



TGD2251 GAME PHYSICS

Project #1

Report

Lecture Section: TC1V

Tutorial Section: TT1V

for

Dr. Wong Ya Ping

From:

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

Introduction

The game is an endless runner-style game similar to Jetpack Joyride, Tiny Wings. The player has to travel as far right as possible to obtain higher scores. The game stores all the scores that has been accumulated throughout different play sessions to emulate the old arcade style of gameplay.

Documentations of Your Game

Goal: To travel as far as possible towards the right




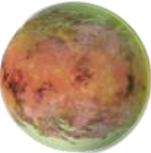
Environment: Outer space


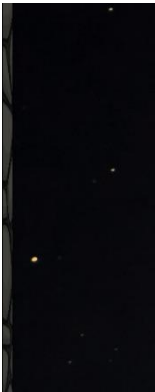
Lose Condition: Lose all health

Mechanics:

1. Control the angle of the ship's trajectory
2. Strength of the trajectory

Game Objects

Texture	Name (SFML, Box2D)	Body
	Player	Dynamic
  	Planet	Static

		
	Wall	Static

Description of Game Objects and User Interface (UI)

Player

The player is represented by the small red spaceship. The force of its movement is determined by how long the player presses the space bar. The maximum force is $100.0f \times 5$, while the minimum force is $100.0f \times 1$. The spaceship is on the left of the planet as shown in Figure 1.1.

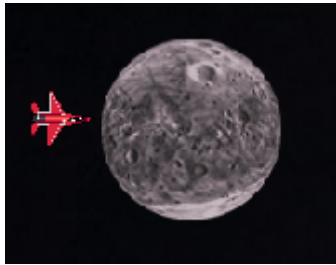


Figure 1.1: Spaceship

The linear damping and angular damping of the spaceship is set at $0.5f$ and $2.0f$ respectively to prevent the ship from spinning nor travelling at a constant velocity.

Planet

Planets are generated as the game progresses based on the location of the player. The generation is checked against the player position and all the existing planets so that the newly generated planet does not overlap with them. The texture of the planets is randomly selected from an array of 8 textures. It has a radial pull with a force of $0.02f$ that affects the player if the player is within 5 meters of the planet.

Wall

There are three walls used in our game, which act as borders for our game. The walls exist to guide the player so that they do not move too far towards the left, top or bottom. This will prompt the player to move towards the right naturally.

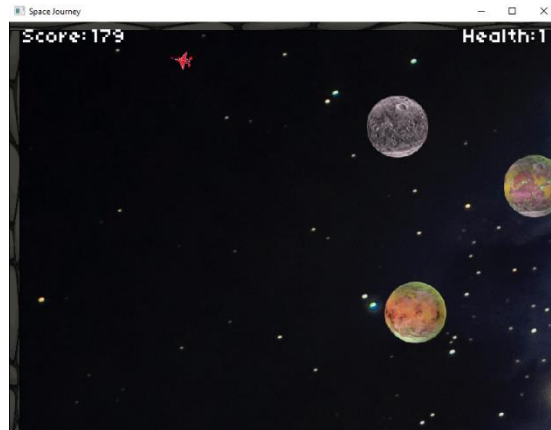


Figure 1.2: Player reached the top

Whenever the player approaches the left wall, top wall or bottom wall, the view stops following the player to further indicate that the player has reached the end of our game world as shown in Figure 1.2. The view resumes as soon as the player leaves the area.

Charge bar

The charge bar has 5 green boxes that represent how long the player holds the space bar. Each box takes 0.3f to charge up. The maximum charge one can accumulate is 5. When the player charges to the maximum strength and they do not let go, the charge bar goes down again. The charge bar is on the left of the space ship as shown in Figure 1.2.



Figure 1.3: Charge bar

User Manual/Instructions

How to install our game

1. Ensure that you are at the directory of the game folder in cmd.
2. To compile our game and generate an executable file, please copy the following line and paste it to your cmd:

```
compile Main.cpp Planet.cpp Strength.cpp Player.cpp Wall.cpp MyContactListener.cpp
```

3. Type a in the cmd to run our game

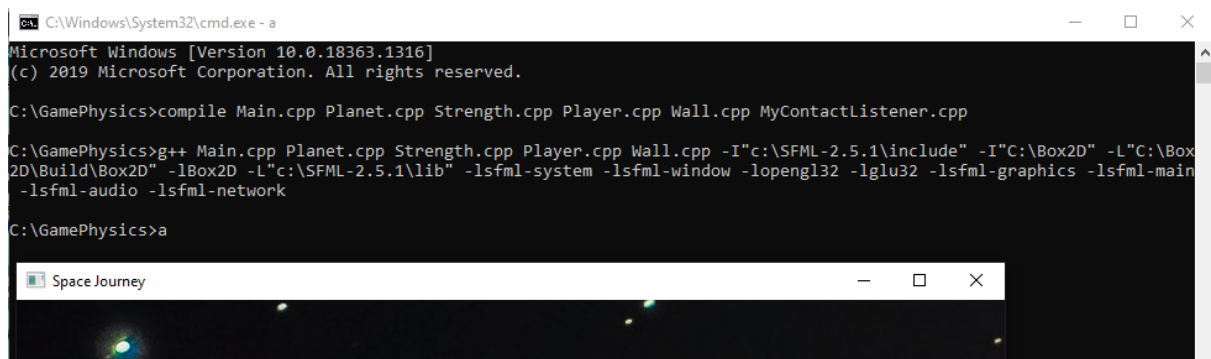


Figure 1.4: Demo

How to play our game

Keys	In-game action
↑	Counter-clockwise turn
↓	Clockwise turn
Space	Strength of the force

