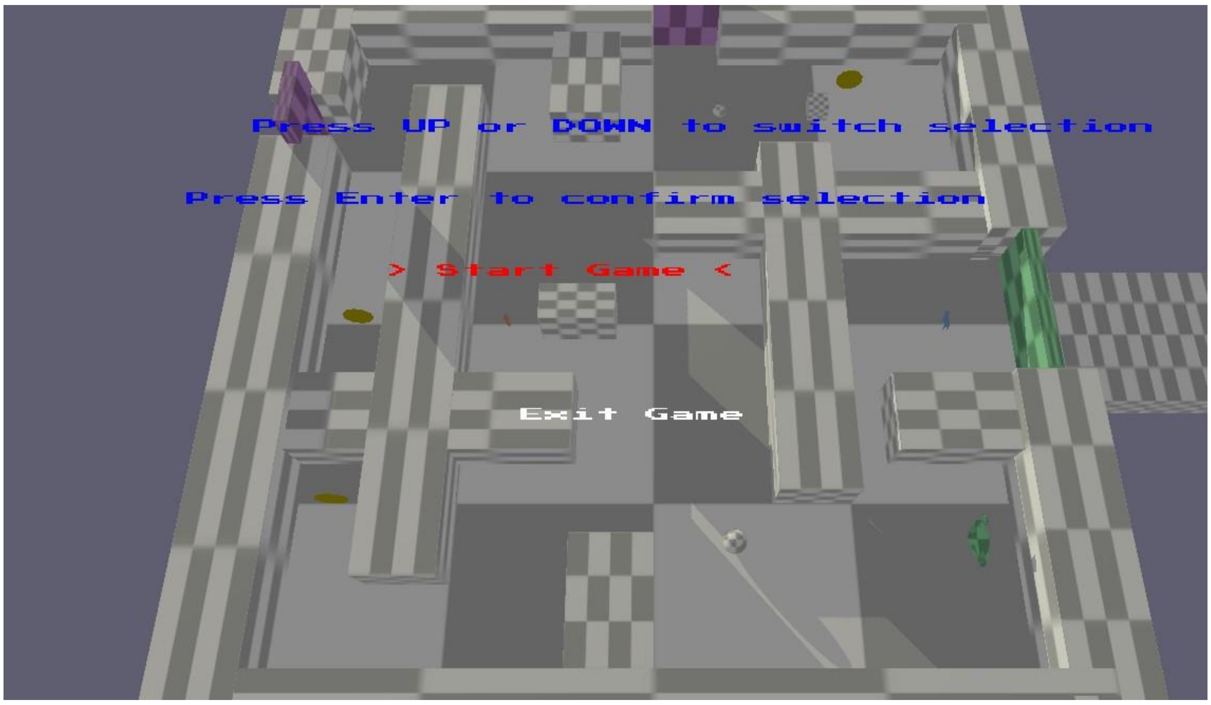
CSC8503 Coursework description

YouTube Link: CSC8503 (youtube.com)

Menu:


Press UP or DOWN to switch selection

Press Enter to confirm selection

Menu is down by pushdown automata. I created 6 pushdown states. Including Win state, lose state, game start state, exit state, menu state (start game selected) and menu state (exit selected).

The first state is menu state which start game is selected, After UP or DOWN pressed, exit selected (still on the Menu page) is pushed. If ENTER pressed push the exit state and the window stop update, the program exit. In the menu page the scene paused.

ExitState

Push to the top if ENTER pressed.

MenuState(ExitSelected)

MenuState(StartGameSelected)

On the exit selected state, if UP or DOWN pressed, exit selected state is popped, back to the start game selected state.

