American University of Sharjah

COE 487

Knight-em' All

Table of Contents

1. Game Description and Goal

The game is a spin-off of the typical arcade game Whack-A-Mole. The game setting will follow medieval aesthetics, and it will take place in an old kingdom with green scenery. The goal of the game is to hit the goblins and dragon characters as they appear while avoiding hitting the princess character. Each level will have a time limit and a score goal that the player must achieve to move on to the next level of the game. The levels will differ in the number of moles appearing at once and the time limit. If the player hits the princess more than three times or the timer runs out before achieving the score, the game is immediately over. Different characters award the player different points based on their rarity. In addition, the player can accumulate a combo to add a multiplier to the points rewarded by not missing hitting any characters in a sequence. If the player misses a character, one point is deducted. The game levels will get progressively more difficult by reducing the amount of time the characters appear from the hole before disappearing and the number of characters appearing at once.

2. Game Mechanics

2.1. Space

The game space is 3D, discrete, and has nine connected cells. The cells, in this case, represent the nine holes from which the characters will appear from. The space is discrete since the characters can only be positioned in one of these holes only, and the positioning within these cells is not significant. Given that the nine holes have adjacency, they are considered connected cells. Moreover, since the characters will be jumping up and down the holes, moving forward and backward, and sideways the game space is 3D.

2.2. Rules

There are two main rules that the players must follow to win the game and progress in the quest, which are:

- The player must achieve the minimum score required for the level within the time limit to progress to the next level
- The player must not accidentally hit the princess three or more times

2.3. Objects and Attributes

The first game object is the scoreboard. The scoreboard will keep track of the total score required to bypass the level, which is a static attribute. It will also store the user's current score and time passed, which are dynamic attributes. The scoreboard will also have a variable to keep track of the number of times the princess has been mistakenly hit to be able to stop the game after the third hit, this is also a dynamic attribute. Finally, the scoreboard will contain a combo variable which will act as a multiplier for the points received, and it can be accumulated by not missing any characters as they pop up; this is also a dynamic attribute.

The second game object is the characters. The characters can have multiple types, such as princesses, dragons, or goblins. Some characters should be whackable, and others should not be, which is a static attribute. Each character will have a different reward point depending on its rarity. Each character will also have a different animation when appearing and different special effects when it is whacked. All those attributes mentioned are static attributes. The character will also have a current position attribute to distinguish which hole it is in, which is a dynamic attribute.

The third game object is the mole holes. There will be nine holes in total, each one can be one of three states, either empty or occupied with a whackable character or occupied with a non-whackable character. The state is a dynamic attribute, while the position of the holes is a static attribute.

The fourth game object is the hammer which will be used to hit the characters. The hammer will be modeled after the movement of the controller, and it will have dynamic attributes such as the current position and current orientation. The hammer will also have some static attributes such as color, material, and size.

Other objects in the scene, such as the trees, rocks, mushrooms, and castle, are used for aesthetic reasons and are not interactable with the user. They do have attributes such as color, location, size but all their attributes are static.

2.4. Actions

There are two types of actions that can be performed by the player during gameplay. The two types are operative and resultant actions. Operative actions are base actions taken by the player, while the resultant actions are moves that the player takes in the game that have an indirect impact on contributing to achieving the goal of the game.

2.4.1. Operative Actions:

Hit a mole (dragons, goblin) with the hammer as they appear from the hole.

Avoid hitting the princess to improve the chance of winning.

2.4.2. Resultant Actions:

Avoid hitting a dragon/goblin that would make you hit the princess.

Sacrifice hitting a goblin in favor of a dragon since it is rarer and the player gains more points.

Hit closer characters first instead of further ones to maintain the combo level and increase the multiplier.

Position the controller in the middle of the screen to improve response time when hitting the characters as soon as they appear.

2.5. Characters

There are three types of characters in the game. The first type of character is the princess, which the player must attempt to save from the other characters. If the player mistakenly hits her more than three times, the game is over, and the player will lose. The princess is the second-rarest character to appear. The second type of character is the goblin. The goblins are the most common types of characters to spawn, and thus they are worth the least amount of points when hit. Each goblin is worth 5 points once successfully hit. The last character type is the dragon. The dragon is the rarest type of character, and thus he is worth the most amount of points, with 10 points each time he is hit. The characters will be randomly spawned in one of the 9 moles holes. The goblin have a 50% chance of spawning, princess has a 30% chance of spawning, and the dragon has a 20% chance of spawning.

