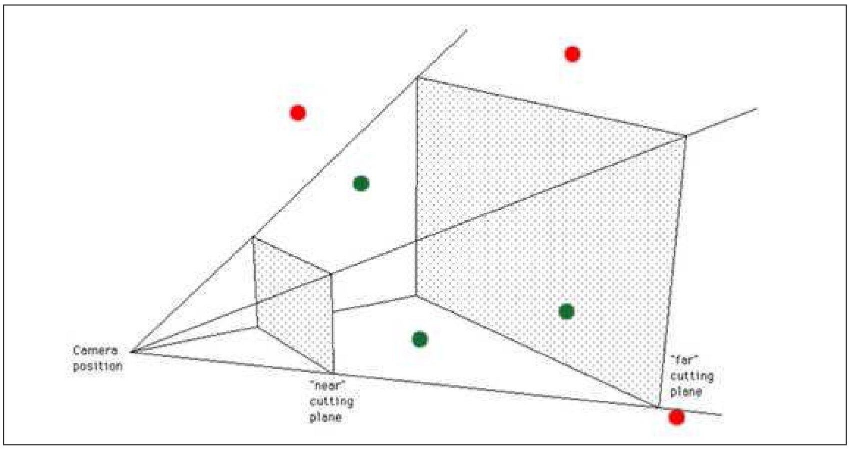

Figure 4.6: View Frustum Culling: Inside-Outside Test. (a) Positions marked as green are models inside view frustum (b) Positions marked as red are models outside view frustum

tem. Bounding boxes for each entity is constructed at the initialization process. 3D model geometry is not accessible for the host but the type of the entity is known. Bounding box construction is based on the type of the entity. Predefined dimensions are used in this process. Pre-rendering view frustum is constructed with the same parameters used in the image generator. The scene is rasterized using this frustum (Figure 4.7).

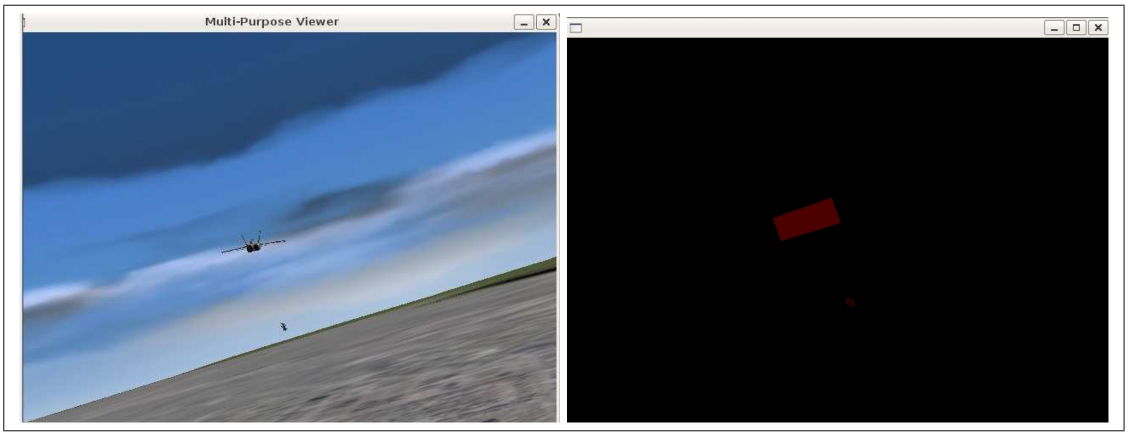

Figure 4.7: Pre-Rendering of the Scene

Each entity has a unique color. All faces of the bounding box of an entity is rendered with this unique color value. A hash function is used to calculate the color values of the entities. RGB color model is used in this calculation. Color values are within range {0,1}. The ID of the entity is used as the input of the hash function (Table 4.1).

$$R_{index} = (index\%10) \times 0.1 \quad (4.1)$$

$$G_{index} = ((index/10)\%10) \times 0.1 \quad (4.2)$$

$$B_{index} = (((index/10)/10)\%10) \times 0.1 \quad (4.3)$$

Table 4.1: RGB Color Calculation Examples

ID	R	G	B
1	0.1	0.0	0.0
2	0.2	0.0	0.0
10	1.0	0.0	0.0
11	1.0	0.1	0.0
12	1.0	0.2	0.0

The number of pixels belonging to each entity is counted. The entities completely occulted by other entities come up with zero pixel counts in the output image. These entities are then eliminated. The pixel counts for the remaining entities are also stored. These values are used as the size attribute for the final elimination pass: perception based elimination.

4.2.2.3 Third Pass: Perception Based Elimination

In the third pass, the remaining entities are eliminated via a visual perception based algorithm if total count still exceeds the image generator limit. A visual importance value for each remaining entity is calculated. This value is then used to decide to eliminate the entity or not.
To derive the visual importance, four features are used in the model: remaining (not occulted in the second pass) geometry, distance from eye point, altitude and time passed from last update. Each feature has weight coefficients and the importance value is calculated by summation of weighted features. Detailed information on these features are presented in Section 3.4: Perception Based Culling Pass.
After the visual importance values of the entities are calculated, they are then inserted into a priority queue. The key attribute of the priority queue is the visual importance value of the entity. Finally, constant number of entities are popped out of the queue (Figure 4.8). This constant number is image generator dependent and is the model count limit that the image generator can process in one frame. The positional update opcodes of only these popped out

entities are sent to the image generator in that simulation frame.

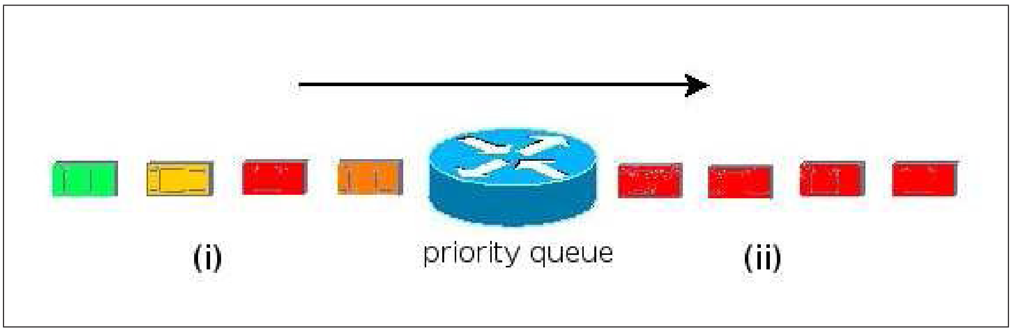

Figure 4.8: Perception Based Elimination. (i) Remaining entities from first two passes with calculated visual importance values (ii) Popped out entities

4.2.3 Visualization Component

In this final step, the positions of the popped out entities are converted into opcodes. Prepared opcodes are sent to the image generator. Common Image Generator Interface (CIGI) is used as the communication protocol between host and image generator. The positional data of the entities is converted into CIGI Entity Control opcodes. CIGI Class Library is imported for this wrapping operation. The CIGI Class Library (CCL) is a library to group, format, pack, and unpack the data to the specification of CIGI. It can be used by both the host and the IG. The CCL handles packing, unpacking, and byte swapping automatically. Further information on CCL can be found in the CCL User's Guide [19].

