Description

We have made a two player ping pong game using drRacket's inbuilt graphic libraries : 2htdp.
Rules of the game are standard.
The game is played till a player reaches 10 points.

Overall Idea

The main objective was to simulate the game without exceeding the default memory limit provided by drRacket . The concept of BigBang , which is based on updating an existing world at triggers like clock ticks, key presses, key unpresses served as an apt solution to our initial objective.

The flow of the game is as follows :

1. Main Menu

On running the main program, this is the first screen. It contains the instructions to play the game, along with the options to start the game and exit it. Pressing 'z' takes us to the main game, described below.

2. The Game

As it is a 2 player game, there are 2 paddles , one for each. The left paddle is controlled by the 'w'(for moving up) and 's'(for moving down) keys. The right paddle has the up and down arrow keys as the corresponding controls. The score board on the top keeps track of the no of points of each player so far. At the start of a game, the left player serves using the spacebar. The player gets a point if he is able to get the ball past the other player's defences. In the next round, the losing player serves and the serve can be again initiated by the spacebar irrespective of the player. For the unsporting player, the 'g' key can be used to quit the game midway in case of an impending defeat. The serving of the ball results in random values assigned to the ball velocities on every serve. Some special effects are :
Speed of the ball determines its color, it varies from yellow/orange shades(low speeds) to reddish hues(breakneck speeds).
Ball direction can be changed by the player by hitting it in the appropriate position on the paddle. Hitting on the upper side of the facing edge results in giving more velocity of the ball in the upper direction and similarly for the downward direction too.
The moment a player reaches 10 points, the game goes to the next stage, where the result is displayed.

3. The Result
The result displays the winner and provides an option to take you back to the menu to replay the game(which you would want to anyway ;)).

The Implementation

The code is modularised into three files: ping.rkt which has the main code, update.rkt which has the updation functions and constants.rkt which stores all the necessary constant for the game.

The main function simulating the game is the “big-bang” which is a part of 2htdp library. It takes in six arguments : The world(which is to be manipulated at every step), the “next” function which specifies the updation of the world on a clock tick(approx 1/28 second), the “press” function which specifies the actions to be taken on pressing any key, the “release” function, which specifies the actions on releasing any key, the “draw-world” function which draws the world at any instant, and finally the “stop” function , which specifes when to quit the game.

“WORLD” is an object of the mutable structure “world” used to maintain the different states of the game: y-coordinates of the paddles, position,colour and velocity of the ball, simulation count, presence of the ball, scores of the players, player who won the last point as well as the player who won the game and finally the state which denotes the game has been quit or not.

1. When the clock ticks :
There are three updations taking place :

a. “update-bars”
It takes in two arguments, the world and the hashtable for the keystates, and using this information accordingly updates the positions of the paddles in the world.

b. “update-ball”

