Immersive Multiplayer Games With Tangible and Physical Interaction

Jefry Tedjokusumo, Steven ZhiYing Zhou, and Stefan Winkler

***Abstract*—In this paper, we present a new immersive multiplayer game system developed for two different environments, namely, virtual reality (VR) and augmented reality (AR). To evaluate our system, we developed three game applications—a first-person-shooter game (for VR and AR environments, respectively) and a sword game (for the AR environment). Our immersive system provides an intuitive way for users to interact with the VR or AR world by physically moving around the real world and aiming freely with tangible objects. This encourages physical interaction between players as they compete or collaborate with other players. Evaluation of our system consists of users' subjective opinions and their objective performances. Our design principles and evaluation results can be applied to similar immersive game applications based on AR/VR.**

***Index Terms*—Augmented reality (AR), first-person shooter, physical interaction, tangible user interface, virtual reality (VR).**

I. Introduction

VIDEO games are usually played using standard input devices like keyboard, mouse, and joysticks. The graphics are displayed on monitors, televisions, or on projector screens. Such configurations can only provide limited immersive experience of the game and lack physical and social interactions.

In this paper, we present a new immersive system for multiplayer games. In our system, the players wear a head-mounted display (HMD) with head tracker, and carry a wand in their hand that is also tracked (see Fig. 1). With these tracking devices, we can offer a new way to play multiplayer games. The player's position and viewing direction are tracked by the head tracker. The wand's pose is associated with the weapon's position and orientation in the game. The whole tracking system is placed in a rectangular room without obstacles (see Fig. 1). We have developed three game applications for evaluation purpose. The system design principles and the results of the user study presented in this paper can be applied to other similar immersive multiplayer games based on virtual reality (VR) or augmented reality (AR).

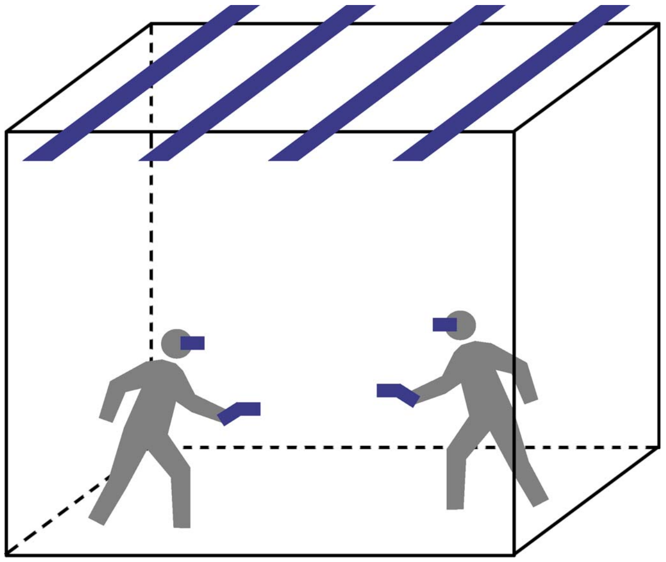


Fig. 1. Game environment. The players are wearing HMD (with head tracker) and holding a wand tracker. The aluminum bars on the top are fitted with ultrasonic emitters tracked by the wearable devices (HMD and wand).

1) First-person shooting (FPS) game in VR environment (FPS-VR): It is a virtual immersive game. We place game items (health, armor, weapon, bullets) in the corners of the room. The player must physically walk to these locations to collect the items.
2) FPS game in AR environment (FPS-AR): The setup and game play are similar to FPS-VR. We added a small camera in front of the HMD so that the player can see the real world with virtual weapons, items, monsters, and other game objects augmented on it.
3) AR Sword (ARSword): This is an AR game. We augment the virtual swords and game items over the video captured by the HMD camera.

This paper is structured as follows. Section II reviews the related works on immersive games. Section III describes our system setup. Sections IV and V discuss the implementation and subjective evaluation of our FPS-VR game system. Sections VI and VII discuss the implementation and subjective evaluation of our FPS-AR game system. In Section VIII, we compare the objective performances of three different game systems (Keyboard and mouse, VR, and AR). Section IX concludes this paper.

Manuscript received March 5, 2008. First published October 30, 2009; current version published December 16, 2009. This work was supported by Singapore A*Star Thematic Strategic Research Programme (TSRP) on Mobile Media under Project WBS R-263-000-458-305. This paper was recommended by Associate Editor C. M. Lewis.

J. Tedjokusumo was with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576. He is now with Ubisoft Singapore, Singapore 138633 (email: jefryted@gmail.com).

S. Z. Zhou is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576 (e-mail: elezzy@nus.edu.sga).

S. Winkler is with Symmetricom's Quality of Experience Assurance Division, San Jose, CA 95131 USA (e-mail: stefan@winkler.net).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2009.2028432

TABLE I
FEATURE COMPARISON OF IMMERSIVE GAME SYSTEMS

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

II. RELATED WORK

CAVE Quake [1] provided a good tangible user interface for FPS. CAVE is a $10 \times 10 \times 10$ ft "cube" with images projected onto three walls and the floor. The player stands at the center of the cube, experiencing the virtual world rendered on the walls in the front, left, and right, as well as on the floor. The player aims the gun using his hand as if he is in the real world. The player typically uses a joystick to move around the virtual world. CaveUT [2] is a similar system, with the latest version [3] supporting stereoscopic rendering extension. CaveUT is much less costly (around U.S.$25 000) compared to CAVE Quake, which may cost up to a million dollars. Our system differs in the way the virtual world is presented. Instead of using projections, we attach a tracking device to the HMD so that the user receives updated first-person views regardless of where he/she moves in the room. In this way, multiple players can have their own individual viewing experience.

ARQuake [4] is a single-player mixed-reality FPS. The player in ARQuake has the freedom to move around the world. However, the gun aiming is limited to the center of the view of the HMD, which means that the aiming is done by the player's head rather than the player's hand. This is not intuitive since it is contradictory to real practice. It is also difficult to aim properly using head movement while avoiding enemies' attack. The game was initially designed as a single-player game, so it has no interactivity with other human players. Although this game can be extended to support multiple players, the tracking systems that are used (GPS and markers) are meant for outdoors which offer less accuracy. Therefore, such games will have difficulty in determining whether a bullet shot from one player has actually hit another player. Human Pacman [5] is another example of outdoor AR games. However, it also uses GPS for outdoor tracking; it suffers from the same accuracy problem as ARQuake. Our game is based on indoor ultrasonic tracking system (InterSense IS900) which has much higher accuracy that is essential for FPS games.

Touch-Space [6] is an indoor mixed-reality game that is playable in a room-size space, using a high-accuracy ultrasonic tracking system from InterSense similar to ours. It has three stages: physical land exploration, virtual land exploration, and virtual castle exploration. The stages range from AR to VR. Our system differs from Touch-Space in that we add in more physical interaction components, e.g., moving close to virtual armor boxes to collect armor and jumping to avoid bullets. Unlike the tasks in Touch-Space that emphasize collaboration, our game involves intensive competition between users. This allows us to evaluate the system in more scenarios.

Beckhaus *et al.* [7] proposed ChairIO and a Game Gun as new control devices for FPS games. The ChairIO is basically a chair that tilts. The tilting 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 sitting on the chair hence physical movements are very limited. Our system offers free body movements by using wireless trackers. Users can move freely within the room and even jump to avoid bullets. Furthermore, in Beckhaus' work, the gun aiming is done by the Game Gun and is also limited to the center of the screen, i.e., the game view will always follow the gun's orientation. A similar game controller concept is used in GameRunner [8]. However, in physical life, the user's viewpoint does not necessarily follow the gun. Our solution of separating the views offers a new and intuitive approach for FPS games.

