Final Project of DRL

PPO Snake AI

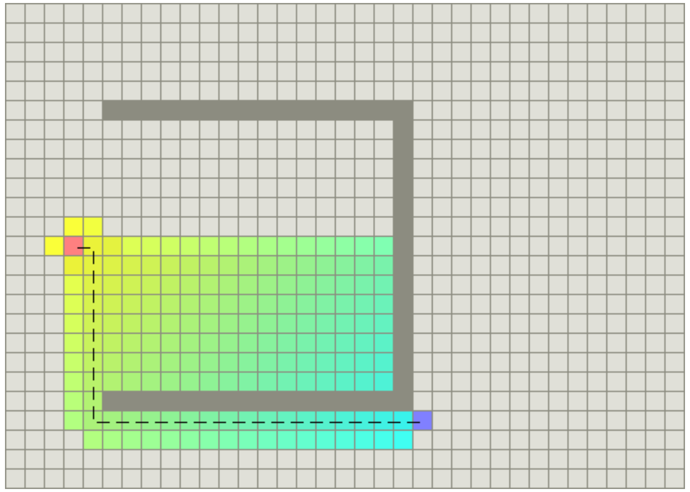
Report

Background

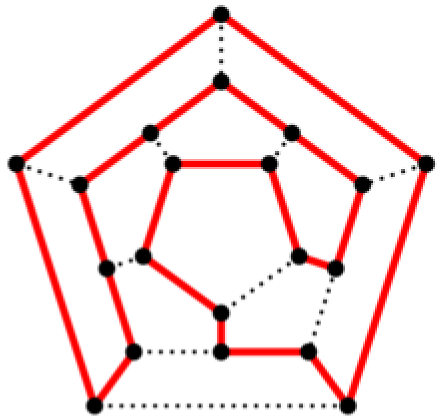
- An arcade game called Blockade in 1976
- Non-Reinforcement Learning Algorithms for Snake AI