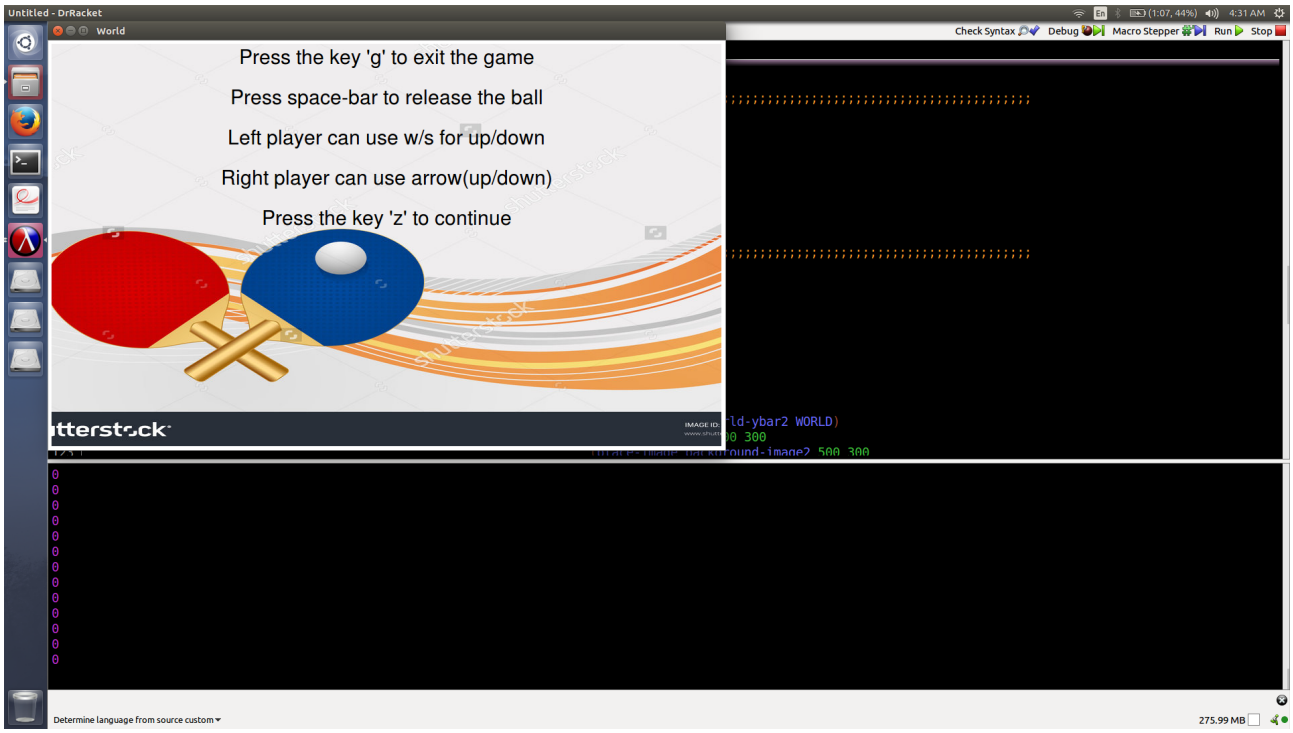
It updates the position of the ball on each tick according to the current velocity. Also, the collision with the paddles and the walls are implemented here. The collision with the walls follow the natural laws of reflection, while the collision with the paddles have been customized to make the game more exciting. Every collision with the paddle changes the vertical velocity of the ball by a factor proportional to the difference of the center of the ball and that of the paddle. Also Edge sensitivity has been implemented to ensure that the ball gets reflected even if the ball is caught on the edge upto a distance of (the radius of the ball)/3 .

c. “update-color”: The color of the ball is updated in this function using the inbuilt “color” struct. The red and blue components are set to 255 and 0 respectively while the green component varies inversely with the speed of the ball. This leads to the effect that reddish hues represent high speeds whereas yellowish hues represent low speeds.