Table I summarizes the feature comparison between our system and the other related works. Our system provides an intuitive and tangible controller in the virtual world and, at the same time, maintains collaborative/competitive physical interaction between the players in the physical world. The key novelty lies in the detachment of the player's first-person view and the aiming view that should be associated with the handheld device (e.g., gun in FPS).

III. SYSTEM

Our system consists of a game engine and an ultrasonic tracking system (InterSense IS900) with highly accurate tracking (2–3 mm for position and $0.25^\circ - 0.4^\circ$ for orientation). Its coverage is limited to the space below the sensors, but it can easily be extended by adding more sensors. The tracking system covers a space of 2×4 m^2 (see Fig. 1). The tracking system is connected to a computer (the tracking server) that sends user datagram protocol (UDP) packets containing tracking data (position and orientation) to the tracking clients at a rate of 90 Hz (see Fig. 2).

We implemented a two-player system, with two tracking devices (a head tracker and a wand tracker) for each player. The head tracker is mounted on the HMD to track the player's head pose. The game engine then renders the virtual environment based on this information to provide a first-person view. The wand tracker is used as a gun or a sword for the player to aim or slash the target in the games.

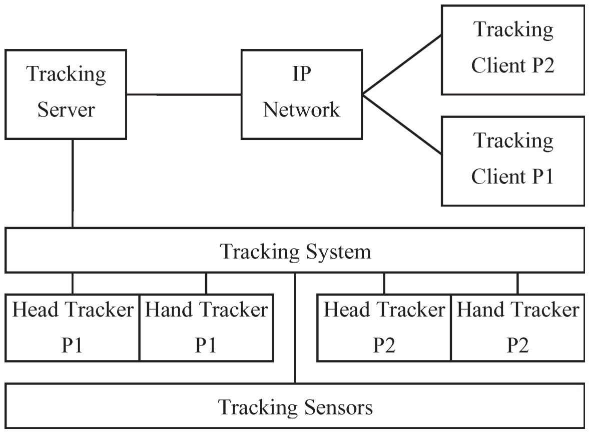


Fig. 2. Tracking system diagram. The tracking system is a proprietary device of InterSense which sends the tracking data to a tracking server trough serial port (RS-232). The tracking server will then distribute the data to the clients through LAN.

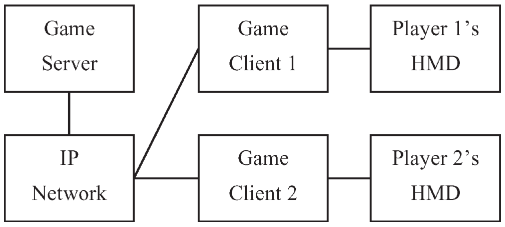


Fig. 3. Game system. This is a standard client-server design used by many multiplayer games. Each client will render the video images to be displayed on the HMD.

The game engine can support multiple clients. Currently, our implementation uses one server and two clients (see Fig. 3). We attach a small camera in front of the HMD for the AR-based application's purposes. The video images from the camera are captured and processed by the client. In general, there are two design paradigms to build a client-server architecture.

1) All the game logic is implemented on the client side. The server only receives the information from all the clients and then broadcasts it to all other clients. Most game engines are designed this way.
2) All the game logic is implemented on the server side. The server only sends necessary information (for rendering graphics and sounds) to the clients.

The open-source VR game engine we use incorporates design 1 while our custom-built AR game engine uses design 2 for easier data synchronization between the clients.

IV. VR Game Implementation

We developed an FPS game for our VR game system. The FPS-VR game style is "death match," which means that the two players shoot each other until one player's health level reaches zero. The winner is the one who kills more. The game style can be easily changed to support more than two players, so there can also be a collaborative death match between two groups, provided more trackers are used in a larger space.

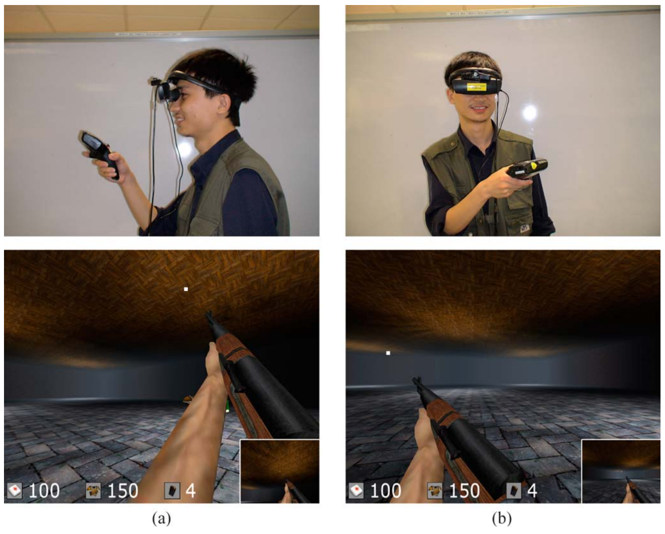


Fig. 4. In-game screenshots and the corresponding player poses. We separate the weapon and head view. Hence the player can aim using their hand. The small window on the right corner is the weapon point of view. This separation allows the player to shoot enemies outside his/her view range (even at the backside).

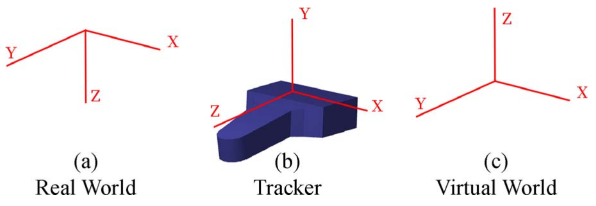


Fig. 5. Coordinate systems. (a) and (b) Coordinate system of IS900. (c) Game engine defined coordinate system.

For implementation of the games, we used the Cube engine [9] which is an open-source game engine written in C++. The Cube engine has a built-in world editor and supports .md2 models. The player's HMD displays the image, as shown in the second row of Fig. 4. The small window in the bottom right corner of the display is the gun radar that shows the gun view. With this gun radar, the player can aim and shoot in any direction inside or outside the player's own field of view. This interface adds an element of excitement, as the player can aim and shoot the target even when the target is at the player's back. The white dots shown in the second row of Fig. 4 are the weapon pointers. The weapon pointer helps the player to aim at targets.

A. Calibration of FPS-VR

The first step to integrate the tracking system and the game engine is calibration. Fig. 5(a) shows the real-world coordinate system of the tracking system (where the players physically move). Fig. 5(b) shows the coordinate system of the wand tracker. Fig. 5(c) shows the virtual-world coordinate system of the game engine (where the virtual characters move). The z-axis in the real world points to the opposite direction of the z-axis in the virtual world. The game engine defines yaw as rotation around the z-axis, pitch as rotation around the y-axis, and roll as rotation around the x-axis.

We calibrated the hand tracker to give yaw, pitch, and roll values of zero when the axes align as in Fig. 5(a) and (b). Calibration of the position is simple. Because the x-axes and the y-axes of the real and the virtual world are aligned; we only need to adjust the scaling.

B. *Weapon Orientation in FPS-VR*

Like most game engines, the Cube engine's weapon orientation points to the center of the screen and is tied to the player's view. To make the weapon orientation independent of the player's view, we add new variables for yaw, pitch, and roll of the weapon. We render the weapon's orientation according to the data that we get from the wand tracker. This feature is shown in Fig. 4. Note that we assume the weapon's position to be the same as the head position.

C. *Gun Radar in FPS-VR*

The Gun Radar (a small window that shows the weapon's point of view) is one of the important features of our system (cf., Fig. 4). This radar makes it possible to aim and shoot accurately in an arbitrary direction (even if it is out of the player's view). To render this Gun Radar, we create 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 the orientation and the position of the player. Note that the center point of the gun radar becomes the target.