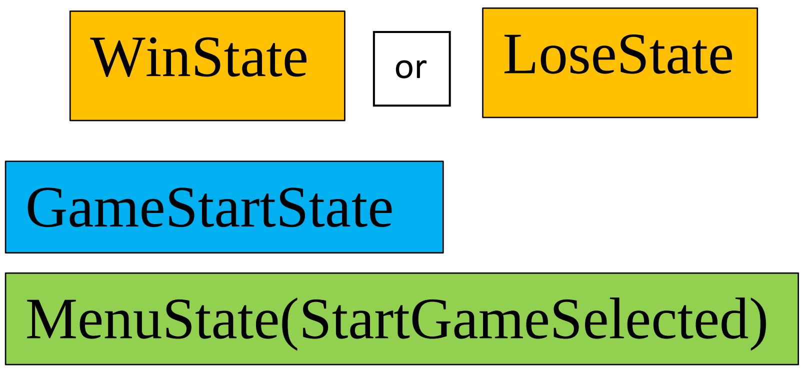
MenuState(StartGameSelected)

On the start game selected state, if enter pressed, start game state is pushed.

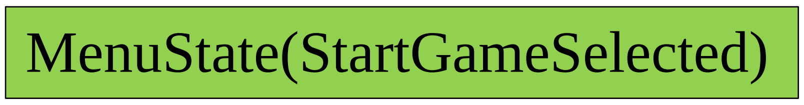
GameStartState

MenuState(StartGameSelected)

In the game start state, if win the game, win state will be pushed. On the other hand, if lose the game, lose state will be pushed.



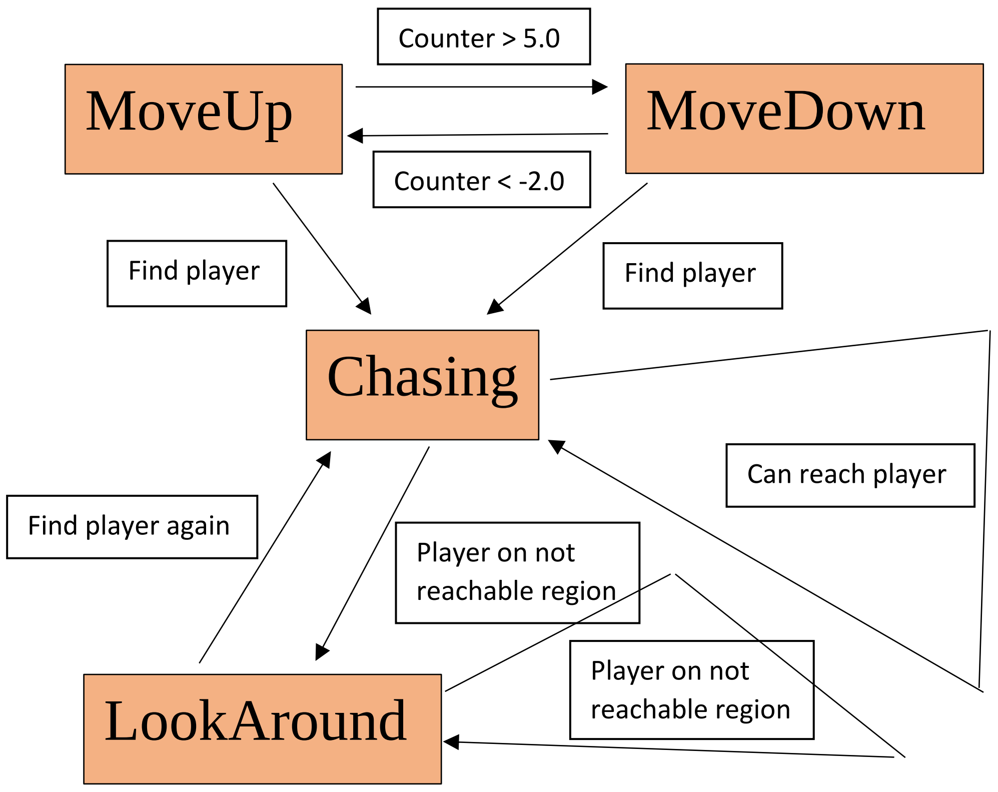
On the win or lose state, if ESC pressed it will *Reset*. I added a *pushdownResult Reset*, which will return back to the initial state (menu state, start game selected state) and before *Reset*, everything in the scene will be initialised.