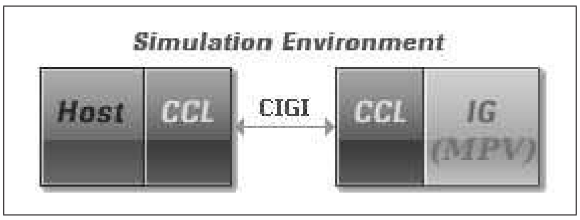

Figure 4.9: HOST - MPV Communication

4.3 Image Generator

Multi-Purpose Viewer (MPV) is used as the image generator component of the simulator. It is an Open Scene Graph (OSG) based tool. It uses Common Image Generator Interface (CIGI) as the communication protocol with host (Figure 4.9). The Open Scene Graph library is used to render the scene. MPV does not require any external utilities and any special environment (such as a system with real-time extensions) in order to be functional. More information on the installation, requirements and usage of MPV can be found in the MPV User's Guide [17].

CHAPTER 5

RESULTS

The effectiveness of virtual environments are linked to the sense of presence reported by users of those virtual environments. Presence within the presented flight simulation is measured to reflect the performance of the developed algorithm. Immersive tendencies questionnaire (ITQ) is used to measure differences in the tendencies of individuals to experience presence and presence questionnaire (PQ) is used to measure presence of the simulation environment. Firstly in this chapter, the test environment is presented. Secondly, degraded version of presence questionnaire (PQ) is presented. Finally, questionnaire results for the developed algorithm and a discussion on the performance of the algorithm is provided.

5.1 Test Environment

Multi Purpose Viewer (MPV) is used in single-channel configuration. There is one rendering channel, which communicates directly with the Host. Host and image generator (MPV) are physically connected via an Ethernet crossover cable.

Tactical environment is constructed with ten models (Table 5.1). 3D models are given in Appendix E. The primary task of the participants is following an F/A-18 Hornet model for 1-2 minutes. Image generator processable model count is limited to five. Half of the models are eliminated and positional updates of only remaining models are done within each simulation frame.

Table 5.1: Tactical Scenario

Model IG ID	Model Type
1, 2, 4, 6	F-22 Raptor
3, 5, 7	F/A-18 Hornet
8, 9, 10	Mi-24 HIND

5.2 Performance of the Algorithm

The effectiveness of the simulated environment is linked to the sense of presence reported by its users. An increase in the presence of the simulated environment increases the effectiveness of the training session.

The presented algorithm predicts missing positions of the entities and produce smooth motion paths in the first step. Then entities outside the frustum or occulted by other entities are eliminated in the second step. In the third step importance values of the remaining entities are calculated and the most important entities survive.

PQ test is applied twice to each participant to find an increase in the presence of the simulated environment due to the presented perception based algorithm. In the first test session, the third step (perception based culling) is excluded from the algorithm. In the second test session the whole algorithm (all three steps are included) is tested.

Using the simulator for the second time may cause quicker adjustment to the control and display systems. Half of the participants tested the session with the whole algorithm first and the other half tested it in the second session to minimize the effects of quick adjustment.

5.2.1 Degraded Version of PQ

PQ is degraded by excluding the questions that are not in correlation with PQ total score. Firstly, PQ total scores of the participants are calculated. Secondly, Pearson product moment correlation coefficient of the questionnaire items are calculated (Table 5.2). Questionnaire items 25,28 and 29 are not in correlation with PQ total score so they are excluded from total score calculation process.

Table 5.2: Pearson Product Moment Correlation Coefficients

PQ Item	Pearson r
1	0.41
2	0.87
3	0.79
5	0.46
7	0.66
10	0.47
12	0.62
13	0.46
14	0.54
18	0.57
19	0.56
20	0.80
23	0.40
25	0.17
26	0.63
27	0.55
28	-0.04
29	-0.08
30	0.52

5.2.2 Participant Characteristics

Questionnaires are applied to twenty participants. All of the participants are simulator pilots. They are familiar with the flight systems and instruments of an aircraft. They have at least ten hours of flight experience with a flight simulator. Their ages are between 25 and 40.

5.2.3 Test Procedure

Participants are informed about the differences between the sessions but they are not informed which session will be tested first. System components are introduced briefly before the sessions. After the first session, participants are given a five minutes break and PQ is applied for the first session. Second PQ is applied after the second session. Questionnaire results are given in Appendix F.

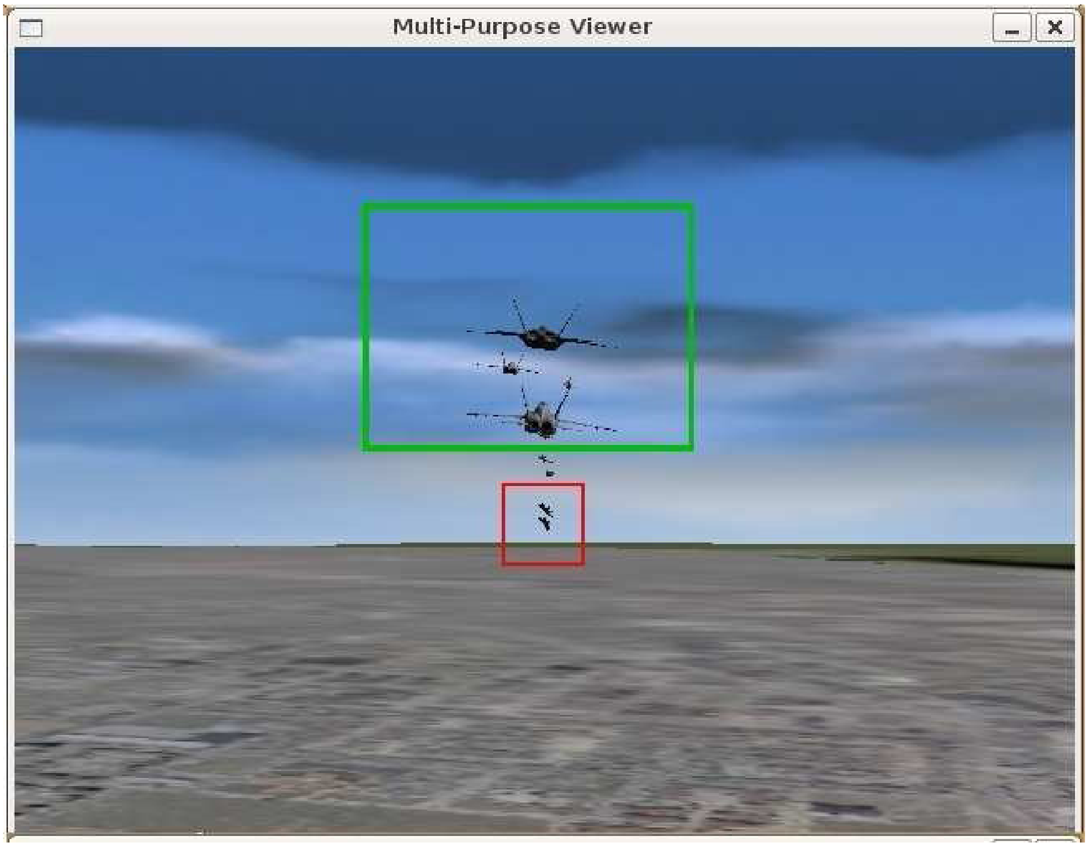

Figure 5.1: Importance of the Entities in the Scene. Entities within the green box are the most important entities. Entities within the red box are the least important ones. Flickering is driven to the models within the red box.