D. *Weapon Pointer in FPS-VR*

The weapon pointer is designed like a laser pointer. It shows the position where the bullet will hit the target. This feature facilitates aiming in the virtual world. It is dependent on the gun radar. We need the z-value of the pixel in the center of the Gun Radar (the target point), which can be read from the z-buffer. Then, we have the x-, y-, and z-coordinates 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 coordinates. We draw a transparent red dot to mark this area.

E. *Jumping in FPS-VR*

If the player jumps in the real world, the virtual character in the virtual world will also jump. To achieve this, we detect the delta-z from the head tracker; if it is greater than a certain threshold, we make the player's avatar jump. This feature can be useful to dodge enemy bullets. It also encourages physical movement. Fig. 6 shows this feature.

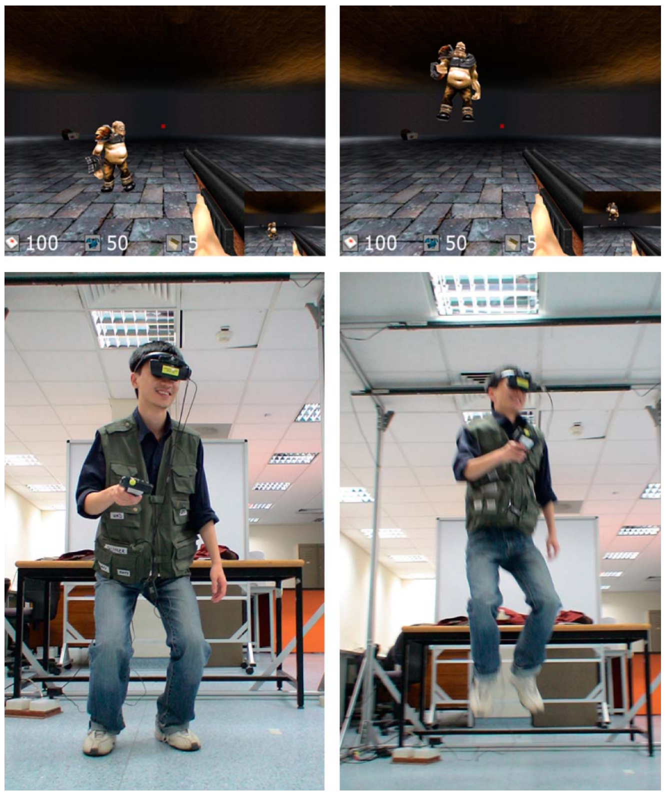


Fig. 6. Jumping. On the left, the player is preparing to jump. On the right, the player is jumping. The player's avatar in the virtual world mirrors the player's actions.

V. VR Game Subjective Evaluation

The subjective user study was conducted with 30 students (25 males and 5 females). Their age ranged from 19 to 29 years. We conducted 15 sessions of 3 min each, in which two persons played against each other in a death match battle. For comparison, we also asked them to try the traditional version of the game using keyboard and mouse. After they had tried both interfaces, we asked them to fill out a questionnaire. Most of the questions are comparisons between our system and the traditional FPS computer interface. The respondents answered on a scale from one (prefer traditional implementation) to seven (prefer immersive system).

We also included some introductory questions to find out about their experience with computers and FPS games. Eleven of them had been using computers for more than ten years, and 23 of them had played FPS games before. The first question was "How would you categorize your self as an FPS gamer?" They have to answer on a scale from one (beginner) to seven (expert). The results are shown in Fig. 7(a). The mean is 3.04. Then, we asked how often they play FPS games. Most of them answered "several times" (14 persons).

As reported by Joseph and LaViola [10], using an HMD in a standing position for a long period increases the probability of getting cyber sickness. We wanted to evaluate how comfortable the HMD was for short term use (3 min) compared to a computer screen. The results on a scale from one (computer screen preferred) to seven (HMD preferred) are shown in Fig. 7(b). The mean is 4.43, indicating that our participants are quite comfortable with the HMD in this short-term task.

We also asked the participants to compare the mouse and the wand in terms of accuracy as well as excitement on a scale from one (mouse better) to seven (wand better). The results are shown in Fig. 7(c). The mean is 3.93 for accuracy and 5.66 for excitement, which indicates that our wand is perceived to be about as accurate as the mouse, but in terms of excitement, the wand is preferred.

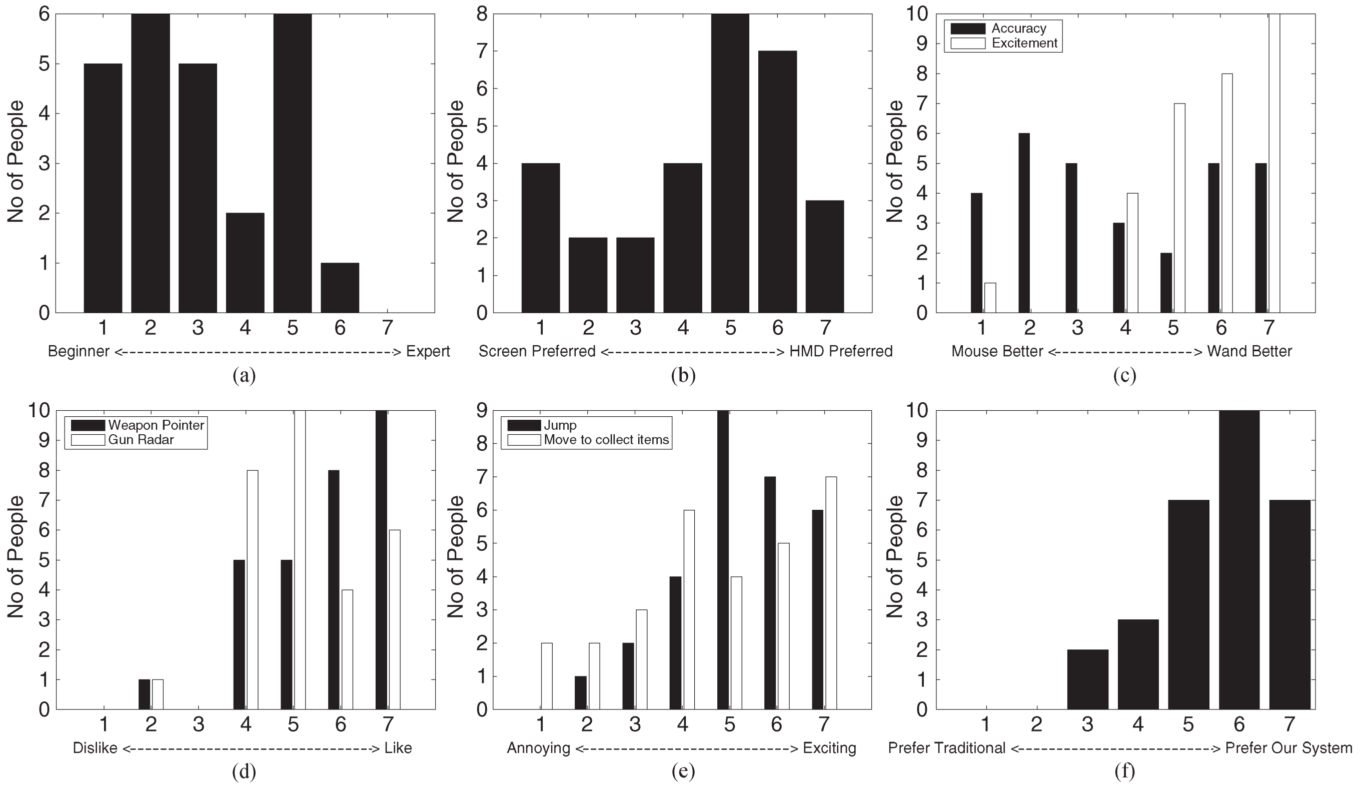


Fig. 7. Answers of the FPS-Traditional versus FPS-VR user study participants. There are 30 participants: 25 are males and 5 are females. (a) How would you categorize yourself as an FPS gamer? (b) How comfortable do you feel about HMD compared to monitor screen? (c) Aiming gun using mouse and aiming gun using hand tracker, which one is better? (d) How do you find these features compared to traditional FPS game? (e) How do you feel about these physical movements? (f) Overall preference of the user study participants.