I not only added the *Reset* to the *pushdownResult*, but also changed the *NoChange pushdownResult*. On the no change push down result the state will OnAwake itself, so the state is keeping running the OnAwake but not only call OnAwake once. This make sure the scene is keeping update and would not cause conflict with other states.

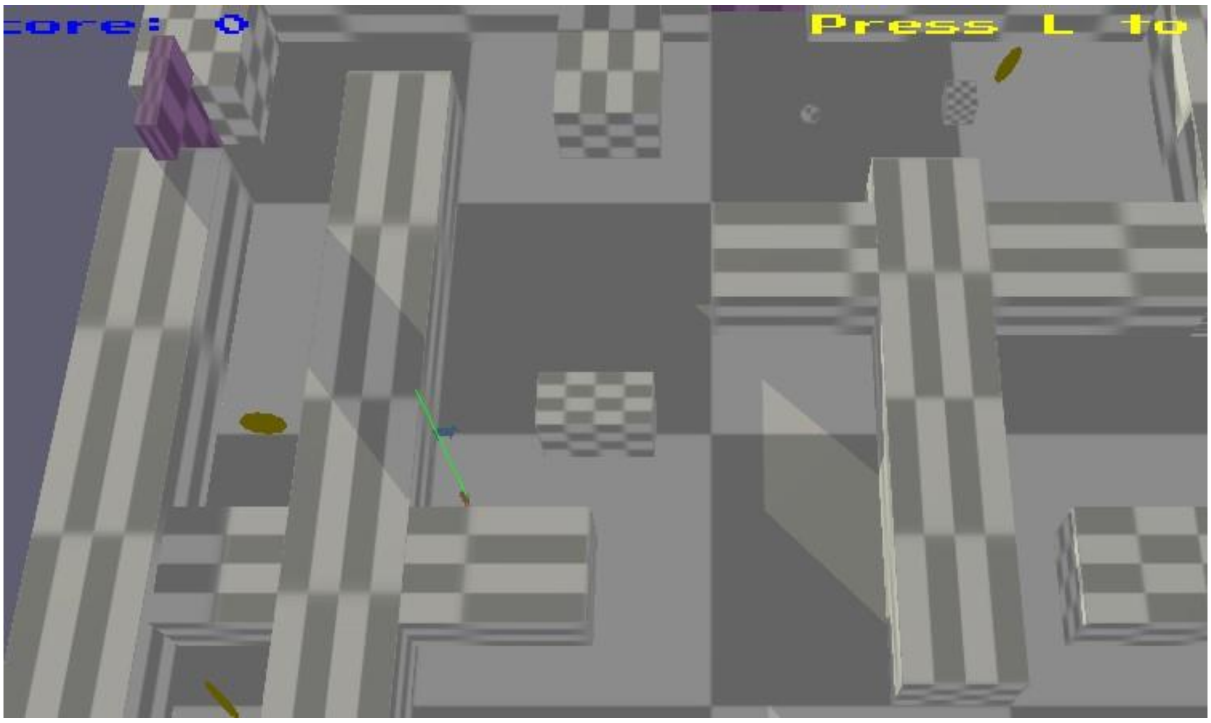
AI and pathfinding:

In the game, I create an AI enemy with state machine. There are four states.



The state machine is shown above. The initial state of the enemy is just move up and down. If the player spotted by the enemy, the enemy would start to chase player.

There are two ways for enemy to spot the player. The first is that the player close to the enemy, in the range of 30< *enmeyPosition.x* <30 and 30<*enmeyPosition.z*<30. The other way is that the enemy "see" the player. The enemy uses ray cast, the ray's direction is the enemy facing direction, and the origin position of the ray is the enemy position.

