Artificial intelligence competition

Context

Each years, in the city of LittleSandBox, a war rage. The best programmers from around the world, armed with their tank, regroupe to demonstrate their AI programming skill. The goal is to keep your tank alive will fighting the tank of the other team. The last team standing will be declared winner of this years.

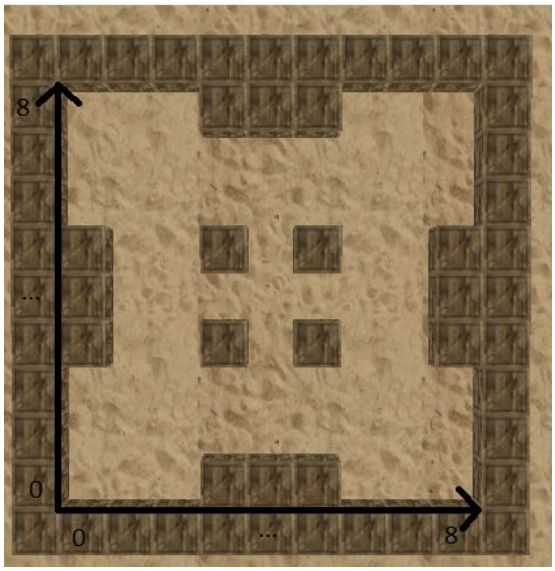
Game description

Two team will control **two tanks** to **eliminate their opponent tanks**. You must implement the artificial intelligence for your two tanks to eliminate the tanks of your opponent. Your team will be compare with the other participant team.

You must note that a none working artificial intelligence(infinite loop, crashing code) will give a forfeit victory to its opponent. Because your A.I. send real time order, your A.I. can take all the time it needs. But, a faster A.I. will send its order faster, which is an advantage. All the calculation are made on the server side.

World description

The size of the map is fixed and it have a border made with box that prevent the tanks to get out of the map. The map is 9 tiles long by 9 tile wide for a total of 81 possible tile to move on. A total of 16 tiles have a box on it, a box cannot move, which reduce the possible moving tile to 65. The coordinate of the tile and their axes are shown on the next image.