We also investigate the user acceptance of the new game features (Weapon Pointer and Gun Radar). The results are shown in Fig. 7(d) on a scale from one (dislike) to seven (like). From the chart, we can see that most answers are four or higher. The mean answer is 5.69 for the Weapon Pointer and 5.17 for Gun Radar. This shows that the participants like the new features.

Our system incorporates two major types of physical movements, namely, jumping to dodge bullets, and moving to a particular location to collect certain items (health, armor, bullets, etc.). We wanted to know how users feel about these physical movements. Fig. 7(e) shows the results on a scale from one (very annoying) to seven (very exciting). The mean answer is 5.28 for jumping and 4.76 for moving to collect items. This shows that physical movements are well received by the users. Jumping gives more excitement than just moving around by walking.

In the last question, we asked “Overall, how do you feel about using physical body movement to play the FPS game compared to using traditional keyboard and mouse?” The results are shown in Fig. 7(f) on a scale from one (prefer traditional interface) to seven (prefer our system). The mean answer for this question is 5.59, which shows that our system is significantly more attractive than the traditional keyboard and mouse interface for FPS gaming.

VI. AR Game Implementation

Our game applications, namely, FPS-AR and ARSword, share the common game engine which is different from the game engine we used in FPS-VR, which is not suitable for AR application. The game style for both games is also “death match.” The difference between FPS-AR and ARSword is the weapon used by the player. FPS-AR uses a gun for shooting while ARSword uses sword for striking. A similar AR system using Polhemus magnetic sensor to track the sword has been developed by Ichikari *et al.* [11]. This paper is targeted at using AR movement for visual augmentation in film making. Our focus in this paper is the development and evaluation of such a system in the entertainment context.

Almost all game engines are developed for VR environment with terrain, sky box, and navigation controls that are not needed in AR environment since they will be replaced by the physical scenes captured by the camera and the physical movements. Therefore, the software structure in a VR game engine is generally hard to be applied in AR applications. We hence built the AR game engine ourselves using the DirectX application programming interface. In the physical setup, a camera is mounted in front of the HMD (see Fig. 8) to capture the player’s real-world view. For simplicity, we do not work on stereo camera/HMD. MXRToolkit is used [12] only for calibration purposes to register the camera and IS900 coordinate system. In this section, we will focus on the calibration process because this is the fundamental technical challenge faced by most AR applications.

A. *Related Work of Calibration in AR System*

One of the earliest work in AR calibration was done by Janin *et al.* [13] in 1993, who developed an HMD calibration

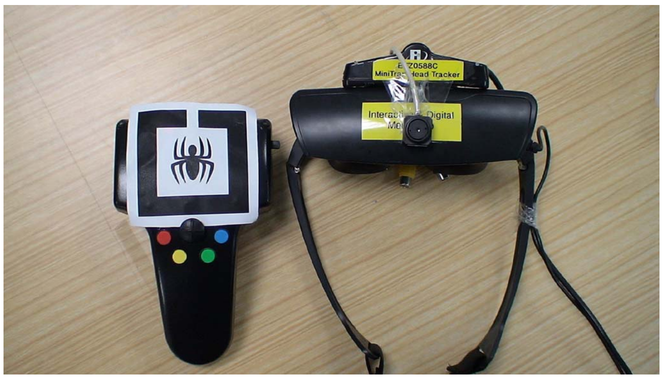


Fig. 8. Wand/Hand Tracker with a marker and the HMD with a small camera. The marker on the wand tracker is only used for calibration.

method for AR applications and used it for touch labor manufacturing. State *et al.* [14] used hybrid tracking that combines magnetic tracking and stereo vision tracking. Their algorithm can successfully register the virtual objects over real objects on the table. Fuhrmann *et al.* [15] proposed a user-friendly method to calibrate the HMD, stylus, and virtual table for AR systems. Grasset *et al.* [16] developed a multiuser table-top setup, where each user wears an HMD and interacts with one another in an AR environment. These calibration methods were designed for close-range AR applications (i.e., on the table with interaction distance less than 1 m from the HMD). For close-range AR applications, small calibration error is not noticeable. However, for our room-size application, the alignment error is noticeably big, particularly at the corners of the room.

Most recently, Hua *et al.* [17] proposed a systematic calibration method that estimates both intrinsic and extrinsic parameters of the display system and thus establishes the viewing and imaging transformations for rendering 2-D images from 3-D virtual environments, to address the registration issue in a customized head-mounted projection display (HMPD)-based AR system. Although HMPD has been recognized as an alternative solution for a wide range of 3-D visualization applications and has recently been explored extensively, we are still using HMDs for convenience.

Computer vision algorithms are very CPU intensive and prone to fail when the features being tracked are blocked. Researchers try to combine camera with other sensors to develop sophisticated and robust AR applications. The method of combining more than one sensor is often called hybrid tracking. You *et al.* [18], [19] and Satoh *et al.* [20] combined inertial tracking system with vision-based tracking to develop AR applications. We are using similar combination but only for calibration purpose. Our application ARSword, which involves fast movement of the swords, is not suitable for using computer vision tracking; hence, we fully rely on IS900's ultrasonic tracking.

B. Calibration Method in FPS-AR

Baillot *et al.* [21] presented a method for aligning trackers in AR systems. The alignment process is easy and intuitive: A user is asked to stand at a known location and align the display with known objects in the environment. Our method is similar to what Baillot *et al.* proposed, but with the help of the

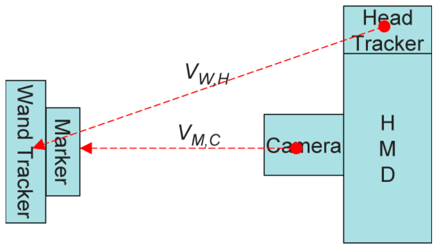


Fig. 9. Calibration diagram for mixed-reality application using IS900 and MXR Toolkit.

marker from MXRToolkit, the user can stand anywhere (inside the room) and just look at the wand tracker (with marker on it) and the calibration process will run (see Fig. 9).

We need to find Transformation matrix T that transforms the coordinate of head tracker to the camera. All the positional and orientational data that we get from the IS900 will be transformed by this matrix T (note that we do not need marker except for calibration). We stick a marker on the wand tracker (see Fig. 8). MXRToolkit will give the vector $V_{M,C}$ of marker position from the camera. Wand Tracker to Head Tracker translation $V_{W,H}$ vector can be calculated from IS900 data. The translation component of matrix T can be calculated as $V_{W,H} - V_{M,C}$. Figs. 9 and 10(a) show the process of calibrating the translation component.

For the rotation component of matrix T, we manually align the virtual walls to the real one by setting the yaw, pitch, and roll values using the joystick on the Wand tracker. Once the value of T (translation and rotation) is calculated, we can save it and use it for the next time. Fig. 10(b) and (c) shows the process of calibrating the rotation component. Before calibration, even though the orientation difference between the camera and the head tracker is small, the misalignment effect at the corners of the room is still noticeable.

