COP290: Ping Pong Game

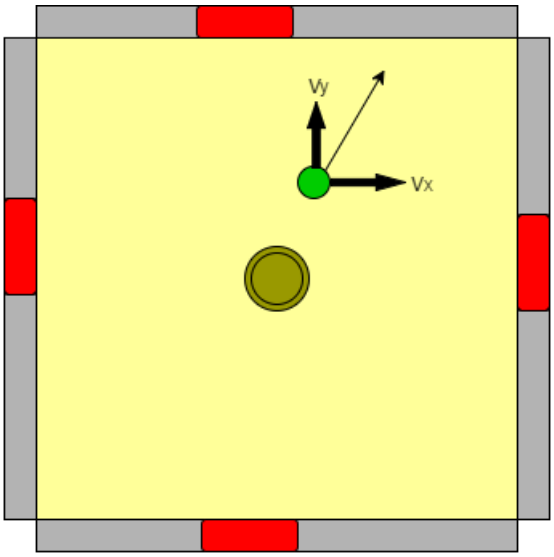
Abhijeet Anand (2014CS10202)
Kapil Kumar (2014CS50736)
Sourav Das (2014CS10258)

April 27, 2016

1 Introduction

In this assignment we will be designing and implementing a networked multi-player Ping-Pong game that can be played on a desktop machine. The game can be played by a maximum of four players (less than four is also allowed). We can specify the number of players in the game as well as the number of computer players in the game. The basic aim of the game is to prevent the ball from touching your wall and make it do so for other players. The players whose wall is touched by the ball 3 times is considered ”dead” and the paddle is removed from the game board. The player remaining till the last is considered the winner.

2 Description



Our game starts with a screen which displays the "Ping Pong" i.e. name of the game. After the name screen we show the users a screen containing a menu out of which the user has to choose a type of game he/she wants to play. The various options in the menu are

1. **Quick Play:** This part will be the complete online game. Where Random people connected to a server will come together and will play the game. The inclusion of computer player will be on vote basis. If the more people out the total vote for a YES to computer player than computer player will play or else not.

   This is also one of the interesting feature of the game. Where you will play with random people around the world. When a user tries to play a quick play game, we connect the user to our central server and find the users who also requested for a quick play at the same time and make a connection between them and let them play.

2. **Solo Play:** Solo play will be an offline version of the game which will run even without the internet connection. In the solo play the user has to select the number of computer player (1 or 3) after selection the user can start playing.

   When the user clicks on the Solo Play option it navigates the user to a page where the user can choose the number of computer player he wants to play with 1 computer player or 3 computer player. After selecting the number of computer players he has to select level of the game. After selecting all the required things he/she can start playing the game.

3. **Multiplayer:** In a multiplayer game one of the user has to start a local network and have to host the game. Other people can join his game by connecting to his/her network locally and can proceed the game. Here also the users will have a choice to include computer player or not.

   This facility is provided to play the game locally using LAN or Bluetooth service. In these, one of the user has to host the game and rest can join the game and start playing. The decision regarding the computer player will remain in the hands of userZero.

4. **Advanced:** This part is for those people who want to play with their friends but they live far apart. In these part they can login to game using their account and invite other friends to their game. After the invitation is send the receiver will get 2 minutes time to either accept the challenge or to deny it.

   For now this is an optional feature and we will implement this in near future.

5. **Options:** This option will be more related to user information update thing, where user can update their username and also change their password if they want. It will also provide the user to invite other friends to use or download the game.

6. **Exit:** This will be the classical exit which terminates the game.

3 Scope of the game

1. **Extra Balls:** We will be providing a feature to choose the number of balls used in the game. Since all the players are equal they get to vote in choosing the number of balls. In case of a tie the game kicks on with one ball.

2. **Splitting Balls:** The game contains a feature that each ball splits up into two after a certain period of time. This will help in making the game a bit interesting.

