Third Person shooter using pygame

Final Project Report

Compiled by:

Jerico Tjan (2802523042)
L1BC

Algorithm and Programming
BINUS University International
Jakarta
2024

TABLE OF CONTENTS

Chapter 1

PROJECT SPECIFICATIONS

1.1. Project Description

For my final project I used Pygame to develop a 2D shooter game where the player controls a character (player sprite) that shoots at enemies and gains points when killed. They all not only have mechanics such as shooting with different kinds of weapons, reloading, health bar system and movement system but. A player has a goal when it comes to facing enemies: it is to eliminate these enemies and deal with health and ammo at the same time.

The core gameplay loop involves:

- Movement of the player on the field and aiming the shot by employing keyboard and mouse solely.
- Multiple guns exist each with its own characteristics (damage, firing rate and reload time).
- Spawning an enemy and an alternative AI that will make an enemy chase the player and cause damage when collided.
- An environment in which there are walls and floors consisting of a player so that there are tiles and objects during the game.

1.2. Project Link

The GitHub Repository of my final project can be accessed through the link provided below:

GitHub Repository

1.3. Essential Algorithm

The game relies on the following core algorithms and principles:

1.3.1. Player Movement:

The player's acted by using the WASD keys to make movements. The Dungeon class outlines what a tile map of the game seems like. The can_move method tests if a player can move to a given tile using tile_map … This function checks that no wall (which is represented by 1 in the tile map) lies in the target. This makes certain that the player can only move at areas that there is a possibility of navigating.

1.3.2. Shooting Mechanism:

The player can shoot using the mouse left click. The shoot method also determines the direction from a position where the player currently is to a position where the mouse is, using basic trigonometry (atan2). A bullet is then fired in that direction based on damage variables, or dispersion by one or the other feature of the gun.

The Gun class has the fire_rate attribute which reverses shots at specified intervals (to consider or simulate the time spent in reloading).

Bullets are objects of the Bullet class, as for each of them at each frame their rect attributes are modified according to the velocity.

1.3.3. Enemy AI:

The Enemy class or the AI class has functionalities to place an enemy, to move and attack the player. The enemies have a fixed wandering AI with which they chase the player: They keep an eye on the player's position and then glide toward the player position using the basic vector mathematics (Determining the direction to the player and then using normalization vector).

When an enemy reaches the player it inflicts damage on the player and the enemy is destroyed.

Use the visible attribute to get the enemies more visible, or set it to false to make them less visible to control their spawn on the screen.

1.3.4. Health and Ammo System:

To do that the Player class retains health (hp) and ammo count for the weapon currently picked up by the player. There is a health bar and ammo count written on the screen for the purpose of the player interface.

Ammo is restocked through the reload system The amount of ammunition indicated is permanently lost. The reload method in the Gun class refuels the ammo off to the maximum limit of current weapon type.

The health bar is a simple graphical that is drawn and redrawn every frame to display to reflect the player's health.

1.3.5. Camera System:

There is the implementation of an always-follow camera with some amount of smoothing to introduce a more fluid playstyle. The camera can only move within the boundaries of the dungeon map so it doesn't go beyond this area.

The value for the position of the camera is adapted each frame and depends on the position of the player but leaves the player within the center of the screen when he is not close to the edge.

1.4. Modules

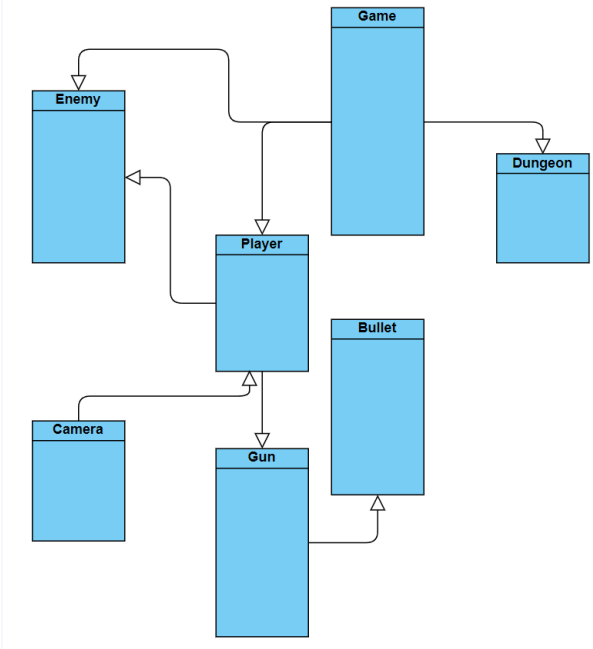
In the 2D shooter game, certain external modules are incorporated, which performs or handles certain general-purpose duties, thereby making it easier to handle the complicated tasks of game display, movement, etc., random and any mathematical computations. The primary modules used in this program are Pygame, Math and Random.

Chapter 2

DIAGRAMS

2.1. Activity Diagram

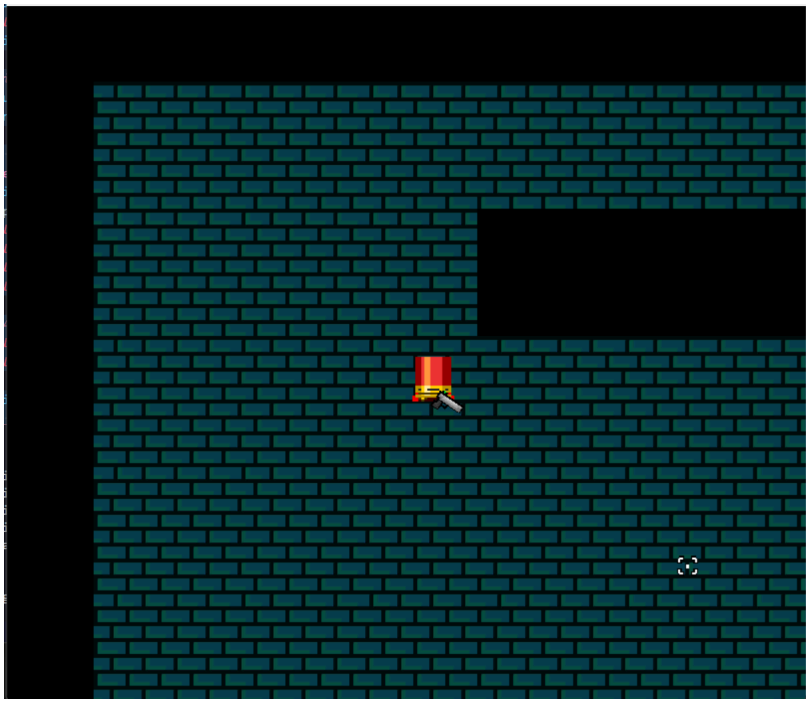
2.2. Class Diagram



Chapter 3

DOCUMENTATION

3.1. Screenshots



Chapter 4

EVALUATION AND REFLECTION

4.1. Lessons Learnt and future lessons

There are a lot of lessons that I was able to learn through the development of the final project game that will enhance future undertakings and one of the most important lessons I learned was that of the game loop. The smooth performance depends greatly on the fact of managing the game loop correctly, that is, the interaction of objects such as enemies, bullets and actions of the player. Another important is the understanding of input management in particular with players One important idea is mastering player inputs, managing inputs. These include movement, aim and shoot controls in order to enhance the gameplay and any hitch or latency can dampen the gameplay. And lastly time

management on unnecessary features on the game such as dynamic camera and recoil camera, this took an unprecedented amount of time creating it and was the major problem, if i was to focus on other things first and the extra features later perhaps this build could become a better game overall.

REFERENCES