CS583_PingPongGame

Project Report

Yinchao Zhu, zhuyin@oregonstate.edu

Haoyuan Qiu, qiuha@oregonstate.edu

Shukan Nieh, niehsh@oregonstate.edu

A high-level overview

The domain of our project is gaming domain, the users should be players of all ages as it is a game with very simple rules.

In our project, users can control the bats up and down using the keyboard to catch and rebound the ball. Two modes are available in our games, one is player-to-player mode and the other is player-to-AI mode.

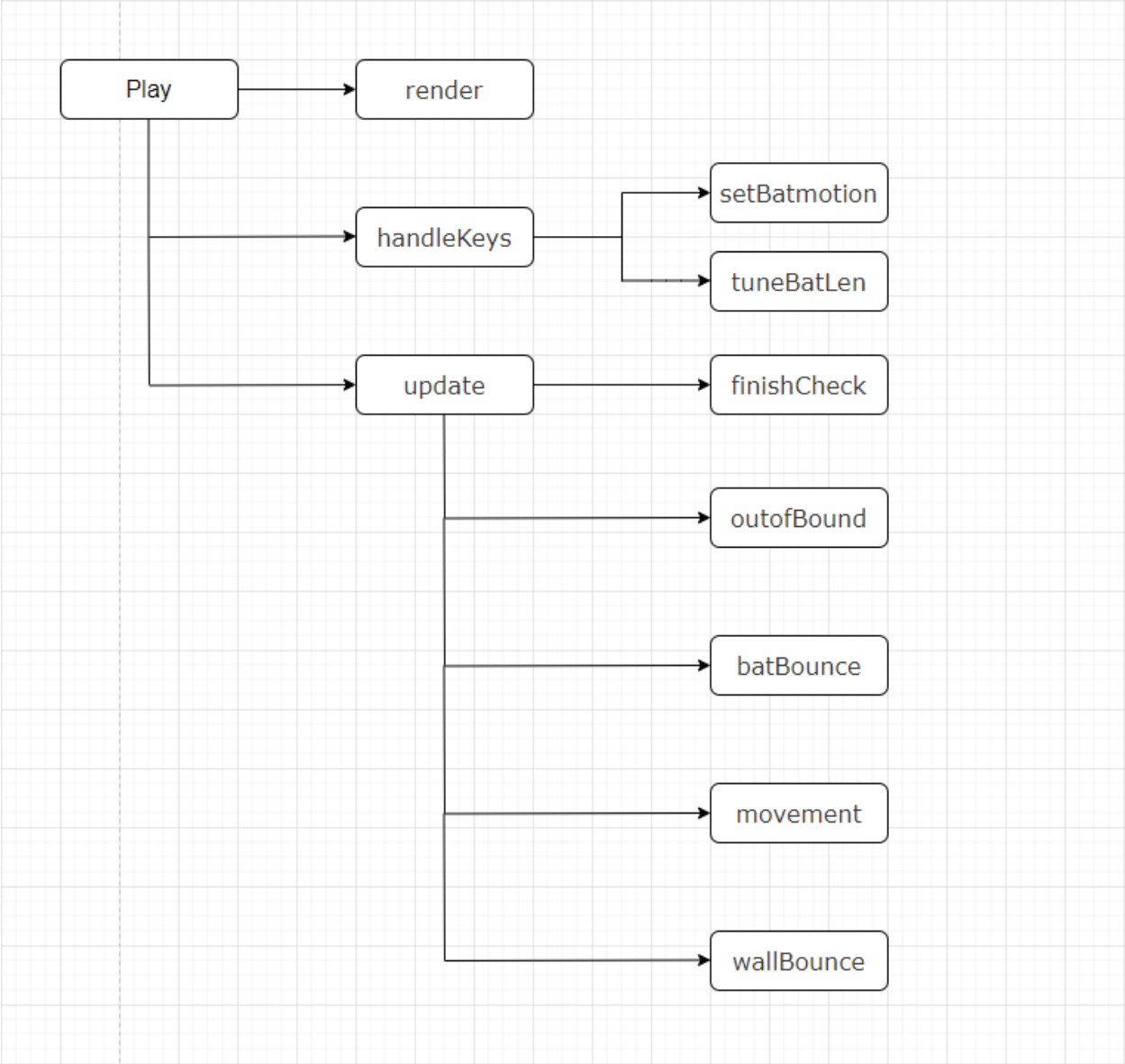
There are existing programs developed in some languages like JAVA, but we developed the game in Haskell language.

Our project is based on an existing tutorial:
https://andrew.gibiansky.com/blog/haskell/haskell-gloss/

However, we still add many self-designed parts, such as "changing scene", "AI control bat", "changing difficulty", "handle key", and so on.

The architecture and data types

[Architecture]



Here is the brief description of our functions, for detailed information please check the comments in the source code.

1. play :: Display->Color->Int->world->(world -> Picture)->(Event -> world -> world)->(Float -> world -> world)->IO ()

This function is the core of our project. It is a built-in function from the Gloss module.

2. Render :: PPG -> Picture

This function is to visualize the game state, so that users can see the text, and the geometric object.

