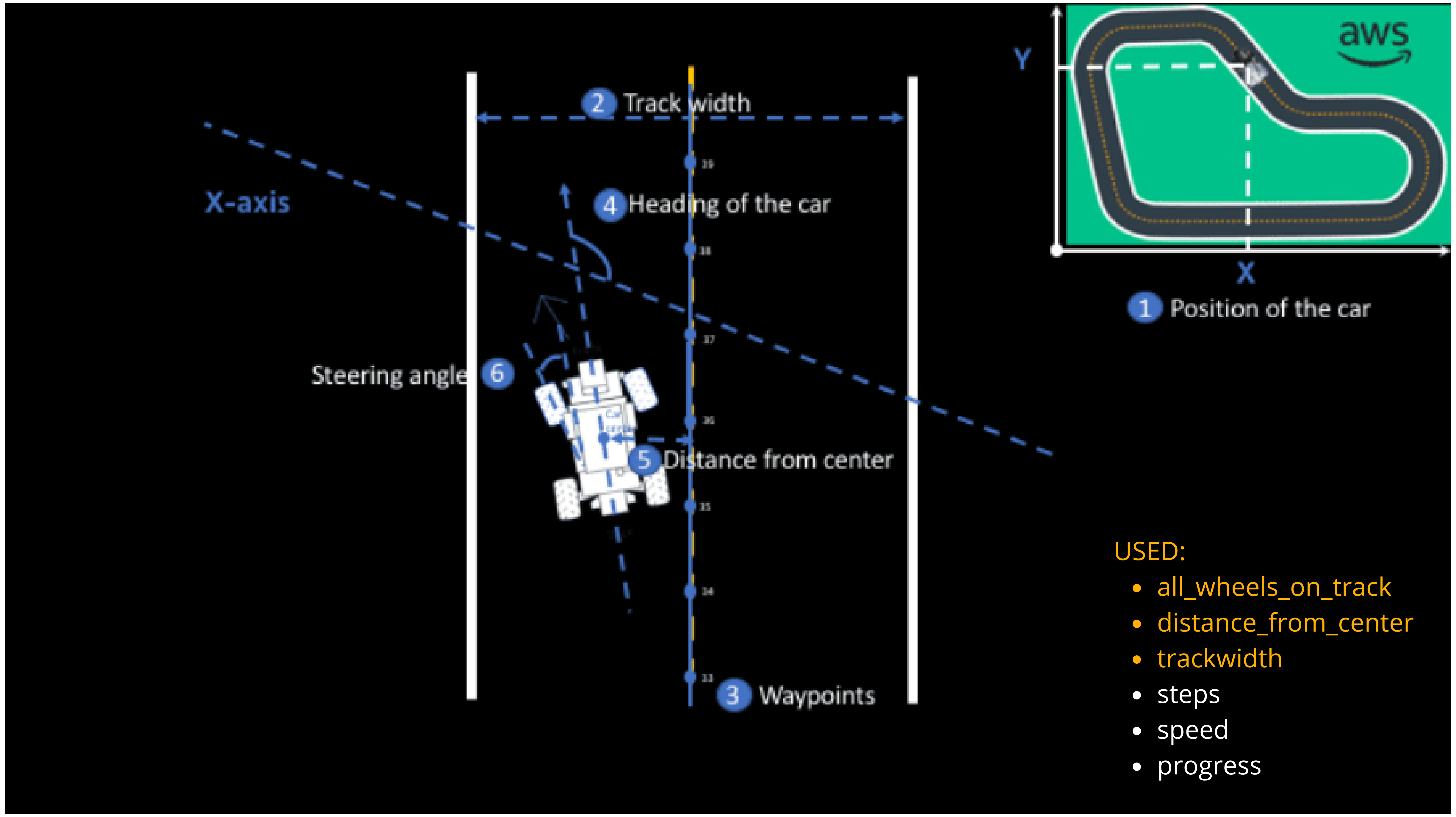
CASE STUDY 3
AWS DEEPRACER

by Jacqueline Genido