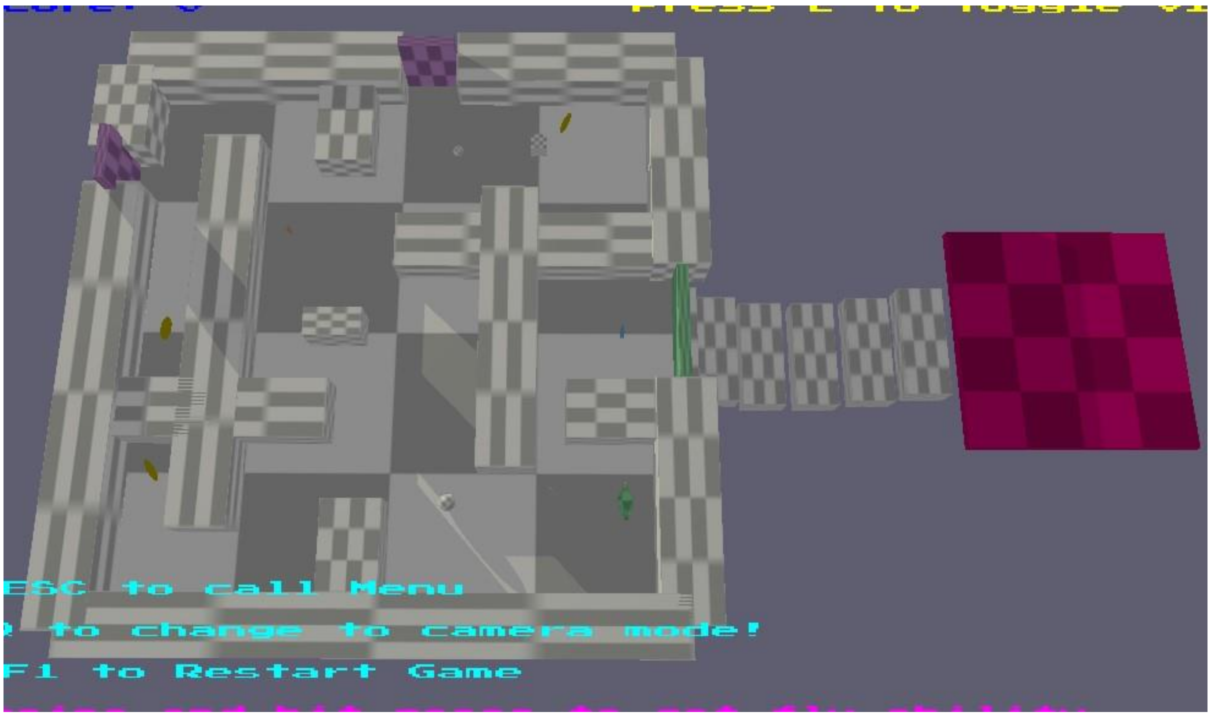


Pathfinding, green line is the next path.

