Milestone Reports and Demo

For both **Milestones #1 and #2**, write a simple progress report, highlighting:

1. Work done/accomplish so far

2. Work to do next, or upcoming tasks

3. Problems encountered, and how they were fixed / not fixed yet

4. Any revisions from the original proposal and reasons behind it.

Note: In **Milestone #1**, please also include a **Gantt chart**, showing your planned project tasks and schedule.

Milestone Reports

Please hand-in your milestone reports (ZIP archive) to MMLS by

20th August 2020 (Thu) 959 hours (for Milestone #1)

10th September 2020 (Thu) 0959 hours (for Milestone #2)

Demo

During the lab session on the same day, you will have to demo your project, showing whatever you have done so far until the milestone. **All members** of the project group are expected to be present during the demo, as contribution checks and questions will be posed to all members.



**GAME ALGORITHMS (TGD3351)**

**TRIMESTER 1 2020/2021**

**Milestone Report #1**

|  |  |
| --- | --- |
| **Student ID:**  1171101517 | **Student Name:** Michelle Chai Mei Wei |
| **Student ID:**  1171100973 | **Student Name:**  Foo Fang Jee |

# Work Done

1. **Player**

The player class can move, shoot bullets and fire missiles. The player is moved by using the arrow keys, left arrow, right arrow, up arrow, and down arrow. When the space bar is pressed, bullets are fired in a straight line at a firing rate of 150 milliseconds with a speed of 300 milliseconds. When the z key is pressed, a missile is fired at the enemy. We are currently working on the selection of enemy and the path finding algorithm.

1. **Enemy**

For enemies, we have completed the simple enemy\_1 with kinematic seek algorithm.

1. **Scene Management**

To swap between different scenes, i.e. main menu, the actual gameplay, and the game over screen, we used enum to define each scene, and switch statements under Update() and Draw() of our Game1 class, where depending on certain conditions, the game will update different things and draw different things.

We also created a button class for the “start game” button and “exit” button in the main menu scene, as well as the “restart” button and “exit” button in the game over scene.

1. **Collision**

The direct collision between player and asteroid works; The collision between player bullets and asteroids works;

1. **Camera**
2. **Asteroid**

It is randomly generated from the top of the screen and dynamic wander algorithm is implemented.

# Upcoming Task

## Coding

1. Add boss
2. Add turret
3. Add line of sight algorithm for all NPC objects
4. Add second enemy
5. Add missile – pathfinding algorithm
6. Add GUI – score, health, menu button, missile
7. Add sound effects
8. Add powerup
9. Code more levels
10. Fix bugs
11. Tutorial pop-ups

## Documentation

1. Draft report for milestone 2

# Problem Encountered

## Lack of documentation

The Monogame framework lack complete documentation, where it was hard for us to find the necessary information to code certain features that we want. Not to mention, with different versioning, the code is different as well so despite finding solutions on the forum, we have to filter them one by one, ensuring that we are using the same version of Monogame.

For example, in terms of loading the terrain map:

For Monogame assemblies’ version 1.1.0, the TiledMapRenderer’s update function accepts two arguments, while for the version 3.7, the update function only accepts one argument.

# Proposal Revision

1. **Powerup**

We plan to add some powerups such as changing the bullet pattern/invulnerability and such.

1. **Tutorial (pop up)**

The first level will be a super easy level, where we allow the player to learn the mechanics of our game, shoot (space bar), missile (z), move left, right, up, and down.

1. **Pause Scene**

We may also add a pause scene when the player presses the “Enter” key, where the player can choose to continue or quit the game.