A-star

Hamilton circuit



A-star

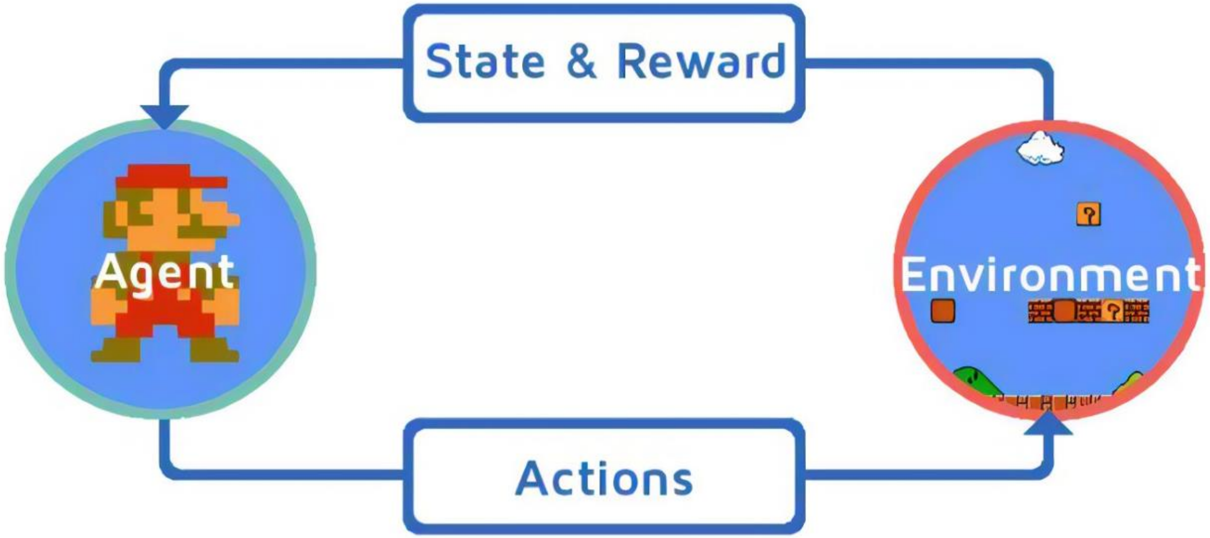


Hamilton circuit

Problem

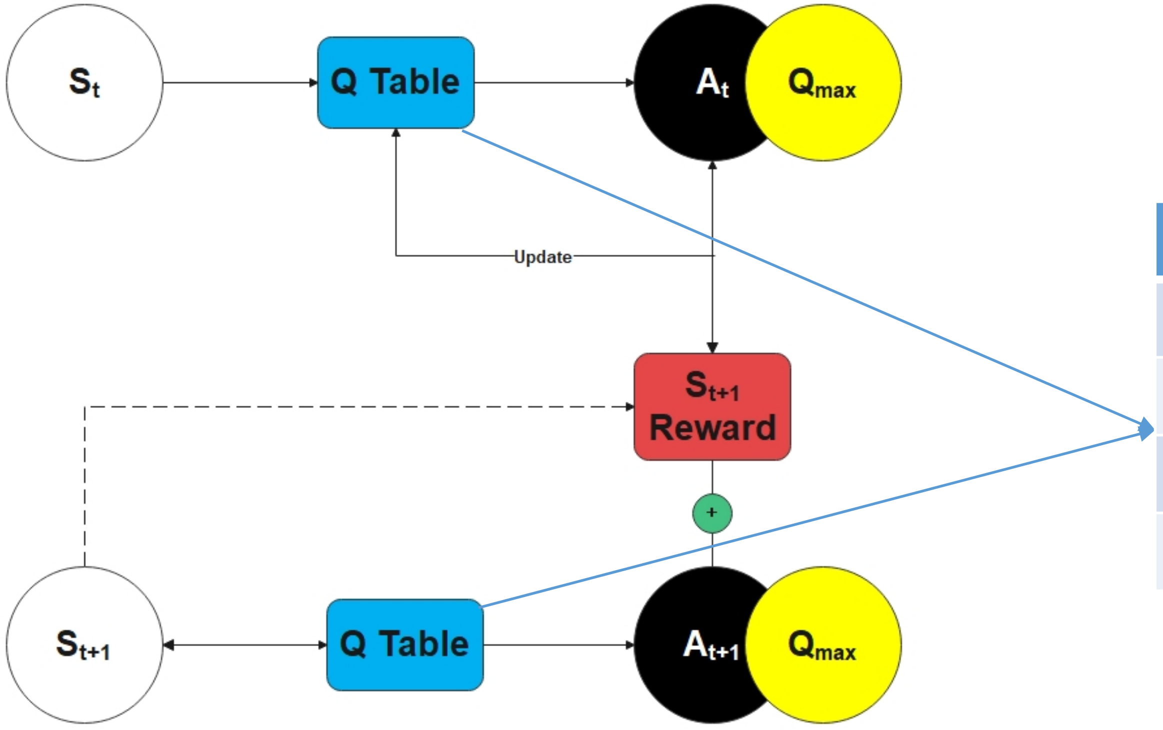
- Use Deep Reinforcement Learning Algorithms to play the Snake game

- Agent: The snake with AI
- Environment: Game world
- State: Snake status
- Reward: Scores by eat, win or lose
- Actions: What to do at next step



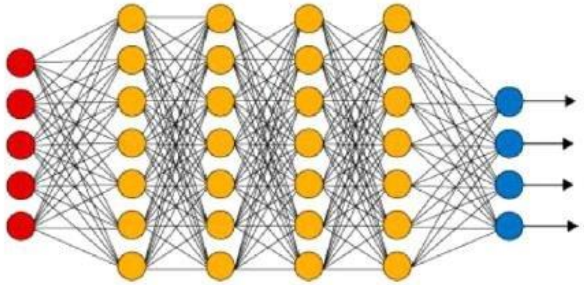
Method

- DQN



A=F(s)

