Eye Tracker Input in First Person Shooter Games

Poika Isokoski
TAUCHI
Department of Computer Sciences
University of Tampere
FIN-33014 University of Tampere, Finland
poika@cs.uta.fi

Benoît Martin
LITA
University Paul Verlaine - Metz
Ile du Saulcy
57045 METZ CEDEX 1, France
benoit.martin@univ-metz.fr

Keywords
Eye tracker, game, input device, first person shooter, aiming

Introduction

We report ongoing work on using an eye tracker as an input device in first person shooter (FPS) games. In these games player moves in a three-dimensional virtual world that is rendered from the player's point of view. The player interacts with the objects he or she encounters mainly by shooting at them. Typical game storylines reward killing and punish other forms of interaction.

The reported work is a part of an effort to evaluate a range of input devices in this context. Our results on the other devices in the same game allow us to compare the efficiency of eye trackers as game controllers against more conventional devices. Our goal regarding eye trackers is to see whether they can help players perform better. Some FPS games are played competitively over the Internet. If using an eye tracker gives an edge in competitive play, players may want to acquire eye tracking equipment. Eye trackers as input devices in FPS games have been investigated before (Jönsson, 2005), but that investigation focused on user impressions rather than on the efficiency and effectiveness of eye trackers in this domain. However, Jönsson's results on eye tracker efficiency in a non-FPS game were encouraging.

The Game

Rather than using an existing game engine for our experiment, we contracted a student group to build a new one. A new game was necessary, because we wanted the source code to be very simple and compact so that it would be easy to modify. It was possible to avoid most of the complicated code because we did not need network or mulpiplayer capability, artificial intelligence for game creatures, or special techniques to speed up the graphics rendering. For input device experiments, unintelligent targets are better because they make the experimental situation more controlled. In experiments we can use powerful hardware instead of clever coding to keep the frame rate high enough.

The game world consisted of a square area covered by randomly generated hills and valleys with randomly placed trees and tufts of grass. The randomness helps to avoid map-specific bias in the results.

A screenshot from the game is shown in Figure 1. The task of the player was to move in the world and shoot as many targets as possible. The targets were round plaques with a portrait of a penguin on them. The targets moved slowly along the terrain to make hitting them at least moderately difficult. In our experiments the targets did not shoot back. Our purpose was to focus on the efficiency of moving and aiming. The efficiency of aiming while evading enemy projectiles was left for further work. Whenever a target was hit, it disappeared, and another was generated at a random location in the world.

Figure 1. A scene from the penguin hunting game.

When planning the use of an eye tracker in a game like this, one has to find a way that does not interfere with the use of gaze for acquiring information about the world. This excludes the use of gaze as a simple pointer replacement for controlling the player's movement on the map. It may also be important to be able to move to all directons while freely observing the scene. Another possibility for eye-tracker use is to contol the direction where the player is facing. However, we found that the need to do swift 180 and even 360 degree turns is frequent because one needs to survey the environment to find targets. We did not find a natural way of doing this with the gaze. Also, a direct mapping of the gaze position to camera orientation would lead to either a jittery display or slow movements due to averaging in order to avoid the jitter (Jönsson, 2005).

Finally we decided to use the gaze for aiming the weapon within the scene shown on the display. The mouse in the right hand was used for controlling the camera angle, and the left hand operated the arrow keys on the keyboard for moving the player around in the world. The white crosshairs in the center of the display showed where the player was facing and acted as the only aiming device when the eye tracker was not used. When the eye tracker was used, the red reticle (pointed at by the white arrow for the benefit of grayscale printout readers) showed where the player was looking. Shooting to the position of the white crosshairs was possible by pressing the left mouse button, and shooting to the position of the gaze-controlled red reticle was possible by pressing the right mouse button.

