University of Alberta

Library Release Form

Name of Author: Oscar Ernesto Meruvia Pastor

Title of Thesis: Level of Detail Selection and Interactivity

Degree: Master of Science

Year this Degree Granted: 1999

. . . .

Oscar Ernesto Meruvia Pastor
51 Sur 5112
Fracc. Estrella del Sur
C.P. 72190
Puebla, Pue.
Mexico

Date: . . . .

University of Alberta

Level of Detail Selection and Interactivity

by

Oscar Ernesto Meruvia Pastor

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 1999

Para mi mama por todo tu apoyo, amor y cariño.
Para mis hermanos y aquellos quienes me han apoyado en el logro de este titulo.
(To my mom for all your support, love and care.
To my brothers and those who have helped me in the completion of this work.)

Abstract

The amount of detail contained by a 3D model or scene can easily exceed a system's capability to render fully detailed images at interactive rates. Existing level of detail selection techniques allow control of this detail. During interaction, one set of control techniques reduces visual detail and limits the amount of visual error introduced; these control techniques do not consider system responsiveness and delay. Another set of control techniques limits delay, allowing smooth interaction, but ignoring loss of visual detail.

In this thesis, two level of detail selection techniques that balance visual with temporal accuracy are proposed. The speed-based technique balances visual with temporal accuracy according to interaction speed. The disparity-based technique balances visual with temporal accuracy based on the spatial difference between the displayed and input positions. Both techniques have been implemented for 3D rotation tasks. Experimental results show that disparity-based control effectively improves rotation performance under conditions of delay.

Acknowledgements

I am specially thankful with my supervisor for being very supportive, critical and helpful all the time. Thanks, Ben!

I also want to give thanks to:

- Lourdes for being so supportive all the time, for your company, friendship and for the things to come, gracias chiquita! :)
- Dima and Michael for being excellent friends and roommates, for all those adventures in the Rockies, and for making my stay in Canada much more enjoyable.
- Dr. Jorge Sierra for your unconditional support in every stage of my career.
- The people at the graphics lab for your contributions, lunch times and technical support.
- The grad and undergrad students who participated as guinea-pigs.
- The guinea-pigs of the pilot studies who were subjects of extreme testing conditions, program crashes and generously helped us.
- The Department for providing such a great learning experience.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Interactive 3D graphics .

Real-time three-dimensional graphics (3D graphics) has been present from the very first days of Computer Graphics:

"The first CIG (Computer Image Generation) device was built by GE's Electronics Laboratory in the late 1950's. It used a caligraphic display and analog circuitry to produce a repeated pattern over a ground plane. Its main significance was to prove the feasibility of real-time graphics."-[Scha83]

The first applications that used interactive 3D graphics were flight simulators meant for aviation and aeronautic training purposes. The increase of computing power and the fall in costs of computer equipment has enabled the use of real-time 3D graphics in a wider variety of fields: navigation or driving systems, virtual reality (VR), human training, architectural design, computer aided design (CAD), computer animation, games, teleoperation, medical and scientific data visualization among others.

However, it is still challenging for todays' computer systems to achieve high-quality interactive 3D graphics, the main reasons being:

1. *3D graphics rendering requires a significant amount of computational power.* To make a 3D image look realistic, it is necessary to simulate the complex properties of the objects and the environment being represented. These properties include object's shape, color, reflectivity, transparency and/or texture; scene properties include illumination, shadowing, viewing parameters and special effects such as fog and motion blur. These properties must be simulated for all the

objects in the scene. The inclusion of each of these objects increases scene complexity, sometimes by several orders of magnitude (if, for example, reflections are calculated).

2. *The demands regarding the simulated situations are high.* The purpose of some 3D graphics applications, like Virtual Reality, is to simulate real life situations in complex environments. The need then arises to implement better communication channels that provide more natural feedback (like force feedback) and better simulate the environment's behaviour (like character animation). At the current state of technology, only the minimal requirements for some applications have been met for a variety of situations, but much work remains.

3. *Interface design is challenging for 3D interaction.* 3D interaction implies the existence of six degrees of freedom (DOFs): three to define position and three to define orientation. It is still an open question whether all degrees of freedom can or should be used at the same time or in a certain fashion. Currently, there are various interfaces for 3D graphics systems but the need to produce adapted interfaces according to the nature of the task being simulated still poses challenges in interface design.

4. *Interactive systems are strictly time-constrained.* For interactive 3D graphics, the images displayed need to be updated every certain amount of time to reflect changes in the environment or changes due to user input. It is usually acceptable to update images at rates of 10 to 20Hz (images per second), which allows for 50 to 100 msec to display an image. Simulation and rendering at the highest levels of detail is often not possible within these time constraints.

While each of these challenges needs to be addressed to improve the quality of interaction with 3D graphics systems, in this thesis we[1] focus on the last issue.

[1]In this thesis the term "we" will be used to refer to the author and his supervisor, in acknowledgment of the team effort that lead to completion of this work.

1.2 Controlling the trade-off between delay in interaction and visual detail

Visual detail and delay during interaction are important factors to take into account by designers of interactive 3D graphics systems.

As mentioned before, high visual detail is necessary to improve the level of realism of the objects and scenes displayed, and delay during interaction in a 3D graphics system is very limited.

Both factors compete for the system resources and designers are typically forced to chose levels of visual detail and delay during interaction that are less than ideal.

It is hard to assess what the appropriate balance between visual detail and delay during interaction is. A first step towards this assessment is to implement a precise control of this visual-temporal trade-off.

In this thesis we present two techniques that achieve this control. These techniques were designed to balance the visual complexity of the image being rendered with the rate of user interaction.

1.3 Thesis overview

Effects of delay in interactive systems To provide a clear explanation of the techniques presented in this thesis, we need to understand the nature of delay and its effects.

In Chapter 2, we describe the delay present during interaction with 3D graphics systems, in terms of the processing stages that take place from the occurrence of an input event to its eventual display. Next, we examine the effects of delay on various tasks. Finally, we describe the ways in which delay can be reduced or compensated.

Level of detail management 3D models are normally rendered using a polygonal mesh that approximates the object to be rendered. The complexity of a model is described by the term 'level of detail' (LOD). Typically, level of detail is measured by the number of polygons that constitute a model.

Level of detail research in computer graphics is concerned with methods that reduce the complexity of the models while preserving the features that are most relevant to the application. These techniques are typically able to produce a set of simplified representations at multiple resolutions (levels of detail) out of a complex model.

LOD management techniques determine which LODs to select during interaction with 3D graphics systems. Most LOD management techniques aim at the elimination of detail in regions of a model where it is clearly not needed, like the back-facing regions. Some other techniques aim at refining critical regions of a model, like the silhouette.

Existing LOD and LOD management techniques are reviewed in Chapter 3.

Controlling the spatio-temporal error During interaction, LOD selection techniques deal with different types of error:

1. Simplification error: Is the difference between a model and one of its simplified representations. Any reduction in model complexity will produce a certain amount of error due to simplification. This error can be measured in the 3D model space.

2. Spatial error: Is the difference between the displayed position of an object and the position where this object should be, according to the system input and the simulation. This error occurs in 3D space.

3. Visual: This is either the simplification error or the spatial error measured once the objects have been projected to the screen-plane. Some LOD control techniques are based on measures of this error. Such a control is practical, because it reduces measurement of spatial error to measurement of pixels in the screen-plane.

4. Temporal: this error is basically due to the delay that occurs between an input event and its eventual display, and can be understood in terms of the age of the sample being displayed, measured from the time of occurrence of the corresponding input event.

5. Spatio-Temporal error: Is the amount of spatial error introduced when temporal error is not accounted for.

6. Visuo-Temporal error: Is the amount of visual error introduced when temporal error is not accounted for.

7. Total error: Is the sum of the spatio- or visuo-temporal error and the visual error due to simplification.

Most LOD control techniques reduce visual detail and limit the amount of visual error introduced; these techniques do not consider delay during interaction.

Other LOD control techniques limit delay in interaction (temporal error), but do so regardless of the loss of visual error that might be introduced.

In Chapter 4 we present two techniques that limit the amount of spatio-temporal error introduced during interaction.

With the first technique, level of detail is selected according to the user's speed of interaction. When a user moves fast, a low LOD representation is used, so that a high number of intermediate samples is displayed during motion, thereby reducing sample age and increasing the perception of visual continuity. When a user moves slowly, a high LOD representation is used, since there is less change.

The second technique proposes the selection of LOD based on a measure of disparity between the displayed model position and the position where the model should be according to the user input. During interaction, the spatial error tends to increase as the user's input changes over time. Under this technique, a lower representation is used whenever the difference between the latest user input and the model position in the display exceeds a certain threshold. This lower LOD representation is almost immediately rendered and provides a coarse approximation of the latest model position, thereby limiting spatio-temporal error. This technique increases the fidelity between the user's input and the position of the model being displayed.

3D Rotation This work presents an implementation of the concepts of velocity- and disparity-based LOD control in the domain of 3D object rotation using mice. In Chapter 5 we discuss the interfaces developed for 3D rotation using mice. We also review two user studies with these interfaces, which provide a framework for our own experimental design.

3D rotation using a mouse offers a good starting point to test our hypothesis. LOD selection based on control of spatio-temporal error can be easily interpreted in terms of 3D rotation. The technique based on interaction speed can be interpreted as angular speed based, where angular speed is measured in degrees per second during rotation. The technique based on spatio-temporal error can be interpreted as angular disparity based, where angular disparity is measured by the amount of degrees required to accomplish a 3D rotation between the displayed orientation and an orientation corresponding to a subsequent input sample.

Experimental analysis Two experiments were performed in order to test the effectiveness of the disparity-based control under various amounts of delay. In both experiments the task was 3D orientation matching.

The first experiment compared the disparity-based technique to two other techniques: one that produced no visual error and ignored spatio-temporal error, and one that limited spatio-temporal error as much as possible and ignored visual error. Results showed that the angular disparity switch effectively reduces the effects of delay on performance, specially when delay is high (at low frame rates). Users indicated a preference for our technique.

In the second experiment we examined the effects on performance of various thresholds and low LOD representations when using the disparity-based LOD control. The experimental analysis is presented in Chapter 6.

Conclusions and Future Work Chapter 7 presents conclusions about the results obtained this thesis and areas for further research.

Chapter 2

System Responsiveness and Human Performance

A factor present in all 3D graphics systems is delay. A significant amount of research has been dedicated to the study of the effects of delay on human performance on various conditions. In this section a formal framework to quantify the effects of delay will be discussed, followed by a review of research on the effects of delay on performance.

2.1 Delay and System Responsiveness

System responsiveness (SR) is a measure of delay. It is defined as the time between a user action and the resulting response of the systems [Wats98].

System responsiveness can be decomposed based on the events that occur during the lapse that defines it. In the following definitions we assume we are working with a graphics system made out of three subsystems: the user and his/her actions, a tracking or input sampling system, and a rendering system for display (see figure 2.1).

Event-sample delay. A user event can generate several input samples, but only one input sample is displayed per frame. Normally, the sample taken by the rendering subsystem is the last sample available, so it is very likely that several of the first input samples will be discarded. The lapse of time between the moment a user event starts and the moment a sample of this event is taken by the rendering subsystem is the event-sample delay, which is on average one half of the frame time.

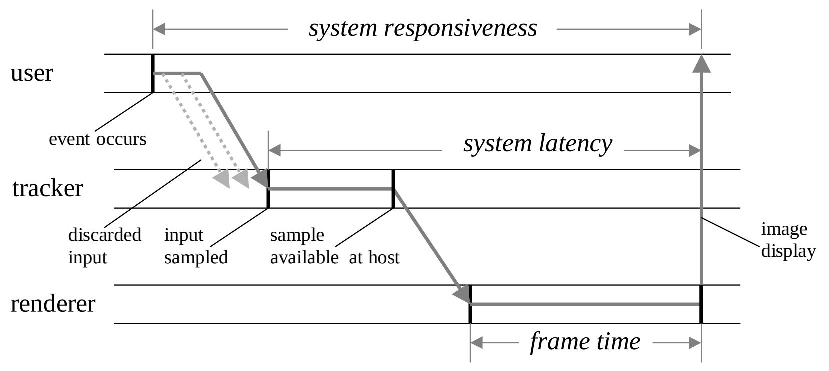

Figure 2.1: System Responsiveness, adapted from Watson (1998)

System latency is the lapse of time that occurs between the moment a user event is sampled and the corresponding image is displayed by the graphics system. System latency can also be viewed as the age of each sample presented on the display.

System latency includes the lapse of time between the moment a user's action is sampled with an input device and the time this motion is processed and transmitted as input data to the rendering software. This lapse of time varies according to the complexity of the input device and can be a significant element of delay.

System responsiveness contains system latency, because *SR* is measured from the moment a user event has occurred and the moment a sample of this event has been displayed.

Frame time is the time elapsed between displayed samples.

Frame rate (FR) is the inverse of frame time. It is simply the amount of samples displayed per second. Frame rate should not be confused with *refresh rate* of a graphics system. The refresh rate is bounded to the hardware properties of the display system, and is the number of images presented per second. Frame rate is bounded by the system's refresh rate.

Sources of delay

The sources of delay can be categorized in two types: delay that affects system latency, and delay that affects frame rate [Brys93]. In any case, both kinds of delay affect system responsiveness and introduce spatio-temporal error.

In figure 2.2, the effects of these two types of delay are shown, for movement of an object in one dimension only. The first case illustrates delay that affects system

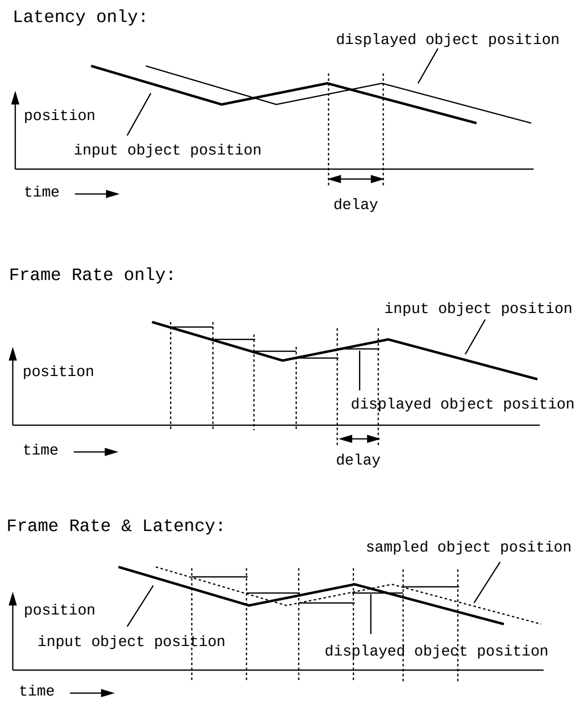

Figure 2.2: The different types of delay, adapted from Bryson (1993)

latency. In the second case, only delay due to frame rate is represented. The third case illustrates how the two kinds of delay are combined in a real graphics system, where some sources of delay act in the input subsystem and others act in the rendering subsystem.

To identify each of the sources of delay, it is necessary to take a look at the processing loop that occurs in 3D graphics systems.

[Mine95] decomposed the Head Mounted Display (HMD) Pipeline in four main stages (see Table 2.1). These stages can be extrapolated to most 3D graphics interactive systems, because there is a correspondence between the elements of the HMD Pipeline and every other system.

The delay occurring in each of these stages determines the system responsiveness of an HMD-based graphics system.

Spatial position/orientation sensing is the determination of the position and orientation of the user's head using one of the available tracking devices. Generalizing this to any other graphics system, this is the stage where the input from the user is sampled. Sources of delay in this stage are all due to the complexity of generating this

HMD Pipeline Stage	**Delay Component**
Spatial position/orientation sensing	Tracking System delay
Application host Processing	Application Host delay
Graphics Computation	Image Generation delay
Image Display	Display System delay
Synchronization (*Wloka*)	Synchronization delay

Table 2.1: HMD Pipeline Delay Components, adapted from Mine (1995).

data and leaving it available to the application, taking mechanical, optical, acoustic or magnetic signals, converting them into an input sample and transmitting them to the application host. Tracker input delay is known to be around some 50-80 msecs. Delay in this stage affects system latency (and SR), not frame time.

Application host processing includes the collection of sensor data and any other external input to the application. This stage also includes the execution of any simulation process that is inherent to the application being executed. Examples are the control of the behaviour of characters in an animation or the calculation of the kinematics of a mechanical system. Delay in this stage affects system latency and might also affect frame rate, depending on whether the rendering subsystem waits for this information to display an image or whether it renders images asynchronously.

Graphics computation is the generation of the left and right eye images necessary for the presentation of a 3D image. If the application does not display in stereo, only one of the images is necessary. In HMD systems, the generated images depend both upon the input samples and the simulation information generated by the application process. In this stage, delay depends on the number of primitives being rendered and affects frame rate and system latency.

Image display is the presentation of the generated images to the user via the HMD displays. In the general case is the presentation of the image at the display. Delay here depends on the refresh rate of the display, which is typically 60 to 72 Hz, accounting in this case for 13.8 to 18 msec delay. This delay affects latency and frame rate.

Synchronization [Wlok95] considers the additional synchronization delay introduced in a parallel processing system where each of the processing stages runs independently. Wloka defines this delay a the total time data is waiting in-between stages without being processed. Synchronization delay is inversely proportional to the throughput rates of the various stages. On the other hand, using multiple processors reduces delay significantly because it allows every stage to proceed at maximum throughput rates.

Event-sample delay Wloka and Mine do not consider event-sample delay, discussed in the earlier section, but event-sample delay is always present and is in average half of a frame time.

In this research we have chosen a system configuration where the expected delay in the first processing stage is minimal, because we use the mouse as the input device. The delay under study in this research is the one that occurs due to graphics computation and image display.

2.2 Techniques to reduce delay

Ideally, we would like to reduce delay in all stages of the human-computer processing loop.

Prediction

Prediction methods take into account the history of the input data to predict the future user's position. Prediction methods help to reduce effective delay occurring in the spatial position/orientation sensing stage.

This approach has been implemented by [Lian91] and [Wlok95] using Kalman filters and a model of the user's (head) motion. [Azum94] includes gyros and accelerometers to aid both orientation and position prediction.

These methods work well in practice, but they might introduce spatial inaccuracies under the following three conditions:

1. The user input device throughput is too low.

2. The prediction is done too far in the future.

3. The input device acceleration is too high.

Late sampling

In order to provide the most updated image according to user input, sampling has to be done as late as possible. This meaning that if there are some computations that are independent of the user input (of the Host Application stage) they need to be done first, leaving at the end the computations that depend on user input [Ware94].

Parallel processing

Multiple processors can be used either for pipelining the application, running several instances of it, or assigning one CPU for each processing stage.

The latter use is the most common. According to [Wlok95] there are four main advantages to parallel processing:

1. The user input device is independent from all other stages and thus runs with maximum throughput, allowing prediction.

2. Rendering proceeds at maximum throughput reducing delay in the rendering stage.

3. The distribution of synchronization lag is narrower, because all stages work at maximum throughput.

4. If the user-input is allowed to communicate directly with the rendering process, a cursor that presents low delay can indicate the user's position in support of the application data.

Reduction of synchronization delay Synchronization delay can also be reduced if the scheduling of the processing stages is coordinated appropriately. [Wlok95] presents a technique that minimizes this kind of delay.

Independence between input devices [Ware94] suggest to separate head tracking delay from hand tracking delay. In their experiment head motion occurs less frequently and to a lesser degree than hand motion, so they propose to sample the head position first, draw most of the scene, and then sample the hand position and draw the 3D cursor and the target. The idea is to do the latest sampling for the most important input sources (the ones that change the most), on the assumption that they might be relatively small parts of the 3D environment. Additionally, they suggest to create higher update rates for the cursor and other relevant objects in the scene, if possible.

Reduction of network delay

Nowadays most graphics applications work in distributed environments which transfer information through the network.

Networks are a decisive factor for the first three stages of input data processing. It is desirable that network delay and its variation are kept at a minimum, especially when delay is high [Shin99]. Also, the network protocol should be sensitive to the nature of the application, since it is not always necessary to ensure arrival of input samples in the correct order, but to ensure prompt arrival of the latest input information.

Reduction of application host delay

Delay occurring in the application host stage can be reduced by simplified computation schemes, code optimization or by parallelizing the application.

Reduction of image complexity

Delay occurring in the graphics computation stage is due to time consumed for primitive rendering. This type of delay has been an ever-present issue in computer graphics, as described in [Scha83]. The most effective way to reduce delay in this stage is by sending a limited amount of primitives to the renderer. Ways to accomplish this are studied with further detail in Chapter 3.

2.3 Effects of Delay on User Performance

To study the effects of delay one needs to simulate delay in real world systems with as much fidelity as possible. Most studies control the system's mean system responsiveness (MSR), where a certain value of MSR is desired, and variation in MSR is kept at a minimum. However, in real systems, system responsiveness (SR) is variable. Some studies on delay take this into account and control not only MSR, but also the variation in SR, by measuring the standard deviation of SR (SDSR). For the sake of simplicity, the reader should assume that in the following studies only MSR is controlled. If a specific study analyzes the effects of SDSR, it will be explicitly mentioned.

2.3.1 Control of system responsiveness.

To make appropriate simulations of the different situations where delay is present, careful control of system responsiveness needs to be done. There are three ways to affect SR by means of software [Wats97b]:

1. *Latency-only manipulation (lom)* affects system latency (and SR), but not frame rate. It simulates effects of delay in the input device or interfacing software, and can be done by queuing input samples and delivering them to the rendering subsystem after some controlled delay (see figure 2.2, top).

2. *Frame-only manipulation (fom)* affects frame rate only (and SR). It simulates a change in the rendering or simulation that is not dependent on the input, but on the level of complexity of the scene or object, like increasing the level of detail in an animation. It can be achieved by adding delay before the receipt of the input sample (see figure 2.2, middle).

3. *Frame-latency manipulation (flm)* affects system latency and frame rate. It is achieved by adding a software delay in the rendering system between the receipt of an input sample an its eventual display. It simulates changes in scene complexity due to a change in user input, as when walking in a virtual environment with varying scene complexity. *flm* is also achieved when visual detail is manipulated during interaction (see figure 2.3). In this research we

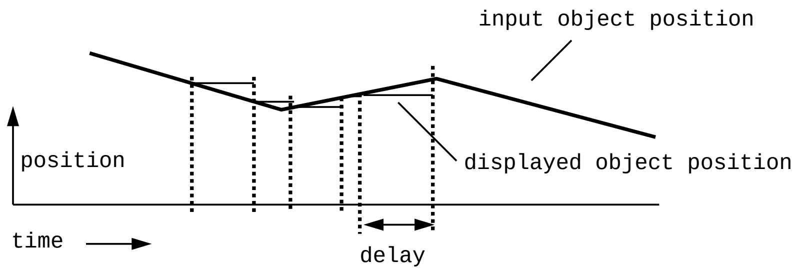

Figure 2.3: A system controlled with frame-latency manipulation

explore the idea of balancing delay and visual detail and hence perform *flm*. Our detail control techniques are described in Chapter 4.

2.3.2 On the variety of tasks in 3D graphics systems

Several kinds of tasks can be found when working with 3D graphics systems. Different tasks require different levels of feedback. For example, prompt feedback is more relevant when the task involves constantly following a moving target than when the task involves just moving an object to a single target location.

The interaction of a user with the system can be described as a sequence of action-response loops that occur from the start of a task until its completion. In this case, the task is defined as a closed-loop task.

