Novel Interface for First Person Shooting Games on PDAs

Chen Wei
Collaborative Visual Computing Lab
Computer Science Department
University of Cape Town
cwei@cs.uct.ac.za

Gary Marsden
Collaborative Visual Computing Lab
Computer Science Department
University of Cape Town
gaz@cs.uct.ac.za

James Gain
Collaborative Visual Computing Lab
Computer Science Department
University of Cape Town
jgain@cs.uct.ac.za

ABSTRACT
This paper explores novel interfaces for First Person Shooting (FPS) games on Personal Digital Assistant (PDA) devices. We describe a new approach inspired by a study of the interaction patterns used in desktop FPS games. Intelligent gesture recognition, based on these patterns, is used to create an optimal implementation of basic game functions (i.e., jump, shoot, walk forward). This new interaction system is evaluated through a prototype 3D FPS game. We believe the newly designed interface more adequately leverages the interaction capabilities of current PDAs, to better solve the problem of rapidly and accurately executing a large number of gaming commands.

Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: User Centered Design

General Terms
Design

Keywords
Interaction techniques, gesture recognition, first person shooting games, 3D mobile games

1. Introduction
Although considerably less powerful than contemporary desktop systems, the ability of mobile hardware has improved to the point where PDA devices such as the Dell x51v are now capable of rendering simple three-dimensional (3D) environments. Whilst the effects of Moore's Law means that this improvement in rendering capability will continue to increase, attention must be give to the inherent difficulties in interacting with a small device when playing a game as interaction heavy as a First Person Shooter. Assuming a current PDA or smart-phone, executing commands in gaming is restricted to the stylus pen and a limited set of hardware buttons. Other problems we observed in the preliminary stages of this research included user fatigue due to holding the PDA, difficulties in coordination when performing multiple commands simultaneously and restricted camera movement. These problems present an interesting challenge, not just to FPS players, but also serve as an extreme test case for new interaction techniques for controlling and interacting with complex information on a PDA screen.

OZCHI 2008 Proceedings ISBN: 0-9803063-4-5

Our goal is to uncover how best to take advantage of the baseline interaction facilities found on current PDAs, namely stylus input and hardware-button pressing. We wish to enable players to manage a large number of commands and execute crucial action commands simultaneously, so as to enhance the FPS game playability.

To investigate new interaction techniques, a 3D FPS game engine prototype, *InteractionPro*, written in C#, was created from scratch. This prototype is able to offer a comprehensive investigation of interaction through the help of the .net compact framework 2.0 and Mobile DirectX. By creating this engine, we were able to evaluate our technique empirically and show that it has advantages over current systems.

2. Background of Mobile Game Interaction
The stylus pen is the standard input device on most Windows-based PDA devices. The stylus, together with a hand-writing recognition system, provides a pointing and clicking interface and solves the problem of effective text input. With the help of the four hotkeys, four directional buttons and one confirm button, users are able to manipulate the PDA's normal applications smoothly. Although the convenience and efficiency of this input system cannot compare to that of a desktop PC, it provides users with an appropriate way of completing common tasks such as entering text, surfing the web and taking notes.

It is this hardware configuration that we will use as our base-line implementation, namely: a touch-screen, four soft-buttons and one confirm button. Ever since the Palm Pilot was released, this layout for a PDA has been followed by a majority of manufacturers (e.g., HP, Dell, Fujitsu, etc.). Whilst there are many devices that do not meet this standard (most cellular handsets, for example, lack a touch screen) and other gaming devices surpass this standard (the Nintendo DS has a custom touch pad separate from the main screen) we believe the PDA (as described above) to be the 'modal' configuration.

Game playing is very different from operating most GUI-based applications. Many games have complex interactions which demand a high degree of collaboration between both keyboard and mouse. The initial investigation of this research showed us that it is very difficult to play these games using only the buttons and stylus pen on a PDA. For instance, in playing GeoRally Ex[6] on a PDA device, the user cannot make the race car accelerate and turn left at the same time, which violates the intention of 'accelerating while turning'. This is because the user is not able to simultaneously activate two directional buttons.

The problem of interaction in FPS games is much worse than car racing games. It involves many more commands and more complex user-interface coordination. A new interaction design is proposed in this paper to solve these problems.

To date, we have found no other research looking at the problems of interacting with similar games on PDAs. There has been interest in how to render 3D games and virtual environments on PDA devices [3,4,5], but none of this work analyses the choices made in terms of the mechanics by which the user interacts with that environment.

3. Methodology

The methodology adopted for this research follows a typical interaction design lifecycle starting with an observation phase followed by formative evaluations of prototypes and then a final summative evaluation of a high-fidelity prototype [7].

- **Observation**: To understand the users, background research on traditional interaction practices of FPS games on desktop PCs was conducted. An application, named InteractionLog, was built to log the patterns of behavior when playing games on a desktop machine. Our goal is to make the sure the interface supports the most common actions in the most efficient way.
- **Prototyping**: In this research, a computer-based low-fidelity prototype and a fully-functional prototype [8] were implemented as the best combination to evaluate the usability of the new interaction design in a low cost approach. The low-fidelity prototype is a Flash application while the latter one is a real PDA 3D application, namely, InteractionPro.
- **Evaluation**: There were two aspects to the final evaluation. The first determined gamers' satisfaction with the interface based on heuristics for mobile game design. The second was a quantitative investigation into the interaction logs generated by users of the prototype.

4. Investigation and Design

When it comes to the usability evaluation of games on mobile devices, Korhonen and Kovisto [10], recommend that the function keys (the buttons that control the specific game commands) should be consistent and follow standard conventions (e.g., the keys 'W','S','A' and 'D' are conventionally used for moving 'Forward', 'Backward', 'Left' and 'Right', respectively). However, for FPS games there are no "standard conventions" on a PDA. Therefore, the first goal of this research is the development of such conventions.