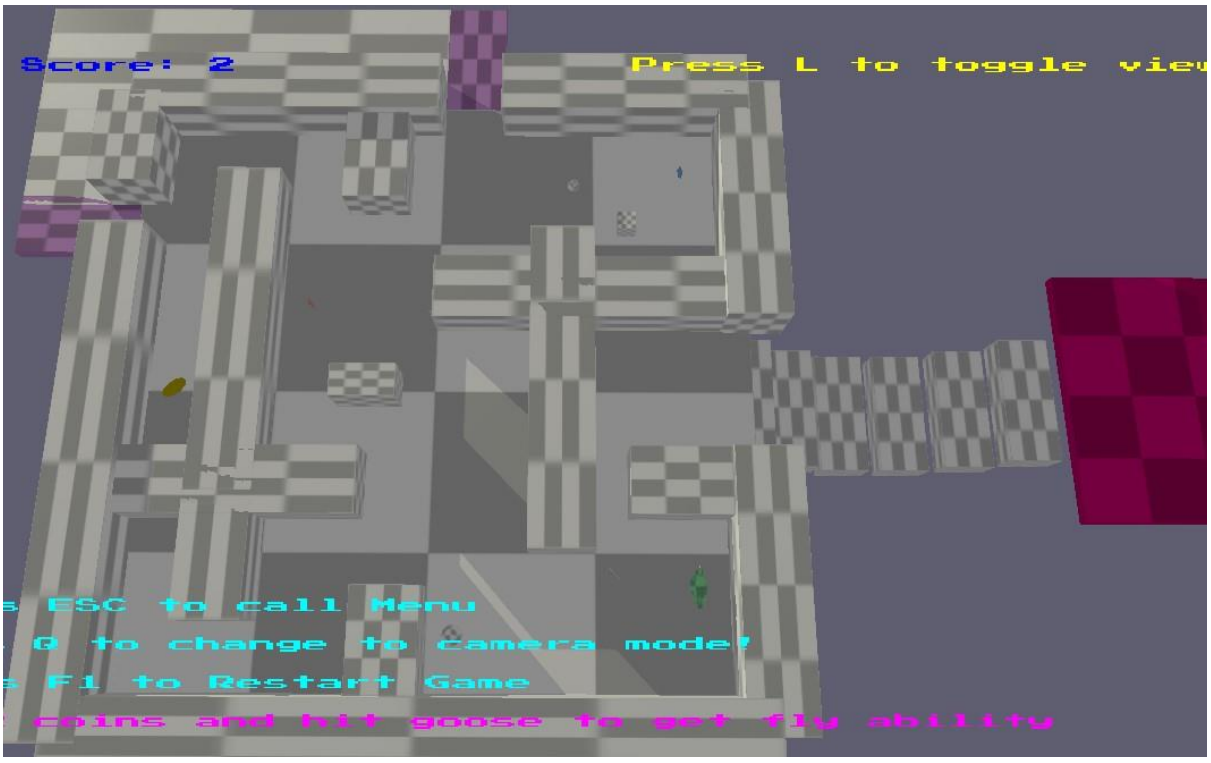
The enemy could keep chasing the player if player is spotted. To achieve that, if player is on the reachable position the state will call itself, so as long as the player can be catch, the pathfinding method will get the current position of the enemy and the player to provide the path for the enemy. If the chasing state not updating the enemy only follow the path of initial positions. For each time execute the chasing state, the enemy only move from the current node to the next node, but not go through all the path nodes in on frame. It will cause many different directions of force add to the enemy at once which can't correctly follow the path and move along with the path each time.

Because of my map and gameplay design, the player could get out of the map or the map.txt which is my grid file not drawn very accurate so the enemy can not find the path in sometime so I added a look around state, the enemy would stay and rotate around to looking for player. If player spotted again, the enemy start chasing.

Map and gameplay:



The blue goat is the player, the brown person is the AI enemy. There are three golden coins in the maze, pick up each coin can get one score. If the score reaches two, the two purple doors will rotate to become the floor and additional floor added on the top left outer the maze. Also, the green door will open. If the player hit the green goose on the bottom right of the maze, the player can get the ability to fly.


Score: 2
Press L to toggle view
ESC to call Menu
Q to change to camera mode!
F1 to Restart Game
coins and hit goose to get fly ability

New Map

The enemy can’t reach out of the maze, but the player can. The player can use the new route to steal the coin guarded by the enemy.

The bridge between the maze and the destination floor uses constraint.

If the player is catch by the enemy or drop down the ground, the game loses.

If the player reaches the red ground on the right side the game win. The player must get at least two coins to open the green door and get the fly ability from goose to win. So, get at least two scores to win the game.

Collision information

Sphere Collision Objects: player (blue goat), green goose, some spheres in the maze.