Lilliam Norori

Parameters

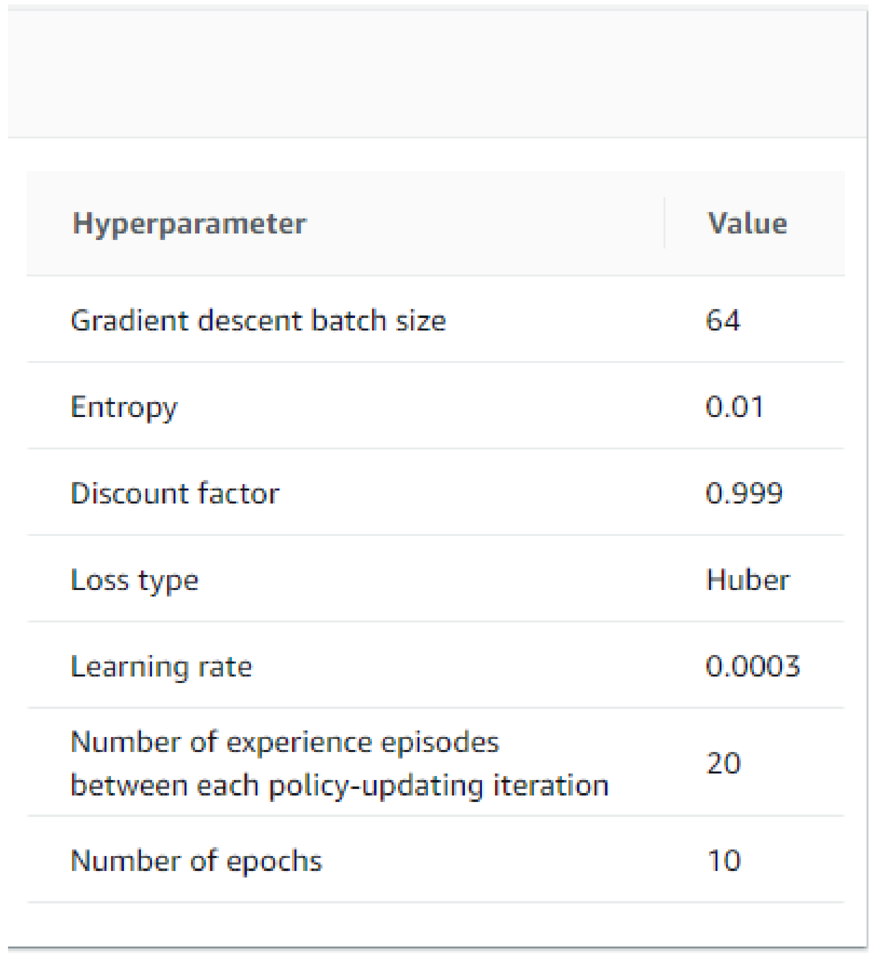


USED:

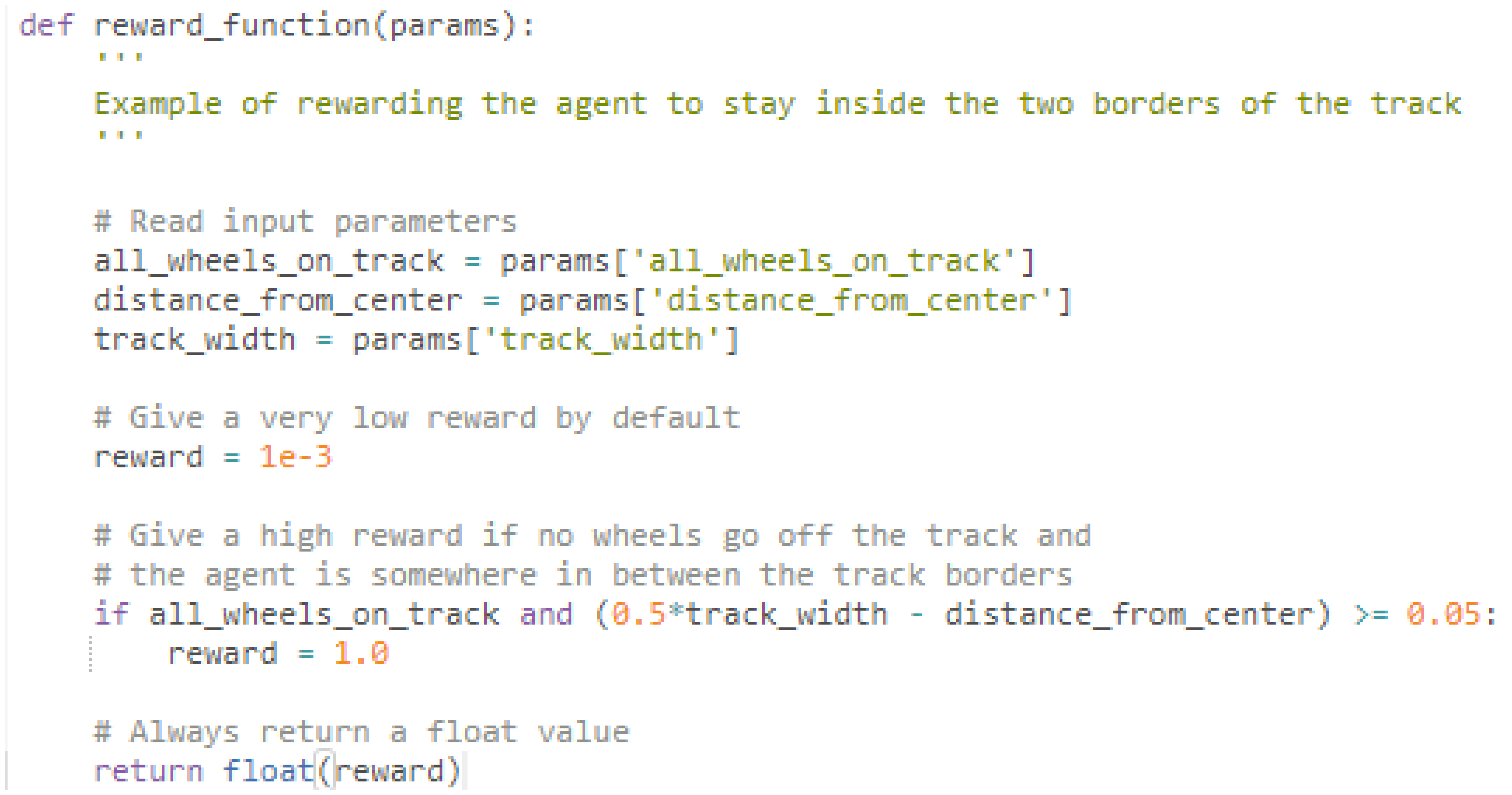
- all_wheels_on_track
- distance_from_center
- trackwidth
- steps
- speed
- progress

Basic Model

Example of rewarding the agent to stay inside the two borders of the track - Initial function



Hyperparameter	Value
Gradient descent batch size	64
Entropy	0.01
Discount factor	0.999
Loss type	Huber
Learning rate	0.0003
Number of experience episodes between each policy-updating iteration	20
Number of epochs	10



```
def reward_function(params):
    '''
    Example of rewarding the agent to stay inside the two borders of the track
    '''

    # Read input parameters
    all_wheels_on_track = params['all_wheels_on_track']
    distance_from_center = params['distance_from_center']
    track_width = params['track_width']

    # Give a very low reward by default
    reward = 1e-3

    # Give a high reward if no wheels go off the track and
    # the agent is somewhere in between the track borders
    if all_wheels_on_track and (0.5*track_width - distance_from_center) >= 0.05:
        reward = 1.0

    # Always return a float value
    return float(reward)
```

Hyperparameters

01 Learning Rate

Controls the speed at which your algorithm learns (it enlarges or shrinks the weight update after each epoch)

(From 0.0003 - 0.001)

02 Entropy

Added uncertainty to help the AWS DeepRacer vehicle explore the action space more broadly.

(From 0.01 to 0.1)

03 Discount Factor

How much of the future rewards contribute to the expected reward.

