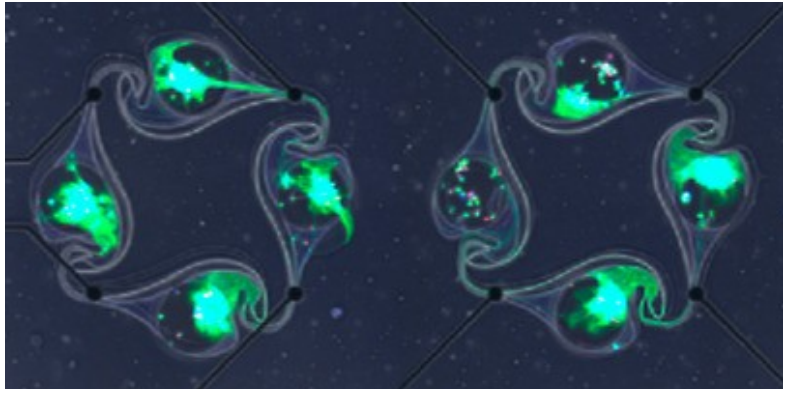
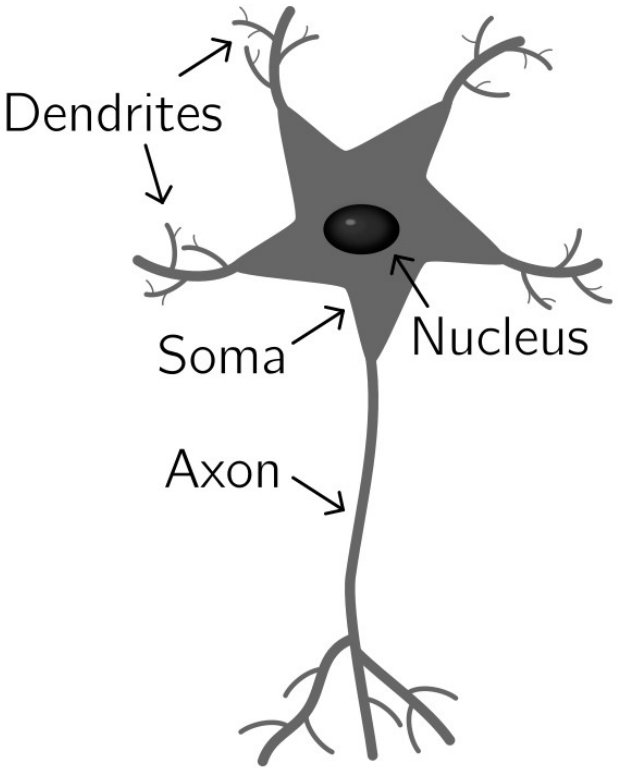
Reinforcement learning and biological networks

Stephan J. Ihle

Eric and Wendy Schmidt AI-Postdoctoral Fellowship
Biological Science Department
Physical Science Department
Data Science Institute University of Chicago



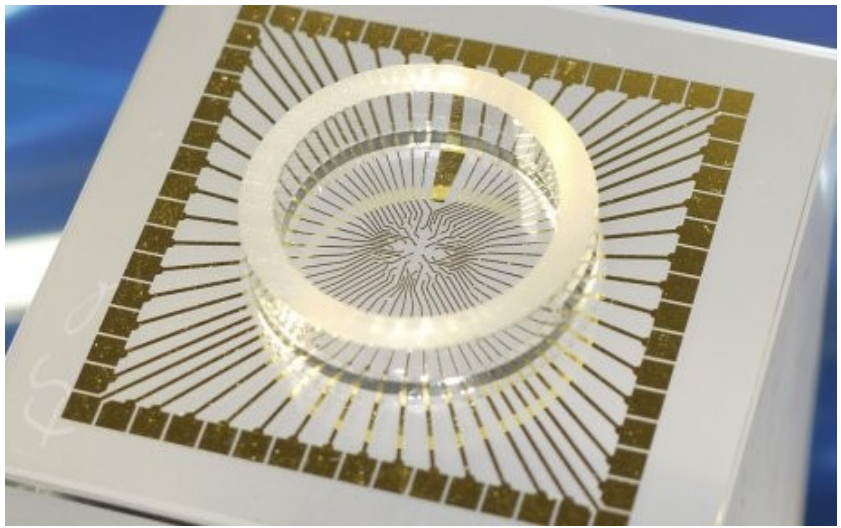
How neurons work



Adapted: Plosney and Barr, Springer Science, 2007
Hille, Biophysical journal, 1992