2.6. Skills

Since this game is going to require the user to be able to hit all the moles successfully while avoiding hitting the princess. One of the core skills needed is good hand-and-eye coordination with fast reflexes in order to hit the characters as they appear from the mole hole as soon as possible before they disappear again. Additionally, they require good attention to detail to make sure they only hit the moles while avoiding the princess. The game also requires quick planning and decision-making skills in cases such as deciding between going for further characters who appeared first or closer characters on the way first. The player should also keep attention on the timer and score to assess their performance mid-game and see if they need to improve and change their strategy to win.

2.7. Chance

The game is going to have nine holes, with each hole being able to randomly pop up either a mole or a princess. The AI is going to determine how many moles can appear at one time and their time frames (for how long they stay on screen), and the same also applies to the princess. There is a level of probability and uncertainty in the game as it is not known to the player which holes the character will pop out of, and the player does not know which type of character will pop out, whether it is the dragon, princess, or goblins. Moreover, the probability of spawning a certain character out of the holes varies depending on the rarity of the character. Therefore, each time the game is played, different characters will appear from different holes in different sequences completely.

3. User Interface

The user interface will act as a bridge between processing the user input received from the physical input, which is the controller in this case, and the virtual world displayed through the VR head-mounted display to the user. The user interface must be well-designed to amplify the power and control a player has in the game world. The game will consist of multiple scenes, each containing different UI elements, such as menus, buttons, sliders, and modals, to allow for user interaction and control.

Game Description Scene: This scene provides information to the player to ensure that they are aware of the aim of the game and how to play it. The interface will have an intractable book, which will allow the user to flip through pages to understand the game story and its aim, as well as how to win.Those are important information for the user to be able to play the game. The interface will also have a start button on the last page to allow the user to click it once he/she is ready to start the game.

Menu Scene: The menu scene will allow the user to select a level to play. The interface will contain several interactable buttons which the user can click using the control to select a level and be redirected to the game scene. Upon clicking on a specific button, a new vertical menu will appear with the list of options. The option to select a level will have an image thumbnail for each level difficulty as well as a text caption to describe the level.

Game Scene: The game scene is where the player would have the most interaction with the game, aside from selecting and clicking buttons. The design of the scene itself will include a garden background with holes. Princess, dragons, and goblins as characters that pop up as moles from the holes. Only one character can pop up from a hole at a time in the easy level, but more will appear at once in the more difficult level. Multiple holes can have characters popping up simultaneously. There will be special effects and animations playing as the user interacts with the world through the controller by swinging it and hitting it.

Summary Scene: The summary scene is where the player can get feedback on the round he just played. The scene will have a simple interface with text describing is he won or lost. The interface will also show the time the player took in the round as well as the score achieved. The scene will also have a button to redirect the player to the menu scene to either choose another level to play or exit the game.

4. Visuals

The game setting will have medieval aesthetics, where it takes place in an old kingdom with green scenery. Outside in the garden is where the holes are going to be, where the princess, dragons, and goblins pop up as moles from the holes. Some animations and special effects will

take place when hitting the moles. The scene will have a castle in the background and will take place in a forest. There will be pine trees around, as well as rocks, grass, and mushrooms to decorate the scenes. The characters will also have animation when appearing from the mole holes and when they disappear after being hit.

5. Audio

During the game, whenever the player whacks a dragon or a monster, a whacking or hitting sound is played. However, when the player hits the princess, the audio played would be distinct from that of when the dragon or monsters are hit; this way, the player will be aware that they have hit the princess. Moreover, when the player misses or hits the princess, the player will be told to concentrate. Similarly, when the player hits combos, they will hear words of encouragement. Thus, the commentary will be based on the player's performance in the game. Furthermore, when the player hits the princess three times, or in other words, the player loses the game, audio will be used to notify the player that the game is over. Lastly, the game will have a background soundtrack that would be medieval-themed.

