Multiplayer First Person Shooting Game with Tangible and Physical Interaction

Zhou Zhiying
National University of Singapore
Department of Electrical and Computer Engineering
Interactive Digital Media Lab
elezzy@nus.edu.sg

Jefry Tedjokusumo
National University of Singapore
Department of Electrical and Computer Engineering
Interactive Digital Media Lab
elejt@nus.edu.sg

Stefan Winkler
National University of Singapore
Department of Electrical and Computer Engineering
Interactive Digital Media Lab
elews@nus.edu.sg

ABSTRACT

In this paper, we present a new application of the first person shooting (FPS) game. Currently the multiplayer FPS games in the market use keyboard and mouse to interact with the virtual world. This method is not intuitive and lacks physical interaction between the players. Our system provides an intuitive way to interacts with the virtual world, the user moves and aims freely as they are in the real world. This encourages more physical interaction between the players as they are competing or collaborating with each other.

Categories and Subject Descriptors

H.4 Information Systems Application: Miscellaneous; H.5.1 Multimedia Information Systems: Artificial, augmented, and virtual realities; H.5.2 User Interfaces: Input devices and strategies

General Terms

Design, Human Factors.

Keywords

First Person Shooting, Virtual Reality, Augmented Reality, Physical Interaction, Tangible User Interface.

1. INTRODUCTION

First Person Shooting game is a popular computer game genre. The game required high level accuracy of aiming, which currently is provided by mouse [1]. This traditional method lacks physical and social interaction.

In this paper we present a new application of multiplayer FPS game. In our system, the players wear head tracking, Head Mounted Display (HMD), and wand tracking (see Figure 1). With these tracking devices we change the way multiplayer FPS games are played. The player's view orientation and position tracked by the head tracker. His hand holds a wand tracking device. This wand's orientation corresponds to the gun aiming orientation in the game. In our system it is possible for the player to shoot backward, even though his head facing forward. To make this feasible, we give a small screen showing the gun point of view in the player's eye view (see Figure 3).

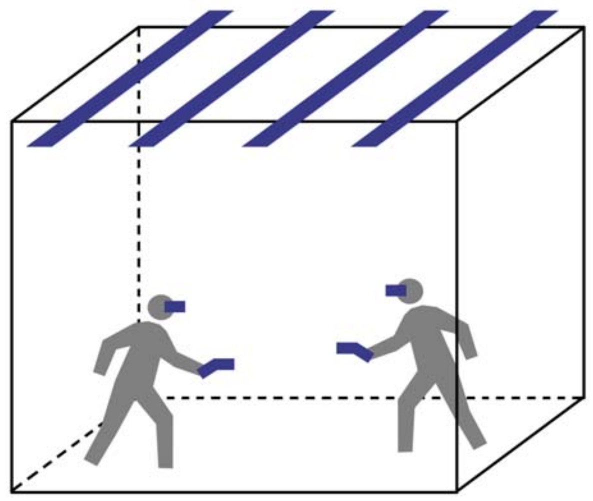


Figure 1. The Game Environment.

The game is situated in a rectangular room with no obstacle in it (see Figure 1). We place game items (health, armor, weapon, bullets) in the corners of the room. The player must go to this location to get these items. The game style is "death match", which means the players shoot to each other until his opponent's health reach zero. The winner is the one who kills the most. The game style can be easily changed to support more players so it has collaboration component i.e. the battle between two groups. However this is subject to the availability of trackers.

Our system gives new perspective of how multiplayer FPS games are played. It provides tangible and physical interactions among users both collaboratively and competitively.

In section 2 we introduce previous work on FPS games. Section 3 describes our system set up. Section 4 discusses the implementation of our system. Finally in section 5 we conclude our paper, and discuss some of the possible future work.

2. RELATED WORK

CAVE QUAKE [3] was built by Paul Radjlich in 1997. It had done a very good job in delivering tangible user interface for FPS. CAVE is a 10x10x10 foot "cube" with images projected onto 3 walls and the floor. The player stands in the middle of