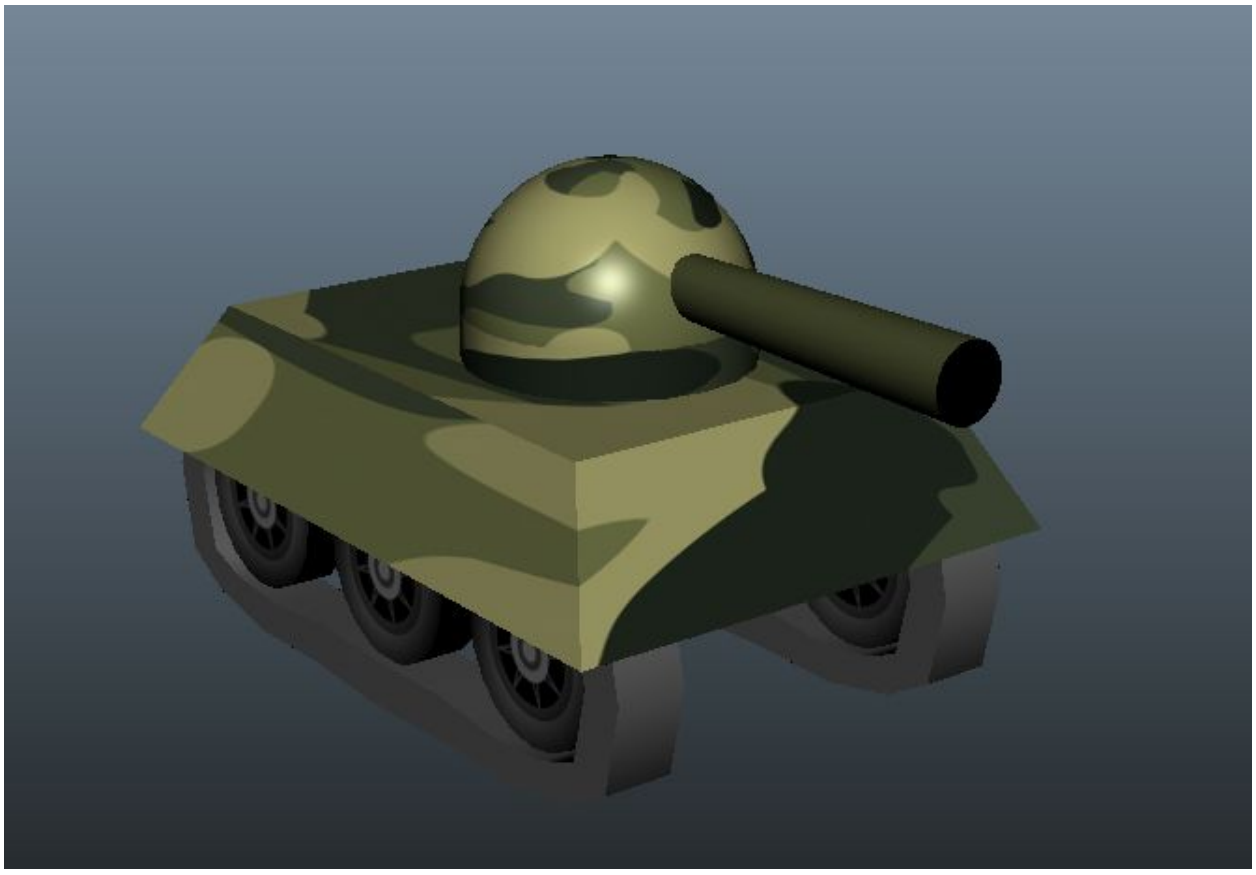
Team

Each team control two tanks. While at least one of the two tank is still alive, the team can continue to send order to this tank.

Tank

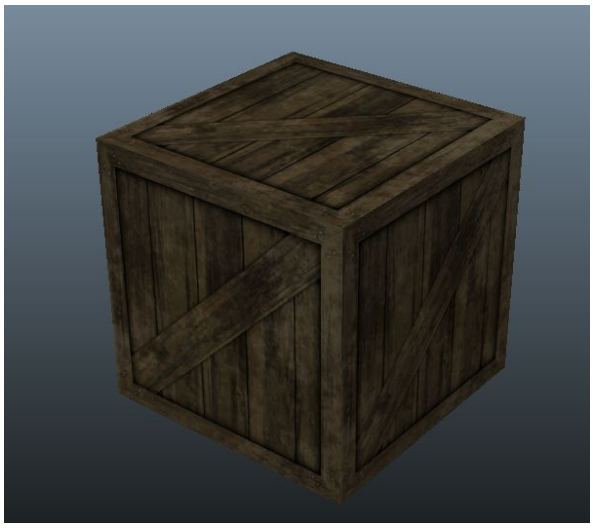
The tanks execute individual order. The starting position are given randomly between the 4 corners of the map, which mean the points: [(0,0), (0,7), (7,0), (7,7)]. Each tank start with a total of **3 life point**. The life total for the tank cannot increment during the game. When a tank collide with a mine or a missile it loose one life point. All the possible action for the tank will be describe in the section “Possible action for the A.I.”.

Only one tank can be on a tile. If a tank have order to move on a tile occupied by another tank, it will try to find a alternative path. There’s **no tank collision possible**.



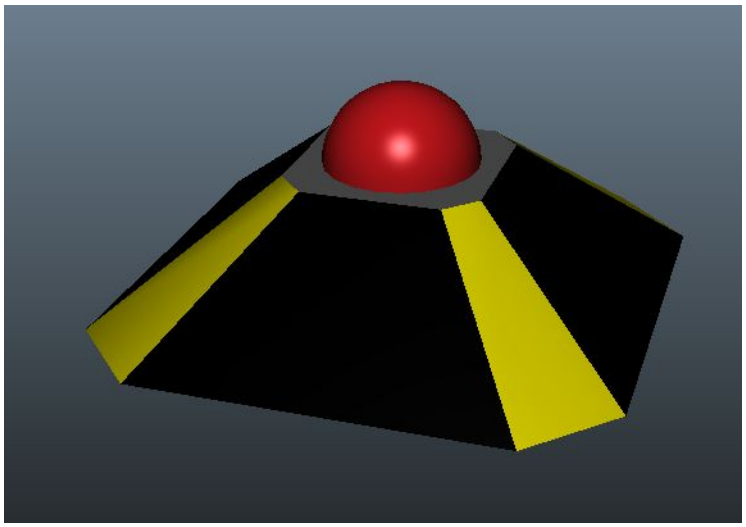
Box

The box doesn't move during the game, every team have access to their position. You can use the function “isBoxAtPosition” from the object “World” to check if there’s a box on a certain tile.



Mines

The mines are drop by the tank but doesn’t appear on the map. You must watch where you put your feet. **Only one mine at a time** can be drop by each tank. A tank will be able to drop another mine only if the first one is destroy. A mine can only be drop on a tile not between two tile. If a tank have the order to drop a mine but the tank is between two tiles, it will wait to finish his movement before dropping the mine. If a missile collide with a mine the missile and the mine will be destroy.