In other cases the interaction may involve a single sequence of observation, appropriate prediction and an action. This kind of tasks are categorized as open-loop.

[Wats97b] describes three major groups of tasks when working in VR systems: catching, placement and tracking.

1. Catching tasks require the acquisition of a target in an open-loop fashion, normally, a prediction is done about the target's position and a trigger is activated to obtain the target.

2. Placement involves moving an object to a target position. It relies mostly on feedback of the user's actions, and requires less prediction than catching.

3. Tracking is the continuous pursuit of a moving target. It is the most difficult task since it involves prediction about the target position and requires continuous feedback to be successfully performed.

In this thesis we have chosen to use 3D rotation for our experimental analysis (see Chapter 6). The 3D rotation task used in this thesis is a form of placement task. During trials, a user is presented two representations of an object at different orientations. The task is orientation matching, and occurs under several conditions of delay. Two representations at different levels of detail will be used during interaction: full detail and detail. The fully detailed representation of a model will be subject to the effects of delay discussed in this chapter. According to a control criteria explained in Chapter 4, we will select either the fully detailed representation or the coarsely detailed representation, enhancing system responsiveness depending on user input. We expect to find in our work the same trend as the one observed in the reviewed literature, that delay affects performance in a linear way. That is, we expect longer completion times for conditions where delay is high and shorter completion times for situations where delay is low. We also believe that delay will vary with difficulty. For the rotation task, it is expected that delay will have a stronger effect when the difference between the two orientations is large than when it is small. Large differences in orientation will require more iterations of the human-processing loop; hence, they will be more subject to the effects of delay than small rotations.

2.3.3 Delay in placement and catching tasks

Fitts' law and delay Fitts' law [Fitt54] is used by many researchers in 3D graphics to appropriately model effects of SR on performance. In the original formulation it describes a linear relationship between task difficulty and placement time in one dimensional placement tasks:

$$PT = a + b\log_2(2D/W) \tag{2.1}$$

where PT is placement time, D is the distance to the target, W is the width of the target, and a and b are coefficients obtained through linear regression on experimental data. $\log_2(2D/W)$ is also referred to as the Index of Difficulty (ID).

The ratio of ID/PT is called index of performance (IP) and it is used to measure the effectiveness of an input device or system, because it captures the amount of difficulty per time unit that can be handled with while using a certain device or system. IP is remarkably stable across devices and tasks [Mack93].

The Fitts' law has been adapted to explain the effects of delay on task performance. MacKenzie and Ware, [1993] presented a study using a placement task of the Fitts' type in 2D using the mouse. The effects of system latency (lom) were measured for speed, accuracy and performance (using Fitts' IP).

A set of $4 \times 3 \times 4$ combinations of latency *(8.3, 25, 75 and 225 msec)*, distance *(96, 192 and 384 pixels)*, and target width *(6, 12, 24 and 48 pixels)* were tested.

The results indicated main effects for latency on movement time, error rate and IP. Error rates were especially bad at the highest latency of 225msec. Task difficulty (ID) and the interaction $latency \times ID$ had significant effects on movement time and error rate. As tasks became more difficult IP decreased, especially at the highest levels of latency *(75 and 225msec).*

For this experiment, MacKenzie proposed the following model as a variation of Fitts' law which integrates delay with ID in a 2D task:

$$PT = 230 + (169 + SystemLatency)ID \tag{2.2}$$

this model explains 93.5% of the variance (r^2=0.935) in placement time (PT).

Latency and frame rate in Fish Tank VR Ware and Balakrishnan [1994] presented a study of the effects of latency *(lom),* frame rate *(flm* and *fom),* direction of movement in 3D space, and stereoscopic images by using Fish Tank VR systems.

Fish Tank VR is a display where a conventional monitor is used to create a VR image localized around the screen; glasses are used to create stereoscopic vision, and the user's head position is tracked in real time to ensure that a correct perspective view is obtained. The effect obtained is like looking at a 3D scene through a window.

Three experiments were reported. In one experiment, the task was one dimensional placement (the target was composed of two parallel square planes, like a pizza box); in the other two experiments the task was 3D placement (the target was a 3D cube). Placement for each experiment occurred either along the viewing plane (horizontal) or in depth (into the screen).

For the experimental analysis the authors used the following model to explain variation in placement times:

$$PT = a + c(b + SystemDelay)ID. \tag{2.3}$$

where *SystemDelay* represents the delay (in *SR*) caused by the graphics system (controlled through *lom, flm* or *lom*); a represents the sum of the initial response time and the time required to confirm the acquisition of the target; b represents the human processing time required to make a corrective movement; and cID represents the average number of movements (or movement corrections) required to acquire the target, in other words, the number of times the human-machine processing loop is executed.

In the first experiment, latency caused by head tracking delay was combined with latency caused by hand tracking delay (both delays *lom*). Head delay showed no significant effects on performance, while hand delay was significant. This is probably due to the fact that head motion is not as relevant as hand motion for the placement task in the Fish Tank setup. The model in equation (2.3) was somewhat successful in explaining variation in PT ($r^2 = 0.86$).

In the second experiment, the target was modified to be a 3D cube, instead of a pair of 2D planes. Latency showed significant main effects again, and the model for PT explained 95% of the variance (r^2) for all the conditions, except the ones with the smallest target size, which were excluded from the analysis.

In the third experiment, effects of frame rate (*fom*), latency-only and frame-latency manipulation *(lom* and *flm)* were compared. The effects of delay were modeled producing very similar models that correlated with r^2 from 0.90 up to 0.99 for the different conditions; the regression of all data combined showed $r^2 = 0.89$. These results indicate that the effects of frame rate using either *latency-only manipulation* or *frame-only manipulation* are similar between each other.

It is worth noticing that Fitts' law is applicable only to certain types of placement tasks (essentially one- and two-dimensional). Ware and Balakrishnan [1994] found that IP values were considerably lower for the cube target than for the pizza box target. They suggest that none of the extensions of Fitts' law reviewed by them can be valid for 3D placement. They also mention that although more substantial evidence is required, the low IP values and substantial acquisition times suggest that reducing a three-dimensional task to a one-dimensional task is not satisfactory for the purposes of modeling. Finally, they suggest that probably the measurements of difficulty (ID) should be done in terms of the ratios of the target volume to the workspace volume and not to the linear distances. We didn't attempt to model our task in terms of Fitts' law because our task is 3D rotation and doesn't seem compatible with the one dimensional placement tasks used in Fitts' law.

Effects of variation of delay on performance. [Wats98] presented a series of three experiments designed to study the effects of variation in system responsiveness on user performance in virtual environments. The experiment trials involved a catching task composed of an element of prediction and interception (open-loop), and a placement task (closed-loop) in a simple virtual environment.

The independent variables were mean system responsiveness (MSR) and standard deviations of systems responsiveness (SDSR), controlled using frame-latency manipulation (flm). The dependent variables were *grasp time* (the time elapsed to do a grasp) and *number of grasps* (grasp attempts) for the interception task; *placement time* and *placement accuracy* for the placement task.

The first experimental results (using *MSR = 9, 13* and *17 Hz* and *SDSR = 0.5, 2.0 and 4.0 Hz*) indicated that both MSR and SDSR can affect performance. Placement time was significantly longer with low levels of MSR and placement accuracy was lower in the 9 Hz mean frame rate than in the 17 Hz mean frame rate. Grasp time was longer and a higher number of grasp attempts were made at the lowest level of MSR (when delay is highest). Grasp time was also longer when SDSR was at the highest level.

In the second experiment ($3MSR \times 3SDSR$) the levels of MSR were improved and SDSR was controlled as a percentage of mean frame rate. Placement times were still sensitive to the MSR mean frame rates, taking longer for the lowest levels. When SDSR was lowest, placement times were shorter too. Interestingly, grasp time was not significantly improved with the fastest frame rates. This suggests that grasping was only affected at poor levels of MSR. The experiment showed that effects of SDSR were significant even when working with superior levels of MSR, and that effects of

SDSR did not interact significantly with MSR. A third experiment ($3MSR \times 3SDSR$) with improved levels of MSR and control of SDSR in terms of absolute values (as in experiment 1) confirmed that grasping was not affected significantly at high frame rates.

This results indicated a relationship between SR and the required feedback for the task: the grasping task (open-loop) was much less affected by longer MSR than the placement task (closed-loop).

It was also shown that severe levels of SDSR have a significant effect on human performance. In the case of the grasping task, SDSR was significant at poor levels of MSR. For the placement task, SDSR showed significant effects at improved levels of MSR, but not at lower levels. The research indicated that it is not necessary to tightly control SDSR.

Finally, [Wats98] suggested that since SR effects are task-dependent, LOD management techniques should also be sensitive to the task.

In this thesis we control SR using *flm* and attempt to affect MSR dynamically. In our implementation there are two types of MSR during interaction. The first type corresponds to MSR levels observed in real world systems, and the other type corresponds to the minimum attainable value of MSR. We believe that variation of MSR in this way should in fact improve performance during interaction.

2.3.4 Delay in Tracking Tasks

Latency and frame rate effects in 2D tracking. [Brys93] compared the separate effects of latency *(lom)* and low frame rate *(fom)*. The tasks chosen, tracking and placement, occurred in 2D using the mouse.

The experiment consisted of two parts, one where latency was controlled at values between 16.7 and 333 msec of delay (and FR of 60Hz), and one where FR varied at the same levels of delay, between 60 and 3Hz, with no latency delay.

In the tracking task, error was linearly dependent on latency and frame rate by similar amounts at corresponding values of delay. In the placement task, latency and frame rate had also similar effects on performance at corresponding values.

Results from both experiments indicate that the effects of delay coming from different sources (latency or frame rate) are qualitatively equivalent at corresponding delay values. Unfortunately, only two users were tested in this study.

Latency and frame rate effects in 3D tracking [Thar92] also studied the separate effects of system latency *(lom)* and frame rate *(fom)* in the context of teleoperator systems assisted with a helmet mounted display.

A 3D tracking task was analyzed. For tracking tasks in general, performance is evaluated by obtaining a measure of error obtained from the difference between the target's path and the user's effective path.

In the frame rate condition, FR levels varied from 0.1Hz(10 sec) to 30Hz (33 msec). Results for five subjects suggest that a plateau in performance occurs for frame rates of 10Hz (100msec) or higher.

Latency, modeled as a communication delay between a local and a remote site, was set between 0 msec and 1000 msec. As latency increased, performance decreased in a linear way. After 400 msec, the measured error in performance degraded to a point such that that the user could just as well leave the cursor at the center of the box trajectory and obtain the same error value.

Tharp's results reveal that it is more convenient for the present work to study delay for frame rates lower than 10Hz (~150 msec SR), due to the plateau effect.

Combined effects of latency and frame rate [Elli99] examined the effects of spatial sensor distortion, and combined latency and frame rate for a 3D tracking task. The system used a stereo head-mounted display system with hand and head position 3D sensors.

A set of three frame rates (6, 12 and 20 Hz, or 16, 8.3 and 50 msec frame time) was combined with five latency categories (480, 320, 230, 130, and 80 msec) into 11 conditions of delay, which were crossed with the two spatial distortion correction conditions. The measured variables were tracking error and a subjective questionnaire filled by the participants. The questionnaire included an Adapted Cooper-Harper (ACH) controllability scale, which is sensitive to workload effects [Elli99] and other subjective scales including nausea, neck ache, head/eye ache, eye tearing and a judgment about user stability.

The study revealed that the correction of the spatial distortion introduced by the sensors did not produce significantly better performance. Latency in particular seemed to be the strongest and more statistically reliable factor, degrading performance in a linear way. Latency and frame rate correlated significantly with the measure of tracking error, the ACH scale and the stability judgment.

Besides confirming that delay has a linear relationship to performance, the study is valuable in that it simulates typical systems more closely, where latency and frame rate are both present , it also stresses the point that the effects of delay are not limited to user's performance, but may also affect the user physically.

Network latency effects in Collaborative Virtual Environments. A CVE is a VR system that allows people in remote locations to work together over a network. In this type of system, latency is caused by network delay and performance is highly dependent on the Quality of Service (QoS) provided by the network. Network latency and variability in the delay, called jitter, are important elements of QoS.

[Shin99] studied the changes in performance generated by latency (lom, MSR) and jitter (lom, SDSR) in a network connecting a pair of CVEs. Four network conditions were tried along four paths of variant difficulty. The network conditions were: a fiber-optic LAN (Local Area Network) Scramnet simulating average Ethernet latency with a constant 10 msec delay (Scramnet-10msec), an Ethernet LAN subject to jitter of up to 500 msec, a Scramnet-200 msec simulating the average ISDN latency and a ISDN subject to jitter of up to 2 seconds. Both Scramnets represented conditions where there was no jitter (SDSR=0).

The experimental task involved collaborative movement of a ring along different paths by a team of two persons, one in each location. Results were evaluated in terms of task completion time and accuracy, which was measured in terms of number of collisions of the ring with the 3D path.

Completion times were significantly less for the Scramnet-10msec than for the Scramnet-200 msec and the ISDN. This indicates that the Ethernet LAN and the Scramnet-10 were qualitatively very similar conditions, suggesting that the effects of jitter are negligible when latency is low. There was also a significant interaction of $Network \times Path$ on the mean number of collisions, indicating that the longer latencies affected the performance to a greater degree when subjects were working on more difficult tasks. Overall performance was affected by the characteristics of the network used to connects the CVEs [Shin99]. It was observed that high latency can impact user's coordination, and the variability of the network latency reduces the ability of the subjects to use prediction in performing the task.

This results coincide with the observations of [Wats98], indicating that the effects of delay depend on the level of feedback or the difficulty required by the task. At the same time, both studies reveal that variation in latency SDSR does not affect performance if delay (controlled either by lom or flm) is kept at a minimum and the level of difficulty is relatively low; 10 msec (100Hz) for CVES and 17Hz FR and higher for the grasping task in[Wats98]. These results stress the relevance of the appropriate control of SDSR when MSR is poor (delay is high).

Effect of update rate in the sense of presence in a navigation task. The sense of presence in a virtual environment is attained when, based on computer-generated input, the participant believes that he/she inhabits the virtual world as a place. [Barf95] presented a user study evaluating the effects that frame rate (fom) has in the subjective perception of presence in VEs. The sense of presence was evaluated according to two factors: the perception of spatial realism from the user in the virtual environment, and the perception of fidelity of the user's interaction with the virtual world (when compared to the real world).

Results suggested that when frame rate (*flm*) is sufficiently high as to produce smooth motion, the simulation speed essentially becomes transparent. Frame rates above 15Hz seemed to produce no additional increase in presence. Results also supported the view that fidelity of interaction is an important factor in inducing presence, suggesting that the subject's impression about the realism of the virtual environment is affected not only by the visual scene itself (degree of "photorealism"), but also by the frame rate. These results suggest a plateau in the sense of presence for frame rates above 15Hz.

2.4 Summary

In this chapter we have described the nature of delay occurring in interactive 3D graphics systems. First, a definition of system responsiveness and corresponding terminology was presented. Then, the stages that conform the process of reading an input event until the appropriate output is displayed were defined. Next, several sources of delay corresponding to any of the processing stages were presented. Similarly, different ways to reduce delay on each of these stages were presented. Finally, we present a number of studies that attempt to model and characterize the effects of delay.

Across these studies it has become clear that delay affects performance in different kinds of tasks: two and three dimensional placing, tracking and catching. Results suggest that tasks requiring more feedback are most affected by delay. Results also indicate that when a system provides high frame rates (between 10 to 20Hz) a plateau in performance is reached, above which no improvement in performance is observed.

There is a direct relationship between the amount of visual detail and system responsiveness. In this research we explore the idea of balancing delay and visual detail, we have variable frame times and hence perform *flm*.

For the experiments we use a 3D orientation matching task, a form of placement. During trials, a user is presented two representations of an object at different orientations. The task consists in orienting one of the objects to match the orientation of the other (for more information on the experiment setup please refer to Chapter 6). Under our approach, two representations at different levels of detail will be used during interaction. A fully detailed representation of a model will be subject to the effects of delay discussed in this chapter. According to a control criteria explained in Chapter 4, we will select either the fully detailed representation or a coarsely detailed representation, enhancing system responsiveness depending on user input.

Chapter 3

Level of Detail and LOD Management

As mentioned in Chapter 2, delay during rendering (i.e. Graphics Computation stage), is mostly dependent on the number of polygons or primitives sent to the renderer. In many cases, the amount of primitives that could be sent to the renderer exceeds by far the maximum amount of primitives that can be processed within the time available for interactive display (˜100msec).

The need then arises to simplify the complexity of the scene or objects before sending the primitives to the renderer. The goal is to produce the best looking image possible using a limited number of primitives so that the renderer can process the primitives at an interactive rate.

There are a number of ways to reduce image complexity, the most basic ways are mentioned in section 3.1. These approaches are limited in various ways; in section 3.2, we describe more sophisticated approaches to model simplification, collectively called level of detail techniques. Since in this thesis we deal with the more general question of level of detail control, we present current approaches to LOD management in section 3.3.

3.1 Basic approaches to reduction of image complexity

In this section we discuss current alternatives to reduce the delay caused by image complexity available in some applications.

Elimination of photorealistic effects or computer intensive calculations An almost obvious step to reduce image complexity is to switch off all the algorithms that make 3D images look photorealistic. This includes shininess, reflectivity and some shading schemes among other possibilities. A flat shading scheme, color and textures are some properties that still need to be represented even when some simplification is required.

Change of primitives Some commercial applications offer the possibility to render simpler primitives to represent the objects. In this case polygons are substituted with wireframes or point clouds. This approach is useful to some extent, because certain calculations (like illumination or shading) need not be applied on lines or points. But the benefit obtained is marginal, given the fact that current graphics hardware is optimized to render polygons as fast as lines and points. On the other hand, if the original set of primitives to be rendered is already prohibitive in terms of interactivity, the new set of primitives will also be prohibitive.

Arbitrary reduction of primitives It is possible to arbitrarily reduce the number of primitives to be rendered. Some schemes include random undersampling to produce a points cloud of a certain density.

Bounding boxes The most widely used simplification scheme is to represent an object by its bounding box, formed by a 3D parallelepiped that minimally contains the object it represents. Bounding boxes are popular because they are extremely easy to compute and fast to render.They do reduce the number of primitives to be rendered, but only represent poorly the object being displayed.

Complex bounding boxes Complex bounding boxes were designed during this research to provide an abstract, simplified version of a model. This simplification can be easily computed in real time when a model is loaded into the system, and is bounded to a certain level of complexity, which makes it appropriate for use with the proposed LOD selection criterion.

Unfortunately, the algorithm that produces the complex bounding boxes had a level of complexity equivalent to vertex clustering, a simplification scheme that pro-

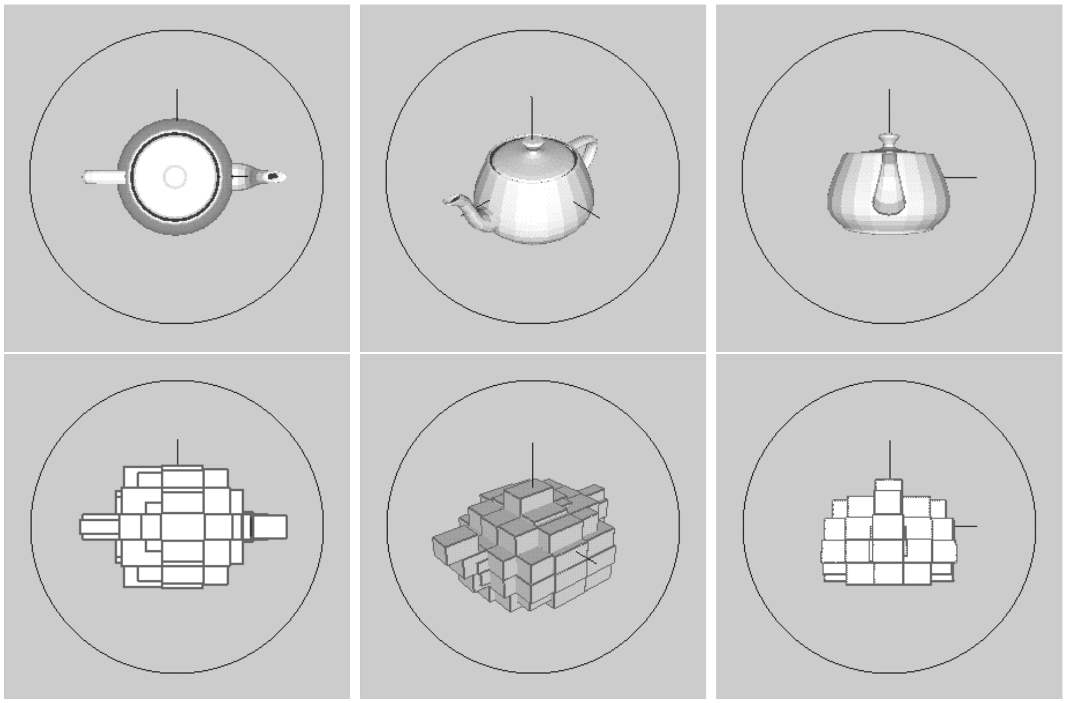

Figure 3.1: The complex bounding boxes

duced LODs of better quality in the same amount of time, using the same amount of primitives. As a consequence, the Complex Bounding Boxes were discarded for further use in this research. Figure 3.1 presents some views of the algorithm applied onto a well known teapot model.

3.2 Mesh Simplification and Level of Detail Techniques

A fair amount of research in the graphics community has been dedicated to automatically reduce complexity of polygonal models in various ways. The common goal of these techniques has been to produce simplified representations of an original model at various levels of detail (multiresolution representations), which can be used to speed up the rendering process. Figure 3.2 shows the teapot model at three levels of detail. These levels of detail were generated using Q-slim, a LOD technique by [Garl97].

These level of detail (LOD) techniques differ in the way they achieve the simplifications and in the final effects they produce on the models. Some of them happen

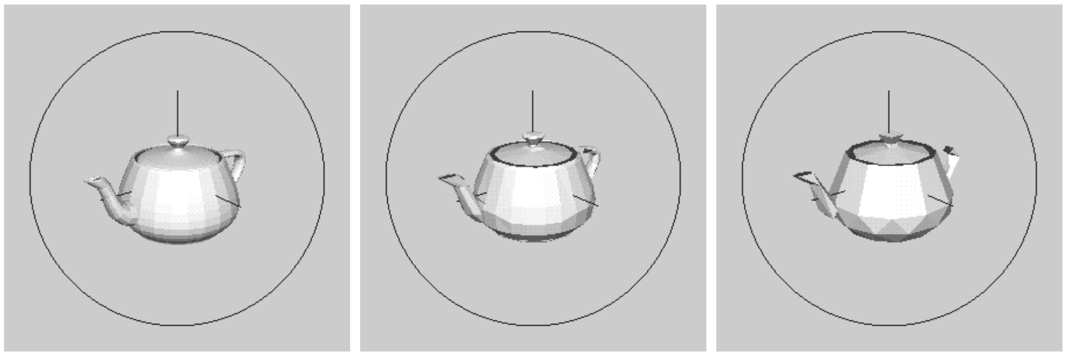

Figure 3.2: The teapot model at various levels of detail. Left, the model represented with 2256 faces; middle, 500 faces; right the model with 150 faces

to be more appropriate than others for a certain purpose or task. As an example, a technique that smoothes edges [Turk92] can be more appropriate for simplifying molecular models, while another technique that eliminates small holes or cavities in CAD models [El-s97] might be more useful in a Virtual Reality application.