In order to develop conventions, we examined users' behavior whilst playing desktop FPS games, and then effective interaction conventions were developed which best support those behaviors.

To examine usage of the keyboard and mouse (these are the only input systems in FPS games on Desktop PCs), these questions were asked:

1. What keys and mouse buttons are most frequently used when playing a game?
2. How often are these keys and buttons used during the game?
3. How are these keys and buttons used together?
4. What is the relationship of the actions executed by these keys and buttons?

To answer these questions by manual observation is impossible, therefore custom logging software – InteractionLog – was built to capture the results precisely and objectively.

4.1 InteractionLog

InteractionLog is a .net Windows application which logs interaction events of the keyboard (key presses) and mouse (button clicks and movement rate) in the background while the player is playing any FPS game. All these user interaction events are recorded in an XML file which is visualized; this visualization can be performed instantaneously while the game is being played or offline, to reduce lag on the game and for future reference. The visualization, such as in Figure 1, helps with the analysis and understanding of the relationships between different interaction events.

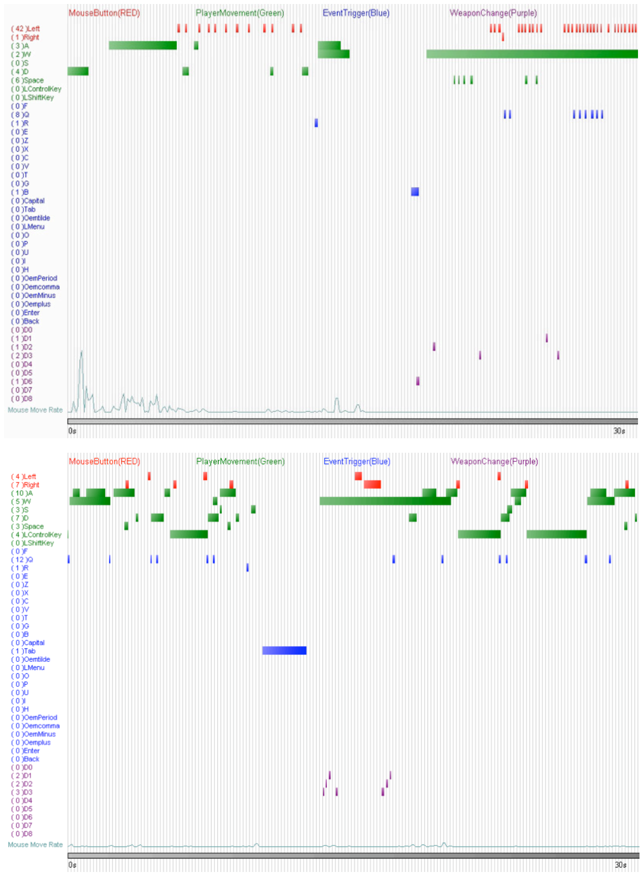


Figure 1. Visualizations of 30 second logs from a Counter Strike player. Avatar moves are shown in green, MouseButton events in red, EventTriggers in blue and WeaponChanges in purple. The upper plot shows assault rifle usage, while the lower one shows sniper rifle usage.

Counter-Strike 1.5 [12] was the FPS game chosen for the test. It is one of the most popular FPS games ever created, and most FPS game players are familiar with it. Ten volunteer Counter Strike players with varying degrees of expertise were contacted and had the InteractionLog application installed on their machines. After one week of play, their log files were sent back for analysis.

Action commands vary from FPS game to game, but basic functions are common across most games. These common interaction actions are divided into four main types according to their characteristics:

a) Avatar movement

b) Camera movement

c) Aiming and Firing

d) Advanced Function Commands (e.g., triggering events, changing weapons, etc.)

These actions are highlighted in different colors in the visualized plot. For instance, avatar moves are shown in green, camera moves (mouse movement rate) in cadet blue, MouseButton events in red, EventTriggers in blue and WeaponChanges in purple. Taking Figure 1 as an example of the log, there are many time sections (episodes) where the red, green and cadet blue segments overlap. This means players often engage in the first three activities simultaneously (moving the avatar, moving the camera and firing). This situation frequently occurs when the player intends to dodge the attack from opponents and shoot back. Whilst the choice of weapon affects the exact pattern of usage (e.g., an assault rifle is used in a completely different way to a sniper rifle), the actions of moving, aiming and firing still tend to be executed together. Clearly, the PDA solution must support moving, firing and camera actions that can be executed simultaneously without interference with each other.

The two most common activities observed were those of firing and moving. Again, the plot in Figure 1 shows 42 shots fired and 15 separate avatar movements inside the 30 second period. Solutions must make such actions rapidly accessible.

Finally, the advanced function commands, such as weapon changing, happened in isolation, which means they can be supported in a less direct fashion.

4.2 Observed Play on PDA

In order to gain some insight into how complex it was to interact with an FPS game on a PDA, seven players were recruited to play DoomGL. All the recruits had played FPS games on the desktop, but none had played it on the PDA. A number of serious problems were observed:

- Not all commands were accessible. This is because of the limitation of the physical buttons embedded on the PDA device. Despite the directional buttons, the hotkeys on the PDA are assigned to the 'Game Menu', 'Map View', 'Status Bar' and 'Door Open' actions. The available commands are insufficient for a complete FPS game. For example, there are 46 commands in Counter-Strike 1.5.
- Users did indeed struggle to press the fire button whilst moving the avatar. The form-factor of the device required users to employ a single thumb for moving and firing, meaning that it was not possible to execute both actions simultaneously. This problem is serious since combined 'Dodging and Firing' are frequently necessary.
- Camera movement is overly restricted by the fixed size of the PDA screen. Unlike using a mouse on a large desk, the stylus movement range on the PDA screen is fixed and very limited. The fixed movement rate makes it difficult to balance between micro-adjustments (e.g., aiming at the target through tiny adjustments) and big-turns (e.g., to change the camera horizontal angle by 100 degrees).
- Most users felt that their left hand was exhausted after playing DoomGL since they were holding the entire weight of the PDA in that hand and the left thumb had to stabilize and balance the device. In DoomGL, the PDA has to be held in a landscape fashion, with the left hand solely supporting the whole weight of the device from the edge. This is more tiring than holding the PDA in a portrait orientation as it severely affects the stability of holding the PDA and fatigues the hand, wrist and arm.