Table 1. Our System compared to others

System Name	Weapon Aiming Method	Virtual Character's Movement	Accuracy	Ability to Shoot Outside the View Range	Physical Interaction between Players
CAVEQUAKE	Hand	Joystick	High	None	None
CAVEUT	Hand	Joystick	High	None	None
ARQUAKE	Head	Player's Movement	Low	None	None
ChairIO + Gun	Hand	Tilting the chair	High	None	None
Game Runner	Handlebar	Treadmill	High	None	None
Human Pacman	Not Applicable	Player's Movement	Low	Not Applicable	Yes
Touch Space	Hand	Player's Movement	High	None	Yes
Our System	***Hand***	***Player's Movement***	***High***	***Yes***	***Yes***

the cube and the CAVE renders the virtual world in front, left, right, and bottom of the player. The player aims the gun using his hand as it is in the real world. The player usually uses joystick to move around in the virtual world. CaveUT [5] uses the same idea. The difference is in the system cost. CaveUT is built using low-priced hardware equipments (around US$ 25.000) compared to CAVE Quake which was built using million dollars equipments. Our system differs in the way that the graphics are presented. Instead of using projection on walls that doesn't allow update of viewpoints when user moves his head, we attach a camera to the HMD so that the user can always receive updated first-person views whenever he moves.

ARQUAKE [2] is a single player mixed reality FPS. The player in ARQUAKE has freedom to move around the world, but the gun aiming is limited to the center of the view of the Head Mounted Display (HMD), this means the gun aiming is done by player's head rather than player's hand. This is unintuitive and it is difficult to aim accurately using our head while we are evading our enemies. The game was initially designed as a single player game, so it has no interactivity with other human player. Though this game can be extended to support multi players, the tracking systems that they use (GPS and markers) are meant for outdoor application. Therefore, ARQUAKE may have accuracy problems to determine whether a bullet shot from one player has actually hit another player. Human Pacman [7] is another example of outdoor augmented reality game. However, as it is also using GPS for tracking, it suffers the same accuracy problem as ARQUAKE. Our system (IS900) is an indoor system with accuracy of position 2mm - 3mm, orientation 0.25^o - 0.4^o – which is sufficient for FPS games.

Touch-Space [8] is an indoor mixed reality game that is situated and carried in a room-size space, using high accuracy ultrasonic tracking system from intersense which is similar to ours. It has 3 stages: physical land exploration, virtual land exploration, and virtual castle exploration. The stages ranged from augmented reality to virtual reality. Our system differs from Touch Space in the following aspects. First, we add in more physical interaction components, for example, collecting armors by moving close to virtual armor boxes, jumping up to avoid bullets. Second, we implement a popular FPS game which involves intensive competitions between users, while the tasks in Touch-Space require collaborations.

In 2005 Beckhaus, et al. [1] propose a new device to control a FPS game. The device consists of a ChairIO and a Game Gun. The ChairIO basically is a chair that tilts. The tilt is used to control the movement of the virtual character in the game (forward, backward, left, and right). The ChairIO also support jumping (by bouncing from the chair). However, the user is constrained to be sitting on the chair hence physical movements are very limited. Our system offers free body movements by using wireless trackers. User can move freely within the room and even jump to avoid bullets. In Beckhaus's work, the gun aiming is done by the Game Gun which is also limited to the center of the screen; this means the game view will always follow the gun's orientation. Similar game controller concept is used in GameRunner [4]. However, in physical life, user's viewpoint is absolutely not always following the gun post. Our solution of separating the views offers a new approach of developing FPS games.

Table 1 summarizes the comparison between our system and the others. Our system gives intuitive and tangible controller in the virtual world and at the same time maintains collaborative/competitive physical interaction between the players. Our system novelty is on the ability to aim and shoot other players outside the virtual character's eye view.