Table 1: Basic Gameplay Mechanics

As the player moves, they must watch out for the planets since they are randomly generated. Colliding against the planet will reduce the player's health by 1. The player only starts with 5 health. Getting too near to the planets will cause the player to be pulled towards the planets.

Features

1. Audio – Our game has a cute 8-bit BGM
2. Textures – Our game has textures for all the important game objects
3. Scoreboard – Our game has a scoreboard which has all the previously obtained scores

Screen Shots

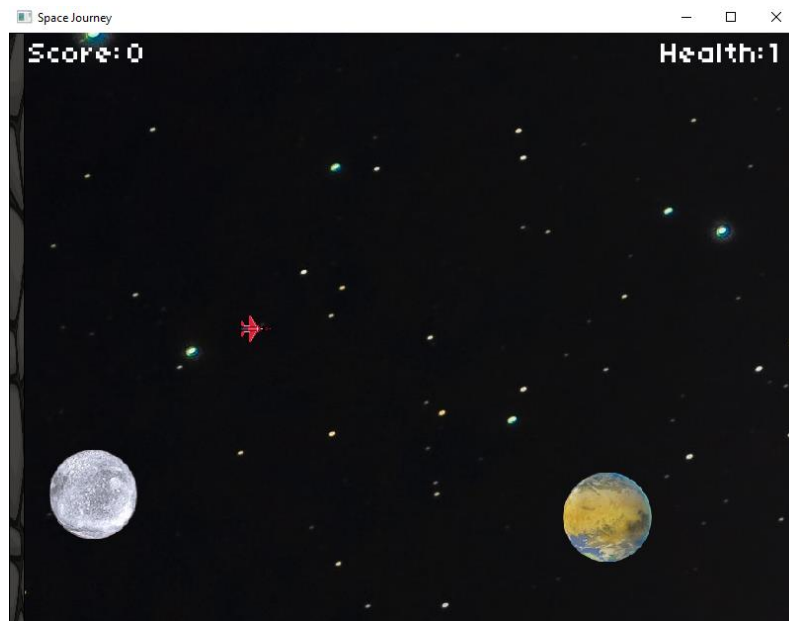


Figure 1.5: Gameplay

When the player starts our game, they will see Figure 1.5. The spaceship will be positioned towards the left of the screen. The planets will be automatically generated based on a timing of 1.0f. The player can change the angle of the spaceship by either pressing the Up arrow key or the Down arrow key. The player can then move by holding the Space bar.

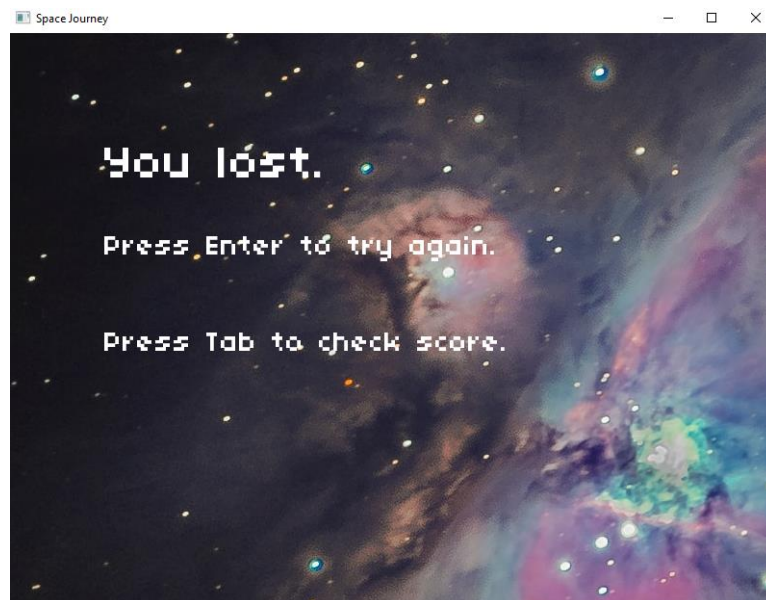


Figure 1.6: Lose screen

When the player loses all 5 of their health, the lose screen will appear as shown in Figure 1.6. If the player presses the “Enter” key, the game will restart. If the player presses the “Tab” key, the score board will be displayed as shown in Figure 1.7.

