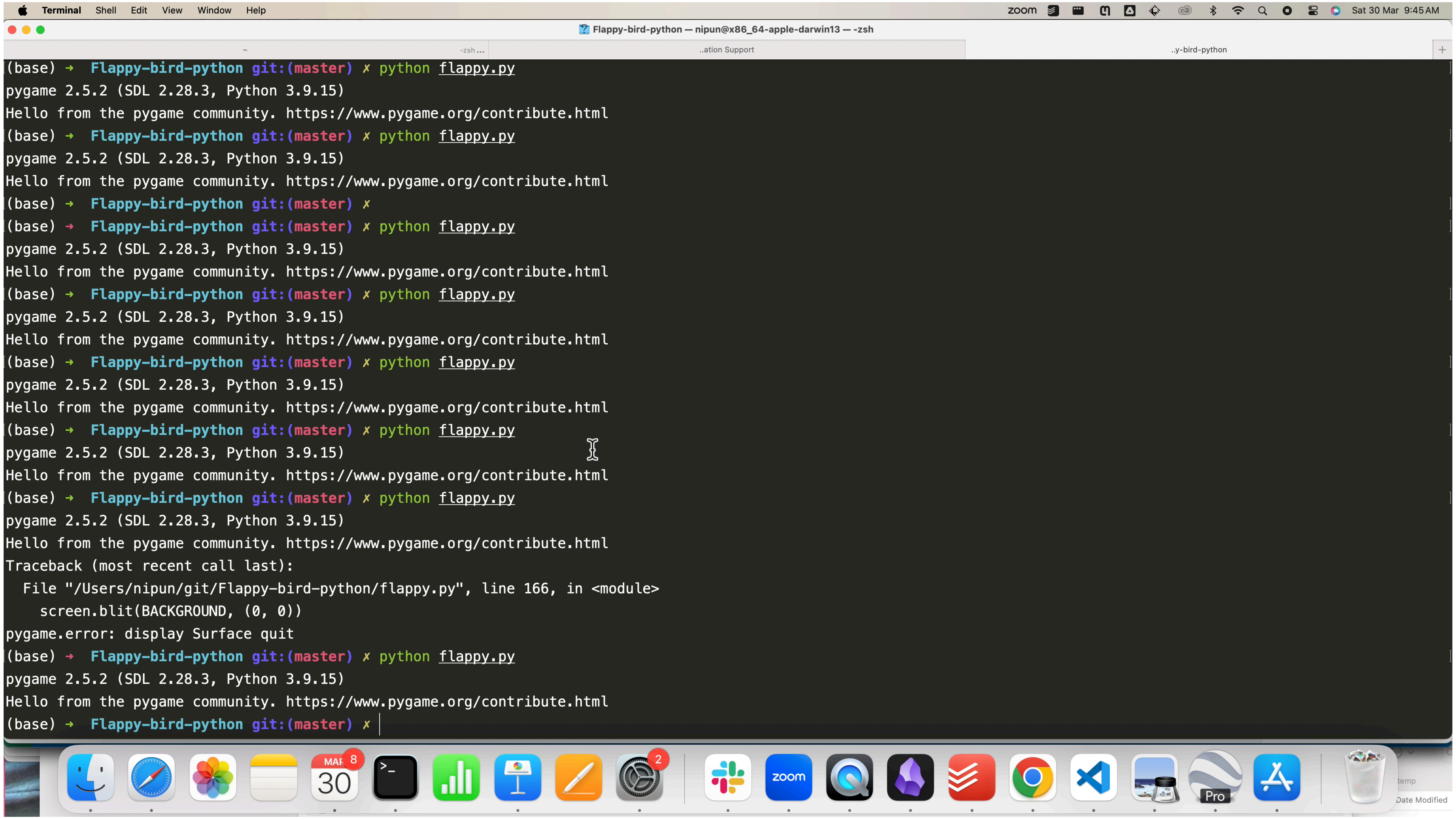
Reinforcement Learning

Nipun Batra, 1 April 2024

Flappy Bird

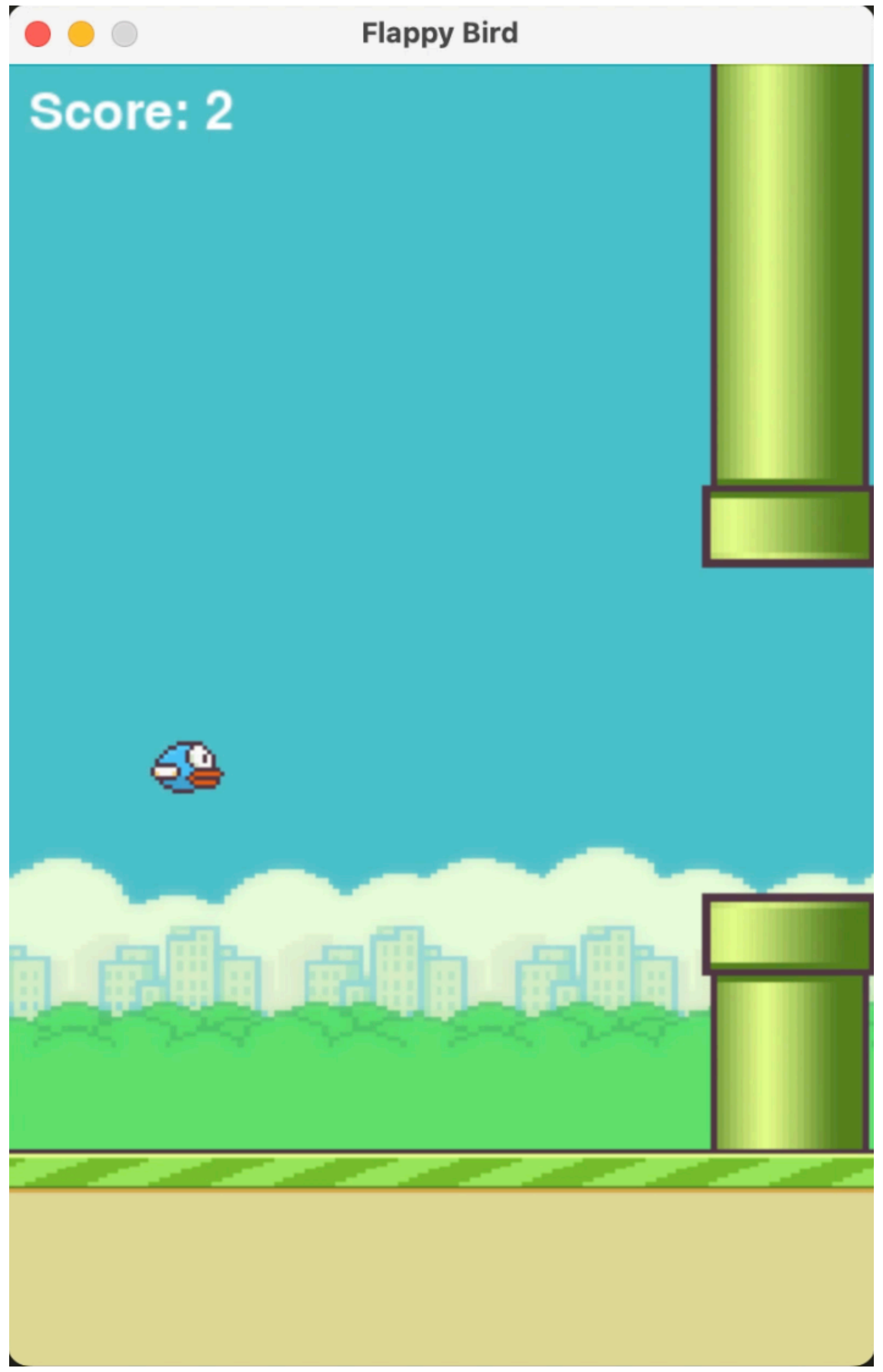

Terminal Shell Edit View Window Help
zoom
Sat 30 Mar 9:45 AM
Flappy-bird-python — nipun@x86_64-apple-darwin13 — -zsh
~
-zsh ...
..ation Support
..y-bird-python
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
Traceback (most recent call last):
File "/Users/nipun/git/Flappy-bird-python/flappy.py", line 166, in <module>
screen.blit(BACKGROUND, (0, 0))
pygame.error: display Surface quit
(base) → Flappy-bird-python git:(master) ✗ python flappy.py
pygame 2.5.2 (SDL 2.28.3, Python 3.9.15)
Hello from the pygame community. https://www.pygame.org/contribute.html
(base) → Flappy-bird-python git:(master) ✗
MAR
30
zoom
Pro
temp
Date Modified

Flappy Bird

- Game demo (Code modified from: https://github.com/LeonMarqs/Flappy-bird-python)

