ACG Interactive Game Project Midterm Report

Bowen Yang
ybw22@mails.tsinghua.edu.cn

Xinghan Li
xh-li22@mails.tsinghua.edu.cn

1 Introduction

In the course project, we choose the project's topic as an Interactive Game. The whole game is implemented using the Three.js language. We respectively introduce the basic concept, the play mode, and the schedule of our game in the following sections.

1.1 Game Basic Concept

The idea of the game is inspired by two classic games, *Snake* and *Zuma*. Specifically, each player controls a snake character, whose body is composed of a sequence of colorful balls. As the snake continues to move forward steadily, the player aims to control the orientation of the snake's head and avoid running into the walls, other snakes, or the body of itself.

Meanwhile, player can control the snake to shoot a colorful sphere bullet flying forward. If the bullet collides with one snake, it will be merged into the snake as one of its body balls. We also introduce the Match-3 mechanism as in game *Zuma*. That is, as long as three or more adjacent body balls are the same color, then they are matched and removed.

1.2 Play Mode

There are two agents in the battle of our game. We support two play modes, either Player versus Player or Player versus AI agent. The goal of two modes are the same. Each agent needs to manage to shoot bullets to the opponent's snake, aiming to match and remove body balls of the opponent's snake as many as possible. Each agent should also avoid collision between its snake's head and any other non-bullet objects in the scene. When any such collision occurs to a snake or all body balls of a snake are removed by Match-3, it dies and loses, and the survivor wins. In the case where two snakes collide head-to-head, the longer snake survives and wins, or ends in a draw if they have the same number of body balls.

1.3 Schedule

The whole schedule of the game project is to implement all basic game functions and complement visual and physical simulation details as rich as possible. The basic game functions include:

- Basic control of snake movements.
- Basic control of the shooting mechanism.
- Detection and response of bullet collisions.
- Growing and removing mechanism of the snake body.
- Snake death detection and game ending judgment.
- AI control algorithm for Player versus AI agent mode.

The visual and physical simulation details include:

- GUI interface for starting and ending the game.
- GUI components to provide useful game information for players during playing.
- Basic scene layouts.
- Diverse scene mechanism and objects for more complicated and versatile battle environments.
- Animation response to the player's control commands.

The technical points included in the project involve:

- (Done) A interactive control system that is user-friendly to learn but also challenging to win for more interesting gameplay.
- (Done) Local multi-player mode and easy installation.
- (Basic Done) Reasonable scene assets setting.
- (Basic Done) Reasonable UI interface for game control and game information.
- (Basic Done) Increasing difficulty level for the AI agent battle.
- (TBD) Animations of scene objects.
- (TBD) Built-in beginner's guide.
- (Maybe) Online access and online multi-player mode.
- (Maybe) Special visual effects such as motion blur to generate a sense of tension in battle.

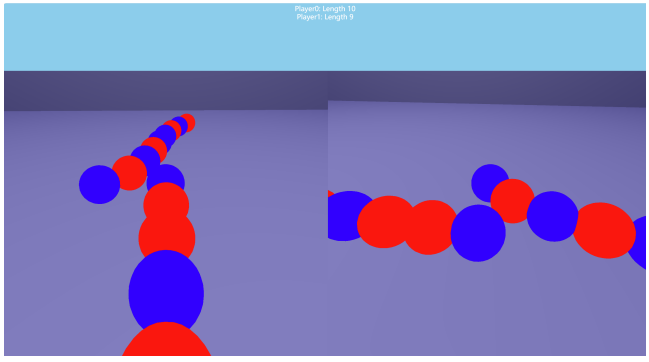

Figure 1: Visualization of our game. The screen is divided into half, each for a player to control its zuma-snake.

2 Current Methods & Results

We have already implemented the interactive control system and play modes in the project, and implemented the basic version of scene layouts and UI interface to support the whole game pipeline. One visualization result of the game is shown in Figure 1.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.1 Interactive Control System

There are three control commands for one player: turning left, turning right, and shooting. When continuously pressing the key of turning left/right, the orientation of the snake's head rotates at a constant angular velocity. When the key of shooting is pressed down, the snake shoots a bullet. Notice that the snake's head automatically moves forward at a relatively steady speed. Actually, the head's moving speed slowly increases, linear to the time of the game. The movement of the whole body is calculated from head to tail, that is, each body ball moves towards the updated preceding body ball. For the shooting, we add a transparent hint ball to show the color of the current bullet. The hint ball locates at the shooting target if targeting at a snake, or locates above its own head if targeting at nothing. We have also implemented a key to switch the color of the current bullet, but we remove it temporally for more challenging gameplay.

2.2 Play Mode

We have implemented the game ending judgment, as described in Section 1.2. Both single-player mode (player vs. AI) and multi-player mode (player vs. player) are implemented.

For single-player mode, we design an AI agent, who can automatically detect the obstacles in a local forward sector and greedily decide to keep moving straight, turn left, or turn right. When there are no nearby obstacles, it will actively turn to the opponent. When its current shooting target point locates at the opponent's body ball with the same color of the bullet, or locates at itself's body ball with a different color, it automatically shoots the bullet, without taking the flying time of bullets into account.

For local multi-player mode, we divide the display screen into half, each for a player's view. Two players share the same keyboard to control, one using 'w' (shoot), 'a' (turn left), 'd' (turn right) keys, and the other using the corresponding arrow keys.

2.3 Scene layouts

We have implemented a basic scene to support the game, composed of a square ground plane and its four surrounding walls.

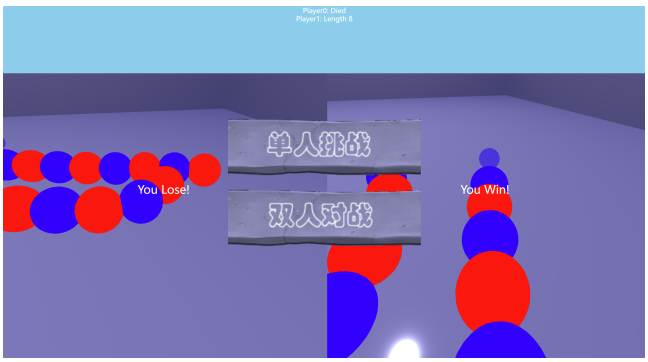

Figure 2: Visualization of the UI interface when the game ends. Status of snakes are shown at the top of the screen. The game results are shown at the center of each player's displays, respectively. At the middle are the restart buttons supporting choosing play modes.

2.4 UI interface

We have implemented basic UI buttons to start or reset the game with the chosen play mode. Besides, during the game, we display the status of each snake on the top of the GUI, and show game result messages for each player when the game ends, as showing in Figure 2.

3 Future Plan

The major task for the future is to complement the visual and physical simulation details. Current objects leverage the basic materials and geometries in Three.js. We decide to design more delicate materials and geometries in Blender, or search shared resources in online communities if the game requirements are beyond our mesh design ability. For example, some curtain is proposed to be hung in the scene to softly separate the battle area, inducing higher decision complexity. And a floating colorful fountain will be placed in the scene to randomly change colors of balls within a ring area. More animations will be added on snakes and those new objects. These variations are expected to induce more randomness for fun and more interesting visual effects. Besides, we will collect some public sound effects and background musics or design them using Fmod, for more engaging acoustic environment.

The UI interface will also be modified as a more fascinating and complete version, utilizing proper background and title images. A necessary built-in beginner's guide will be added soon. Since multi-player mode is already supported, an online version may also be supported if our resources are enough.