Eric Tew
Chris Bentson
Nicolas Chaim
Matthew Romanelli

Coinball

Introduction

We have developed a multiplayer game inspired in the breakout game. Our game features support for two gyroscope-enabled remote controllers for a maximum of two players on the same machine. In addition to the traditional brick behaviour, some bricks uncover coins. To advance levels, the user is required to collect all coins.

Game Objectives

Gameplay is divided into two modes: single player and multiplayer. In the single player mode, there is a single paddle which is controlled by the player. In single player mode, you need to collect all of the coins in the level in order to move onto the next level. The final objective is to finish all levels. In multiplayer mode, two players each control their own paddle. A player can only collect coins if the Ogre ball hit their paddle last. Each player competed to collect the highest number of coins by the time all levels are completed. If you miss the Ogre ball, the level starts over from the beginning.

Game Objects

- **Paddle:** Located at the bottom of the board, players control one or two paddles to prevent the ball from falling though.
- **Coins:** Coins are collected and disappear when the ball hits them.
- **Bricks:** Each brick needs to be hit a specific amount of times before it destroys itself and optionally uncovers a coin. The color of the brick changes depending on the remaining number of times it need to be hit.
- **Obstacles:** Upon hitting an obstacle, the speed of the ball increases.

Controls

The game can be played with remote controls or with a single keyboard. When using the remote controls, the user points at the screen to move the paddle. Zoom in and zoom out keys are used for tilting the paddle to the sides. Also, the user can use the enter button to acknowledge level completion dialogs, and use the mute button to mute or unmute game sounds, independently of system audio. When using the keyboard, left and right keys are used to control the paddle. In multiplayer mode, the second player can use A and D keys for the same purpose. For navigating the game menus, the mouse needs to be used.

Menus and UI

Upon starting the game, the user is presented with the main menu which contains the following options: Single Player Mode, Multiplayer Mode, Settings, How to Play, and Exit. The Single Player button will take the player to the connection screen which gives the player the option to play with either a wireless controller or the keyboard keys. Selecting Multiplayer mode takes the users to a similar screen. The Settings button gives the option to disable/enable sounds and the How to Play button gives a brief explanation of the game objectives and controls. While playing, there is a pause game button which allows players to resume the game at any time, stop the game and go back to the main menu, and disable/enable sounds.

Software design

Our game is based on the Ogre game engine in conjunction with the Bullet physics engine, used via the OgreBullet wrapper. For sound, we use the OgreOggSound library. For supporting the remote controls, we use X Input Extension 2 (XI2), a part of X11.

Challenges

- **Remote controls:** The gyroscope-enabled remote controls, also called Air Mice, are normally used to control the cursor on presentations and Smart TVs, and are recognized as regular mice. Normally, there is no use for multiple mice on the same machine, as their motion cannot be handled separately. We learned that the latest version of the X11 API offered support for handling multiple pointers separately, however, there were no examples or extensive documentation, besides the original API function list with simple descriptions. It took significant trial and error to learn how to dettach the air mice from the system pointer, as well as how to read their independent raw motion data, instead of virtual pointer coordinate values that would be clipped to a specific window region.

- **Game States:** In order to organize gameplay code and make the UI more efficient, especially in determining if the game was in single player or multiplayer mode, we had to develop a system of game states that kept track of if the game was currently in menu mode, single player mode, multiplayer connection mode, pause mode, etc.

- **Brick colors:** Changing the colors of the bricks programmatically was not straightforward. Our solution uses shaders, with a custom parameter we control to change colors. The initial CG shader was not supported on our Ogre version, whereas our final GLSL shader has to be adjusted several times so it worked on all of our machines.

Planning adjustments

Within the time scope of this project, we successfully implemented all of our primary proposed features, except for particle effects. To enhance gameplay, we added bricks, which must be hit a certain number of times before being destroyed and potentially uncovering a coin.

Division of Labor

- **Eric:** Gameplay and playability adjustments including camera control and ball movement tuning
- **Chris:** Code refactoring, design and implementation of levels
- **Nicolas:** Support for input devices, managing Bullet physics objects, basic brick implementation
- **Matthew:** CEGUI, UI, and use of game states
- **Everyone:** Game Testing and code reviews, bug fixes, general planning and brainstorming

Conclusion

In the end, we are all happy with the game that we have produced. We were able to integrate wireless controllers, multiple game modes, CEGUI graphics, Bullet physics, and various game objects into our code to create a great, playable game.

Game Trailer

Available on https://youtu.be/Yslpye2h_iM