...	A1	A2	A3...
...	0	0	0
S_{t-1}	0	0	0
S_t	0	0	2
...	0	0	0



Method

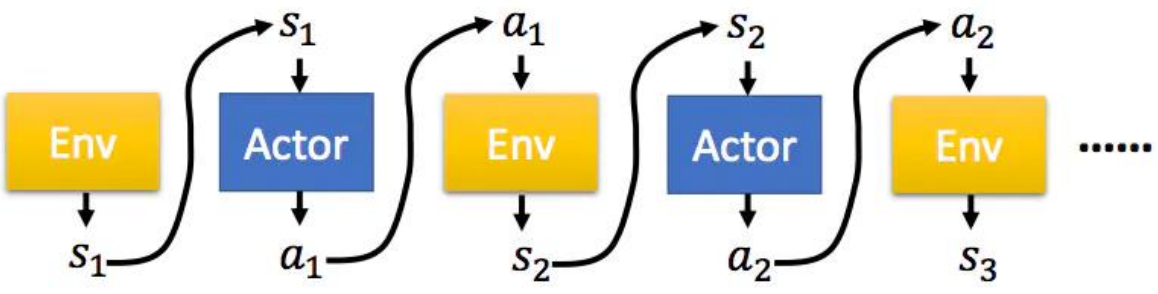
- PPO

Proximal Policy Optimization

On-policy

An improved algorithm for PG(Policy Gradient)