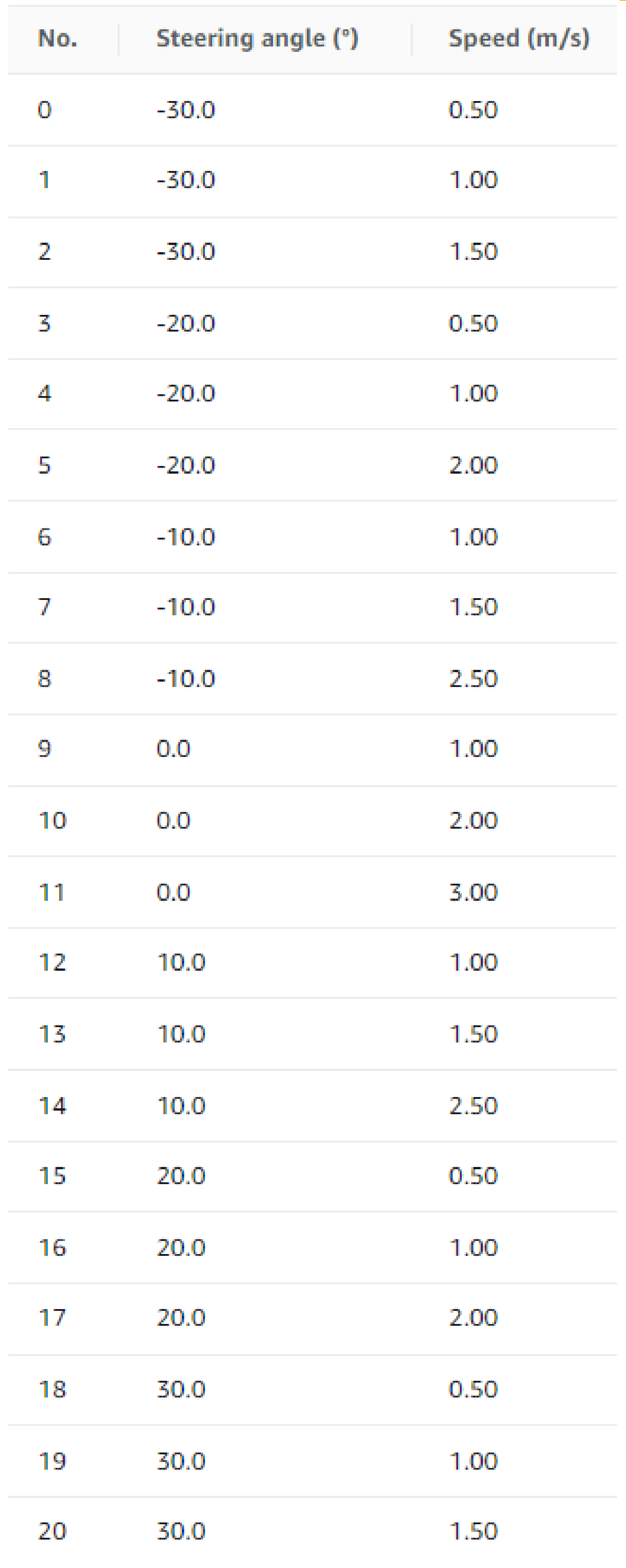
(From 0.999 to 0.9999)

Reward Function

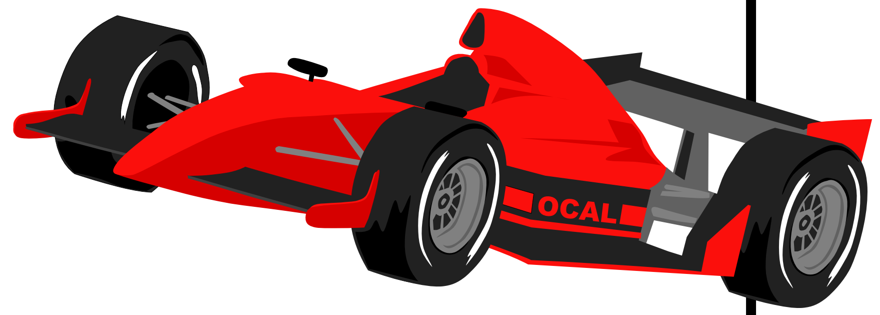
```python
def reward_function(params):
    '''
    simple reward function to keep agent on track and get through in as few steps
    '''

    if params["all_wheels_on_track"] and params["steps"] > 0:
        reward = ((params["progress"] / params ["steps"]) * 100) + (params["speed"]**2)
    else:
        reward = 0.01


    '''
    more rewards the closer agent gets to the finish line with example of rewarding the agent
    to stay inside the two borders of the track
    '''
    # Read input parameters
    all_wheels_on_track = params['all_wheels_on_track']
    distance_from_center = params['distance_from_center']
    track_width = params['track_width']

    # Give a very low reward by default
    reward = 1e-3

    # Give a high reward if no wheels go off the track and
    # the car is somewhere in between the track borders
    if all_wheels_on_track and (0.5*track_width - distance_from_center) >= 0.05:
        reward = 1.0
    if not params["all_wheels_on_track"]:
        reward = -1
    else:
        reward = params["progress"]
```