3. SYSTEM

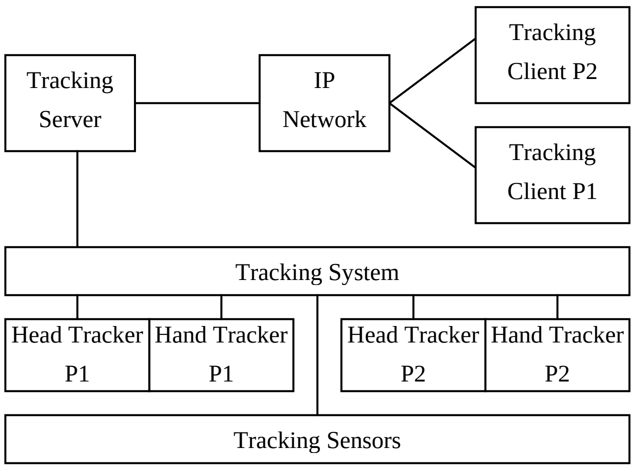
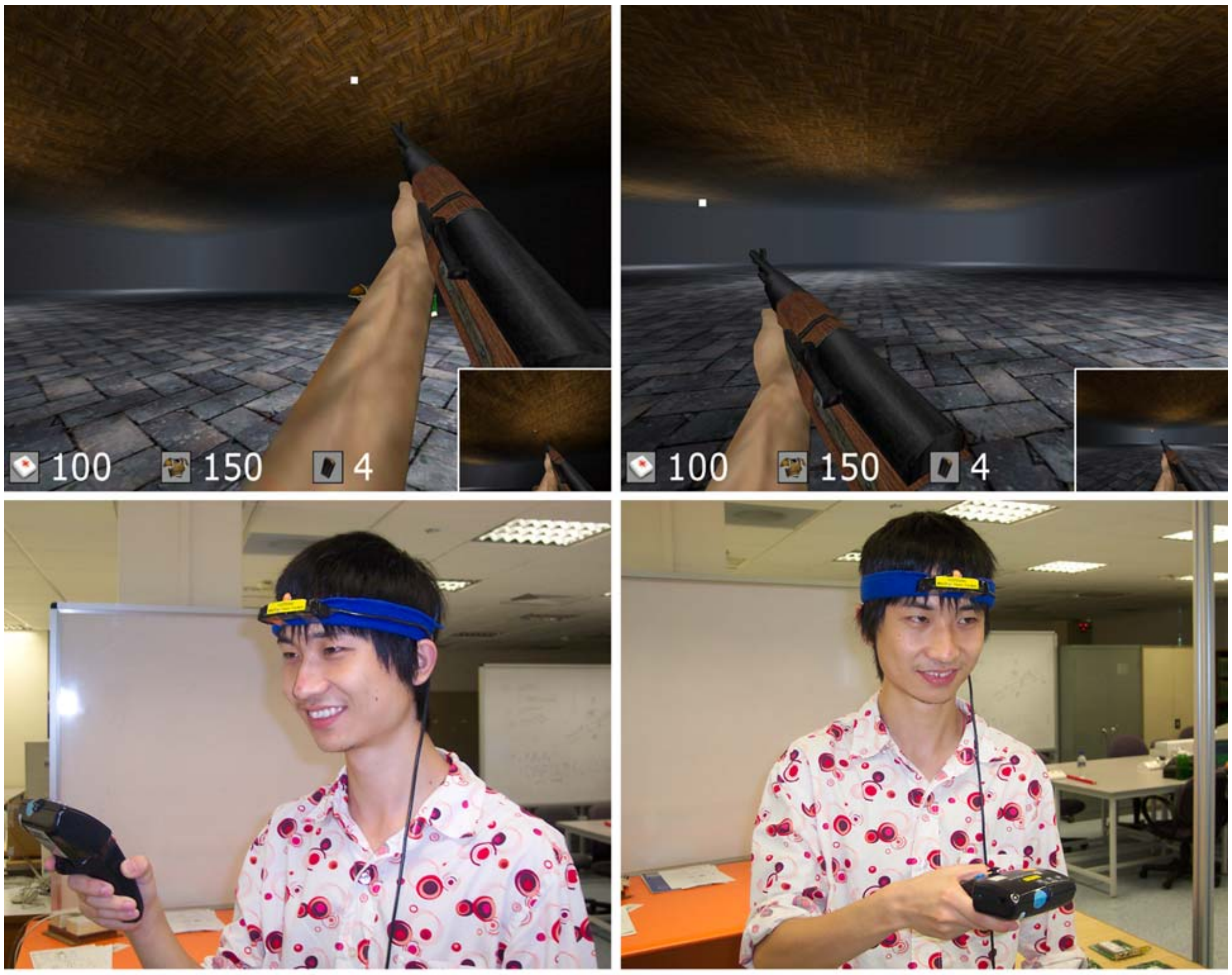


Figure 2. The Tracking System Diagram.



(a) (b)

Figure 3. Game's screenshots and the player

Our game system consists of game engine and a tracking system (InterSense IS900). We implement the system with 2 players (four trackers), as each player needs 2 tracking devices (1 head tracker, and 1 wand tracker). IS900 is an ultrasonic tracking system. It has high degree of accuracy (2mm - 3mm for position and 0.25^o - 0.4^o for orientation), but it has limited coverage. We can easily extend the coverage by putting more sensors on the ceiling. The tracking system is built in a room of 2 meter width, 4 meter length, and 2.4 meter height (see Figure 1). The tracking system is connected to a computer, we call it tracking server. The tracking server will send UDP packets which contain tracking data (position and orientation) to the tracking clients at rate 90 Hz (see Figure 2).

