Obscura: A Mobile Game with Camera based Mechanics

Albert Quek  
Faculty of Computing and Informatics  
Multimedia University  
Cyberjaya, Selangor, Malaysia  
Email: quek.albert@mmu.edu.my

John See  
Faculty of Computing and Informatics  
Multimedia University  
Cyberjaya, Selangor, Malaysia  
Email: johnsee@mmu.edu.my

Abstract—The introduction of Augmented Reality (AR) technology in games for mobile devices has greatly improved user experience on these devices. Moving on from basic multi-touch interactions, built-in camera and gyroscope functions in mobile devices can also be harnessed to create a different interaction experience. This concept has not been fully explored or realized in the form of a game. In this paper, we introduced camera based mechanics to a new AR mobile game to create a gaming environment that immerses mobile devices with their physical surroundings. The mobile game "Obscura" was implemented with Unity3D, with a wide range of features: the use of camera and gyroscope to create movements, artificial superimposition of enemy creatures, and multi-touch interactions to create the enemy-capturing mechanics. Visual, audio and haptic feedback were also provided to the player to heighten gaming experience.

By qualitative user evaluation on a target interest group, all participants responded positively to the gameplay of Obscura. An accomplished score of 5.3 out of 6 for overall game experience was achieved.

Keywords—Camera based mechanics, Mobile Game, Augmented reality game

I. INTRODUCTION

A recent report showed a significant increase in the usage of smartphones with the number of smartphone users estimated to surpass 2 billion in 2016 [1]. Mobile gaming is no longer new as many mobile games can be easily downloadable online from "app markets" such as Google Play and AppStore. Most mobile games are free; therefore this attracts mobile users to regularly check for the latest games from the app market. The video game industry was generally dominated by console and PC gaming in the early and late 1990s but mobile gaming has come to the forefront in recent years. According to Ulyana Chernyak [2], about 56% of gamers are using mobile devices to play games, a clear indication that mobile gaming is the current trend and will continue growing in the upcoming years.

The concept of Augmented Reality (AR) [3] involves superimposing digital assets in the application onto the real world environment, merging them into a single screen. This technology has been around for a long time and it is less popularly utilized for games. Nevertheless, with the appropriate use of AR, gaming styles and experience can be greatly improved [4]. Most mobile devices are now equipped with powerful processors and sensors that can handle AR processing. Nilsen et al. [5] hypothesized that AR is fast becoming less of a novelty and more of a utility.

Fig. 1. Reality-Virtuality (RV) Continuum

In the mixed reality (MR) continuum introduced by Paul Milgram [3] in Figure 1, real content stands at one end while virtual content occupies the other. In between these two ends are the concepts of Augmented Reality (virtual content added to real environment), and Augmented Virtuality (real content added to virtual environment). Basically, there are two categories of AR games—simple and complex. For simple AR games, devices such as ultra-mobile PCs (UMPCs) or smart phones with built-in features (for e.g. cameras, Global Positioning System (GPS) sensors, touch screens, touch pads, and etc.) are utilized. Standard interaction techniques used are such as touch screens, markers, mice, or built-in buttons. Developers can then design and develop their gameplay and levels according to these specific equipment. With the game being independent of a specific environment, gamers can allow their game to be as closely connected to their real local environment. The second type of AR game is more complex and event-based, in which it provides a better user experience compare to first type. Event-based games often require a team of experts to prepare the location (location-based system), set up the game (involved advanced sensors and devices), train the participants in hardware and interface mechanisms (sophisticated interactions), support and supervise the gameplay [6].

We look particularly into the gameplay of Ghostbusters: Paranormal Blast and iButterfly; both simple AR game applications that can be downloaded to the smartphone. Ghostbusters: Paranormal Blast’s [7] gameplay is fairly simple; player chooses a physical location, and uses weapon to attack the enemy (ghost) that appears in front of the player. The enemy will move around dodging from the player’s attacks, while sometimes hitting back at the player. There are health bars for both player and the enemy; the one who runs out of health points first loses the game. The gameplay is very repetitive and the GPS function in the game is minimally
utilized. iButterfly [8] is an AR mobile application that uses camera to capture butterflies flying on the device’s screen. iButterfly is a promotional application that allows the user to catch butterflies in exchange for product promotions in participating stores. Motion sensors are used to determine the movement of the player while searching for the butterfly. GPS is greatly used in this application to determine the current location of the user. Different locations will spawn different butterflies, which carry different promotions and rewards.

Meanwhile, a complex AR game, ARQuake [9] was developed as a first-person-perspective game allowing players to move freely in the real world. The player’s viewpoint is determined solely by the orientation and position of the player’s head using a Head Mounted Display (HMD). It is controlled using easy-to-understand real-life props (i.e. plastic gun) with some basic metaphorical representation. The level of the game resides in the campus environment at University of South Australia. In order for the player to walk around the world (campus), a complete model of all the buildings was created and stored as a Quake map. All buildings were first created as solid black objects so that they do not render to the display; but in real life, the actual buildings are present in the game to provide both visible and haptic feedback on their locations. Another example of a complex AR game, Battlefield Augmented Reality System (BARS) [10] is a complex AR military simulation application developed to simulate AR military operations in a real urban environment. The system consists of a wearable computer, a wireless network system and a tracked see-through HMD. The user’s perception of the environment is enhanced by superimposing images onto the user’s field-of-view. The graphics are registered (aligned) with the actual environment.