3. handleKey :: Event -> PPG -> PPG

We use it to detect the specific key event, then update our game state PPG.

4. setBatMotion :: Bat -> Motion -> Bat

This function changes the state of the bat, either Stop, Up, Down

5. tuneBatLen :: Bat -> Bat

This function changes the length of the bat, and the length could be either 40, 60, 80, 100, 120.

6. update :: Float -> PPG -> PPG

This function updates the game by applying the rule, and the status of both ball and bats.

7. finishCheck :: PPG -> PPG

This function checks who wins the whole game by win_score.

8. movement :: Float -> PPG -> PPG

Based on the different modes, this function will update the position of the ball and bats.

9. outofBound :: PPG -> PPG

This function judges Win/Lose for each matchup.

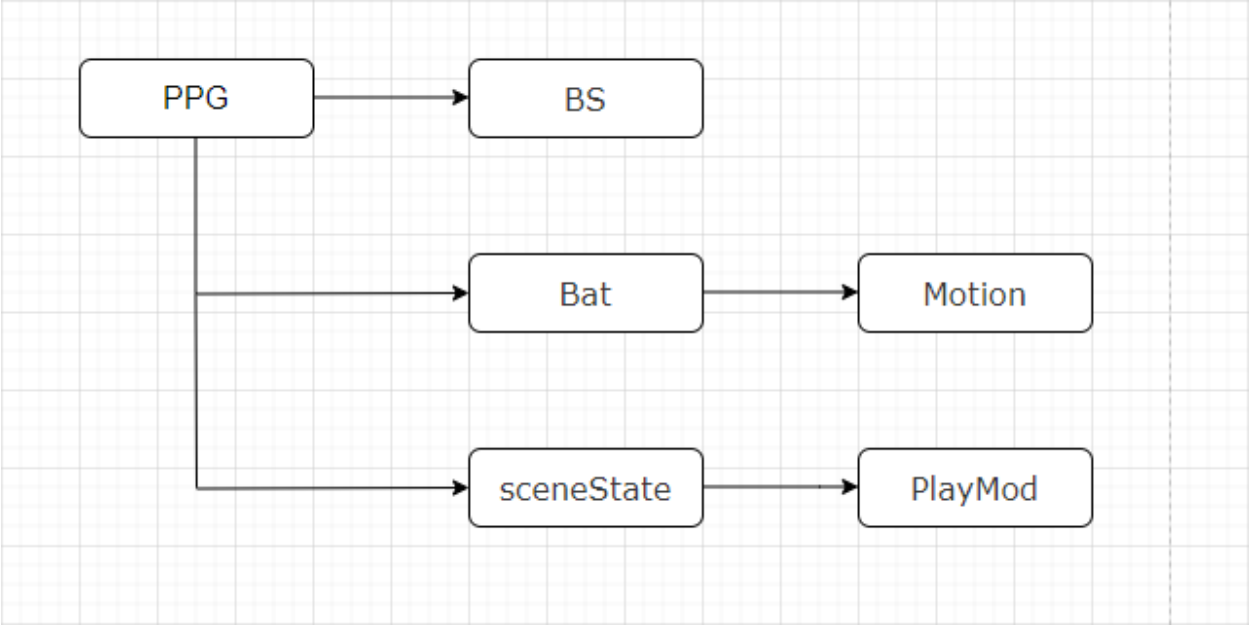
10. batBounce :: PPG -> PPG

This function detects a collision with a bat, and changes the ball’s direction

11. wallBounce :: PPG -> PPG

This function detects a collision with a wall, and changes the ball’s direction.

[Self defined Data Structure]



PPG - contains the essential game state of our project

BS - contains the data of ball state

Bat - contains the data of player

Motion - shows the bat's motion, including stop, up and down

sceneState - decides which scene are we in, including instruction scene, play scene and end scene.

PlayMod - decides whether the player would play with another player or play with AI.

Design Decision

1. We implemented the Gloss module to draw the graph and handle the key reaction. After researching some information, Gloss is the best choice because it is based on OpenGL but it is easy to use and can generate cool graphics in a short time.

2. In the beginning, we created a data structure “PPG” (game state) that includes all features because we can develop the functionalities of the game much more easily. After we have almost completed the game, we based on professor’s and Danila’s suggestions to split the original “PPG '' and compose some data into another data type, so that we can consider each data type as an object for maintaining the status of it. Therefore, “BS” (ball state) contains the position and the velocity of the ball. Also, “Bat” (bat state) includes the bat’s position, bat’s length, bat’s motion, and the score for the bat’s controller. By doing so, many similar functions could be combined together since the control method for two bat’s movements is similar.

3. According to Danila’s feedback, we realized magic number problems in our render part. Therefore, we make those positions related to the size of windows. Using some number like (window_width/2) to present the position. Besides, we apply a number called instruction_adjust to slightly adjust the position to ensure those instructions can be placed correctly.

4. When we developed the AI control method, the calculation is within the float type. It’s hard to imagine how to modify the type from float to IO float in calculating the bat’s position. Therefore, we still use the unsafePerformIO to obtain the random values.