4.3 New Interaction Design

Interaction design in mobile games is very different from a normal application interface design. Korhonen and Kovisto's [10] guidelines for heuristic evaluation on "Game Usability" give guidance on what is desirable from a game interface. Specifically:

"Navigation is consistent, logical and minimalist"

"Game controls are convenient and flexible"

With these goals in mind, we developed "Gesture Interaction" a new interface which uses a stroke recognition system combined with the optimum implementations of basic game functions.

The term "Gesture" refers to the fact that many of the functions are now available by recognizing gestures made with the stylus on the screen. By adopting a gesture-based system, we can address the "convenient and flexible" goals. The system is "convenient" in that the screen is there all the time (you do not need to access commands by invoking a menu or command box), it is also "flexible" in that there are, potentially, an infinite number of gestures that may be recognized. Whether or not these gestures are "consistent, logical and minimalist" will need to be determined through experimentation.

Figure 4 shows the different command strokes we adopted for the first prototype. The bold dot on each stroke represents where it starts from. These initial gestures were the result of a tradeoff between suggestions from gamers and strokes that could be recognized rapidly and unambiguously. Other functions were provided as follows:

- The avatar's movement is controlled by the Directional Pad (D-Pad) which follows FPS game convention.
- The camera view is controlled by the stylus pen. The view from the avatar follows the movement of the stylus pen (e.g., the view turns to the left when the stylus pen slides to the left on the screen and looks up when the stylus slides upward). It guides the direction the avatar faces; turning or steering the avatar's movement. The avatar's pitch range for looking up or down is limited between 1.5 and -1.5 radians. This range prevents the confusion caused by the avatar turning upside down.

While the 'Gesture Enable' button (the 'Return' button) is being pressed down, strokes can be drawn anywhere on the screen. The recognition of the stroke and the execution of the commands will occur as soon as the stylus pen is removed from the screen in the 'Gesture Enable' mode.

Shooting is now also invoked through the stylus. From our observations of PDA DoomGL, we created two separate shooting modes, namely: 'AutoAiming & Shooting' and 'Manual Shoot'. These work as follows:

- 'Manual Shoot' is triggered by double tapping the stylus anywhere on the screen, causing the avatar to start shooting toward the center of the screen.
- 'AutoAim & Shooting' is engaged when the stylus pen is tapped on an enemy drone in the scene and persists shooting for as long as the stylus is kept on the touch-screen. The center of the camera is panned to the tapped position. As soon as the panning motion is finished, shooting starts.

Only one function command button, the 'Return' button, is used in the new interaction style. The remaining four hotkeys can be used for other function commands, which greatly enhances the efficiency of this interaction style. However, these buttons are not assigned any commands forcing users in the final evaluation to use the 'Gesture Interaction' method. In some ways, this biases the evaluation against the proposed system as it is clearly desirable to map functions to buttons if they are available. However, we felt that greater insight would be gained into which were the crucial functions by insisting that all interactions (other than avatar movement) rely on gesture input.

4.4 Flash Prototype

A computer-based low-fidelity [8] prototype which presents the new interaction design was built before implementing the fully interactive PDA high-fidelity [8] prototype. This prototype was built using Flash on the desktop. The aim of this prototype was to validate the main concept of the new interface through user comments. It was also intended to evaluate other major usability issues such as unclear terminology, graphical representation issues and appropriate positioning of interface elements. Other issues like physical handling and operation, comparison with other similar products and performance-related issues were evaluated by the final fully functional high-fidelity prototype.

Due to the functional limitations of Flash, the Flash prototype simulates the interface and the newly designed interaction method only; not the game. It is not fully 3D - movement is simulated by panning.

Seven players who attended the test of Doom-PDA were called again to evaluate the prototype informally. (Obviously direct comparisons with Doom could not be made due to the limited nature of the prototype.) They were introduced to this Flash prototype and explored it at their own pace. Individual semi-structured interviews were conducted with the users after exposure to the prototype. All of the participants were pleased with the main conception of the new interaction design, which are:

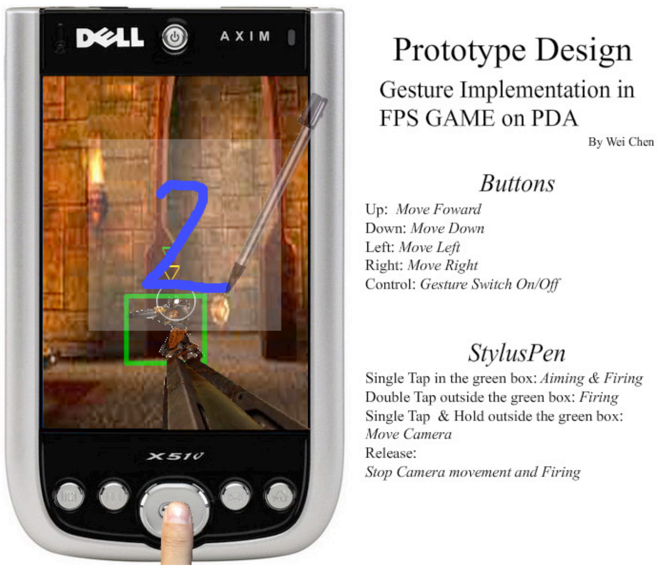

DELL
AXIM
x51v
Prototype Design
Gesture Implementation in FPS GAME on PDA
By Wei Chen
Buttons
Up: Move Foward
Down: Move Down
Left: Move Left
Right: Move Right
Control: Gesture Switch On/Off
StylusPen
Single Tap in the green box: Aiming & Firing
Double Tap outside the green box: Firing
Single Tap & Hold outside the green box: Move Camera
Release: Stop Camera movement and Firing

Figure 2. Flash Prototype Design

(a) Separating the functions 'Firing' and 'Moving' between two hands.