6. Story

Once upon a time, in the kingdom of Geneva, a beautiful princess named Adriana lived in a gigantic castle. She was loved dearly by everyone for her kindness and humble character. One fine day, at the break of dawn, the kingdom was attacked by the most vicious group of dragons. In order to save their kingdom, the king and queen fight bravely alongside their armada but are unfortunately killed in the battle. Being the commander of the knights, the dying queen whispers to you on the battlefield to protect her daughter and take her to a land far, far away from dragons. It is your duty to find the princess, let no goblin and dragons stay undefeated, and save the kingdom. You must serve to your last breath!

7. Level Specs

The game has three levels: easy, medium, and hard. The player first begins in the easy level, where they have the longest time to achieve the score needed. Similarly, the player will have less time in the subsequent normal and hard levels. As the name suggests, the hard level is the hardest out of all three levels, while the easy level is the easiest out of all three levels. As the player progresses in terms of levels, the game becomes harder as the time the player has is cut down, and there will be more than one-mole spawning at a time. If the player misses a mole, there is a score penalty making it harder to meet the score requirement for the level. In addition to this, in the harder levels, the delay before the mole disappears will be shorter. In other words, the dragons and monsters spend a smaller duration of time visible to the player for the player to whack them. In order to progress through the levels, the player must meet the minimum score required within the time limit, and the player must not hit the princess three or more times; otherwise, the level is reset, and the player must replay it.

8. AI

Artificial Intelligence in the game will be driven based on the performance of the player. Every 10 seconds, the average is calculated by taking the sum of the score of the player in the last 10 seconds divided by the number of moles that popped up within the 10 seconds. The performance average at the end of the most recent 10 seconds is compared with the performance average of the previous 10 seconds.

If the most recent average is greater than the previous average:

- Lauding comments such as “Good job!”, “You’re on fire”, and “That’s whacked!” are given.

- The delay time for the pop-up of the moles will be reduced, making it harder to not miss any moles.

Else if the most recent average is less than the previous average:

- motivating comments such as “You got this!”, “Hurry, hurry, hurry”, and “Live up to your name, knight!” are given.
- The delay time for the pop-up of the moles will be increased to make it easier

The above AI implementation will regulate the difficulty of the game based on the players’ performance, thereby making the game more exciting and intelligent.

9. Unity Implementation

9.1. Scenes

For the unity implementation, there are a total of six scenes. The first scene is the story scene, where the game story is explained as well as game control to play the game. The next scene is the menu scene which allows the player to choose a difficult game level to play and exit the game. Scenes 3, 4, and 5 are the three game levels available for the user, which are easy, normal, and hard. All three have slightly different game mechanics. The easy level only has one mole popping up at a time and 5 minutes to clear the level. The normal level has 2 moles popping up at a time and 3 minutes to clear the level. The hard level has 3 moles popping up and 2 minutes to clear. The last scene is the summary scene which is shown when the player wins or loses, and it contains their score, time taken, and a button to take them back to the menu scene.

9.2. Prefabs

The implementation requires certain assets for the characters and the scene. For the characters, there are princess, dragon, and goblin prefabs. Those characters' prefabs are instantiated from

the game script and placed in a certain mole hole randomly and destroyed either when it is hit or after a certain delay has passed. The scene also contains multiple prefabs, which are the castle, trees, mole holes, rocks, mushrooms, and mountains. Those prefabs are used to create and beautify the scene.

9.3. Scoreboard

One key aspect of the unity implementation of the game is the scoreboard. The scoreboard contains the current score, a countdown, the combo multiplier, and the number of lives remaining. The score starts with 0, and the goal is to reach 100. The score is incremented by 5 every time a goblin is hit and incremented by 10 when a dragon is hit. The score is decremented by 1 if a princess is hit. If a dragon or goblin character is missed, the player is penalized by decrementing the score by 2 in order to make the level harder and more entertaining. The timer count downs either from 5 minutes for the easy level, 3 minutes for the normal level, and 2 minutes for the hard level; once the timer runs out, the game is over, and if the score is not achieved, the player loses and is redirected to the summary page. The combo multiplier increments by 1 each time a dragon and goblin character is hit without a miss. The combo multiplier caps out at 5 and is added to the gained score. If a dragon or goblin character is missed or a princess is hit, the sequence is broken, and the combo resets to 0. Finally, the lives variable starts out with 3 and decrements each time the princess is hit. Once the lives reach 0, the game is over, and the player loses and is redirected to the game summary scene.