3. **Power Ups:** The game contains a feature that each ball splits up into two after a certain period of time. This will help in making the game a bit interesting.

4 Physics of the game

We are implementing the game using real life physical equations to make this game more realistic and user- friendly.
The physics we have thought of implementing in the game are as follows:

1. **Collision between the paddle and the ball:** The collision between the paddle and the ball is completely elastic. In this way the component of velocity perpendicular to the paddle remains constant before and after the collision. The final velocity of the ball parallel to the paddle will depend upon the initial velocity of the paddle and the ball. For this, we will use the principle of conservation of momentum.

2. **Collision between the ball and the wall:** The collision between the ball and the wall is completely elastic. This equation combined with the equation of conservation of momentum can determine the final velocity of the ball. The final velocity perpendicular to the wall will be equal and opposite the initial velocity perpendicular to the wall. The final velocity parallel to the wall will remain the same as before. In this fashion, we can determine the path of the ball after collision.

3. **Speed of the paddle (for computer players):** The deviation of the speed of the paddle from an ideal situations will be generated randomly depending upon the difficulty level. For example, suppose v is the required velocity for the paddle to hit the ball in the middle. So, in case of easy level the velocity may vary from 0 to $2v$ while in difficult level it may vary from $0.8v$ to $1.2v$. In this way we can control the gameplay of different levels in the game.

4. **Corner collision case:** In the normal scenario if the ball collides at the corner then it will retrace its path and it will continue doing so forever. To check this case, we will provide an angular deviation of a small angle (*say* 10°) so that the game carries on smoothly.

5. **Increasing speed of the ball in case of a power up:** If the ball touches a boost power up then the speed of the ball will increase abruptly while the direction will remain the same as before. If the ball touches a wall and a player loses a point the power up effect will be gone.

5 Algorithm for the computer player:

The key of the computer algorithm of our game is that the "paddle will always follow the ball not with absolute perfection but yes it will follow the ball" with slight variation in horizontal velocities of the ball and direction same as the ball's current direction. More elaborately:
Let us say that we are playing in level i and the current velocity of the ball in the corresponding level is

$$v = v_x + v_y$$

where,

$v \rightarrow$ *net velocity*$v_x \rightarrow$ *current velocity in x direction*$v_y \rightarrow$ *current velocity in y direction*

Case I: 2 Player Game

Since the board is frictionless as of now the vertical velocity of the ball in the whole level will remain same which will be a measure of the difficulty. And now the main part of deciding the fate of the game is the allowable range of horizontal velocity the paddle can achieve. Let us say that the horizontal velocity of the ball is v_x in a particular level then the velocity of the paddle will be in range $v_x + \delta$ to $v_x - \delta$. It is clear from the velocity that the error in reaching the ball will completely depend on the delta value we choose.
The delta value for level 1 will be equal to v_x and for level 2 will be $\frac{v_x}{2}$ and for level 3 will be $\frac{v_x}{2^2}$ and so on.
The number of computer players in a game can be either 1 or 0.
Case II: 3 or 4 player Game

In the game where more than 2 player is playing there is no constraint on the vertical velocity of the ball. In these ball's velocity can change in both the direction. Here also the board will be frictionless and the algorithm for computer player will be almost same for this one also. The important thing is that if we have more than one computer player than each computer player will play their game independent of other computer player.

6 Analysis of Computer Algorithm

:
As we mentioned above ball will be always in motion once it starts all the computer players will also start following the ball. By the term the following I mean to say that every computer player will move in the direction of ball in their respective horizontal direction.