Flappy Bird

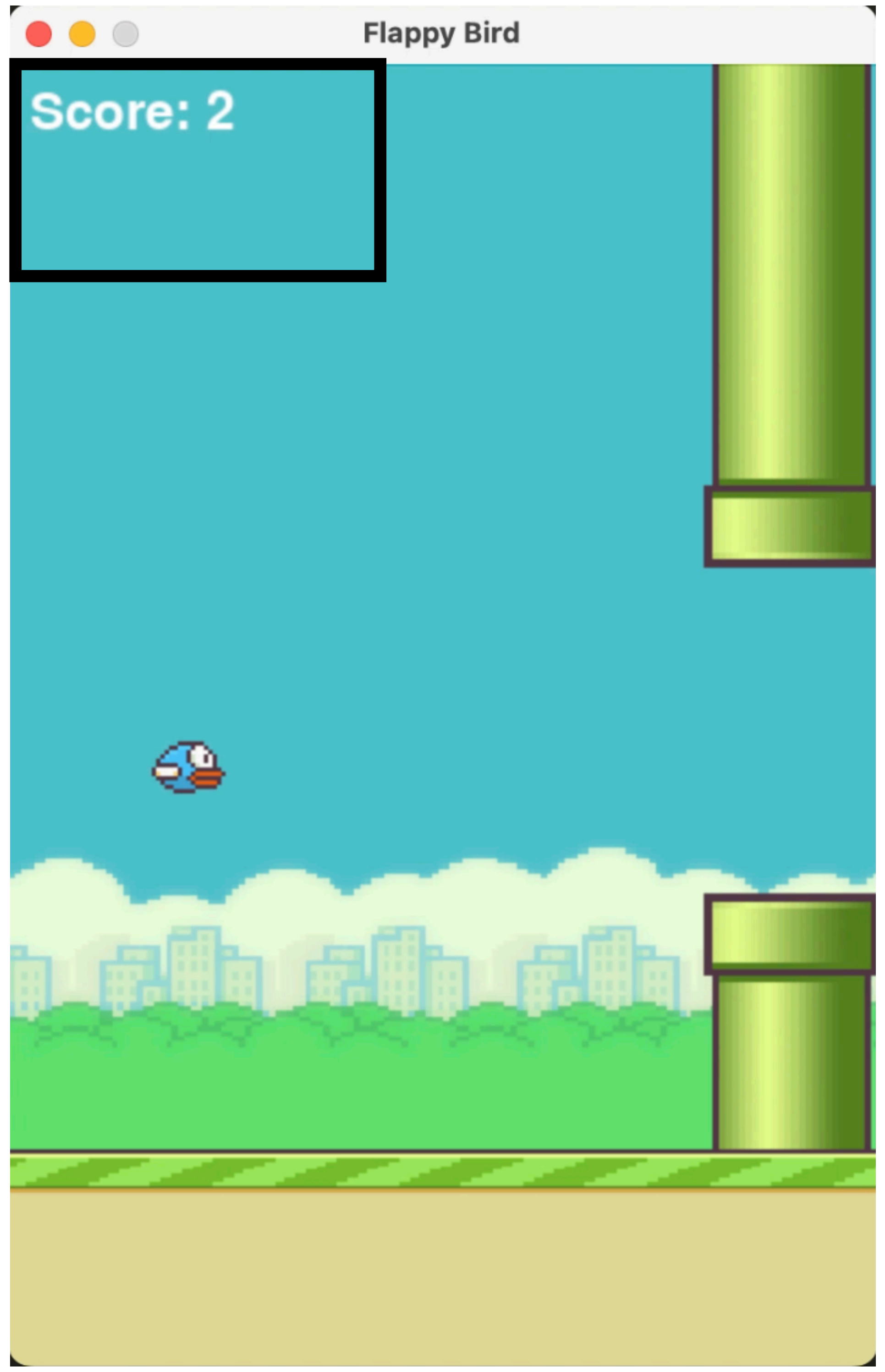
What is the goal/objective?



Flappy Bird

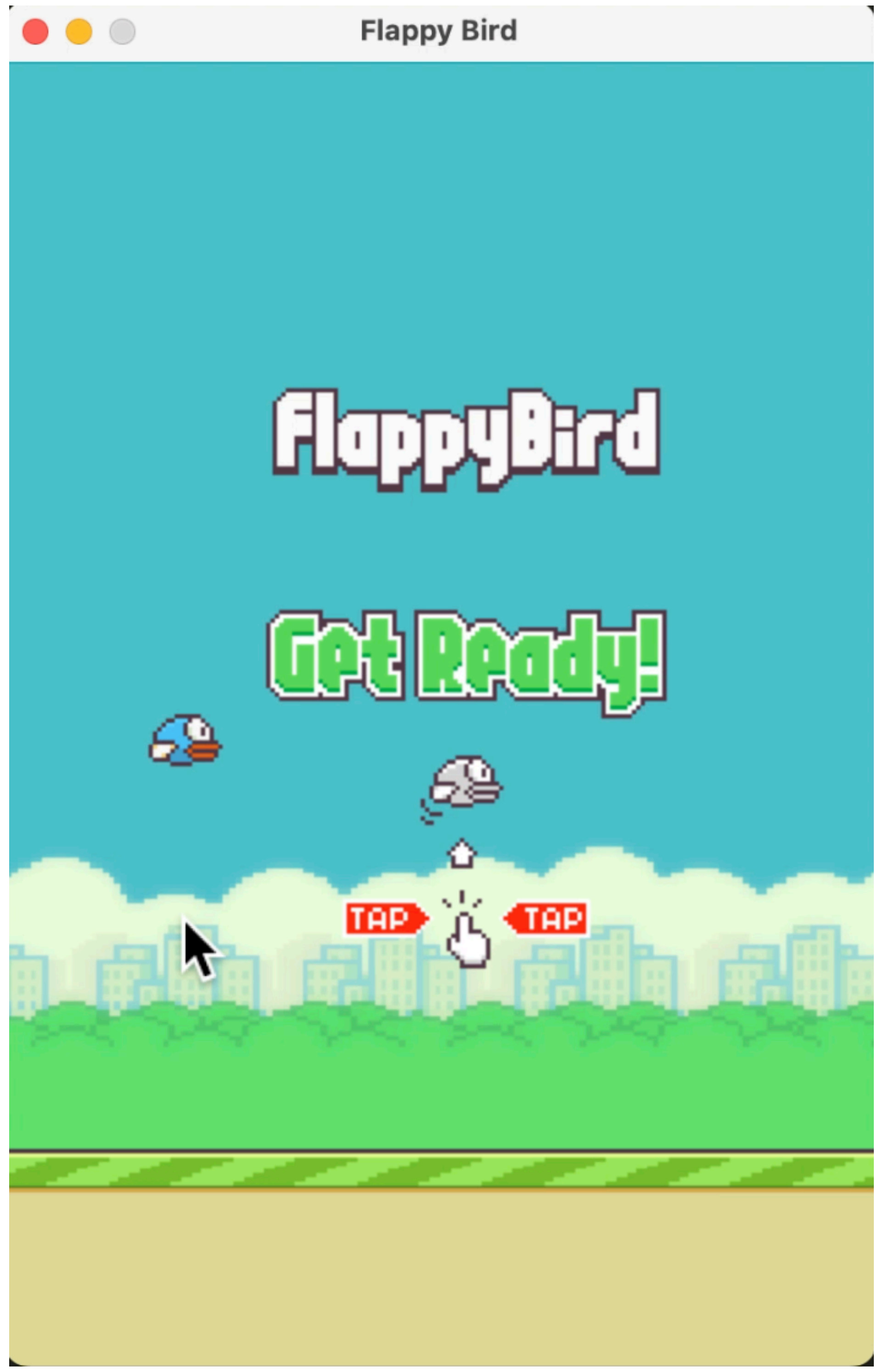
What is the goal/objective?

- Maximise score



Flappy Bird

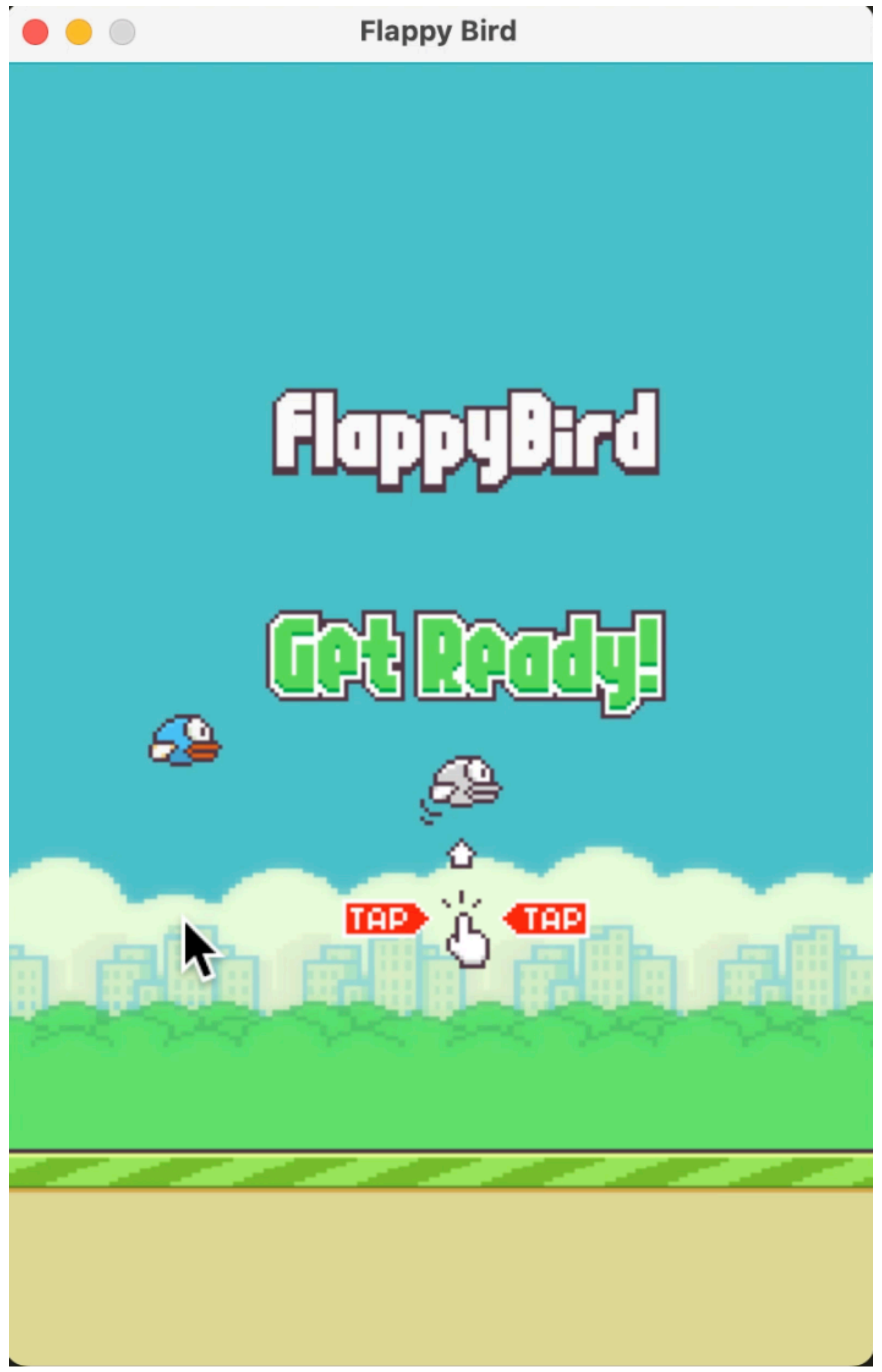
What are the actions we can take?



