Patricia Capitan, Aida Gomezbueno, Marie Plouy
CS 457- Intro to AI

Clobber Bot

Hours spent on the project:
Patricia - 10hrs
Aida - 11hrs
Marie - 10hrs

Algorithm

Overview

In this project, we have developed an algorithm that implements the game strategy of a bot willing to be launched in the Clobber Bot game. The game's goal is to escape the bullets emitted by the other adversary bots in the environment and, in turn, attack them in the same way. The goal is to maximize the number of victorious attacks (dead bots) accumulating 1 point for each one and, most importantly, surviving, which adds to the total amount of 10 points.

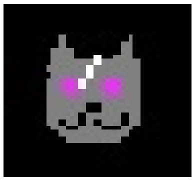
Our CatBot (MariePlouy.java) inherits from the ClobberBot class, overwriting the methods toString, drawMe, and takeTurn, which are responsible for giving a name to our bot in the final report and analysis of the game, the LookFeel, and the way it acts according to the current state of the environment or its own perception, respectively. The following is a description of each of the methods and functions necessary to accomplish its correct implementation in terms of functionality and performance optimization.

drawMe(Graphics page, Point2D me);

For our bot, we decided to do a cat that changes colors when shooting. The cat is drawn using the following Graphics methods:

- **fillOval()** - for the head, eyes, and nose
- **fillPolygon()** with 3 sides (triangle) - for the ears
- **drawArc()** - for the mouth
- **drawLine()** - for the stripe

The main cat is gray with magenta eyes and a white stripe. The shooting cat turns orange with red ears, white eyes, and a white stripe. Below are pictures of each cat version.



Moving

Shooting

takeTurn(WhatIKnow currState);

Takes into account the shotClock variable. If negative or equal to zero, shoot, else move either to avoid a bullet or to get closer to the bot it's attacking.

When shotClock is negative or equal to zero, takeTurn calls getBotDirection() to get the shooting direction, then calls ClobberBotAction with given shooting direction, and then shotClock is incremented.

When shotClock is positive, takeTurns calls either getEscapeDirection() to get the accurate direction of the closest bot or bullet and avoid it, or getBotDirection() if not being attacked to get nearest bot direction and attack it.

toString();

Prints out bot name as well as our names.
"CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy"

Added functions:

getAction(WhatIKnow currState);

Checks for closest bot and bullet, and then returns a action depending on what is closer.

getBotDirection(WhatIKnow currState);

Checks for the closest bot, and return the directional move.

getEscapeDirection(WhatIKnow currState);

Checks if a bullet or bot is getting close, and return the directional move to move away from object.

getNearestBot(WhatIKnow currState);

Returns closest bot.

getNearestBullet(WhatIKnow currState);

Returns closest bullet.

botDist(BotPoint2D bot1, BotPoint2D bot2);

Calls getDist to calculates the distance between two bots.

bulletDist(BotPoint2D bot1, BotPoint2D bot2);

Calls getDist to calculate the distance between bullet and our bot.

getDist(double point1X, double point1Y, double point2X, double point2Y);

Calculates the distance between two points.

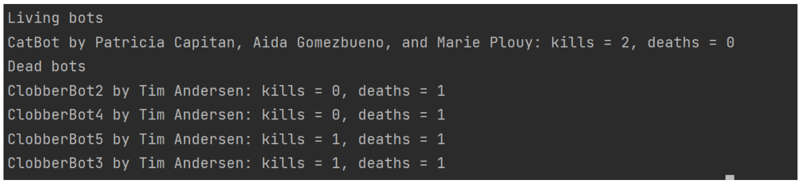
Results

1. **Rlist**
   - ClobberBot2 - ClobberBot5
2. **Tlist**
   - p13 - p16 bots
3. **Plist**
   - p1 - p12 bots

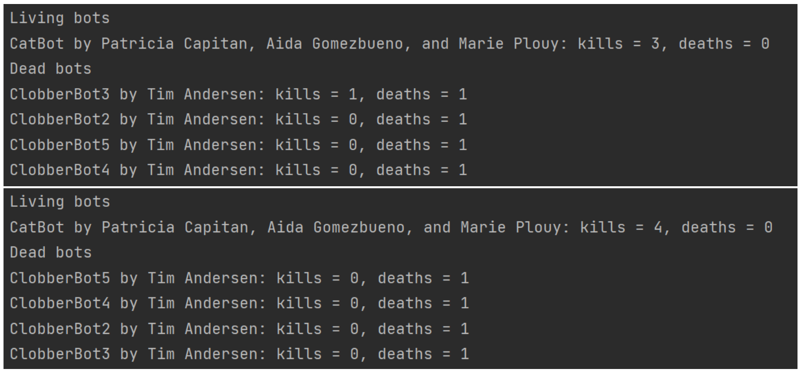
	Rlist		**Plist**		**Tlist**	
Position	#	**Average kills**	#	**Average kills**	#	**Average kills**
1st	9	2.77	1	1	0	0
2nd	0	0	3	2	2	0.5
3rd	1	2	2	2	1	1
Total Average	10	3	10	1.4	10	0.4

