

2022

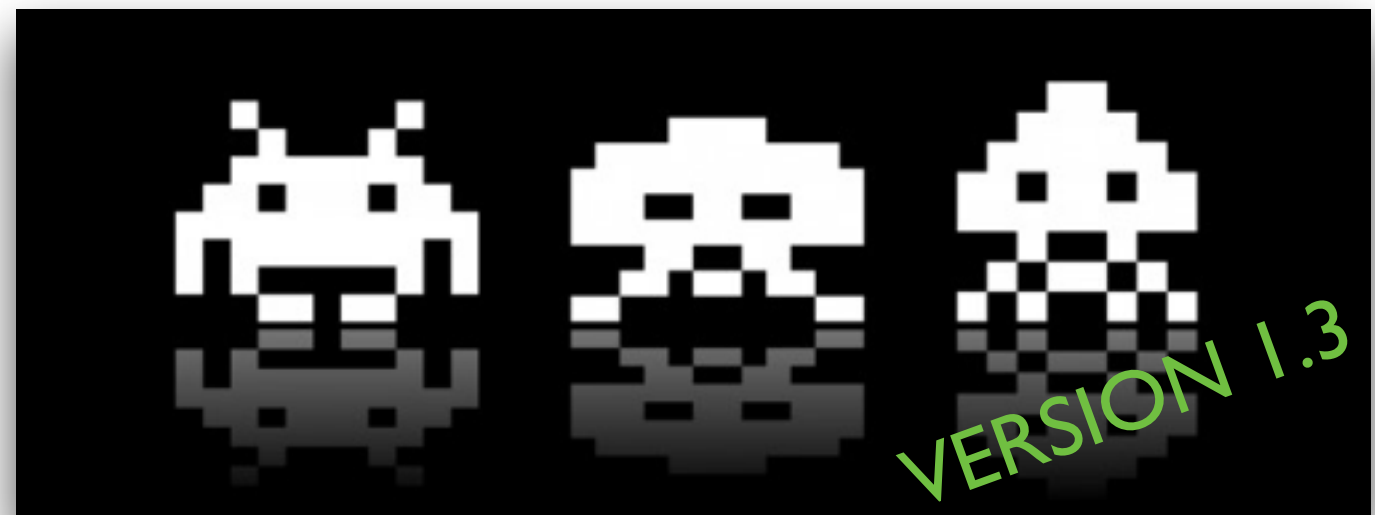
# INTRODUCTION TO NUMERICAL ANALYSIS

## Tournament

Kai-Feng Chen  
National Taiwan University

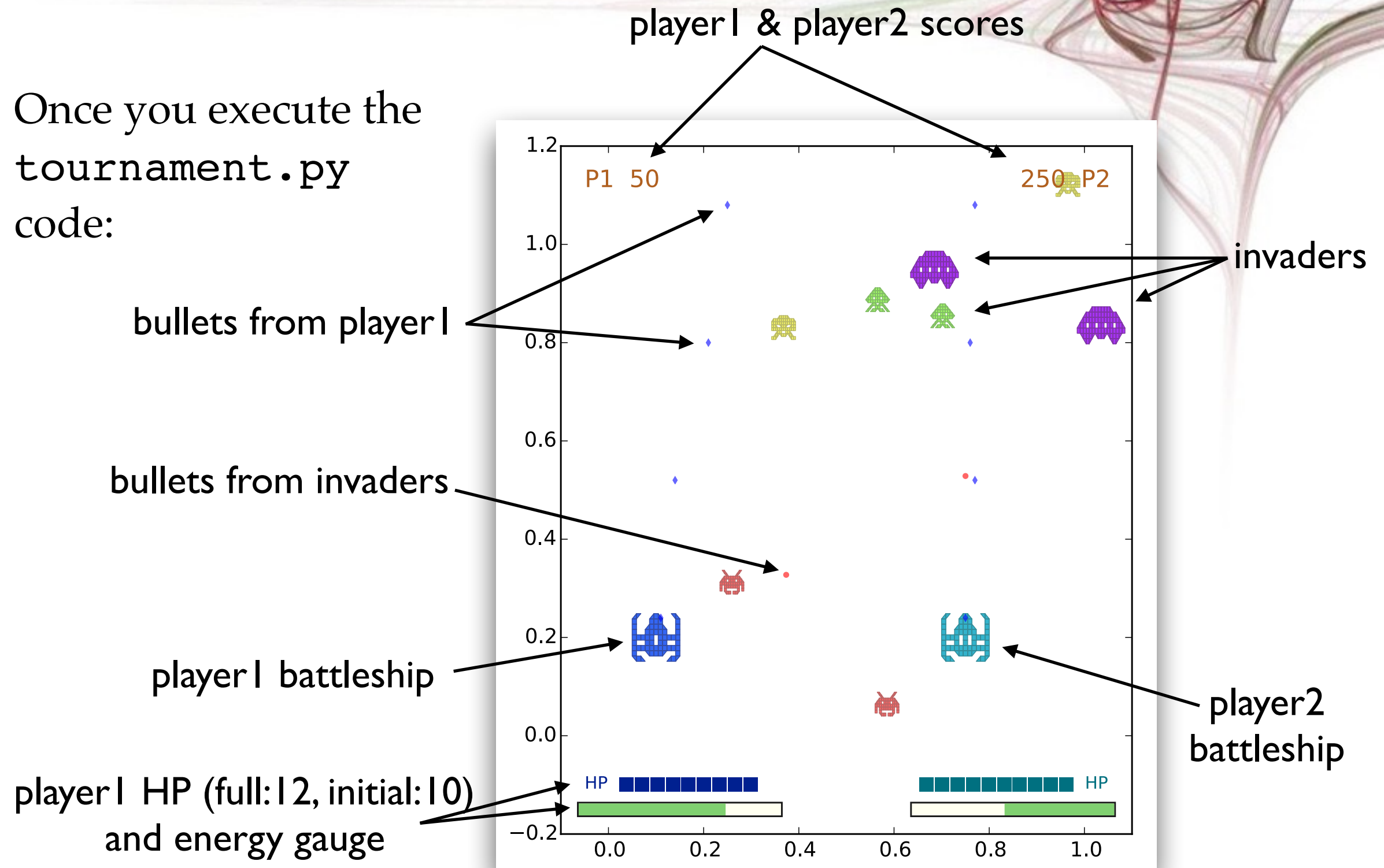
# ALL YOU NEED TO KNOW

- We have the video game tournament!
- And we are going to play a not-so-classical **space invaders**.
- All you need to do is derive a good AI program to control your space battleship, hide from the attacks, and shoot those invaders down!
- We are going to run this tournament with pair of groups and who gets **higher scores** who win!