5.2.4 Result Analysis

The average score for PQ item 28 is 3.35 for the first session and 2.65 for the second session (Table 5.3). There were flickering models in both sessions but the presented algorithm drove flickering to the models that were less important for the user (Figure 5.1). The primary task in the simulation sessions was following a 3D model. As the wingman and the other important models followed smoother paths, visual display quality as a distraction factor for the assigned task has a decreased score in the second session.

The participants reported that the control mechanism in the second session was more natural (PQ Item 7). Also control device as a distraction factor for the assigned task has a decreased score in the second session (PQ Item 29). The control device and control mechanism were exactly same in the sessions. The actions performed by the users were task oriented. When the wingman followed a smooth path, participants were able to anticipate its position and so the next own ship position relative to the wingman in response to their actions more accurately. This accuracy was a result of wingman's following a smooth path but participants thought that there was an enhancement in the control mechanism.

Table 5.3: Average Scores for PQ Items

Item No	Item	Avg. Score (PQ1)	Avg. Score (PQ2)	Rate of Increase (%)
7	How natural was the mechanism which controlled movement through the environment?	3.55	4.85	36.62
28	How much did the visual display quality interfere or distract you from performing assigned tasks or required activities?	3.35	2.65	-20.90
29	How much did the control devices interfere with the performance of assigned tasks or with other activities?	3.6	3.15	-12.50

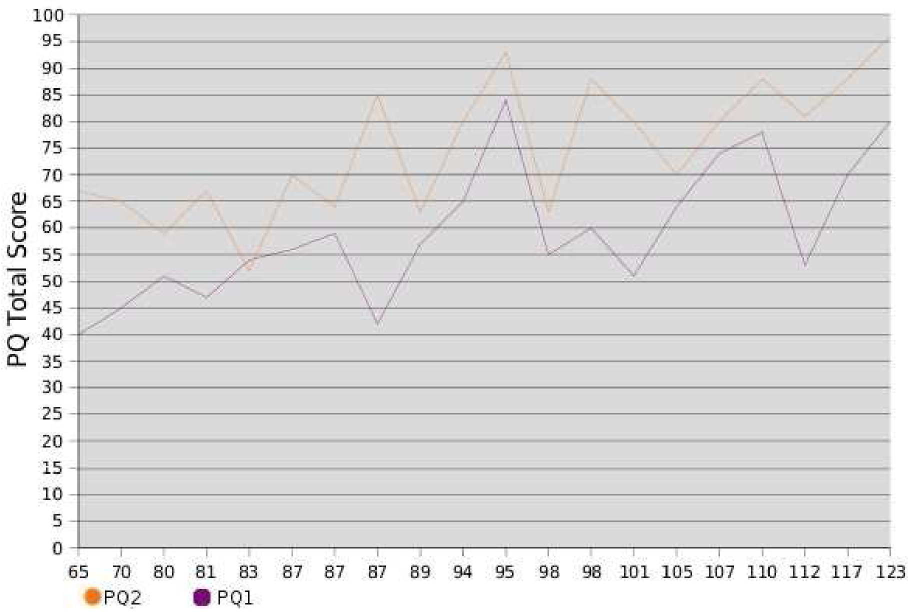

Figure 5.2: PQ Total Scores of Both Sessions

The total scores of the presence questionnaires are presented in Figure 5.2. PQ total score of the session with the whole algorithm is greater than the total score of the session with the third pass (perception based culling) excluded algorithm for most of the participants. The total scores show that the presented perception based algorithm increases the presence sensed by the participants and so increases the effectiveness of the simulation training sessions.

5.2.5 ITQ Score Contribution

An important aspect influencing human virtual environment performance is the effect of user differences. User characteristics that significantly influence virtual reality experiences need to be identified in order to design virtual environment systems that accommodate these unique needs of users [22].

In order to determine if the tendencies of the participants to experience presence are influential in our simulated virtual environment, ITQ total scores of the participants are also examined. PQ total scores of the participants belonging to the perception based culling step excluded session are given in Figure 5.3. PQ total scores of the participants belonging to the all steps included session are given in Figure 5.4.

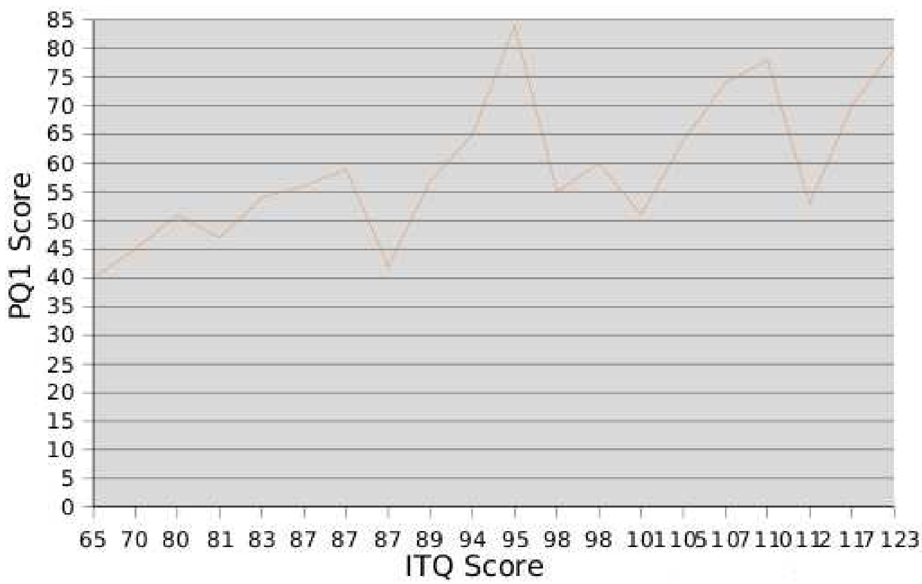

Figure 5.3: PQ1 Total Scores of the Participants vs ITQ Scores

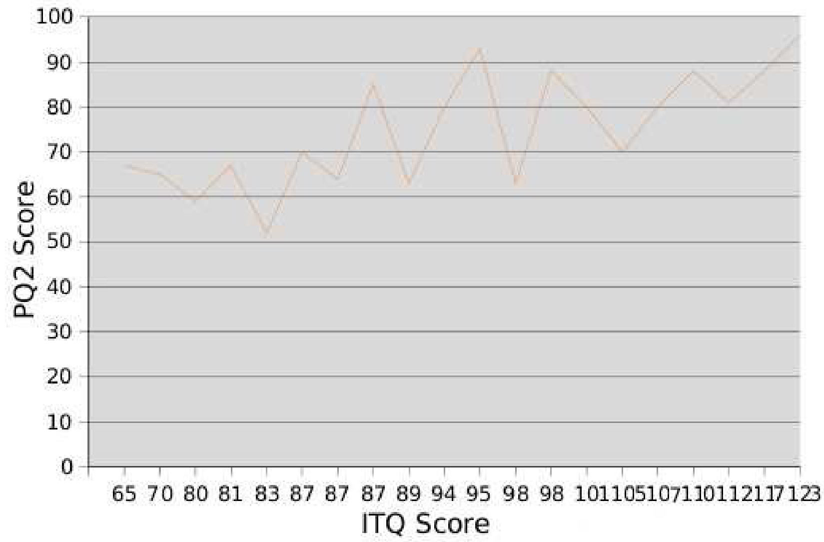

Figure 5.4: PQ2 Total Scores of the Participants vs ITQ Scores

PQ total score increase rates between the sessions are also ordered by the participants' ITQ scores (Figure 5.5).

