

Computer Games Development SE607

Technical Design Document

Year IV

Danial Nor Azman

C00253517

[26/04/2023]

**DECLARATION**

****

Muhammad Danial Hakim Nor Azman Azman

26/4/2023

Danial Nor Azman

C00253517

Contents

Contents

[2 Accessibility in game for disabled people 3](#_Toc133245022)

[3 Summary of the game 3](#_Toc133245023)

[4 Bullet Elements 3](#_Toc133245024)

[5 Gun Types 4](#_Toc133245025)

[6 Enemy 4](#_Toc133245026)

[6.1 States 4](#_Toc133245027)

[6.2 Decision Making - Fuzzy Logic 4](#_Toc133245028)

[6.3 Adaptive system 5](#_Toc133245029)

[6.4 Vision Cone 5](#_Toc133245030)

[6.5 Pathfinding 5](#_Toc133245031)

[7 Procedural level generation 6](#_Toc133245032)

[8 Accessibility settings 6](#_Toc133245033)

[8.1 Magnifier 7](#_Toc133245034)

[8.2 Rebind Controls 8](#_Toc133245035)

[8.3 Visual cues as alternate to Audio cue 9](#_Toc133245036)

# Accessibility in game for disabled people

The purpose for this Technical Design Document is to outline the design and development for a game called LabEscape that focus on testing the accessibility features for people with disabilities.

# Summary of the game

LabEscape is first person rogue like shooter game in which the level layout and obstacle are procedurally generated at run time so that player can have a different experience every time. The in-game objective is cleared out a number of rooms depending on the size of the lab room generated. Once the number of required rooms cleared has been reached, a door will spawn in the starting area where player must make their way back without dying as there is no healing option. Once player pass through the door. A message will appear declaring they have won. They can use different types of weapons and ammo type against enemy that have adaptive system to the guns the player is using.

# Bullet Elements

Logo

Description automatically generated with medium confidenceA picture containing bubble chart

Description automatically generatedA picture containing icon

Description automatically generatedA picture containing bubble chart

Description automatically generated

There are 4 different types of elements and different effects.

Ice – Freeze the enemy movement for 2 seconds.

Fire – Set the enemy on, deal damage over time for 3 seconds.

Electric – Deal area of effect damage.

Water – Slow the enemy movement for 4 seconds.

Player can change their bullet types by interacting with these objects in game. The game used Enum class called elements and when player interact one of these, the bullet will update its element. The effects are applied when the bullet collide with the enemy.

# Gun Types

A picture containing text

Description automatically generatedA picture containing text, sign

Description automatically generatedA picture containing text, sign

Description automatically generated

There are 3 different types of guns in this game.

Assault – Allow player to hold the fire button and keep firing until it run out of ammo or reload.

Burst – Fire 3 bullets with 1 click but required the fire button to be clicked again.

Shotgun – Fire 5 bullets at shorter range which random spread

This is achieved due to the custom gun script which control how the gun works. Some of settings are listed below:

* Time between shots
* Bullets per tap
* Spread
* Magazine size
* Allow button hold.

Changing the value for some of these settings would allow the gun to behave differently.

# Enemy

## States

The enemy have 3 main states which is listed below.

Patrolling: Enemy will patrol around the room, avoiding the obstacle for both player and enemy. There are multiple points in the room where the enemy must go to, the points are randomly chosen once the enemy has reach one of the points. This will go in a loop so enemy will always patrol unless an event occurs.

Chasing / Attacking: Enemy will chase the player if player is found. Player is considered found if enemy can see the player with their vision cone. Player will be attacking once player is in attacking range of the enemy.

Retreating: Enemy will heal up if the enemy health is below a certain threshold unless an event occurs.

## Decision Making - Fuzzy Logic

The enemy use fuzzy logic in their decision making.

Decision for the AI is decided like the table below where low, mid and high represent the threat level.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Player Dying | Player Hurt | Player Healthy |
| Enemy Dying | Mid | High | High |
| Enemy Hurt | Low | Mid | High |
| Enemy Healthy | Low | Low | Mid |

Each threat level also calculates the speed the enemy will be moving. If enemy health is 30 which is considered low and player health is 80 which is considered high, they will move at the speed of 4. If the enemy health is at 20 and player health is at 95, the enemy will move at speed of 7 to heal up which is almost double than their previous speed.

If the enemy health is low where it should be healing, but the player health is lower, the enemy will prioritize attacking the player. The threat level basically makes the enemy either more passive or aggressive depending on the situation.

## Adaptive system

As there are multiple type of gun and bullet elements, the enemy will adapt to it with their stats. The game keeps tracks on how the player kill their enemy. If player keep on using the fire bullet, enemy will take reduce over time damage and at one point will be immune to it. If player keep on using ice bullet where it froze the enemy for 2 seconds initially, the frozen duration will go down to 0 if player keep on using it, making it just like a normal bullet.

This also mean the longer the player plays the game, the harder it is to kill the enemy as they will get immune to the element’s perks.

## Vision Cone

The enemy have a vision cone which act as the enemy eyes. The vision will be displayed on screen so player can see to avoid them. The vision cone cannot go through walls and obstacle so player can choose to take cover so the enemy will stop chasing them.

## Pathfinding

As the game is generated at run time, I use NavMesh which is a component from Unity which allow the enemy to automatically walk on the terrain based on the settings that I set.

# Procedural level generation

A picture containing diagram

Description automatically generated

The game level generation is by using templates of room that were created before run time. The room, however, is flat surface with different entry and exit points. The room spawn by checking if there’s a room at the exit point of the current room. If there’s no room, it will spawn a room corresponding to that room.