(b) Optimizing the function of 'Aiming and Shooting'.

(c) Triggering commands through 'Drawing Strokes'.

Encouraged by this informal feedback, we set about creating a high-fidelity, highly-interactive prototype to garner more realistic formal feedback.

4.5 PDA Prototype

The final high-fidelity prototype is a 3D FPS game called *InteractionPro*, which is implemented specifically for the Dell x51v. It was developed in Visual Studio .net 2005 (C#) Compact Framework v2. It uses Mobile Direct3D as the 3D API.

The aim of this prototype is to evaluate the novel interface design of FPS games on PDA devices. Our approach is to compare the results of two different interaction styles in a series of game tasks. These two different interaction modes are:

Interaction A (Doom Mode). This mode uses a traditional button pressing interaction style as in Figure 3.

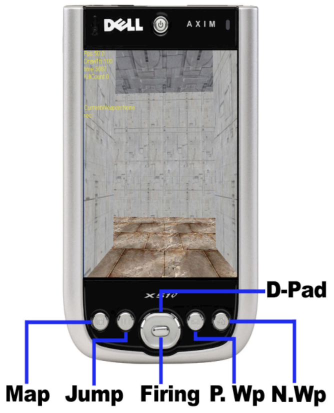


Figure 3. The Function Distribution of Interaction A

Interaction B (Gesture Mode). This mode uses the newly designed interaction style which involves 'Stroke Drawing', 'Stroke Recognition' and an optimized set of commands as in Figure 4.

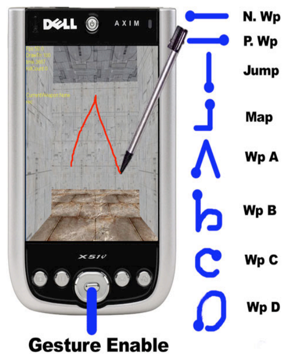


Figure 4. Interaction B (Gesture Mode)

Once the application launches, users can choose one of the two interaction styles. The selected style will be used in all stages of the game (a user cannot switch between the styles once the game starts).

There are in total four stages in this application. These four stages are designed to test different interaction aspects, which are 'Moving and Jumping', 'Shooting', 'Moving and Shooting' and 'Function Execution'. These are the four most common activities observed in game play and are discussed in detail below. All the testers' movements whilst running the application were recorded into an XML file. The data recorded includes the time the user spent on the different stages, the number of times each button was pressed, how long each key was held and the mouse movement rate (which logs the pixel distance (pd) of the stylus pen on the touch-screen between every 50ms – pd/50ms).

The mini 3D game engine involved in this application was created from scratch. Unlike traditional game engines which try to make the game attractive and entertaining, the aim of this mini 3D game engine is to help investigate the new interaction system in a fast and effective way. This required unique functionality, such as the action logging described above, which is not found in other game engines.

5. Expert Evaluation

5.1. Experiment Implementation

Ten people evaluated the system. These were all IT literate students, two of whom had never played an FPS before. The experiment was run as a within-group study, where the order of exposure to systems was balanced. There were seven sessions during each experiment, which took around one hour on average. These sessions were:

1. Introducing the experiment and the interaction styles.

2. Running the application as a practice round using the first interaction style to familiarize the tester with the application and the interaction. Each participant was given a simple task to complete and were able to take as long as they liked in order to complete the task successfully.

3. Running the application as a formal evaluation round using the same interaction style. In this stage, participants were required to complete four different scenarios.

4. Running the application as a practice round using the alternative interaction style. Again, participants could take as long as they wished to familiarize themselves.

5. Running the application as a formal evaluation round using the alternative interaction style. Again, the four scenarios were identical to those in stage 3.

6. Answering the questionnaire. There is no standard questionnaire for evaluating mobile games. Instead we adapted the relevant mobile game design heuristics of Korhonen and Koivisto [10]. For example, guideline 8 is – "Game controls should be convenient and flexible". We then created a question asking the user to rate the interaction styles relatively in terms of convenience and flexibility.

7. Semi-structured and critical-incident interviews with the user.

5.2. Experiment Results

Three categories of data were collected from the experiments:

1. The logs containing all the data of the interaction events during the experimentation.

2. The questionnaires representing the subjective feelings of players towards the new interaction.

3. The text of the interviews with the testers after the experiments.

The information of interaction events was compiled to an XML file and imported to a database. A visualization application called *InteractionAnalysor* was built to draw graphs of these results for better understanding and investigation of the new interaction style.

5.3 Individual Stages

Stage1: Moving and Jumping

In the first stage, the most basic actions in FPS games, 'Moving' and 'Looking & Jumping', are implemented. The aim of this stage is to test how the stylus mechanism handles these basic functions. The task of this stage is to walk through a long aisle by following the red arrows indicated on the wall, jump over two bumps and arrive at the terminus as soon as possible. As Figure 5 shows, the aisle includes 90 degree turns (left/right), 180 degree turns (left/right) and 'W' bends. There are also two bumps in the route that require the testers to execute 'Jump' commands.

