



DMSC AI Tournament



Third edition



The day's program

Schedule:

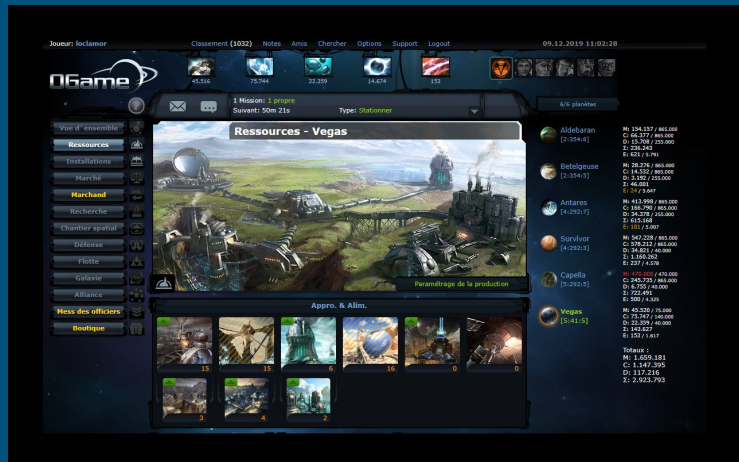
- 13:00 - 13:30: Game presentation
- 13:30 - 18:00: Work individually on your AI (use the DMSC office space)
- 18:00 - 19:30: Tournament (and food!) in Curie/Holmes

How we run this day:

- Have fun with the challenge
- It is (most probably) possible to cheat, please show good sportsmanship