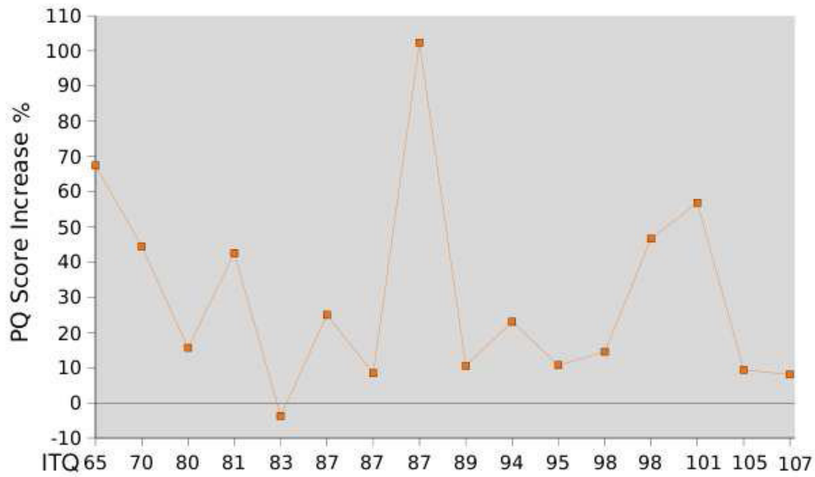

Figure 5.5: PQ Score Increase Rate of the Participants Ordered by ITQ Scores

Pearson product moment correlation coefficients for questionnaire results are given in Table 5.4. There is a high correlation between ITQ scores and PQ scores which means there is a positive linear relationship between the tendencies of the participants to experience presence and the presence of the simulated environment.

The correlation obtained from the increase rates is -0.25 showing us that there is a small negative correlation between the tendencies of the participants to experience presence and rate of increase in the presence of the simulated environment between 1^{st} and 2^{nd} sessions.

Table 5.4: Pearson Product Moment Correlation Coefficients

	ITQ Score to PQ1 Score	ITQ Score to PQ2 Score.	ITQ Score to Rate of Increase
Pearson r	0.68	0.72	-0.25

5.2.6 OpCode Statistics

The opcode traffic between the host and the image generator is analyzed in this section. During the experiments, simulator is not controlled by a user, it flies in auto pilot mode. The tactical scenario previously experienced by the participants is used in the flight sessions. The entities are positioned at the same initial points and same PDUs are injected into the tactical interface.

Table 5.5: Opcode Traffic Statistics for Test Sessions with High Air Speed

Session	Total Opcode #	Opcode per Sec.	Opcode per Frame	Model Count per Frame
1	2995	149.75	4.99	6.66
2	2995	149.75	4.99	6.57

In the first experiment, the simulator flies with high air speed ($\approx$ 225 knots). The flight is replayed twice. In the first session, the third step (perception based culling) is excluded from the algorithm. In the second session the whole algorithm (all three steps are included) is tested. Opcode traffic statistics for both sessions are presented in Table 5.5 and opcode statistics for each entity in the tactical scenario are given in Table 5.6.

Table 5.6: Opcode Statistics for Entities in the Tactical Scenario with High Air Speed

ID	Total Opcode (1^{st}Session)	Total Opcode (2^{nd}Session)	Opcode per Sec. (1^{st}Session)	Opcode per Sec. (2^{nd}Session)	Opcode per Frame (1^{st}Session)	Opcode per Frame (2^{nd}Session)
1	600	600	30.0	30.0	1.0	1.0
2	0	0	0.0	0.0	0.0	0.0
3	92	61	4.6	3.05	0.15	0.1
4	600	541	30.0	27.05	1.0	0.9
5	241	241	12.05	12.05	0.4	0.4
6	80	55	4.0	2.75	0.13	0.09
7	520	500	26.0	25.0	0.87	0.83
8	491	552	24.55	27.6	0.82	0.92
9	359	217	17.95	10.85	0.6	0.36
10	17	233	0.85	11.65	0.03	0.39

In the second experiment, the simulator flies with lower air speed ($\approx$ 50 knots). The scenario is cruised slower and the model density within the view frustum is higher. The flight is again replayed twice. In the first session, the third step (perception based culling) is excluded from the algorithm. In the second session the whole algorithm (all three steps are included) is tested.

Table 5.7: Opcode Traffic Statistics for Test Sessions with Low Air Speed

Session	Total Opcode #	Opcode per Sec.	Opcode per Frame	Model Count per Frame
1	2995	149.75	4.99	8.94
2	2995	149.75	4.99	9.01

Opcode traffic statistics for both sessions are presented in Table 5.7 and opcode statistics for each entity in the tactical scenario are given in Table 5.8.

The total number of entity control opcodes sent to the image generator is same in both sessions (Table 5.5, 5.7). In the first sessions, the entities within the view frustum are order by their ids. In the second step, they are ordered by their importance values. If the number of models within the view frustum exceeds image generator model count limit, maximum number of opcodes are sent to the image generator in both sessions. When the image generator model count limit is not reached, positional data of all models within the frustum are updated in both sessions. Perception based culling step of the presented algorithm does not cause a retrenchment in total number of opcodes. This step just adds an opcode budget distribution mechanism to the system. Change rates in number of opcodes sent to the image generator per entity between the sessions are presented in Table 5.9.

Table 5.8: Opcode Statistics for Entities in the Tactical Scenario with Low Air Speed

ID	Total Opcode (1^{st}Session)	Total Opcode (2^{nd}Session)	Opcode per Sec. (1^{st}Session)	Opcode per Sec. (2^{nd}Session)	Opcode per Frame (1^{st}Session)	Opcode per Frame (2^{nd}Session)
1	600	314	30.0	15.7	1.0	0.52
2	31	19	1.55	0.95	0.05	0.03
3	551	600	27.55	30.0	0.92	1.0
4	600	391	30.0	19.55	1.0	0.65
5	579	577	28.95	28.85	0.97	0.96
6	569	600	28.45	30.0	0.95	1.0
7	49	13	2.45	0.65	0.08	0.02
8	21	439	1.05	21.95	0.04	0.73
9	0	22	0	1.1	0	0.04
10	0	25	0	1.25	0	0.04

Table 5.9: Rate of Change in Number of Opcodes Sent to the Image Generator per Entity between the 1^{st} and 2^{nd} Sessions

Model ID	1	2	3	4	5	6	7	8	9	10
Rate of Change in Opcode # with High Air Speed (%)	0	NAN	-33.3	-10	0	-30.8	-4.6	12.2	-40	1200
Rate of Change in Opcode # with Low Air Speed (%)	-48	-40	8.7	-35	-1.03	5.26	-75	1725	NAN	NAN

During low speed flight, the tactical scenario is cruised slower and the relative positions of the entities within the view frustum change slower. The most important entities are not likely to lose their importance and the opcode budget is spent mostly for these entities. Other entities can not make use of this budget.

Table 5.10: Variance Values based on the Number of Entity OpCodes Sent during the Flight Sessions

	Variance
High Speed 1^{st} Session	59288.44
Low Speed 1^{st} Session	87378.44
High Speed 2^{nd} Session	52621.11
Low Speed 2^{nd} Session	66529.56

During high speed flight, the most important entities are more likely to lose their importance as the relative positions of the entities within the view frustum change faster. The number of opcodes per entity are more scattered during low speed flight than the high speed flight; so higher variance values are observed for the sessions with low speed. Variance values based on the number of opcodes per entity are presented in Table 5.10.