# THE SETUP

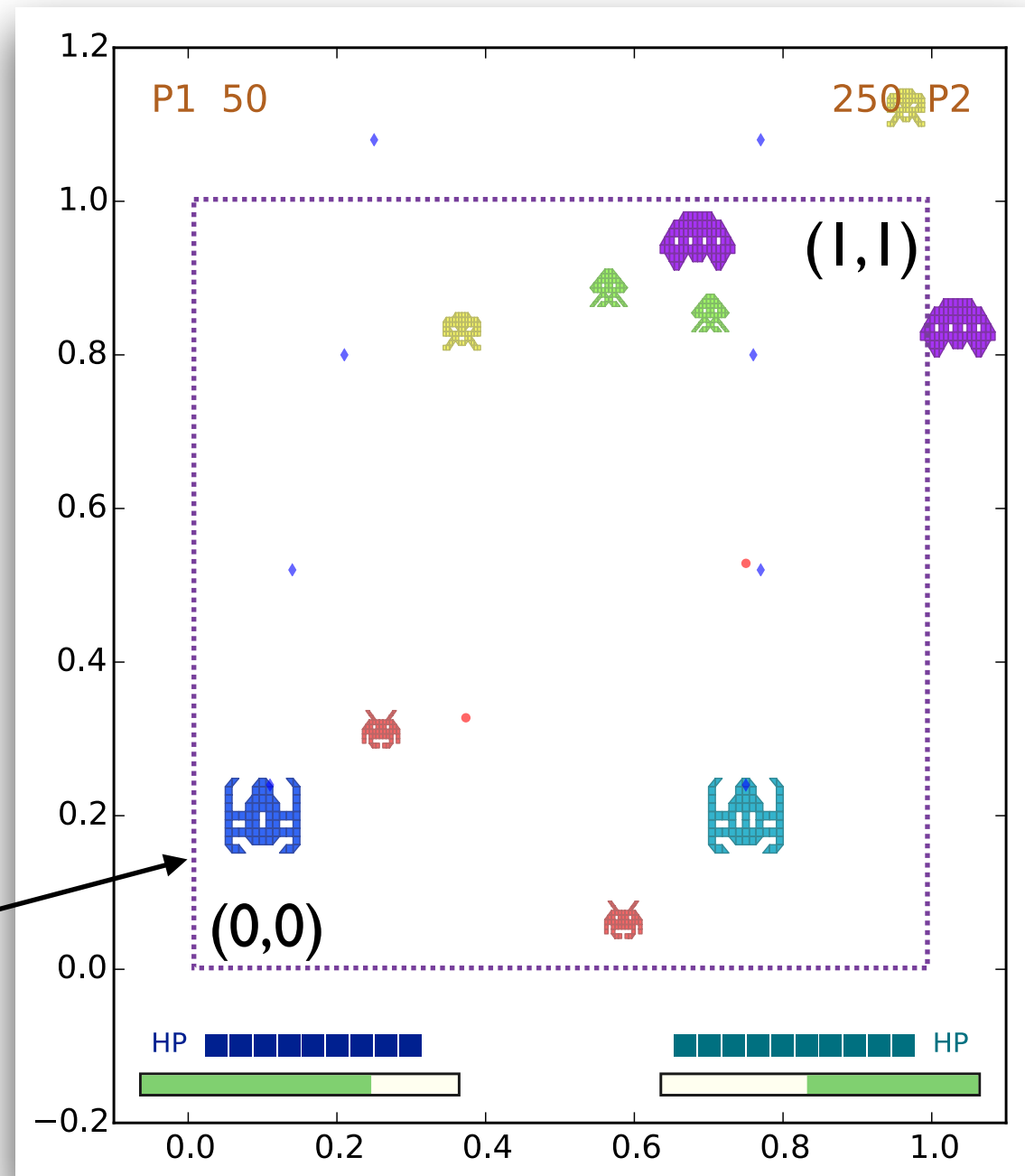
- Once you execute the `tournament.py` code:



# COORDINATION SYSTEM

The invaders are basically falling from top

player's battleship are limited to move within this square



The maximum moving speed of your battleship is **0.01 unit per unit time frame** (as well as the invaders)

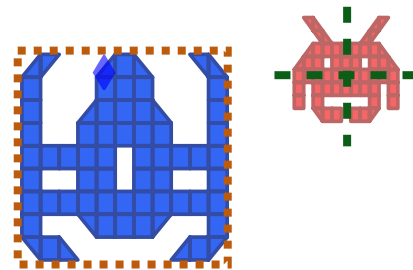
Your code needs to provide the speed scale (0.0–1.0) and the direction (0.0– $2\pi$ ) for every time frame



# COORDINATION SYSTEM



The size of your battleship and the UFO are 0.1x0.1;  
The size of invaders are 0.05x0.05








Collisions of **invaders** with your battleship:  
whenever the center of the invader enter your square region.



Collisions of **bullets** with your battleship or invaders:  
whenever it enters the square region.