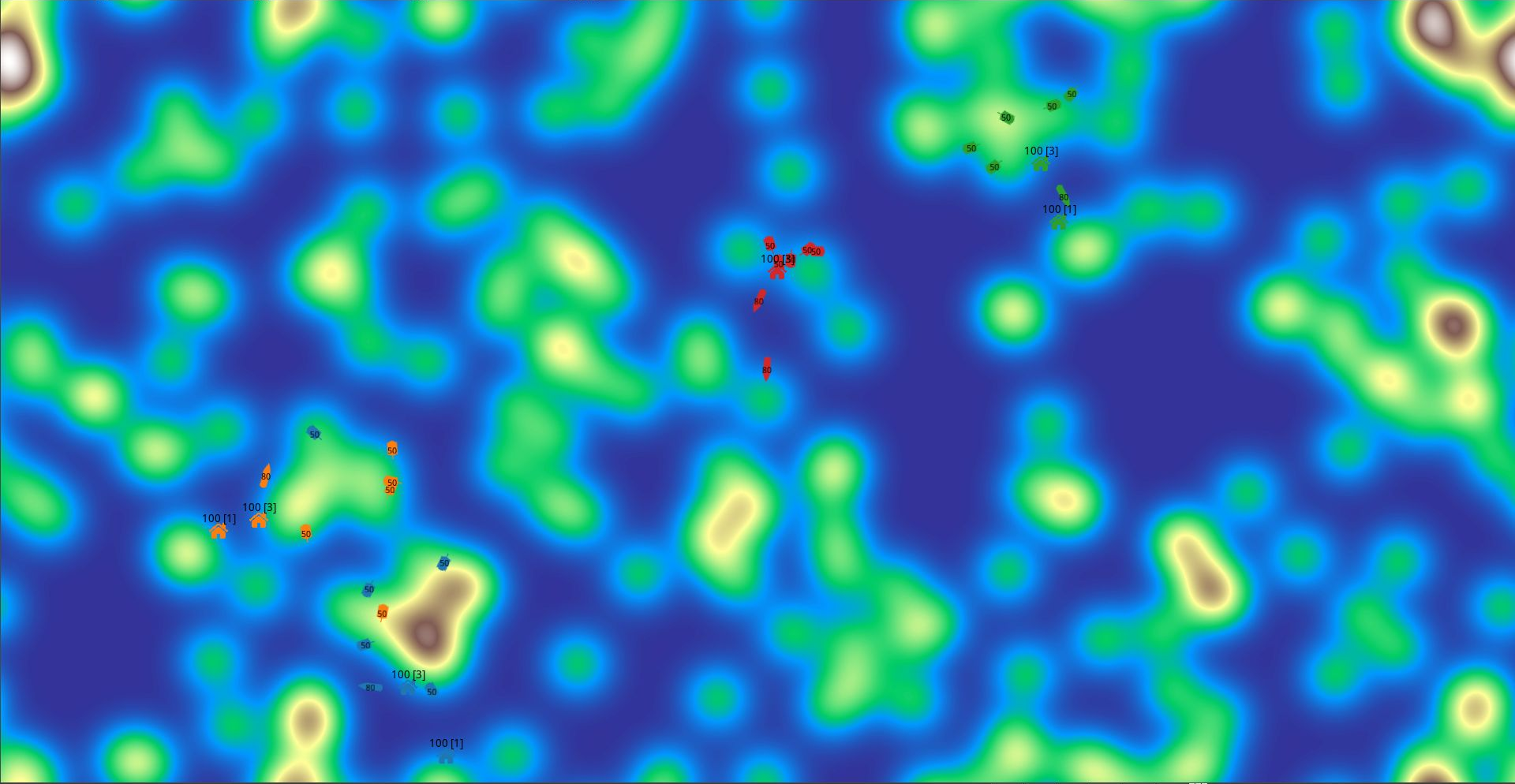
About the game

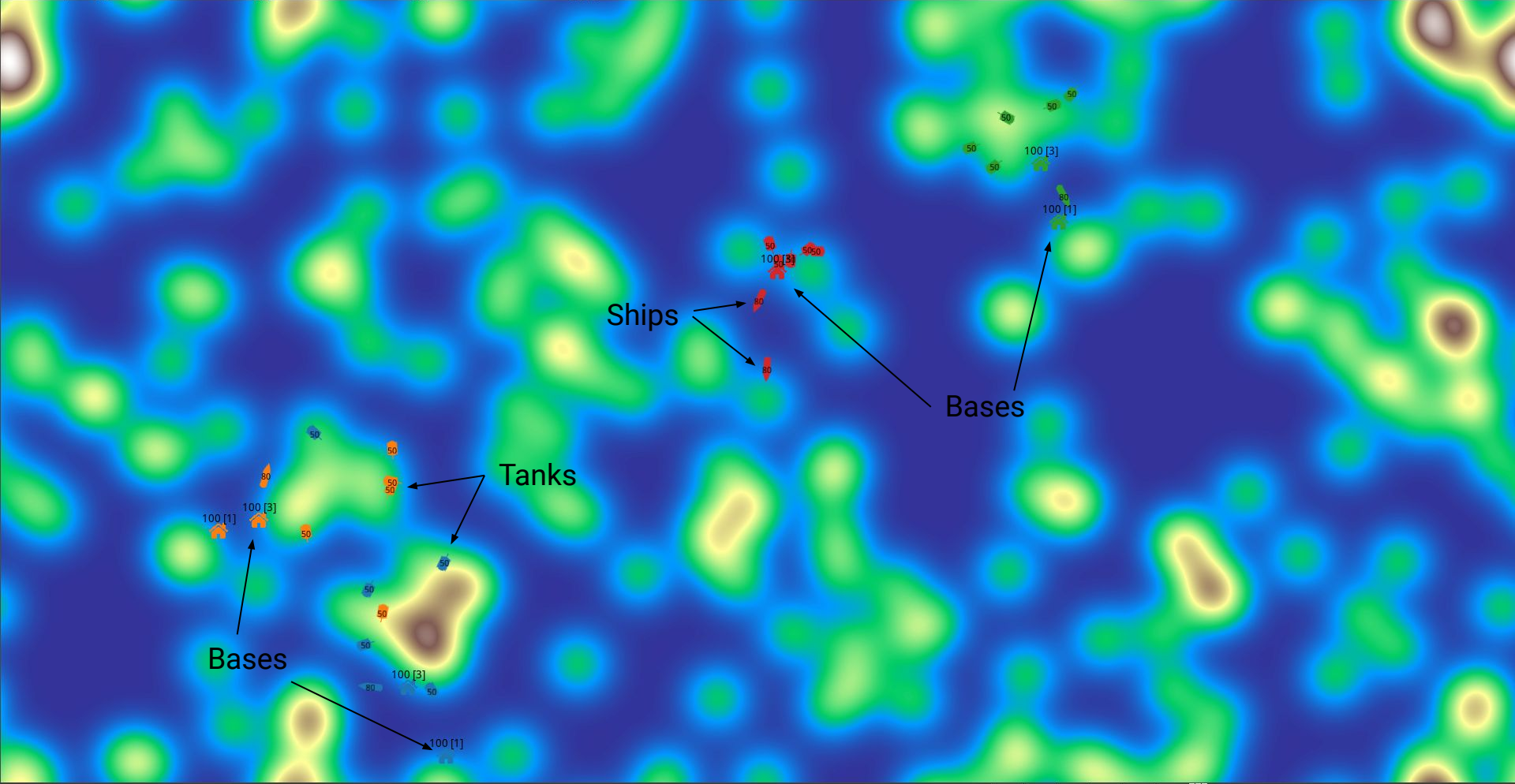
- Inspired by Starcraft, OGame, ...
- Implementation reused almost nothing from “Quest”
- Python with **pyglet** graphics (1300 lines, 0 unit tests)
- Install via **pip install .**