The head tracker is mounted together with the Head Mounted Display (HMD) to track the player's position and orientation. The game engine will render the virtual environment based on this information. The wand tracker acts as a gun. The wand tracker is used by the player to aim the weapon in game. The player's HMD will display the image like that is shown in Figure 3. The small window in the bottom right is the gun radar that represents the gun's point of view. With this gun radar, the player can aim and shoot to any direction inside or outside his eye view. This adds an element of excitement. The player can aim and shoot the enemy at your back. The white square dot in Figure 3 is the weapon's pointer. The weapon's pointer is very useful to help the player to aim accurately.

For the game engine, we use cube engine [6]. This game engine is an open source program written in C++, so we can easily modify it. The engine also has a built in world editor and it supports .md2 models. The game engine itself consists of two parts. One is the server and the other is the client, the game engine supports multiple clients. In our implementation we will use one server and two clients. Each client will do the rendering and sends the result into the player's HMD. The server only receives the information of the client's position, orientation, and action. Then the server broadcasts the information to all other clients (see Figure 4).

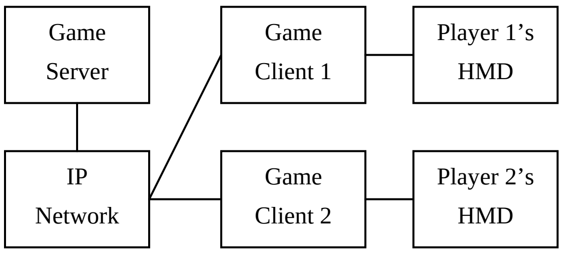


Figure 4. The Game System Diagram.

4. IMPLEMENTATION

This section will discuss major issues on the implementation. We have made some modification in the game engine to support our main features

4.1 Calibration

The first step to integrate the tracking system and the game engine is calibration. Figure 5 (a) shows the orientation of the tracking system (where the players are moving in real world), Figure 5 (b) shows the axis orientation of the wand tracker, and Figure 5 (c) shows the axis orientation of the game engine (where the virtual character moves in virtual world). The game engine defines yaw as rotation around z axis, pitch as rotation around y axis, and roll as rotation around x axis.

We calibrate the hand tracker to give yaw, pitch, and roll value of 0 to the game engine when the hand's and the tracking system's axis align as in Figure 5 (a) and 5 (b). The z axis in the real world is the inverse to z axis in the virtual world (see Figure 5 (a) and 5 (c)). This makes the yaw value that we track in the real world is the inverse of the yaw value in the virtual world (i.e. different in the ± sign).

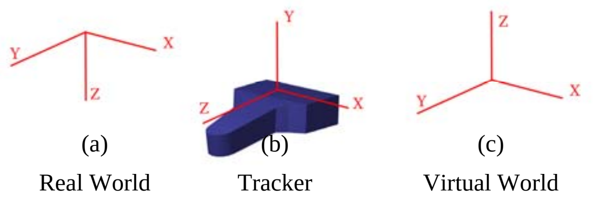


Figure 5. Axis Orientation.

Calibrating the position is simple. Because the x axis and y axis of the real world and the virtual world are aligned. We only need to adjust the scaling.

4.2 Modifying Weapon's Orientation

Like most of the game engine, cube engine weapon's orientation is pointing to the center of the screen and it is tied to the player's eye view. To make the weapon's orientation independent of the player's eye view, we add new variables: yaw, pitch, and roll for the weapon. We render the weapon's orientation according to the data that we get from the wand tracker. In Figure 3 (a) the player orients the wand tracker upward, and you can see in the player's HMD, the gun is pointing upward. In Figure 3 (b), the player orients the wand tracker to the right, and the HMD shows that the gun is pointing to left.

4.3 Gun Radar

Gun Radar (a small window that gives the weapon's point of view) is one of the important features. This radar makes our system possible to aim and shoot at arbitrary direction (even if it is out of the player's eye view). To render this Gun Radar we make a new view port. In this view port we align the player's position and orientation to the gun's position and orientation. Then we render the virtual world in the view port. After that we restore back the orientation and the position of the player. Note that the center view of the gun radar becomes the target point.