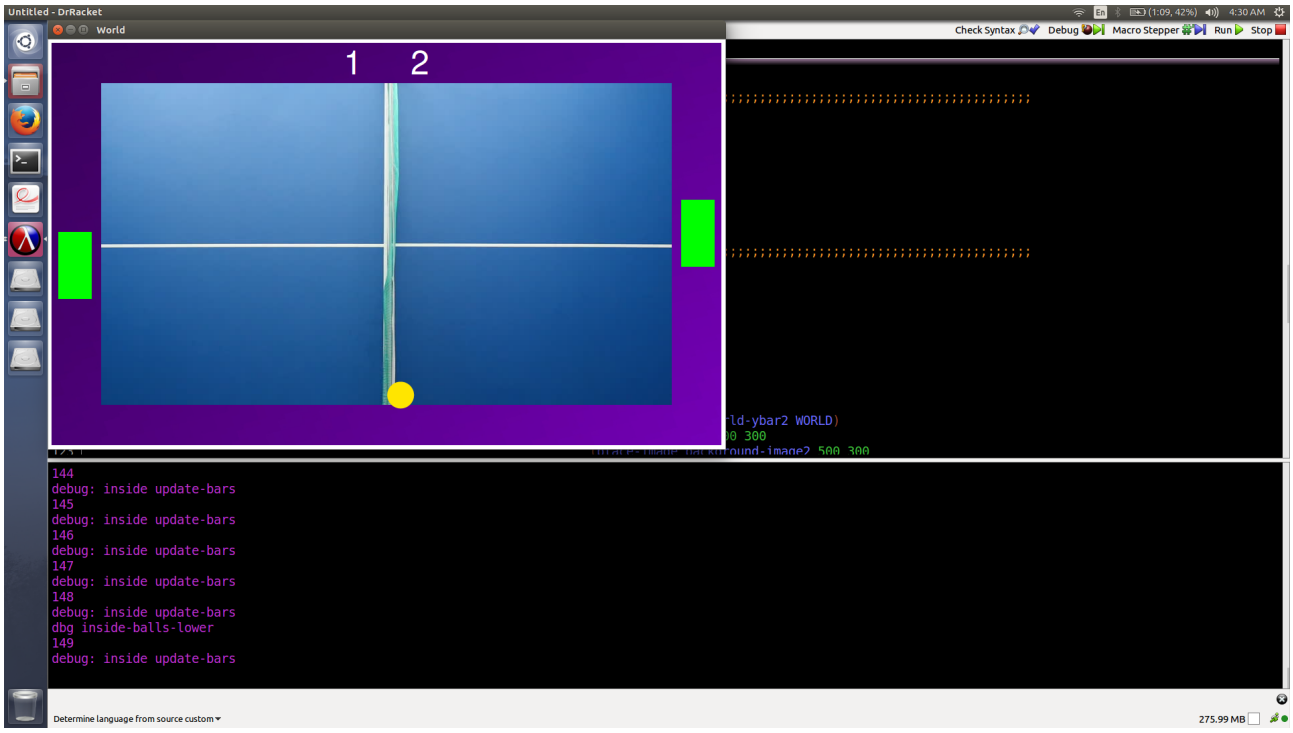
2. Keyboard actions:
Initially, for the paddle movement, the keys were mapped to the corresponding actions. **But, there arose a problem of only one key-press being executed at a time (in case of simultaneous key-pressing).**
To solve this problem, the ***idea of a hash table*** was incorporated to maintain the state of the various keys(a key is either in a pressed state or unpressed state). Then on the next clock tick, the hash table is referred and accordingly, the actions corresponding to the states are executed. This enabled simultaneous key pressing(as both the players may want to move their paddles at the same time instant).
The output of various keys, when pressed, was explained in the game flow.

3. Graphics :
The 2htdp library was priceless in this area
The gaming arena is a window simulated by using 2htdp/image .
There are three basic screens, one for the menu, one for the gameplay, one for the result. The screens are setup using nesting of “place-image” functions. The ball is a circle with varying hues (depending on speed as mentioned before) and the paddles are implemented as blue rectangles.