CHAPTER 6

CONCLUSION

Flight simulators are used by the aviation industry for the training of pilots and crew members in both civil and military aircrafts. Using simulators, pilots can be trained for the situations that are dangerous and unsafe to be experienced with a real aircraft. These situations include engine failures, malfunctions of aircraft systems, threat avoidance and so forth. The effectiveness of a training session in a flight simulator is linked to the degree of presence sensed by the pilots.

Flickering 3D models in the simulation environment is a commonly observed situation in complex tactical environments which decreases presence in the training sessions and so decreases the effectiveness of the training. In this study, an algorithm for the host systems using an image generator component for rendering is presented to minimize flickering within the visual scene.

The algorithm first produces smooth position sequences for each entity in the simulation environment. The missing positions in the sequences are predicted via observing previously calculated positions. Incoming positional data units are not also directly used. A smoothing procedure is applied to these units and models are mounted into their real paths after smoothing steps are finished.

Smooth position sequences for each entity in the simulation environment are prepared but not all of them are sent to the image generator. Image generators have model count limits. This limit is the number of positional data updates that an image generator can process in one frame. If the total number of entities in the simulation environment exceeds this limit, some of them should be eliminated. In the second step of the algorithm, the entities that are outside the view frustum or totally occulted by other entities are eliminated. Occlusion culling algorithm needs model geometry to be functional. 3D model geometry is not accessible for the host but the type information of the entity is available. Bounding boxes based on entity type for each entity are constructed at the initialization process. Each model is also given a unique color index. The scene is pre-rendered at the host side. Models are represented by their bounding boxes rendered with the given color indices. The entities completely occulted by other entities do not have any pixels in the rasterized image.

A final elimination based on visual perception is applied to remaining entities. An importance value is calculated for each entity and the most important entities survive. The importance value is calculated by summation of weighted importance features; size, distance, altitude and last update time.

The performance of the algorithm is tested via Presence Questionnaire. PQ is applied twice to each participant to sense the increase in the presence of the simulated environment. In the first session, the third pass (perception based culling) is excluded from the algorithm. In the second session the whole algorithm (all three steps are included) is tested by the participants. It is observed that the presence sensed by the participants increases if the perception based culling step is included in the algorithm.

The influence of the tendencies of the participants to experience presence in our simulated virtual environment is also examined. A high correlation between ITQ scores and PQ scores is observed which means there is a positive linear relationship between the tendencies of the participants to experience presence and the presence of the simulated environment. Namely, the participants which have higher tendencies to experience presence within a virtual environment are immersed more by the tested simulation environment. But a very small negative correlation between the tendencies of the participants to experience presence and rate of increase in the presence of the simulated environment between 1^{st}(perception based culling excluded) and 2^{nd} (all step included) sessions is obtained. No direct link between the performance of the presented algorithm and user tendency to experience presence can be established.

The opcode traffic between the host and the image generator is also analyzed. Perception based culling step does not cause a retrenchment in total number of entity control opcodes sent to the image generator but manages its distribution between entities. This step is applied when the image generator model count limit is still exceeded after the first two steps. In this step, the number of models to be updated to the image generator is not decreased. The budget is still totally spent but for the most important models.

Increasing presence is an important research area in the simulation industry. Rendering task is mostly the main focus for today's studies on visual systems but unlike other approaches, in this study an algorithm is purposed for image generator host component.

6.1 Limitations and Future Work

Limitations and future work related to host systems and image generators are discussed separately in the following paragraphs.

6.1.1 Host

Limitations and future works related to host systems is summarized as follows:

- Dead reckoning and smoothing algorithms do not take entity aerodynamics into consideration. An algorithm taking entity type as a parameter should be developed for more realistic entity motion.
- In the perception based culling step of the algorithm, speed can be added as a new perceptual feature.
- In the perception based culling step of the algorithm, the weights of the features are static. Dynamic weights can be used in this step.
- Parallel implementation should be considered for faster processing. Algorithm steps can be parallelized and use the data prepared by a former step within the previous frame.

6.1.2 Image Generator

Within every simulation frame, the presented algorithm selects the entities whose positional data will be sent to the image generator. The number of these entities are bounded to the processable model count limit of the image generator component. This limit should be increased as it affects the overall performance of the system. Providing additional interfaces for entity control opcodes and parallelization of the opcode handling process can be considered at this point.

REFERENCES

[1] G. Riva, F. Davide, W. IJsselsteijn. Being there: concepts, effects, and measurement of user presence in synthetic environments (pp. 41-57). Amsterdam: IOS Press, 2003.

[2] IEEE Std 1278.1-1995. IEEE Standard for Distributed Interactive Simulation Application Protocols. ISBN 1-55937-572-8

[3] Katz A. Synchronization of Networked Simulations. In Proc. of the 11th DIS Workshop on Standards for the Interoperability of Distributed Simulation, pp.81-87, Sept. 1994.

[4] Interface Control Document for the Common Image Generator Interface Version 3.2 (2006. April 7)

[5] P. Heckbert, M. Garland. Multiresolution Modeling for Fast Rendering. Proceedings of Graphics Interface '94, pp.43-50, May 1994

[6] J. H. Clark. Hierarchical Geometric Models for Visible Surface Algorithms. Communications of the ACM, vol. 19, no. 10, pages 547-554, October 1976

[7] T. A. Funkhouser and C. H. S'equin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In Proceedings of SIGGRAPH '93, pages 247-254, 1993.

[8] A. E. W. Mason and E. H. Blake. Automatic hierarchical level of detail optimization in computer animation. Computer Graphics Forum, 16(3):191-200, 1997.

[9] C. Zach, S. Mantler, K. Karner. Time-critical Rendering of Discrete and Continuous Levels of Detail. Proceedings of the ACM symposium on Virtual reality software and technology, Pages:1-8, 2002

[10] Fei-Ah Law, Tiow-Seng Tan.Preprocessing Occlusion for Real-Time Selective Refinement. Symposium on Interactive 3D Graphics Atlanta GAUSA, 1999

[11] Teller, S. and Sequin, C. Visibility Preprocessing for Interactive Walkthroughs. Pmt. of ACM Siggraph, pp 61-69, 1991.

[12] A. Grundhöfer, B. Brombach, R. Scheibe, B. Fröhlich. Level of detail based occlusion culling for dynamic scenes. Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia GRAPHITE '05. Pages:37-45 , 2005

[13] Jurriaan D. Mulder, Robert van Liere. Fast perception-based depth of field rendering. Proceedings of the ACM symposium on Virtual reality software and technology. Pages: 129-133, 2000

[14] Gerald Pitts, Daniel Cornell. Peripherality based level of detail switching as a visualization enhancement of high-risk simulations. Proceedings of the 1999 ACM symposium on applied computing. Pages: 98-104, 1999

[15] Ross Brown, Luke Cooper, Binh Pham. Visual attention-based polygon level of detail management. Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia. Pages: 55-ff, 2003

[16] A. K. Beeharee, A. J. West, R. Hubbold. Visual attention based information culling for Distributed Virtual Environments. Proceedings of the ACM symposium on Virtual reality software and technology. Pages: 213-222, 2003

[17] A. Sampson. Multi Purpose Viewer (MPV) User's Guide. 22 August 2005

[18] Simple DirectMedia Layer, last accessed date: 23 July 2008, http://www.libsdl.org

[19] CIGI Class Library (CCL) User's Guide, Version 1.7.4, last accessed date: 23 July 2008, http://cigi.sourceforge.net/manual/version1/ccl/section_1.html

[20] B. G. Witmer ,M. J. Singer. Measuring Presence in Virtual Environments: A Presence Questionnaire. Presence, 1998 - MIT Press

[21] H. T. Regenbrecht, T. W. Schubert, F. Friedmann. Measuring the Sense of Presence and its Relations to Fear of Heights in Virtual Environments. International Journal of Human-Computer Interaction. Pages 233-249, 1998

[22] K. M. Stanney, R. R. Mourant, R. S. Kennedy. Human Factors Issues in Virtual Environments: A Review of the Literature. Presence, vol. 7, pp. 327-351, August 1998.