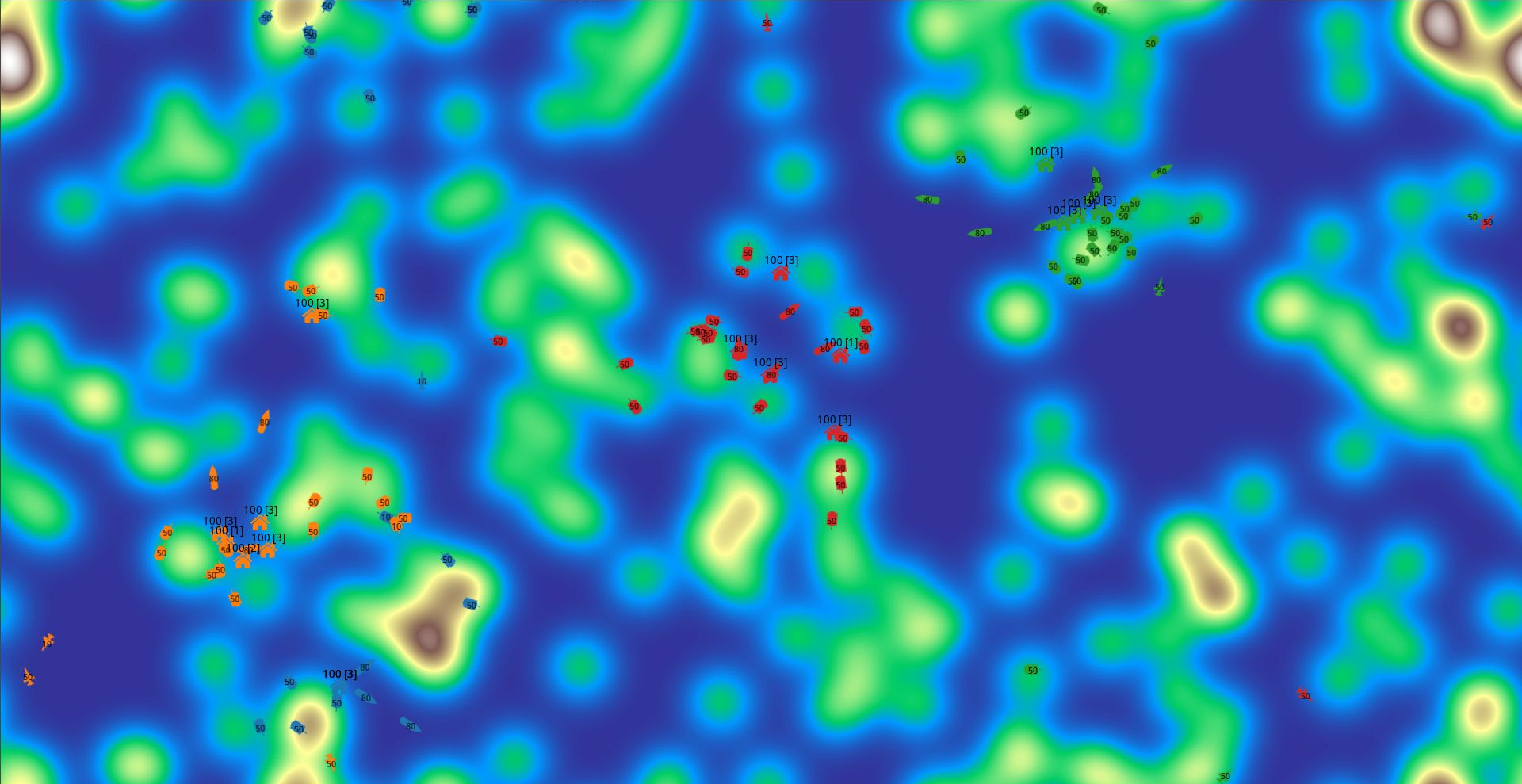


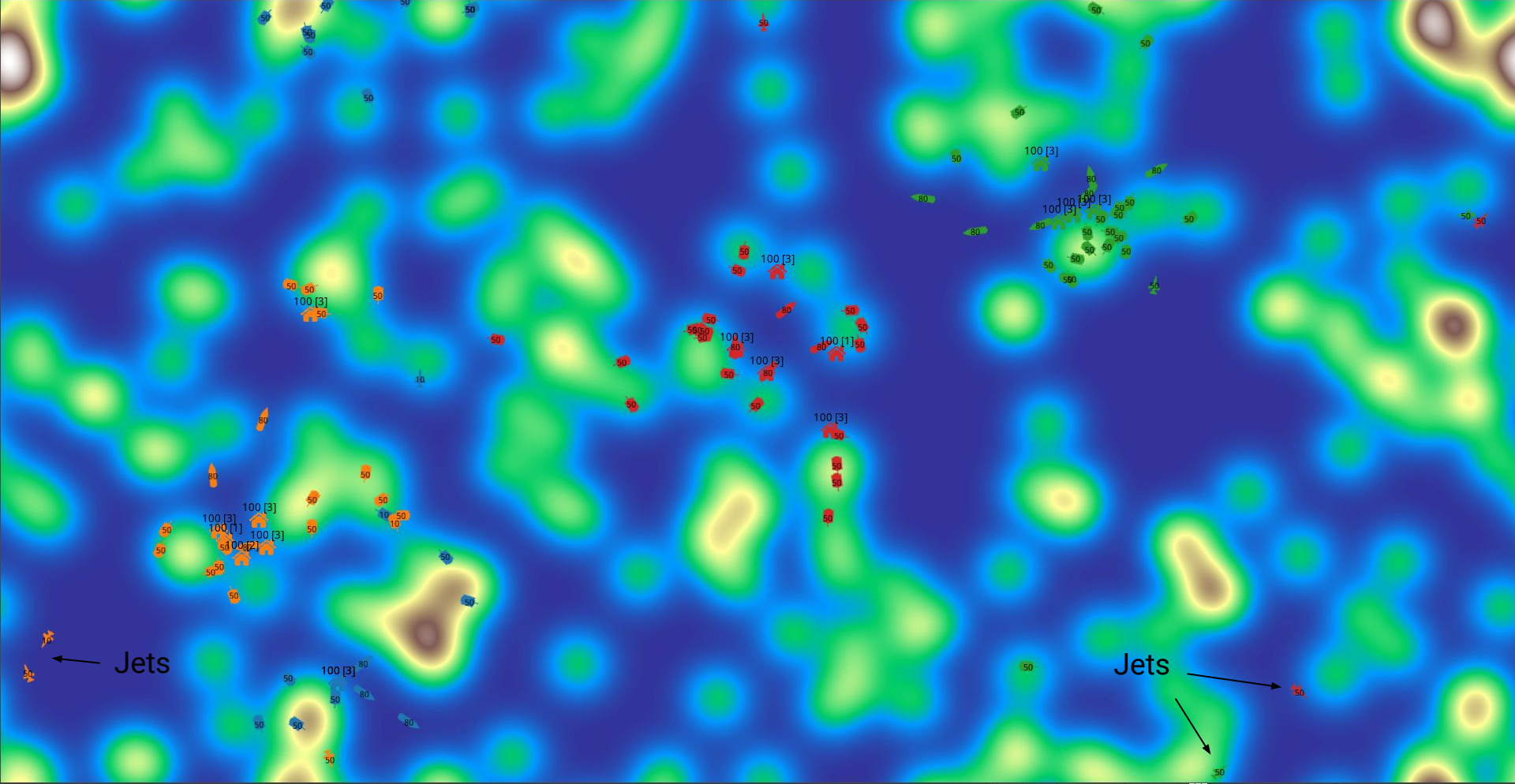


SUPREMACY









Game rules (1/3)

