

Amazon DeepRacer

1. Creating a model


Create model

Race type

Choose a race type


☒ **Time trial**

The agent races against the clock on a well-marked track without stationary obstacles or moving competitors.




☐ **Object avoidance**

The vehicle races on a two-lane track with a fixed number of stationary obstacles placed along the track.



☐ **Head-to-head racing**

The vehicle races against other moving vehicles on a two-lane track.



Training algorithm and hyperparameters [Info](#)

☒ **PPO**

A state-of-the-art policy gradient algorithm which uses two neural networks during training – a policy network and a value network.

☐ **SAC**

Not limiting itself to seeking only the maximum of lifetime rewards, this algorithm embraces exploration, incentivizing entropy in its pursuit of optimal policy.

► **Hyperparameters**

Cancel

Previous

Next

2. Training algorithm and hyperparameters

Training algorithm and hyperparameters [Info](#)

☒ **PPO**
A state-of-the-art policy gradient algorithm which uses two neural networks during training – a policy network and a value network.

☐ **SAC**
Not limiting itself to seeking only the maximum of lifetime rewards, this algorithm embraces exploration, incentivizing entropy in its pursuit of optimal policy.

▼ **Hyperparameters**

Gradient descent batch size

☐ 32

☒ 64

☐ 128

☐ 256

☐ 512

Number of epochs

Integer between 3 and 10.

Learning rate

Real number between 0.00000001 (1e-8) and 0.001 (1e-3).

Entropy

Real number between 0 and 1.

Discount factor

Real number between 0 and 1.

Loss type

☐ Mean squared error

☒ Huber

Number of experience episodes between each policy-updating iteration

Integer between 5 and 100.

Cancel

Previous

Next

3. Reward function

Reward function [Info](#)

The reward function describes immediate feedback (as a score for reward or penalty) when the vehicle takes an action to move from a given position on the track to a new position. Its purpose is to encourage the vehicle to make moves along the track to reach its destination quickly. The model training process will attempt to find a policy which maximizes the average total reward the vehicle experiences. [Learn more](#) about the reward function and the reward input parameters you can use in your function.

Code editor

[Reward function examples](#)[Reset](#)[Validate](#)

```
1 def reward_function(params):
2     """
3     Example of rewarding the agent to follow center line
4     """
5
6     # Read input parameters
7     track_width = params['track_width']
8     distance_from_center = params['distance_from_center']
9
10    # Calculate 3 markers that are at varying distances away from the center line
11    marker_1 = 0.1 * track_width
12    marker_2 = 0.25 * track_width
13    marker_3 = 0.5 * track_width
14
15    # Give higher reward if the car is closer to center line and vice versa
16    if distance_from_center <= marker_1:
17        reward = 1.0
18    elif distance_from_center <= marker_2:
19        reward = 0.5
20    elif distance_from_center <= marker_3:
21        reward = 0.1
22    else:
23        reward = 1e-3 # likely crashed/ close to off track
24
25    return float(reward)
```

4. Training Configuration

Training configuration

Race type

Time trial

Environment simulation

2022 re:Invent Championship - Counterclockwise

Reward function

Show

Sensor(s)