APPENDIX A

PRESENCE QUESTIONNAIRE ITEM STEMS

- Major Factor Category
  - CF = Control Factors
  - SF = Sensory Factors
  - DF = Distraction Factors
  - RF = Realism Factors
- Subscales
  - INV/C = Involvement/Control
  - NAT = Natural
  - AUD = Auditory
  - HAPTC = Haptic
  - RES = Resolution
  - IFQUAL = Interface Quality
  - ITCorr = Pearson correlation coefficients between PQ item scores and the PQ Total Score

Table A.1: PQ Item Stems

Item	Stems	Factors	Subscale	ITCorr
1.	How much were you able to control events?	CF	INV/C	0.43
2.	How responsive was the environment to actions that you initiated (or performed)	CF	INV/C	0.56
3.	How natural did your interactions with the environment seem?	CF	NATRL	0.61
5.	How much did the visual aspects of the environment involve you?	SF	INV/C	0.48
7.	How natural was the mechanism which controlled movement through the environment?	CF	NATRL	0.62
10.	How compelling was your sense of objects moving through space?	SF	INV/C	0.51
12.	How much did your experiences in the virtual environment seem consistent with your real-world experiences?	RF,CF	NATRL	0.62
13.	Were you able to anticipate what would happen next in response to the actions that you performed?	CF	INV/C	0.43
14.	How completely were you able to actively survey or search the environment using vision?	RF,CF,SF	INV/C	0.59
18.	How compelling was your sense of moving around inside the virtual environment?	SF	INV/C	0.62
19.	How closely were you able to examine objects?	SF	RESOL	0.55
20.	How well could you examine objects from multiple viewpoints?	SF	RESOL	0.49
23.	How involved were you in the virtual environment experience?		INV/C	0.52
25.	How much delay did you experience between your actions and expected outcomes	CF	INV/C	0.41
26.	How quickly did you adjust to the virtual environment experience?	CF	INV/C	0.42
27.	How proficient in moving and interacting with the virtual environment did you feel at the end of the experience?	CF	INV/C	0.45
28.	How much did the visual display quality interfere or distract you from performing assigned tasks or required activities?	DF	IFQUAL	0.44
29.	How much did the control devices interfere with the performance of assigned tasks or with other activities?	DF,CF	IFQUAL	0.44
30.	How well could you concentrate on the assigned tasks or required activities rather than on the mechanisms used to perform those tasks or activities?	DF	IFQUAL	0.51

APPENDIX B

IMMERSIVE TENDENCY QUESTIONNAIRE ITEM STEMS

- Subscales
  - INVOL = Tendency to become involved in activities
  - FOCUS = Tendency to maintain focus on current activities
  - GAMES = Tendency to play video games
- ITCorr = Pearson correlation coefficients between ITQ item scores and the ITQ Total Score

Table B.1: ITQ Item Stems

Item	Stems	Subscale	ITCorr
1.	Do you ever get extremely involved in projects that are assigned to you by your boss or your instructor, to the exclusion of other tasks?		0.26
2.	How easily can you switch your attention from the task in which you are currently involved to a new task?		0.26
3.	How frequently do you get emotionally involved (angry, sad, or happy) in the news stories that you read or hear?		0.27
5.	Do you easily become deeply involved in movies or TV dramas?	FOCUS	0.49

Table B.1: ITQ Item Stems (Continued)

Item	Stems	Subscale	ITCorr
6.	Do you ever become so involved in a television program or book that people have problems getting your attention?	INVOL	0.47
7.	How mentally alert do you feel at the present time?	FOCUS	0.40
8.	Do you ever become so involved in a movie that you are not aware of things happening around you?	INVOL	0.56
9.	How frequently do you find yourself closely identifying with the characters in a story line?	INVOL	0.53
10.	Do you ever become so involved in a video game that it is as if you are inside the game rather than moving a joystick and watching the screen?	GAMES	0.55
13.	How physically fit do you feel today?	FOCUS	0.30
14.	How good are you at blocking out external distractions when you are involved in something?	FOCUS	0.46
15.	When watching sports, do you ever become so involved in the game that you react as if you were one of the players?		0.43
16.	Do you ever become so involved in a daydream that you are not aware of things happening around you?	INVOL	0.56
17.	Do you ever have dreams that are so real that you feel disoriented when you awake?	INVOL	0.50
18.	When playing sports, do you become so involved in the game that you lose track of time?	FOCUS	0.46
20.	How well do you concentrate on enjoyable activities?		0.49
21.	How often do you play arcade or video games? (OFTEN should be taken to mean every day or every two days, on average.)	GAMES	0.35
22.	How well do you concentrate on disagreeable tasks?		0.29
23.	Have you ever gotten excited during a chase or fight scene on TV or in the movies?	FOCUS	0.51
25.	Have you ever gotten scared by something happening on a TV show or in a movie?	INVOL	0.42
26.	Have you ever remained apprehensive or fearful long after watching a scary movie?	INVOL	0.31
28.	How frequently do you watch TV soap operas or docu-dramas?		0.28
29.	Do you ever become so involved in doing something that you lose all track of time?	FOCUS	0.49

APPENDIX C

VISUAL IMPORTANCE QUESTIONNAIRE

Prepared by Burak Çiflikli - Engineer / Havelsan
Supervisor: Assoc. Prof. Dr. Veysi İşler

This questionnaire is prepared to have a better understanding of the features effecting visual perception of military aircraft pilots. Questionnaire results will contribute to visual perception based solutions for visualization problems within simulation industry.

1. Hours of Flight

2. Hours of Flight with a Simulator

3. Flight Categories (Check all that apply)

[ ] Attack Helicopter
[ ] SAR or Transporter Helicopter
[ ] Jet Aircraft
[ ] Propeller Aircraft
[ ] Other

4. Average Flight Level

5. In what kind of training scenarios, the trainee should only use cockpit instruments without

observing outer world?

6. In what kind of training scenarios, the trainee should observe outer world?

7. Are there any objects that you pay extra attention in a mission (ex: a cave on the mountain)? If so please state the content of the mission.