9.4. Hitting Characters

Hitting the character will be modeled using the right controller, which will have a handheld hammer that will be used to whack the characters. In order to register when a character is hit or not, the characters will have a collider surrounding them. The hammer will also have a collider around it. Any collision between the two objects triggers a function that will be called.

The function will check which type of character was hit and play a whacking sound effect. If the character is a princess, the lives will be decremented by 1, the score will decrement by 1, the princess object will be destroyed, and the combo multiplier will be reset to 0. Otherwise, if the character is a goblin, the score is incremented by 5 and the combo, and the combo multiplier is incremented by 1 by capped at 5. Otherwise, if the character is a dragon, the score is incremented by 10 and the combo, and the combo multiplier is incremented by 1 by capped at 5.

9.5. Controllers and Tracking

Using the XR interaction toolkit, we will be instantiated an XR origin object in each of the scenes. The XR origin will have a tracked position script in order to track the headset and controller's position and orientation. In addition, the XR origin will have a locomotion system and a continuous move provider to allow the user to use the left joystick to move around the scene. The XR origin will also have a snap turn script to use the right joystick to change the head orientation and perspective of the game view. Moreover, the player will also have the ability to teleport in order to move around the scene faster and get to the moles. To do that, the ground plane will be set as a teleportation space, and the XR origin will have a teleportation provider so that the left trigger can be used to teleport. The controllers will have a ray cast interactor. The ray of the left controller will be used for teleporting. The player’s head will be tracked, and the game view will be adjusted accordingly.

9.6. Audio Effects

The game implementation will have several audio effects. Firstly, a medieval background soundtrack will be played throughout the level. Once a character is whacked, a short audio clip of a whacking sound effect will be played. Moreover, as described in the AI implementation, there will be commentaries played depending on the player’s performance. If the player’s

average score per appeared mole in the current 10 seconds is more than that of the past 10 seconds, an encouraging commentary is played. There are two versions. Either an audio clip saying “good job” or “keep going” will be randomly played. However, If the player’s average score per appeared mole in the current 10 seconds is less than that of the past 10 seconds, an encouraging commentary is played. There are three versions. Either an audio clip saying “what are you doing?” or “are you the mole?” or “try harder” will be randomly played. Finally, in the summary scene, an audio clip saying “you win” will be played if the user wins, or an audio clip saying “you lose” will be played if the user loses.

9.7. Scripts

There are several scripts needed for the unity implementation. Firstly, there is a script needed in the menu scene to redirect the user to the correct level of the game. Secondly, a game controller script is needed. The game controller script will have the AI logic of the game to assess the player’s performance and play the appropriate audio clip commentary and either increase or decrease the delay between the moles appearing. The game controller script will also have the logic to maintain the timer countdown and keep checking if the user wins or loses by ensuring that the lives are greater than 0 and the timer has not run out to continue the game. Otherwise, if the timer runs out or the lives are 0, the game is redirected to the summary scene with a game-over text. There will be 3 versions of the game controller script one for the easy level, the normal level, and the hard level. The normal level differs from the easy level as it has less delay between the moles appearing, the duration is 3 minutes, and there are 2 moles appearing at a time. The hard level differs from the normal level as it has less delay between the moles appearing, the duration is 2 minutes, and there are 3 moles appearing at a time. Moreover, a hammer script is needed to check if its collider has a collision with a character collider; if so, the audio clip of whacking is played, and the hit function in the character script

is called. The character script contains a hit function that checks the type of the character and updates the score, and combo, lives accordingly as previously described, and then it destroys the character object. Finally, the summary scene script will read values of the score, time, and win/lose status from the player preferences saved and loads them to a UI canvas to display to the user at the end.