C. Game Design of FPS-AR

In typical FPS games, the player reduces the enemy's health by shooting bullets or slashing with the virtual sword at the enemy. We developed two types of bullets: 1) instant hit bullets and 2) noninstant hit bullets that leave a trail, which we call projectile. To help the player aim accurately, we draw a translucent red line from the gun position directed toward the gun's crosshair (see Fig. 13).

To incorporate these functionalities, we also needed to implement collision detection. To calculate collision detection efficiently, we make a bounding volume around the players and all the virtual items (swords, bullets, etc.). We choose an oriented bounding box (OBB) because of its high accuracy and relatively low computation cost. We use a collision detection algorithm developed by Gomez [22]. Fig. 11 shows the player and the swords with their OBB.

The interactions in the game are mostly triggered by collisions of virtual objects. Here is the list of collisions between objects and the results of the collision.

1) Sword-Sword: Play a sound of sword striking at each other.
2) Sword-Player and Projectile-Player: Reduce the player's health, augment the whole player's view with a red

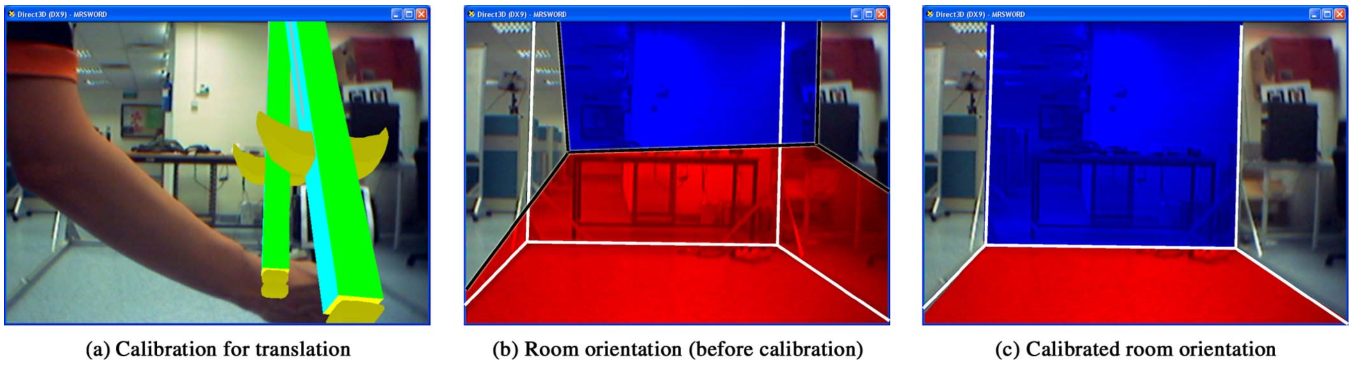

(a) Calibration for translation
(b) Room orientation (before calibration)
(c) Calibrated room orientation

Fig. 10. (a) Sword on left is from IS900; the sword on right is from marker. (b) Black lines are room edges perceived by head tracker; (c) both coordinate system of camera and head tracker are aligned; the white lines are room edges perceived by camera.

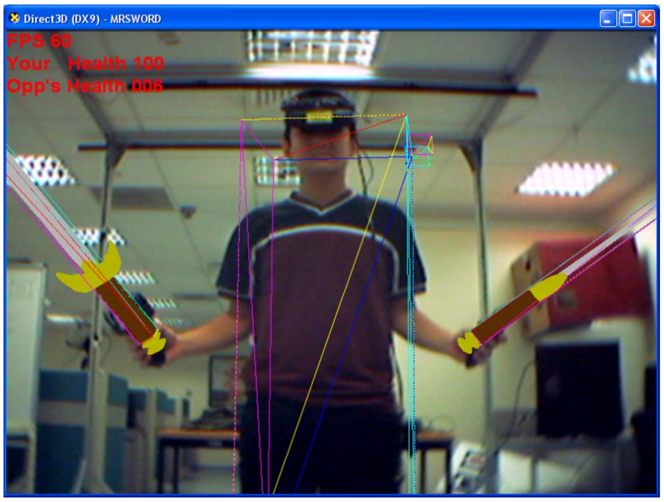


Fig. 11. MRSword in action. The player body is holding two virtual swords. Player's body and the swords are covered by OBB.

colored translucent layer representing the "danger" effect, and play a speech "ouch."

3) Sword-Projectile: Change the direction of projectile movement and play a sound effect.

As explained in Section III, all the game logic (collisions detection, health calculation, and round changing) are calculated only by the game server. This is necessary for synchronization and to maintain data integrity between the clients. The client's only task is to render the graphics on the HMD. This design allows us to make more complex game rules (at the server side) and better graphics (at the clients' side).

D. Limitations of the Tracking Method Used in FPS-AR

We found that IS900 tracking may not be accurate enough for mixed-reality applications in a large room. Suppose at position A the sensor values result in transformation (pose) matrix T, then we move the tracker to some arbitrary position A' and move it back to A, the sensor values give matrix T' which is different from T. Although the difference is small, it affects the positioning of augmented virtual objects at far distance (i.e., > 1 m), i.e., all the augmented virtual objects can be misplaced. This problem is caused by drift that is inherent in inertial tracking and time latency which is caused by the fact that the speed of sound $\ll$ speed of light. A similar problem is also reported by Qi [23].

IS900 incorporates two sensors: inertial and ultrasonic. The orientations of devices are usually dependent on inertial gyroscope while the position is dependent on the ultrasonic sensors. The inertial gyroscope works quite well, while the ultrasonic sensors sometimes fail. In order to accommodate the aforementioned limitations, we implement a low-pass filter to the tracking results to avoid jittering of the augmented graphics.

VII. AR Game Subjective Evaluation

The subjective user study was conducted with 15 students (12 males and 3 females), whose ages range from 16 to 27 years. We conducted 15 sessions of 10 min each. We asked the subjects to try the FPS-VR for 5 min and FPS-AR for another 5 min. After they had tried both interfaces, we asked them to fill out the questionnaire. Most of the questions are comparisons between FPS-VR and FPS-AR. The subjects needed to give their answers based on a Likert scale.

We wanted to compare FPS-VR and FPS-AR in terms of aspects like dizziness during game play and ease of navigation. Subjects' answers are shown in Fig. 12(a) with the mean 3.73. This shows that they prefer the VR environment. They are constrained by the cables. They have a very limited movement (i.e., only three or four steps). In AR, they hardly feel that they are moving, while in VR that has a clear texturing of the walls and floor, their steps translate to greater displacement. This exaggeration in the VR world gives better feedback that tells the user whether they are moving, while some who prefer AR claim that seeing the real environment makes them feel safer to walk (i.e., not afraid of hitting something or falling down because of the cables).

Next, we compare the aiming of the weapons in terms of easiness and excitement/fun. The results are shown in Fig. 12(b). The mean is 2.8 for easiness and 4.4 for excitement/fun. Aiming in AR is more difficult without stereo vision. To aim accurately, they have to put their wand tracker in front of their head (see Fig. 13). In VR, they can relax their hand and use the weapon pointer for aiming. We cannot provide this weapon pointer in the AR configuration, unless we have the depth information of the real-world objects.