There is an enormous array of techniques that achieve curve and surface simplifications. It is not the purpose of this thesis to explore the gamut of level of detail techniques available. The interested reader can read [Lind98], [Pupp97], [Heck97] to obtain insight into LOD algorithms and simplification.

3.2.1 Image based simplification

Another alternative to LOD simplification is offered by image based simplification techniques. These techniques generate different views of a 3D scene as a pre-processing step. During run-time, these views are rendered as textures projected on parallelograms that replace the actual geometry.

This brings along a drastic reduction on the complexity of the scene. On the other hand, the scheme introduces artifacts due to the fact that the images correspond to a unique position, and obviously not all possible positions can be sampled to generate an image. A smaller set of views generated from the most probable viewing points is generated, and these views are interpolated as the user moves through the scene. The challenge is to produce smooth transitions as the interpolations occur.

Image based rendering is most appropriate for rendering static environments, because the images can be preprocessed and need not be updated during interaction.

It is especially used to simulate architectural walkthroughs. These techniques are evolving in conjunction with level of detail generation techniques and some interesting hybrid approaches have been presented, as in [Alia99].

For the purpose of this thesis, we will focus on the use of LOD simplification techniques. In practice, the work presented in this thesis can be applied to a number of LOD techniques and alternative representations.

3.3 Level of Detail Management

Level of detail management, also known as LOD selection or LOD control, refers to the set of techniques developed to manage the trade-off between level of detail (or image quality) and display speed.

The nature of the selection varies and becomes highly dependent on the application type and its goals. For some applications, fast interaction is most important, as in real-time systems. For some other applications, detailed visualization in specific areas is most important, as in medicine or science. A more general purpose application will require an acceptable balance of interaction and visual detail. A review on level of detail and LOD selection techniques is presented in [Lind98].

Most LOD management techniques aim at producing the best attainable image, while limiting the amount of visual error introduced under a certain LOD simplification scheme. A smaller group of techniques aim at maintaining a certain level of temporal quality. A dynamic simplification system, i.e. a system that uses several LOD representations at run-time, can have implemented more than one of the LOD selection techniques described below.

3.3.1 Visual quality based LOD selection

This section presents the set of techniques that enforce a certain level of image quality. In most cases, the preservation of the image will be applied locally in a model or region of the scene. In some fewer cases, the criteria affects the image globally. Most of these techniques are dependent on view-point, but some depend more on perceptual characteristics of the human visual system.

View-point dependent criteria

These criteria rely on the relationship between the viewing point and the objects in the scene, in order to eliminate geometry that doesn't need to be accurately rendered or, conversely, refine the model in specific regions.

Distance based selection A common approach to LOD selection is based on the objects' distance to the viewers point of view. Objects located far from the user's point of view are smaller and can be rendered with less detail than objects located closer to the viewpoint. A variation of this approach was used for early flight simulation systems, where textures representing terrain were used at several levels of detail, according to the distance from the viewing point for terrain rendering [Scha83].

Object size Based on the same principle of the distance based selection, object size selection selects LOD based on the size of the object (or its bounding box) in pixels, when projected on the screen. Object size dependent selection is more general than distance based selection, since small objects can be quite close to the view-point and yet still occupy a small part of the visual field. This approach is used by [Funk93] and [Redd97].

Visibility culling Under this approach the scene graph is traversed to refine only those parts of the mesh which lie within the view frustum. The view frustum is composed of four planes shaping a semi-infinite pyramid which projects from the user's viewpoint into the display. [Hopp96] implemented this technique for progressive meshes.

[Alia99] implements visibility culling by splitting the space of the model into *virtual cells*. Geometry inside the cell is considered "near", and is the one that is selected for 3D rendering. Geometry outside the cell is considered "far" and is generated using precomputed textured depth meshes (TDMs) which cover the inner walls of the cell, and contain precomputed textures that represent the outer environment of the cell.

In Aliaga et al's approach a balance between the size of the cull box and the LOD selected for the inner geometry is looked after. Small cull boxes are bad because the representation of the far geometry is more subject to distortion. Big cull boxes include more elements to be rendered but coarser LODs have to be applied to the geometry in order to maintain the polygon budget.

Surface orientation and silhouette preservation The purpose of this technique is to increase detail only where the surface is front-facing to the view point. To do this, the cone of normals of a vertex and its descendants is tested against a viewing point to determine whether a vertex lies in the back facing region of the model.

Similarly, the cone of normals can be used to determine whether the node potentially lies on the silhouette, in which case the vertex can be refined, thus enhancing quality in this region of the model. This approach is used by [Hopp96], [Lueb97] and [Xia97], although Hoppe uses a different criteria for silhouette preservation.

Local illumination The purpose of this criterion is to increase detail in the parts of the surface that are illuminated. If normal-based local illumination (such as Phong illumination) is used, a cone of normals of the vertex and its descendants can be compared with the range of the reflection vectors to determine whether they contain the view's direction or not. [Xia97] uses this approach.

Screen-space error The goal of this criterion is to do mesh refinement in every place where a measure of geometric error in model space exceeds a certain screen-space tolerance when projected on the screen. In this way, a certain level of quality on the simplification is enforced.

Some mesh simplification techniques rely on two basic operations, vertex-split (*vsplit*) and edge-collapse (*ecol*) (see figure 3.3), which allow for refinement of specific regions of the model. The screen-space error criterion is used for selective refinement. The error introduced by collapsing vertices can be thought of as the maximum distance a vertex can be shifted during the collapse operation. By splitting those vertices whose screen space error exceeds a user specified error threshold, [Lueb97] and [Xia97] guarantee a certain level of quality on the simplification. On the other hand, if the maximum possible screen-space projection of the *ecol* distance is less than the threshold, it means that the region occupies very little screen space and hence small features can be simplified.

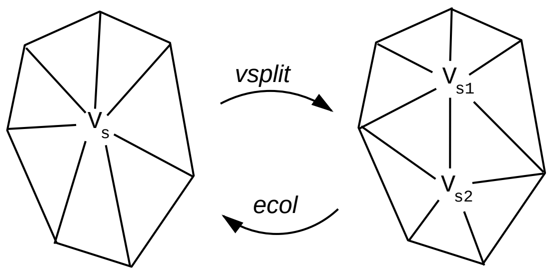

Figure 3.3: Vertex-split and edge-collapse operations for selective mesh refinement.

Additionally, a measure of the screen-space error can also be used to do mesh refinement in the silhouettes of an object. [Hopp96] implemented such a measure. In this case, the distance between the approximate surface and the original is compared with a screen-space tolerance. This test results in more refinement near the model silhouette where surface orientation is orthogonal to the view's direction.

Geomorph refinement Runtime geomorphs try to combine a high and steady frame rate with absence of popping artifacts. While popping can be avoided if the screen-space error tolerance is kept low, the number of faces can vary greatly depending on the viewpoint, causing a non-uniform frame rate.

Runtime geomorphs aim primarily for a constant frame rate by adjusting the screen-space error tolerance and eliminate popping by smoothly morphing the geometry. When the refinement criteria indicate the need for an *ecol* or *vsplit* operation, instead of performing the transformation instantaneously, it is performed as a geomorph by gradually changing the vertex geometry over several frames. Geomorphs are implemented by [Hopp98]. The implementation adapts the screen-space error metric by anticipating the future location of the user, based on a measure of the viewer velocity per frame.

Human-vision dependent criteria

These criteria takes advantage of the physiological characteristics of our visual system and proposes LOD selection according to these characteristics.

Object Motion Humans do not easily recognize details in objects that are moving in the visual field, like the ones that rotate or move across the field [Seku94]. LOD selection can be achieved depending on the object's angular velocity. If the object moves with small angular velocity, a high LOD is used. Similarly, a low LOD is used if the angular velocity is high. This approach is used by [Funk93], [Redd97] and [Ohsh96].

LOD similarity Humans are very sensitive to popping effects, the artifacts that are present when a system switches from one LOD representation to another. In LOD based selection two different LOD's, the one currently being displayed and the one being considered for future use, are compared in order to determine their similarity. If there is little similarity between both representations, it might be convenient to select another LOD with greater similarity, to minimize popping artifacts (see [Funk93]). One LOD technique that does not present popping is presented in [Redd97]. Under this approach, LOD generation and selection is based on perception; a lower LOD is selected whenever the change is not perceivable.

Reduction of peripheral detail Peripheral detail management is based on the fact that the perceptual ability of the human visual system is much higher in the center of the field of view than on the periphery. This facilitates reduction of detail on those areas of the display located far from the center of the field of view, where the user normally sets his/her focus.

Under Watson's approach two kinds of detail can be removed: imperceptible and irrelevant detail. Imperceptible detail is defined as the detail that is finer (in terms of pixels per degree of visual angle) than the corresponding visual acuity for each level of eccentricity (distance in degrees from the center of the field of view). Irrelevant detail varies by task. [Funk93],[Ohsh96], [Redd97]and [Wats97a] have used this approach.

Selection based on fusion area The phenomenon of stereo image fusion occurs only in a certain area located across the field of view and around to the point where a person sets his focus. This area is called the Panum's fusion area. When an object is out of the Panum's fusion area, a duplication of images occurs (called diplopia, [Ware94]). In this case, the duplicated images are out of focus and visual acuity is extremely low for these images, enabling the use of low LODs according to the amount of separation between the images [Ohsh96].

Stereo composition of conflicting images Stereopsis can be achieved even when two corresponding images present certain differences between them. For example, most of the people can perceive stereo images without glasses if the corrections required to see sharp images is not too big, even if the amount of correction is different for each eye. [Fris76] proposed that stereopsis in stereograms with unmatching textures (rivalrous texture stereograms) is achieved trough visual channels that compose stereo images based on the frequency of the patterns in the image. This approach is used by [Dins89] in the domain of image compression of 3D images, and is also supported to some extent by [Smet95].

Summary The idea of exploiting several characteristics of the human visual system makes sense from the perceptual point of view. However, in the case of the techniques that require head or eye tracking, a technological challenge arises, because such systems need to have low latencies and error free, accurate head and eye tracking. Also in the case of peripheral detail selection [Wats97a] mentions that in most cases display resolution never exceeds visual acuity at any eccentricity.

3.3.2 Temporal quality based LOD selection

There is a smaller group of LOD selection techniques that emphasize temporal quality (low delay) as opposed to visual quality. Temporal quality oriented techniques attempt to produce an image every certain period of time, allowing the perception of apparent motion [Seku94] and limiting delay. The most sophisticated LOD management schemes combine one or more visual quality selection techniques with one of the techniques presented below.

Reactive LOD management In this approach a certain target frame rate is set. During interaction, a level of detail is selected and the scene is rendered. If the rendering time was within the limits set by the target frame rate, the same LOD is used for the next iteration. If the rendering time was longer than desired, the information about the difference between the expected time and the produced frame time is used to select a more appropriate (coarser) level of detail. This approach is discussed in [Funk93] and [Lind98].

Polygon budget If a certain frame rate is desired only a certain amount of primitives can be rendered per frame. Many of the simplification applications that guarantee fast frame rates work under a "polygon budget" basis. The idea is to produce the best attainable image, by minimizing the error due to simplification, while keeping the target budget. A simple approach used in mesh refinement is to sort the model according to screen space error, and add polygons where error is greatest until the budget is used up. This approach is implemented in [Hopp98] and [Lueb97]. In the case of [Alia99], (see *visibility culling*) a fixed number of polygons is devoted to render the far geometry, and the remainder of the budget is used to define the near geometry.

Predictive LOD management According to [Funk93], reactive LOD works well for scenes or environments that are consistent in complexity, but it does not work so well in situations where scene complexity varies drastically, because the system adapts gradually (across several frames) to the new level of complexity being rendered. To solve this problem, these technique predicts rendering time. Scene complexity is anticipated and appropriate LODs are selected for rendering, maintaining a limit on delay.

In predictive LOD management an image is constructed taking into account that every object included will contribute to the scene quality to some extent, but will also consume some rendering time. For this purpose two functions are defined, one estimates the benefit of including an object in the scene and the other estimates the cost (in frame time) of including this object in the scene.

The benefit function depends on the projected size of the object, quality, semantic relevance, motion blur and affinity between current and planned LOD. The cost function, which is used to estimate the expected frame rate, is dependent on the size, number of polygons and number of vertices of the object to be rendered, and is also dependent on a certain rendering algorithm and machine.

Finally, the best attainable image is composed of the objects that have maximum cumulative benefit, for which their added cumulative cost falls in the range of the target frame time.

Asynchronous Simplification With this approach the simplification and rendering tasks run asynchronously, where the simplification process simplifies the next frame to be rendered, while the rendering process takes the result from the earlier simplification step. Asynchronous simplification offers a mixed way to limit rendering delay by having both a polygon budget and a time limit for frame time.

The simplification process in this technique initially takes a coarse representation and increases detail up to a certain polygon budget, according to some refining LOD criteria.

Normally, the rendering process takes longer than the simplification process, and the total time is defined by the rendering time. But in some situations, like when the scene complexity increases abruptly, it may happen that the simplification process falls behind the rendering process. In this case, the rendering process does not wait for the simplification process to render an image; instead, the simplified representation is taken by the renderer as it is. If this happens, the scene rendered is somewhat coarse in quality, until the simplification process catches up and then the scene gradually changes back to the expected quality, while the frame rate is kept at a desirable level. This scheme is presented by [Lueb97].

Image caching based on a disparity measure In this technique, presented by [Shad96], a hierarchical image caching scheme based on a spatial partition of the elements in a scene is generated. This approach is defined in the context of image-based techniques (explained above), but is revised here because it uses an error metric based on spatial disparity. Basically, in this approach the spatial error metric assists in deciding if a cached image should be used. If the angle between the viewing direction stored in a cached image and the desired viewing direction exceeds a certain threshold, the cached image is discarded and a new one is used. This approach is in principle similar to our approach for angular-disparity LOD selection. In the case of image caching the measured disparity is used for visual continuity, and does not limit delay. In our case we use the disparity measure to limit both visual and temporal error. A detailed description of these techniques is given in Chapter 4.

Summary By guaranteeing a certain frame rate, or by limiting the complexity of the images rendered, these techniques improve interactivity, because they reduce delay in system response. However, these techniques are not necessarily sensitive to the rate of change in user input (speed of interaction). That is, they do not attempt to increase frame rate or reduce delay when the rate of change in user input is particularly high.

In this thesis, we propose two LOD selection techniques that emphasize reductions in delay when the speed of interaction is high. This is achieved through a control of the spatio-temporal error. The techniques we propose are in some sense hybrid, because they provide either temporal quality or visual quality, depending on the user input.

3.3.3 Subjective LOD selection

Subjective LOD selection occurs when the user decides to use a certain LOD for specific objects or features in the scene.

Context-based importance Here the user designates the relevance of certain features or objects in the scene, which should be rendered at a higher LOD. This approach is used for example in cartography, to emphasize features like ridges, peaks or rivers [Funk93].

Manual selection In manual selection, a user specifically selects a certain LOD for an object or a scene. This approach is used in commercial software for design (3D Studio Viz), and object manipulation (Inventor's IvView).

While this approach is probably very useful as an immediate solution to achieve LOD management, automatic LOD selection is much more powerful, since it can be more effectively applied to a much more complex environment, as can be inferred from the discussion in previous sections.

3.4 Summary

There are a number of ways available to reduce the complexity of an image. In this thesis we focus on level of detail (LOD) techniques.

Level of detail generation techniques produce a set of simplified representations of a model at various resolutions. The question then arises, about how to select a certain level of detail from this set. LOD management techniques are used to answer this question, and an extensive variety is presented here.

Most LOD selection techniques enforce a certain level of visual quality in the simplification, techniques in this category can be subdivided between techniques that decide selection based on the user's viewpoint and techniques that base the decision on certain characteristics of the human visual system. Other LOD techniques enforce a certain level of temporal quality to enhance interaction and limit delay. These focus mainly on maintaining a certain frame rate, but they are not sensitive to the rate of change in user's input (speed of interaction).

In this thesis, we present two techniques that take into account the speed of interaction and limit spatio-temporal error. The goal is to provide temporal quality whenever necessary, at the expense of LOD. These techniques will be explained in detail in the next chapter.

Chapter 4

Control of temporal and visual detail by measuring spatio-temporal error

4.1 Speed of interaction

Evidence in psychology about human perception of motion states that the visual system is less sensitive to detail when objects are in motion [Seku94], giving support to the theory that detail can be reduced in exchange of a higher degree of interaction.

In Chapter 3, we discussed some LOD selection techniques based on the object's velocity. These techniques achieve LOD selection depending on the object's angular velocity. If the object moves with small angular velocity, a high LOD is used. Similarly, a low LOD is used if the angular velocity is high.

[Redd97] computes the visual acuity for an object in motion, then combines this value with a measure of visual eccentricity and object size to select an appropriate LOD. [Ohsh96] takes into account the angular speed of an object, the visual eccentricity and the position in the area of stereoscopic vision fusion to compute the visual acuity, and select appropriate LODs. [Funk93] simply assigns a certain value that represents the object speed for each object in a scene, and selects a lower LOD for objects with high speed.

While all of these techniques take into account the object's speed to effectively reduce level of detail, they really do not enforce a higher temporal detail when objects are moving.

We have found that when the speed of interaction (the rate of change in user input)

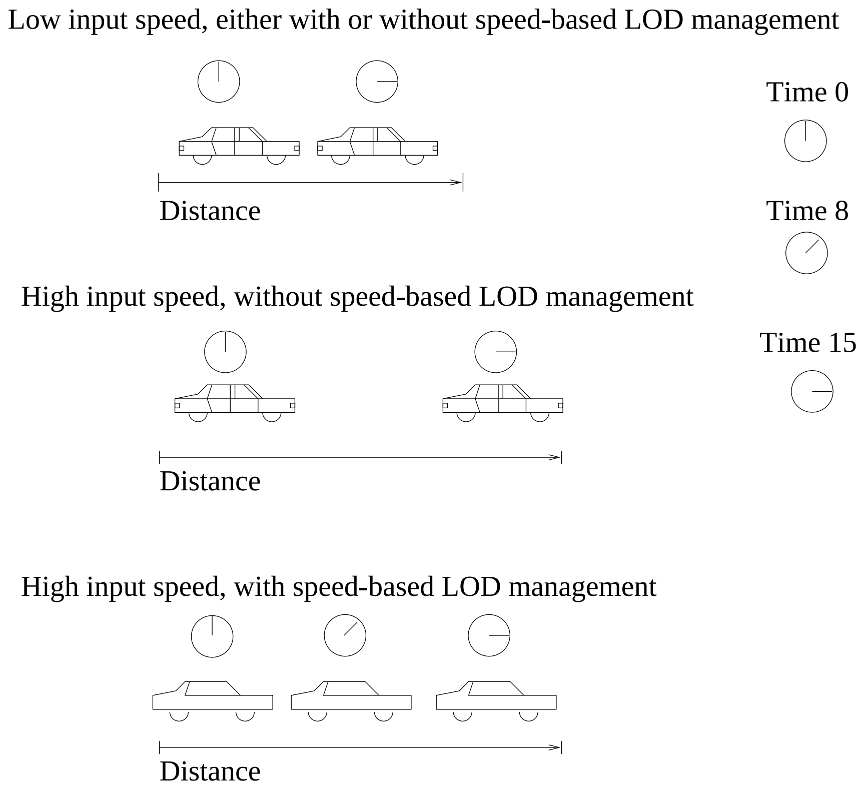
Low input speed, either with or without speed-based LOD management
Distance
Time 0
Time 8
High input speed, without speed-based LOD management
Time 15
Distance
High input speed, with speed-based LOD management
Distance

Figure 4.1: Input-speed based LOD selection

is high. it is important to provide higher temporal accuracy, because more changes need to be represented. Higher temporal accuracy can be obtained by providing a higher amount of intermediate samples of lesser visual quality when the speed of interaction is high. Figure 4.1 shows how higher input sampling using low detail representations can increase the perception of visual continuity when the speed of interaction is high.

In this chapter we describe two LOD control techniques designed to increase spatio-temporal accuracy. The first technique takes into account speed of interaction and the second technique directly controls spatio-temporal error during interaction.

4.1.1 LOD selection based on speed of interaction

During interaction in a 3D environment or just while manipulating 3D models, the speed of user interaction varies depending on the user's behaviour. For example, during a modeling task, a designer might want to visualize the model being manipulated from a totally different point of view, rapidly rotating the model until a desired view is obtained. Then, the designer might want to work on this view of the model, making slight changes in position while doing so. In this case, we can identify two general classes of motion: large ballistic motion, for a change in the point of view, and small motion for finer adjustments of the model's position.

LOD selection can be done depending on the user's speed of interaction. In our case, we contemplate a bimodal switch, where only two levels of detail are available for selection; one is full detail, used when the speed of interaction is slow; the other is low detail, or coarse, used when the speed of interaction is high. Such a switch emphasizes temporal detail at the expense of visual detail when the speed of the manipulation is high. It also emphasizes visual detail at the expense of temporal detail when the speed of interaction is low, providing a balance between visual and temporal detail according to the speed of the interaction.

The monitoring of the speed of interaction can be done in parallel with the rendering process, and switching can be achieved immediately as soon as a threshold of speed is exceeded, increasing system responsiveness. Figure 4.1 shows the behaviour of a system that controls LOD based on this switching technique.

4.1.2 Switching based on spatio-temporal error

As discussed in Chapter 1, LOD techniques deal with several kind of errors during interaction. The spatial error is the difference between the displayed position of an object and the position where this object should be, according to the system input and the simulated situation. The spatio-temporal error is the amount of spatial error introduced when temporal error is not accounted for.

The spatio-temporal error directly depends on the amount of delay present in the system. Figure 4.2 illustrates how spatio-temporal error varies for a pair of conditions of delay where the rendering system consistently receives old input samples and renders them immediately (*latency-only manipulation*). The diagonal lines indicate the unidimensional input and displayed positions of an object moving at a constant speed.

In systems where frame rates are low (and *frame-only manipulation* is used), some spatio-temporal error is also introduced, but in this case the spatio-temporal disparity increases as the input object position changes while the image in the display remains

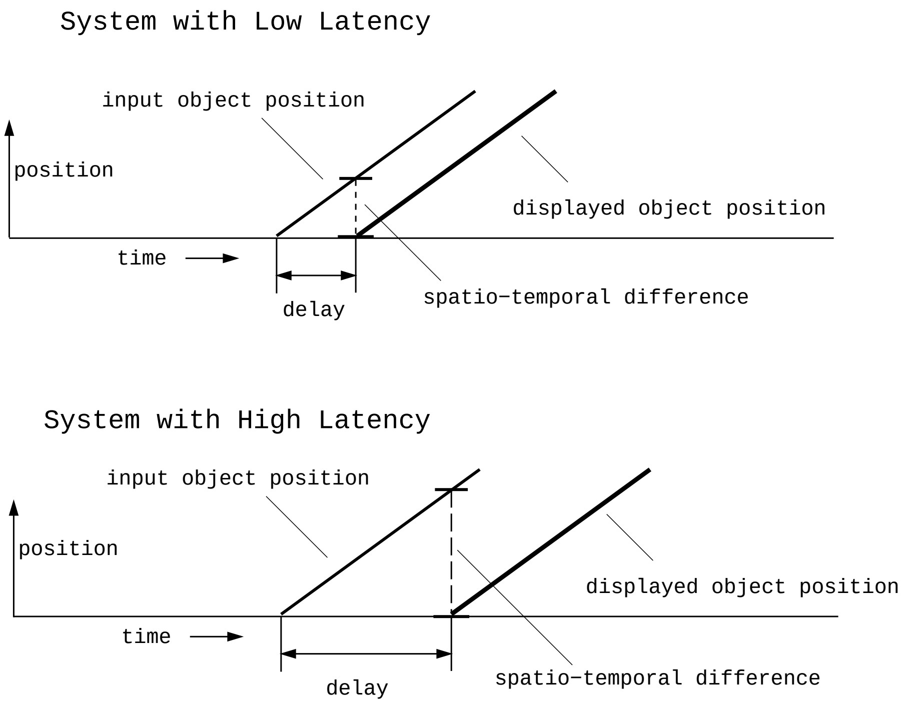

Figure 4.2: Spatio-temporal difference present under two conditions of delay (latency-only manipulation).

unchanged. This situation is presented in figure 4.3. It is important to remember that in a real system, the displayed output depends on the combined effects of the different sources of delay (see Chapter 2).