Out of 10 runs- number of times it got 1st, 2nd, or 3rd, as well as the average number of kills

Rlist



```
Living bots
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 2, deaths = 0
Dead bots
ClobberBot2 by Tim Andersen: kills = 0, deaths = 1
ClobberBot4 by Tim Andersen: kills = 0, deaths = 1
ClobberBot5 by Tim Andersen: kills = 1, deaths = 1
ClobberBot3 by Tim Andersen: kills = 1, deaths = 1
```

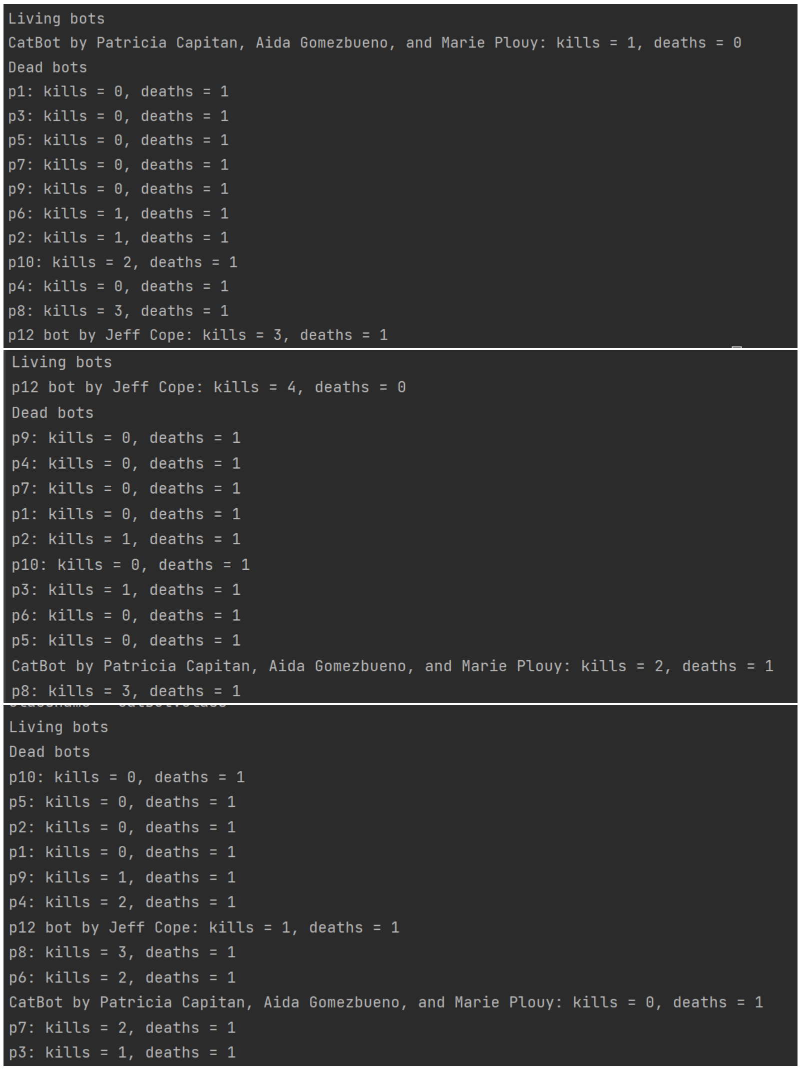


```
Living bots
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 3, deaths = 0
Dead bots
ClobberBot3 by Tim Andersen: kills = 1, deaths = 1
ClobberBot2 by Tim Andersen: kills = 0, deaths = 1
ClobberBot5 by Tim Andersen: kills = 0, deaths = 1
ClobberBot4 by Tim Andersen: kills = 0, deaths = 1
```

```
Living bots
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 4, deaths = 0
Dead bots
ClobberBot5 by Tim Andersen: kills = 0, deaths = 1
ClobberBot4 by Tim Andersen: kills = 0, deaths = 1
ClobberBot2 by Tim Andersen: kills = 0, deaths = 1
ClobberBot3 by Tim Andersen: kills = 0, deaths = 1
```