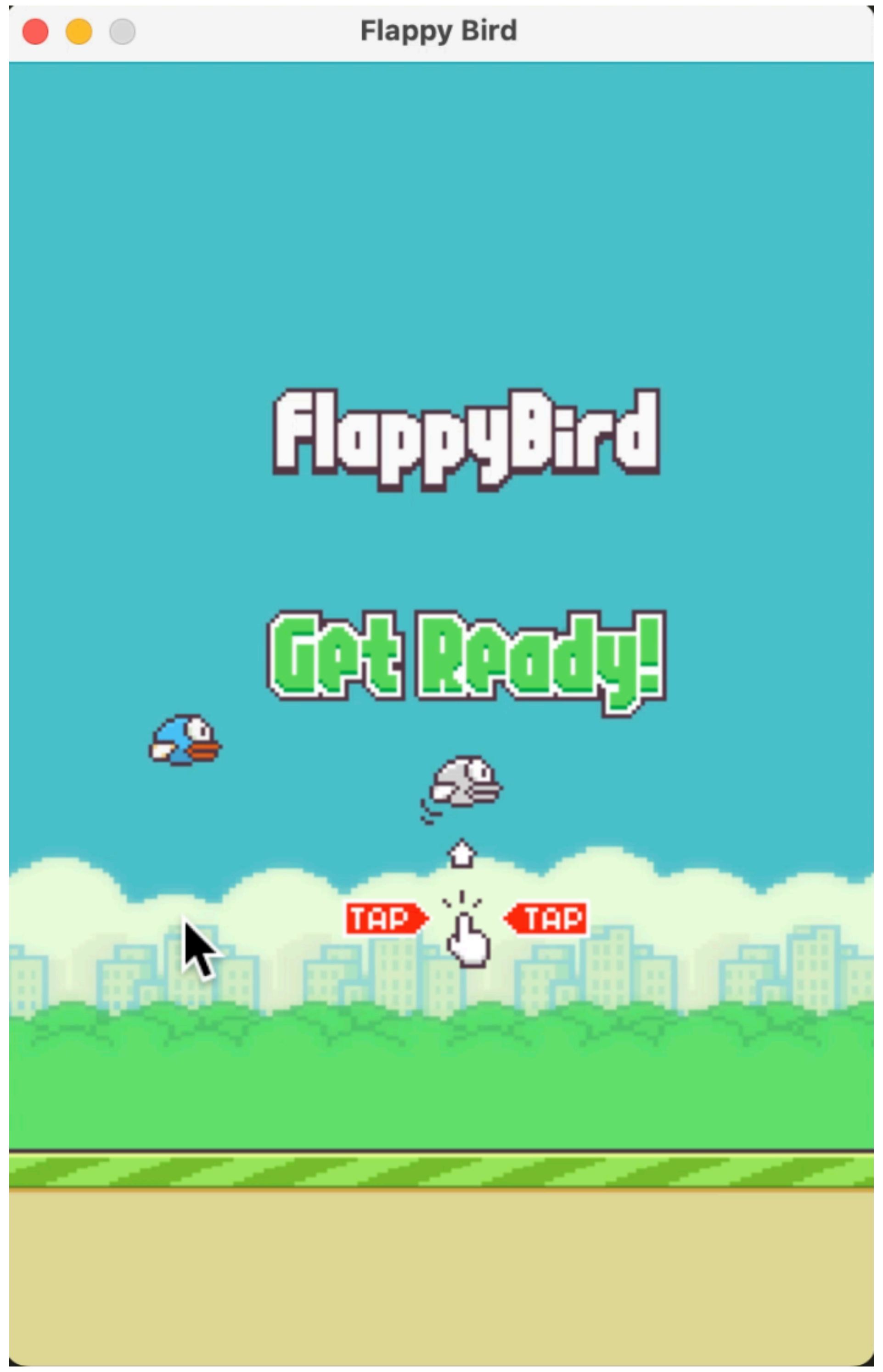
Flappy Bird

What are the actions we can take?

- Two actions
    - Tap (Space)
    - No tap



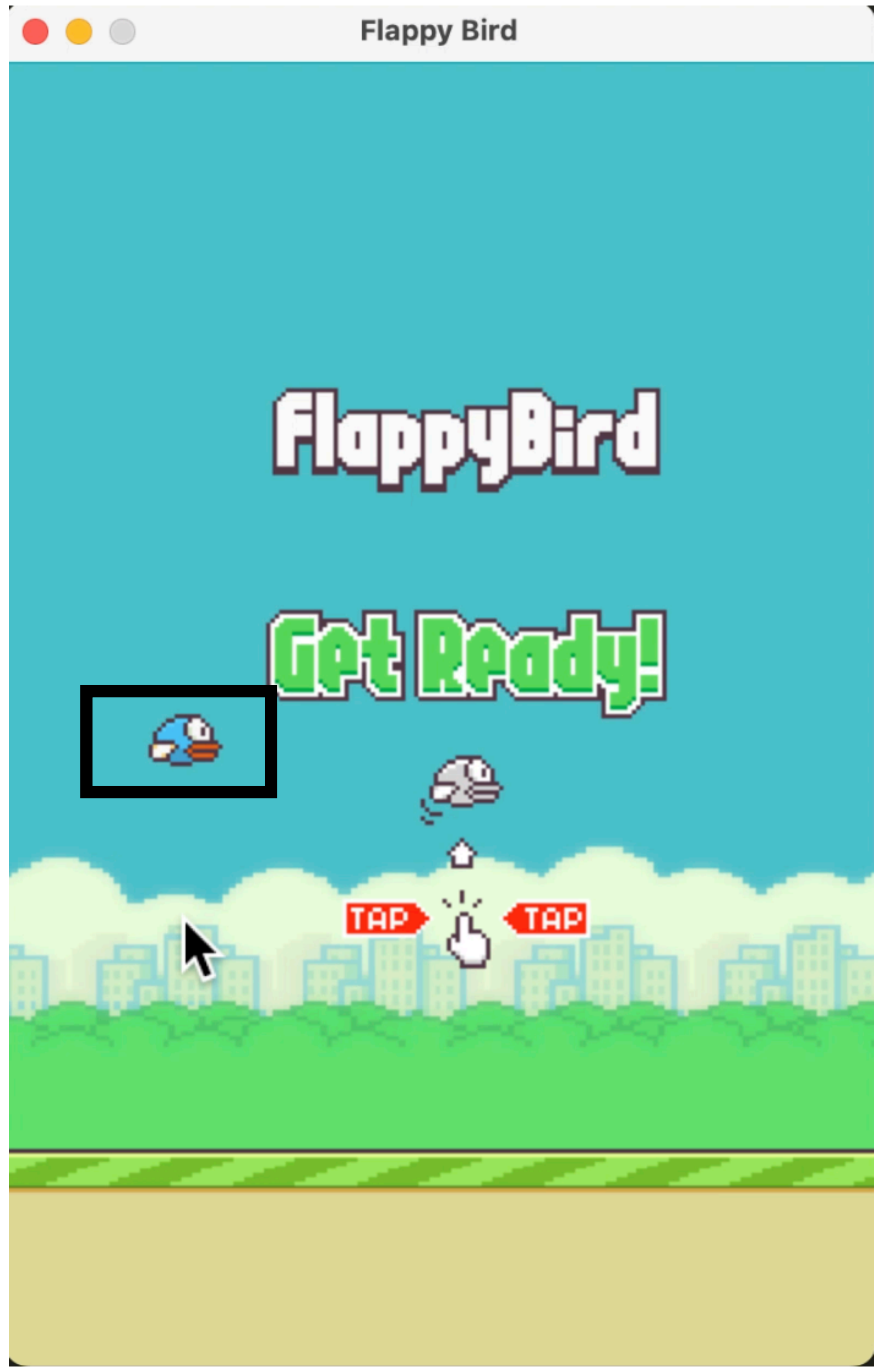
Flappy Bird
Who is playing?



Flappy Bird

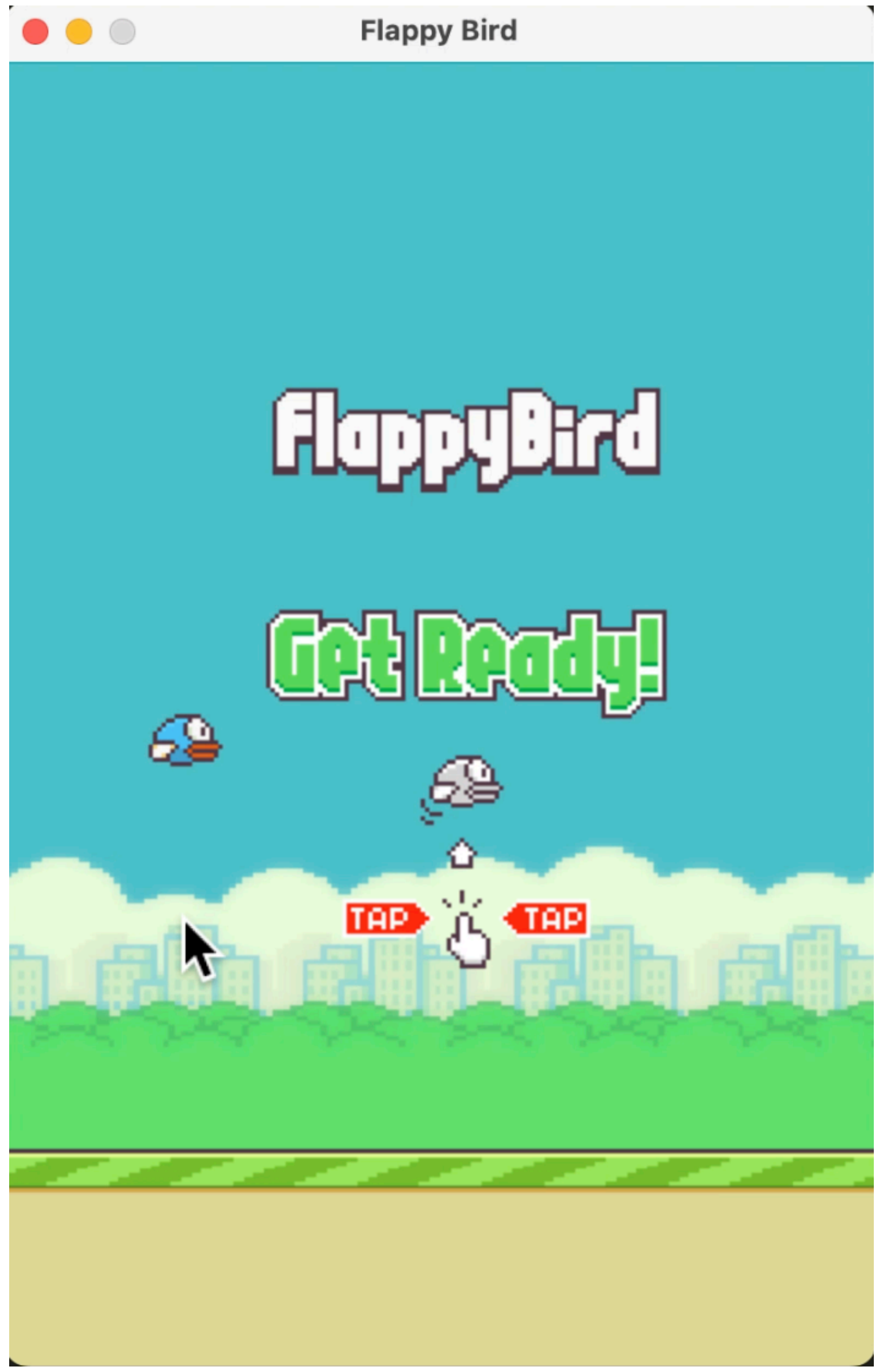
Who is playing?