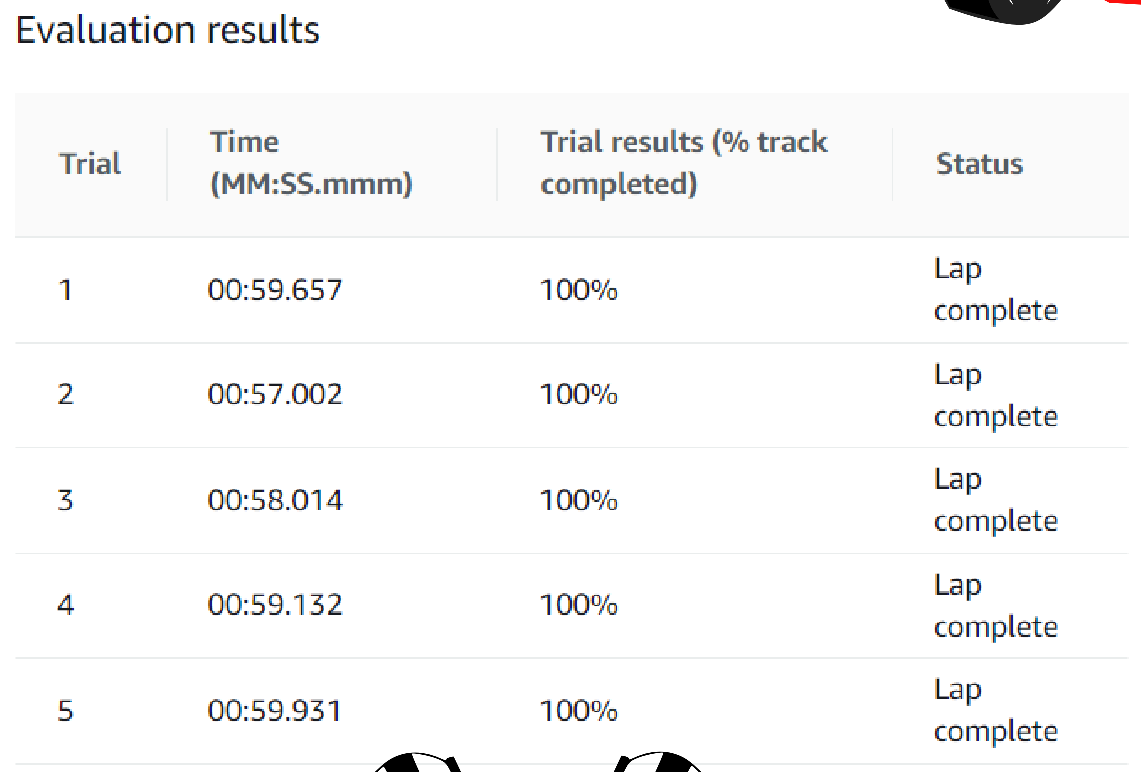
```python
    '''
    Example of rewarding the agent to follow center line
    '''
    # Read input parameters
    track_width = params['track_width']
    distance_from_center = params['distance_from_center']

    # Calculate 3 markers that are at varying distances away from the center line
    marker_1 = 0.01 * track_width
    marker_2 = 0.125 * track_width
    marker_3 = 0.25 * track_width

    # Give higher reward if the car is closer to center line and vice versa
    if distance_from_center <= marker_1:
        reward = 1.0
    elif distance_from_center <= marker_2:
        reward = 0.5
    elif distance_from_center <= marker_3:
        reward = 0.1
    else:
        reward = -1  # likely crashed/ close to off track



    return float(reward)
```