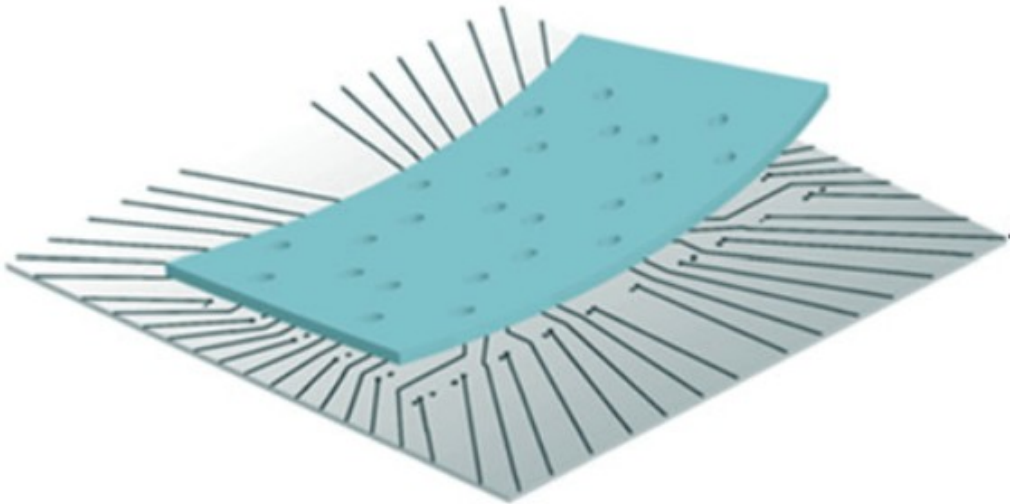
Culturing neurons in a dish

Microelectrode array for neuronal recording



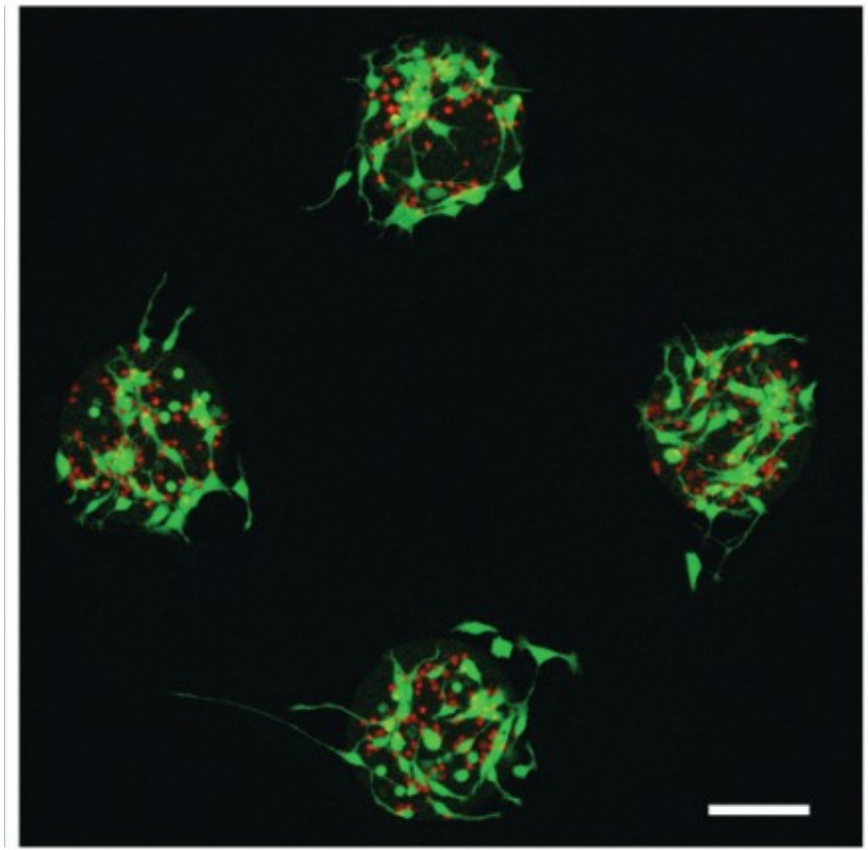
multichannelsystems.com

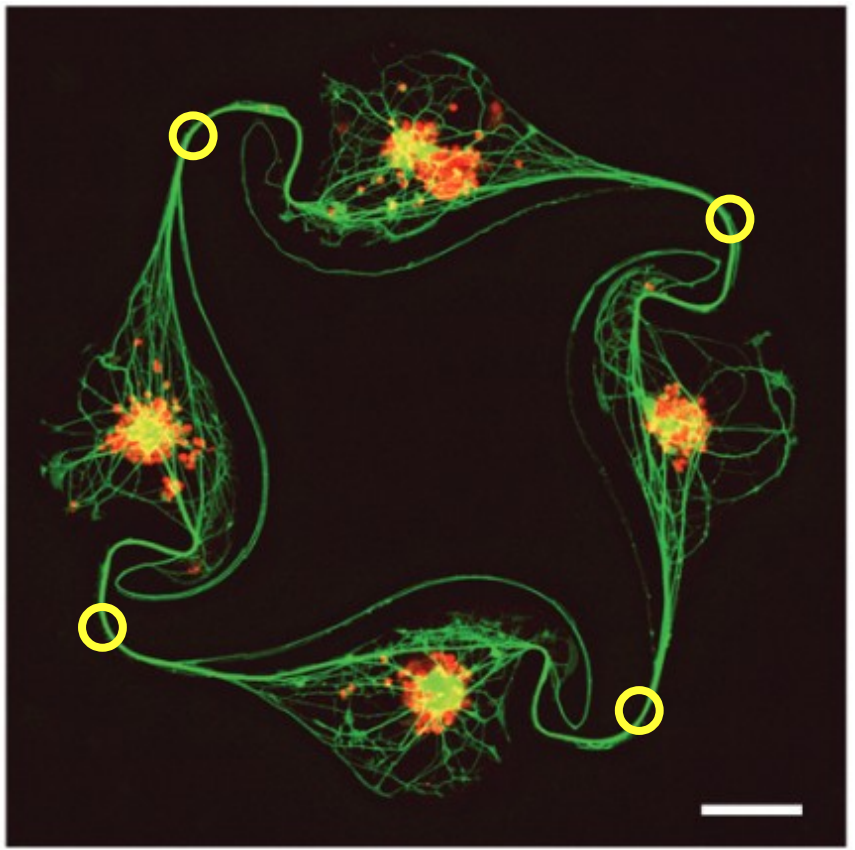
Microstructure to constrain growth