- Agent
  - You
  - Or some algorithm



Flappy Bird
Where are we playing?

```python
while begin:

    clock.tick(15)

    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
        if event.type == KEYDOWN:
            if event.key == K_SPACE or event.key == K_UP:
                bird.bump()
                pygame.mixer.music.load(wing)
                pygame.mixer.music.play()
                begin = False

    screen.blit(BACKGROUND, (0, 0))
    screen.blit(BEGIN_IMAGE, (120, 150))

    if is_off_screen(ground_group.sprites()[0]):
        ground_group.remove(ground_group.sprites()[0])

        new_ground = Ground(GROUND_WIDHT - 20)
        ground_group.add(new_ground)

    bird.begin()
    ground_group.update()

    bird_group.draw(screen)
    ground_group.draw(screen)
```

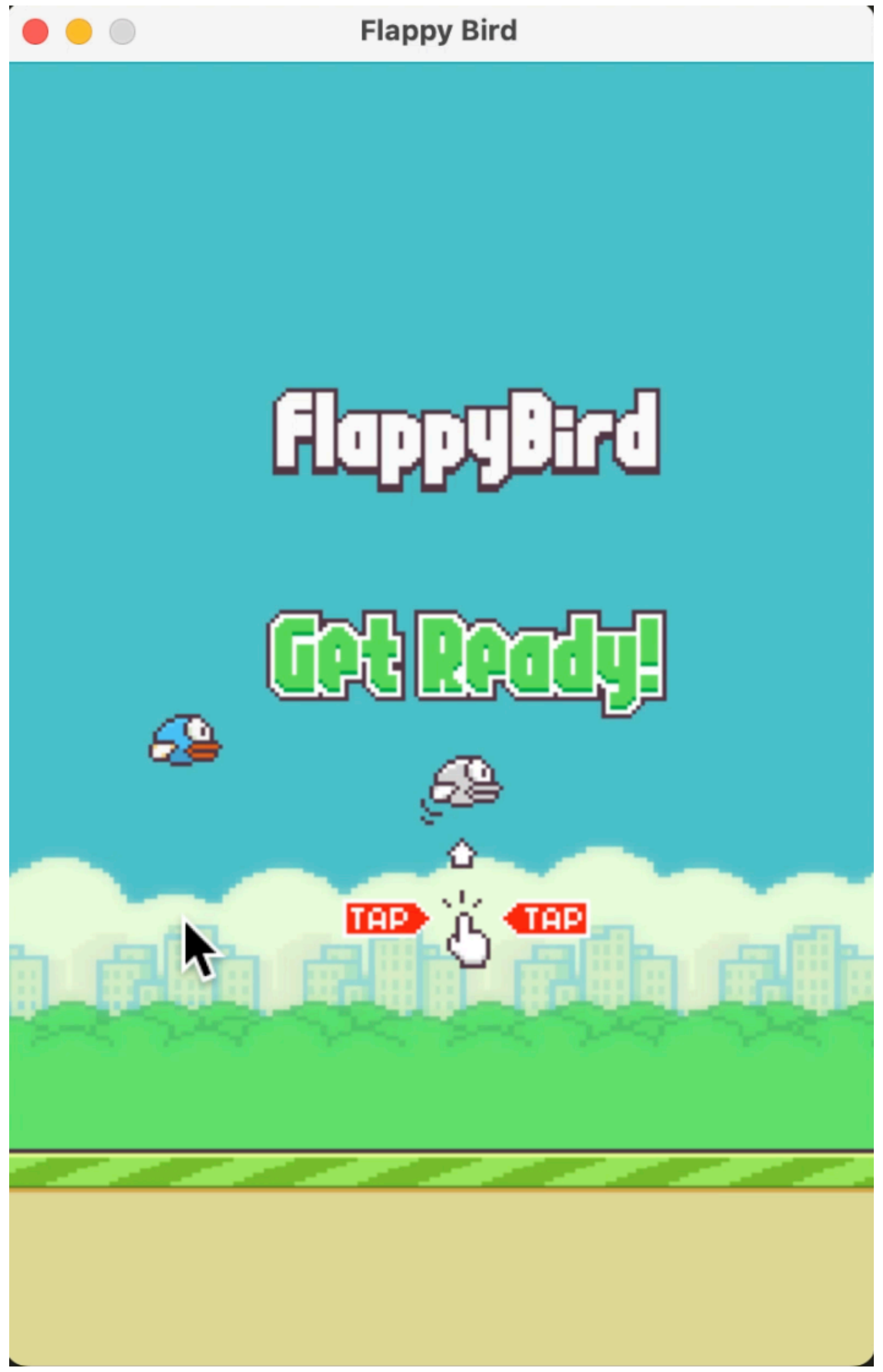


Flappy Bird

Where are we playing?

- Environment
  - Code
    - generating the graphics
    - Physics rules
      - What happens when you tap
      - What happens when you hit pipe

```
while begin:

    clock.tick(15)

    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
        if event.type == KEYDOWN:
            if event.key == K_SPACE or event.key == K_UP:
                bird.bump()
                pygame.mixer.music.load(wing)
                pygame.mixer.music.play()
                begin = False

    screen.blit(BACKGROUND, (0, 0))
    screen.blit(BEGIN_IMAGE, (120, 150))

    if is_off_screen(ground_group.sprites()[0]):
        ground_group.remove(ground_group.sprites()[0])

        new_ground = Ground(GROUND_WIDHT - 20)
        ground_group.add(new_ground)

    bird.begin()
    ground_group.update()

    bird_group.draw(screen)
    ground_group.draw(screen)
```



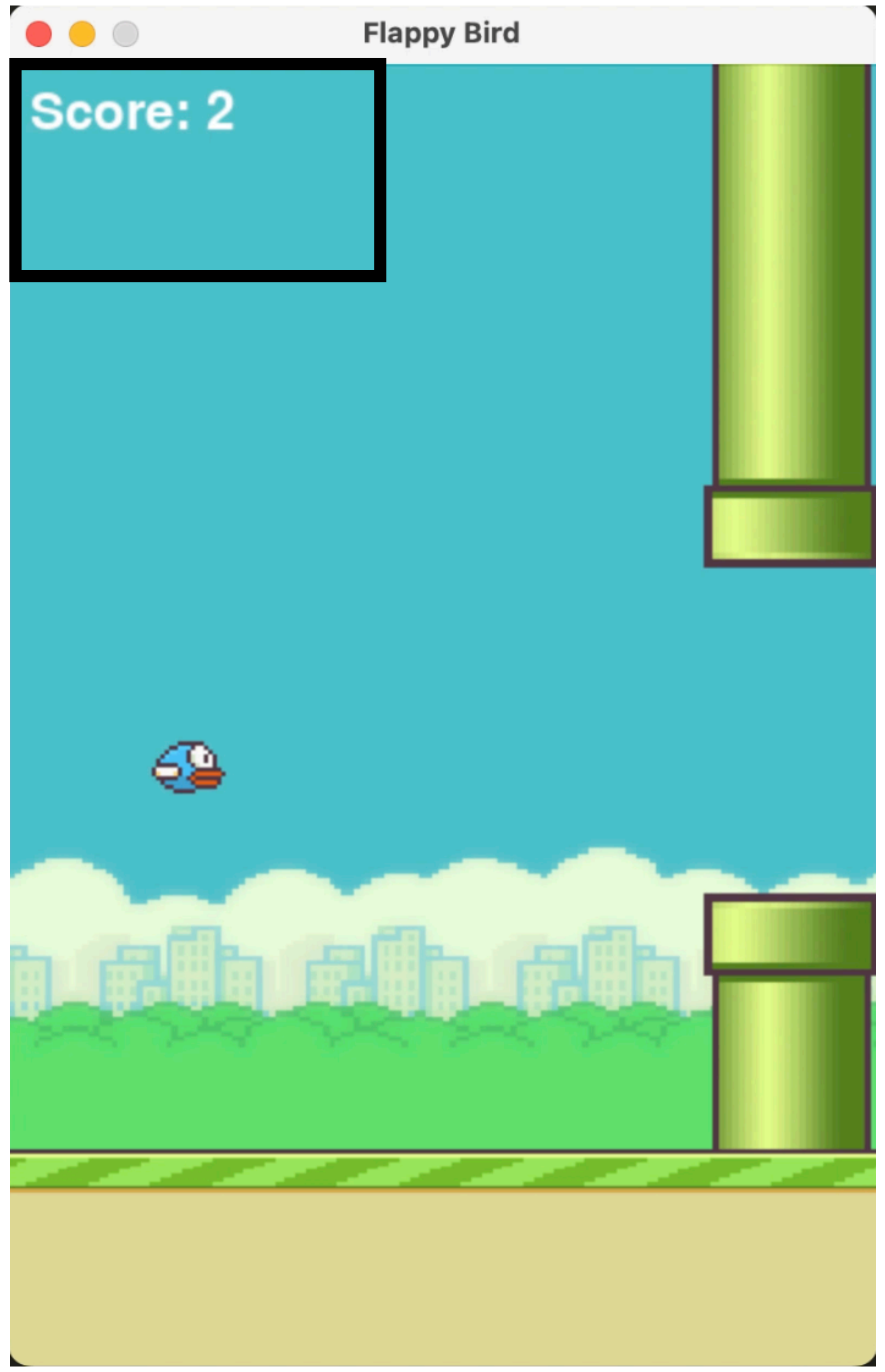
Flappy Bird

What does the environment provide to an agent?

Flappy Bird

What does the environment provide to an agent?

