World Z

Formal Game Proposal

Elias Hanken

Tor Gikling

September, 2022

Table of contents

1. Game Description

1.1 Idea

Our goal with this project is to create a game, in which a single player can experience an FPS 3D zombie game. The world which the player spawns in is explorable to an extent. The main goal of the game is for the player to survive rounds/areas of where zombies spawn/are. Some areas contain missions to further advance. To finish the game, all areas/rounds must be completed.

1.2 Story

The player woke up one day and the world around him was not the same as before. The zombies had taken over the world, and there was only one thing that could be done. Defeat all the zombies and find the cure against the zombie infection.

1.3 Gameplay

The core gameplay of our game is a built around an FPS controller. You can move around the environment freely, pick up weapons, ammo, med-kits, etc. You can shoot the zombies that try to kill you with the gun you have picked up. If you receive damage you can use/pick up medkits spawning in the environment. Missions will pop up on the screen which makes it easy to complete the objective of the level/area.

1.4 Implementation

1.4.1 Design

Each entity of the game have a visual representation of the environment, this furthermore contributes to the possibility of each entity having intelligence or logic which defies its behaviors. When the player starts a game a controller container is created of instance player for the user to use. Since we need to place all the entities/environment stuff in a playable map, we use a map editor of choice which we then compose the scene and put everything needed for the game function as intended.

1.4.2 Classification

Static entity

This class contains all entities of a map, which do not change position or form during the whole game. Static entities have infinite mass, they do collide with other entities but have no further logic. Examples of this class are walls, floors or stairs.

Semi-static Entity

This class contains entities which can be given a path and react to events. They collide but have infinite mass and cannot be moved by collision. Examples of this class are elevators, doors and switches.

Dynamic Entity

Entities which can be moved by physics and external forces are called dynamic. They do also have a certain mass and react on collision. Examples of this class are boxes, weapons.

Character entity

These are intelligent entities of the world. As shown above this is the Player character. This player controller can interact with the environment and complete tasks, and shoot zombies. There are player characters and non player characters which are the enemies(zombies). Characters have health and can be damaged and killed.

Abstract Entity

This class have the purpose to query themselves for information to complete it. They have no physical representation. Example of this is spawn points for different types of game objects.

1.5 Visual Aspect

The characters are realistic and must look realistic to the best possible extent. The same with the map. It should feel and look realistic.

2. Development Schedule

2.1 Team

2.1.1 Elias Hanken

Semester: 5

Related Courses: Game Development, Algorithm and Data structure, Java Programming, c++ development.

2.1.1 Tor Gikling

Semester: 5

Related Courses: Game Development, Algorithm and Data structure, Java Programming.

2.2 Tasks

2.2.1 Functional Minimum

The functional minimum should be the prototype of the game. It should contain all elements required for a proof of concept:

- Standard map with enemy entities
- One playable FPS character.
- Working physics for map and entities.
- Combat system.
- Health for player and enemies.

2.2.2 Low Target

The low target is a game which could be released.

- Simple start menu.
- One loadable map.
- Weapon pickup (simple inventory)
- Pickups (health, ammo)
- Animation for player entity and enemy entity.
- Sounds for all events.
- Missions to complete.
- Game completion (complete missions).

2.2.3 Desirable Target

The game should now look good and could be released. It contains:

- Improved Enemy AI system.
- Controllers for swithes, elevators and doors, with Tooltips (hud showing what actions can be done).
- Effects: Gun muzzle, blood, etc.
- Character improvement: sneak, crouch, run, jump, slide, lay on floor.
- Inventory feature: store medkits/ammo. Drop items.
- Improved visuals.
- Game menus: Main menu, in game pause, settings.
- Playable finished map with missions, objectives, and explorable areas.
- Finished game loop.
- Mission HUD and mini-map.

2.2.4 High Target

High target is an improved desirable target.

- Bigger map.
- Different enemy types.
- Drivable vehicles.
- Saturation.
- Storytelling.
- Alternative endings.
- Friendly NPC with AI.

2.2.5 Extras

- Multiplayer

2.3 Schedule

Autumn break is colored gray.

Date	Course Items	Elias	Tor
30.09		Entity sketches	Map sketches Other object sketches
07.10	Formal Game Proposal	Simple Player Controller Simple enemy	Simple map Physics Dynamic entities
14.10		Shooting Damage handler	Health/ammo consumption 'Abstract entity in map' including spawn possibility
21.10	Game prototype	Simple enemy AI Combat system	Add all entity types with functionality if possible.
28.10		Sounds Simple Inventory Missions/rounds to complete	Start menu Animation Playable map
04.11	Report, Low Target	Finish Low target	Finish Low target
11.11		Work on desirable target	Work on desirable target
18.11	Playtesting / Desirable target	Work on desirable target	Work on desirable target
Week 47 Last day to finish game		Finish desirable target Present project	Finish desirable target Present project