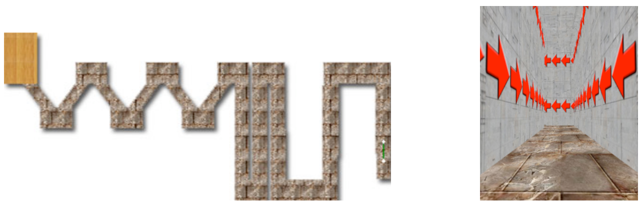


Figure 5. The map of the stage 1 (left) and the First Person View in the stage (Right)

As expected, testers spent more time using gesture input than Doom-style interaction because it takes longer to draw strokes than press buttons. But this slight delay did not affect the sense of presence [13] of the testers according to the questionnaire. The result of the questionnaire shows that testers prefer 'Gesture-mode' to the 'Doom-mode' in triggering the 'Jump' command in this stage. Both scores in this question are fairly low, indicating testers did not find it intuitive to trigger 'Jumping' in this stage. See Figure 6.

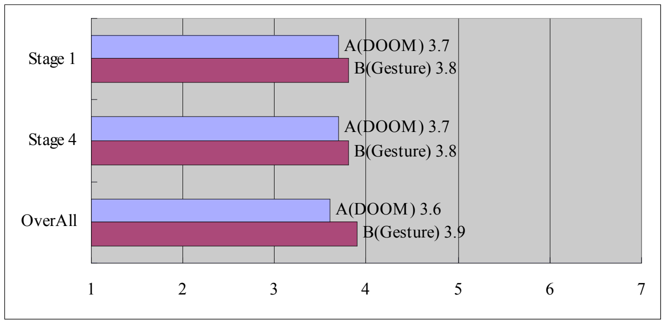


Figure 6. The score of satisfied with 'Jumping' in all stages

As 'Jumping' is frequently used in combination with other movement commands, such as walking and looking, testers spending a long time practicing the combination.

In Doom-mode, the tester executes a 'Jump' by pressing the button left of the D-Pad. Once the avatar is in the air, the tester has to move the left thumb from the 'Jump' button to the D-Pad immediately to press the 'Forward' button before the avatar falls back to the ground. Some of the testers withdrew their stylus pen from the PDA's screen so as to use their index finger to press 'Move Forward' on the D-pad. They re-engaged the stylus after the 'Jump Forward' motion was finished. During the interviews after the experiments, users explained that this way of 'Jumping Forward' is a habit carried over from the desktop. This habit is hard to change in a short period. This mechanism of 'Jumping' is uncomfortable to the users in Doom-style interaction.

In Interaction 'B' (Gesture), the way to execute 'Jumping' is by holding the middle button ('Return') down, then drawing a straight upward stroke anywhere on the screen. The 'Return' button must be released once the avatar is in the air, so as to press the 'Forward' button to jump forward instead of jumping straight up. Testers were not familiar with this new mechanism. Some of them became irritated after repeatedly failing to execute the 'Jump' command.

Figure 7 shows that in 'Gesture' mode, it took on average eight seconds more to complete the stage than it took in 'Doom' mode. This is understandable due to the extra 'Stroke Drawing' motion the 'Gesture' mode has. But the main concern of the evaluation of a game is the user's subjective feeling. Users showed similar levels of satisfaction with 'Jumping' in 'both modes (Interaction B scoring marginally higher).

Stage	Interaction	Respawns	Total Time(s)	Average of Mouse Rate(pd/50ms)
1	Doom(A)		34.52	3.65
	Gesture(B)		42.45	3.49
2	Doom (A)		32.21	4.45
	Gesture (B)		42.12	4.46
3	Doom (A)	2.25	44.42	4.78
	Gesture (B)	1	37.41	3.66
4	Doom (A)		126.31	8.08
	Gesture (B)		187.62	9.96

Figure 7. Table of the Log Results

Although the 'Gesture' mode is slightly better than the 'Doom' mode, the score clearly shows that the users were not happy with 'Jumping' in all stages.

We suspect the main reason behind the users' dislike of the 'Jumping' command in 'Gesture' mode is because of the lack of practice and unfamiliarity with the way it is performed: 'Jump' is the only command that needs 'stroke drawing' in stage 1

Some of the testers blamed their failure in executing the command on the stroke recognition system. However, the system's logs showed that the stroke recognition system was indeed working correctly but the testers were failing to execute the command in the correct way: some of them released the middle button before finishing strokes whilst some of them removed the stylus pen from the screen and released the middle button at the same time. All of these testers had trouble coordinating the rhythm of using the middle button and stylus pen together to accomplish the 'Jumping' action.

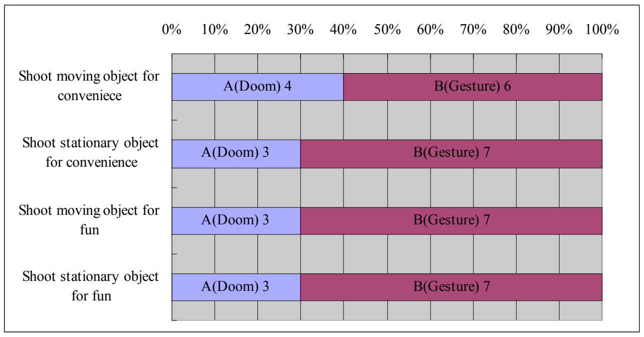
Of course, it could be that these results are simply reflecting a lack of familiarity with a new interaction method and that these scores would improve over time. On the other hand, this data provides strong motivation for implementing 'Jump' using a hardware button, should the target device have one available.

Stage2: Stationary Shooting

A 'Shooting' action consists of 'Camera View Changing', 'Aiming Adjustment' and 'Firing'. The goal of this stage is to compare these actions, which are very different in the two interaction styles. To remove confounding factors and focus on these basic actions the avatar remains stationary in this stage. Testers were expected to be very familiar with these actions using the traditional 'Button Pressing' style. In this style, 'Firing' is triggered by pressing the 'Firing' button and 'Aiming and 'Look Around' are handled by the stylus pen, which is similar to the use of the mouse on desktop PCs. In the new interaction style, these three actions are coordinated only by the stylus pen.