Action Space



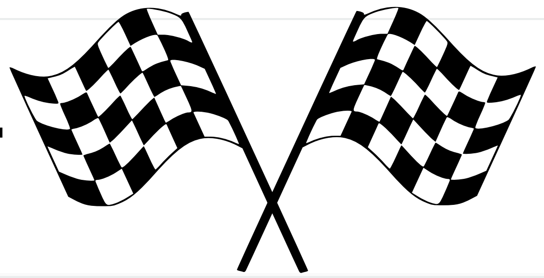
No.	Steering angle (°)	Speed (m/s)
0	-30.0	0.50
1	-30.0	1.00
2	-30.0	1.50
3	-20.0	0.50
4	-20.0	1.00
5	-20.0	2.00
6	-10.0	1.00
7	-10.0	1.50
8	-10.0	2.50
9	0.0	1.00
10	0.0	2.00
11	0.0	3.00
12	10.0	1.00
13	10.0	1.50
14	10.0	2.50
15	20.0	0.50
16	20.0	1.00
17	20.0	2.00
18	30.0	0.50
19	30.0	1.00
20	30.0	1.50

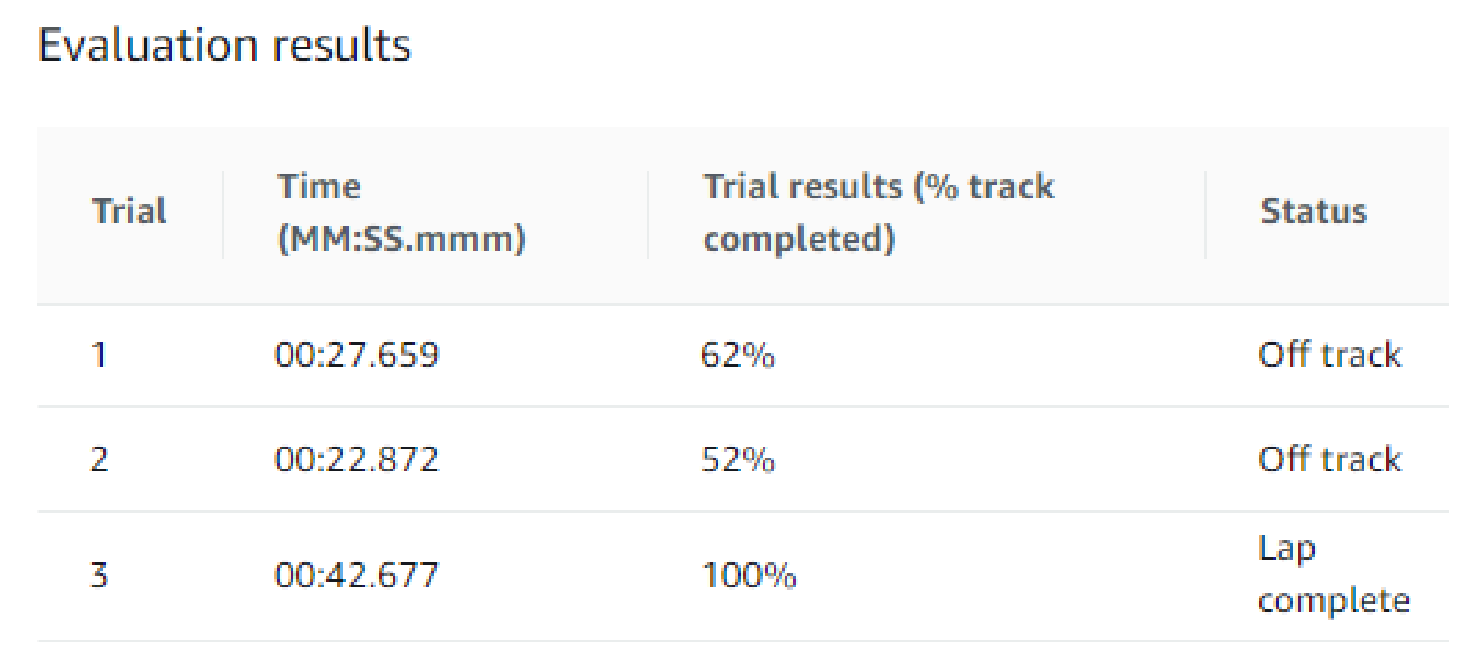




Evaluation results

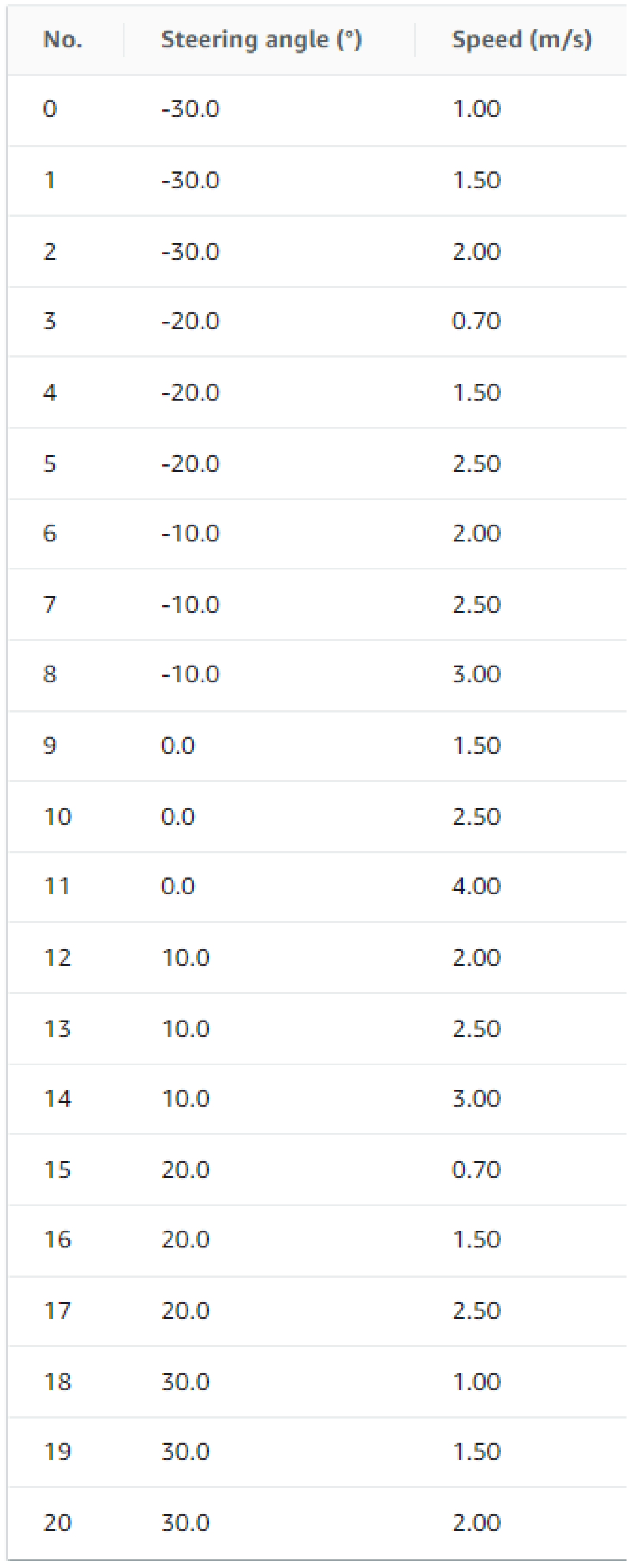
Trial	Time (MM:SS.mmm)	Trial results (% track completed)	Status
1	00:59.657	100%	Lap complete
2	00:57.002	100%	Lap complete
3	00:58.014	100%	Lap complete
4	00:59.132	100%	Lap complete
5	00:59.931	100%	Lap complete