PG

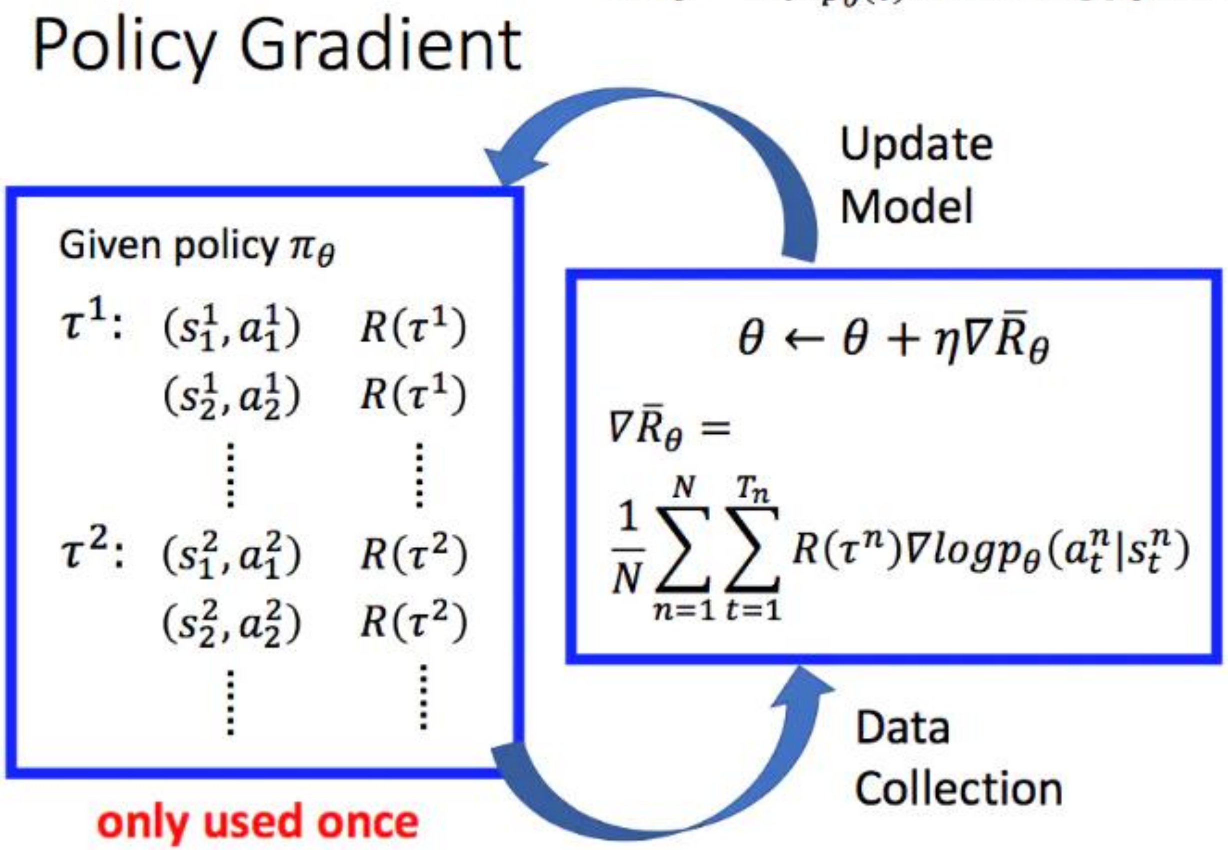


Trajectory $\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

$$\begin{aligned} p_\theta(\tau) &= p(s_1)p_\theta(a_1|s_1)p(s_2|s_1,a_1)p_\theta(a_2|s_2)p(s_3|s_2,a_2)\cdots \\ &= p(s_1)\prod_{t=1}^{T} p_\theta(a_t|s_t)p(s_{t+1}|s_t,a_t) \end{aligned}$$

$$\bar{R}_\theta = \sum_{\tau} R(\tau)p_\theta(\tau) = E_{\tau\sim p_\theta(\tau)}[R(\tau)]$$

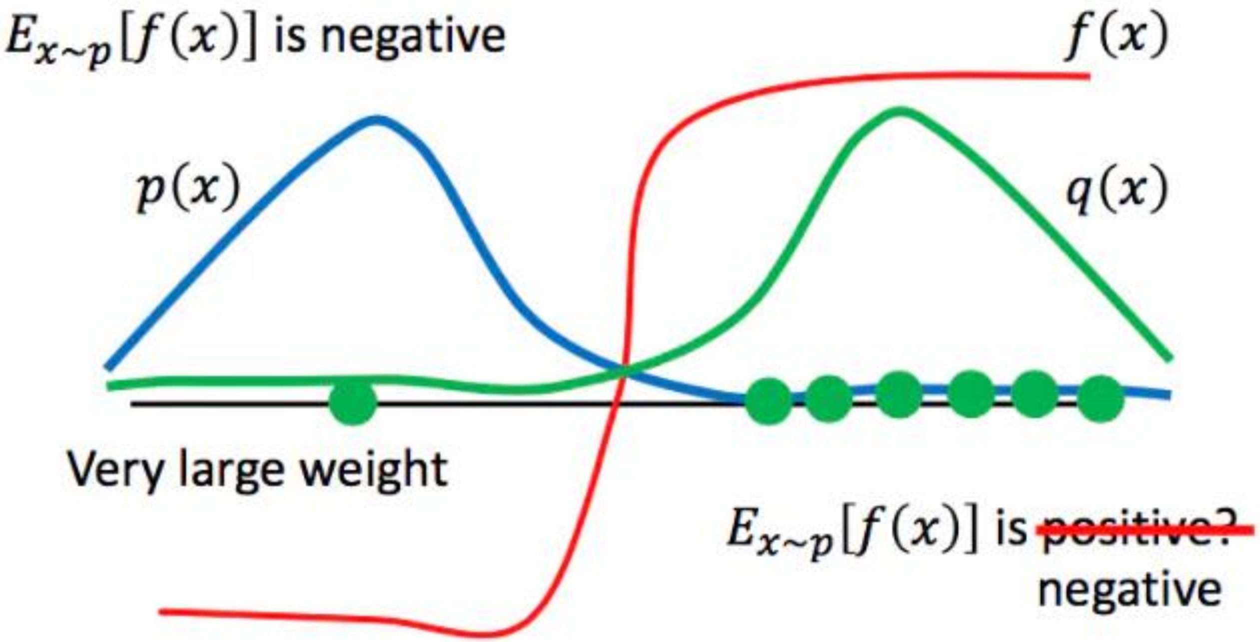
$$\nabla\bar{R}_\theta = E_{\tau\sim p_\theta(\tau)}[R(\tau)\nabla \log p_\theta(\tau)]$$