In this stage, there are ten 'Drones' (white cubes) randomly floating around in front of the avatar. Once one of these 'Drones' gets shot, it respawns in another randomly generated position in front of the avatar. Users have to shoot twenty drones to accomplish this stage.

All the testers accomplished this stage easily. Although the new shooting style takes longer than using the Doom game style, as is shown in Figure 7, it offers much more fun and convenience. Results from the questionnaire and interview confirm this – see Figure 8 below.



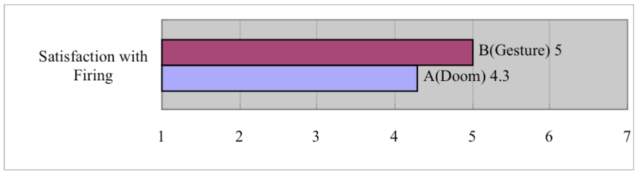


Figure 8. The scores in Stage 2. The upper graph reflects a binary preference choice; for example the first line shows that 4 participants preferred the Doom-style, whilst 6 preferred the gesture system. The bottom graph represents an average of a Lickert scale used to rate satisfaction level

The testers experienced the new shooting style 'Auto Aiming & Shooting' in this stage. Some of them verbally expressed that they enjoyed this new shooting mechanism. The average score for satisfaction with this shooting method is 5.

Stage3: Moving & Shooting

Situations in which players need to dodge and shoot simultaneously happen frequently in most FPS games. The design of Stage 3 is intended to simulate this kind of situation. The interaction actions taken in this situation are very intensive and good coordination between 'Moving', 'Aiming' and 'Firing' are essential. The main goal of this stage is to test the performance of the new interaction system is such intensive situations.

This stage consists of a closed cuboid room, which is shown in Figure 9. The room is divided into two sections by a wall with a hole in the middle. There are in total six drones (white cubes) in the big section. There are four stationary drones at the corners and two drones floating around randomly. The red cube in the plot represents the starting position of the avatar. Once the avatar starts moving, the player is not allowed to stop for more than two seconds, otherwise, the player will be respawned at the starting position. The wall in this room blocks the avatar's direct view of the drones from the starting position. This requires that the user move, changing this stage from a 'Stationary Shooting' stage to a 'Moving and Shooting' stage.

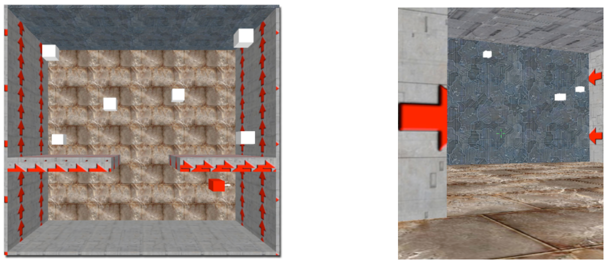


Figure 9. The top view of the stage3 (Left) and its view (right)

This is the most challenging phase in the entire application. Both of the participants who had no previous FPS experience failed to accomplish this stage. Even the experienced FPS testers had trouble finishing. However, the new interaction style shows better performance results than the original. Figure 7 indicates that using the 'Gesture' mode enables users to finish the stage in less time and with less mouse movement than the 'Doom' mode.

The most difficult part in this stage is the execution of the 'Shooting' command. Using this function in 'Doom' mode is especially inefficient, since users have to release the 'Moving' buttons first and then press the 'Return' button (all of these buttons are controlled by the left thumb), which means the avatar is only able to shoot in a stationary situation. This impedes performing the combination of dodging and counter attacking and severely impacts gameplay[10]. The new interaction design converts the execution of the 'Firing' command from the user's left hand to the right hand, which solves the problem of implementing the most commonly used commands, 'Moving' and 'Firing', simultaneously. The coordination of these functions has clearly been improved, and the testers were satisfied with these new changes, as shown in Figure 10.

Only a few testers tried the 'Auto Aiming & shooting' mechanism in this stage. This is because the habit of performing this shooting mechanism in an intense task had not had time to form; in intense game situations, they forget about this function. However, testers who used this method rated it above average.

Stage4: Function Execution

In addition to the basic 'Shooting' and 'Walking' actions, the interactions of advanced commands such as 'Weapon Changing', 'Events Handling', 'Menu View', 'Map View', etc., are also part of FPS gameplay. Although these commands are used less than the basic commands, according to the study in the InteractionLog section, they are necessary and crucial for enhancing the game usability and playability.

In order to prevent the results from being affected by advanced commands, task designs of previous stages focused only on the commands which were to be evaluated. The evaluation of the advanced function commands is different from those basic function commands.

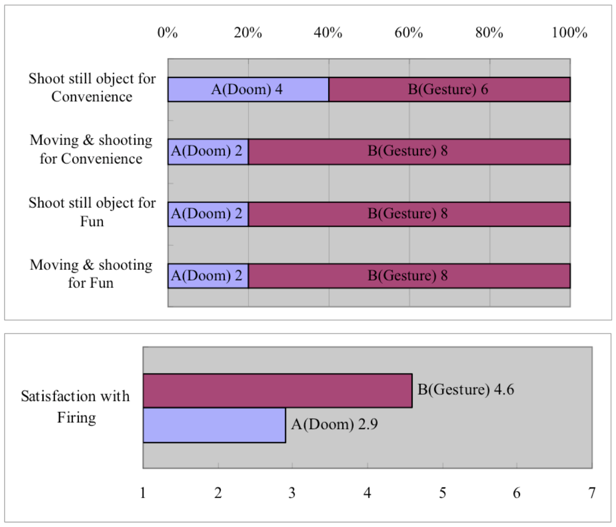


Figure 10 – Scores from Stage 3. Again, the upper graph is a summary of binary choice and the lower graph is an average of Lickert scores.