Similarly as with the speed switch, a spatio-temporal threshold can be set in order to enforce a maximum error in the difference between the displayed object position and the actual object position in a configuration using frame-latency-manipulation. As with speed-based LOD selection, monitoring of the spatio-temporal error can be done in parallel with rendering. When the spatio-temporal threshold is exceeded, a lower LOD representation can be used to provide fast, updated feedback to the user. This would enforce a certain level of spatio-temporal accuracy.

In a situation where the input speed of the object is variable, the amount of spatio-temporal error is also variable, depending on the input acceleration of the object and the direction of movement. Figure 4.4 shows this phenomena in the top and middle images; the image at the bottom of this figure shows the behaviour how a system where the spatio-temporal error is controlled by means of a certain spatio-temporal threshold.

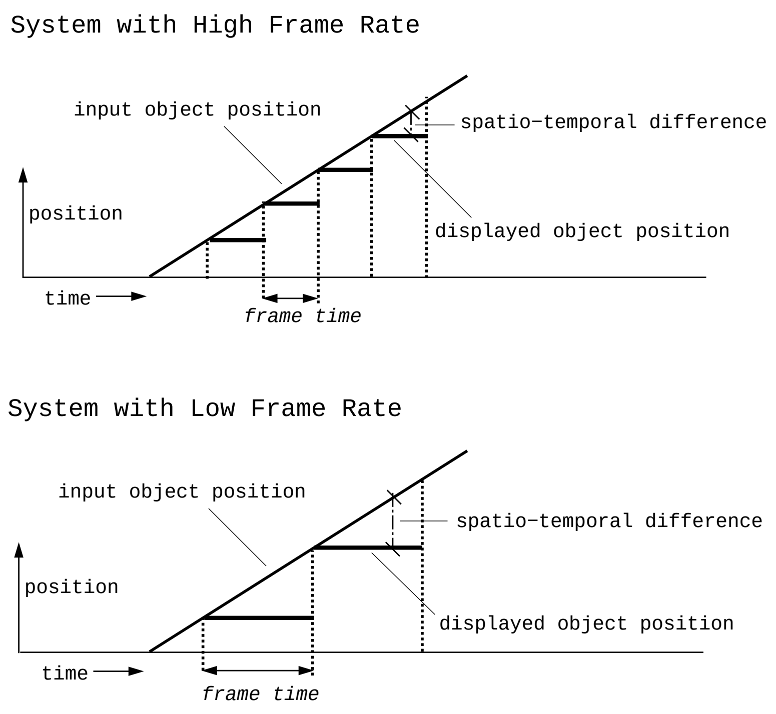
System with High Frame Rate
input object position
spatio−temporal difference
position
displayed object position
time
frame time
System with Low Frame Rate
input object position
spatio−temporal difference
position
displayed object position
time
frame time

Figure 4.3: Spatio-temporal difference occurring in systems with varying frame rate. (using frame-only manipulation).

4.2 Control of Spatio-temporal error in 3D Rotation

So far, we have referred to speed of interaction and spatio-temporal error in general terms. LOD selection based on control of spatio-temporal error can be applied to 3D rotation tasks and has been implemented in this thesis. LOD selection based on interaction speed can be interpreted as LOD selection based on angular speed based, where angular speed is measured in degrees per second during input rotation. LOD selection based on spatio-temporal error can be interpreted as LOD selection based on angular disparity, where angular disparity is measured by the amount of degrees required to accomplish a 3D rotation between the displayed orientation and the input orientation. Figure 4.5 shows the response of a system to varying input angular speed where neither angular speed nor angular disparity is controlled. Figure 4.6 shows the response of a system that has implemented LOD control based on either angular speed of angular disparity; in this Figure α is an angular value and the two images at the bottom right are low detail representations of the original model.

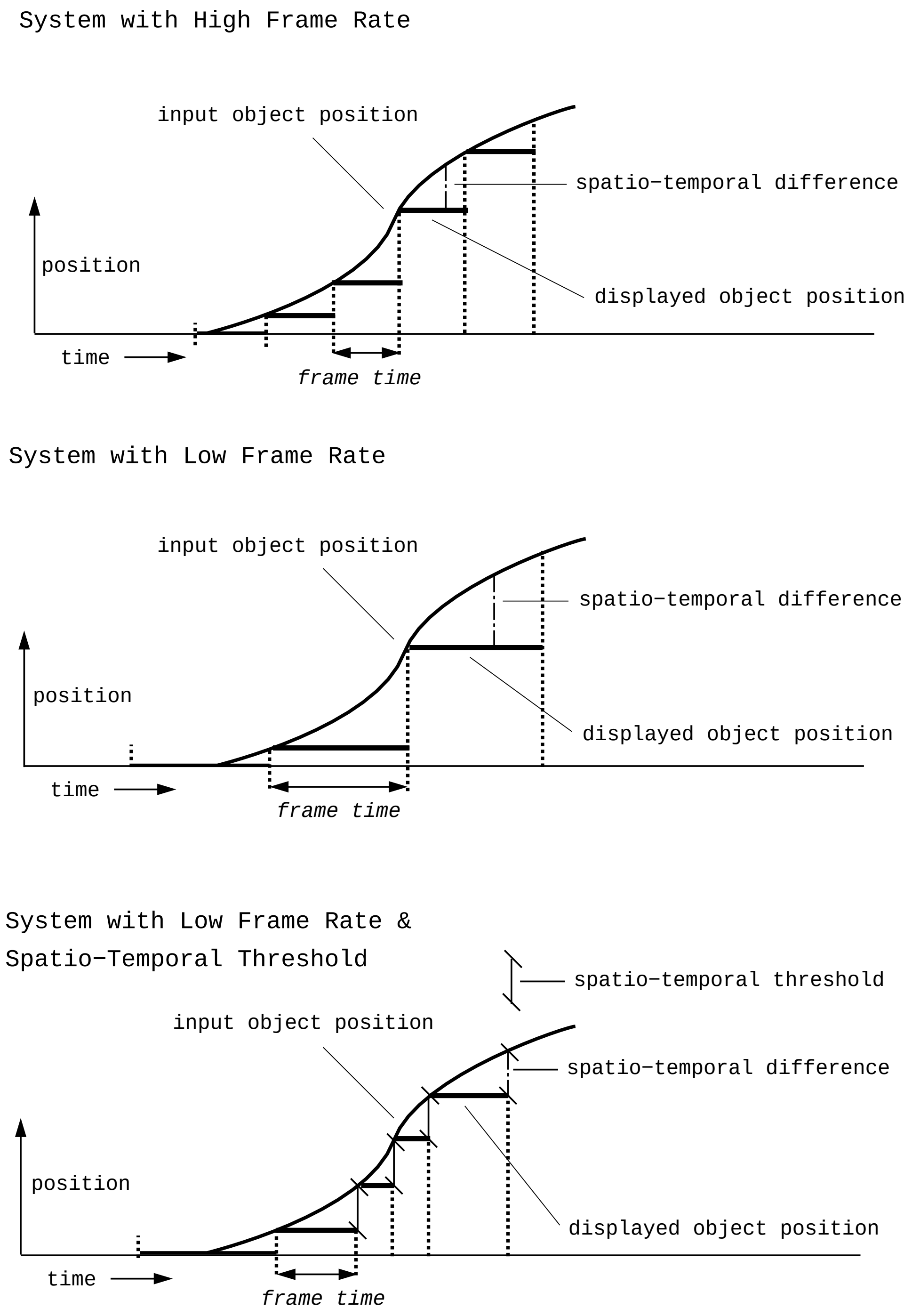
System with High Frame Rate
input object position
spatio−temporal difference
position
displayed object position
time
frame time
System with Low Frame Rate
input object position
spatio−temporal difference
position
displayed object position
time
frame time
System with Low Frame Rate &
Spatio−Temporal Threshold
spatio−temporal threshold
input object position
spatio−temporal difference
position
displayed object position
time
frame time

Figure 4.4: Spatio-temporal error for an object with variable speed (frame-only manipulation).

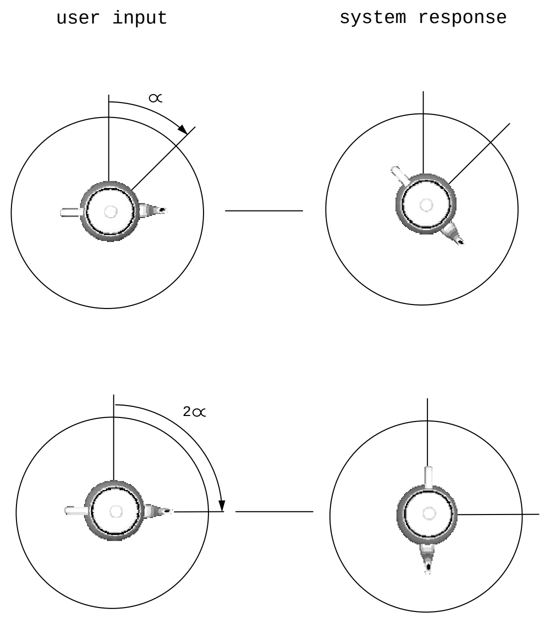

Figure 4.5: Spatio-temporal error present in 3D orientation tasks. Both input and response occur during successive lapses of time of equal length.

The implementation of these two techniques using a switching mechanism between two levels of detail (full detail and coarse detail) is presented in the following sections.

4.2.1 Description of the angular-speed switch

In this section we describe our implementation of the angular-speed based LOD selection technique. In our case two LODs, full detail and coarse detail, are available for selection.

Angular speed (ω) is the rate of motion of an object that moves around a certain axis of rotation, measured in degrees per second. To obtain a measure of input angular speed, we compute two quaternions Q_{t0} and Q_{tf}. (quaternions are mathematical representations that describe the orientation of an object in 3D space, they are presented in Chapter 5). Q_{tf} is computed from the latest mouse sample received from an input-reading subsystem that runs in parallel to the rendering process. Q_{tf} represents the position where the model should be at the current time (t_f), according to the user input. Obtaining Q_{t0} is a bit more difficult. Q_{t0} reflects the position of the model as it was at a certain time point in the past (t_0). First, we determine t_0 based on the current time and a time constant, which is the desired age of the sample

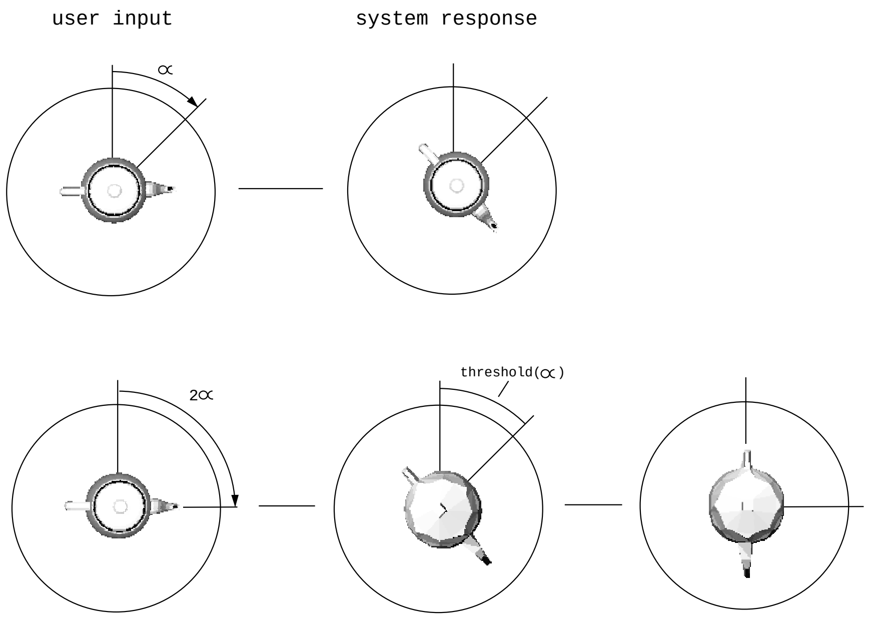

Figure 4.6: LOD selection based either on angular disparity or angular speed. Both input and response occur during successive lapses of time of equal length.

that we want to compare to, in our case $t_0 = t_f - 250\,msec$. Next, we determine Q_{t0} by first calculating a linear regression of the mouse position at time t_0, based on the mouse samples received since t_0, and then computing the orientation corresponding to the result of the regression.

We then compute a quaternion Q_r that represents a rotation R between Q_{t0} and Q_{tf} and we obtain an axis and an angle or rotation (α) out of Q_r. Finally, we obtain a value of angular speed (ω) by dividing α by the elapsed time between t_0 and t_f.

$$\omega = \frac{\alpha}{t_f - t_0} \tag{4.1}$$

During interaction, we compare the angular speed obtained in (4.1) with a threshold angular velocity t_ω to determine if detail should be switched.

Switch from *full-detail* to *coarse-detail*. If the value of the input angular speed exceeds the threshold level, then the current drawing operation is discarded and a representation at a low LOD is used producing near to instantaneous update of the current model position. The following pseudocode illustrates the usage of the criterion:

```
while (rendering full detail){
  omega= get Omega(last input sample, previous input sample)
  if (omega > threshold full to coarse)
  then switch to coarse detail
  }
```

Switch from *coarse-detail* to *full-detail.* If coarse detail is being rendered and the input angular speed is below a certain threshold angular speed, the system switches back to full detail:

```
while (rendering coarse detail){
  omega= get Omega(last input sample, previous input sample)
  if (omega < threshold coarse to full)
  then switch to full detail
  }
```

Behaviour of the angular speed switch. The angular speed switch was satisfactory in the sense that it behaved as expected. Practical values for the angular threshold were found around 30 degrees per second. At these values, the system would switch soon enough to allow for smooth manipulation of the model at a low LOD.

An unanticipated observation was the fact that the responsiveness of the angular speed switch was not very good during the initial rotation of the model. It was observed that if the model was static and it was suddenly moved, the reaction from the switch was delayed. The possible reason for this lies in the integration of samples over time. The integration of samples over time introduces undesired attenuation of sensibility to change in mouse position. This attenuation depends on the time interval selected for the regression. For example, if the required age sample is one second, then we should integrate samples from one second before the current time, and it will be hard that the current input outweighs one second of input samples where no change has occurred.

The switch seems to be more appropriate for switching from coarse-detail to full-detail. In this case, the delay due to sample integration over time does not necessarily slows down the user, and it is most likely that the delay due to full detail rendering becomes a more dominant factor. In the next section we present an enhanced proposal for a switch from full detail to low detail.

4.2.2 Switching based on angular disparity

As discussed above, it is possible to obtain a measure of spatio-temporal error in 3D rotation by measuring the angular-disparity caused by delay during interaction. A measure of angular disparity is given by α, the angle of rotation between a displayed orientation of a model and the latest model orientation according to user input. Then we can use this measure of angular disparity to accomplish LOD selection.

To obtain α we need first to obtain Q_r based on a rotation that brings the model currently being displayed to the orientation specified by the latest input sample:

$$Q_r(Q_{display}, Q_{input}) \rightarrow R(axis, angle)$$

After that, the angle of rotation α obtained is compared to a certain angular threshold t_α (see figure 4.6). If α is greater than t_α, then a low-detail representation of almost instantaneous rendering time is used to update the position of the object in the display.

This type of thresholding mechanism will guarantee that the difference between the actual object position and the displayed object position is never greater than a certain value.

The coding of such a condition is very simple:

```
while (rendering full detail){
  alpha= get Alpha(input position, displayed position)
  if (alpha > threshold alpha)
  then switch to low detail
  }
```

Behaviour of the angular disparity switch. The switch provides a big advantage over the angular speed switch, in terms of the response time when sudden changes in user input occur. Since angular disparity does not require integration of samples over time, the switch can be activated as soon as the disparity-threshold is exceeded.

In this case it is essential that we have a parallel input sampling subsystem and are able to compare user input to displayed input while rendering.

Also, since the angular disparity switch is a spatio-temporal LOD control technique (see section 4.1.2), the switch adapts to the application's frame rate. If the system has low frame rates, the angular disparity will tend to be significant, and it will be very likely that at a reasonable value of t_α is exceeded. If the system exhibits high frame rates the switch be activated a lesser amount of times, provided the same user input is given in both cases.

It is hard to envision use of the angular disparity switch to change from coarse-detail to full-detail, since the disparity will always be under a certain threshold. In this case, the angular speed switch can be more effectively used to set a switch from coarse-detail to full-detail.

We chose to focus our experimental research in the angular-disparity based switch because it provides better reaction times to sudden variation in user input, and it adapts to different conditions of delay.

4.2.3 Switches comparison

In principle there is a correspondence between an angular speed threshold and an angular disparity threshold. Both switches will be activated if the change in user input specifies an increase in speed, but the angular speed switch will be less reactive.

A key disadvantage of the angular-speed based for switching from full-detail to coarse-detail is that it does not take into account the system's responsiveness. The angular disparity switch will be activated regardless of whether there is significant spatio-temporal error when using the full-detail representation or not.

In general terms, it is not desirable to set a rotation switch with a low angular-speed threshold when the system has good frame rates, because the switch would be activated. In order to use this switch we would need to adjust the setting of the angular-speed threshold in correspondence to the frame rate: systems with low frame rates would require low threshold levels, and systems with high frame rates would require high threshold levels.

Although the angular disparity switch is more sensitive to frame rate delay, it is also possible to suggest that lower thresholds of angular disparity are more appropriate for systems with low frame rates, and higher angular-disparity thresholds are more appropriate when used in systems with high frame rates. It would be desirable to find a function that would map certain values for the angular speed or the angular disparity thresholds to levels of frame rate.

Part of the goals of this research is to find out whether a single angular disparity threshold can be useful under different conditions of frame rate delay and visual feedback, since it is possible that a single value of t_α will not be appropriate at several frame rates. It would be useful to find a hopefully small set of values of angular-disparity thresholds, and their range of applicability in terms of frame rate delay. Also, appropriate setting of t_α will probably depend on the task type. Some tasks that require finer manipulation in general will be less sensitive to spatio-temporal error than task that require ballistic motion in general.

In sum we believe the best configuration attainable for the proposed switches is to use of the angular disparity switch to switch from full-detail to coarse-detail, and to use of the angular speed switch to regulate the change from coarse-detail to the full-detail.

The experimental work presented in this thesis made use only of the angular disparity switch. The use of both switches in conjunction opens a series of questions about the interaction between both switching values under several conditions of delay due to Graphics Computation which are left for future work.

Chapter 5

Three-Dimensional Object Orientation

5.1 Orientation in 3D graphics

Almost all 3D graphics applications need to allow some sort of object manipulation. Model manipulation is essential in activities that involve design of 3D models and scenes by computer, such as engineering design (CAD), computer animation (3D Studio Max) and 3D model design (Alias wavefront). Many other applications in 3D graphics use model manipulation for the purposes of visualization of a model, such applications include browsers for 3D environments on the internet (CosmoView) and 3D drawing packages (like Silicon Graphics' Showcase).

Two essential elements of model manipulation are object orientation and positioning. In this thesis, we chose to study object orientation because it is a practical problem in 3D graphics, and our concepts of LOD selection based on speed of interaction and spatio-temporal difference can easily be adapted to the domain of 3D rotation tasks. Furthermore, there are standard ways to accomplish 3D orientations using mice. Motion in 3D Euclidean space does not offer so many advantages and is more challenging in terms of implementation of our concepts (see Chapter 7).

5.2 Mental Rotation

Rotation is a very common task in everyday life. Rotations are performed both physically and mentally. Mental rotations precede actual rotations in most cases, and occur more often than one might think. For example, mental rotations are performed in order to recognize an object that is not in its normal upright position (see [Joli88]).

[Shep71] showed that the time required to recognize two images displaying objects of the same three-dimensional shape in dissimilar orientations was direct and linearly dependent on the angular difference between the two objects displayed. [Joli88] indicates that for rotations of up 120 degrees there is a linear relationship between object recognition time and the degree of disorientation of the object to be recognized. [Shep71] used simple rotations, i. e., rotations around axes aligned with the picture plane (see Figure 5.1). [Joli88] used only rotations around the picture plane (see Figure 5.1, top). In any case, this research indicates that rotations require *per se* a certain amount of time to be performed, proportional to the extent of the rotation.

In 3D graphics, rotations occur around arbitrary axes of rotation (see Figure 5.2), because in real life we achieve rotations in this way. Rotations around arbitrary axes of rotation (not aligned with the screen or picture planes) certainly take more time than rotations around primary axes, as indicated by [Chen88]. In fact, [Zhai98] mentions that humans cannot effectively perform rotations about arbitrary 3D axes. In Zhai's experiment a 3D docking task was used and a measure of coordination was defined. Their results indicate that in experiments involving simultaneous rotation and translation by means of a 6 DOF docking task, subjects were significantly less efficient (in terms of the coordination measure) in trials with arbitrary initial rotation mismatch than in trials with rotation mismatch about primary axes.

A study by [Pars87] showed that recognition times for objects in different orientations vary according to the alignment of the axes of the orientation difference between a pair of objects with respect to the observer's visual reference frame. The slopes of the functions that correlated recognition time and orientation difference varied for different axes; for example, the slope was steepest for axes not in any principal plane of the observer's reference frame and was slightest for the horizontal axis perpendicular to the line of sight. These results let us expect an additional source of variation in our experimental data, since we use random orientations for matching. We used random orientations to be able to make conclusions about human performance for arbitrary rotations in general.

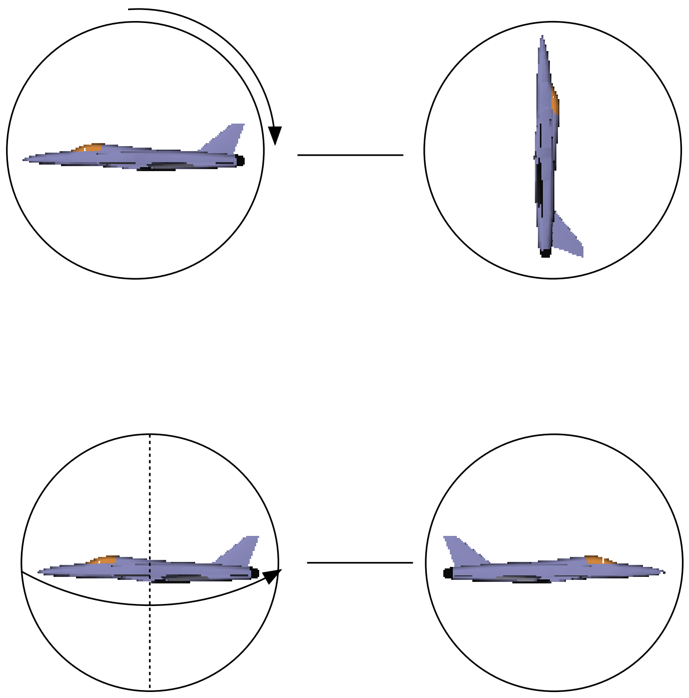

Figure 5.1: Rotations about primary axes of rotation (or simple rotations). Top: rotation in the picture plane. Bottom: rotation around a vertical axis in the picture plane.