In this paper, we propose a new simple AR Game called “Obscura” that utilizes the smartphone’s camera, gyroscope and a set of touch-based interactions to create the gameplay with capture mechanics. Camera filters (grayscale, sepia and night vision) are used to create an eerie impression of the depicted real-life scene. The player’s aim is to capture creatures using the appropriate camera filter and capture mechanic. Vibration and audio feedback were also incorporated to improve user experience during the process of capturing enemy creatures (voiders). The rest of the paper are organized as follows: Section II introduces the game concept of Obscura, user interface design, gameplay mechanics, and other in-game components. Section III describes the implementations of gyroscope, camera and its filters, and collider functions of the game. Finally, Section IV presents user evaluation results while Section V concludes the paper.

II. GAME DESIGN

A. Game Concept

Obscura is an interactive AR camera-based mobile game implemented on Unity3D [11]. The player takes up the role of a researcher of strange paranormal creatures (enemies called “voiders”), who is embarking on a quest to search for hidden voiders in different dimensions, capturing them for scientific research.

The game level depicts the actual real environment where the player is in. More to that, the level environment is artificially distorted with different camera filters to simulate a dark, eerie environment. The player is required to find and capture enemies that can only be viewed through the special in-game camera. The player has to move the camera around to find the enemy (hidden away somewhere in the scene). When the enemy appears, a noise cue will be activated, prompting the player to get ready to locate and capture the enemy.

In the process of capturing the enemy, player has to apply different capturing mechanics to capture different types of enemies. For example, by performing multiple taps to the enemy, each correctly aimed tap at the enemy will reduce the Hit-Point (HP) of the enemy. Once the enemy’s HP is reduced to zero, the enemy will be successfully captured and added to the player’s diary. On the other hand, there is a limit to the camera usage. The camera battery will continue to deplete when the camera is activated. The depletion increases further when an enemy is in sight, hence it becomes more crucial to aim precisely. The player loses the game when the camera runs out of battery. The player is also provided with options to upgrade the camera features such as battery lifespan, filter type and focus level. Battery upgrade will prolong the camera battery life (effective against fast moving voiders), while the filter and focus upgrades will increase the chances of capturing higher-level (and rarer) type of enemies. To balance the game well, each upgrade requires a certain amount of money which is earned through the capturing of enemies.

Obscura was developed using Unity3D Professional [11] with C# as the main scripting language. Camera filter functions in Unity3D were utilized to create the desired screen effects. The “Scene” concept was used to represent the game level, where all “Game Objects” (textures, sprites, texts, etc.) reside within the “Scene” and are attached with scripts that determine their behavior.

B. User Interface

We designed the game interface in a simple and informative manner, as the screen size for the smartphone is relatively smaller than most AR-based games for PC. To avoid having a cluttered user interface (UI), we have decided to only include UI elements that are essential to the gameplay. The game was designed and tested with a 4.8-inch screen size. Figure 2 displays the in-game UI design, with its components described as follows:
- **Battery Life**: The battery life of the in-game camera that the player uses to capture enemy. Player will lose the game when the battery life reaches 0.

- **Player’s Health**: The player’s HP will be reduced when the enemy attacks the player. If the player successfully captures the enemy, the player will have the chance to increase their HP by 1.

- **Back Button**: This button quits the current game and goes back to the main menu screen.

- **In-Game Camera**: The player can select three types of camera: Grayscale, Sepia and Night Vision to find hidden enemies in the level.

- **Enemy Stats**: The HP of the enemy. The player has to perform the necessary capturing mechanics to reduce the enemy’s HP to 0.

- **Overall Stat**: The in-game message (bottom middle of screen) prompts the player on the current status of the game.

- **Crosshair**: The focus area of the camera, which the player uses to position the camera to capture the enemy.

### C. Gameplay Mechanics

The simplicity of the gameplay is the main focus of Obscura. The game introduces only one movement mechanic and three simple capturing mechanics. The capturing mechanics (multiple tap, hold and timed tap) are triggered by the different types of enemies. The player is required to aim correctly at the enemy, before performing the associated capturing mechanics. The gameplay mechanics are explained as follows:

- **Aim**: Player moves the camera and targets the enemy.

- **Multiple Tap**: Player to perform multiple taps on the enemy in order to reduce the enemy’s HP to 0.

- **Hold**: Player to perform hold on the enemy until the HP is reduced to 0.

- **Timed Tap**: Player to wait for the auto-focus indicator before performing the tap to reduce the enemy’s HP.

Figure 3 shows the three capturing mechanics in the game.

### D. Enemy Non-Player Character (NPC)

There are a total of 9 hidden enemies to be found and captured. These 9 enemies can be detected using the in-game camera under specific settings – filter type and focus level. This is possible by upgrading the camera filter and focus level function for the in-game camera. For every enemy captured, player has the chance to increase the player’s HP by 1. Table I shows the list of enemies (in order of rarity level), with their associated camera type, focus level and capturing mechanics.

