Uncle Stormtrooper

~By Swapnil Srivastava

Introduction

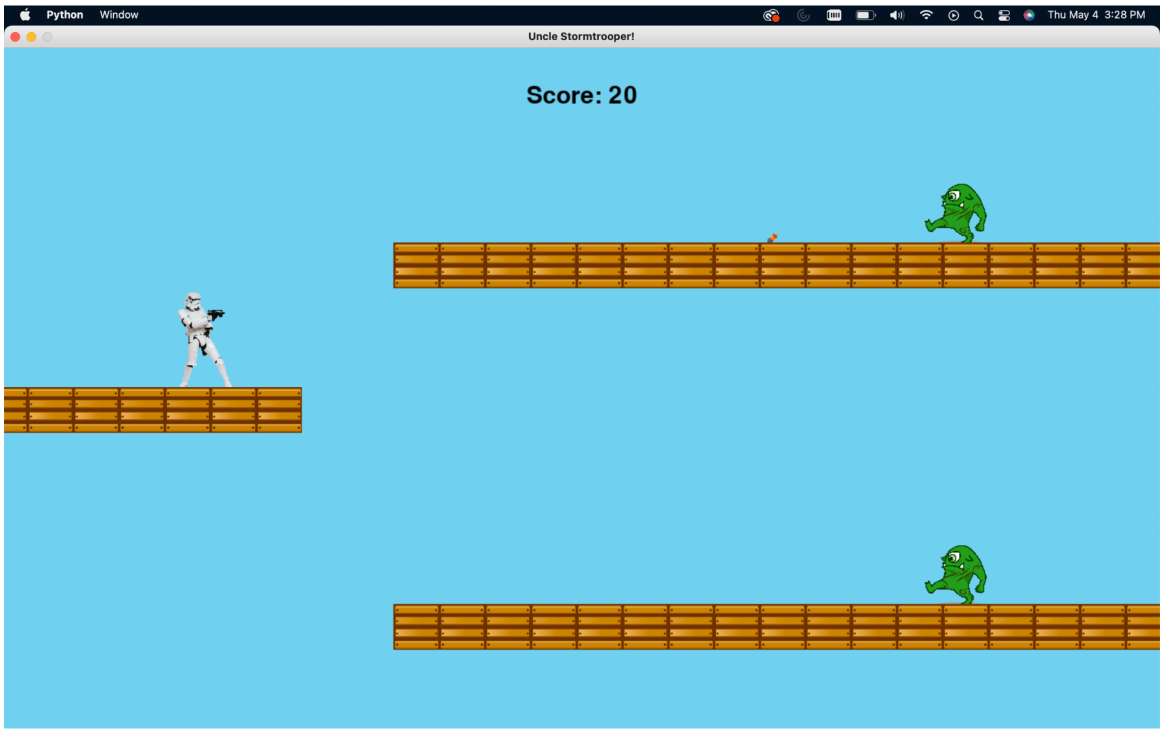
Uncle Stormtrooper is a 2D platformer game centering around a stormtrooper in his mid-life crisis. We must help uncle stormtrooper navigate his way through a horde of monsters to reach his beloved dog. This game's architecture and code were significantly changed in the last few days to make this game scalable. We hope to keep adding to the functionalities of this game to make it a complete 2d shooter platformer game.

The game can be found here: https://github.com/swapnil1198s/Uncle-Stormtrooper

Requirements: pygame 2.1.3 (SDL 2.0.22, Python 3.8.3)

Instructions:

- After downloading the files and required software, run the uncle_stormtrooper.py file to play the game.
- Avoid touching monsters and getting hit by them
- Use the teleportation grenade to teleport at the destination where the grenade lands.



Controls	**Action**
Up arrow key	Jump(tap when in air to double jump)
Left Arrow key	Move character to the left
Right Arrow Key	Move character to the right
Tab	Throw teleportation grenade (If thrown, teleport to location of grenade)
Mouseclick	Shoot

Game Design

Mechanics/Technology

- The gameplay loop housed within the main method acts as a controller of the game. The player inputs, game states, and updating of views is all handled by this game loop.
- The gimmick of this game is the use of a teleportation grenade. The player can teleport to the location of this grenade once it is thrown by pressing the spacebar.
- I think making the use of a teleportation grenade is not something I have seen in other platformer games.

Story

- Uncle Stormtrooper is a 2D platformer game centering around a stormtrooper in his mid-life crisis. We must help uncle stormtrooper navigate his way through a horde of monsters to reach his beloved dog.

Player Experience

- The player has to use good timing and coordination to get through the various map levels as falling results in instant death
- The player can shoot monster to increase their score
- The current score can be seen at the top of the screen
- Through the use of images to represent the various dynamic components of the game, the player’s experience is enhanced.

Game Design Changes

Although the original design for the game hoped for animation to be implemented, the tight timeline made it so that the functionality of the game was prioritized. The aiming mechanics for the bullets and grenades seemed to be challenging. They would work accurately to a certain extent but bug out in certain cases. So the aiming mechanics were omitted for this milestone. The game's architecture and code was edited in the last few days to account for scalability. We hope to keep adding to the functionalities of this game to make it a complete 2d shooter platformer game. Including the implementation of different bullet types with varying aiming mechanics. The biggest challenge of the game development so far was tackling the moving components in the game in line with physics.

Game Development & Documentation

- The main() method acts as a controller for the game and handles all events including player input.
- The Player class is the model for our player object that moves around and jumps.
    - This object has various properties such as speed, gravity, and vertical speed.
- The TeleportationGrenade class handles the view and model of the grenade object used for teleportation.
- The Box object is used to create the floor in varying combinations depending on the level.
    - generate_floor() handles the arrangement of the boxes depending on the current level
- The Bullet object handles the view and behavior of the bullets fired.
- game_over() function handles the view and inputs for the game over screen
- The Monster class provides the model for different types of monsters and their characteristics. It also handles the behavior and view for monster entities.
- The Bullet class models the behavior and handles view of all fired bullets.
- Tools used: VS code and github

Group Member Roles, Tasks, and Performance

All work was done by myself

Milestone 1:

Task 1: Player Controller:

- Inputs via keyboards are now working to control the player movement.
- Implemented Jump, but restrictions apply.
- Double jump mechanics implemented

Task 2: The floor and scene:

- The floor is implemented by a group of ‘box’ objects.
- Collision detection is implemented to prevent player from falling through the floor
- Design for other variations of the scene have been finished.

Task 3: Bullet mechanics:

- Bullet mechanics are not finished and will roll over to the next milestone.

Milestone 2: April 12

Task 1: Bullet mechanics: Deadline: April 11

- Implemented shooting mechanism upon mouse click.
- Updated bullet functionality for simplification.

Task 2: Monster mechanics: Deadline: April 11

- Implemented monster movement
- Implemented monster attacks and game termination on contact.
- Postponed termination of monsters to final submission

Task 3: Make map sections 1 to 3: Deadline: April 11

- Implemented new map sections based on player position.
- These update once the player moves out of the screen edge. Giving a feeling of a larger world.

Task 4: Polish up jumping mechanics”

- Jumps are no longer clunky.
- Double jumps are adjusted to fit the game map more appropriately.

Final Game Submission: April 26

- Implemented different monster types
- Implemented the teleportation grenade.
- Implemented win condition.
- Completed and polished game
- Completed Game Document game

Demo Video

There is a video named gameplay.mov running through the game at:
https://github.com/swapnil1198s/Uncle-Stormtrooper