Deliverable #1

Matthew Kong

April 17, 2017

Contents

1 Project Proposal

For the term project, I propose to make a 2-player version of Zuma, where the two players cooperate to eliminate all the balls before they collide into each other. Single-player mode can also be chosen at the beginning of the game.

I propose to use Pygame for the main game structure, and use socket to implement the multiplayer mode, which would also include the _thread and the queue modules. Other modules that I will probably use include but are not limited to:

- math
- random
- time
- numbers
- ...

One problem that I can imagine now is how to implement the curvy path the string of balls should follow. I see several ways to solve this problem:

1. I can make the balls behave individually, where they "jump" according to a list of coordinates that represent the path. I can write a helper program to acquire the list of coordinates efficiently, and hard-code it into the path implementation.
2. Still making the balls behave independently, I can try to express the paths with trigonometric functions, and calculate the next positions for each of them, provided their current position.
3. I can implement a physical engine, where the balls are bound within walls that I can hard-code into the maps with the method mentioned in 1. Meanwhile, "Gravity" acts on all the balls and pull them along the path with an invisible and incline, collisions among balls themselves prevent them from reordering in the narrow path.
4. Similar to 3, I can implement an attractive force between successive balls, to emulate the backtracking effect when large chunks of balls are eliminated in the original version of Zuma.

Methods 1, 3, and 4 all involve hard-coding, but due to the nature of puzzle games, I believe hard-coding maps is most of the times inevitable, and thus acceptable.

2 Competitive Analysis