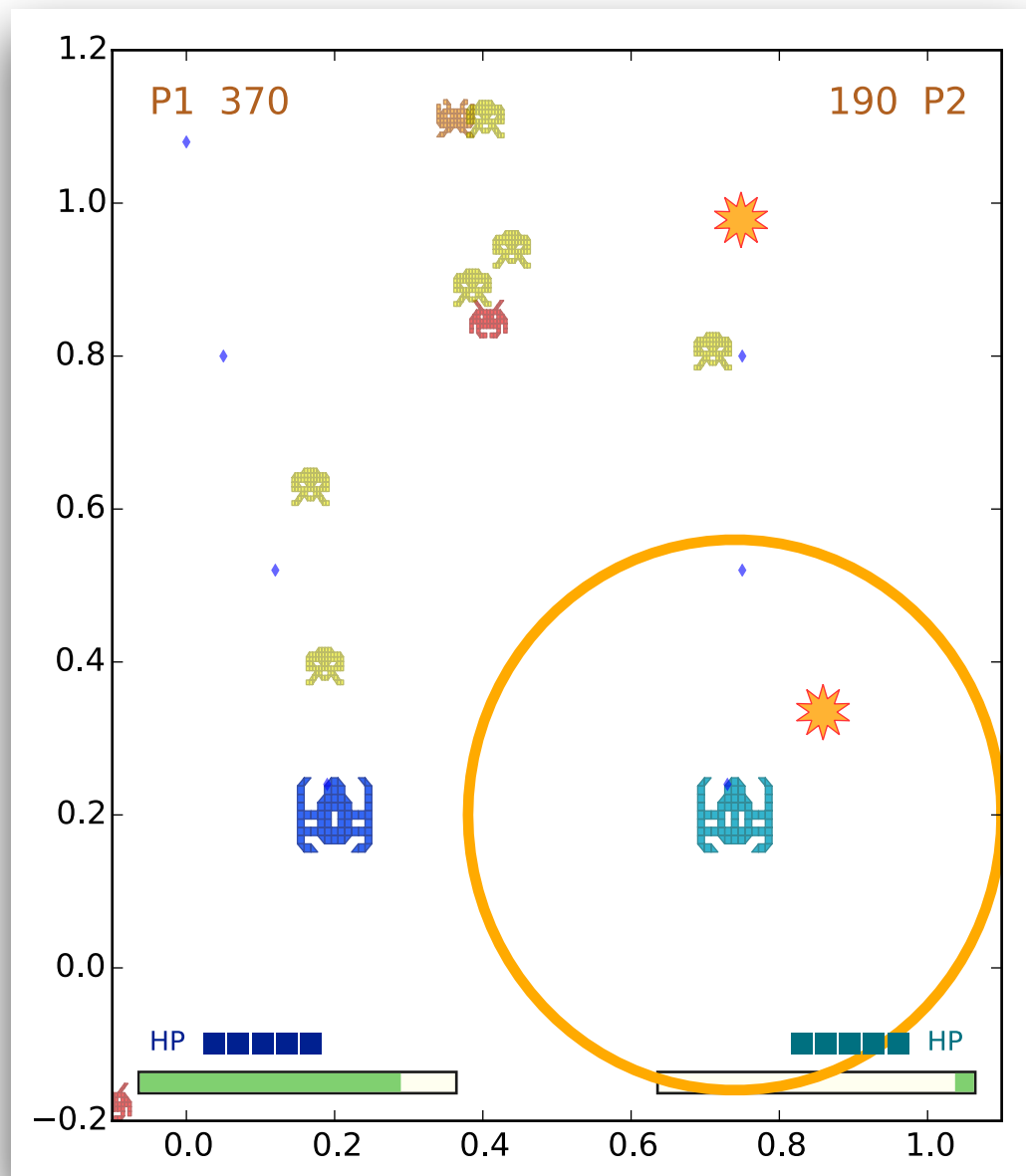
# TYPES OF THREATS

- type 0: bullet shoot from invaders / speed =  $2x$  / 5 points\*  
(speed of your bullet is  $4x$ !)
-  type 1: moving along a straight line / HP 1 / speed =  $0.75x$  / 10 points
-  type 2: moving along a curve / HP 1 / speed =  $0.75x$  / 10 points
-  type 3: heading toward you / HP 1 / speed =  $0.3\sim 0.78x$  / not shooting / 15 points
-  type 4: random walk / HP 1 / speed =  $1x$  / not shooting / 15 points
-  type 5: moving horizontally / HP 5 / speed =  $0.2x$  /  
doubled shooting rate / 20 points per hit

\* you can only “clear” those bullets with the EMP bomb!