Camera

Action space type

Continuous

Action space

Speed: [0.5 : 1] m/s

Steering angle: [-30 : 30] °

Framework

Tensorflow

Reinforcement learning algorithm

PPO

Hyperparameter

Value

Gradient descent batch size

64

Entropy

0.01

Discount factor

0.999

Loss type

Mean squared error

Learning rate

0.0003

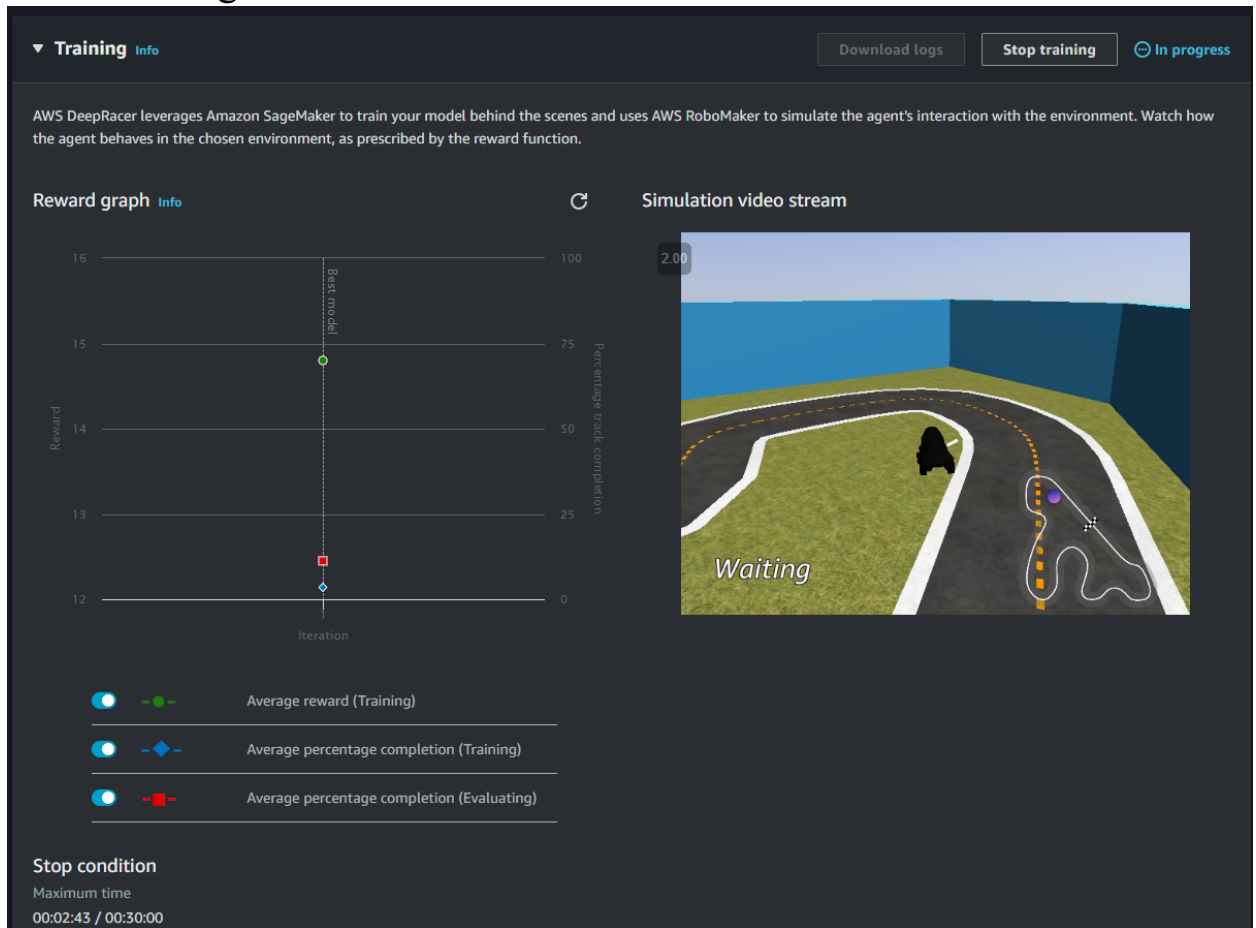
Number of experience episodes between each policy-updating iteration

20

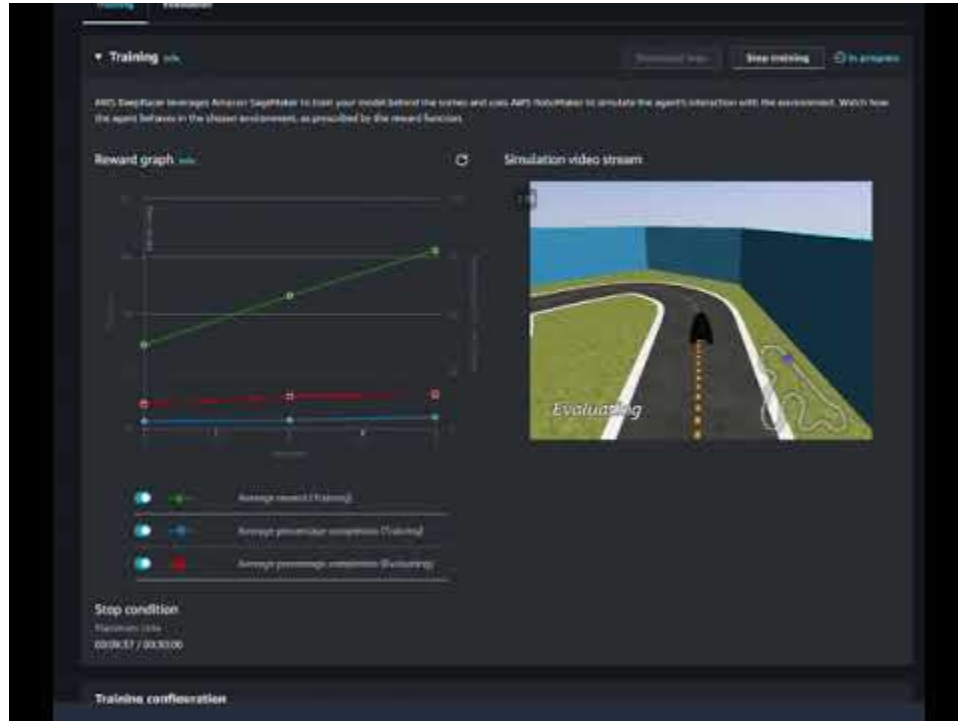
Number of epochs

10

5. Mid-Training



6. Video recording



7. Training completed