Goal:

- Mine resources to build and army
- Destroy enemy bases and eliminate other players
- All players play on the map at the same time
- Each round lasts 8 minutes

Game map:

- Dimensions: $nx = 1920$; $ny = 992$
- Coordinate system: lower left = $(0, 0)$, upper right = $(1920, 992)$
- Periodic boundary conditions



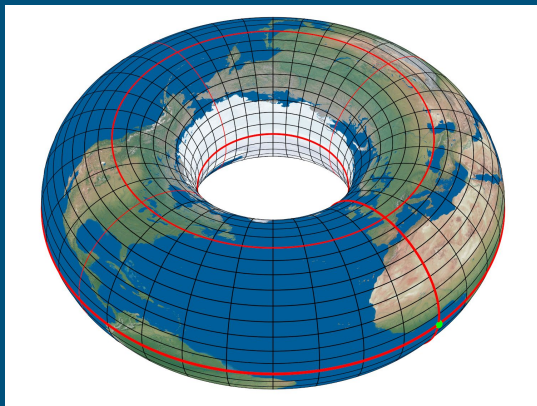
Game rules (1/3)

Goal:

- Mine resources to build and army
- Destroy enemy bases and eliminate other players
- All players play on the map at the same time
- Each round lasts 8 minutes

Game map:

- Dimensions: $n_x = 1920$; $n_y = 992$
- Coordinate system: lower left = $(0, 0)$, upper right = $(1920, 992)$
- Periodic boundary conditions



Game rules (2/3)

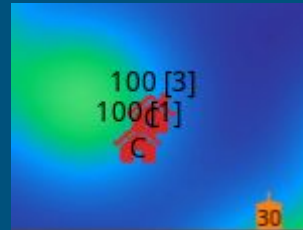
Mining

- Everyone starts with 1 base, housing 1 mine
- Every timestep, each mine will extract $\text{crystal} = 2 * \text{number_of_mines}$
- Crystal is used to build mines and vehicles
- Mines too close to other mines compete for resources:

$$\text{crystal} = 2 * \text{number_of_mines} / \text{number_of_bases_inside_square_of_80px}$$

Fights

- Whenever two or more vehicles or bases from opposing teams come within 5px from each other, they will fight
- Every object hits all the others with its attack force, and it takes damage from all other objects
- No cooldown, fights are resolved (almost) instantly (in a single time step)



Vehicles



	Tank	Ship	Jet
Speed	10	5	20
Attack	20	10	30
Health	50	80	50
Cost	500	2000	4000
Can travel	On land	On sea	Anywhere
		Turns into base	

Game rules (3/3)

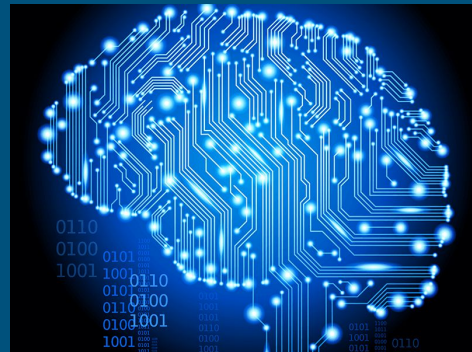
- Mine cost is $\times 2$ for every new mine on a given base (first mine = 1000)
- Base has health = 100, mine has health = 50 (both have 0 attack)
- Vehicles move at $\text{speed} * dt$ ($dt = 1/30s$)
- A ship can be turned into a new base, by calling `convert_to_base()`
- Only works if there is land in the immediate vicinity
- If a player dies, all his/her vehicles disappear

Scoring

- 1 point if you destroy a base
- If a player dies, gets a number of points equal to the number of dead players
- At the end of the round, every player still alive gets points equal to the number of dead players

Demo!