Every time a ball touches some one's wall, the board will re-order itself to the initial position which is as shown below. Every paddle will be at the centre position, and the ball will start from a random location with a predefined velocity and moving towards a wall which centre point will which is the farthest from the ball(not necessarily towards the centre of the wall).
After setting the initial condition, now comes the question "What will be the velocity of paddles?" We are solving this problem by taking into consideration of balls direction and balls velocity (more specifically horizontal velocity of the ball with respect to a computer's paddle). Every time the horizontal direction of the ball changes with respect to a computer player her paddle's direction also changes in the same direction. As mentioned above let us consider delta to be 0 then paddle will always exactly follow the ball and the computer player will never lose. Since this is not the case and delta is a positive value, there is a probability that the computer will lose the game. The probability of defeating the computer the player will depend upon the delta value we will choose. Let us say that the delta value in v_x then the paddle will move twice fast as the ball which implies that the paddle will go much further in x distance than y. In this case we can easily see that the computer player can be easily defeated. In our game we will choose intermediate delta value for different level with an invariant that the delta value of a harder level will be always smaller than the delta value of an easier level.
Where does the computer player run?
The answer to the question depends upon the type of game the user/users are playing.

1. **Solo Play:** In case of Solo Player, since there will be only one user, the computer player will run on his individual machine. Since solo play can be played even without network we must host the game in the user's system.

2. **Multiplayer:** In case of multiplayer game, it is necessary for someone to start and host the game. Let call this user as userZero, to play this game with other player userZero has to start his/her own personal network and connect others. So we will run the computer player in the userZero's system.

3. **Quick Play**: Since in this game the users can vary rapidly and also their connections may vary we will host the computer player in the server.

Events handled by the Computer Player:

1. Once the game starts computer player will ensure that the number of non-computer player is more than 0. If the number of non-computer player is 0 then the computer player will terminate the game and show a pop-up message.

2. If the computer player wins the match then after winning its very task is to terminate the game and show the popup message.

Network Communication

1. **Information Exchange:** The following information will be exchanged, in an array list, between different machines of connected players:

(a) Position and velocity of the ball: We'll exchange the x and y coordinates as well as the speed and direction of the ball in an array list.

(b) Position and speed of paddles: The x and y coordinates of the centre of the paddle and speed of the paddle.

(c) Position of visible power-ups: The x and y coordinates of the power-ups which are currently visible.

1. **Keeping the same state for all players:** For ball, we'll send the position and velocity of the ball to other players only when the ball strikes the paddle of that player and update the state at all the machines.

For paddles, whenever a player moves the paddle, the direction of movement will be sent to the other players and paddle of this player start moving at other players' machines and when this player stops moving the paddle, then we'll send a value to other players which will make the paddle of this player to stop.
For power-ups, power-ups will be generated at fixed points of time, different for each player, at some player's machine and the position of the power-ups will be sent to other players and then it will become visible at all players' machines.

1. **Event Flow for network message:** The event flow in this game will depend upon the type of game the user wants to play and we will describe all of them separately

   (a) **Solo Play:** Since the user's system is the only node in the system and As the solo play mode doesn't require any internet connection and all the responses in the game will be handled by the end user only. In these case the event flow will just be like function calling form one java file to other java file.

   (b) **Multiplayer:** The multiplayer platform will have a proper working network, during the starting the major task will be done by the host say zeroUser. The zeroUser will create a network and other will request a connection. Once connection is established then they can start the game by following the appropriate rules.

   (c) **Quick Play:** The event flow in this case is quite similar to the multiplayer case but the only difference is that the host is developers own server and the only task of the server is create a connection between them.

When my code concludes that a user has been disconnected from the network, various action is taken depending upon the situation. The causes for disconnection may vary, some may be intentional quitting of game, or some time it may be because of internet connection. In the first case there is very low chances of returning to the game but in the second the user may return within few seconds (say 2 or 3 seconds). So when the game encounters that a player is unable to send or receive request and responses then, at first we wait for 5 seconds before completely removing the user from the game.

Let us assume that the user hasn't connected back in 5 seconds then we replace that's users paddle with static wall and push the name of the user to appropriate position.

If the user connects back to the network within 5 seconds, in that case we need continue the game smoothly including the current user also. To establish that condition what we will do is that whenever we face a network issue