5.3 3D Rotation in Computer Graphics

Rotation and translation studies have been done by a number of researchers in virtual reality and human-computer interaction.

[Zhai96] suggests the use of the fingers in 6DOF input devices. The task analyzed was 3D docking. Two interfaces were compared to achieve the docking task. The first was a modified 6DOF glove (the original glove has many more DOFs), and the second was the FingerBall, a 6DOF magnetic tracker shaped for easy manipulation with the fingers. Results showed significant improvements in task completion time for the FingerBall. Task completion times decreased rapidly from 12 to 8 secs across trials. The learning effects seemed to decrease after 20 minutes of practice. In our experiments, we detected strong learning effects and provided users with approximately 20 minutes of practice as well (24 trials).

[Ware98] studied the comparative effects of rotating a real handle versus rotating a virtual handle for manipulation of a model in a VR environment. The results suggested that the position mismatch that occurs between the location of the input

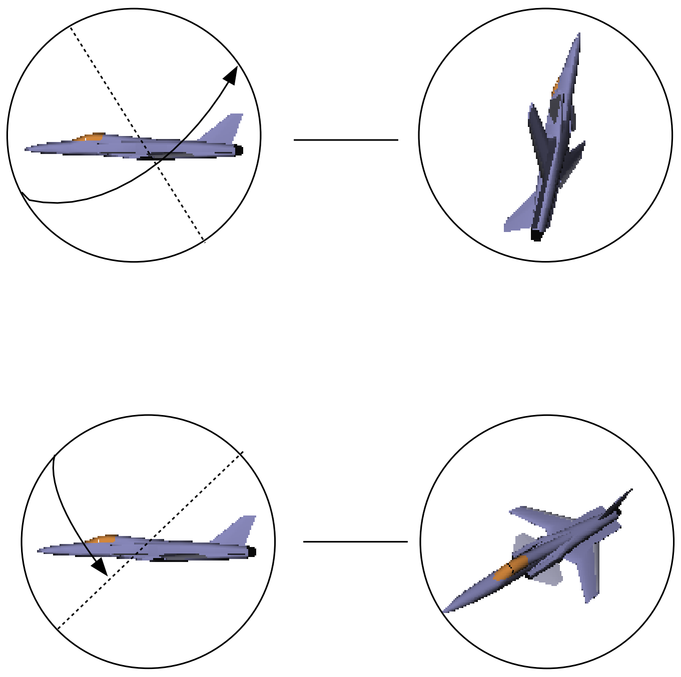

Figure 5.2: Rotations about arbitrary axes of rotation. The dotted line represents the axis or rotation.

device and the object manipulated in the 3D display increases task completion time (5 versus 3.7 secs in the optimal condition). Results also indicated that for a two-handed rotation matching task where the manipulator device is spatially superimposed with the 3D object, virtual object orientation can be as fast as real object orientation. Users of this kind of setup accomplished rotations within 1.8 to 3.2 secs in the second experiment, which used two-handed input. The speed of these rotations is extremely high when compared with the Arcball or the Virtual Sphere (discussed below) and correspond to an ideal configuration not often attainable in practice.

[Wang98] studied the interrelationship between object transportation and orientation for a linear docking task (i.e., placement and alignment). The proposed framework suggests that the structure of object transportation and orientation is concurrent and interdependent, where orientation occurs simultaneously to transportation, short after the start of transportation and ending before the end of transportation.

In this thesis, a 3D rotation matching task using the mouse as an input device has been selected. A number of reasons lead us to this decision. As mentioned in Chapter 4, the concepts of virtual speed and distance can be adapted in the 3D rotation domain as angular speed and angular disparity.

3D rotation using a mouse is a solved problem. The Virtual Sphere and the Arcball (explained below) have become the most efficient ways to accomplish 3D rotation using the mouse, and programming code to do 3D model manipulation using this input device is readily available.

In this thesis we propose the use of delay control through frame-latency manipulation to enhance interaction. In order to achieve this kind of control over frame rate and delay, it is important to track input with a device that has low delay. The mouse is an ideal interface, because it tracks with very little delay, and is one of the most widely used input devices.

5.3.1 The Virtual Sphere

Michael Chen designed and implemented the Virtual Sphere [Chen88]. The Virtual Sphere is a software abstraction that allows 3-dimensional object manipulation using the mouse.

The Virtual Sphere, also referred to as the Virtual Trackball or 3D Trackball [Fole93], simulates the existence of a hemispheric trackball device (like those on many of today's laptop computers) on top of the 3D model on the screen. The hemisphere of the trackball projects to a circle in the screen.

The mouse pointer indicates the point of contact with the trackball, and the model is manipulated by depressing a mouse button. Dragging the mouse pointer from left to right in a horizontal line along the center of the window will cause sideways rotation around the model's vertical axis (y-*axis*), as in Figure 5.1, bottom. A drag from the top to the bottom will cause the corresponding movement of the model. A drag along an arbitrary line inside the circle will cause the object to rotate in an axis perpendicular to the line of the drag (as when using a real trackball, see Figure 5.2). If the mouse pointer is dragged outside the boundaries of the circle, the object can be rotated around the depth axis, perpendicular to the screen space (see Figure 5.1,top).

A full sweep from one edge of the circle to the opposite side will rotate the model 180 degrees, and a full sweep around the circle will rotate the model 360 degrees.

Since the mechanism maps a sphere in the model space to a circle in the screen space, rotations of greater angular extent occur when doing model manipulation close to the inner border of the circle than when doing manipulations on the neighborhood of the center of the circle. This effect is hardly noticed by novice users.

5.3.2 The Arcball

Ken Shoemake implemented the Arcball [Shoe94]. The Arcball is a controller that behaves in a very similar way to the Virtual Sphere, but has some improvements in terms of implementation, functionality and visual feedback. The Arcball controller uses quaternions (see *Quaternions* below) in its implementation.

The Arcball presents no hysteresis, which implies that the Arcball is not path-dependent: in the Virtual Sphere, dragging the mouse in a closed circular loop will move the object in such a way that the orientation of the object will be different at the end of the action compared to the orientation at the start of the loop; with the Arcball, a closed-loop drag of the mouse anywhere in the window will leave the object in the same position as when the drag started.

The Arcball implementation provides a simple way to add constraints to the motion, so that the rotation is restricted to specific axes of rotation. The axes of rotation that are constrained in the implementation by [Shoe94] are either axes relative to the model coordinates, or axes relative to the screen space (used to make simple rotations). Visual cues, like arrows in the direction of rotation, provide feedback not available with the Virtual Sphere.

For the work presented in this thesis, we accomplish 3D rotation with an unconstrained implementation of the Arcball. Only the circle that represents the boundary of the Arcball was provided as feedback (in addition to the manipulated model itself).

With the Arcball a full sweep across the circle will rotate an object 360 degrees, and a full sweep around the circle will rotate the model 720 degrees. Shoemake reported that this difference is generally not significant from the user's point of view.

A full description of the Arcball is provided in [Shoe94], along with the code for implementation.

Quaternions

A quaternion is a mathematical way of describing orientation. It offers several advantages over the more widely known Euler angles [Fole93]. A quaternion's four values store an axis of rotation and an angle of rotation around the axis. A set of three orthogonal vectors that define a unique orientation (or base) can be derived from a quaternion and associated with an object's orientation.

Two orientations represented by two quaternions can be composed by quaternion multiplication, which does not require the use of trigonometric functions. Thus, rotations can be composed in a fast and accurate manner.

5.3.3 The Rolling Ball

The Rolling Ball, presented by [Hans92], was produced after the implementations of Chen and Shoemake.

The Rolling Ball transformations work to define quaternion rotations like Shoemake, but in a slightly different way: with the Rolling Ball small clockwise rotations about a line perpendicular to the screen plane (*z-axis*) are carried out by moving the mouse in small, counterclockwise circles. This behaviour does not seem to be as intuitive as the one provided by the Arcball, the technique implemented in the present work.

5.3.4 User Studies on 3D Orientation Using the Mouse

When Chen introduced the Virtual Sphere, he presented a comparative study testing the Virtual Sphere against other approaches available at the time. A few years later, Ken Shoemake produced the Arcball, but did not present a comparison between the Arcball and the Virtual Sphere. In 1997, [Hinc97] presented an extended usability analysis of 3D rotation techniques, which included 3DOF input devices. In the following sections the studies by Chen and Hinckley are described, taking a careful look at the experimental design and performance results, since these studies provide a useful framework for the experimental analysis to be presented in this thesis.

A Study in Interactive 3D Rotation Using 2D Control Devices [Chen88]

In this study the Virtual Sphere was presented and compared with three other controllers that used the mouse (see Figure 5.3):

1. A controller based on 3 horizontal sliders, where each slider was used to rotate

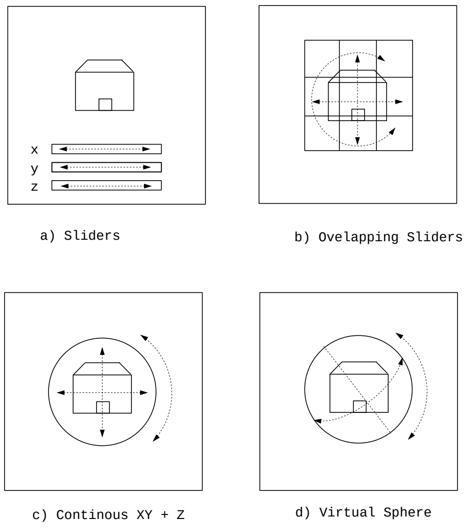

Figure 5.3: Screen display of the four virtual controllers with object in center; the dotted lines indicate the regions available for manipulation and direction of motion (adapted from Chen et al, 1988).

the model in each of the *x*-, *y*- and *z*-axes in the screen space (Figure 5.3a).

2. A controller made up of a set of overlapping sliders represented by a squared grid of 3 by 3 sliders, which was superimposed on the object being rotated (Figure 5.3b).

3. A dual-mode XY control with additional Z, where a circle was overlapped on top of the model, and rotation was either bounded to the *x*- and *y*-axes if the drag occurred inside the circle, or about the *z*-axis if the drag occurs around the circle. This controller was a constrained version of the Virtual Sphere (Figure 5.3c).

Experiment Task Subjects were shown a solid-rendered, upright house in color on the right hand-side of the screen and were asked to match its orientation to a tilted house on the left-hand side of the screen. Subjects were told to press the space bar when satisfied with the match, and were instructed that both speed and accuracy were important. Subjects' performance was categorized as either "Excellent", "Good Match", or "Not good enough, try harder next time". A rating of "Excellent" would be obtained if the rotation mismatch was less or equal to 5.7 degrees. A rating of "Good Match" would be given is mismatch was not "Excellent", but was less than or equal to 7.6 degrees. All greater mismatches were "Not good enough". The categorization was provided as feedback to the subject.

Experimental Design The goal of the first experiment was to compare subject performance using the four controllers mentioned above. The main performance measures were time to complete the rotation and accuracy in performing the task.

Chen tested 12 right-handed male subjects, each subject tried all four controllers (within subject design) and the order of controllers was counterbalanced using a Latin square design [Elme92].

For each controller there were nine different non-upright house positions to be matched. Each one of these orientations were presented three times for a total of 27 trials per controller. Three of the nine orientations required only simple rotations about the x-, y- or z-axes. The other six required rotations about arbitrarily defined axes.

Results and conclusions The average time registered for complex rotations with the Virtual Sphere was around 17.5 secs.

Results showed that for complex orientations, the Virtual Sphere and the dual mode XY-Z controller were clearly faster than the sliders. On the other hand, the slider controllers produced significantly faster performance for simple, single-axis tasks.

The rotation task used in this study was adopted for experimental tests in this thesis. Pilot studies for the research in this thesis indicated that it was fairly difficult to get a rating of "Excellent". The rating of "Good Match" was also rarely obtained, as the requirements of accuracy were not that much different. We believe that this was because we included conditions of delay, thereby increasing the level of difficulty of the task. We decided to slightly relax the accuracy requirement in our tasks, to avoid user frustration and reduce the expected completion time for our experiments.

Usability Analysis of 3D Rotation Techniques [Hinc97]

Hinckley et al performed a formal user study of interactive 3D rotation using the Virtual Sphere and the Arcball, as well as two free-space 3D input techniques based on magnetic orientation sensors.

The goal of the study was to provide solid performance data about each controller for the experimental rotation matching task. Another goal was to collect qualitative observations about the strengths and weaknesses of each technique.

The four interaction techniques used were the Virtual Sphere, the Arcball, the 3D Ball, and a standard 3D Tracker.

The 3D Ball is a a hand-held ball-shaped orientation sensor instrumented with a magnetic tracker, which is used as a 3DOF rotation controller to manipulate the virtual object. The orientation of the object being manipulated always matches the orientation of the 3D Ball.

The 3D Tracker is identical to the 3D Ball in all regards except the physical packaging, the Tracker is just the magnetic orientation sensor as shipped by the manufacturer; it is much smaller and irregularly shaped.

The hypotheses tested in the study relevant to this thesis are:

H1: Users can effectively use coupled rotation axes, and integrated control of all three degrees of freedom for rotation will provide faster input of orientation data than what is possible with decoupled controllers like the Virtual Sphere or the Arcball.

H2: A multidimensional input device can provide fast orientation input without necessarily sacrificing any accuracy, as opposed to mouse-based 3D input (Virtual Sphere and Arcball), in which faster interaction comes at the expense of precision.

H3: The Arcball includes several apparent improvements over the Virtual Sphere. As such, the Arcball is expected to outperform the Virtual Sphere in terms of task performance, user acceptance, or both.

Experiment Task The same same orientation matching task employed by [Chen88] was selected by [Hinc97]. The only difference was that to end a trial, users clicked a foot pedal. Feedback about performance was also given in exactly in the same way.

Experiment Design A within-subjects Latin square was used to counterbalance the order of presentation. Twenty four users, twelve male, twelve female tried all four interfaces in a single session lasting about 1 hour. The users performed matches for 15 unique orientations with each interface,but only the last 10 of these were included in the results analysis to reduce learning effects.

The dependent variables were task completion time and accuracy, which was measured by the shortest rotation between the final user-specified rotation and the expected matching orientation.

Results and Conclusions The 3D Ball was 36% faster than the 2D techniques, while the Tracker was 33% faster, supporting hypothesis H1.

There was little variation in the mean accuracy obtained through all conditions, and errors due to inadequate control of the input device can hardly be perceived, supporting hypothesis H2.

Participants in Chen's experiment had been faster (averaging a mean of 17.5 secs versus 27.7 secs for complex rotations), but less accurate (averaging a mean of 8 degrees versus 6 degrees).

Sex and order of presentation were statistically significant factors. In the analysis by sex, the 12 males reported an average task completion time of 22.1 secs, and an average of accuracy of 6.3 degrees, while the females reported an average task completion time of 33.5 secs, and an accuracy of 5.9 degrees. These results indicated that performance in the rotation matching task varies between sexes. Due to limited time and human resources it was decided that only males would participate in the studies for this research.

There was no information that supported hypothesis H3. The Arcball was not any better than the Virtual Sphere either in subjective terms or in performance times or accuracy.

Hypothesis H1 is in apparent contradiction with [Zhai98], who state that users cannot effectively perform 3DOF rotations. However, Zhai refers to a measure of user coordination using 6DOF devices, not completion time. No matter how inefficiently multiple DOFs input devices are used, it is clear that this sort of manipulation results in faster task completion times.

Results from Hinckley indicated that for our experiment, the choice of interface to be used for experimentation (either the Virtual Sphere or the Arcball) could be done more or less freely, since both interfaces produce similar performance.

Chapter 6

Experimental Analysis

Two experiments were performed to quantify the potential benefits in human performance that may be obtained through angular-disparity based switching in 3D rotation tasks. Angular-disparity based switching is discussed Chapter 4.

The first experiment was carried out to compare the proposed technique with existing control alternatives. The second experiment was performed to examine the relationship of the disparity based technique to visual detail.

6.1 First Experiment

6.1.1 Experimental Motivation

The first experiment aims at quantifying the benefits provided by angular disparity-based switching.

Under the switching technique in question an alternative low detail representation will be used whenever a certain amount of angular disparity (in degrees) is reached. This technique will be compared against two alternatives:

1. No switching at all, or "Never switching"

2. Switching every time motion is detected, or "Always switching"

In the context of the proposed switching technique, we have found through pilot studies that loss of detail harms performance in two ways: (1) by inserting popping artifacts and (2) by loss of visual cues.

Popping artifacts are the visual changes that are present when a system uses two different level of detail representations and switches back and forth between them. Loss of visual cues can be explained due to simplification. During simplification some relevant features may be lost (a relevant feature is one whose presence or absence influences the task completion). Loss of visual cues can be harmful to performance depending on the relevance of the features lost, as some pilot studies indicate.

Hypotheses

We formed the following hypotheses:

H1: Performance is affected by delay. We expect performance to degrade as delay increases, in conformance to literature on effects of delay (see Chapter 2).

H2: High visual-temporal quality and limits in delay improve performance, even at the expense of visual detail. Therefore, the introduction of any alternative low detail representation which compensates delay should allow for better performance than no compensation at all, especially when frame rates are low.

H3: Performance is affected by popping. Popping occurs whenever a switching technique is used, but when frame rates are low the improvements in performance that we expect according to H2 should dominate over the effects of popping. Therefore, the effects of popping by itself are only measurable when delay is the same for all switching techniques. In our experiment, that happens only when delay is at the lowest level (frame rate is highest).

H4: Neither "Never switching" nor "Always switching" are sufficiently good management strategies. "Never switching" will become worse as delay increases. "Always switching" will be good when delay is high, but undesirable when delay is low, because it unnecessarily degrades visual detail and introduces popping.

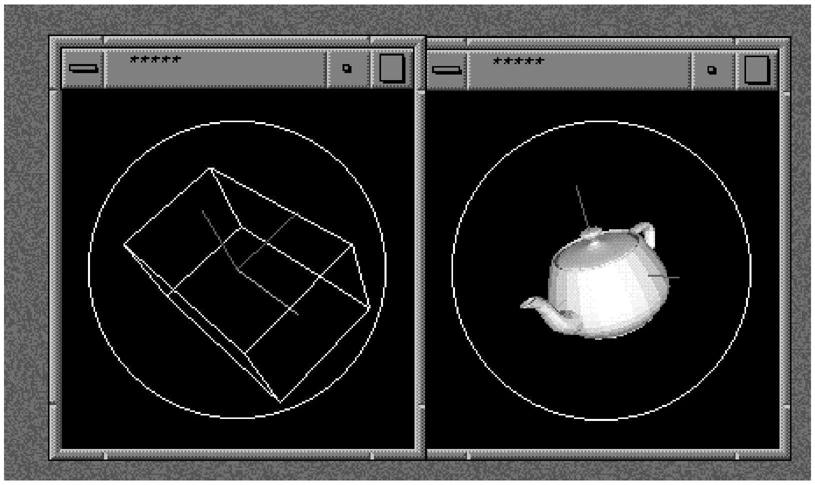

Figure 6.1: Experiment setup. Users normally manipulate a fully detailed model. When the angular disparity threshold is exceeded, a bounding box is used as a low LOD representation.

H5: By limiting delay and enforcing feedback accuracy, the thresholding technique not only yields better performance, but is also more comfortable to use. We expect subjects to agree with us in a subjective evaluation of the experiment conditions.

6.1.2 Experimental Design

Conditions

Based on the hypotheses mentioned above, we decided to use two independent variables. We combine four frame rates with three switching techniques ($4FR \times 3SwTech$) as outlined below.

Switching techniques The system has a built-in angular disparity based switch, based on the concepts presented in Chapter 4. During interaction, a full-detail representation is normally used. During rendering of frames, the position corresponding to the latest input sample is compared against the position of the model being displayed, and an angle of rotation between these two positions is computed. If this angle exceeds a certain threshold value, a low-detail representation is used. In this implementation, the low detail representation used is the bounding box of the model, drawn with three short axis that describe the model's orientation (see Figure 6.1). The three switching

FR(Hz)	Delay(msec)
20.0	50
6.67	150
4.00	250
2.20	450

Table 6.1: The Four levels of frame rate used in the experiment

conditions are:

1. No switching at all, or "Never switching"

2. "Threshold-12 switching". If the angular disparity between the latest input sample and the model shown at the display exceeds a threshold value of 12 degrees the switch to a lower detail representation is activated. We chose 12 degrees because we found in pilot studies that this value allows for smooth interaction during ballistic input motion and also allows the use of the full detail model for fine input motion.

3. Switching every time motion is detected, or "Always switching". In this technique the system switches to an alternative representation whenever motion is detected.

Both of the last two switching conditions eliminate most of the delay during manipulation. When the switch is activated, the low detail representation is almost instantaneously drawn (at a 72Hz frame rate). The system switches back to a full detail representation when the user stops moving. In this case, the delay corresponding to the simulated frame rate will be introduced, and the user will have to wait to obtain the full-detail representation.

Levels of frame rate To simulate several conditions of delay due to rendering, four levels of frame rate were selected based on a frame time of 50 msec, and adding 100, 200 and 400 msecs to the base frame time (see Table 6.1). These values were intended to represent typical conditions of delay present in current graphics systems.

In this study, frame rate is controlled with frame-latency manipulation (*flm*) as follows: an input sample is read and delay is introduced before sending the corresponding image to the renderer.

Frame rate was controlled at these levels to simulate delay whenever the system was using the full detail representation.

Task

The experiment involved an orientation matching task similar to that used by [Chen88] and [Hinc97].

Users were instructed to manipulate the model on the left to match the orientation of the other model. In some cases, they were allowed to interchange the windows if they wanted to. When they decided that they had matched the orientations, they pressed the spacebar key.

The system then evaluated the quality of the match by measuring the minimum angle of rotation about an arbitrary axis of rotation needed to complete the match (that is, the error in accuracy).

With this information, some feedback about the user's performance was provided. If the error with respect to the final orientation was less than or equal to 15 degrees the feedback was: "That was a Good match!" and "The time it took was __ secs", indicating the completion time for the trial. If the error was more than 15 degrees, the feedback was "Not Good enough, Try harder next time, please!". In this case we chose 15 degrees, because we found the accuracy requirements of [Chen88] and [Hinc97] were too high (see Chapter 5). With these requirements users tended to either express frustration or spend large amounts of time trying to achieve good matches. Since a large number of trials needed to be done, we decided to tolerate more error in accuracy to reduce user frustration and allow faster completion times.

Administration of conditions First, users were given 24 practice trials, in blocks of eight, grouped by switching technique. In each practice trial a random orientation was presented together with a random frame rate chosen from the available values.

After that, the twelve conditions were administered to each subject ($4FR \times 3SwTech$). Subjects performed 10 trials in each condition. Conditions were grouped by switching technique in three blocks. Within each block, the four different frame rates were used. A set of 40 random orientations was used to define the target rotations in each block. All users worked with the same set of 40 orientations.

Scale	Meaning
1	Very Bad
2	Not Good
3	Acceptable
4	Good
5	Very Good

Table 6.2: Subjective rankings applied to each switching technique.