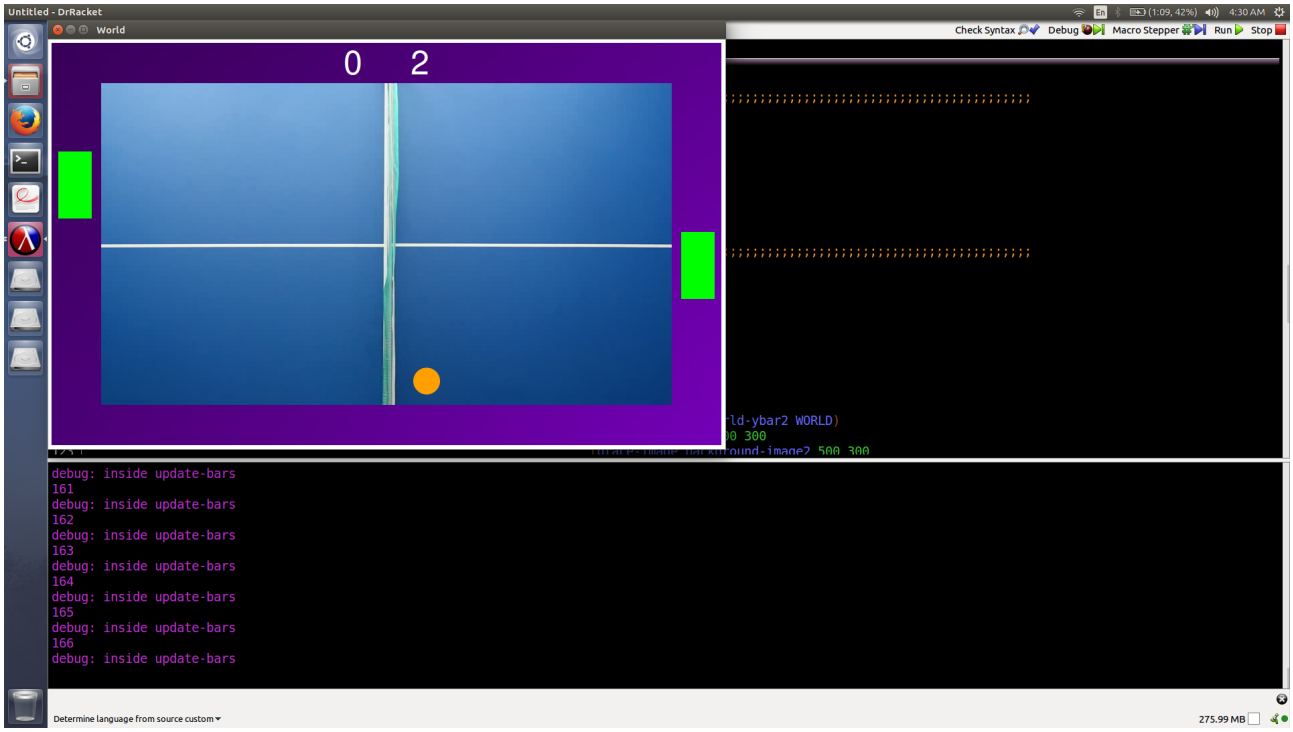
Screenshots



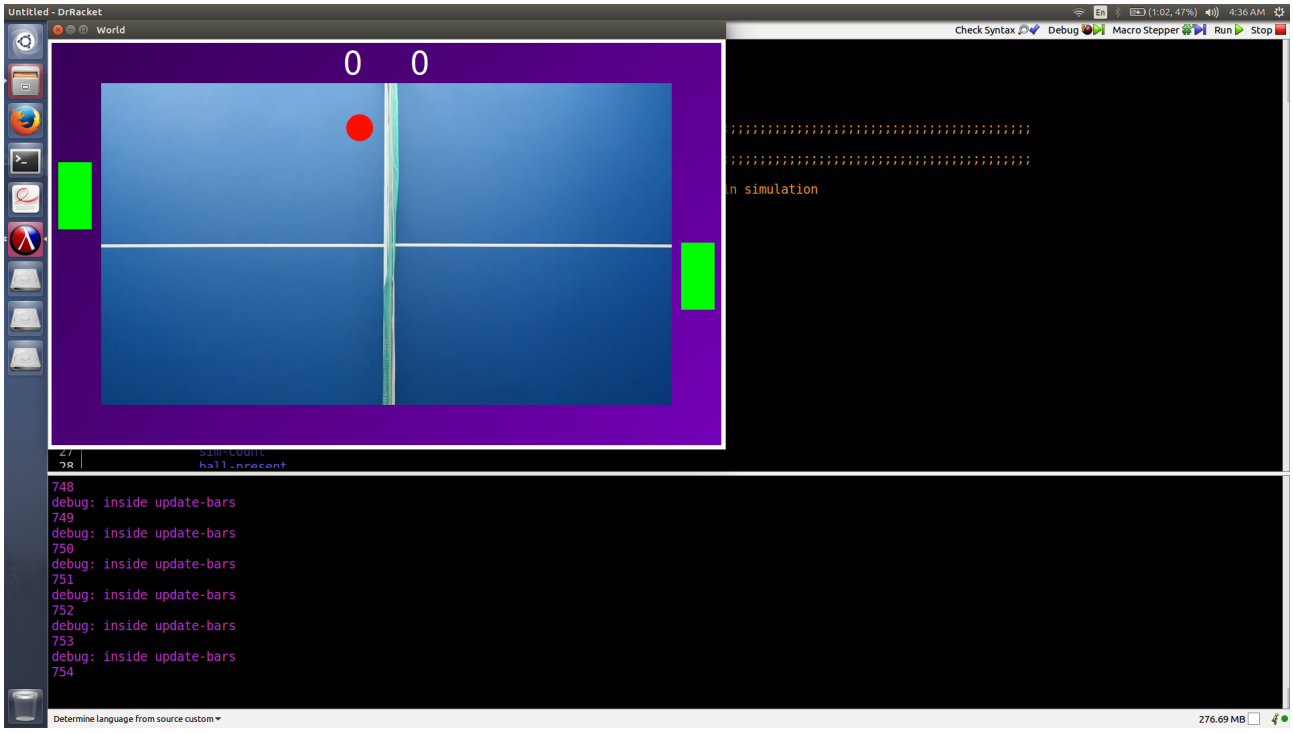
Start Menu



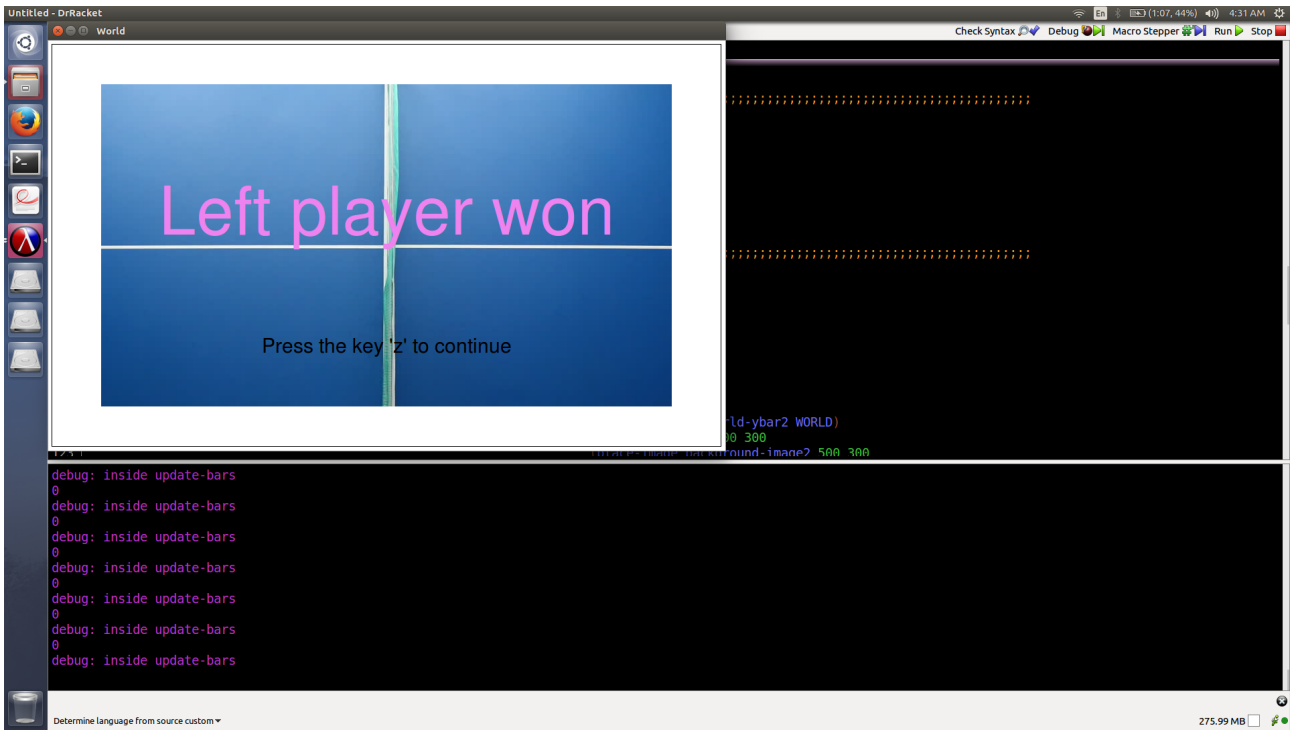
Ball speed low



Ball speed medium



Ball speed high



Result

Limitations and bugs:

To make a particular round of game more competitive, we tried to increase the horizontal speed of the ball using the simulation counts (Note that the collision with the paddles only affected the vertical speed of the ball). We realised that the ball started “tunneling” through the paddles. This was mainly because the definitions of collision with paddle fails to hold as we increase the horizontal speed of the ball.