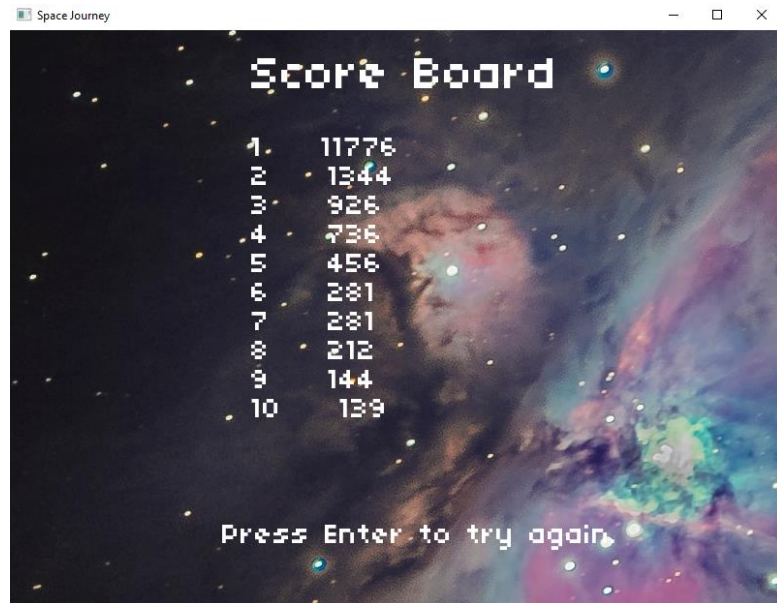


Figure 1.7: Score board

Link to Demo (if any)

<https://youtu.be/Q4K4ntGyjFM>

Acknowledgment

First of all, we would like to express our gratitude and utmost thank to our beloved lecturer, Dr. Wong Ya Ping for his teachings. Other than that, we would like to thank our classmates for the help they have provided us.

References

Lab Materials from TGD2251 - Game Physics

Main Page (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 10, 2021, from <https://www.sfml-dev.org/documentation/2.5.1/index.php>

sf::Clock Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 10, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1Clock.php

sf::CircleShape Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 10, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1CircleShape.php

sf::RectangleShape Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 10, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1RectangleShape.php

sf::Event::KeyEvent Struct Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 12, 2021, from https://www.sfml-dev.org/documentation/2.5.1/structsf_1_1Event_1_1KeyEvent.php

sf::Window Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 13, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1Window.php

sf::Vector2< T > Class Template Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 13, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1Vector2.php

sf::Sprite Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 13, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1Sprite.php

sf::View Class Reference (SFML / Learn / 2.5.1 Documentation). (n.d.). Retrieved January 14, 2021, from https://www.sfml-dev.org/documentation/2.5.1/classsf_1_1View.php

Set fixed sprite position when sf::View is active? (n.d.). Retrieved January 15, 2021, from <https://en.sfml-dev.org/forums/index.php?topic=14156.0>

BodyDef class - box2d library - Dart API. (n.d.). Retrieved January 16, 2021, from https://pub.dev/documentation/box2d_flame/latest/box2d/BodyDef-class.html

11 planet sci-fi set | OpenGameArt.org. (n.d.). Retrieved January 18, 2021, from <https://opengameart.org/content/11-planet-sci-fi-set>

game physics - Slowing down objects in box2d with no gravity - Stack Overflow. (n.d.). Retrieved January 19, 2021, from <https://stackoverflow.com/questions/8934889/slowing-down-objects-in-box2d-with-no-gravity>

C++ SfmL 2.0 Made Easy Tutorial 18 - Screen Scrolling (Sf::View) - YouTube. (n.d.). Retrieved January 19, 2021, from <https://www.youtube.com/watch?v=ghgd-R1gRmc>

I Learn Some Things: Air Resistance in Box2D. (n.d.). Retrieved January 20, 2021, from <http://ilearnsomethings.blogspot.com/2013/05/air-resistance-in-box2d.html>

Simulating Multiple Sources of Gravity in Box2D | Mental Grain. (n.d.). Retrieved January 20, 2021, from <https://mentalgrain.com/box2d/simulating-multiple-sources-of-gravity-in-box2d/>

Simulate radial gravity (also known as “planet gravity”) with Box2D as seen on Angry Birds Space | Emanuele Feronato. (n.d.). Retrieved January 20, 2021, from <https://www.emanueleferonato.com/2012/03/28/simulate-radial-gravity-also-know-as-planet-gravity-with-box2d-as-seen-on-angry-birds-space/>

ofstream - C++ Reference. (n.d.). Retrieved January 21, 2021, from <http://www.cplusplus.com/reference/fstream/ofstream/>

Collision callbacks - Box2D tutorials - iforce2d. (n.d.). Retrieved January 21, 2021, from <https://www.iforce2d.net/b2dtut/collision-callbacks>

8 Bit Space Groove | HeatleyBros. (n.d.). Retrieved January 21, 2021, from <https://heatleybros.bandcamp.com/track/8-bit-space-groove>