Complete counterbalancing was used in the administration of the blocks. Since there are six possible block orderings, each order was applied to two subjects. The order of application of the frame rate conditions within each block was chosen at random. After each block a break was offered; in practice, users took short breaks of about five minutes or less.

At the end of the experiment or during breaks, subjects were asked to give their general impressions about the switching mechanisms they tried.

Dependent Variables There were five dependent variables: Task completion time, Accuracy, Coordination, Percentage of time in full detail, and Subjective evaluation.

1. *Task completion time* measures how fast a user completed a match. It is measured from the moment the two images are rendered until the moment the user presses the spacebar to indicate a match.

2. *Accuracy* measure how good a match was. It is the difference in degrees between the target orientation and the orientation indicated by the user at the end of the trial. Values below 15 degrees were judged acceptable in feedback to subjects.

3. *Coordination* evaluates user efficiency in performing the task. It is measured by dividing the length of the path the user followed by the length of the most optimal path that could be taken to complete the required match. This measure was used by [Zhai98] for evaluation of 3DOF input devices.

4. *Percentage of time in full detail* describes the effects of the switching technique in use. It is the percentage of time the system has been displaying or drawing a full detail representation with respect to task completion time. While

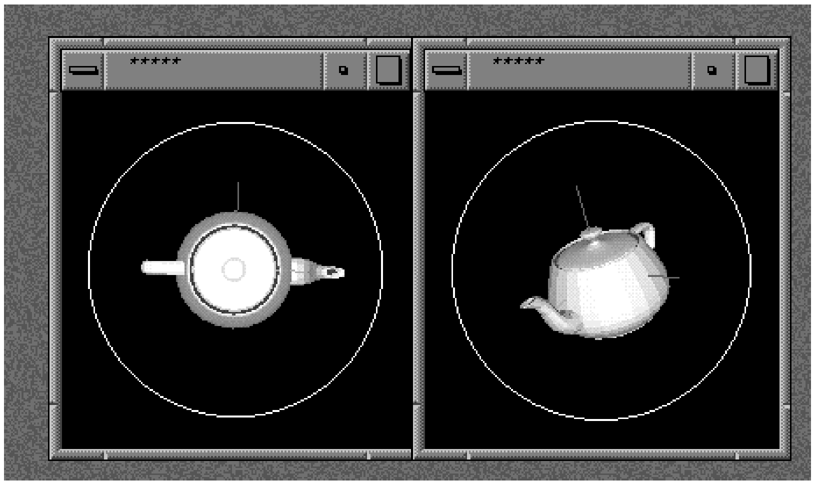

Figure 6.2: On the left, the teapot in the canonical position. On the right, the position to match

this measure is heavily dependent on switching technique, it is also sensitive to subject behaviour with respect to the switching approach; specifically, this percentage of time varies depending on whether subjects need to stop and evaluate the full detail representation or they feel comfortable using the low detail representation.

5. *Subjective Evaluation.* Subjects were asked to evaluate each of the three switching techniques they tried with a discrete scale as defined in Table 6.2.

Setup

Before the start of the experiment, users received a sheet with information about how to use the Arcball. They were instructed about the experimental procedure and were asked to be as fast as possible while keeping the errors at a minimum level.

At the beginning of the experiment, two windows (600 by 600 pixels) were shown side by side. Both windows showed a well known teapot model (well known in the graphics community). The window on the left showed a model of the teapot in the canonical position. The window on the right showed a static view of the teapot in a random orientation (see Figure 6.2).

The model was drawn flat-shaded with black background and white light. It is made up of 1177 vertices and 2256 faces and is drawn with three short axes. This set of axes (also called the base) describe the orientation of the object. In this case, each axis is color coded: purple, red and green for the x- y- and z-axes relative to the model. The Arcball, described in Chapter 4, is used to manipulate the model on the left. The circle drawn around the model indicates the boundary between the two manipulation modes available.

Apparatus The experiment was performed on a Silicon Graphics Octane/SI+t-128 with two 175 MHZ IP30 Processors (only one was used), a MIPS R10000 CPU, 128 MB memory in RAM and a 20" monitor (1280x1024) @ 72Hz. The software was implemented in C and C++ using OpenGL and the Graphics Library for X-windows (GLX) running under IRIX v. 6.5.3.

Subjects Twelve male volunteers were recruited in total, eleven from the Department of Computing Science and one from the Department Physical Education of the University of Alberta. Participants were between 22 and 40 years old, the average age was 26 years. Five people reported having some previous experience with 3D object manipulation using the mouse. Like [Chen88], we decided to have only male participants in our study, to eliminate variation due to gender differences. Variation of results due to gender for the task in this experiment was reported by [Hinc97].

6.1.3 Results

A repeated measures (within subjects) analysis of variance was performed on the results for Task completion time, Accuracy, Coordination, and Percentage of time in full detail; significant results are summarized in Table 6.3.

1. *Task completion time.* A comparison of completion times for the three switching techniques across several frame rates can be seen in Figure 6.3. The exact values for each condition is shown in Table 6.4.

   A main effect in completion time was found for frame rate, confirming that the effects of delay are significant, in support of hypothesis H1. For the lowest frame rate, the effects of the switching technique (*SwTech*) are significant; this indicates that switching is especially necessary at low frame rates. For the

Dependent Measure	Experiment Factors	Significant Results
Task completion time	FR	F(3,33)=23.588, p=0.000
Task completion time	$FR \times SwTech$	F(6,66)=3.07, p=0.01
Task completion time	$SwTech(FR = 2.2Hz)$	F(2,22)=4.606, p=0.021
Task completion time	$SwTech(FR = 20Hz)$	*F(2,22)=2.134, p=0.142 n.s.*
Accuracy	$FR \times SwTech$	F(6,66)=2.999, p=0.012
Accuracy	$SwTech(FR = 2.2Hz)$	F(2,22)=8.572, p=0.002
Accuracy	$SwTech(FR = 6.7Hz)$	F(2,22)=2.851, p=0.079
Coordination	$FR \times SwTech \times Order$	F(12,54)=2.362, p=0.016
Percent of Time in FD	FR	F(3,33)=25.858 p=0.000
Percent of Time in FD	$SwTech$	F(1,11)=12.253 p=0.005
Percent of Time in FD	$FR \times SwTech$	F(3,33)=32.051 p=0.000

Table 6.3: ANOVAS for the first experiment. Factors are frame rate (FR) and switching technique (SwTech).

highest frame rate, the "Always switch" condition completion times were longer than the two other techniques, and *SwTech* approached significance. This last result provides some support for hypothesis H3. The last two results support hypothesis H4, that neither "Always switch" nor "Never switch" are sufficient management strategies.

In general, the interaction $FR \times SwTech$ was also significant, this supports the theory that both switching techniques reduce the effects of delay and increase performance, as mentioned in hypothesis H2. The order of administration of conditions was not a significant factor for completion times.

2. *Accuracy.* Accuracy results are shown in Figure 6.4. No significant main effects were found for either *frame rate* or *SwTech* conditions. *SwTech* was a significant factor at the lowest frame rate, where users were less accurate with "Never switch". *SwTech* almost reached significance at the *6.7Hz* frame rate, where "Never switch" was the best condition. These results give some support to hypothesis H3.
In general, accuracy did not decrease under higher levels of delay the way task completion times did. It was observed that under conditions of delay most users were still careful to accomplish good matches (which was probably due to

the feedback provided); we believe this contributed to longer task completion times. The mean value for accuracy was 8.59 degrees. Ordering effects were not significant.

3. *Coordination.* The mean value for coordination was 275%, which is similar to the mean value for one of the 6DOF controllers tested by [Zhai98]. Results of the coordination measure are shown in Figure 6.5.

   No significant main effects were found either for frame rate, *SwTech* or $FR \times SwTech$ in this measure. A three-way interaction existed for order, frame rate and switching technique. If "Never Switch" was the first technique tried, "Always switch" and "Threshold-12" were the best conditions for the two middle frame rates. "Never Switch" was best for the two other frame rates. If "Threshold-12" was tried first, coordination was the same across all conditions. Finally, if "Always switch" was the first technique tried, coordination was especially bad in the highest level of frame rate for the "Never Switch" condition. We couldn't come up with an explanation of this three-way interaction, but subject differences might have been a factor.

4. *Percentage of time in full detail.* As expected, "Threshold-12" shows a linear relationship with respect to frame rate (see Figure 6.6). The time spent in full detail is more or less constant (85% of total time) for the "Always switch" technique. For statistical purposes, only "Threshold-12" and "Always switch" were analyzed. "Never Switch" is always in full detail.

   Frame rate and *SwTech* were both significant main effects, The interaction $FR \times SwTech$ was very significant, confirming the results expected, that the switches react differently under various conditions of frame rate. The effects of *SwTech* on each level of frame rate were significant, except for the *4Hz* frame rate. For the lowest frame rate, more time in full detail was spent in "Always Switch" than in "Threshold-12". At first, this seems to be counter-intuitive. A plausible explanation is that when delay is high, the user spends more time planning future movements in "Always switch". In this condition switching to

Frame Rate:	2Hz	4Hz	6.7Hz	20Hz	Average
Always switch	23.80	23.32	20.89	21.49	22.38
Threshold-12	23.50	23.36	20.48	18.89	21.39
Never switch	27.91	22.88	21.29	18.06	22.54
Average	25.07	22.85	20.89	19.48	

Table 6.4: Task completion times for experiment one

low detail occurs for every movement, such switches bring along a time penalty in the switch back to the full detail representation.

5. *Subjective Evaluation.* "Threshold-12" was the condition most preferred by the users. The mean value for this condition was 3.75 which is very close to a grade of "Good". The other two switching techniques were considered "Acceptable" in average. Figure 6.7 shows the results (mean and standard deviation) for each switching technique. Most users said that flickering (high frequency popping) was annoying in the "Always switch" condition. These results support hypothesis H5.

Discussion

We have found evidence that thresholded switching is the best LOD management technique. "Threshold-12" was on average the best condition. This switch behaved like "Always switch" at low frame rates, and like "Never switch" at high frame rates.

The results of this experiment confirmed our expectations that frame rate delay affects performance. It has been shown that under conditions of delay, Task completion time is affected the most, while a smaller effect is occurs for accuracy.

Popping seemed to have a limited impact on user's task completion time. In the case of the subjective evaluations "Always switch" was only judged "Acceptable", and users complained about flickering (high-frequency popping).

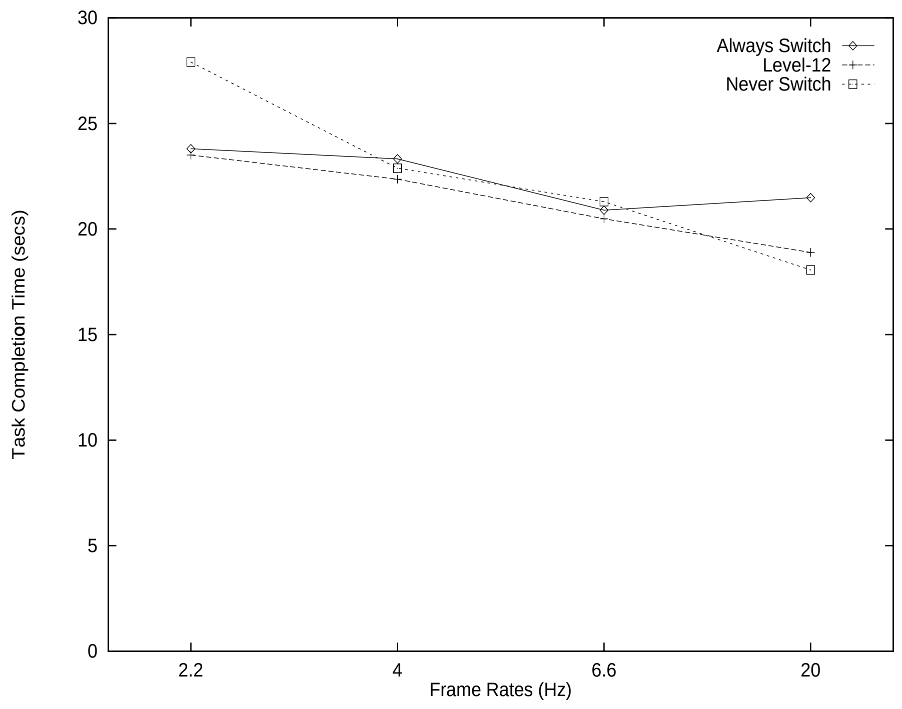

Figure 6.3: Task completion times under the three switching techniques

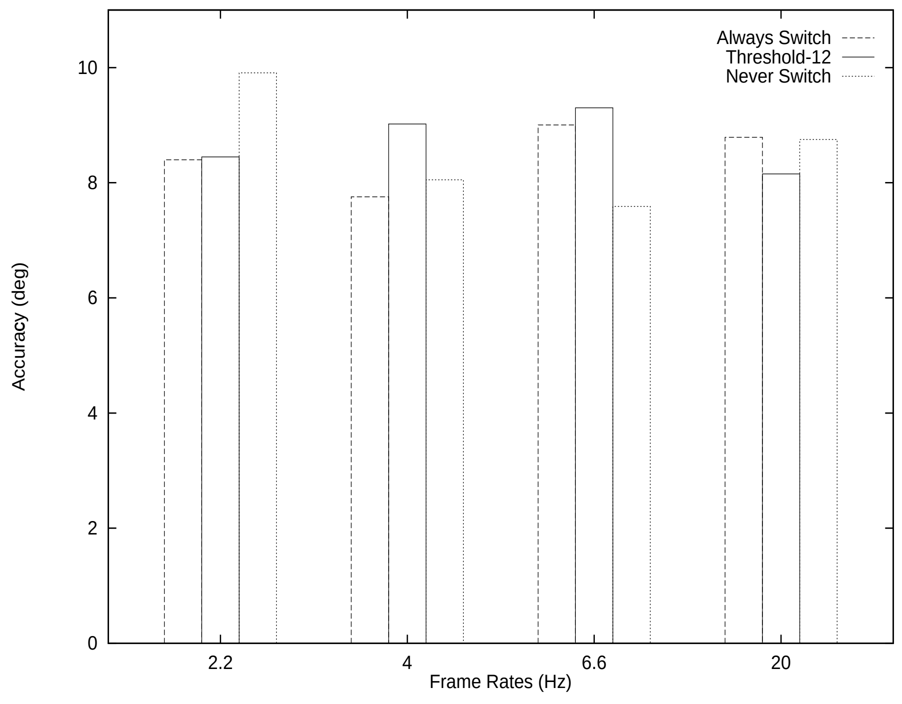

Figure 6.4: Comparison of accuracy under the three switching techniques

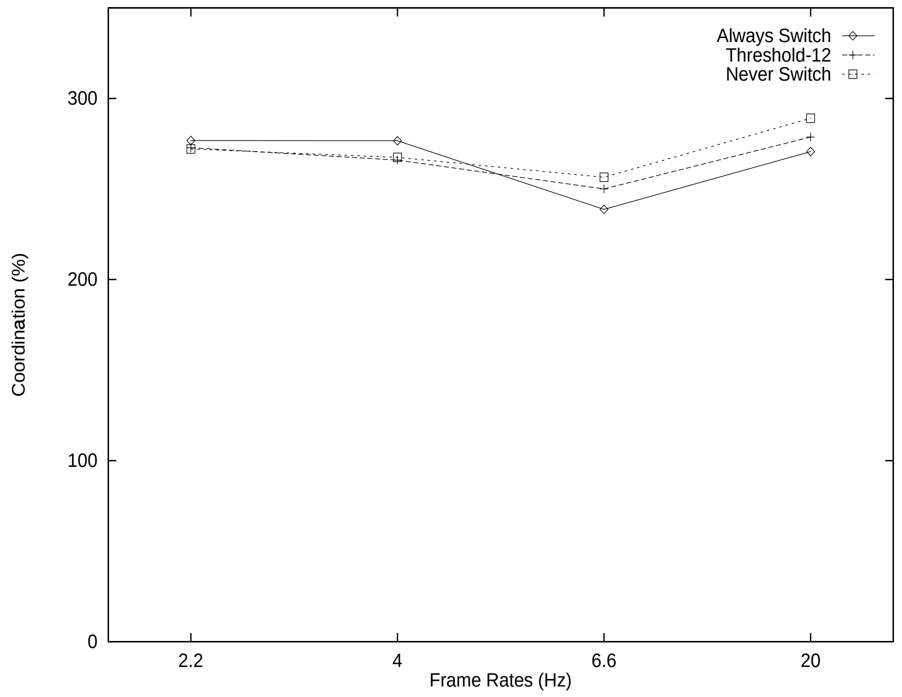

Figure 6.5: Measure of efficiency in coordination for the three switching techniques

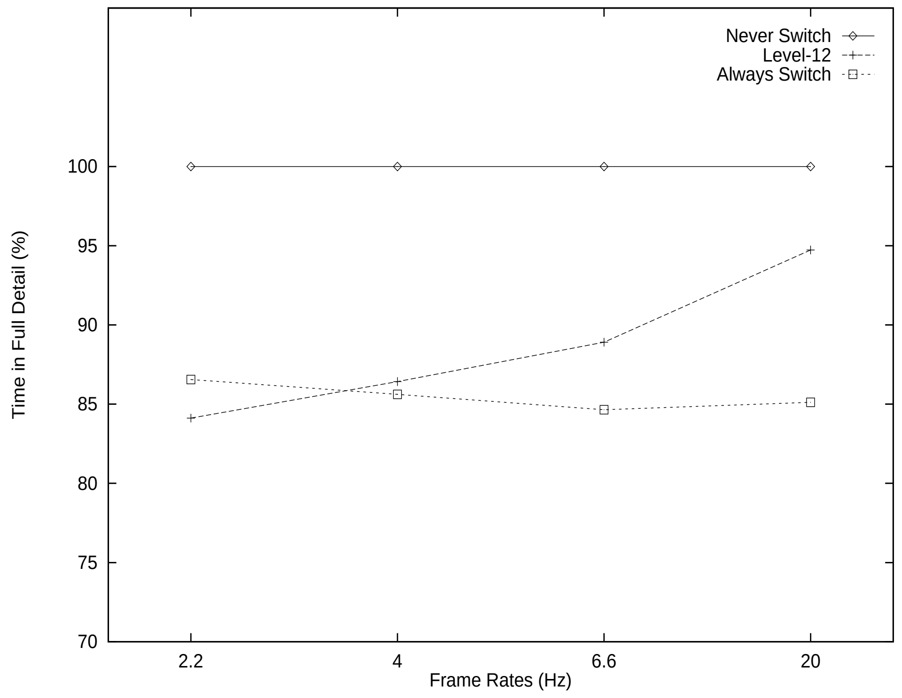

Figure 6.6: Percentage of time in full detail for the three switching techniques

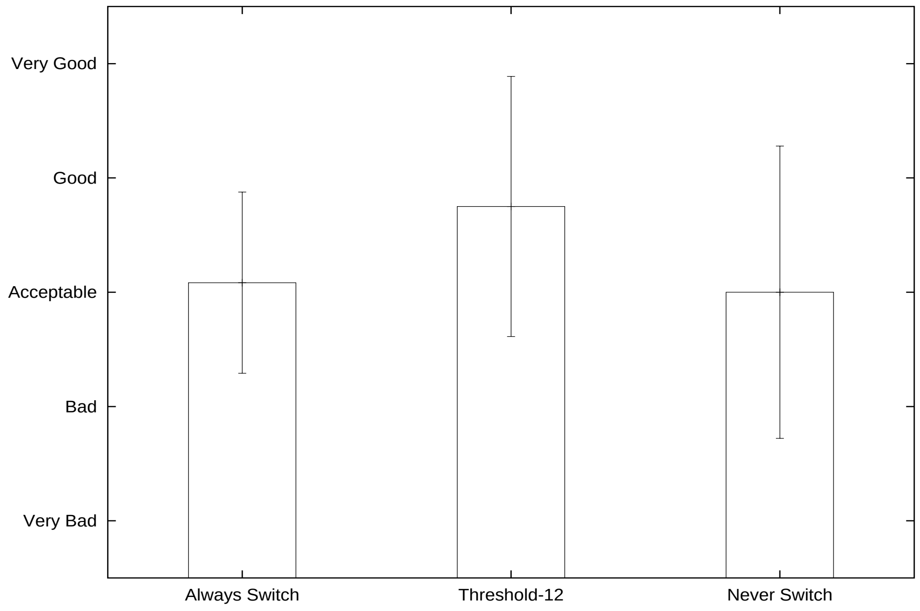

Figure 6.7: Subjective evaluation of the switching techniques

Appendix: Experimental Notes

Always coarse condition. During pilot studies we tested the experiment with an always coarse condition. In this condition, users saw only the low detail representation, a bounding box with the model's axes. The system never switched to another representation. We wanted to find out whether it was either the loss of visual detail or the effects of popping that harmed performance in switching conditions.

The rationale for this condition was as follows: This representation had no popping effects at all, but had poor visual feedback. If popping was the only variable that harmed performance, users should do as well with an always coarse representation as with a full detail representation.

We observed that the always coarse condition produced times sometimes as good as those obtained with the "Never switch" condition. On the other hand, this condition resulted in a significant decrease in accuracy. Errors in terms of accuracy were almost twice as large in this condition as in the other conditions.

We concluded from our pilot studies that loss of detail doesn't impact task completion time in a significant way, but it harms accuracy in a way that none of the other conditions do. This supports the conclusion that some high detail visual feedback is always required, even at the expense of popping effects.

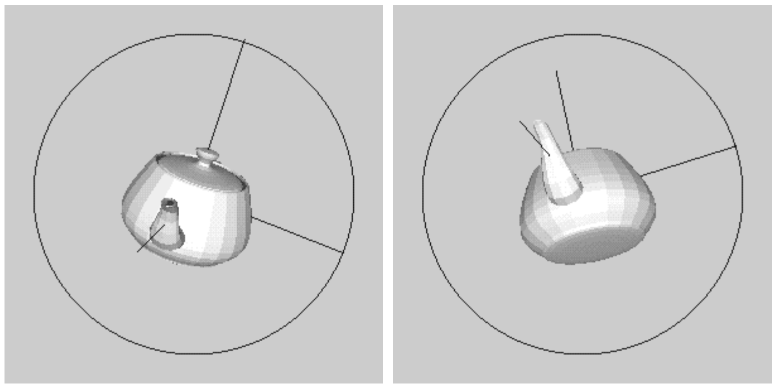

Figure 6.8: Teapot with long axis indicating the model's orientation.

Length of axes of the base. Originally, the axes of the base were quite long, extending to around 75% of the distance from the model's center to the border of the window (see Figure6.8). In many cases, they touched the boundaries of the Arcball's circle. These axes provided two visual cues that users could rely on to match the orientations. The first was the position of the axes on the circle of the Arcball. The second was the aliasing ("jagging") of the lines that composed the axes. Aliasing gave information about the relative slopes of the axis on the screen. It quickly became evident that subjects used these cues to accomplish the orientation task without ever referring to the image of the model itself. To increase reliance on the model itself as a cue, we reduced the size of the axes. In real applications, orientations are not supported by such simple matching cues; we believe that these applications could be improved if they used long axes to indicate the model's orientation.

6.2 Second Experiment

6.2.1 Experimental Motivation

In the first experiment, we found evidence that disparity based switching is the most effective approach to detail management. The second experiment was performed to examine the relationship of disparity based switching to the level of detail used in the alternative representation, and to characterize the effect of different threshold values on performance.