Sample Outputs for Rlist

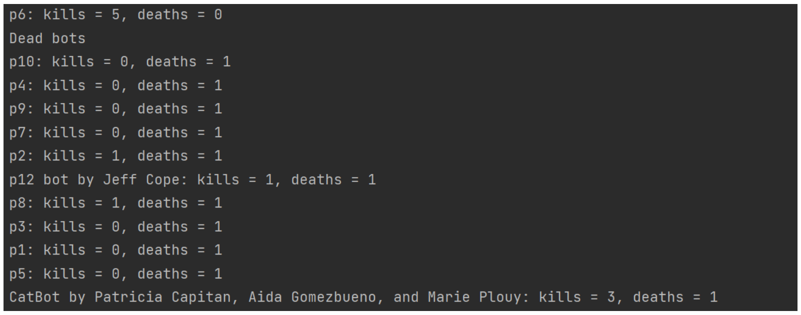
Plist



```
Living bots
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 1, deaths = 0
Dead bots
p1: kills = 0, deaths = 1
p3: kills = 0, deaths = 1
p5: kills = 0, deaths = 1
p7: kills = 0, deaths = 1
p9: kills = 0, deaths = 1
p6: kills = 1, deaths = 1
p2: kills = 1, deaths = 1
p10: kills = 2, deaths = 1
p4: kills = 0, deaths = 1
p8: kills = 3, deaths = 1
p12 bot by Jeff Cope: kills = 3, deaths = 1
```

```
Living bots
p12 bot by Jeff Cope: kills = 4, deaths = 0
Dead bots
p9: kills = 0, deaths = 1
p4: kills = 0, deaths = 1
p7: kills = 0, deaths = 1
p1: kills = 0, deaths = 1
p2: kills = 1, deaths = 1
p10: kills = 0, deaths = 1
p3: kills = 1, deaths = 1
p6: kills = 0, deaths = 1
p5: kills = 0, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 2, deaths = 1
p8: kills = 3, deaths = 1
```

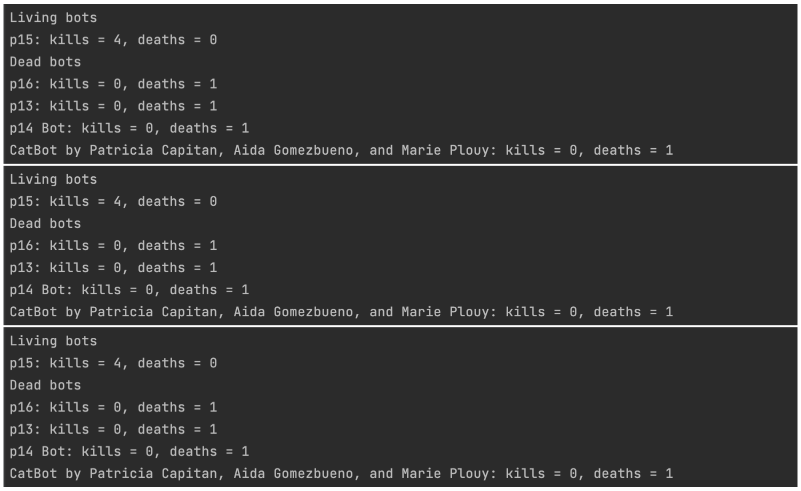
```
Living bots
Dead bots
p10: kills = 0, deaths = 1
p5: kills = 0, deaths = 1
p2: kills = 0, deaths = 1
p1: kills = 0, deaths = 1
p9: kills = 1, deaths = 1
p4: kills = 2, deaths = 1
p12 bot by Jeff Cope: kills = 1, deaths = 1
p8: kills = 3, deaths = 1
p6: kills = 2, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 0, deaths = 1
p7: kills = 2, deaths = 1
p3: kills = 1, deaths = 1
```



```
p6: kills = 5, deaths = 0
Dead bots
p10: kills = 0, deaths = 1
p4: kills = 0, deaths = 1
p9: kills = 0, deaths = 1
p7: kills = 0, deaths = 1
p2: kills = 1, deaths = 1
p12 bot by Jeff Cope: kills = 1, deaths = 1
p8: kills = 1, deaths = 1
p3: kills = 0, deaths = 1
p1: kills = 0, deaths = 1
p5: kills = 0, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 3, deaths = 1
```

Sample Outputs for Plist

Tlist



```
Living bots
p15: kills = 4, deaths = 0
Dead bots
p16: kills = 0, deaths = 1
p13: kills = 0, deaths = 1
p14 Bot: kills = 0, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 0, deaths = 1
```

```
Living bots
p15: kills = 4, deaths = 0
Dead bots
p16: kills = 0, deaths = 1
p13: kills = 0, deaths = 1
p14 Bot: kills = 0, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 0, deaths = 1
```

```
Living bots
p15: kills = 4, deaths = 0
Dead bots
p16: kills = 0, deaths = 1
p13: kills = 0, deaths = 1
p14 Bot: kills = 0, deaths = 1
CatBot by Patricia Capitan, Aida Gomezbueno, and Marie Plouy: kills = 0, deaths = 1
```

Sample Outputs for Tlist

Conclusion

We enjoyed working on this project. Although it took us time to figure out the moving angles and how to use them, we made it to the end and configured a bot that is able to beat all the given random bots and some of the pbots as well. We might not have the best bot, but overall it performs decently, to the point of beating all the p1-p12 bots once out of ten runs, and getting top 3 six times out of 10.

Our bot's best characteristic is its attack mechanism. If not targeted, our bot moves towards the closest bot and attacks it. However, our defense mechanism can be improved. Our bot does not have an optimal way to avoid bullets, and therefore sometimes runs into one while trying to avoid another. If our bot's defense was stronger, it could get closer than it already is when attacking, resulting in more accurate shooting and kills.