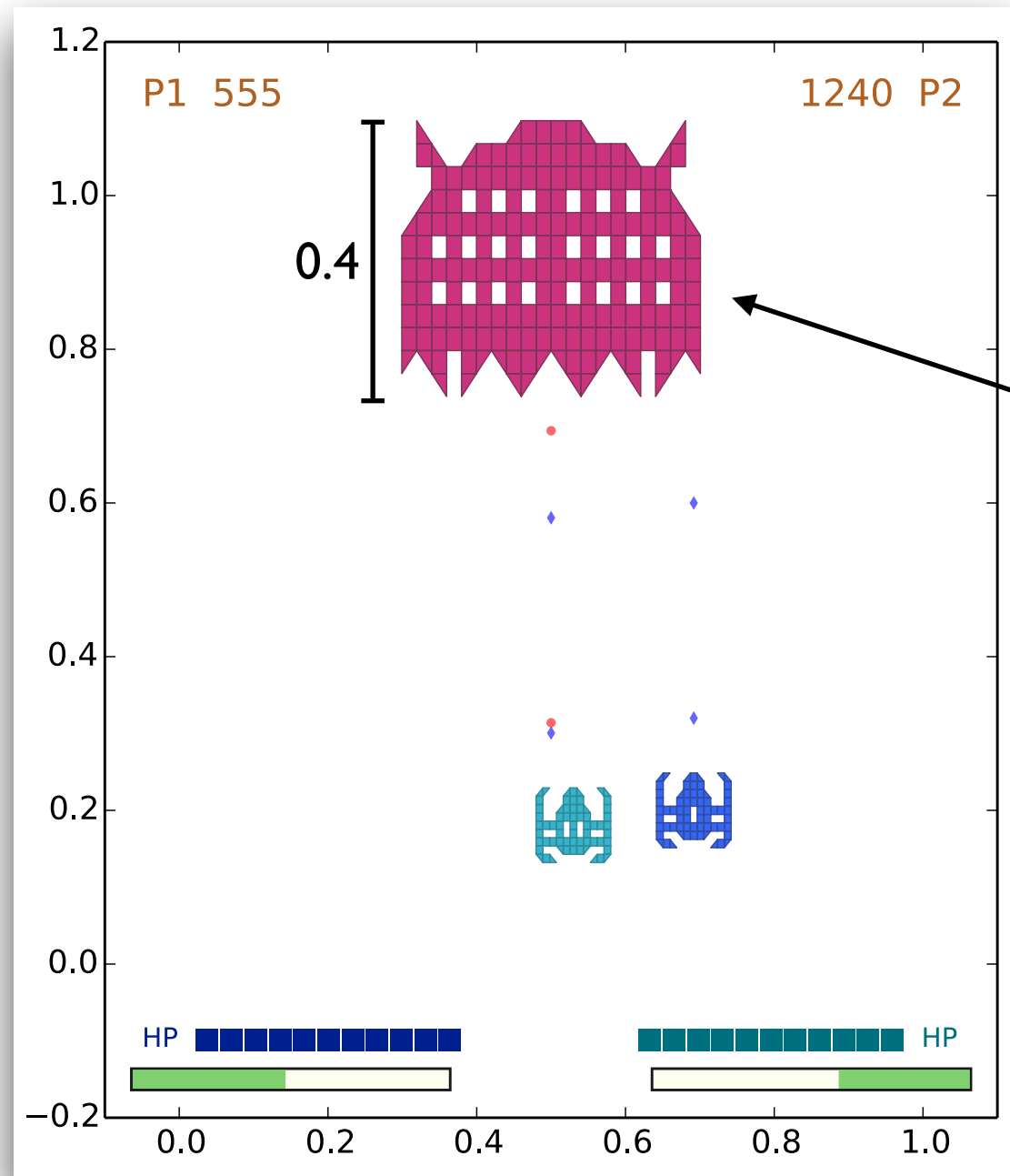
speed  $1x = 0.01$  unit per frame =  
your maximum speed

# RANGE ATTACK



- If your battleship moves slower (speed scale < 1), the “energy gauge” can be accumulated. The amount of energy accumulation is  $4 \cdot (1 - \text{speed scale})$ .
- When it is full (value = 1000), it will shoot an EMP bomb of maximum radius of 0.54.
- The EMP bomb will destroy any enemies within the ring (including the bullets), except the boss!
- However it also has some negative effects to your “friend”!