We think that FPS-AR is a better option for shooting training because you need to pose like a real soldier when shooting,

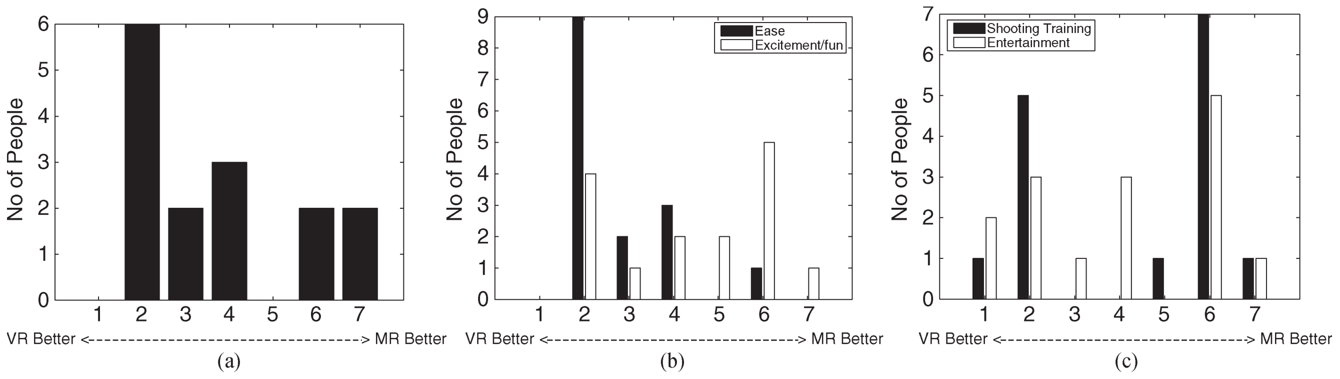


Fig. 12. Results of the subjective user study (FPS-VR versus FPS-AR). (a) Which game mode is more comfortable for you? (i.e., less dizziness, easier to navigate). (b) Which game mode is better for aiming, in term of easiness and in term of fun/excitement? (c) Which game mode is better for shooting training, and which is better for entertainment?

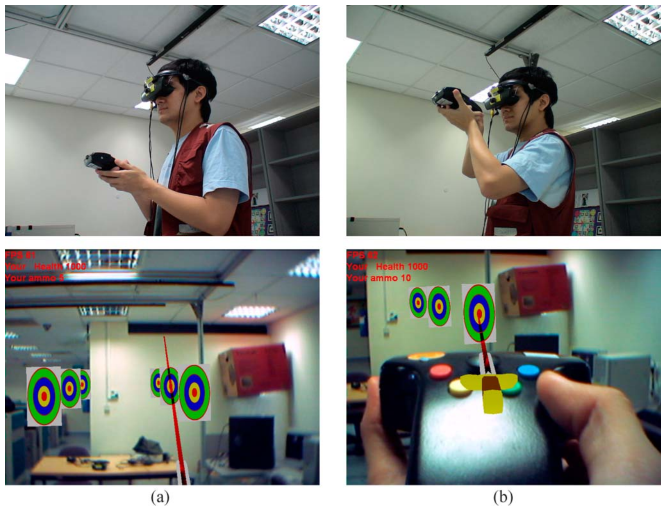


Fig. 13. Corresponding player pose and FPS-AR in game screenshot. We use the sword model as the gun. Aiming with pose (a) will be difficult. Pose (b) is the preferred mode of aiming.

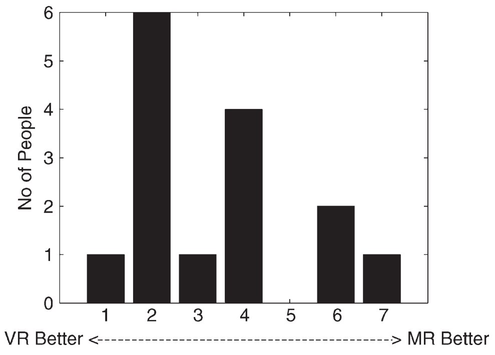
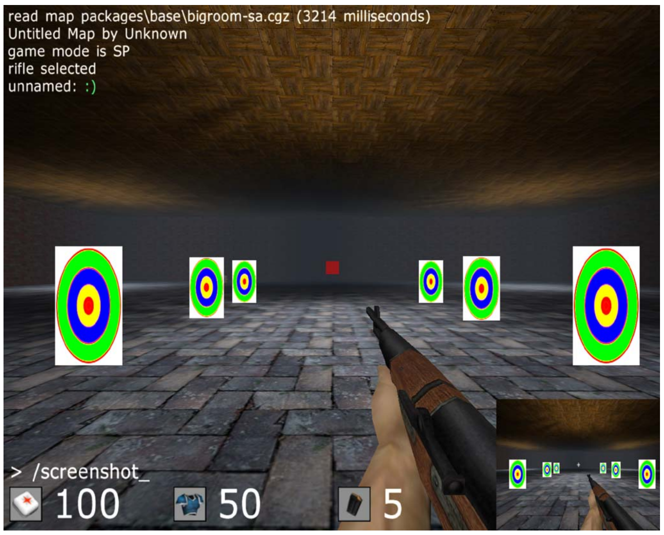


Fig. 14. Overall gaming experience, which mode do you prefer?



```
read map packages\base\bigroom-sa.cgz (3214 milliseconds)
Untitled Map by Unknown
game mode is SP
rifle selected
unnamed: :)
> /screenshot_
100   50   5
```

Fig. 15. Six static targets. This is the screenshot of our objective performance test. We use bull's-eye target boards, so the users know where the center of the target is, which we use to calculate the accuracy of the shooting.

while FPS-VR is more suited for entertainment because of easier aiming and relaxed actions. The results of the participants' opinion are shown in Fig. 12(c). The mean is 4.33 for shooting training and 4 for entertainment. Nine out of 15 agree that FPS-AR is better for shooting training. While for entertainment value, both are on par.

The results of the overall preferences are shown in Fig. 14. The mean is 3.4 which shows that most of them prefer the FPS-VR. The main reason is that the VR environment is more immersive. The exaggeration of the scale and the graphical effects also add more value to the preferences.

VIII. Objective Performance

We conducted three objective evaluations. The objective evaluation involved 9 people using the mouse and keyboard interface, 16 people using FPS-VR, and 15 people using FPS-AR. Their ages ranged from 16 to 27. Before the test, we gave each player 5 min to familiarize themselves with the system. We guided them on how to aim properly, how to collect additional bullets, and briefed them on the main and secondary objectives.

We equipped each player with five bullets, and put additional virtual bullets at the corners of the room. The gun can only shoot one bullet at an interval of 1.8 s. The spare bullets will only reappear 8 s after they are picked up. The main objective is to clear the level (by shooting down all the targets) as fast

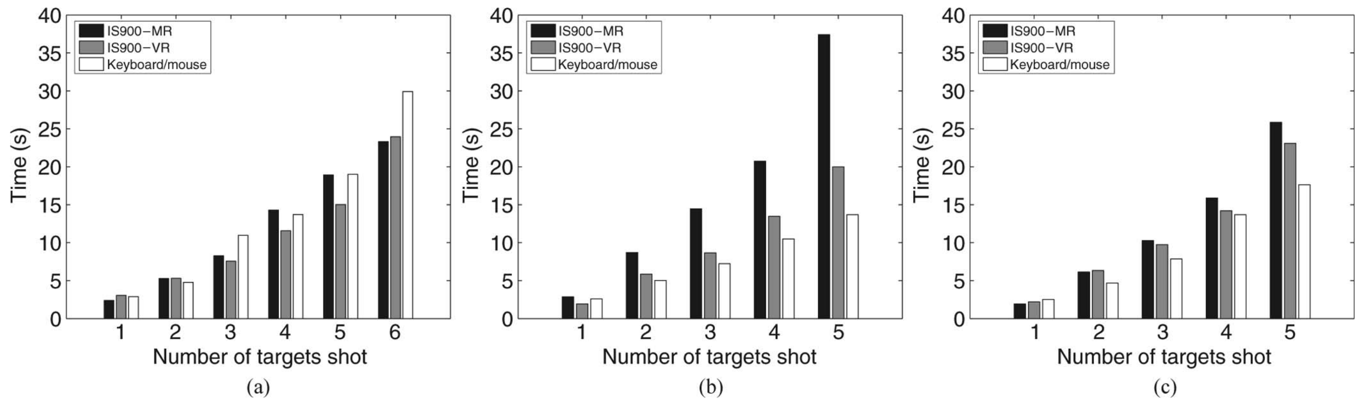


Fig. 16. Results of the objective performance tests. We calculate the average time taken by the users when he has successfully hit n targets. (a) Test 1 (six static targets). (b) Test 2 (five dynamic targets). (c) Test 3 (five dynamic targets + attack).

as possible while at the same time maintaining the shooting accuracy. We designed three different difficulty levels.