PG to PPO

$$E_{x\sim p}[f(x)] = E_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$

- Improve training speed
- The sample data can be reused



$$\nabla \bar{R}_\theta = E_{\tau\sim p_\theta(\tau)}[R(\tau)\nabla log p_\theta(\tau)] \longrightarrow \nabla \bar{R}_\theta = E_{\underline{\tau\sim p_{\theta'}(\tau)}}\left[\frac{p_\theta(\tau)}{\underline{p_{\theta'}(\tau)}} R(\tau)\nabla log p_\theta(\tau)\right]$$

PPO

- The two distributions should not be too distant or it will take a lot of sampling to get approximate results

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t)$$

- Use KL penalty to constrain the distance between the two distributions

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta,\theta^k)$$

If $KL(\theta,\theta^k) > KL_{max}$, increase β
If $KL(\theta,\theta^k) < KL_{min}$, decrease β

Experiments

State Table

State name	Description	Data type
xhead	x coordinate of snake head	int
yhead	y coordinate of snake head	int
snake_coords	Coordinate list of snake body	[{‘x’:int,’y’:int},...]
direction	Snake head direction	RIGHT/LEFT/UP/DOWN
xfood	X coordinate of food	int
yfood	Y coordinate of food	int
deltax	(xfood - xhead) / map_width	float
deltay	(yfood - yhead) / map_height	float

Action List

Action	Description	Real value
UP	Snake goes up if direction ≠ down	0
DOWN	Snake goes down if direction ≠ up	1
LEFT	Snake goes left if direction ≠ right	2
RIGHT	Snake goes right if direction ≠ left	3

Environment

PyGame

pip install pygame

import pygame

Params

Param	Value	Range
max_episode	800	Agent training
act_dim	4	Agent network
obs_dim	6	Agent network
net_dim	512	Agent network
batch_size	256	Agent training
soft_update_tau	2e-8	Agent network
target_step	2e12	Environment
gamma	0.99	Environment
reward_scale	1	Environment
std_actor	1.0	Action output
std_critic	0.5	Q-value output
bias	1e-6	Action & Q-value output

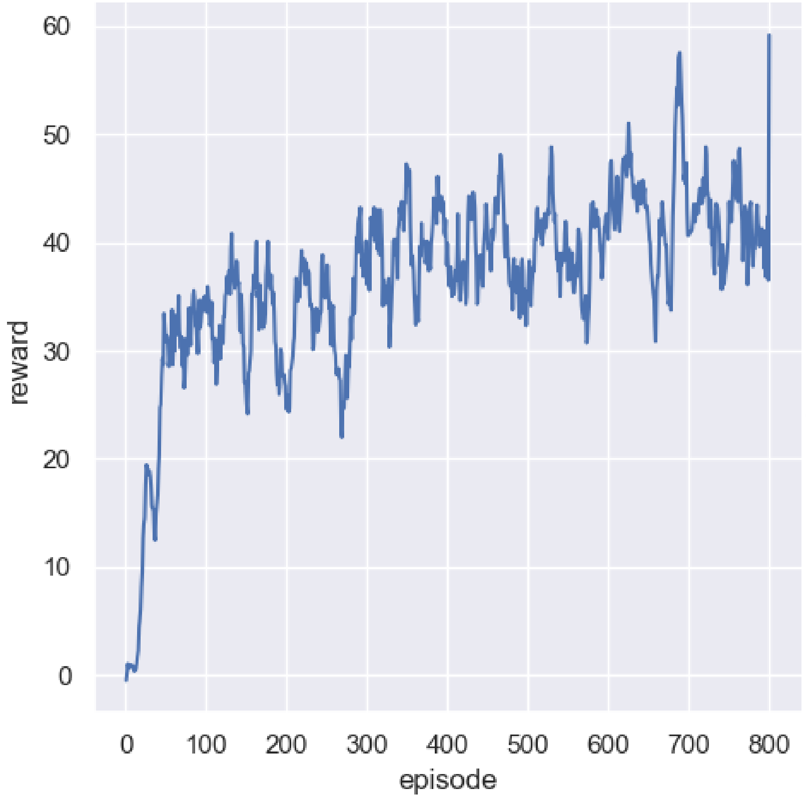
Rewards

Round	1	2	3
Reward_eat	+2.0	+2.0	+2.0
Reward_hit	-0.5	-1.0	-1.5
Reward_bit	-0.8	-1.5	-2.0