Hypotheses

The set of hypotheses tested in this experiment are:

H1: Lower thresholds are better than higher thresholds. In the first experiment we observed that lower thresholds are better when delay is high. We also observed that popping effects were not significant when lower thresholds were used at high frame rates. Generally speaking, lower thresholds are good as long as the effects of popping and loss of visual detail are not exaggerated.

H2: Better alternative LOD representations will improve performance. Higher levels of detail provide a better approximation to the current object's orientation, reducing the loss of visual cues. This enhancement of visual quality also reduces popping effects.

H3: An improvement in visual detail of the alternative representation should be preferred in subjective terms. The reduction of popping and the increase in visual detail should produce a better experience from the user's point of view.

H4: There is an inverse relationship between the thresholding level and the level of detail of the alternative representation. More visual detail should make it possible to lower the threshold level for switching and still allow for good performance. Threshold levels should be higher if a low level of detail is used, because the effects of loss of visual detail are increased.

6.2.2 Experimental Design

To enable comparisons between the two experiments, we tried to emulate the design of the first experiment as much as possible.

Conditions

We used three independent variables: frame rate (three levels), threshold level (two levels), and level of detail for the alternative representation (two levels). The result is a $3FR \times 2Thres \times 2LOD$ design.

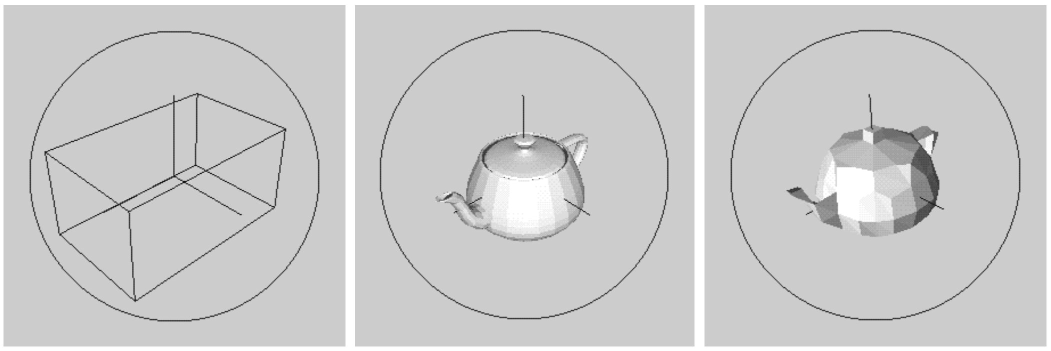

Figure 6.9: Levels of detail used in the second experiment. From left to right: the bounding box, the original model and the simplified model.

Levels of Detail Two levels of detail (*LODs*) for the alternative representations were used:

1. The bounding box (*BB*ox), used in the first experiment (Figure 6.9, left).

2. A simplified model of the teapot This model contained approximately 10% of the faces of the original model, and was generated using vertex clustering [Ross93]. This LOD technique produces fast simplifications of acceptable quality. (see Figure 6.9, right).

Threshold levels Two levels of threshold were selected for switching, based on the angular disparity switch, just like "Threshold-12" in the first experiment.

1. "Threshold-5" This threshold activates switching if the angular disparity exceeds 5 degrees. It was defined to behave much like the "Always switch" condition in the first experiment. This would reduce the effects of delay in the low frame rates, and would show less popping than "Always switch" in the high frame rates.

2. "Threshold-25" Switching is activated for angular disparities above 25 degrees. This threshold is set to enable switching for ballistic motion and disable switching during fine motion. It further reduces popping effects and enables the use of a higher LOD during fine motion. On the other hand, the effects of delay are likely to increase with respect to the other threshold.

FR(Hz)	Delay(msec)
20.0	50
4.00	250
2.20	450

Table 6.5: Frame rates used in experiment 2

Levels of Frame Rate We used three out of the four levels of FR from the first experiment. In the first experiment the intermediate values of FR produced very similar results, so the 6.*6Hz* level was not used. The selected frame rates are shown in Table 6.5.

Experiment setup

The experiment setup, task, and apparatus used were the same as the ones used in the first experiment.

Subjects 12 Male volunteers were recruited from the Department of Computing Science of the University of Alberta. We decided not to use any person who had participated in the previous experiment or in pilot studies, since significant learning effects during the first experiment were observed.

Administration of conditions Threshold and LOD levels were combined into a set of four blocks, which were counterbalanced using 3 balanced Latin squares, according to [Elme92].

In the practice session, users were given 24 trials, in four blocks of six trials for each combination of $Threshold \times LOD$. As with the first experiment, a random target orientation was presented at a frame rate randomly chosen from the preset values.

After that, the twelve conditions ($3FR \times 2Thres \times 2LOD$) were administered to each subject. 10 trials were performed in each condition. For each of the four combinations of $Threshold \times LOD$, three levels of frame rates were tested in sequence. The order of application of FR levels was randomized without replacement. A set of 30 random orientations was used to define the target rotations used in each block. Users took short breaks after each block. At the end of the experiment, participants were asked to give subjective ratings of the $Threshold \times LOD$ combinations.

Dependent Measure	Experiment Factors	Significant Results
Task completion time	FR	F(2,22)=20.931, p=0.000
Task completion time	LOD	F(1,11)=10.468, p=0.008
Task completion time	$Thres(FR = 2.2Hz)$	F(1,11)=7.619, p=0.019
Task completion time	$FR \times Threshold$	*F(2,22)=2.578, p=0.099 n.s.*
Task completion time	$FR \times LOD$	F(2,22)=6.476, p=0.006
Task completion time	$Threshold \times LOD$	*F(1,11)=0.980, p=.343 n.s.*
Task completion time	$Thres(LOD = Simpl)$	*F(1,11)=3.256), p=0.099 n.s*
Task completion time	$Threshold \times Order$	F(3,8)=6.028, P=0.019
Accuracy	FR	F(2,22)=4.467, p=0.024
Accuracy	$Order$	*F(3,8)=3.993, p=0.052 n.s.*
Coordination	LOD	F(1,11)=9.329, p=0.011
Coordination	$Threshold \times Order$	F(3,8)=8.984, p=0.006
Coordination	$FR \times LOD \times Order$	*F(6,16)=2.711, p=0.052 n.s.*
Percent of Time in FD	FR	F(2,22)=80.068 p=0.000
Percent of Time in FD	$Threshold$	F(1,11)=264.24 p=0.000
Percent of Time in FD	LOD	F(1,11)=42.963 p=0.000
Percent of Time in FD	$FR \times Threshold$	F(2,22)=7.192 p=0.004
Percent of Time in FD	$FR \times LOD$	F(2,22)=5.116 p=0.015
Percent of Time in FD	$Order$	F(3,8)=4.480, p=0.040

Table 6.6: ANOVAS for the second experiment. Factors are frame rate (FR), Level of detail (LOD) and Threshold level.

Dependent Variables The dependent variables are the same ones used in the first experiment: Task completion time, Accuracy, Coordination, Percentage of time in full detail and a Subjective evaluation.

6.2.3 Experiment Results

A repeated measures (within subjects) analysis of variance was performed on the results for Task completion time, Accuracy, Coordination and Percentage of time in full detail; significant results are summarized in Table 6.6.

1. *Task completion time.* A main effect of frame rate was observed. In general, completion times increased with delay, as can be seen in Figure 6.10. A main effect of level of detail suggested that higher LODs allow for better performance, at least in the context of our implementation (see Figure 6.11); this provides support for hypothesis H2. The interaction $FR \times LOD$ was significant, LOD differences were large at the lower frame rates and decreased at the highest frame

rate. The interaction $FR \times Threshold$ was marginally significant, threshold was significant at the lowest frame rate *(2Hz)*, indicating that lower thresholds are mostly desirable when delay is high (see Figure 6.12) and giving support to hypothesis H1.

While we found a trend towards a $Threshold \times LOD$ interaction (see Figure 6.13), it was not significant. However when delay is low it is possible to relate threshold level to LOD, figure 6.10 shows that at the higher frame rates high thresholds should be used for low LODs and low thresholds should be used with high LODs, giving some support to hypothesis H4. Finally, an interaction of $Threshold \times Order$ revealed that users who tried the *Threshold-5* first performed better than those who tried the *Threshold-25* first. This suggests that it is easier to learn the task when the threshold is set at a low value (the interaction is smooth) than when the threshold is set at a high value (the interaction is subject to delay).

2. *Accuracy.* Frame rate was a main factor for error in accuracy. Figure 6.14 reveals that, on average, errors in accuracy are higher with high delay and decrease in magnitude as delay decreases. Order was also a main factor, but there were no interactions between order and any other of the experimental factors. No other significant effects were found.

3. *Coordination.* The only main factor was LOD. In general, manipulation using the bounding box was more efficient than manipulation using the simplified teapot, providing support for hypothesis H2. Coordination using the Simplified-LOD and Threshold-25 was especially bad (see Figure 6.15). A $Threshold \times Order$ interaction was found, where users that tried the Threshold-5 first showed worse coordination under Threshold-5 than under Threshold-25. Similarly, if users tried Threshold-25 first they were not as coordinated in this condition as compared to the Threshold-5 condition. This suggests that efficiency in coordination is definitely subject to learning effects. The interaction $FR \times LOD \times Order$ was also significant. In this case, an analysis of ordering factors confirmed that users were more efficient at using the bounding box than the

simplified model. If users tried the bounding box first, they were more efficient overall, and frame rate was not a factor. If users tried the simplified model first, they were as equally as efficient as if they had the bounding boxes first, but when they used the bounding box, the efficiency decreased, especially at the higher frame rates.

4. *Percentage of Time in full detail.* Frame rate, threshold, and the interaction $FR \times Threshold$ were significant factors, due to the adaptive nature of the angular-disparity switch. Overall, users spent more time in full detail when the threshold was set at 25 degrees than when it was set at 5 degrees (because higher thresholds are harder to be reached), and they spent more time in full detail when the frame rate was high than when it was low (because the average angular disparity decreases when delay is low). The interaction of these variables resulted in more time in full detail at high frame rates for the high threshold, due to the combined effects of threshold and delay. LOD was also a significant main factor. In this case, users spent more time in full detail when they used the bounding box than when they used the simplified teapot; this can only be due to a change in the user's behaviour depending on the LOD offered.

   $FR \times LOD$ was the other significant interaction. As can be observed from Figure 6.16, when the simplified model was used, lowering frame rate from 4 to 2.2 Hz caused users to rely more frequently on the simplified model. On the other hand, when the bounding box was used, users spent roughly the same amount of time in full detail whether the frame rate was 4 or 2.2Hz. This suggests that the effects of improved LODs in the alternative representation are more significant as delay increases. Finally, *Order* was a significant main effect, but there were no significant interactions between order and the other factors.

5. *Subjective Evaluation.* Results of the subjective evaluation are presented in Figure 6.17. The two blocks that included the simplified LOD were on average better rated as the two blocks that included the bounding box. Also, the two blocks where threshold-5 was used were rated better than the two blocks where threshold-25 were used. The best condition was Simplified LOD with threshold-

5, rated exactly between "Acceptable" and "Good"; all other conditions were rated as "Acceptable" on average. Many users said they preferred the bounding box because it would allow them to see the axis of orientation. Some users reported that the simplified model was good during fine motion. There were no complaints about flickering (high frequency popping) effects during manipulation. The trends in these results give support to hypothesis H3.

Discussion

Our results gave some support to hypothesis H1, that lower thresholds are better than higher thresholds. Low thresholds were better at high levels of delay and when a high LOD alternative representation was used, otherwise the thresholds were roughly identical. The hypothesis H2, that higher LODs during switching allow for improved performance, was supported by results in task completion time and coordination, especially at the middle and low frame rates. Also, users were more willing to use the simplified model than the bounding box according the the measure of time spent in full detail.

Subjective Results showed trends of preference towards higher LOD in the alternative representation and lower thresholds, supporting hypothesis H3.

We did not find significant interactions between threshold and level of detail. One reason might be that the effects of threshold alone were not very significant. Evidence suggests that higher LODs allow for use of lower thresholds and partial improvements in performance. This gives some support to the first part of hypothesis H4.

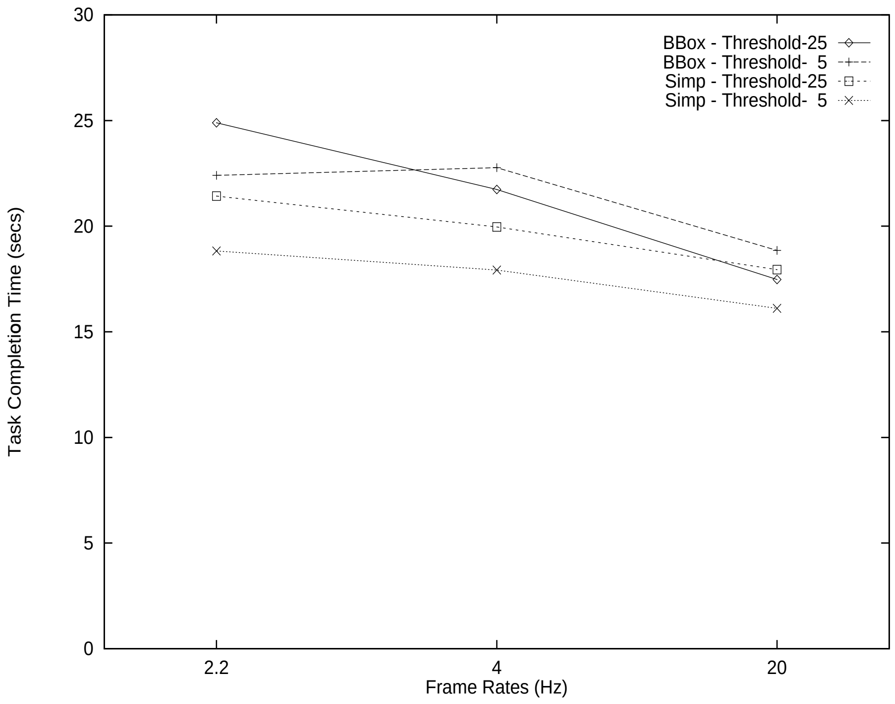

Figure 6.10: Task completion times for the second experiment.

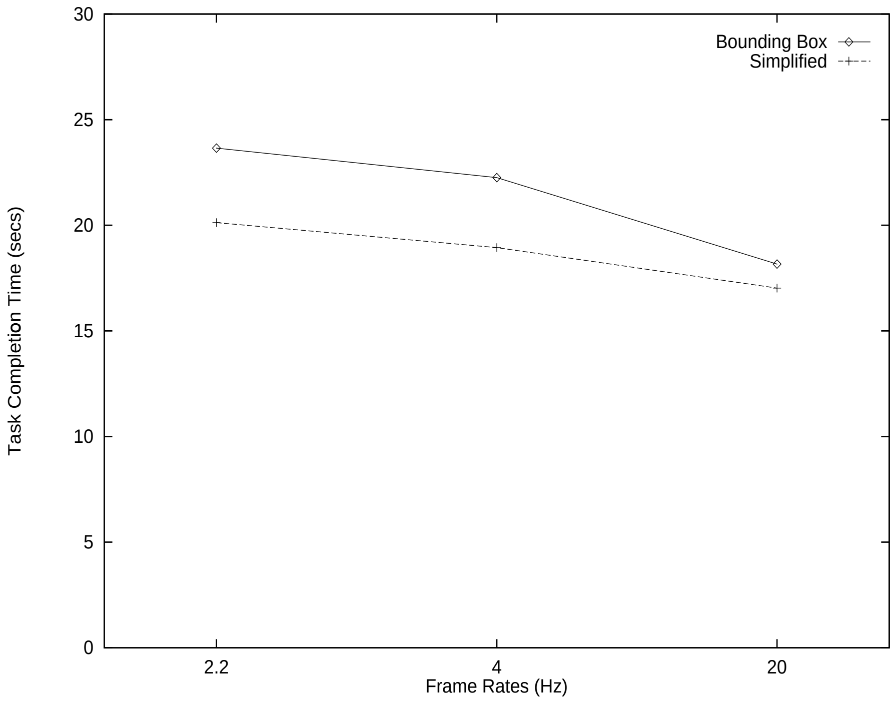

Figure 6.11: Task completion times grouped by level of detail.

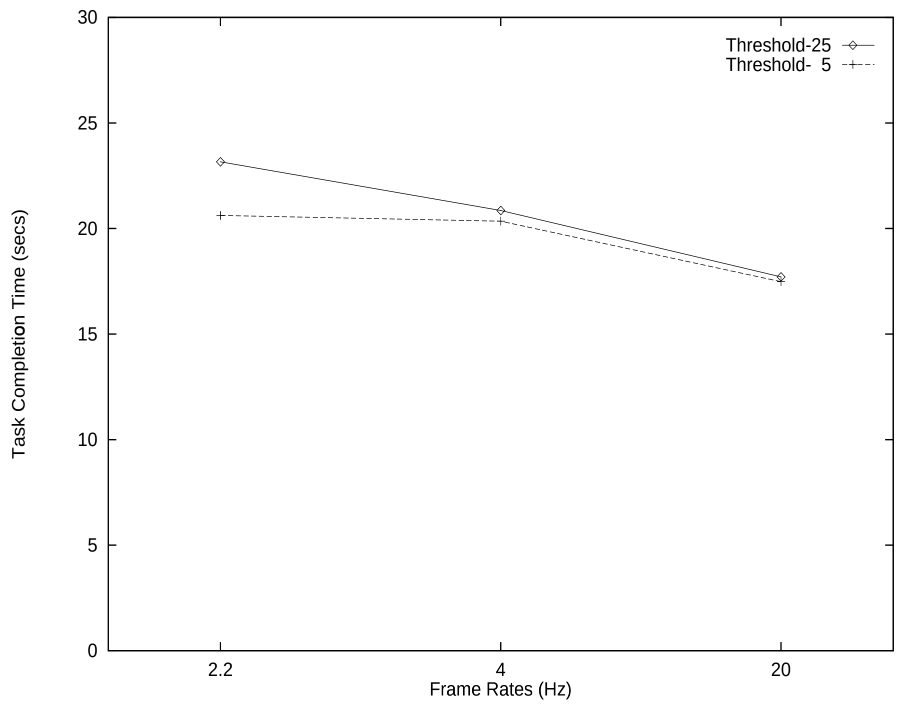

Figure 6.12: Task completion times grouped by threshold level.

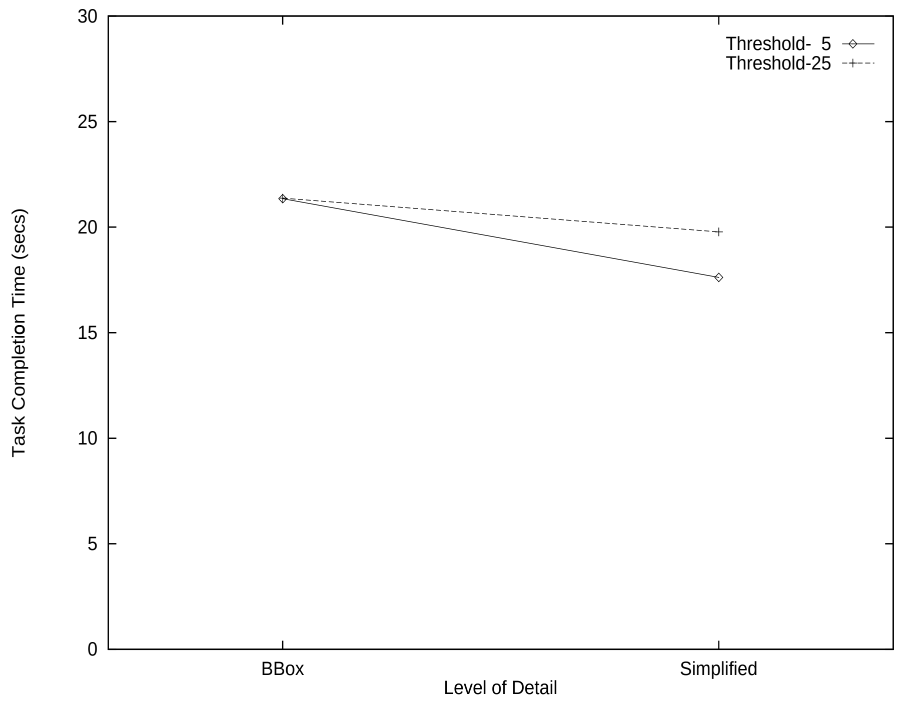

Figure 6.13: Task completion times grouped by threshold level for the two alternative LODs.

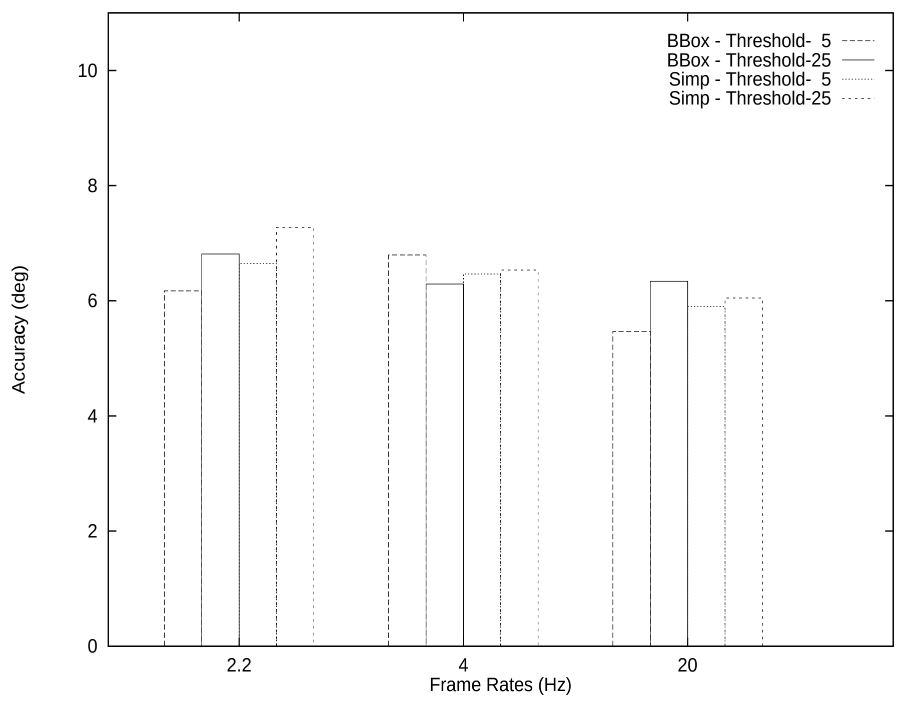

Figure 6.14: Comparison of accuracies across levels of frame rate.

