Pong!

CCC Wien 2014 warm-up

Game Description

Pong is a two-dimensional sports game that simulates table tennis. The player controls an in-game paddle by moving it vertically across the left side of the screen, and competes against a computer-controlled opponent controlling a second paddle on the opposing side. Players use the paddles to hit a ball back and forth. The goal is for each player to reach 30 points before the opponent; points are earned when one fails to return the ball to the other.

Your task is the write a good computer-controlled opponent that connects to our game simulator and can beat our AI.

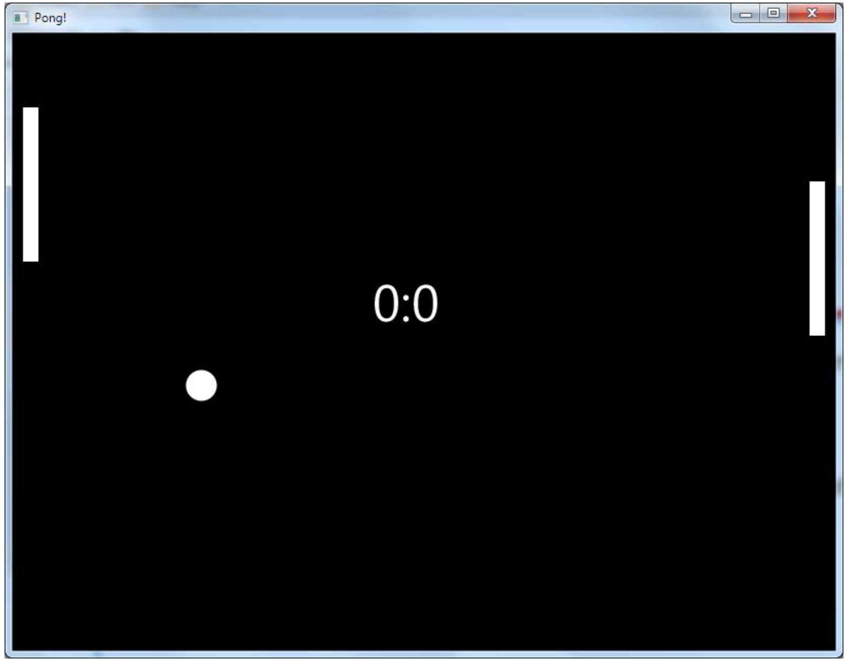
Game Metrics

court size: 800 x 600 (width x height)
paddle size: 15 x 150
ball radius: 15

The point (0,0) is on the top-left.

The paddle position is a value between 0 and 450 and equals to the distance between the top border and the paddle.

The x coordinate of the left paddle is 10 and of the right paddle 775. (distance between the left border and the left side of the paddle)



Simulator

The simulation runs with 60 steps per second. Every 6th step your program will receive an update of the world and is expected to send movement instructions for your paddle.

The simulator is written in Java and to run it you will need Java 7 (Update 6) or higher installed. On Linux please use the OracleJDK. (Unfortunately most OpenJDK packages do not support JavaFX)

There are two modes of communication with the simulator. You only have to implement one method though.

Simulator - Console Mode

The simulator executes your program and redirects stdin and stdout.
If you choose this method your program only has to interact with the "console". The downside of this approach is that you can't easily debug your application. Of course you always can attach your debugger after the program has started. You can also write debug output to stderr, which will be shown in an debug window of the simulator.

Example start commands:

```
 java -jar simulator.jar --level=1 --exec=mysolution.exe
 java -jar simulator.jar --level=1 --exec="java -cp myclasspath my.company.Main"
 java -jar simulator.jar --level=1 --exec="c:\python\python.exe solver.py"
```

Simulator - TCP Mode

The simulator opens a tcp port and your program connects to it.

Example start commands:

```
java -jar simulator.jar --level=1 --tcp=7000
```

Simulator - Usage

```
java -jar simulator.jar --level=<levelnr> [--exec=<solver>] [--tcp=7000] [--speed=2]
```

Arguments

--level=<levelnr>	selects the level (default: 1)
--speed=<factor>	set the speed factor (1-10) of the simulation
--exec=<solver>	start in Console Mode
--tcp=<port>	start in TCP Mode

Communication Protocol

Communication with the simulator is line based. The simulator starts and sends the current state. After that it will send "update" and wait for the client to reply with “move”.

simulator messages:

player <pos> <move>	the position of your paddle (a value between 0 and 450) and the last move command
cpu <pos> <move>	the position of the opponent's paddle and the last move command
ball <x> <y> <vx> <vy>	the center position and the velocity of the ball (float number with 2 decimals)
update	the simulator asks for a new move command

client messages:

move <number>	a number between -36 and 36 inclusive (every tick number/6 will be added to your position)

Communication Example

```
player 0 0
cpu 0 0
ball 400.00 300.00 -7.22 1.42
update
move 36
player 0 36
cpu 0 36
ball 356.70 308.49 -7.22 1.42
update
move 36
player 36 36
cpu 36 36
ball 313.39 316.98 -7.22 1.42
update
move 0
player 72 0
cpu 72 36
ball 270.09 325.48 -7.22 1.42
update
…
```

simulator messages are red
client messages are green

The decimal separator is "."

The effect of the move command is not seen immediately but on the next state. You can predict were your paddle is when your move command will be executed by summing up the position and your last move.

Level 1

Your task is to connect to the simulator with one of the two methods and beat our simple AI. When you win the game the simulator will write a “result.txt” file. Upload this file to CatCoder in order to complete the level.

Goals Summary:

- Beat the simple AI by scoring 30 points