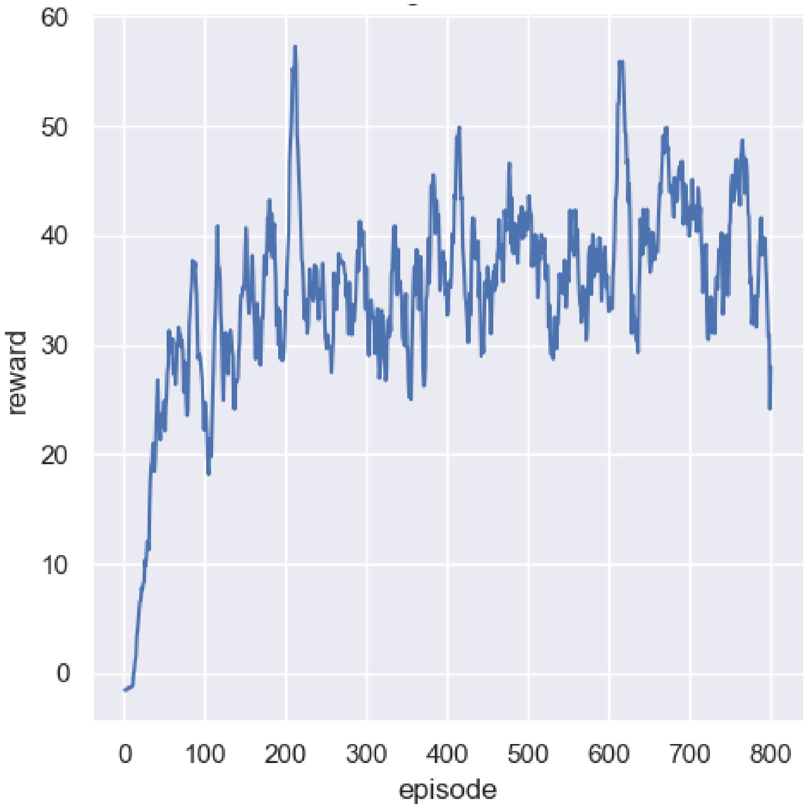
Keep the reward for eating food constant, and gradually increase the punishment for death:

1. Plot the training curves
2. Observe the change in average rewards
3. Briefly describe the conclusions

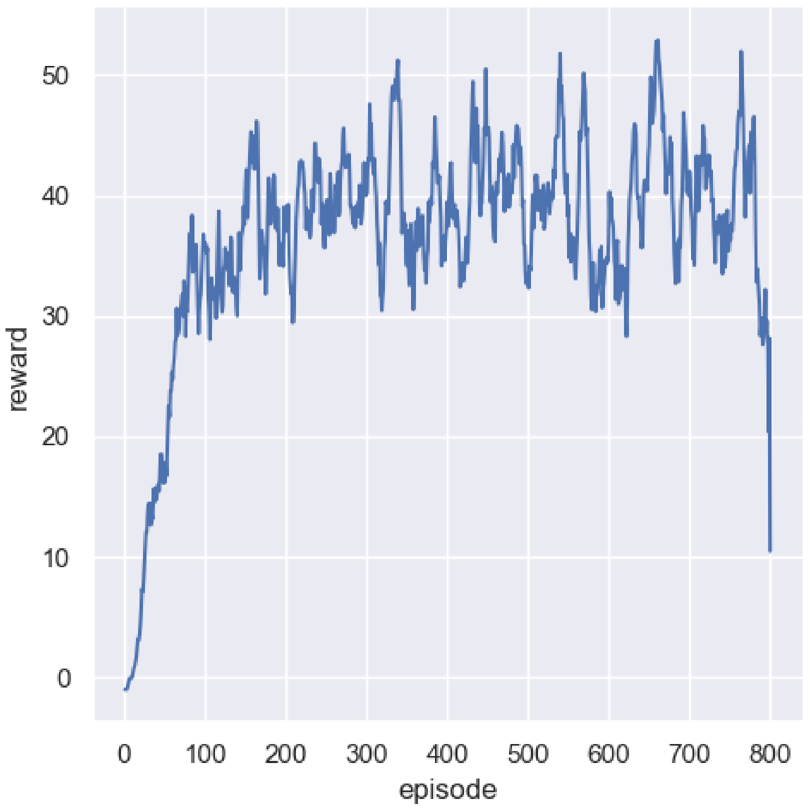
Training



Round 1

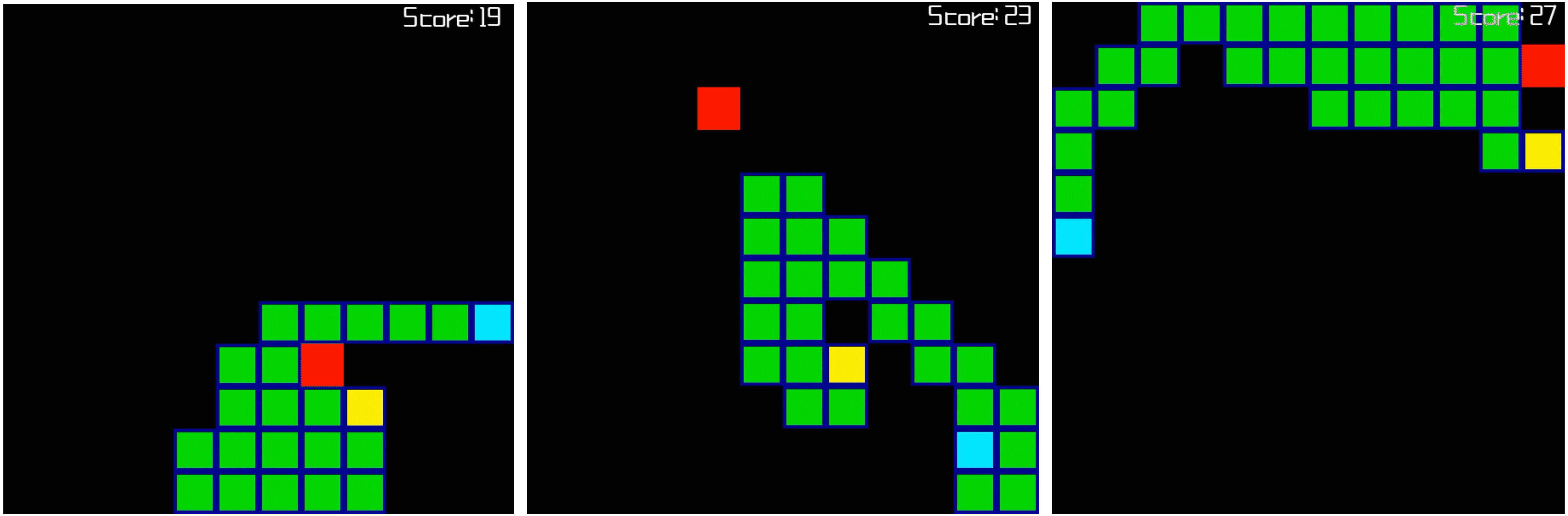


Round 2



Round 3

Results



Avg record	≈19	≈23	≈28

Conclusions

- Increasing the penalty for death leads to higher average records
- The training result of the low death penalty strategy has a low reward curve, but it performs well in the demo
- A particularly high reward for eating food can lead to quick success regardless of long-term safety

Future work

- The zigzag of snake body looks ugly, try to add punishment into reward for too many zigzags.

- Integrate saved model into C++ framework

https://github.com/george-chou/PPO-Snake-AI

https://github.com/george-chou/Snaqe