1) The player had to shoot six static targets (see Fig. 15). We designed the level such that the player will need to collect the bullets at least once, because we only equipped him/her with five bullets.
2) The player had to shoot five moving targets. The fastest way to clear the level is to shoot all the targets without any miss. If he/she only missed one target, he/she will have to collect additional bullets at the corner of the room.
3) The same as the above except that the targets will also attack the player. We also calculated how much damage the player had incurred when the game ended.

A. Task Completion Time

The first objective performance that we measure is the task completion time. Fig. 16 shows the average time taken by the players to hit the target. As the difficulty level increases, the time needed to hit the target also increases. In test 1, the task completion time difference between hitting five targets and six targets is big. This is expected because the player needs to collect extra bullets first before he/she can finish the level. In this test, we found that the keyboard and mouse interface performs the best in most of the cases followed by FPS-VR and FPS-AR.

There was one case found in test 1 in which keyboard and mouse performed badly. A female subject, who has never played any FPS game before, had difficulty in playing the game using both the keyboard and mouse simultaneously. While most players finished the game in less than 25 s, she took 80 s. However, she performed much better in FPS-VR and FPS-AR. This result indicates that our new system is more suitable for those who have never played any computer game before. It also shows that the familiarity with the existing interface contributes significantly to the performance of the aiming task in FPS games.

FPS-AR performs worse in test 2 [Fig. 16(b)] than in test 3 [Fig. 16(c)] that we thought should be more difficult. There were two subjects who needed more than 100 s to finish test 2, mainly due to the fact that the targets were sometimes hidden behind the player. These two subjects performed better in test 3 because the position of the targets could be predicted from the projectile trail when the targets also shoot at the player.

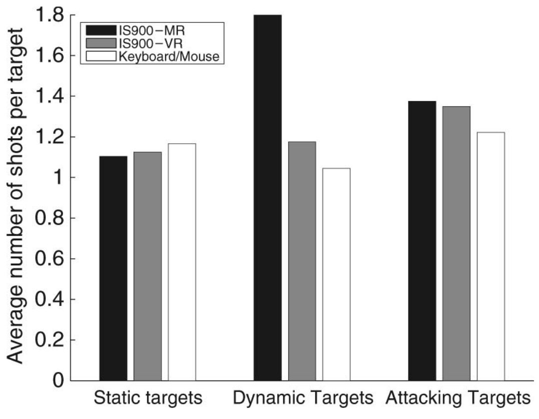


Fig. 17. Average number of bullets needed to shot a target (the best case is one).

B. Shooting Accuracy

The second objective performance that we evaluated is the shooting accuracy. We measured two types of shooting accuracies: 1) the number of hits per target and 2) the normalized distance between the target center and the hit point. The number of shots fired by the player and the number of times the bullets hit the target are counted. Fig. 17 shows the average of the players' performance for each test. In test 3, since targets also attack the player, generally more shots are needed to destroy a target. Note that the FPS-AR test 2 (Dynamic Targets) has the highest number of shots per target because the users who needed more than 100 s to finish the game shot more bullets. In normal cases, keyboard/mouse performs best followed by FPS-VR and FPS-AR.

For every bullet that hit the bull's-eye target, we calculated the intersection coordinate. The distance of that coordinate from the center of the target is measured and normalized to a value from zero to one, where zero is the center of the target (most accurate) and one is the edge of the target (almost miss the target). Fig. 18 shows the results. Keyboard and mouse perform better for all cases followed by FPS-VR and FPS-AR. The reasons are the following: 1) the subjects have got used to mouse/keyboard interface so that they perform better; 2) AR involves subject's physical movements which may slow down the process; and 3) the equipment used in VR/AR setup also

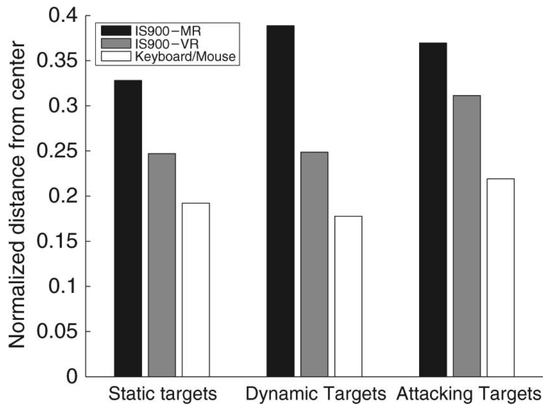


Fig. 18. Average the shoot accuracy of the players for each test.

affects the performance. Training is needed for the user to get familiar with the new interface.

C. Avoiding Attacks

The third objective performance measurement is the player's ability to avoid the attacks from the targets. During the test, all five targets shoot a bullet with projectile at the player. Each time when the player is hit, the health is decreased by five units. We recorded the damage that each player incurred at the end of the game. The mean of the damages incurred is 100 for FPS-VR and 67.22 for keyboard/mouse. We do not include the FPS-AR result in this paper as we have limited the number of projectiles in the AR world, which results in that the health decrement for FPS-AR is the smallest compared to others. However, based on the observation, subjects incurred less damages because they are more agile in avoiding attacks mainly due to the visibility of the physical surroundings, whereas in VR environment, they tended to be more cautious about making big movements. Keyboard/mouse interface still incurs the minimum damage mainly due to the faster response rate and the easiness of repeating the dodges.

D. Discussion

Section VIII shows that keyboard/mouse interface performs the best for almost all cases. By using this interface, the player performs faster to clear the in-game levels, shoots more accurately, and incurs lesser health damage. Possible reasons are as follows.

1) Our immersive FPS-VR/FPS-AR games using the IS900 tracking system constrain users' movement. The cables hinder the players from moving quickly and freely.
2) Real-world situation is harder for the game play than a simulated environment. Keyboard and mouse are used in situations where the player is seated comfortably and his/her attention is directed toward the game's objectives. However, in a real-world scenario, the user is conscious of his/her movements as well. Our experiments show that the performance of the player decreases as the degree of simulation gets closer to real-world game play. The results shown in Figs. 16–18 also confirm this.
3) Almost all of the subjects have used the keyboard/mouse interface for a very long time for a wide range of computer applications including games. VR/AR games will need longer warm-up training.

Although our game system does not perform as well as the keyboard/mouse interface, Section V, particularly Fig. 7(f), shows that users prefer our system in term of excitement. This result shows that such game system can be promising if we can resolve the problems previously mentioned. The success of Nintendo Wii has also indicated that physical body movements contribute a lot to the excitement of the games, although the sensing technology used by Wii Remote might not be suitable for FPS due to its low resolution.

Some subjects commented that our system is as accurate as playing with a keyboard and a mouse [see Fig. 7(c)] even though the objective experiment (see Fig. 18) shows the other way. This indicates that our interface is easy to use in FPS games. It is also worth noticing that a female subject, who had never tried FPS games before, performed better using our interface, compared to keyboard/mouse interface. This shows that the current keyboard/mouse interface for playing FPS game might not be intuitive and natural enough, while our interface is exactly mimicking the physical practices, such as avoiding attack with body movements, looking around with neck movements, and pointing at the target with arm movements.