For example, if a room with an opening at the bottom of it, a room with an opening at the top will be spawned in. If 2 rooms clashed while it was spawning, it will spawn a closed room which acts as a wall, this is to prevent the room from spawning on top of each other as room that handle the spawning can spawn different type of rooms.

Once the layout of the level has been spawn, the game then randomly pick obstacles to spawn in the room so even if the room is of the same layout , the obstacles will make it different.

# Accessibility settings

## Magnifier

Chart

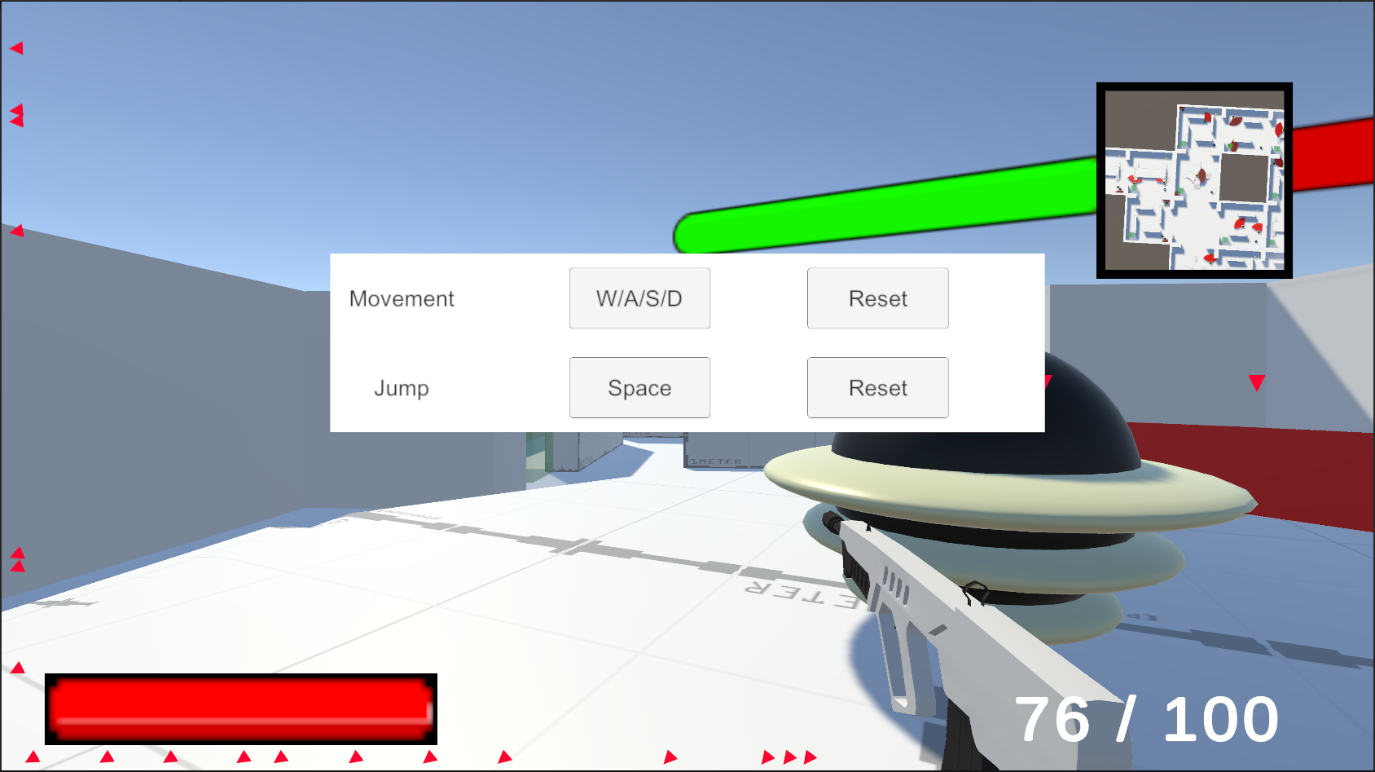
Description automatically generated with medium confidence

Magnifier can be accessed by pressing P on the keyboard. It will enlarge item or objects that are further away from player, so it is easier to be seen. This is done by having another camera in the scene which will handle the zooming in. Put the camera view onto a texture and put it on a game object to create zoom in like effect. Some maths calculation was required to get the effect I need.

If an eye tracker is attached, player move the magnifier around the screen. The position is default to the middle of the screen. Player can also zoom in and zoom out if they close one of their eyes to better fit their needs.

Limitation: only works on PC as you would need to press the P button on the keyboard.

## Rebind Controls



Player can rebind controls to suit their needs. This is done in script where it checks how many buttons is responsible for that particular action such as movement and jumping. If it is a composite action like the movement, it will ask for 4 inputs. The script then updates the bindings and store it in a Json file, so player does not need to rebind it every time they play the game.

Player can also click on the reset button to reset to the default settings. These settings are not limited to keyboard so player can connect their controller to rebind the actions.

Limitations: interact button and the sensitivity of the camera movement cannot be rebind / change to fit the player preference.

## Visual cues as alternate to Audio cue



The red arrows are visual indicators that help players with impaired hearing locate enemies. The arrows can go through walls and show where enemies are if they are in front of the player, and they adjust based on where the player is looking. The arrows also change opacity to show how far away enemies are, with faded arrows indicating greater distance.

There are 3 scripts to achieved this, the first script is on the canvas to calculate the position and where the arrow should be pointing. The second script is attached to the enemy as it needs to be updated all the based on the enemy position. The last script is attached to the player, as the calculation is also based on the player position and the target position.