Missiles

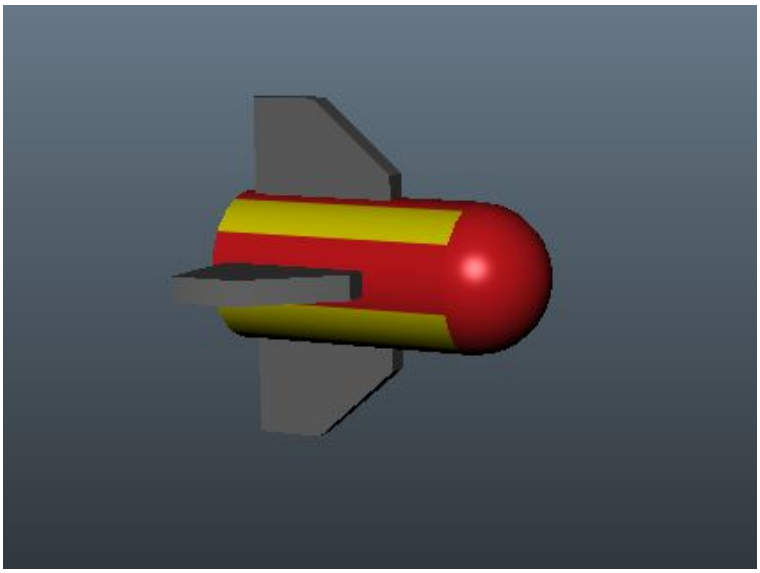
The missiles are thrown by the tanks and appear on the map. The missiles are thrown in a specific direction that cannot change. You can use the function “isMissileAtPosition” from the “World” object to check if there’s a missile at a certain position. Or you can access the missiles information by the “Character” object, according to this pseudocode:

```
world.getOpponentTeam().getFirstCharacter().getMissile().getPosition()
```

The same goes for the direction of the missile, according to this pseudocode:

```
world.getOpponentTeam().getFirstCharacter().getMissile().getDirection()
```

Only one missile can be throw for each tank. A tank will be able to shoot another missile only if the first one is destroy. If a missile enter in collision with a mine the missile and the mine will be destroy. A missile **move 2 times faster** than a tank.



Possible actions for the A.I.

The function “tick()” from your A.I. will be call every 60ms. The function “tick()” is on a single thread, so if it takes more time to compute the order it will takes more time to send them. All the orders are sent at the end of the function “tick()”.

Move

The order to move is given to a single tank with a target position. It’s the “GameClient” that will compute the path for the tank to the target position. **No need to code a “Path Finding” algorithm.** If the target tile is not accessible the tank will continually try to go to that position.

Use case:

Python:

```
aCharacter.goTo(Vector2(6,7))
```

Java:

```
aCharacter.move(new Point(5,5));
```

Shoot

The shoot order is given to a single tank with a direction. The tank will try to shoot once it get on the next tile. If the tank is between two tile, it will wait to be on the next tile before shooting. The tank will **stop moving for one second** after shooting. If the missile collide on a tile next to the tank it will **lose one life point**.

Python:

```
aCharacter.shootMissile(Direction.UP)
```

Java:

```
character.shootMissile(Missile.Direction.UP);
```

Drop a mine

The drop mine order is given to a single tank. The tank will drop its mine when he receive the order to do so. If the tank is between two tile he will go to the next tile before dropping the mine.

Python:

```
aCharacter.dropMine()
```

Java:

```
character.dropMine();
```

Client implementation

Like mention before your order will be send every time the function “tick()” end.
There’s a A.I. example at your disposition.
Python:

AIClient_Python/src/aiclient/AI.py

Java:

AIClient_Java/src/aiclient/AI.java

Change the names

To change the name of your team and the names of your tanks, you can refactor the function “setNames” from your AI. An example is given in the base A.I.

Technical details

The game is compose by two modules. The first one is the “GameClient” module that do all the calcul and give you the 3D environment. It’s in some way the server that your A.I. will connect to. A game start when two A.I. are connected to the server. The two A.I. connected to the “GameClient” are not necessary in the same programming language, so the A.I. is completely independant from the server.

GameClient

The GameClient must be start before your A.I. so the game can start. A script has been made to start the server module:

```
./start.sh
```

Interface navigation

It’s possible to navigate in the 3d environment with the “wasd” keys. You can go in a first person vue if you click with the mouse.

AIClient

The “AIClient” will be connected to the “GameClient” when you start it. You need two A.I. to start a game.

Java

There’s a script to start the Java client:

```
./start_java.sh
```

Python

There’s a script to start the Python client:

```
./start_python.sh
```

Documentation

All the documentation useful for the A.I. developpement can be found with the client code:

Python:

```
/AIClient_Python/docs/index.html
```

Java:

```
/AIClient_Java/docs/index.html
```

Correction

For the correction of the code every A.I. will be revised to be sure that all the rules are respected. Only the files in the “aiclient” folder will be taken. Your A.I. will play against

the A.I. of the other teams to do a first ranking. After that a selection math will determine the best A.I.

Conclusion

Good luck to all of you.
GLHF!