The control center - the AI (1/3)



Info (dict) it received every timestep

- One entry per player (including yourself)
- Inside each entry, a list of **'bases'**, **'tanks'**, **'ships'**, and **'jets'**

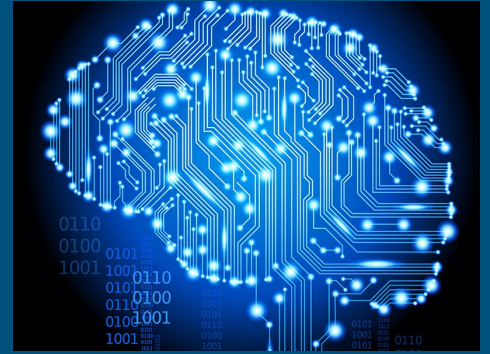
Base info

- **.x**: x position
- **.y**: y position
- **.team**: e.g. **'John'**
- **.number**: player number
- **.mines**: number of mines
- **.crystal**: amount of crystal
- **.uid**: unique id

Vehicle info

- **.x**: x position
- **.y**: y position
- **.team**: e.g. **'John'**
- **.number**: player number
- **.speed**, **.health**, **.attack**
- **.heading**, **.vector**, **.position**
- **.uid**: unique id

The control center - the AI (2/3)

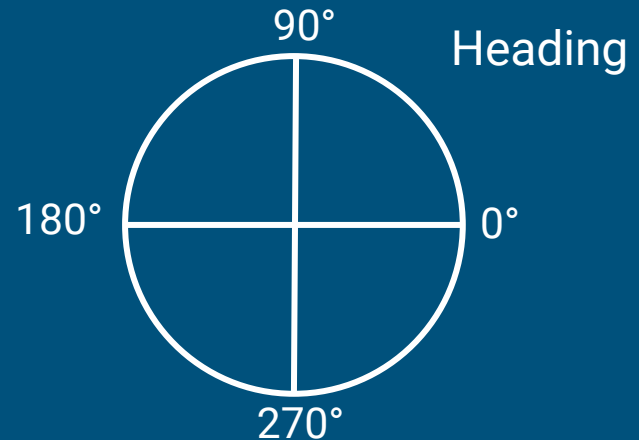


Base methods

- `cost('mine')`: get the cost of an object
- `build_mine()`, `build_tank()`, `build_ship()`, `build_jet()`

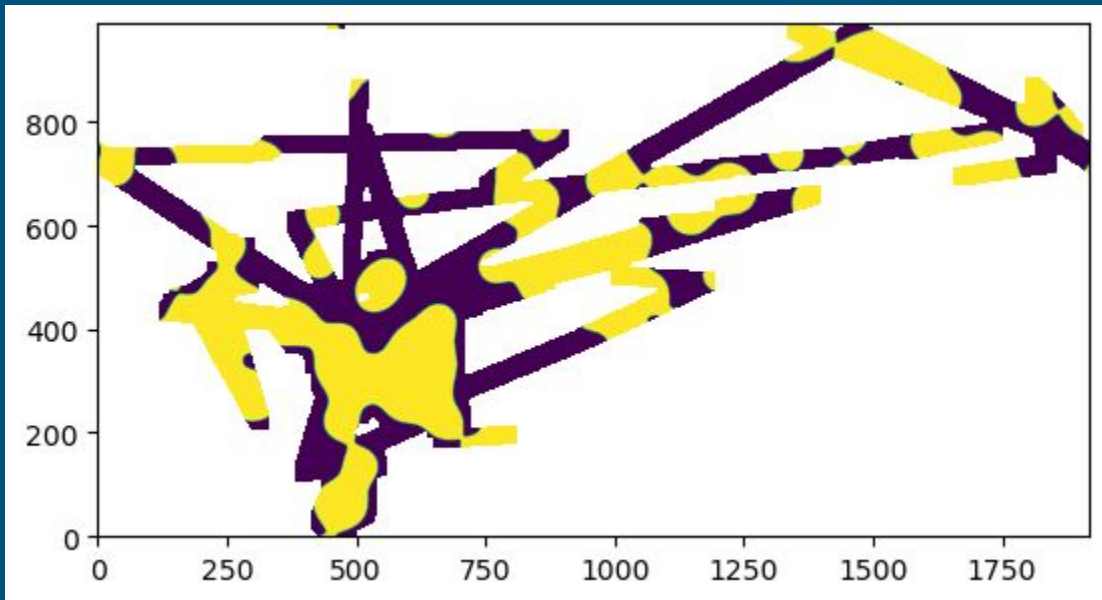
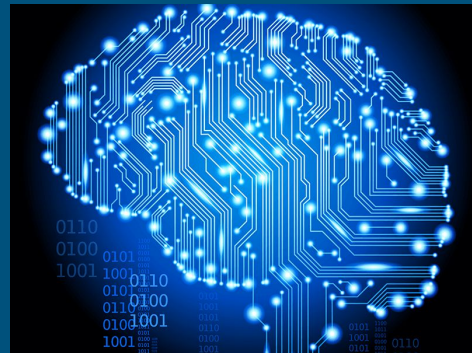
Vehicle methods