8. Are there any aircraft categories that you pay extra attention during your flight?

9. What are the aircraft categories that you fly with during a wingman flight?

10. What are your threat avoidance methods?

11. What kind of behaviors of an aircraft attract your attention?

12. What kind of attributes of an aircraft attract your attention?

APPENDIX D

DEGRADED PRESENCE QUESTIONNAIRE ITEM STEMS

- Major Factor Category
  - CF = Control Factors
  - SF = Sensory Factors
  - DF = Distraction Factors
  - RF = Realism Factors
- Subscales
  - INV/C = Involvement/Control
  - NAT = Natural
  - AUD = Auditory
  - HAPTC = Haptic
  - RES = Resolution
  - IFQUAL = Interface Quality
  - ITCorr = Pearson correlation coefficients between PQ item scores and the PQ Total Score

Table D.1: Degraded PQ Item Stems

Item	Stems	Factors	Subscale	ITCorr
1.	How much were you able to control events?	CF	INV/C	0.41
2.	How responsive was the environment to actions that you initiated (or performed)	CF	INV/C	0.87
3.	How natural did your interactions with the environment seem?	CF	NATRL	0.79
5.	How much did the visual aspects of the environment involve you?	SF	INV/C	0.46
7.	How natural was the mechanism which controlled movement through the environment?	CF	NATRL	0.66
10.	How compelling was your sense of objects moving through space?	SF	INV/C	0.47
12.	How much did your experiences in the virtual environment seem consistent with your real-world experiences?	RF,CF	NATRL	0.62
13.	Were you able to anticipate what would happen next in response to the actions that you performed?	CF	INV/C	0.46
14.	How completely were you able to actively survey or search the environment using vision?	RF,CF,SF	INV/C	0.54
18.	How compelling was your sense of moving around inside the virtual environment?	SF	INV/C	0.57
19.	How closely were you able to examine objects?	SF	RESOL	0.56
20.	How well could you examine objects from multiple viewpoints?	SF	RESOL	0.80
23.	How involved were you in the virtual environment experience?		INV/C	0.40
26.	How quickly did you adjust to the virtual environment experience?	CF	INV/C	0.63
27.	How proficient in moving and interacting with the virtual environment did you feel at the end of the experience?	CF	INV/C	0.55
30.	How well could you concentrate on the assigned tasks or required activities rather than on the mechanisms used to perform those tasks or activities?	DF	IFQUAL	0.52

APPENDIX E

3D MODELS

E.1 F-22 Raptor

Figure E.1: F-22 Raptor

Table E.1: F-22 Dimensions

Developed by	Combat Simulation Project (CSP)
Length	18.90 m
Wingspan	13.56 m
Height	5.08 m

E.2 F/A-18 Hornet

Figure E.2: F/A-18 Hornet

Table E.2: F/A-18 Dimensions

Developed by	Delta3D
Length	17.1 m
Wingspan	12.3 m
Height	4.7 m

E.3 Mi-24 Hind

Figure E.3: Mi-24 Hind

Table E.3: Mi-24 Dimensions

Developed by	Delta3D
Length	17.5 m
Wingspan	6.5 m
Height	6.5 m

APPENDIX F

QUESTIONNAIRE RESULTS

F.1 ITQ Results

ITQ items are listed in the first column. The questionnaire item scores of the participants are presented in the following columns.

Table F.1: ITQ Results

	1	2	3	4	5	6	7	8	9	10
ITQ1	4	5	4	6	2	5	5	6	6	5
ITQ2	6	5	7	6	5	7	5	2	5	2
ITQ3	4	2	3	4	4	5	3	5	5	6
ITQ5	1	5	4	5	4	5	5	6	5	1
ITQ6	1	1	4	7	3	5	3	6	3	1
ITQ7	4	6	3	5	5	4	3	4	6	2
ITQ8	5	2	5	6	2	2	4	7	3	2
ITQ9	2	4	5	2	4	5	3	1	5	2
ITQ10	6	6	4	2	5	5	4	1	6	2
ITQ13	4	4	4	3	5	6	3	4	6	4
ITQ14	4	4	2	6	4	5	6	6	5	5
ITQ15	5	2	1	4	6	5	2	6	5	6
ITQ16	5	1	3	0	6	6	5	6	2	3
ITQ17	4	1	1	3	2	6	4	1	5	1
ITQ18	6	2	2	5	7	7	6	6	5	6
ITQ20	4	5	6	6	7	6	5	5	6	6
ITQ21	7	7	1	1	1	6	1	1	5	3
ITQ22	3	2	3	3	2	4	3	6	4	3
ITQ23	5	5	6	5	5	5	4	5	6	2
ITQ25	7	4	1	5	4	6	3	5	4	1
ITQ26	4	6	2	4	1	5	4	2	4	1
ITQ28	1	1	6	2	6	2	1	3	2	1
ITQ29	6	7	4	4	5	5	5	4	7	5
Total Score	98	87	81	94	95	117	87	98	110	70

Table F.1: ITQ Results (Continued)

11	12	13	14	15	16	17	18	19	20	AVG	MIN	MAX
5	5	5	4	7	5	6	6	4	3	4.9	2	7
5	3	6	3	7	6	4	5	6	6	5.05	2	7
7	5	7	6	5	4	4	7	4	2	4.6	2	7
3	1	4	7	4	6	5	5	6	4	4.3	1	7
7	1	5	5	5	6	2	6	4	4	3.95	1	7
7	5	7	7	4	5	5	5	3	4	4.7	2	7
7	1	2	7	2	4	6	5	4	5	4.05	1	7
1	1	4	7	2	5	5	3	2	4	3.35	1	7
2	1	1	7	4	3	5	5	3	4	3.8	1	7
3	5	7	5	3	3	6	4	4	4	4.35	3	7
6	5	2	2	5	4	6	4	5	3	4.45	2	6
6	1	6	7	4	6	2	6	3	3	4.3	1	7
6	3	6	7	1	5	2	1	4	3	3.75	0	7
6	1	1	2	1	1	5	4	4	2	2.75	1	6
5	5	1	7	3	4	6	4	5	2	4.7	1	7
7	7	5	7	7	7	7	6	5	5	5.95	4	7
1	1	1	4	1	2	4	6	2	2	2.85	1	7
1	3	1	1	3	3	4	3	4	3	2.95	1	6
1	1	2	7	3	6	2	3	4	5	4.1	1	7
7	3	7	7	4	6	6	6	3	2	4.55	1	7
7	1	4	7	1	4	7	7	4	3	3.9	1	7
5	1	1	1	1	3	4	1	2	6	2.5	1	6
7	5	2	6	3	3	4	3	4	4	4.65	2	7
112	65	87	123	80	101	107	105	89	83	94.45	65	123

F.2 PQ1 Results

PQ items are listed in the first column. The questionnaire item scores of the participants to the first test session are presented in the following columns.

REV Total is the new total score after non-corralated items are excluded.