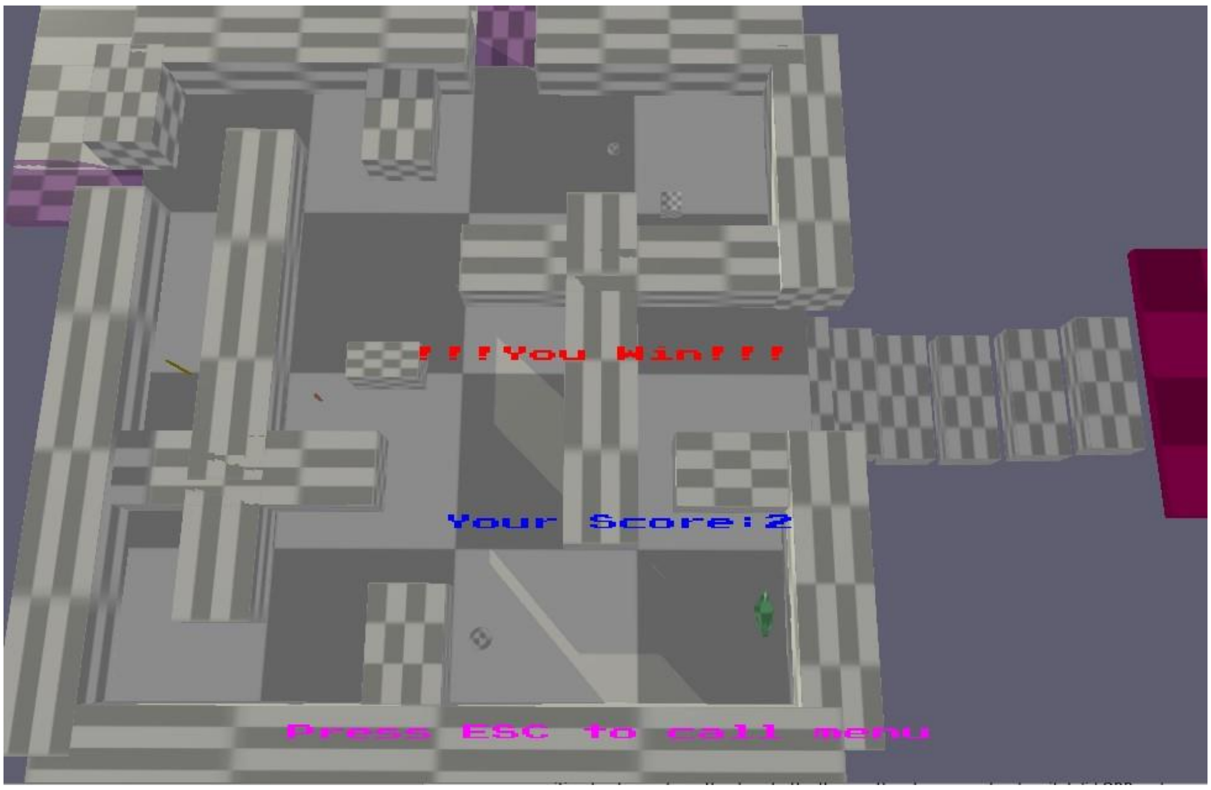
AABB Collision Objects: Walls, floor, destination (red floor), green door, Coins.

OBB Collision Object: two purple doors.

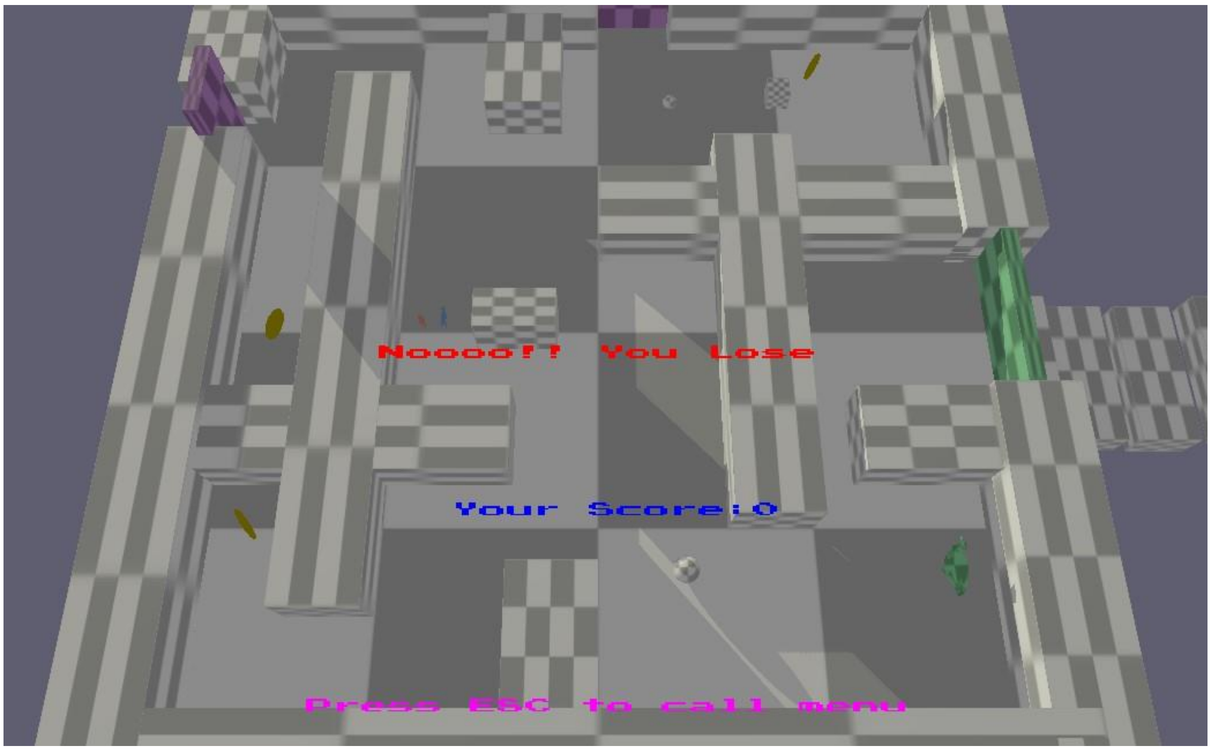
The two purple doors using OBB collision detection, because it will rotate and change position to change from the door to the floor or the player can stand on it. I did OBB and sphere collision detection so the player can collide with the purple door after it rotate to the floor.