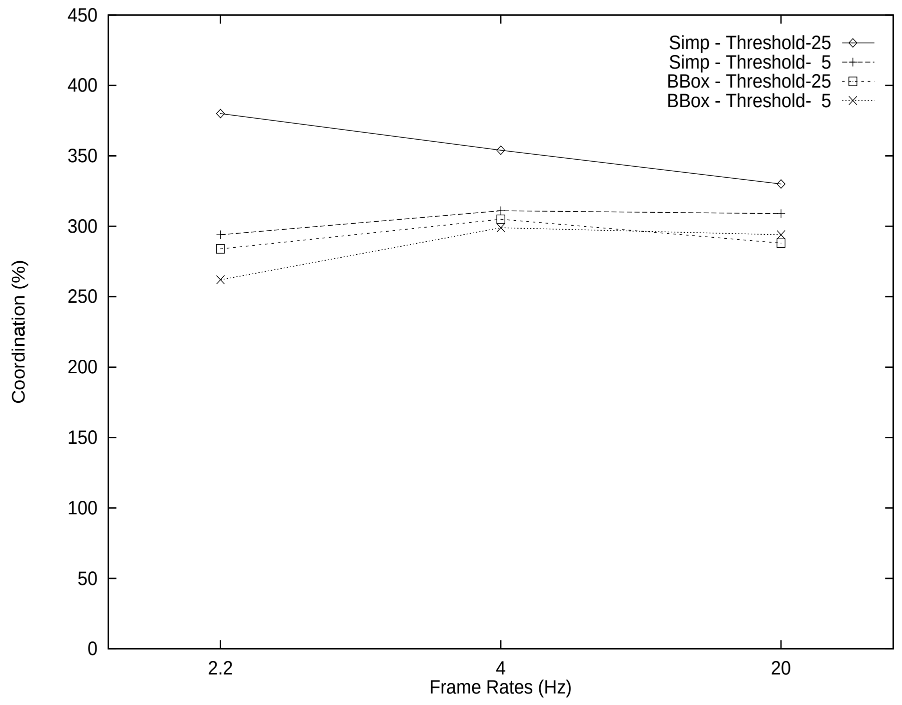

Figure 6.15: Measures of coordination for the second experiment.

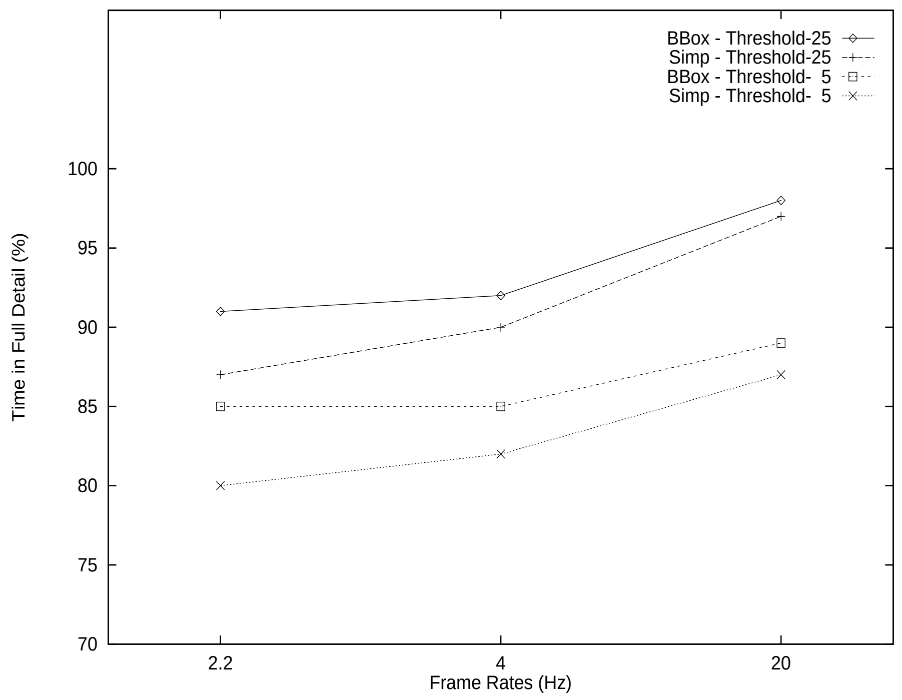

Figure 6.16: Percentage of time spent in full detail for the second experiment.

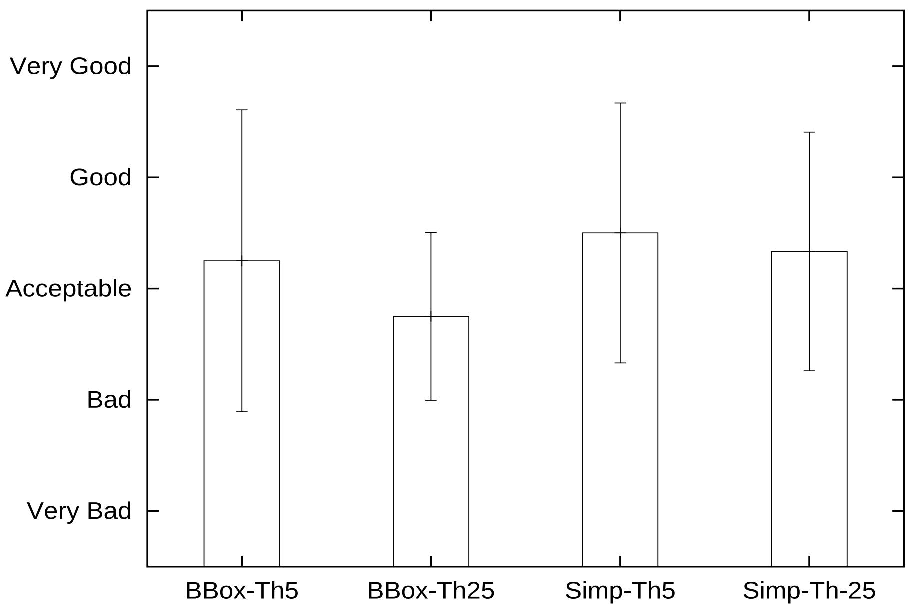

Figure 6.17: Results of the subjective evaluation in the second experiment.

Chapter 7

Conclusions and Future Work

7.1 LOD selection based on speed of interaction and control of spatio-temporal error

In this work we have described the sources of delay that are present during interaction with 3D graphics systems. One of the most effective ways by which system responsiveness can be enhanced is the reduction of the Level of Detail (LOD) of the images being rendered.

Level of detail *selection* techniques attempt to balance the trade-off between amount of visual detail and interactivity. Existing LOD selection techniques can be categorized in two sets, one that limits the loss of visual detail and another that limits delay in response time. The set of techniques that limits the loss of visual detail selects appropriate levels of detail regardless of the amount of time required to render the desired image. These techniques try to reduce detail that is dispensable, according to several criteria related to the user's viewpoint or the visual relevance of the object in terms of human perception. The second set of techniques emphasizes a limit in the delay of the system's response, regardless of the amount of visual detail required to appropriately render the scene.

In this thesis we have presented two LOD selection techniques that emphasize visual and temporal accuracy in an integrated fashion. The first technique, speed-based selection, emphasizes shorter response times over detail when the speed of interaction is high. Conversely, this technique emphasizes higher levels of detail over response time delay when the interaction speed is low. The disparity-based technique emphasizes limits on system response times when the difference between the displayed position and the position corresponding to subsequent input samples (the spatio-temporal error) is large. On the other hand, it emphasizes use of higher levels of detail when the spatio-temporal error is small.

One advantage of using disparity-based LOD selection is that it automatically adapts to conditions of delay. When delay is high, spatio-temporal error tends to increase. In these situations, a disparity-based threshold will be activated often, allowing for smooth interaction. When delay is low, spatio-temporal error tends to decrease; a disparity-based threshold will be less frequently activated, because the the threshold will hardly be reached.

The disparity based LOD selection also adapts to the user's behaviour in terms of the speed of interaction. When the rate of change in user input is high (high speed of interaction), the spatio-temporal error tends to increase. The disparity-based threshold will be activated whenever the rate of change in user input exceeds a limit (threshold) in spatio-temporal error, providing fast feedback that more accurately represents user input, at the expense of visual detail. When the speed of interaction is low, the spatio-temporal error will be low, and the disparity-based threshold will be hardly reached.

Both of the proposed techniques monitor user input during rendering. Selection of an alternative LOD can occur at any time during interaction. This allows for a tighter control of system response delay and enables better responsiveness than the existing LOD selection techniques.

7.1.1 Switching based on angular-speed and angular-disparity

Both selection techniques have been implemented for 3D rotation tasks. The angular-speed based switch is sensitive to the angular speed of the rotations indicated by the user. In this case, if the angular-speed is high, a low LOD representation is rendered, allowing for short response times. If the angular speed is low, a higher LOD representation is used. Note that in our implementation only two levels of detail are available during interaction.

The angular-disparity based switch is sensitive to the angle that is subtended between the position of the model currently being displayed and the position where the model should be according to the input samples received after rendering the displayed position. If the subtended angle exceeds a certain threshold, a lower LOD representation is used, thereby enhancing response times.

The angular-disparity based switch is sensitive to delay and speed of interaction, because it is based on control of spatio-temporal error.

7.1.2 Experimental analysis of the angular-disparity based switch

We performed two experiments to find out the effects on human performance of the angular-disparity based LOD selection switch.

In the first experiment, a comparison was performed between three switching choices tested under several conditions of delay. The second experiment was performed to examine the relationship of disparity based switching to the level of detail used in the alternative representation, and to characterize the effect of different threshold values on performance. The results from these experiments can be summarized as follows:

1. Angular-disparity based switching is the best LOD management strategy, when compared to never-switching or always-switching.

2. Angular-disparity based switching effectively adapts to conditions of delay. When delay is high, it is frequently activated and when delay is low, it is hardly activated.

3. Angular-disparity based switching effectively adapts to user behaviour and to some extent, users can control the triggering of the angular disparity switch.

4. Low threshold values for angular-disparity based switching are better than high threshold values, specially when delay is high.

5. High LODs in the alternative representation allow for better performance overall. Performance is improved to some extent when a higher LOD alternative representation is used in conjunction with a low threshold value.

7.2 Future Work

Several issues need to be further explored related to the use of these switching techniques, some of the challenging questions we have formulated so far are:

1. *Adaptive thresholding.* In this work, the angular-disparity switch depends on a threshold that remains constant during interaction. It has been shown that this configuration is somewhat adaptive to delay in system responsiveness. We would like to find out if a scale of thresholding levels could be used according to system responsiveness. Ideally, we would like to test a configuration where higher threshold values are used for the higher frame rates and lower threshold values are used for the lower frame rates. Results from the second experiment suggest that higher threshold values have the same effect than lower threshold values for systems with low delay, suggesting that adaptiveness should be emphasized when delay is high.

2. *Task dependent thresholding.* For the purposes of our task, it seems clear that lower threshold values are normally preferable to higher threshold values. Literature on delay suggests that the requirements for feedback are different depending on task difficulty. If we were to implement our selection techniques for different tasks, it would be necessary to determine what is the appropriate thresholding policy for each task type.

3. *Control of the switch from low detail back to full detail.* The implemented system switches back to full detail once the user is still. It might be appropriate to elaborate a mechanism to switch sooner to a full detail representation, based on predictions about the expected values of the disparity measure.

4. *Integration of our switching techniques.* The experimental work presented in this thesis referred only to the angular disparity switch. A system has been implemented, where the disparity-based switching technique is used to control change from a full detail to a low detail representation, while the angular-speed based switch is used to control change from low detail back to full detail. Unfortunately, time and experimental constraints made it impossible for us to

further study the effects of this configuration on performance. Before some experiments can be tried, it is necessary to find out how the two threshold values interact between each other, and what is the correspondence between both values; but we believe that this is probably the best configuration that could be obtained using both switching techniques.

5. *LOD selection for multiresolution representations.* Right now, the implementation of our switching technique is just bimodal. This means that during interaction we have just two levels of detail to choose from. An important question is how to combine different threshold levels in the system to allow use of multiple levels of detail at run time. This question is a challenging one, since appropriate selection of alternative levels of detail should take into account not only the angular disparity measure, but also the progress of the image being rendered at the time of selection.

6. *Design of a formal framework to balance visual and spatio-temporal error.* There is a trade-off between visual error and spatio-temporal error introduced when using different LODs in conditions of delay. When a full detail representation is used, the amount of visual error in the image is null, but the amount of spatio-temporal error is completely subject to delay. When the spatio-temporal error is controlled, this error becomes almost null, but the visual error introduced between the alternative and the full levels of detail is unaccounted for. Currently, we empirically assess whether a given threshold balances visual error and spatio-temporal error in a way that effectively allows improved performance. There is a need to develop a formal model to describe how to balance visual and spatio-temporal error. Such a model would help us determine the appropriate thresholding values for different levels of detail and ultimately allow for the best human performance attainable under certain conditions of delay. A formal framework of this type would help us in the assessment of effectiveness of other LOD selection techniques and do comparisons across techniques.

7. *Integration of LOD selection techniques.* The amount of existing literature in LOD selection techniques is significant. It is necessary to envision ways to

integrate our concepts on LOD selection with the rest of the literature. This is also challenging, since our approach focuses both on preservation of visual detail and on limiting spatio-temporal detail, while other techniques focus only on one of these aspects.

8. *Disparity based LOD selection in 3D Euclidean space.* A natural extension of our work includes the implementation of the speed-based and the disparity based LOD selection techniques for movement in 3D space. This poses some challenging questions, since motion in 3D is not restricted to a limited space, and depends on the scale of the objects. Related work in LOD selection based on speed, mentioned in Chapter 3, shows some current approaches related to this purpose. Another question is that in a 3D scene, objects may have different rates of motion depending on the distance to the viewpoint or on the speed of the objects itself, suggesting that update rates of different objects on the scene should also be different, according to the disparity introduced by each object on the display. The problem of presenting different update rates for different objects in a 3D graphics system is a challenging one for 3D graphics systems.

It should be possible to envision the implementation of these concepts for a wider variety of tasks. We deem that real-time applications should be among the first to obtain benefits from the concepts presented in this thesis.

Bibliography

[Alia99] Daniel Aliaga, J. Cohen, A. Wilson, E. Baker, H. Zhang, Erikson C., K. Hoff, T. Hudson, W. Stuerzlinger, R. Bastos, M. Whitton, F. Brooks, and D. Manocha. "MMR: An Interactive Massive Model Rendering System Using Geometric and Image-Based Acceleration". *Proceedings of the 1999 Symposyium on Interactive 3D Graphics*, pp. 199–207, 1999.

[Azum94] Ronald Azuma and Gary Bishop. "Improving Static and Dynamic Registration in an Optical See-Through HMD". *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pp. 197–204, July 1994.

[Barf95] W. Barfield and C. Hendrix. "The Effect of Update Rate on the Sense of Presence within Virtual Environments". *Virtual Reality: Research, Development and Application*, Vol. 1, No. 1, pp. 3–16, 1995.

[Brys93] Steve Bryson. "Effects of lag and frame rate on various tracking tasks". http://science.nas.nasa.gov/ bryson/papers.html, 1993.

[Chen88] Michael Chen, S. Joy Mountford, and Abigail Sellen. "A Study in Interactive 3-D Rotation Using 2-D Control Devices". *Proceedings of ACM SIGGRAPH '88 Conference on Computer Graphics*, Vol. 22, pp. 121–129, August 1988.

[Dins89] I. Dinstein, M.G. Kim, J. Tselgov, and A. Henik. "Compression of Stereo Images and The Evaluation of Its Effects on 3-D Perception". *SPIE Applications of Digital Image Processing XII*, 1989.

[El-s97] Jihad El-Sana and Amitabh Varshney. "Controlled Simplification of Genus for Polygonal Models". *Proceedings IEEE Visualization'97*, pp. 403–412, 1997.

[Elli99] S. Ellis, B.D. Adelstein, S. Baumeler, G.J. Jense, and R.H. Jacobi. "Sensor Spatial Distortion, Visual Latency, and Update Rate Effects on 3D Tracking in virtual Environments". *Proceedings of the Virtual Reality '99 Conference*, pp. 218–221, 1999.

[Elme92] D. Elmes, B. Kantowitz, and H. Roediger III. *Research Methods in Psychology*, chapter 6, pp. 121–140. West Publishing Company, 1992.

[Fitt54] Paul M. Fitts. "The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement". *Journal of Experimental Psychology*, Vol. 47, No. 6, pp. 381–390, June 1954.

[Fole93] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics: Principles and Practice*, pp. 376–381. Addison-Wesley Systems Programming Series, Don Mills, Ontario, 1993.

[Fris76] J. Frisby and J. Mayhew. "Rivalrous texture Stereograms". *Nature*, Vol. 264, pp. 53–56, 1976.

[Funk93] Thomas A. Funkhouser and Carlo H. Séquin. "Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments". *SIGGRAPH 97 Conference Proceedings*, Vol. 27 of *Annual Conference Series*, pp. 247–254, August 1993.

[Garl97] Michael Garland and Paul S. Heckbert. "Surface Simplification Using Quadric Error Metrics". *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pp. 209–216, August 1997.

[Hans92] Andrew J. Hanson. "The Rolling Ball". *Graphics Gems III*, pp. 51–60. Academic Press, Boston, 1992.

[Heck97] Paul S. Heckbert and Michael Garland. "Survey of Polygonal Surface Simplification Algorithms". Technical report, CS Dept., Carnegie Mellon U., URL: http://www.cs.cmu.edu/ ph, 1997.

[Hinc97] Ken Hinckley, Joe Tullio, Randy Pausch, Dennis Proffitt, and Neal Kassell. "Usability Analysis of 3D Rotation Techniques". *Proceedings of the ACM Symposium on User Interface Software and Technology*, 3D Interaction Techniques, pp. 1–10, 1997.

[Hopp96] Hugues Hoppe. "Progressive Meshes". *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pp. 99–108, August 1996.

[Hopp98] Hugues Hoppe. "Smooth View-Dependent Level-Of-Detail Control and its Application to Terrain Rendering". *Proceedings IEEE Visualization'98*, pp. 35–42, 1998.

[Joli88] Pierre Jolicoeur. "Mental Rotation and the Identification of Disoriented Objects". *Canadian Journal of Psychology*, Vol. 4, No. 42, pp. 461–478, 1988.

[Lian91] Jiandong Liang, Chris Shaw, and Mark Green. "On Temporal-Spatial Realism in the Virtual Reality Environment". *Proceedings of the ACM Symposium on User Interface Software and Technology*, Virtual Workspaces, pp. 19–25, 1991.

[Lind98] Peter Lindstrom. "A Survey of Multiresolution Techniques for Arbitrary Polygonal Meshes". Technical report, Georgia Institute of Technology, URL:http://www.cc.gatech.edu/gvu/people/peter.lindstrom/, 1998.

[Lueb97] David Luebke and Carl Erikson. "View-Dependent Simplification of Arbitrary Polygonal Environments". *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pp. 199–208, August 1997.

[Mack93] I. Scott MacKenzie and Colin Ware. "Lag as a Determinant of Human Performance in Interactive Systems". *Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems*, pp. 488–493, 1993.

[Mine95] M. Mine. "Characterization of End-to-End Delays in Head-Mounted Display Systems.". *SIGGRAPH'95 Course*, No. 8, 1995.

[Ohsh96] Toshikazu Ohshima, Hiroyuki Yamamoto, and Hideyuki Tamura. "Gaze-Directed Adaptive Rendering for Interacting with Virtual Space". *Proceedings of IEEE VRAIS'96*, pp. 103–110, 1996.

[Pars87] Lawrence M. Parsons. "Visual discrimination of abstract mirror-reflected three-dimensional objects at many orientations". *Perception & Psychophysics*, Vol. 42, No. 1, pp. 49–59, 1987.

[Pupp97] Enrico Puppo and Roberto Scopigno. "Simplification, LOD and Multiresolution - Principles and Applications". Technical report, Eurographics '97 Tutotial Notes, 1997.

[Redd97] Martin Reddy. *Perceptually Modulated Level of Detail for Virtual Environments*. Ph.D. thesis, The University of Edinburgh, Department of Computing Science, apr 1997.

[Ross93] Jarek Rossignac and Paul Borrel. "Multi-resolution 3D approximations for rendering complex scenes". *Modeling in Computer Graphics: Methods and Applications*, pp. 455–465, 1993.

[Scha83] Bruce J. Schachter. *Computer Image Generation*, chapter 4, pp. 47–124. John Wiley & Sons, 1983.

[Seku94] Robert Sekuler and Randolf Blake. *Perception*, chapter 8, pp. 250–291. McGraw-Hill, Inc., Toronto, Ontario, 1994.

[Shad96] Jonathan Shade, Dani Lischinski, David Salesin, Tony DeRose, and John Snyder. "Hierarchical Image Caching for Accelerated Walkthroughs of Complex Environments". *ACM SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pp. 75–82, August 1996.

[Shep71] Roger. Shepard and Jacqueline Metzler. "Mental Rotation of Three-Dimensional Objects". *Science*, Vol. 171, No. 3972, pp. 701–703, February 1971.

[Shin99] Kyoung Shin Park and R. V. Kenyon. "Effects of Network Characteristics on Human Performance in a Collaborative Virtual Environment". *Proceedings of the Virtual Reality '99 Conference*, pp. 104–111, 1999.

[Shoe94] Ken Shoemake. "Arcball Rotation Control". *Graphics Gems IV*, pp. 175–192. Academic Press, Boston, 1994.

[Smet95] Gerda J. F. Smets and Kees J. Overbeeke. "Trade Off Between Resolution and Interactivity in Spatial Task Performance". *IEEE Computer Graphics and Applications*, Vol. 15, No. 5, pp. 46–51, September 1995.

[Thar92] G. Tharp, A. Liu, L. French, S. Lai, and L. Stark. "Timing Considerations of Helmet Mounted Display Performance". *SPIE Conference on Human Vision, Visual Processing, and Digital Display*, 1992.

[Turk92] Greg Turk. "Re-tiling polygonal surfaces". *Computer Graphics*, Vol. 26, No. 2, pp. 55–64, July 1992.

[Wang98] Yanqing Wang, Christine L. MacKenzie, Valerie A. Summers, and Kellogg S. Booth. "The Structure of Object Transportation and Orientation in Human-Computer Interaction". *ACM CHI 98 Conference on Human Factors in Computing Systems*, pp. 312–319, 1998.

[Ware94] Colin Ware and Ravin Balakrishnan. "Reaching for Objects in VR Displays: Lag and Frame Rate". *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 4, pp. 331–356, 1994.

[Ware98] Colin Ware and Jeff Rose. "Real Handles, Virtual Images". *ACM CHI 98 Conference on Human Factors in Computing Systems (Summary)*, Late Breaking Results: See How You Feel: New Input Techniques and Modalities, pp. 235–236, 1998.

[Wats97a] B. Watson. *Guidelines for Level of Detail Management.* Ph.D. thesis, Georgia Institute of Technology, College of Computing Science, 1997.

[Wats97b] B. Watson. "On Temporal Level of Detail: System Responsiveness, Feedback and User Performance". Unpublished draft, URL: http://www.cs.ualberta.ca/ watsonb/, 1997.

[Wats98] B. Watson, N. Walker, W. Ribarski, and V. Spaulding. "Effects of Variation in System Responsiveness on User Performance in Virtual Environments". *Human Factors*, Vol. 40, No. 3, pp. 403–414, 1998.

[Wlok95] M. Wloka. "Lag in Multiprocessor Virtual Reality". *Presence*, Vol. 4, No. 1, pp. 50–63, 1995.

[Xia97] Julie C. Xia, Jihad El-Sana, and Amitabh Varshney. "Adaptive Real-Time Level-of-Detail-Based Rendering for Polygonal Models". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 3, No. 2, pp. 171–183, April 1997.

[Zhai96] Shumin Zhai, Paul Milgram, and William Buxton. "The Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input". *ACM CHI 96 Conference on Human Factors in Computing Systems : Common Ground*, pp. 308–315, April 13–18 1996.

[Zhai98] Shumin Zhai and Paul Milgram. "Quantifying Coordination in Multiple DOF Movement and its Application to Evaluating 6 DOF Input Devices". *Proceedings of ACM CHI 98 Conference on Human Factors in Computing Systems*, pp. 320–327, 1998.