Reward

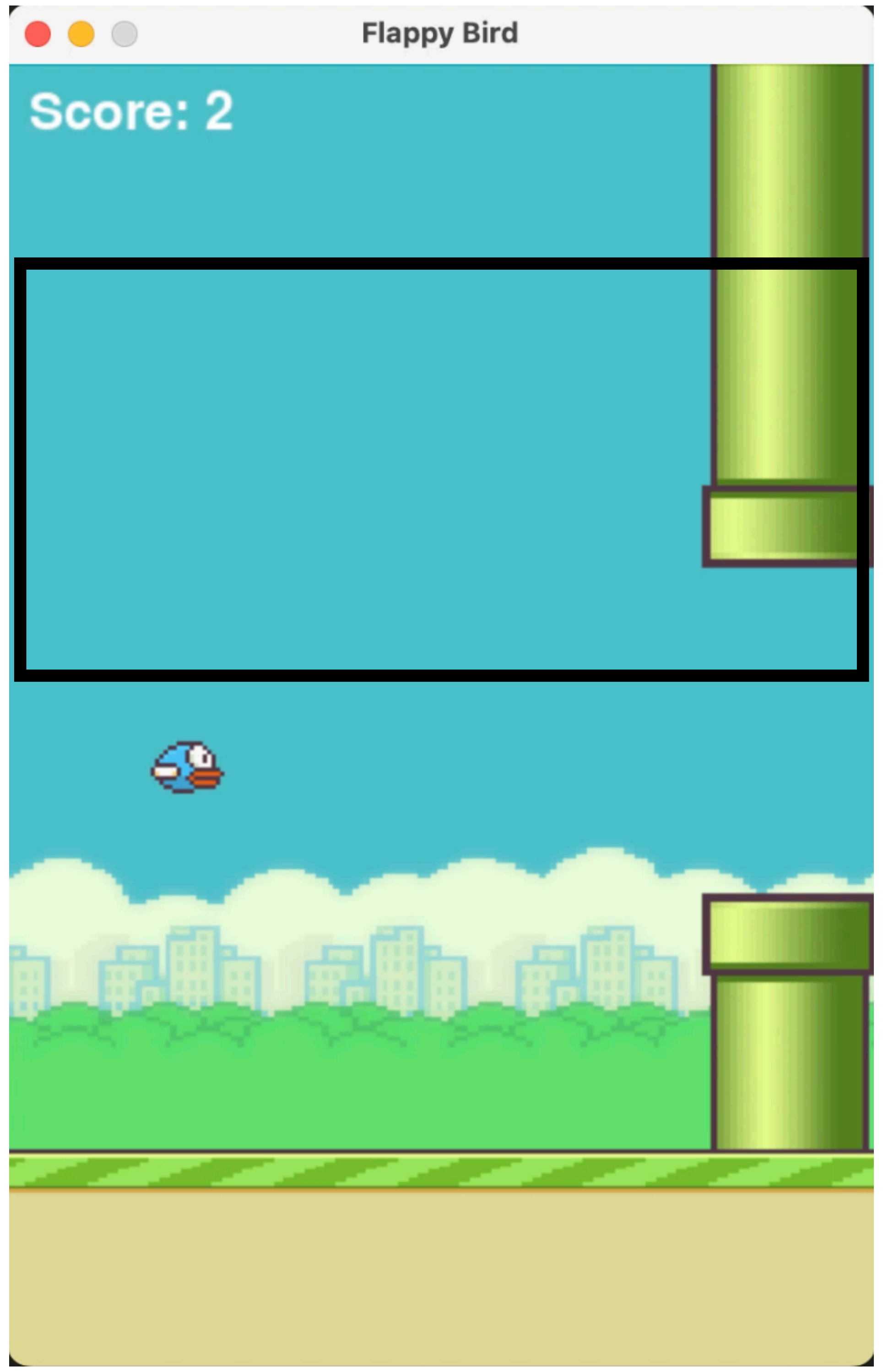


Flappy Bird

What does the environment provide to an agent?

Observations

Pixel level information

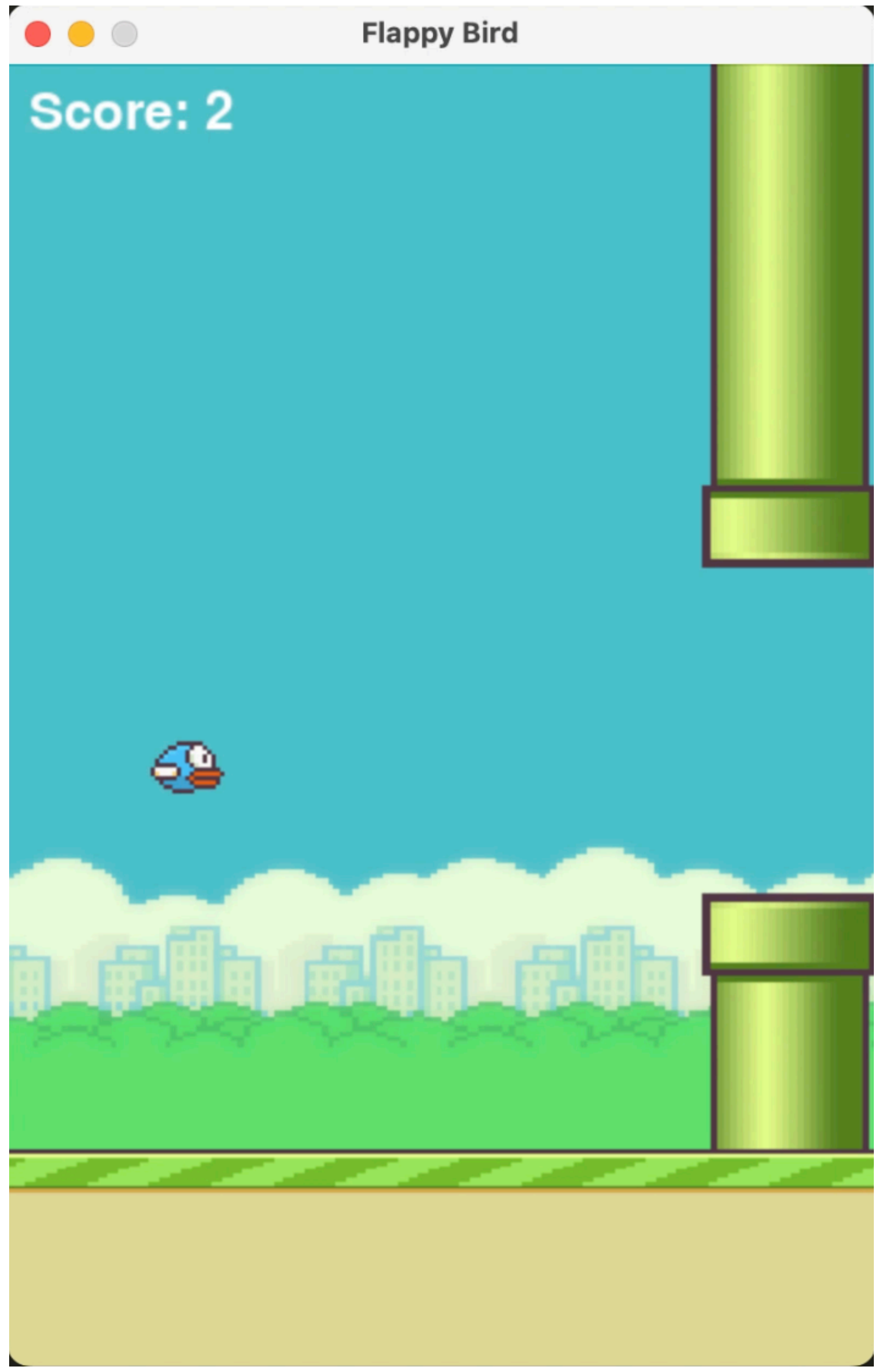


Flappy Bird

How does an agent decide what action to take?

Should the agent

tap or not?

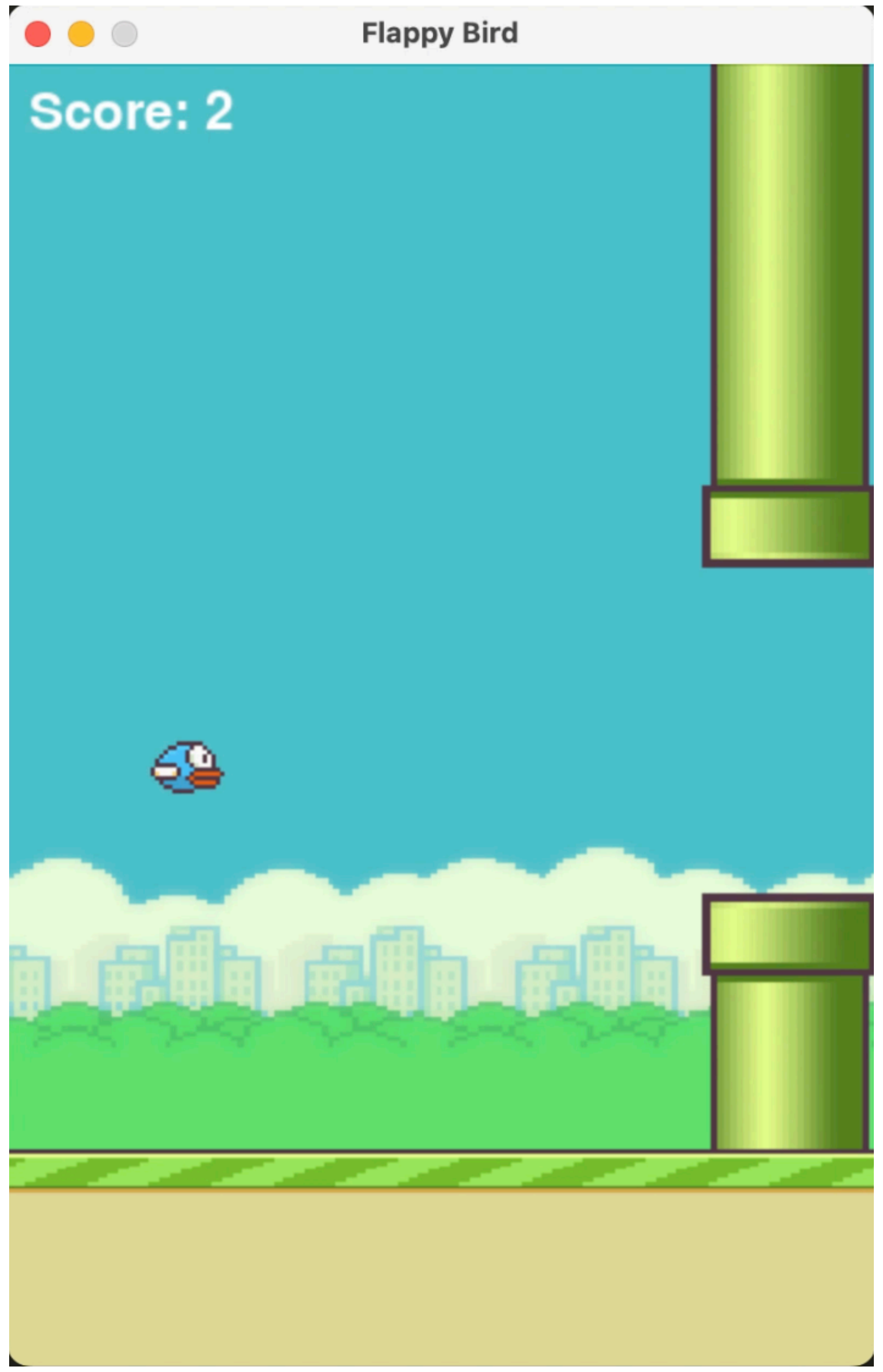


Flappy Bird

How does an agent decide what action to take?