Forró *et al.*, Biosensors and Bioelectronics, 2018

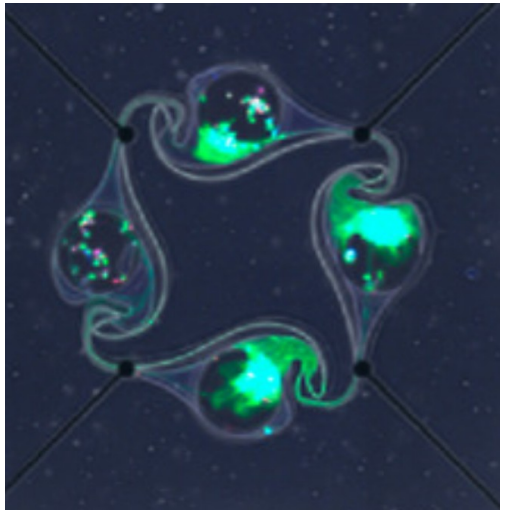
Culturing neurons in a dish

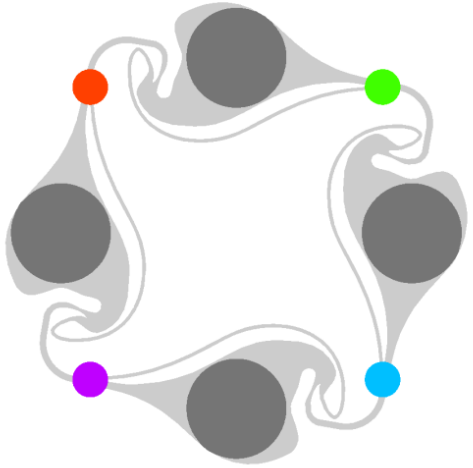




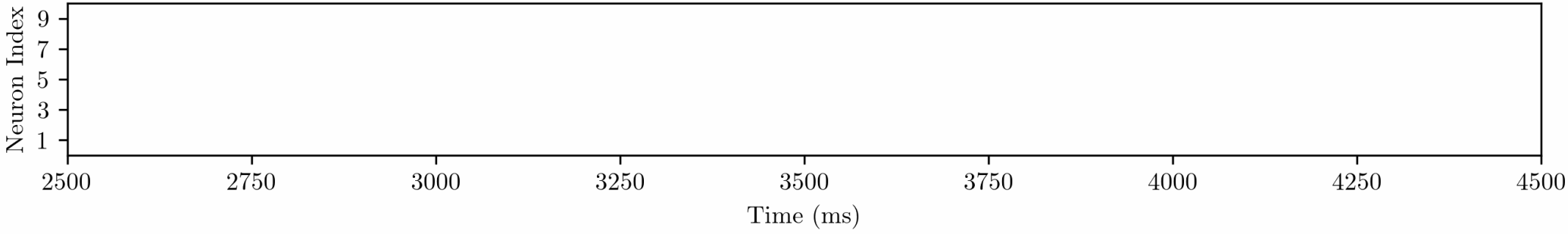
Girardin *et al.*, Lab on a Chip, 2022

Measuring neuronal activity