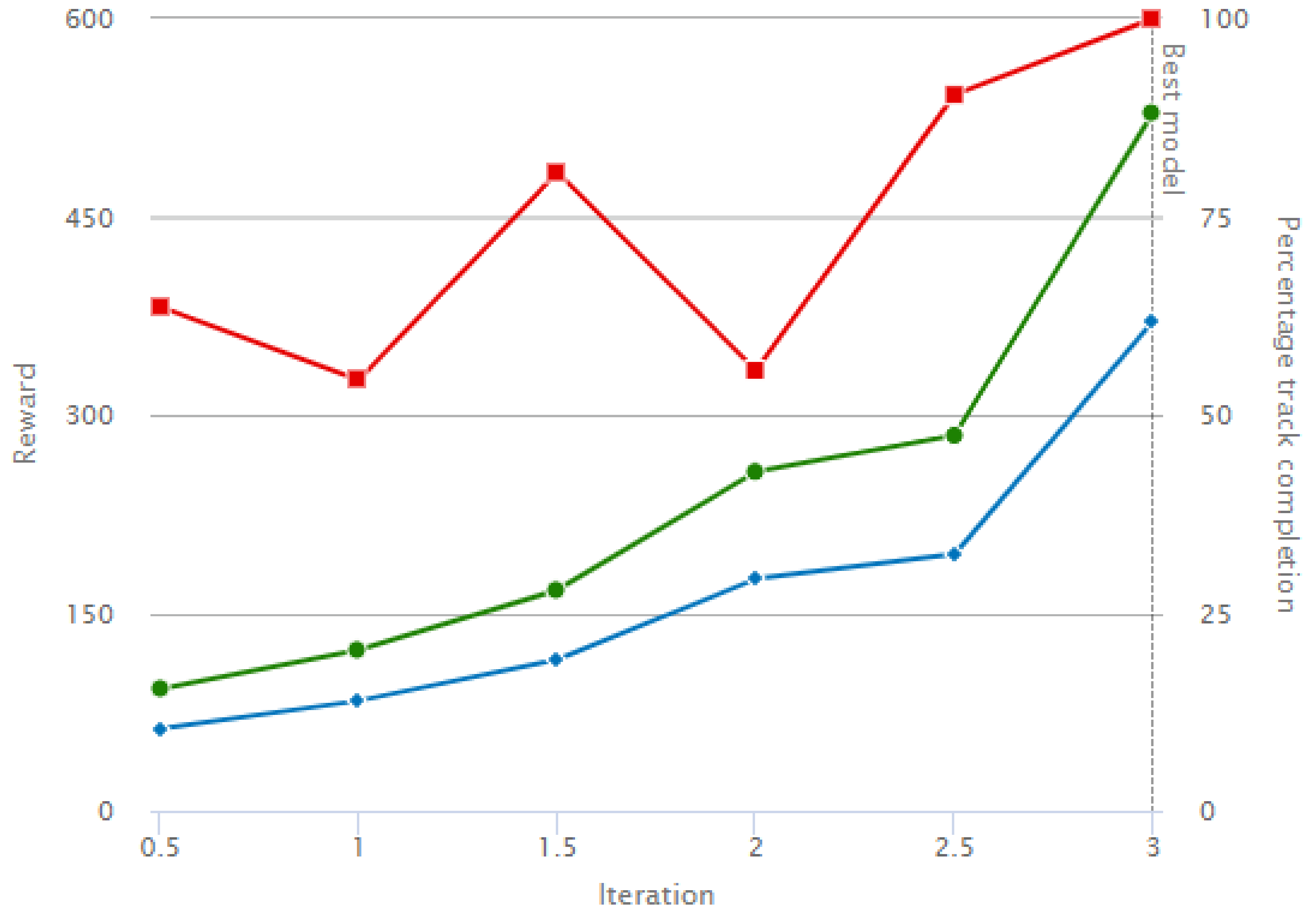
Evaluation results

Trial	Time (MM:SS.mmm)	Trial results (% track completed)	Status
1	00:27.659	62%	Off track
2	00:22.872	52%	Off track
3	00:42.677	100%	Lap complete



No.	Steering angle (°)	Speed (m/s)
0	-30.0	1.00
1	-30.0	1.50
2	-30.0	2.00
3	-20.0	0.70
4	-20.0	1.50
5	-20.0	2.50
6	-10.0	2.00
7	-10.0	2.50
8	-10.0	3.00
9	0.0	1.50
10	0.0	2.50
11	0.0	4.00
12	10.0	2.00
13	10.0	2.50
14	10.0	3.00
15	20.0	0.70
16	20.0	1.50
17	20.0	2.50
18	30.0	1.00
19	30.0	1.50
20	30.0	2.00

Reward Graph



- ● Average reward (Training)
- ◆ Average percentage completion (Training)
- ■ Average percentage completion (Evaluating)

Trial	Time (MM:SS.mmm)	Trial results (% track completed)	Status
5	00:41.867	100%	Lap complete

AWS DeepRacer

Group 3