# NEW SINCE VERSION 1.2

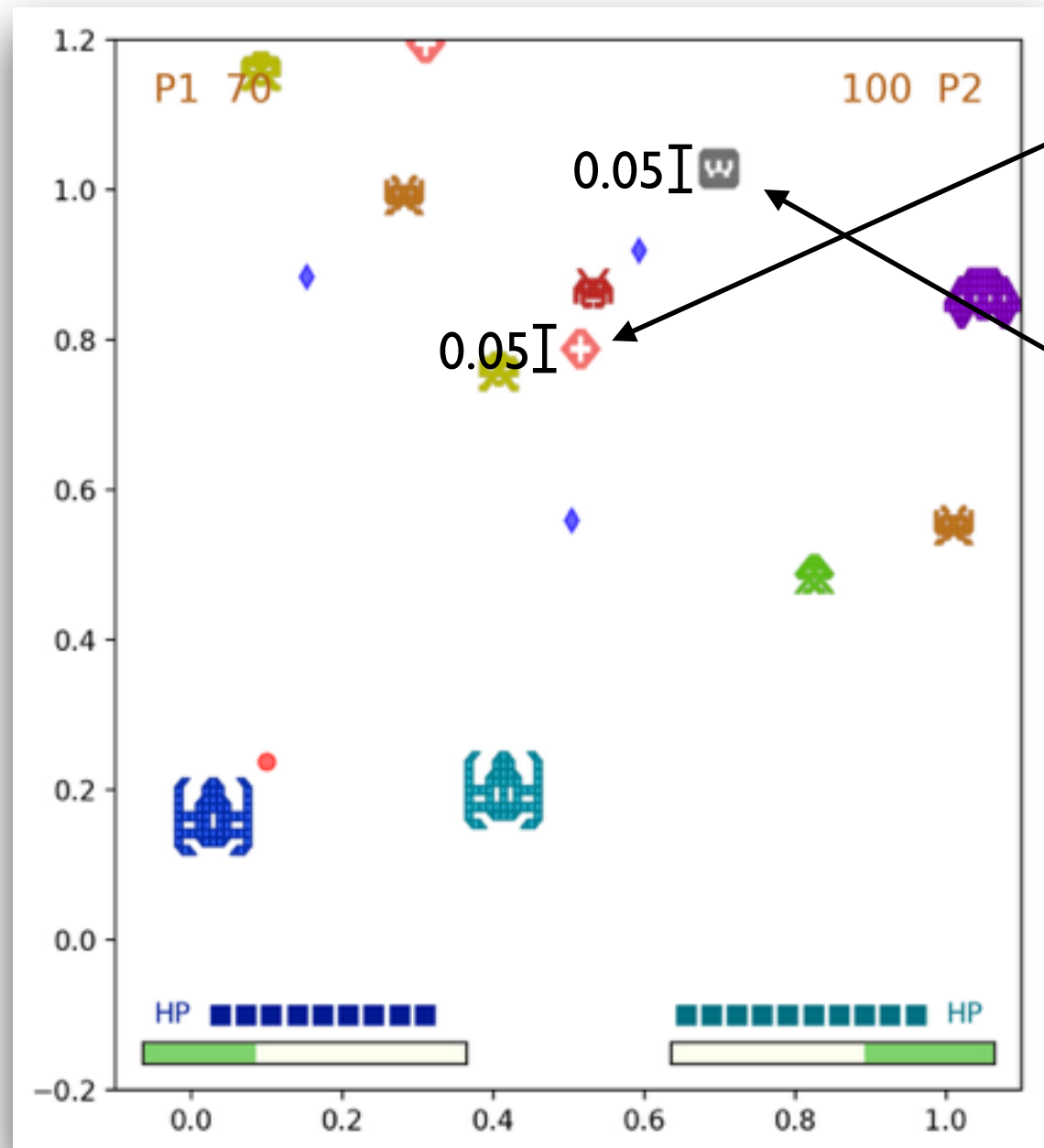


type 6: “Super UFO”  
moving vertically / HP 120+ /  
speed = 0.1x / 4x shooting rate /  
10 points per hit

BOSS FIGHT WITH GIANT “SUPER  
UFO” BEFORE LEVEL UP!!



# NEW SINCE VERSION 1.3



type 7: “Rescue capsule”  
moving vertically / speed = 0.2x /  
restore 1 HP (maximum HP=12)

type 8: “Weapon upgrade”  
moving vertically / speed = 0.2x /  
upgrade your weapon by 1 level  
(maximum level=3)

If you get a hit, your weapon level will  
be downgraded as well!