External Information:

The win screen:



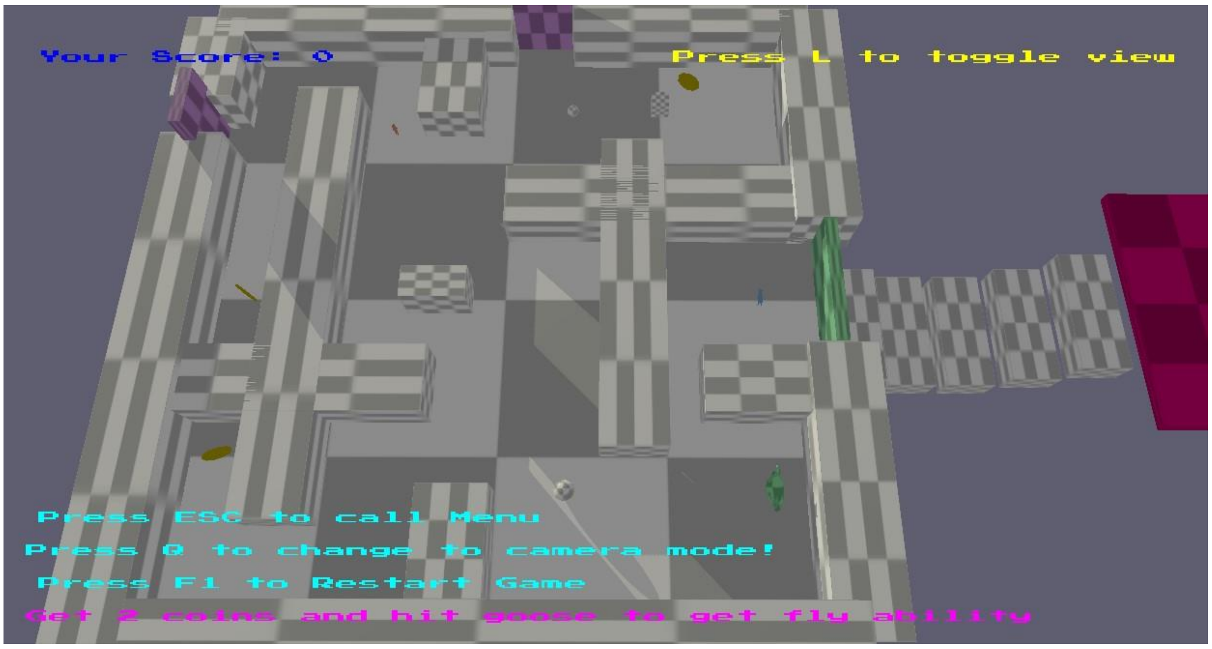
The lose screen:



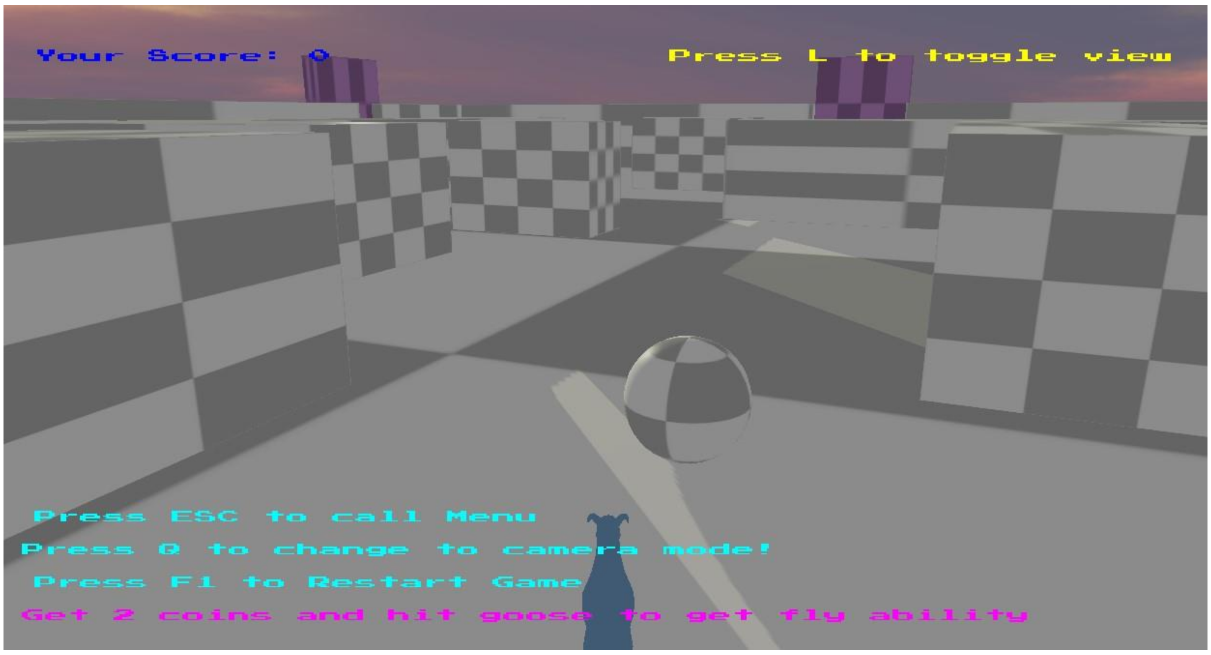
In the menu page and win or lose page, the scene will stop updating, the game will pause.

Press L can toggle camera to switch the view port, in different view port player control are a little different, in the overall view or see from sky, player control is just adding force to up, down, left and right position. In the first-person view, left and right control is adding torque to the goat and change orientation, moving forward and backward control are in goat's forward and backward direction. In different view player controller has different effect and more suitable for the player to control the goat.

In sky view:



In first person view:



Key and Mouse Controls:

In menu page:

Key UP (up arrow) and DOWN (down arrow) to switch selections.

Key ENTER to confirm selection.

In the Game:

Key ESC to call menu.

Key F1 to restart game.

Key L to toggle viewport.

Key Q to toggle selection and camera mode.

WASD, SHIFT and SPACE to move camera:

Key W move camera forward.

Key S move camera backward.

Key A move camera leftward.

Key D move camera rightward.

Key SHIFT move camera downward.

Key SPACE move camera upward.

UP, DOWN, LEFT and RIGHT to move player:

Key UP (up arrow) move player forward (upward).

Key DOWN (down arrow) move player backward (downward).

Key LEFT (left arrow) move player leftward (rotate leftward).

Key RIGHT (right arrow) move player rightward (rotate rightward).

Key ADD move player upward (after getting 2 coins and hit the goose).

Video Link:

https://youtu.be/R-OcFsoGxfk