4.4 Weapon's Pointer

Weapon's pointer is designed like a laser pointer. It shows the position where the bullet will hit the target. This feature makes the aiming in the virtual word easy. This feature is dependent on the gun radar. We need the z value of the pixel in the center of the Gun Radar (the target point). The z value can be taken from the z buffer. After we get the z value, we have the x, y, and z coordinate of the target point in the frustum. We multiply this point with the inverse projection matrix, to get the target point in the game engine's world coordinate. We draw a transparent red square to mark that this area is pointed by the weapon. In Figure 3 we draw it as white square to make it clear.

4.5 Jumping

We implement an interface, such that if the player jumps in the real word, the virtual character in the virtual world will also jump. We detect the delta z from the head tracker, if the delta is greater than certain threshold. We assume that the player is jumping. This feature can be useful to evade from the enemy bullets. This feature makes the physical movement substantial. Figure 6 shows how the player jump, and how his jumping affects the virtual character.

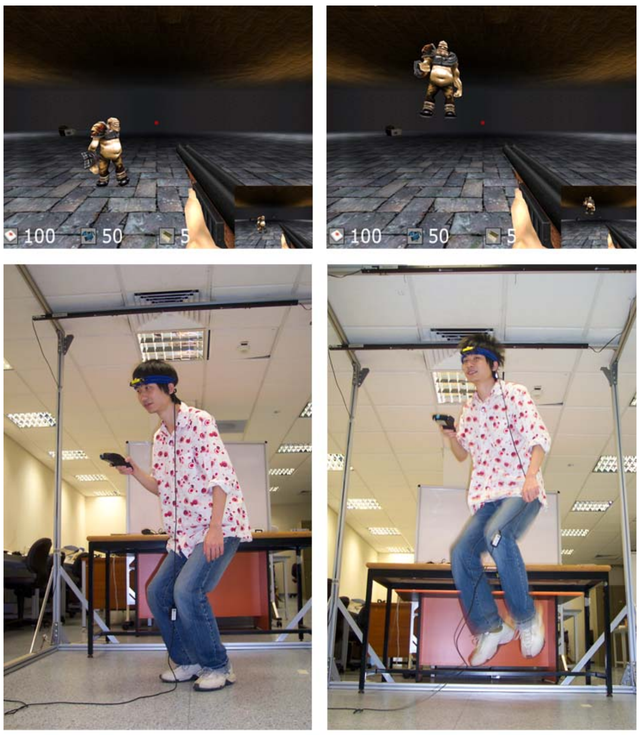


Figure 6. Jumping

5. CONCLUSION AND FUTURE WORK

We have successfully built a system for FPS game using ultrasonic tracking system. The system encourages the tangible and physical interactions between players, especially in a tense competition situation. We use a novel way to present the user's viewpoint and the gun's viewpoint. It allows user to move freely inside the room while aiming at targets using a gun-like handheld device. Jumping is also tracked so that the user has more options to avoid the bullets. This is novel and also very entertaining. As this is an ongoing project, as we wrote this paper we haven't got the customized HMD with tracker and camera yet. Later on when we have full system setup, we will continue our work with the user studies of players' experience during the gameplay.

6. REFERENCES

[1] Beckhaus, S., Blom, K. J., and Haringer, M. *A New Gaming Device and Interaction Method for a First-Person-Shooter.* University of Hamburg, 2005

[2] Thomas, B., B. Close, J. Donoghue, J. Squires, P. De Bondi, and W. Piekarski, *First person indoor/outdoor augmented reality application: ARQuake.* Personal and Ubiquitous Computing, 2002. **6**(2): p. 75-86

[3] Rajlich, P. *CAVE QUAKE.* http://brighton.ncsa.uiuc.edu/~prajlich/caveQuake/

[4] GameRunner. http://www.fpgamerunner.com/index.php

[5] Jacobson, J., Lewis, M. "Game Engine Virtual Reality with CaveUT," *Computer*, vol. 38, no. 4, pp. 79-82, Apr., 2005.

[6] Oortmerssen, W. http://www.cubeengine.com/

[7] Cheok, A. D., S. W. Fong, K. H. Goh, X. Yang, W. Liu, Farbiz, F. Human Pacman: A Sensing-based Mobile Entertainment System with Ubiquitous Computing and Tangible Interaction. In *2nd Workshop on Network and System Support for Games*, pages 106.117. ACM Press, 2003.

[8] Cheok, A. D., X. Yang, Z. Z. Ying, Billinghurst, M., and Kato, H. *Touch-Space: Mixed Reality Game Space Based on Ubiquitous, Tangible, and Social Computing.* Personal and Ubiquitous Computing, 2002. **6**(5-6)