# PLAYER AI TEMPLATE

The **decision function** will be called by the main program **every frame** and you need to return the speed scale and direction of your movement.

```
import numpy as np

class player_module:

    # constructor, allocate any private data here
    def __init__(self):
        self.init_x, self.init_y = -1., -1.    ⇐ Constructor

    # Please update the banner according to your information
    def banner(self):
        print('-'*40)
        print('Author: your_name_here')    ⇐ Put your name and ID here
        print('ID: bxxxxxxxxx')
        print('-'*40)

    # Decision making function for moving your ship, toward next frame:
    # simply return the speed and the angle
    def decision(self, player_data, enemy_data):    ⇐ the main decision function
```

player\_template.py

```
def decision(self, player_data, enemy_data):
```

```
    speed, angle = 0., 0. ← the information to be replied:  
                           speed scale (0 to 1), and the direction (0 to 2π)
```

```
    # your data
```

```
    player1_x      = player_data[0][0]  
    player1_y      = player_data[0][1]  
    player1_hp     = player_data[0][2]  
    player1_score  = player_data[0][3]  
    player1_gauge  = player_data[0][4]  
    player1_weapon = player_data[0][5]
```

*Note the demo code only moves left & right,  
but you can actually move toward any direction.*

← your current data (coordination, HP, etc)

*Note you are always the “player1” here.*

```
    # data for another player
```

```
    player2_x      = player_data[1][0]  
    player2_y      = player_data[1][1]  
    player2_hp     = player_data[1][2]  
    player2_score  = player_data[1][3]  
    player2_gauge  = player_data[1][4]  
    player2_weapon = player_data[1][5]
```

← the data for another player

```
    .  
    .  
    .  
    .  
    .  
    # loop over the enemies and bullets
```

```
    for data in enemy_data:
```

```
        type = data[0]
```

```
        x    = data[1]
```

```
        y    = data[2]
```

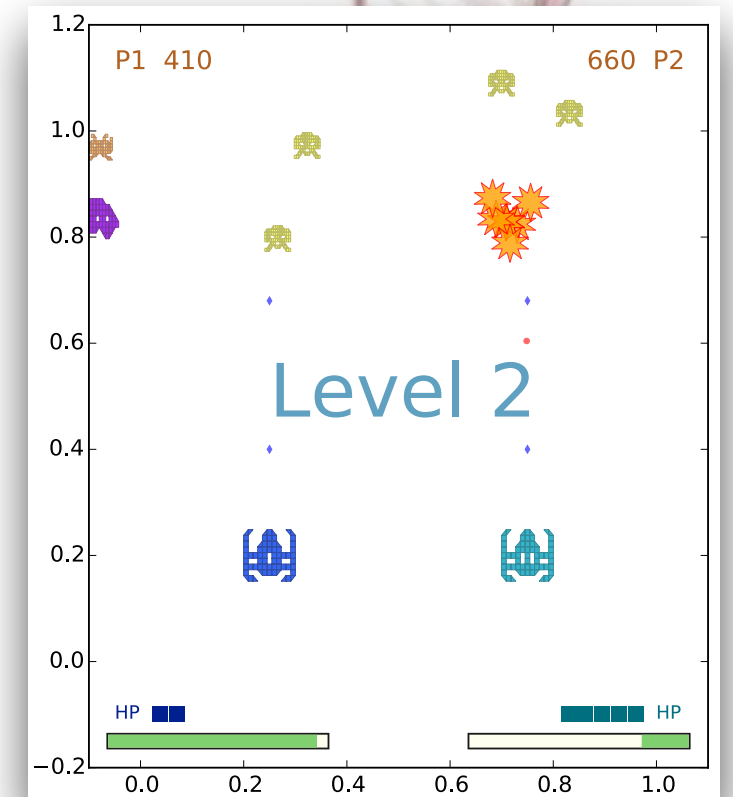
```
        dx   = data[3]
```

```
        dy   = data[4]
```

← invader's information (including bullets)

# HAVE FUN!

- Who gets **more scores** in the end win!
- We will have two rounds of tournament:
  - **Round match:** will reduce the # of players by half, the rest players enter the final match.
  - **Final match:** elimination game (targeting your championship!).
  - Anyone who enters the final match ( $\leq \sim 50\%$  of participants) will get a level upgrade to your final score of this course!
- Please provide the first version of your code on **May/28** for the first round; the final match will be held on **June/11** (with your final code).



*Game level up with higher scores, but with more enemies...*