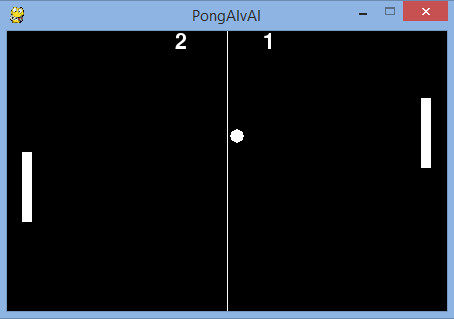
|  |
| --- |
| **“A Tournament for Pong AI Engines” by Michael Guerzhoy is licensed under CC BY-NC. Accessed from www.engage-csedu.org.** |

**The CSC180 AI v. AI Pong Tournament**

We will have a [Pong](http://en.wikipedia.org/wiki/Pong) tournament for AI engines as part of CSC180 this year. The entry submission deadline will be Nov. 25. Teams of up to two people will be allowed.

**Prizes**: 4 points for first place, 3 points for second place, 2 points for third place, 0.5 points for beating The Chaser by at least 150 points when playing to 1000 points.

The game physics are included in [PongAIvAI.py](http://nifty.stanford.edu/2018/guerzhoy-pong-ai-tournament/handout/PongAIvAI.py). A sample AI engine, which you have to beat to qualify for the class tournament, is at [chaser\_ai.py](http://nifty.stanford.edu/2018/guerzhoy-pong-ai-tournament/handout/chaser_ai.py).

**Installing PyGame**

To run PongAIvAI, you need to install [PyGame](http://www.pygame.org/news.html). (For Windows, I recommend downloading the latest version of PyGame from [here](https://bitbucket.org/pygame/pygame/downloads). See the [PyGame download page](http://www.pygame.org/download.shtml) for other platforms.)

**Running PongAIvAI**

You can run PongAIvAI.py in Python if you download chaser\_ai.py into the same folder as PongAIvAI.py, after you install PyGame. See lines 389-390 in PongAIvAI.py for how to set the AI/human input functions for PongAIvAI:

**import** chaser\_ai

# To have The Chaser play against your AI engine,

# store your code in student\_ai.py, import student\_ai,

# and set paddles[1].move\_getter to student\_ai.pong\_ai

paddles[0].move\_getter = chaser\_ai.pong\_ai

paddles[1].move\_getter = directions\_from\_input # chaser\_ai.pong\_ai

Uncomment line 87 and comment out line 83 in PongAIvAI.py once your function doesn't produce errors (line 82 suppresses all errors and simply doesn't move the paddle if your function produces an error). Line 82 only accepts input for 0.3ms.

# The program crashes if move\_getter crashes. The runtime of

# move\_getter is not limited

direction = self.move\_getter(self.frect.copy(), enemy\_frect.copy(), ball\_frect.copy(), tuple(table\_size))

# The program continues if move\_getter crashes. The runtime of

# move\_getter is limited

# direction = timeout(self.move\_getter, (self.frect.copy(), enemy\_frect.copy(), ball\_frect.copy(), tuple(table\_size)), {}, self.timeout)

**Tournament Format**

A round-robin tournament will be held among all the teams that qualify for the tournament. In each match, each opponent will play both on the left and on the right. Game parameters will not change by default (except for the score needed to win), but may change at our discretion for tiebreaking purposes. There is a 0.3 ms limit on how long it takes for your function to return a move. For timing purposes, the tournament will be run on ECF or a faster machine. The parameters will stay the same as in PongAIvAI by default, but may be altered for tie-break purposes.

**Submission instructions**

Please submit a file named **pong\_ai.py**, which must contain the function **pong\_ai**, which will be called the same way chaser\_ai.pong\_ai() is called.

Good Luck!