IX. Conclusion

We have built an immersive system for multiplayer games using HMDs and a high-precision ultrasonic tracking system. The system encourages tangible and physical interaction between players, particularly in a tense competitive situation. It allows users to move and look around freely.

Our immersive FPS-VR application uses a novel way to present the user's view and the weapon's view separately. This is an important design principle in porting traditional VR systems to immersive VR systems. Jumping is also tracked so that the user has more options to avoid bullets.

HMD calibration in immersive Augmented/Mixed Reality application can be simplified by using marker and user's manual adjustment.

There are problems of accuracy in position and time delay with Ultrasonic sound tracker. This is acceptable for VR applications. However, for AR applications, this will cause misalignments between the virtual and real worlds. Using infrared trackers might improve the performance of the AR application.

Our subjective user study shows that FPS-VR is the most preferred game mode (see Figs. 7(f) and 14). A major reason behind this result is that the VR environment is more immersive; it makes you enter to a totally different world. It is exaggerating and imaginary effects (walk faster, totally different environment) add more value to it.

Our objective user study shows that Keyboard and mouse is the most effective tool to accurately shoot all the bull's-eye targets in virtual world. FPS-AR is better choice for shooting training, as the users need to pose properly (see Fig. 13) to aim the target accurately.

Although our game system does not perform as well as the keyboard/mouse interface, Section V, particularly Fig. 7(f), shows that users prefer our system in term of excitement.

REFERENCES

[1] P. Rajlich, CAVE Quake II. [Online]. Available: http://brighton.ncsa.uiuc.edu/~prajlich/caveQuake/

[2] J. Jacobson and M. Lewis, "Game engine virtual reality with CaveUT," *Computer*, vol. 38, no. 4, pp. 79–82, Apr. 2005.

[3] J. Jacobson, M. L. Renard, J.-L. Lugrin, and M. Cavazza, "The CaveUT system: Immersive entertainment based on a game engine," in *Proc. ACM SIGCHI Int. Conf. Adv. Comput. Entertain. Technol., ACE*, 2005, pp. 184–187.

[4] B. Thomas, B. Close, J. Donoghue, J. Squires, P. De Bondi, and W. Piekarski, "First person indoor/outdoor augmented reality application: ARQuake," *Pers. Ubiquitous Comput.*, vol. 6, no. 1, pp. 75–86, Feb. 2002.

[5] A. D. Cheok, S. W. Fong, K. H. Goh, X. Yang, W. Liu, and F. Farzbiz, "Human Pacman: A sensing-based mobile entertainment system with ubiquitous computing and tangible interaction," in *Proc. Workshop Netw. Syst. Support Games, NetGames*, Redwood City, CA, 2003, pp. 106–117.

[6] A. D. Cheok, X. Yang, Z. Y. Zhou, M. Billinghurst, and H. Kato, "Touch-space: Mixed reality game space based on ubiquitous, tangible, and social computing," *Pers. Ubiquitous Comput.*, vol. 6, no. 5/6, pp. 430–442, Dec. 2002.

[7] S. Beckhaus, K. J. Blom, and M. Haringer, "A new gaming device and interaction method for a first-person-shooter," in *Proc. Comput. Sci. Magic*, Leipzig, Germany, 2005.

[8] GameRunner. [Online]. Available: http://www.gamerunner.us/

[9] Cube Engine. [Online]. Available: http://www.cubeengine.com/

[10] J. Joseph and J. LaViola, "A discussion of cybersickness in virtual environments," *SIGCHI Bull.*, vol. 32, no. 1, pp. 47–56, Jan. 2000.

[11] R. Ichikari, R. Tenmoku, F. Shibata, T. Ohshima, and H. Tamura, "MR-based PreViz systems for filmmaking: On-set camera-work authoring and action rehearsal," in *Proc. 6th ISMAR Workshop Mixed Reality Entertain. Art*, Nara, Japan, Nov. 2007, pp. 21–26.

[12] NUS Mixed Reality Lab, MXR Toolkit. [Online]. Available: http://mxrtoolkit.sourceforge.net/

[13] A. L. Janin, D. W. Mizell, and T. P. Caudell, "Calibration of head-mounted displays for augmented reality applications," in *Proc. Virtual Reality Annu. Int. Symp.*, 1993, pp. 246–255.

[14] A. State, G. Hirota, D. T. Chen, W. F. Garrett, and M. A. Livingston, "Superior augmented reality registration by integrating landmark tracking and magnetic tracking," in *Proc. 23rd Annu. Conf. Comput. Graph. Interactive Techn., SIGGRAPH*, 1996, pp. 429–438.

[15] A. L. Fuhrmann, R. Splechtna, and J. Prikryl, "Comprehensive calibration and registration procedures for augmented reality," in *Proc. Eurographics Workshop Virtual Environ.*, 2001, pp. 219–228.

[16] R. Grasset, X. Décoret, and J.-D. Gascuel, "Augmented reality collaborative environment: Calibration & interactive scene editing," in *Proc. VRIC*, May 2001, pp. 1–8.

[17] H. Hua, C. Gao, and N. Ahuja, "Calibration of an HMPD-based augmented reality system," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 3, pp. 416–430, May 2007.

[18] S. You, U. Neumann, and R. Azuma, "Hybrid inertial and vision tracking for augmented reality registration," in *Proc. IEEE VR*, 1999, pp. 260–267.

[19] S. You and U. Neumann, "Fusion of vision and gyro tracking for robust augmented reality registration," in *Proc. VR Conf.*, 2001, pp. 71–78.

[20] K. Satoh, M. Anabuki, H. Yamamoto, and H. Tamura, "A hybrid registration method for outdoor augmented reality," in *Proc. IEEE ACM ISAR*, 2001, pp. 67–76.

[21] Y. Baillot, S. J. Julier, D. Brown, and M. A. Livingston, "A tracker alignment framework for augmented reality," in *Proc. 2nd IEEE ACM ISMAR*, 2003, pp. 142–150.

[22] M. Gomez. [Online]. Available: http://www.gamasutra.com/features/19991018/Gomez_5.htm

[23] W. Qi, "A prototype of video see-through mixed reality interactive system," in *Proc. 2nd Int. Symp. 3DPVT*, 2004, pp. 163–166.

Jefry Tedjokusumo received the B.Comp. and M.Comp. degrees from the National University of Singapore (NUS), Singapore, in 2004 and 2006, respectively.

He was a Research Fellow with the Department of Electrical and Computer Engineering, NUS. He is currently with Ubisoft Singapore, Singapore, as a Game Developer. His research interests include computer graphics, computer vision, augmented and virtual reality, and human–computer interaction.

Steven ZhiYing Zhou received the B.Eng. and M.Eng. degrees from Southeast University, Nanjing, China, in 1998 and 2001, respectively, and the Ph.D. degree from the National University of Singapore (NUS), Singapore.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering and the Director of the Interactive Multimedia Laboratory, NUS. His research interests include augmented and virtual reality, computer vision, and multimodal human–computer interaction.

Stefan Winkler received the M.Sc. degree in electrical engineering from the Vienna University of Technology, Vienna, Austria, and the Ph.D. degree from the Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland.

He was a Chief Scientist with Genista Corporation, which he cofounded in 2001. He was an Assistant Professor with the National University of Singapore, Singapore, and also with the University of Lausanne, Lausanne. He is currently a Principal Technologist with Symmetricom's Quality of Experience Assurance Division, San Jose, CA. He has published more than 50 papers and is the author of the book *Digital Video Quality*. His interests include visual perception, media quality, computer vision, and human–computer interaction.