'Gesture' interaction style can execute many more function commands than the 'Doom' style due to the potentially unlimited patterns of strokes. On the other hand, the efficiency of button pressing is much faster. Irrespective of the speed or the number of commands the interaction style is capable of, the ultimate purpose of this evaluation in PDA games is to find out whether the players like and accept the new interaction style while playing the game. Therefore, the aim of this stage is to test the interactions performance of executing these functions in a proper FPS game context.

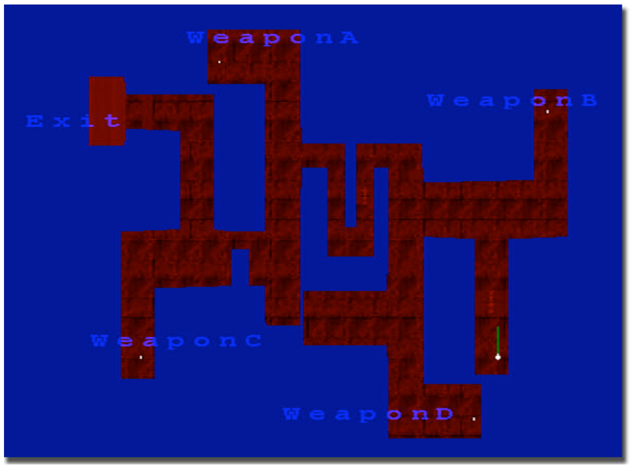


Figure 11. The top view of the maze

Based on this, a maze was created for this stage, as shown in Figure 11. Users have to find their way to the 'Exit' and kill all the drones (each must be killed with a specific different weapon). Users can get help information by executing the 'Map View' command to see the top view of the maze. The map shows the structure of the maze, the location of the 'Exit', 'Avatar', 'Drones' and the type of the weapon required for killing the drones.

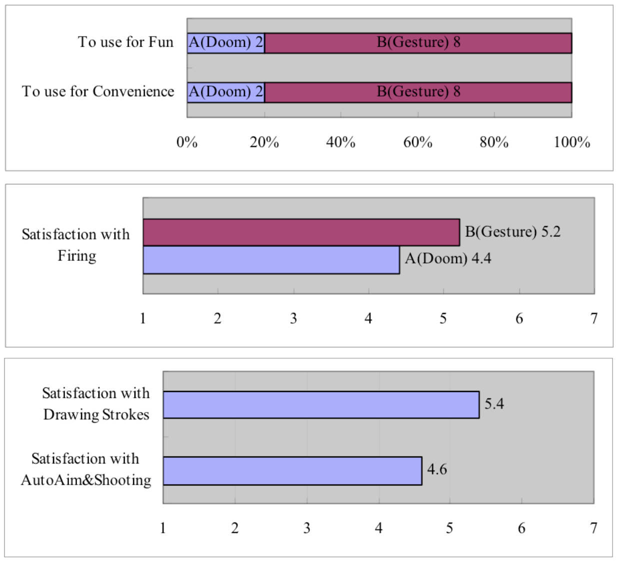


Figure 12. The scores in Stage 4. The top graph is a summary of binary preference choice and the bottom two are averages of Lickert scales.

Due to the fact that the number of physical buttons on the PDA x51v is very limited, only four advanced function commands are implemented in this stage, namely: 'Map View', 'Next Weapon', 'Previous Weapon' and 'Jump'. Besides this, in 'Gesture' mode, users can draw the strokes 'A','B','C' and 'D' in a graffiti style to specify which weapon to use directly instead of pressing the 'Next' or 'Previous' weapon multiple times.

Again, the new interaction mode took more time than the 'Doom' mode as shown in Figure 7. However, the satisfaction scores shown in Figure 12 would tend to favor the new system.

In addition to the unlimited commands, the entertainment and convenience offered by the new interaction system was rated as satisfactory. Another advantage of the new interaction design is shown in Figure 14. In interaction 'A' (Doom) which the upper diagram shows, the tester had to press the other four hotkeys around 40 times in this stage. These four hotkeys are also in the charge of the left thumb. Therefore, the left thumb has to take care of nine buttons in total, which is double the number in 'Gesture' mode. This unbalanced command assignment confused the testers severely. Users sometimes triggered undesired commands by mistake in Doom-mode.

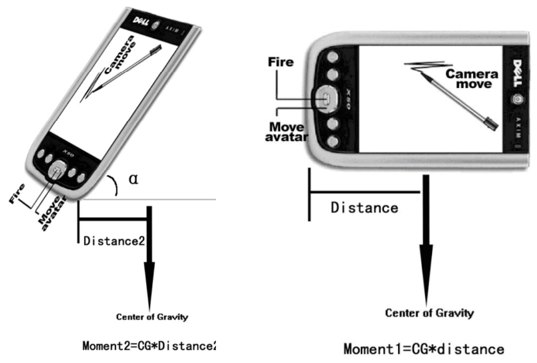


Figure 13: The force exerted on the user's hand is much greater in the landscape-oriented Doom than in our portrait-oriented system.

A further problem with the thumb being in charge of the nine buttons is that it affects how comfortably the PDA can be held. The left palm supports the weight of the device, with the left thumb controlling the balance and grasping of the device. The less movement and jumps the left thumb has to make between different buttons, the more stable and comfortable holding the device becomes. With the lower mouse movement rate afforded by the 'Gesture' mode, users did not feel fatigue from holding the PDA device at all, even though they had to hold it for longer. An explanation for this is shown in Figure 13.

6. Conclusion

A summary of the results are shown in Figure 15. Overall, there was no significant difference (p=0.148) between preference scores for each system. The only significant result (p=0.033) was in the new firing mechanism we developed, which players rated as increasing playability.

So overall the experiment results show that the new interface is certainly no worse than existing systems and, in the case of firing, is capable of making FPS games on PDA devices more practical and enjoyable than current implementations. Although the user spends more time using 'Gesture Interaction' than current interaction styles, the new style solves major problems including function limitation, user fatigue due to holding the PDA, difficulties in coordination when performing multiple commands simultaneously and restricted camera movement.