9.8. UI

All UI elements will be interactable with the rays from the oculus controllers. This will be achieved by using a different event system for the canvas, which is XR UI Input Module. The UI for the menu scene is made up mainly of buttons. If the play button is pressed, a side menu will appear showing 3 pictures for each of the 3 levels for the user to select from. By hovering over each of the images, a short description of the level will be displayed. The second button is an exit button, and upon pressing it, a modal will pop up to confirm that the user wishes to exit. If yes is pressed, the game is exited; otherwise, if no is pressed, the player will return to the menu scene. The summary scene will have a similar setup to the interactable canvas, it will mainly have a button which, once pressed, the user will be redirected to the menu scene to either exit or choose another level to play. The scoreboard in each game level is also a UI canvas; however, it will not be intractable with the controllers since it will simply display the variables.

10. Rendering

For the rendering of the scenes, to change the shading and lighting of the scene, the color space rendering mode was set to linear. The linear rendering mode prepares Unity to do lighting and shading calculations using physically accurate mathematics before transforming the final output into the format that works best for monitors. We changed the color space from gamma to linear since light-striking surfaces have different response curves, and image effects behave

differently. With gamma rendering, the scene looked very dark, and the colors were not as vibrant since the player stood in front and it cast a shadow on them, but with linear rendering, the scene looked better and more vibrant. The Universal Render Pipeline (URP) was also used since some of the assets we imported required it so that the material would be visible. The render pipeline performs a series of operations that take the contents of the scene and displays them on the screen. Moreover, the URP pipeline chosen can easily create optimized graphics across a range of platforms, from mobile to high-end consoles and PCs. However, as a result of choosing URP, it has a slightly different mechanism for handling reflection capturing and reflection blending. Therefore, instead of using a spotlight to introduce lighting, we used a reflection probe, and this change had to be done manually.

11. Tracking

For the project, we are tracking the headset rotation to move the game perspective. We choose to apply a locomotion system for our XR origin as well as a snap turn. This way, the player can use the left joystick to move around instead of physically moving around since the space is limited. Moreover, the right joystick is used to change the perspective in increments of 45 degrees so that the user does not have to move around much physically. The controllers are also tracked in order to model the movement of swinging the hammer, and this is done by using the gyroscope sensors on the controllers to mimic the movements of the player. Moreover, we made the ground plane a teleportation area and the controllers as a teleportation provider in order to let the user teleport around the map.

12. Testing

12.1 Stereo-Vision

The objects of the game appeared similarly to both eyes. The objects appeared to be the same size, and the eye view was not revered. Also, there were no objects that appeared to one eye only. Overall, the game did not suffer from any issues in this aspect. We

tested it by closing one eye and keeping the other one open and switching between the two, and we did not face any issues. Overall, the game passed the stereovision test for the majority of the band, as there were no discrepancies in the object size, location, mirroring, or depth between the two eyes.

12.2 Rotations

In terms of rotation, all rotations were fine and did not cause any glitches or blurriness. Even cartesian rotations were okay and not distorted. There were no irregularities in the depth perception of the objects either. We first tested each rotation separately and then combined them.

12.3 Tracking

For tracking, there were no major issues at all. The tracking of the headset and controllers faced no issues at all and were being tracked properly.

12.4 Vestibule-ocular reflex

For the vestibular ocular reflex, yaw head movement while focusing on an object did not cause any distortion or give an illusion of movement

12.5 Peripheral Perception

There were no issues with peripheral vision, and we did not experience any peripheral distortion when rotating the eyes only, and the head was fixed.

12.6 Latency Perception

Firstly, the game has high latency perception meaning that the player may feel nauseous or take quite some time to adjust to their surroundings after playing the game. Some of the team members did experience some nausea and dizziness after the game. This is mostly because of the constant background music and speed of movement.

13. Limitations

Firstly, limited assets for graphics and animations were available, thereby limiting creative freedom and preventing the developers from implementing their ideas. Next, if we had more time, we would like to include a database in order to store the scores and have a leaderboard system for the game so that users could see their positions across all other players as well. In addition, the movement is linear and feels like sliding through the game; it would cause motion sickness and disorientation. Adding slight head bobbing would help resolve this problem since it would simulate the movement of an actual person. Furthermore, the game could only be tested within limited space, thereby restricting movement and making it uncomfortable for the users.

14. Project Link

https://drive.google.com/file/d/1h7g8Z9099rXIrcwm_hEy3QulPtiZLjmv/view?usp=share_link