The advantage that we envisioned this setup might have over the conventional keyboard and mouse setup was that aiming with the gaze should be faster than aiming with the mouse. We thought that this would be a significant advantage in situations where the player reaches a top of a hill or steps out from behind a tree so that several targets are revealed. It should be possible to shoot the targets rapidly with the combination of gazing and pressing the mouse button.

Note that performance in this kind of scenario is very important in FPS games. The player is often thrown into rooms full of unfriendly creatures and the only way to survive is to aim and kill fast in order to survive.

The disadvantage of aiming with the gaze is that aiming over long distances is difficult because of the accuracy issues with eye trackers. Because there are advantages and disadvantages in the use of eye trackers in FPS games, and theoretical answers to player performance were hard to come by, we decided to approach the problem empirically.

Results

Our experimentation is in early stages. So far we have data for only one of the authors playing 10 five-minute sessions using a Tobii 1750 eye tracker. The results are shown in Figure 2. For comparison we show the last 10 sessions of a 30-session trial completed by the same player with other input devices. These devices are keyboard and mouse without the eye tracker, and the Xbox 360 controller. With the Xbox 360 controller the left stick was used for moving, and the right stick for aiming. The shoulder buttons were used to shoot.

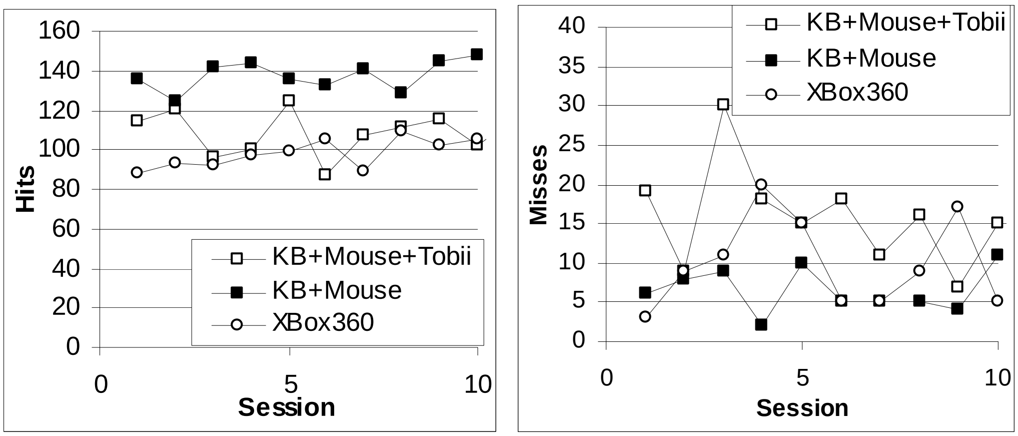

Figure 2. The number of hits and misses per session for three input device configurations for one player.

It appears that using the eye tracker does not improve performance for this player in comparison to the keyboard+mouse condition. However, the keyboard+mouse+tobii condition performs on the same level with the Xbox 360 controller. We should notice that the player had more training with the conditions without the eye tracker. Therefore, with continued training the performance with the eye tracker might improve more than the performance with the other devices. While fairly flat, the data in Figure 2 do not prove that no learning happens. Games are often played for longer than the 50-minute period shown in Figure 2. Even slow improvement adds up in the course of hundreds of hours of training.

Conclusions and Further Work

While it appears that adding eye-tracker support to FPS games will not always improve player performance, we find our preliminary results promising. For example, it may be that eye-tracker input can improve performance with input device configurations other than the keyboard+mouse combination. By the time of the conference, we hope to have more data to report on combining eye tracker with other input devices. It may also be possible to design an eye tracker based input device configuration that allows disabled users without the manual dexterity to aim with a mouse or a gamepad have a satisfying gaming experience.

Acknowledgments

The game was developed by Laurent Gomilla and Maurice Svay, who we thank for their excellent work.

References

Jönsson, E. (2005) *If Looks Could Kill – An evaluation of Eye Tracking in Computer Games*. Master's Thesis, Royal Institute of Technology, Stockholm, Sweden.