State

Process observation into a "state"

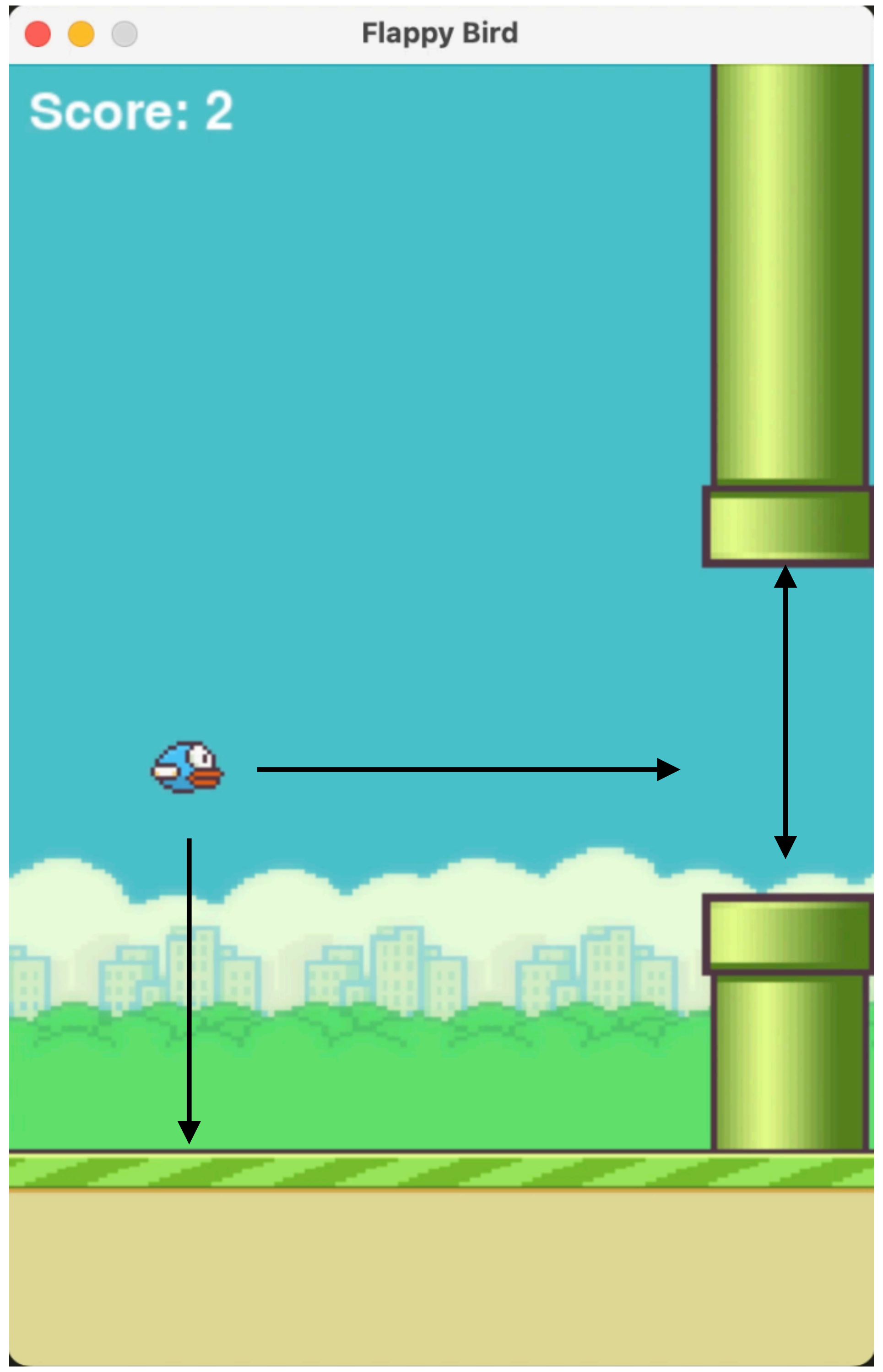


Flappy Bird

How does an agent decide what action to take?

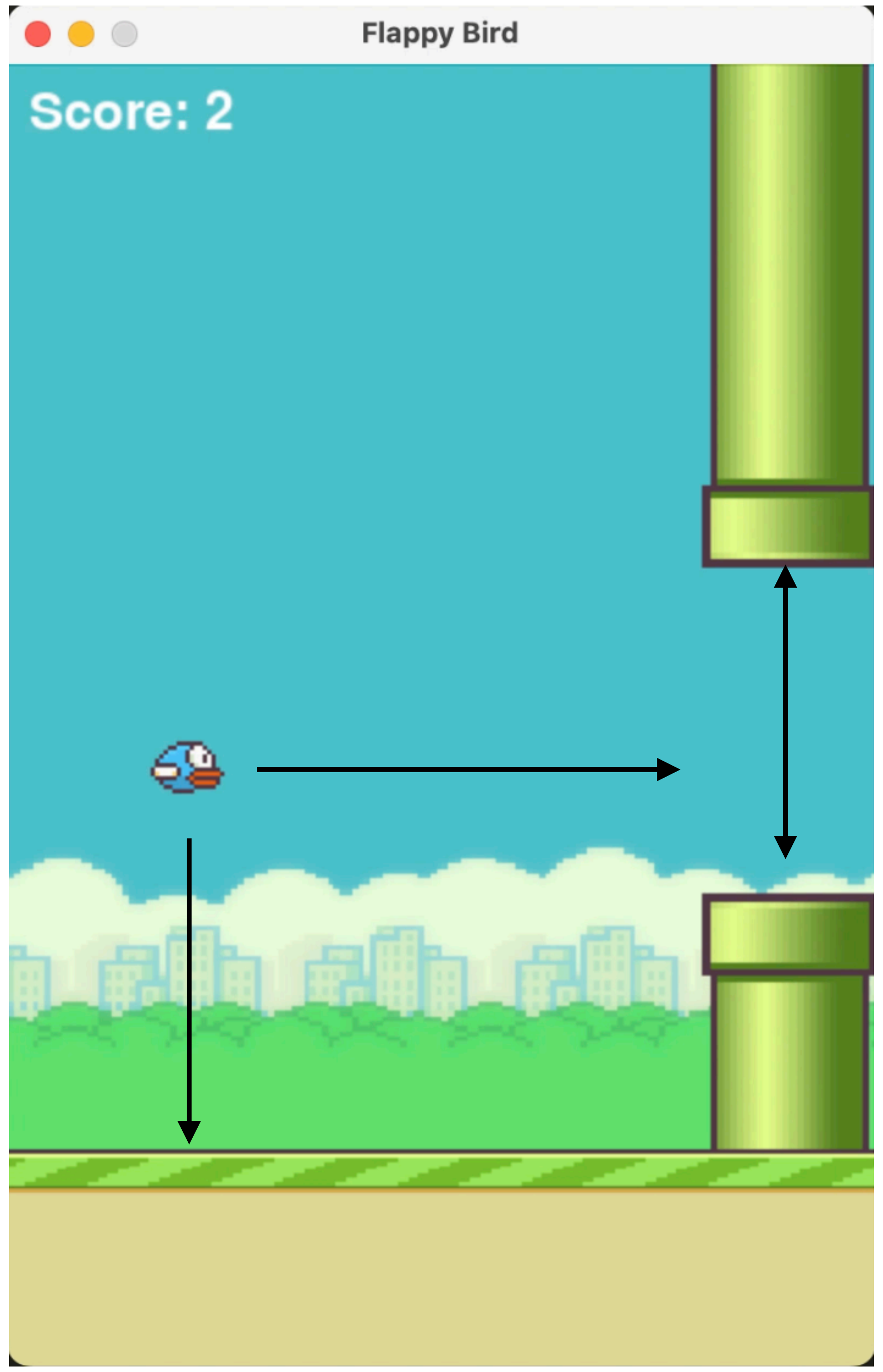
State

Process observation into a "state"



Flappy Bird

Is time important?