- `get_position()`: returns `np.array([x, y])`
- `get_heading()`: returns angle in degrees
- `set_heading(angle_in_deg)`
- `get_vector()`: returns `np.array([vx, vy])`
- `set_vector(np.array([vx, vy]))`
- `goto(x, y)`
- `get_distance(x, y)`
- `convert_to_base()`



The control center - the AI (3/3)

Game map is filled in for you!



- 0 = sea
- 1 = land
- -1 = no info

Template AI

Page 1

```
3 import numpy as np
4
5 CREATOR = 'JohnDoe'
6
7
8 class PlayerAi:
9
10     def __init__(self):
11         self.team = CREATOR
12         self.previous_positions = {}
13         self.ntanks = {}
14         self.nships = {}
15
16     def run(self, t: float, dt: float, info: dict, game_map):
17
18         myinfo = info[self.team]
19         for base in myinfo['bases']:
20             if base.uid not in self.ntanks:
21                 self.ntanks[base.uid] = 0
22             if base.uid not in self.nships:
23                 self.nships[base.uid] = 0
24             if base.mines < 3:
25                 if base.crystal > base.cost('mine'):
26                     base.build_mine()
27             elif base.crystal > base.cost('tank') and self.ntanks[base.uid] < 5:
28                 base.build_tank(heading=360 * np.random.random())
29                 self.ntanks[base.uid] += 1
30             elif base.crystal > base.cost('ship') and self.nships[base.uid] < 3:
31                 base.build_ship(heading=360 * np.random.random())
32                 self.nships[base.uid] += 1
33             elif base.crystal > base.cost('jet'):
34                 base.build_jet(heading=360 * np.random.random())
35
```

Template AI

Page 2

```
36     target = None
37     if len(info) > 1:
38         for name in info:
39             if name != self.team:
40                 if 'bases' in info[name]:
41                     t = info[name]['bases'][0]
42                     target = [t.x, t.y]
43
44     if 'tanks' in myinfo:
45         for tank in myinfo['tanks']:
46             if tank.uid in self.previous_positions:
47                 if all(tank.position == self.previous_positions[tank.uid]):
48                     tank.set_heading(np.random.random() * 360.0)
49                 elif target is not None:
50                     tank.goto(*target)
51                 self.previous_positions[tank.uid] = tank.position
52
53     if 'ships' in myinfo:
54         for ship in myinfo['ships']:
55             if ship.uid in self.previous_positions:
56                 if all(ship.position == self.previous_positions[ship.uid]):
57                     if ship.get_distance(ship.owner.x, ship.owner.y) > 20:
58                         ship.convert_to_base()
59                     else:
60                         ship.set_heading(np.random.random() * 360.0)
61                 self.previous_positions[ship.uid] = ship.position
62
63     if 'jets' in myinfo:
64         for jet in myinfo['jets']:
65             if target is not None:
66                 jet.goto(*target)
```

The high_contrast mode



What's next

Get started

- Install game from <https://github.com/nvaytet/supremacy>
- Start coding!
- Having not tried any strategies further than `templateAI`, I will also participate (but will also help at the same time)

Tournament

- 10 rounds of 8 minutes (15 min hacking allowed at half-time?)
- Alliances (and betrayals) will be key!
- My hope is that every round will end up in a giant mess!

Instructions

```
conda create -n gaming -c conda-forge python=3.10
```

```
conda activate gaming
```

```
git clone git@github.com:nvaytet/supremacy.git
```

```
cd supremacy/
```

```
python -m pip install -e .
```

```
cd tests/
```

```
python test.py
```