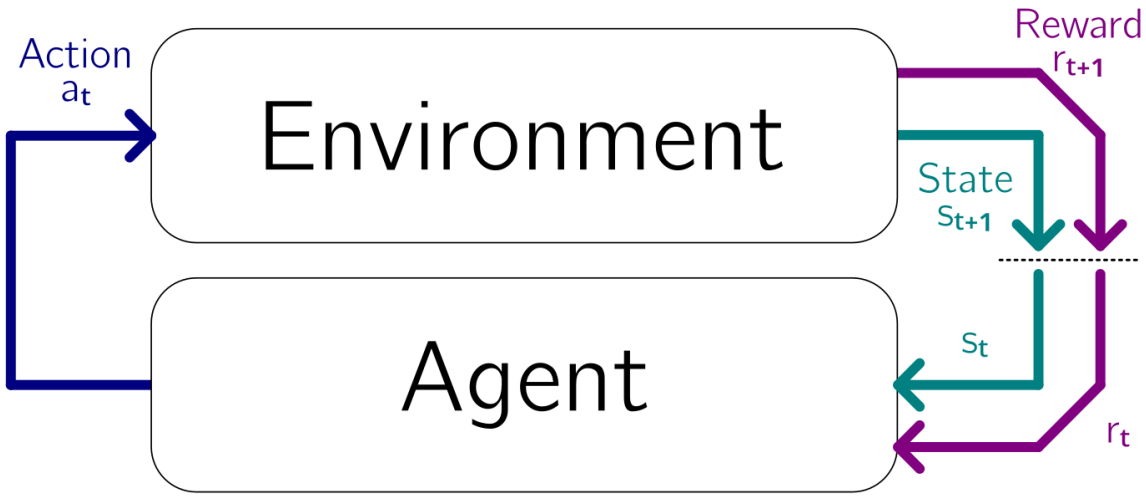




Ihle et al. 2022

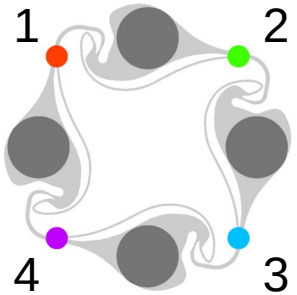


Reinforcement learning



The network is stimulated every 250 ms

What actions should you use?

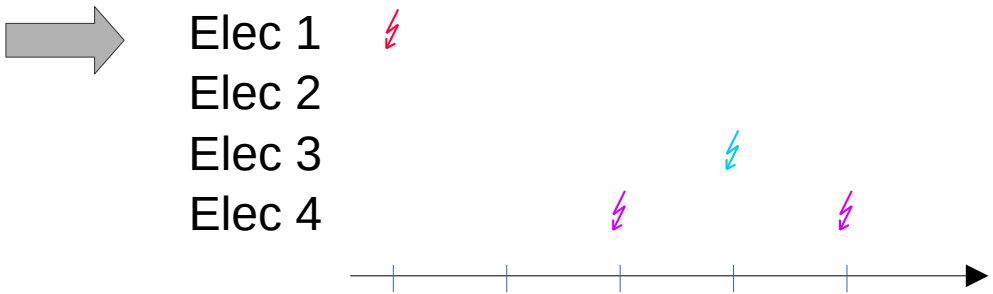


You have up to 5 slots per action.