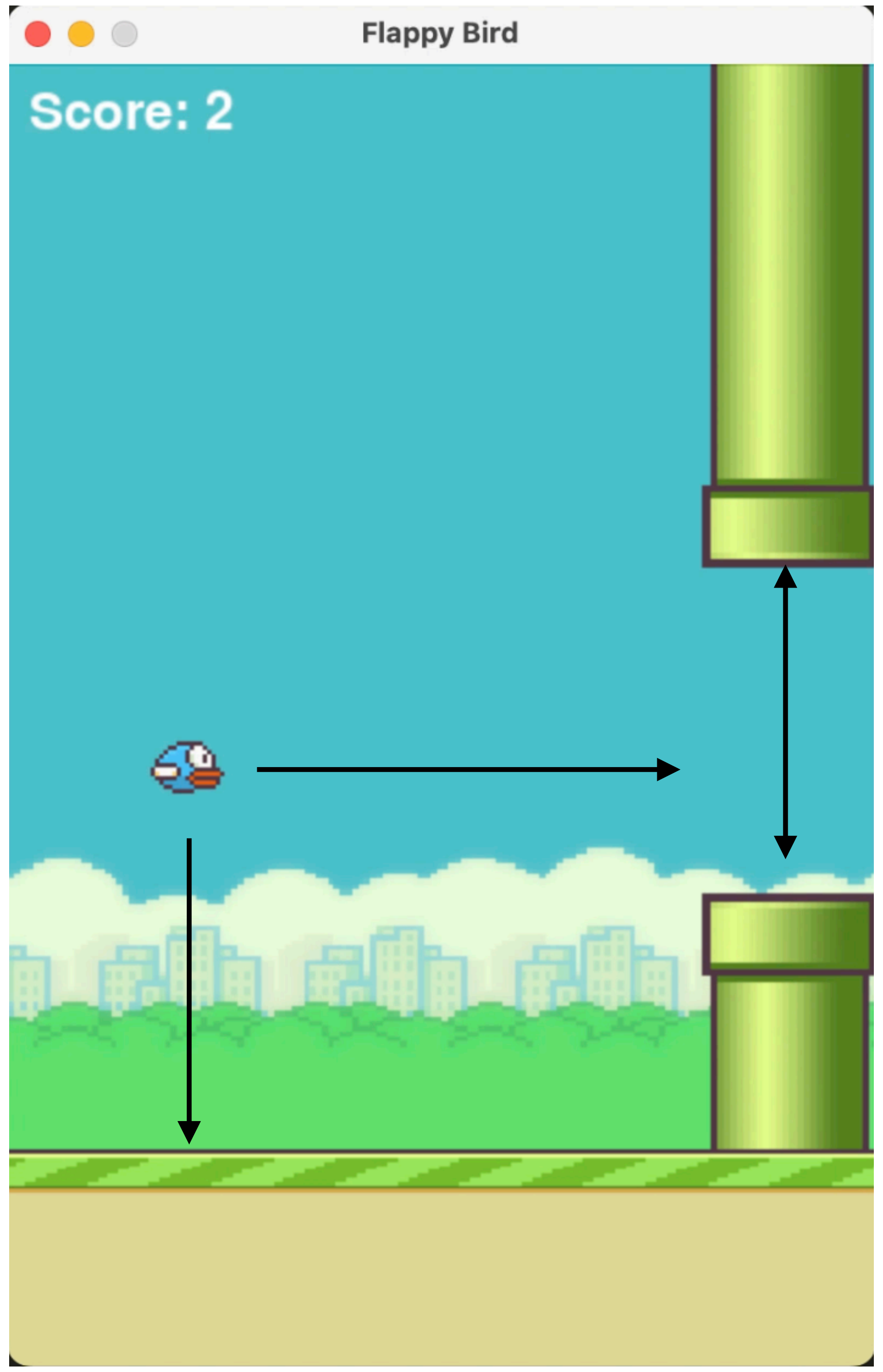


Flappy Bird

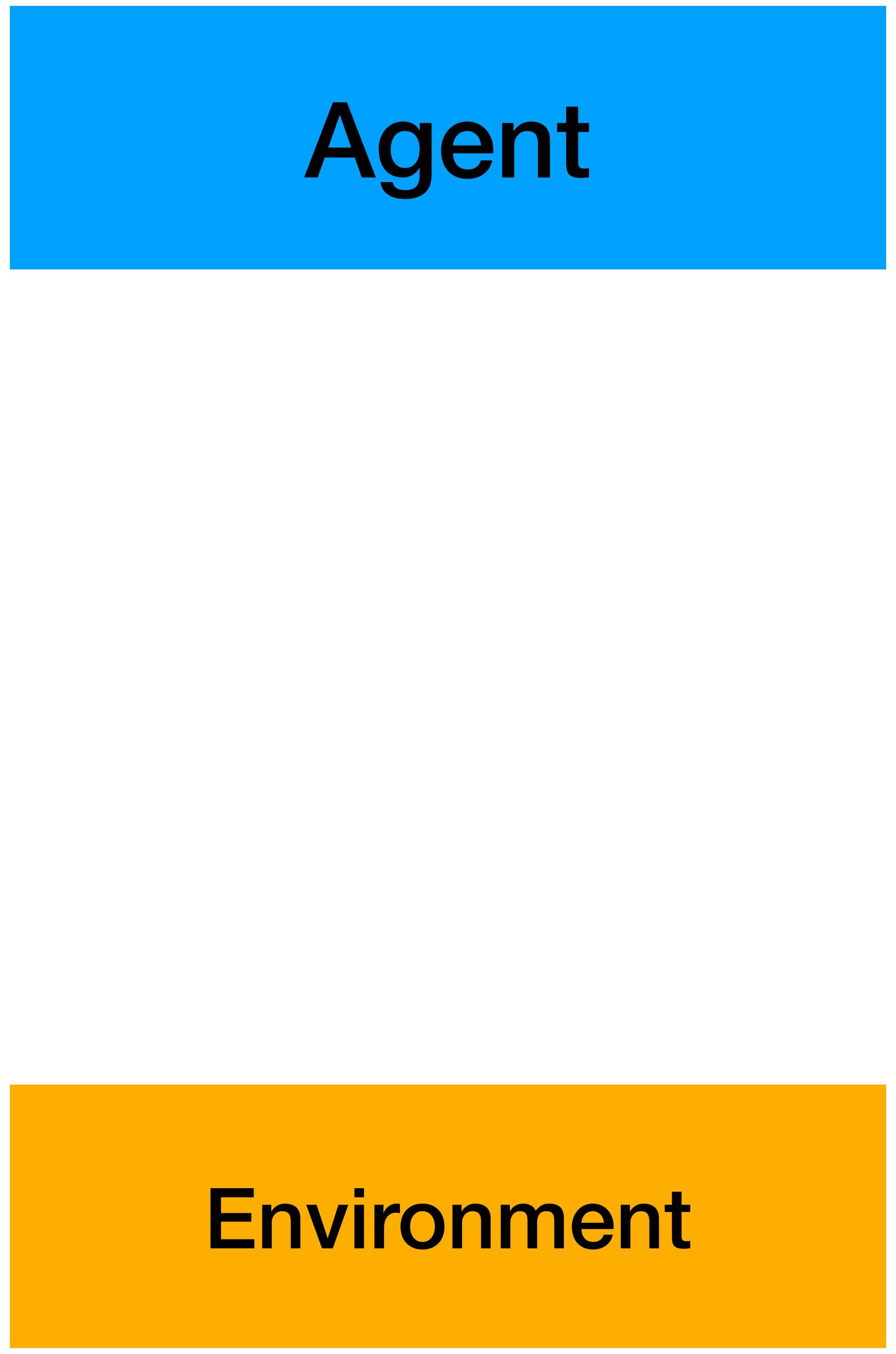
Is time important?

Yes

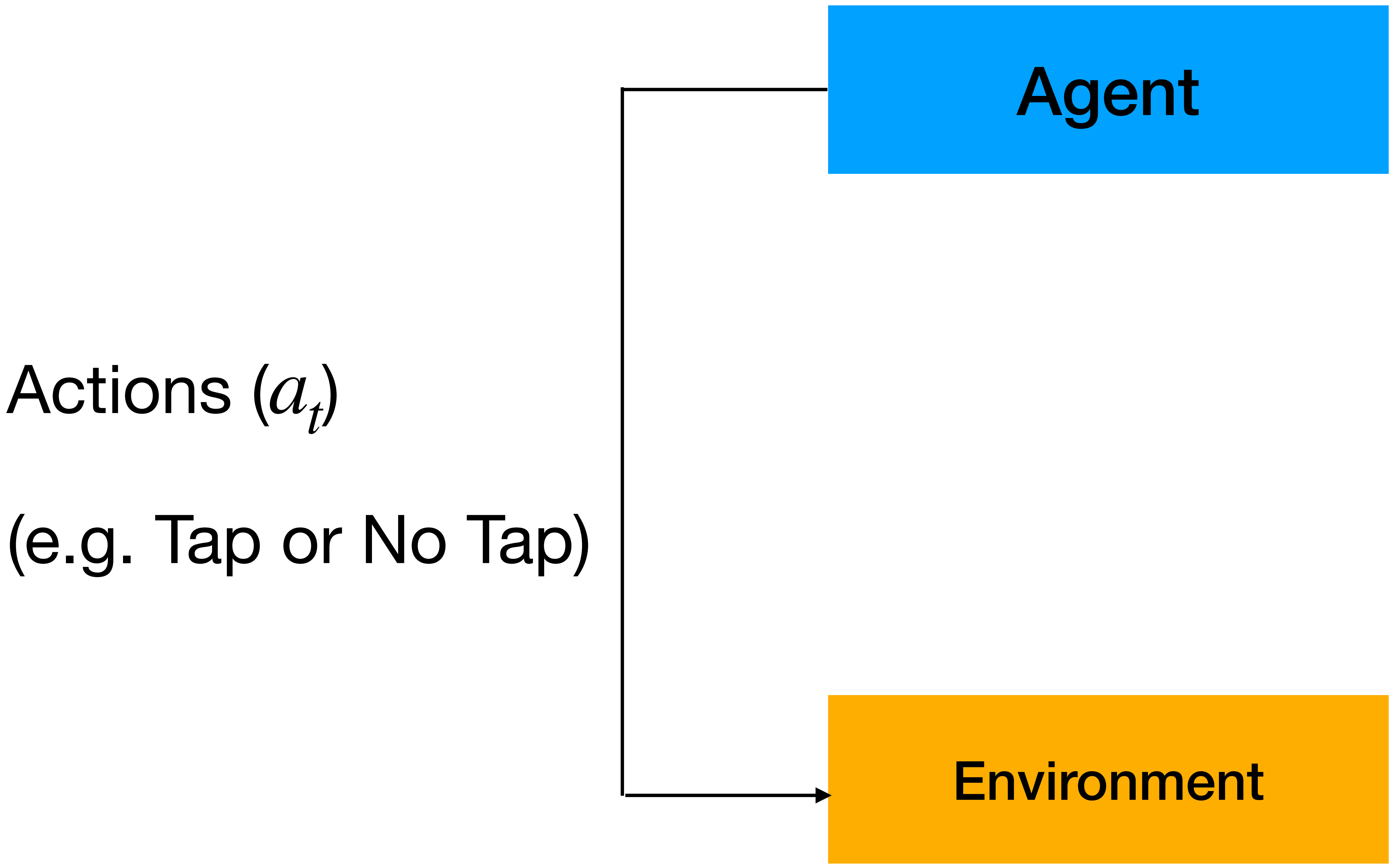
Agent's current state depends on previous state and action