Table F.2: PQ1 Results

	1	2	3	4	5	6	7	8	9	10
PQ1	5	5	3	5	6	5	2	4	5	2
PQ2	3	4	3	3	5	4	2	4	6	2
PQ3	2	3	3	4	5	5	4	4	6	2
PQ5	2	4	4	4	5	4	3	4	5	5
PQ7	3	4	2	3	5	4	2	4	5	2
PQ10	4	3	1	4	2	4	5	5	5	3
PQ12	3	2	2	3	3	3	3	3	4	1
PQ13	2	5	5	5	5	5	6	3	6	2
PQ14	5	2	4	4	6	5	4	3	4	4
PQ18	4	4	3	3	6	4	3	4	4	2
PQ19	4	3	3	4	6	5	4	4	4	5
PQ20	3	3	2	4	6	4	6	5	5	2
PQ23	4	4	2	4	6	4	4	3	4	5
PQ25	6	5	3	6	3	6	6	3	3	6
PQ26	4	2	4	6	6	3	3	4	5	1
PQ27	3	3	2	4	5	4	4	2	4	1
PQ28	2	6	6	2	5	2	3	5	3	3
PQ29	2	6	3	2	3	6	6	4	3	7
PQ30	4	5	4	5	7	7	4	4	6	6
Total Score	65	73	59	75	95	84	74	72	87	61
REV Total	55	56	47	65	84	70	59	60	78	45

Table F.2: PQ1 Results (Continued)

11	12	13	14	15	16	17	18	19	20	AVG	MIN	MAX
1	1	4	2	2	5	3	3	3	3	3.45	1	6
5	3	2	5	3	4	5	4	3	3	3.65	2	6
5	3	2	5	3	1	3	3	4	3	3.5	1	6
3	5	1	2	2	2	4	4	4	4	3.55	1	5
5	3	3	4	3	4	5	4	3	3	3.55	2	5
2	1	2	3	2	3	5	2	2	2	3.0	1	5
1	1	3	3	3	2	3	3	4	3	2.65	1	4
5	7	3	7	5	6	6	5	4	4	4.8	2	7
1	3	2	1	3	2	4	4	4	4	3.45	1	6
1	1	2	7	3	4	5	5	3	3	3.55	1	7
5	3	4	6	1	2	4	4	3	3	3.85	1	6
2	3	2	7	3	4	3	3	5	2	3.7	2	7
2	1	3	7	4	2	5	5	4	3	3.8	1	7
5	5	6	5	4	6	6	3	3	4	4.7	3	6
1	1	4	7	6	3	7	5	4	6	4.1	1	7
7	1	2	7	3	3	5	6	3	4	3.65	1	7
1	1	4	2	2	6	1	4	3	6	3.35	1	6
1	3	2	1	2	3	4	2	6	6	3.6	1	7
7	3	3	7	5	4	7	4	4	4	5.0	3	7
60	49	54	88	59	66	85	73	71	70	71.0	49	95
53	40	42	80	51	51	74	64	57	54	59.25	40	84

F.3 PQ2 Results

PQ items are listed in the first column. The questionnaire item scores of the participants to the second test session are presented in the following columns.
REV Total is the new total score after non-corralated items are excluded.

Table F.3: PQ2 Results

	1	2	3	4	5	6	7	8	9	10
PQ1	3	6	4	7	6	6	4	5	6	5
PQ2	4	5	5	6	6	6	3	5	6	4
PQ3	3	4	3	6	6	6	4	5	6	2
PQ5	3	5	4	4	6	5	4	4	7	6
PQ7	4	5	5	5	5	5	3	5	6	3
PQ10	5	3	4	6	4	6	4	5	5	4
PQ12	4	2	6	4	5	6	3	6	5	3
PQ13	3	6	4	5	5	5	6	6	6	5
PQ14	4	3	2	4	6	6	3	6	5	5
PQ18	5	4	4	4	5	6	3	6	4	4
PQ19	4	4	3	5	6	6	4	6	6	5
PQ20	4	4	2	6	7	6	4	6	6	3
PQ23	5	5	4	3	7	2	3	6	4	5
PQ25	4	4	6	2	3	6	3	2	2	2
PQ26	3	4	6	5	6	6	6	6	6	2
PQ27	4	5	5	5	6	5	6	6	4	4
PQ28	4	6	2	2	2	5	2	1	3	2
PQ29	2	6	4	2	3	5	4	2	3	3
PQ30	5	5	6	5	7	6	4	5	6	5
Total Score	73	86	79	86	101	104	73	93	96	72
REV Total	63	70	67	80	93	88	64	88	88	65

Table F.3: PQ2 Results (Continued)

11	12	13	14	15	16	17	18	19	20	AVG	MIN	MAX
3	3	5	2	2	3	3	4	4	4	4.25	2	7
6	4	5	7	2	5	5	4	3	4	4.75	2	7
6	3	6	6	3	3	4	3	4	3	4.3	2	6
5	4	5	5	3	4	4	4	5	5	4.6	3	7
5	5	6	7	5	6	6	4	3	4	4.85	3	7
5	5	6	4	4	6	6	3	2	3	4.5	2	6
5	4	6	7	4	5	4	4	3	4	4.5	2	7
7	4	6	7	6	6	6	5	5	3	5.3	3	7
1	5	4	4	2	5	4	4	4	2	3.95	1	6
1	4	4	7	3	6	5	5	3	3	4.3	1	7
6	4	5	5	5	3	4	4	4	3	4.6	3	6
5	4	5	7	4	5	4	4	4	2	4.6	2	7
7	4	5	7	3	6	5	5	4	3	4.65	2	7
1	2	2	1	1	4	5	3	2	3	2.9	1	6
7	5	6	7	3	6	7	6	6	4	5.35	2	7
7	4	5	7	3	4	6	6	5	2	4.95	2	7
2	3	2	2	1	2	1	3	2	6	2.65	1	6
2	2	3	1	1	3	3	2	6	6	3.15	1	6
5	5	6	7	7	7	7	5	4	3	5.5	3	7
86	74	92	100	62	89	89	78	73	67	83.65	62	104
81	67	85	96	59	80	80	70	63	52	74.95	52	96

APPENDIX G

ENTITY CONTROL PACKET

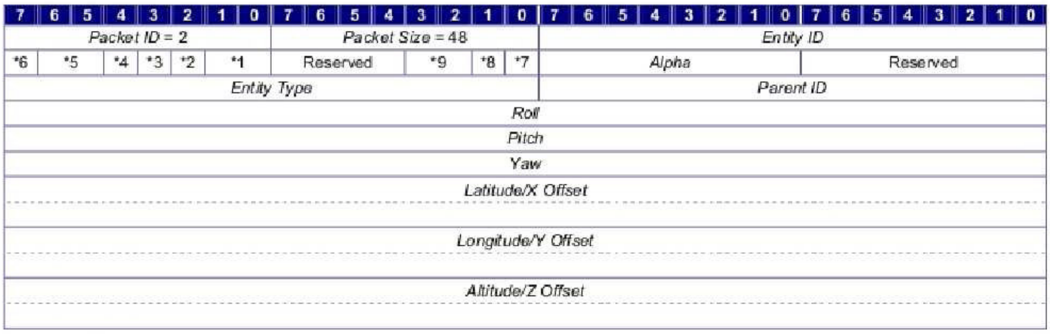

Figure G.1: Entity Control Packet

Table G.1: Entity Control Packet Parameters

Packet ID	2
Packet Size	48
Entity ID	unsigned int16
Entity State	0:Inactive/Standby 1:Active 2:Destroyed
Attach State	0:Detach 1:Attach
Collision Detection Enable	0:Disabled 1: Enabled
Inherit Alpha	0: Not Inherited 1: Inherited

Table G.1: Entity Control Packet Parameters (Continued)

Ground-Ocean Clamp	0: No Clamp 1: Non-Conformal 2: Conformal
Animation Direction	0: Forward 1: Backward
Animation Loop Mode	0: One-Shot 1: Continuous
Animation State	0: Stop 1: Pause 2: Play 3: Continue
Alpha	unsigned int8
Entity Type	unsigned int16
Parent ID	unsigned int16
Roll	float (-180 to 180)
Pitch	float (-90 to 90)
Yaw	float (0 to 360)
Latitude	double float (-90 to 90)
Longitude	double float (-180 to 180)
Altitude	double float