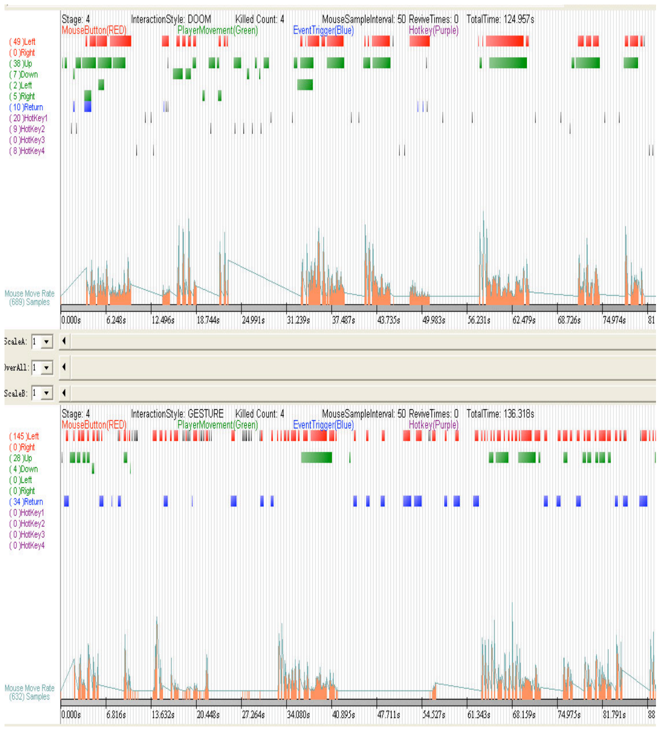


Figure 14. InteractionAnalysor which compare two interaction mode in an intuitive way. It shows activities of the interaction events in a certain period of time.

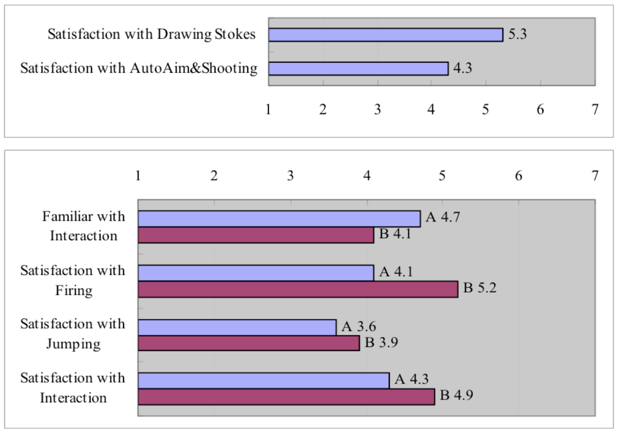


Figure 15. The overall results

Our interface does have the following advantages over current systems:

- Distribution of crucial commands to both hands creates good performance which is especially obvious in intense battle situations.
- The new interaction system is easy to learn. All the testers picked it up quickly. None of them needed to look at the stroke instructions after running the application for a short time.

There is still work to be done, however, and it is clear that some functions, such as 'Jumping', still need to be improved. We believe that a sensible balance between gesture recognition and hardware buttons should support this and provide even greater satisfaction to game players.

In section 4, we stated that there were, as yet, no standard conventions for the controls on PDA-based FPS games. Although we do not believe our research to be mature enough to be adopted as a standard, it does point the way to what that standard might look like – it will almost certainly incorporate some form of gesture recognition and automatic firing system. We hope that other researchers will be able to build on our work to create a canonical set of gestures and a firing system that strikes an engaging balance between automation and user controls.

7. REFERENCES

[1] Quake3CE. http://www.bit-tech.net/news/2005/11/11/q3ce_powervr

[2] DoomGL. http://www.doomworld.com/doomgl/doomgl.htm

[3] Apsman, W., Woodward, C. ,Hakkarainen, M., Honkamaa P., and Hyakka, J. (2004) *Augmented reality with large 3D models on a PDA: implementation, performance and use experiences*. In Proc. SIGGRAPH 2004, Singapore. 344-351.

[4] Marsden, G. and Tip, N. (2005) *Navigation control for mobile virtual environments.* In Proc. MobileHCI 2005, Salzburg, Austria. 279-282.

[5] Webber, M., Pfieffer, T., and Jung, B. (2005) *Pr@senZ - P@CE: mobile interaction with virtual reality.* In Proc. MobileHCI 2005, Salzburg, Austria. 351-352.

[6] GeoRally EX. http://www.ionfx.com/product_EX.php

[7] Preece, J., Sharpe, H. & Rogers, Y. (2007) *Interaction Design*. John Wiley and Sons.

[8] Lim, Y., P, A., Periyasami, S. and Aneja, S. (2006) *Comparative analysis of high-and low-fidelity prototypes for more valid usability evaluations of mobile devices.* In Proc. NordiCHI 2006, 14-18 October 2006. 291-300

[9] Nielsen, J. and Mark, R.L. (Eds.). (1994) *Heuristic evaluation*. In 'Usability Inspection Methods', New York: John Wiley & Sons.

[10] Korhonen, H. and Koivisto, E. (2006) *Playability Heuristics for Mobile Games will be held through the Expert Evaluation.* In Proc. MobileHCI 2006, September 12-15, 2006, Helsinki, Finland. ACM Press. 9-16.

[11] Federoff M. (2002) *Heuristics and Usability Guidelines for the Creation and Evaluation of FUN in Video Games.* Thesis at the University Graduate School of Indiana University, Dec 2002.

[12] Counter Strike 1.5. http://www.steamgames.com/v/index.php?area=game&AppId=240&cc=USA

[13] Barfield, W., Zeltzer, D., Sheridan, T. & Slater, M. (1995) *Virtual Environments and Advanced Interface Design.* Oxford University Press, UK.