### E. Shop and Diary

The game provides options for the player to upgrade the in-game camera’s filter lens to increase the chances of locating rarer types of enemies. The camera filters available in the game – Grayscale, Sepia and Night Vision, are shown in Figure 3. By default, the player will start the game on the Grayscale filter mode, while the other filters have to be purchased separately when there is sufficient money. Figure 4 displays the shop screen for upgrading purposes. The upgrading options in the shop are explained as follows:

- **Camera**: Three filters are available: Grayscale, Sepia and Night Vision. Different cameras will detect different type of enemies.

- **Battery Up**: This improves the battery life during the gameplay.

- **Focus Up**: This increases the possibility of finding rarer types of enemy. Depends on the camera filter.
Fig. 3. Capturing mechanics: Multiple Tap (left), Hold (middle) and Timed Tap (right). In-game camera filter types: Grayscale (left), Sepia (middle) and Night Vision (right).

For every enemy that is successfully captured by the player, it will be recorded in the diary. The player can access the diary in the main menu to check on the details of the captured enemies. See Figure 5 for a screenshot of the enemy diary.

III. GAME IMPLEMENTATIONS

Obscura was developed using Unity3D Pro with C# as the scripting language. The game uses the supported functionalities for gyroscope, camera (and its filters) and collider to implement the gameplay. The gyroscope is used to sense the orientation of the device by calculating its rotation. The camera functions as a viewport to the surroundings of the actual physical environment. The filter performs post-rendered effects on the views captured by the camera.

A. Gyroscope

The gyroscope function is implemented in Unity3D, where the current gyroscope’s rotational axis is first mapped to the game object’s rotational axis. It is then attached to the camera object, so that the player can control the rotation of the camera by simply moving the smartphone.

B. Camera and Filters

To implement the camera function, the view captured by the device’s camera is projected as the texture of a quad game object. The quad (named ”Screen”) is then positioned in front of the in-game camera viewport so that the player sees the ”Screen” no matter where it moves. Figure 6 shows the ”Screen” (white quad) and its position in front of the in-game camera. The Grayscale and Sepia filters can only be found in Unity3D Pro under the image effects [11] while the Night Vision effect can be found in the Unity Asset store [12] (see Figure 3 for the filters). The filters used are as follows;

- **Grayscale**: This effect changes colors into grayscale.
- **Sepia Tone**: This effect tints an image to resemble an old photograph.
- **Night Vision**: This effect brightens the whole image using a very distinct lime green color.

C. Collider

A `PlayerController` script is written to control the behaviour of the player. Actions like tapping the enemy, moving near and away from the enemy are all controlled by `PlayerController`. The `PlayerController` Game Object is attached by a `CapsuleCollider` as shown in Figure 6 (shown by the green lines). Therefore, this collider will determine how close the player is to the enemy.

All enemies also have colliders attached to them. There are two colliders on an enemy (see Figure 7) – `RangeCollider` (green sphere) and `BodyCollider` (green cube). The `RangeCollider` checks if the enemy is close to the player while the `BodyCollider` checks if the player has accurately tapped on the enemy. For instance, if `PlayerCollider` collides with
RangeCollider of the enemy, it signifies that the enemy is in range. Figure 8 shows a selection of other screenshots from the game, capturing mechanics, camera filters and enemy attacks in Obscura.

IV. USER EVALUATION

Obscura was created to introduce camera-based mechanics into a mobile-based simple AR game. Hence the target audience were smartphone users and casual game players. The game was tested on a group of 10 participants ranging from 18 to 20 years old. All participants were given 15 minutes to test the game. The participants were then asked to give a score on the following aspects: learning curve, gameplay experience, user interface, addictiveness and overall experience. Table II shows the average scores for different aspects as evaluated by the participants. The score ranges from 1 to 6, where 1 is poor and 6 is excellent.

In summary, all participants had a fairly good experience, as noted from the overall game experience score of 5.3 out of 6. There were also some interesting feedback (positive and negative) from the participants:

- **Positive Feedback:**
  - “Interesting game concept”
  - “Intuitive and interactive gameplay”
  - “Useful diary options to keep track of the captured enemies”

- **Negative Feedback:**
  - “Only 9 types of enemies in game”
  - “Repetitive gameplay, should include more gameplay variations”

V. CONCLUSION

Camera based mechanics is a new concept that has far-reaching potential in contemporary mobile gaming. In this paper, we present a unique mobile game called "Obscura", which uses camera based mechanics to generate a simple augmented reality experience. In the game, player can switch between different in-game camera (with grayscale, sepia and night vision filters) to find enemy creatures that are “hidden” in the physical world by means of augmentation. The implementation of capturing mechanics (multiple taps, hold and timed tap) further improves the gameplay experience of the game. In the future, we intend to incorporate location-based mechanisms into the game using global positioning system (GPS) information. This will allow specific types of enemies to be placed at different locations, thus improving the replayability of the game.

ACKNOWLEDGMENT

We would like to thank Ong Min Yen for realizing this new game concept and developing the game prototype.

REFERENCES


