Design Practices COP 290

Assignment 3
Multiplayer Ping Pong Game

Abhishek Kedia (2013EE10431)
Arnav Kansal (2013EE10440)
Ashish Ranjan (2013TT10908)

April 7, 2016

1. Scope of the Game

1.1 More than one ball ?

No, the first version of the game will not allow the players to play with more than one ball. However this does not mean that the scope of the game is ball limited.

1.2 Special powers

The game will not allow special powers on the bat per se, but to make the game less monotonous a couple of special features will be added.

- All surfaces including paddles will have friction and thus the ball will not travel in a repetitive path.
- The collision of the ball with the walls will increase the speed of the ball, increasing the difficulty of the game as the game proceeds.

2. Physics of the Game

We will keep the physics of the game relatively simple. The basic equation that we will use is reflection, i.e, on collision with a surface/ paddle the component of velocity pendicular to the surface will get reversed. For the component of velocity parallel to the surface, we can keep it same which will simulate perfectly elastic collision. But in that case we know,

$$\theta_i = \theta_r \tag{2.1}$$

But that would mean that ball would keep moving along the same direction throughout

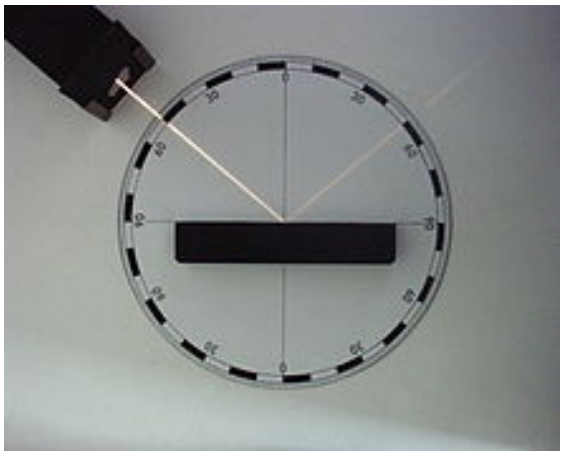

Figure 1: Law of reflection- Angle of Incidence equals Angle of reflection

the game. We don't want that as that would make things very boring. So what we can do

is introduce something like the coefficient of restitution e, due to which the component of velocity parallel to the surface reduces a bit according to

$$\frac{v_{||,r}}{v_{||,i}} = e \tag{2.2}$$

But introducing that would mean that speed of ball decreases. Since there is no source of acceleration for the ball that would mean that ball's speed reduces over time and eventually almost stops. We don't want that to happen. So we will increase the perpendicular component of speed some time during collision will with the paddle / wall.

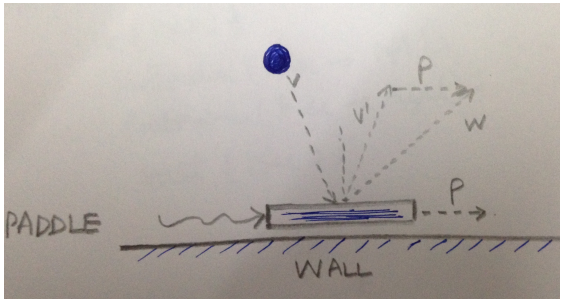

Figure 2: Analyzing the collision of the ball with the paddle

3. Algorithm for the Computer Player

1. As the game is deterministic, in the sense that the AI will know beforehand where the ball is headed, designing an algorithm for the computer player is going to be simple.
   For starters, the computer player will track the x-position of the ball.
   Across varied levels of difficulty, the speed with which the computer player's paddle will lock on to the projection of the ball will increase.

2. The computer player will not run on a central server. Computer players will run on all the machines with close synchronisation in the states of computer player computed separately on all machines.
   This will however pose a problem of synchronisation and will be the most difficult thing to handle in the assignment. For more refer 4.2 **Game synchronization**.

3. The computer player only has to follow the x-position of the ball, and it has to do so in an infinite loop barring a few scenarios, where an interrupt will be issued.

   (a) In case of game pause, all threads of the game will be paused, and saved, and synchronised across all players.

(b) In resume, the game state will be revived from the local saved copy, once the integrity of the saved state is checked, and the computer player will start once again.

(c) In the event of exhaustion of all 3 lives of the computer player, the computer players paddle will be removed from the screen and the wall behind it will turn into a simple wall.

(d) In the event of all other human players getting defeated, the game will pause and an a list of options will be provided to the user. These will include the option to start again or view the remaining game.

4. Network communication

4.1 Information exchanged between different machines

Each machine would send to all of its peer the following messages :

1. A periodic "Keep alive" message : This is make sure that all players are active and connected. On not receiving this message for a significant time from other peers would indicate someone has disconnected. If we receive some keep alive message but not all that would indicate we are alive in the network and someone else has gone down. Appropriate action would be taken in either of these two cases.

2. Periodic Location and Velocity of ball: Each machine locally computes the ball's position in the next frame based on the previous frame and equation of motion governing the ball. But as it may happen each of the peers may be slightly out of sync with each other. Over time this can accumulate and create problems. So periodically send out current of ball, and thus resolve any potential mismatches.

3. Paddle movement: Whenever a paddle is moved by the player in one machine, this information is communicated to each other player.

4. Periodic sync information of game state: Periodically each player would try to sync the all the aspects of game like score with have less frequent changes with other players.

4.2 Synchronization of game state

By synchronization we mean that all players see the game state at all points of time. Synchronization of game state is a standard and fundamental problem in game design. Exact synchronization can never be achieved fully. But we can achieve it to varying degrees of approximation.
Some of the strategies that we plan to use are :

1. Simplest Approach : In this approach no move is executed until all players are informed about it a priori. This means whenever a player moves a paddle, it not

actually moved in that player local system. First this information is communicated to all players and when they receive this message and acknowledge it, only then this move is executed simultaneously on all players. This method is very simple to implement, but does not lead to a good playing experience. Player would immediately feel that game is terribly some and will have difficulty in controlling the paddle because of lack of immediate visual feedback.

2. Interpolation : In this method when a player performs move then that is immediately affected in the local system, and a message is sent to all other peers. On receiving this information other peers, know what the player moved. But they also know that there is some delay in the information so they realise that this was the state of the player at sometime before (which is almost known since network delay is almost constant). Using this information and time lag, new location of the player can be interpolated and thus be placed there. This method is better than the previous strategy in terms of game play experience but is harder to implement in code.

3. Predicting motion of other players : The last method is good but there is an initial jerk since other player was not not moving but then is suddenly at a different location and is moving with a finite non-zero speed. So when we are not receiving movement from other player, this machine tries to estimate where the player should move and slows starts moving it in that direction. This avoids that initial jerky motion. But in certain situations this can do more harm than good.

So we will start off with the simplest methods and progressively add more complexity.

4.3 Event Flow for network messages

All messages sent across will conform to a standard structure specified tentatively as follows:

1. TAG : All network messages will carry a unique tag.

2. All messages will be categorised and sorted according as their unique tag, and all corresponding actions will take place accordingly.

    - pause
    - resume
    - isAlive
    - ballInfo
    - paddleInfo
    - gameState
    - stop
    - start

.

4.4 Trigger event to decide to detect disconnected peers

As mentioned before each peer periodically (say 10ms) sends a keep alive message to other peers. If some peers does not receive this keep alive message for a long a time say 5-10 periods then we can conclude that peer is disconnected.

4.5 On player disconnect

Once we realise that a player has disconnected then we pause the gameplay and inform the user that someone has disconnected. The remaining players will have the option to either continue the game with each other or entirely shut down the game.