Each slot can have value 0 (no stimulus), or numbers 1,2,3,4

Example pattern:

1	0	4	3	4



Advice: Especially at the beginning, do not use all 5 slots, but only the last 1,2, or 3

What states should you use?

There are different state functions.

Reward functions can be static (easier) or dynamic (harder). Dynamic functions need to be trained on the data but contain in general more useful information.

There exists 2 jupyter notebook examples with these two types of functions.

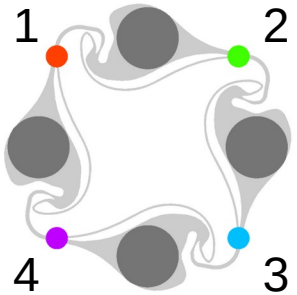
Attention: Your state representation may not be complete

Your state is not necessarily Markovian

Bonus task: Write your own state function (you will likely perform much better)

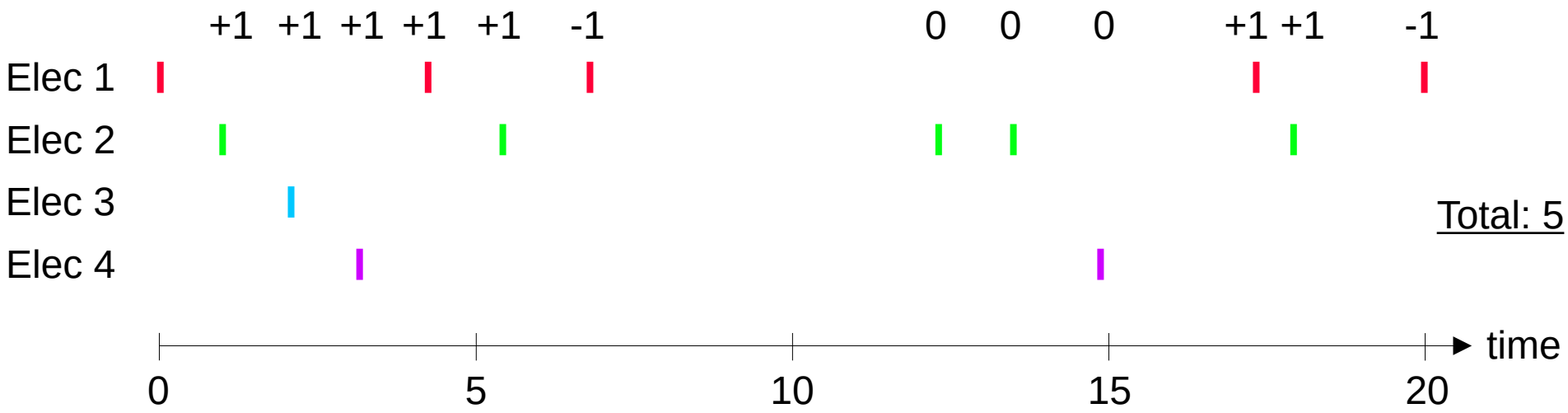
Can you figure out an approach to use the spiking activity directly as a state?

What reward function should you use?



Linear reward: +1 if 2 spikes are clockwise (and Δt < 5 ms)

-1 if 2 spikes are counter - clockwise (and Δt < 5 ms)



Bonus task: Write your own reward function (you will likely perform much better)

Your final score:

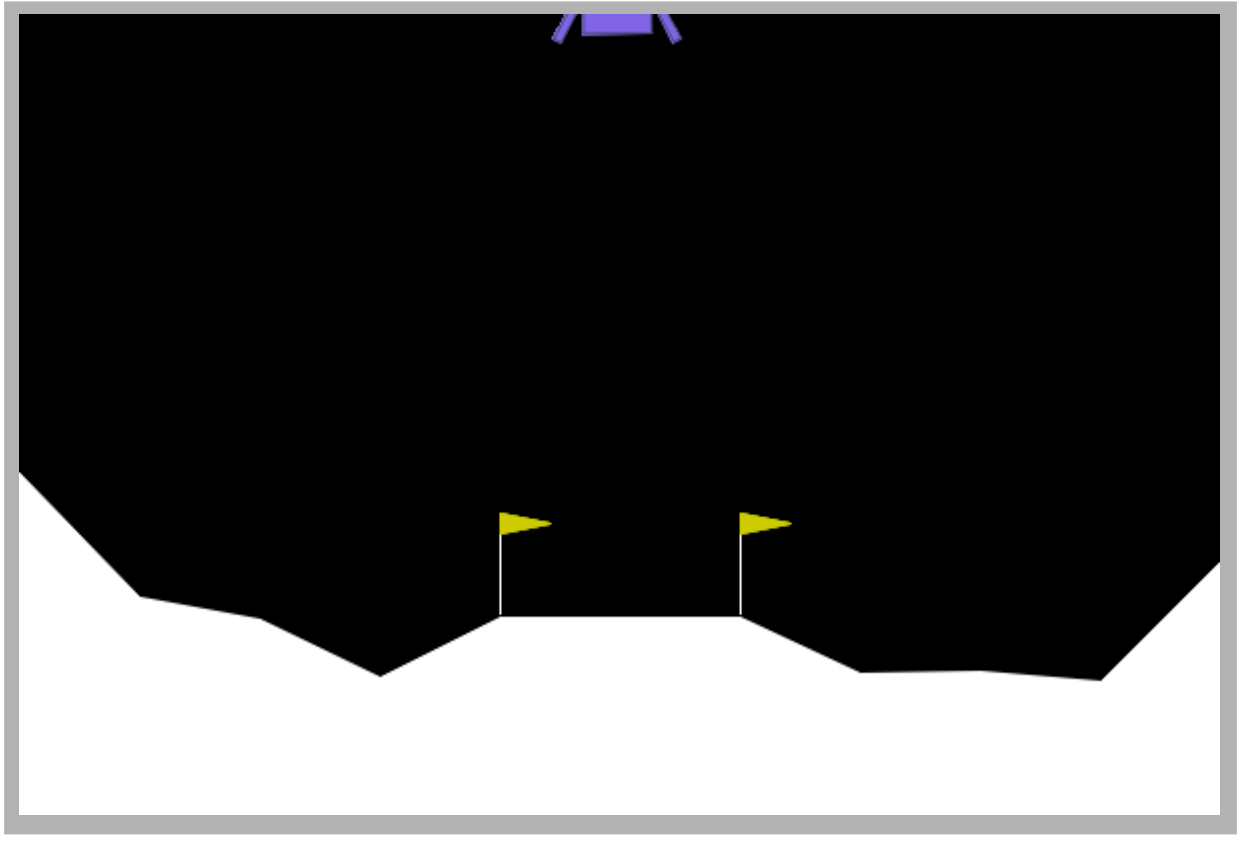
Send me your code by 5pm on XX/XX: ihles@uchicago.edu

We will run your code directly on our setup. It will run for exactly 2 hours on a network (28800 actions). The final score will be your average reward over the last 30 min of this time period (7200 actions).

For the reward function, we are using: “linear_reward()”

Important consideration: Use **RL gymnasium**

https://gymnasium.farama.org/index.html



Important consideration: Use **RL gymnasium**

```python
import gymnasium as gym

# Initialise the environment
env = gym.make("LunarLander-v3", render_mode="human")

# Reset the environment to generate the first observation
observation, info = env.reset(seed=42)
for _ in range(1000):
    # this is where you would insert your policy
    action = env.action_space.sample()

    # step (transition) through the environment with the action
    # receiving the next observation, reward and if the episode has terminated or truncated
    observation, reward, terminated, truncated, info = env.step(action)

    # If the episode has ended then we can reset to start a new episode
    if terminated or truncated:
        observation, info = env.reset()

env.close()
```


https://gymnasium.farama.org/index.html

Important consideration: We are data limited

At the end, you will have **28800 stimulation events** (actions) in total (of which your performance is judged on the last 7200).

Action space size: $5^5 = 3125$ Action space is huge for RL

Rule of thumb: Use small networks!

Number of weights in network should be sqrt(samples) → 170

Use experience replay: https://deeplizard.com/learn/video/Bcuj2fTH4_4

https://arxiv.org/abs/2007.06700

Important consideration: We are time critical

Stimulation:	Requires 10 ms
Recording:	We look at 20 ms of neuronal activity
Processing:	Spike detection / clustering takes 20 ms
Receiving data:	From laboratory computer to RCC (50 ms)
Prediction:	**You have roughly 100 ms to decide on your next action**
Sending data:	From RCC to laboratory computer (50 ms)

Use 1 network to decide on your next action and (at least) one network for training!

Why/how to use multiple networks: https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html

Happy Hacking!

Questions?