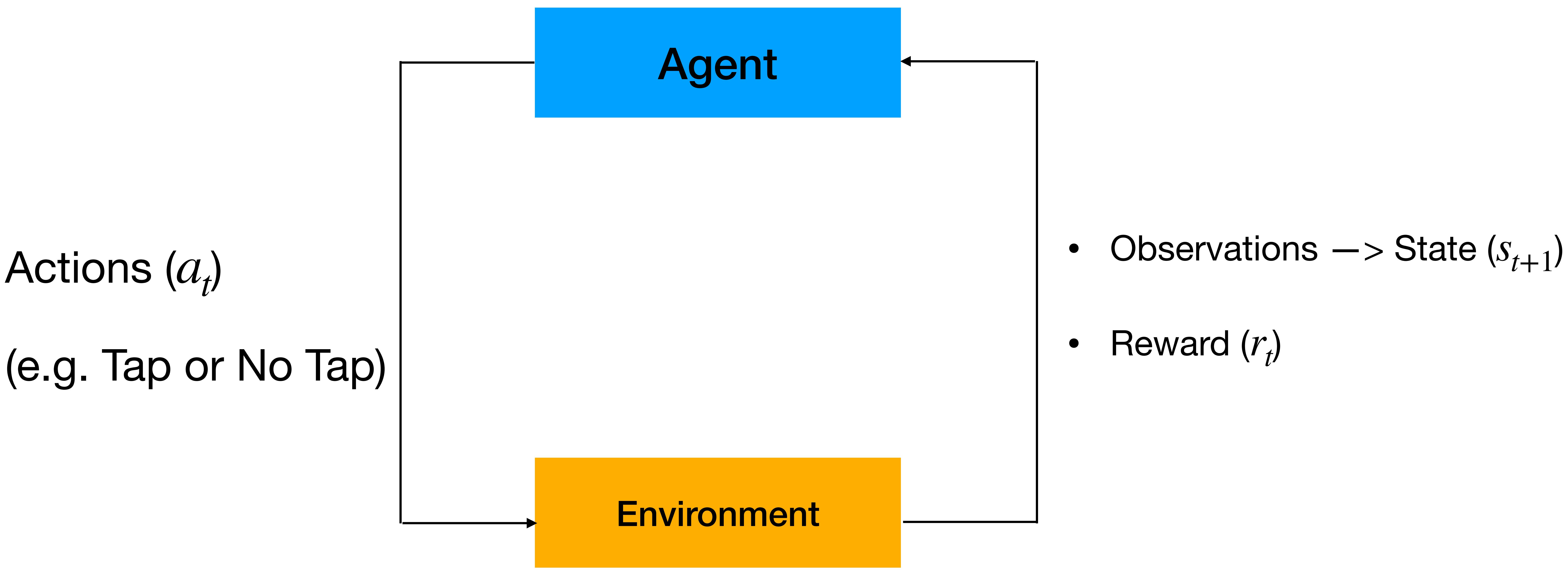
Block Diagram



Block Diagram



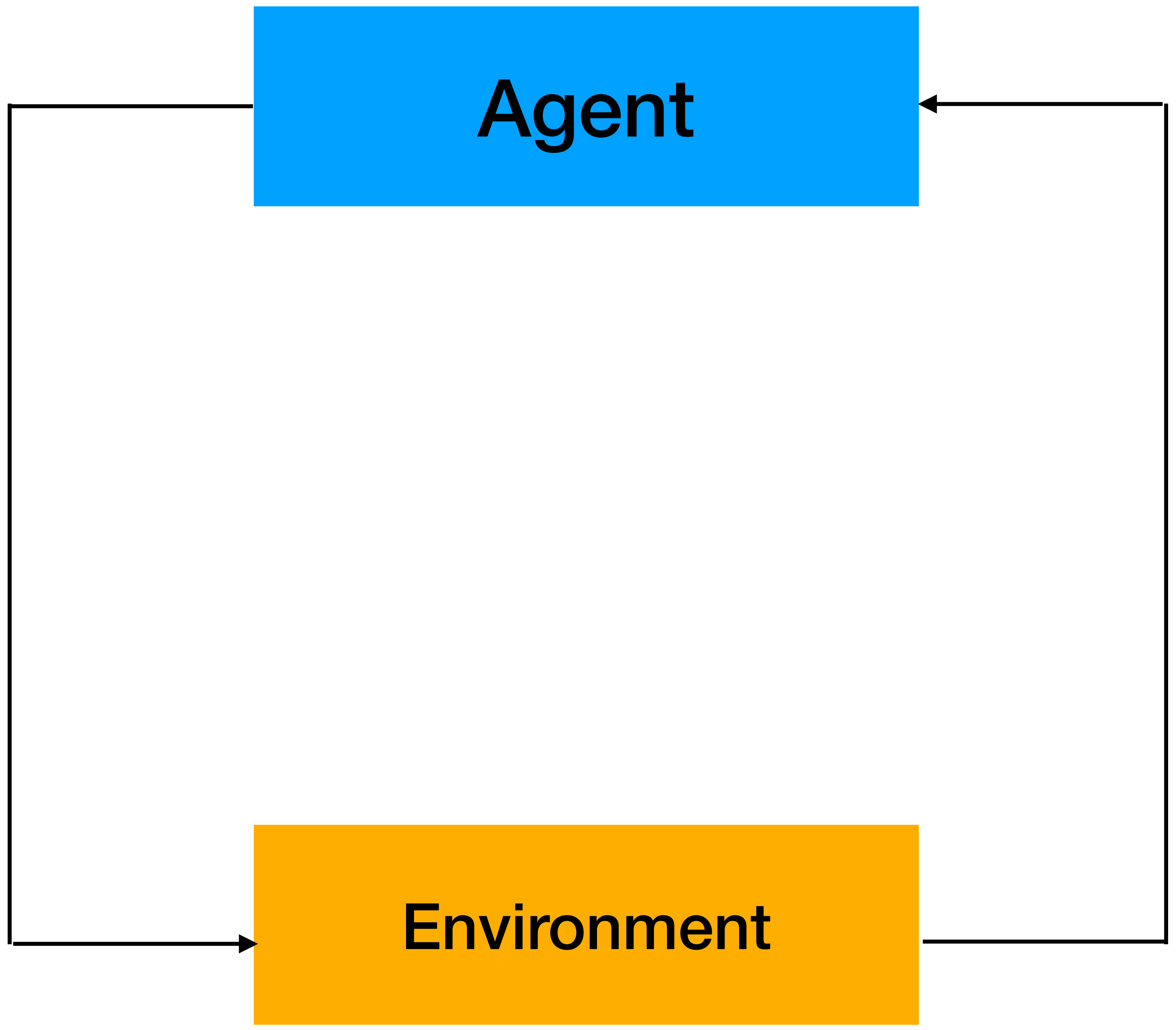
Block Diagram



OpenAI Gym Environment

- Mountain Car
  - Actions?
  - State?

Block Diagram



Actions (a_t)

(e.g. Tap or No Tap)

- Observations —> State (s_{t+1})
- Reward (r_t)

Goal: Maximise total (discounted) reward

- Total Reward (Return) $R_t = \sum_{i=t}^{\infty} r_i = r_t + r_{t+1} \ldots + r_{t+n} + \cdots$

- Total Reward (Discounted Return)

$$R_t = \sum_{i=t}^{\infty} \gamma^i r_i = \gamma^t r_t + \gamma^{t+1} r_{t+1} \ldots + \gamma^{t+n} r_{t+n} + \cdots$$

- γ : discount factor; $0 < \gamma < 1$

Q function

- What we want?
    - Given a state choose an "action" that maximises total discounted reward
- Total Reward (Discounted Return)

$$R_t = \sum_{i=t}^{\infty} \gamma^i r_i = \gamma^t r_t + \gamma^{t+1} r_{t+1} \ldots + \gamma^{t+n} r_{t+n} + \cdots$$

- $Q\left(s_t, a_t\right) = \mathbb{E}\left[R_t \mid s_t, a_t\right]$
- Q-function captures the expected total future reward an agent can achieve by taking an action.

Q table

State	Action 1	Action 2	Action 3
S1	10	20	15
S2	20	30	5
...			
SN	-5	10	20

Q table

State	Action 1	Action 2	Action 3
S1	10	20	15
S2	20	30	5
...			
SN	-5	10	20

What action will you choose if you are in state S2?

Q table

State	Action 1	Action 2	Action 3
S1	10	20	15
S2	20	**30**	5
...			
SN	-5	10	20

What action will you choose if you are in state S2?

Action 2 (as it gives us highest return)

Q table

State <Position, Velocity>	Action 1	Action 2	Action 3
<-5, -2>	?	?	?
…	?	?	?
…			
…	?	?	?

How do we define states for problems like Mountain car where these numbers are not discrete?

Q table

State <Position, Velocity>	Action 1	Action 2	Action 3
<[-5, -4], [-2, -1]>	?	?	?
<[-5, -4], [—1, 0]>	?	?	?
...			
...	?	?	?

How do we define states for problems like Mountain car where these numbers are not discrete?

Discretisation (notebook)

Q function (revision)

- What we want?
  - Given a state choose an “action” that maximises total discounted reward
- Total Reward (Discounted Return)

$$R_t = \sum_{i=t}^{\infty} \gamma^i r_i = \gamma^t r_t + \gamma^{t+1} r_{t+1} \ldots + \gamma^{t+n} r_{t+n} + \cdots$$

- $Q\left(s_t, a_t\right) = \mathbb{E}\left[R_t \mid s_t, a_t\right]$
- Q-function captures the expected total future reward an agent can achieve by taking an action.

Bellman Equation

The Bellman equation for Q-values is given by:

$$Q(s,a) = R(s,a) + \gamma \cdot \max_{a'} Q\left(s', a'\right)$$

where:

- $Q(s,a)$ is the Q-value of taking action
- $R(s,a)$ is the immediate reward of taking action a in state s.
- γ is the discount factor that determines the importance of future rewards.
- s' is the next state after taking action a.
- $\max_{a'} Q\left(s', a'\right)$ is the maximum Q-value over all possible actions in state s'.

Q-learning Update Bellman Equation

$$Q(s,a) = R(s,a) + \gamma \cdot \max_{a'} Q\left(s', a'\right)$$

Q-learning update rule is derived by using the Bellman equation in an iterative manner:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \left(R(s,a) + \gamma \cdot \max_{a'} Q\left(s', a'\right) - Q(s,a) \right)$$

- α is the learning rate that controls the extent to which new information overrides old information.

- $R(s,a) + \gamma \cdot \max_{a'} Q\left(s', a'\right) - Q(s,a)$ is the temporal difference (TD) error, representing the discrepancy between the expected Q-value and the observed reward.