AWS DEEPRACER LEAGUE DATA

General Training Configuration

Training Time	200 minutes
Sensors	Front-facing camera
Action Space Type	Continuous
Action Space: Speed	[1:3.5] m/s
Action Space: Steering Angle	[-20 : 20]^{o}
Framework	Tensorflow
Reinforcement Learning Algorithm	PPO

Hyperparameters

Gradient Descent Batch Size	64
Entropy	0.03
Discount Factor	0.999
Loss type	Huber
Learning Rate	0.01
Number of experience episodes between each policy-updating iteration	20
Number of epochs	10

Competition Details

Track Length	45.16m
Race Type	Time Trial
Ranking Method	Total Time
Style	Individual Lap
Resets	Unlimited Resets
Off-track Penalty	+3 seconds
Entry criteria	3 consecutive laps

Results

Rank	30/1941 (Top 2%)
Off-track	2
Total Lap Time	01:24:254

Reward Function

```
def reward_function(params):
    '''
    Example of rewarding the agent to follow center line
    '''
    # Read input parameters
    all_wheels_on_track = params['all_wheels_on_track']
    track_width = params['track_width']
    distance_from_center = params['distance_from_center']
    steps = params['steps']
    progress = params['progress']

    # Calculate 3 markers that are at varying distances away from the center line
    marker_1 = 0.01 * track_width
    marker_2 = 0.08 * track_width
    marker_3 = 0.2 * track_width
    marker_4 = 0.3 * track_width
    marker_5 = 0.5 * track_width

    # If all_wheels_on_track and (0.5 * track_width - distance_from_center) >= 0.
05
    # reward = 1.0

    # Total num of steps we want the car to finish the lap, it will vary dependin
g on track length
    TOTAL_NUM_STEPS = 10

    if (steps % 100) == 0 and progress > (steps / TOTAL_NUM_STEPS) * 100:
        reward += 10.0

    # Give higher reward if the car is closer to center line and vice versa
    if all_wheels_on_track and distance_from_center <= marker_1:
        reward = 1.0
    elif all_wheels_on_track and distance_from_center <= marker_2:
        reward = 0.5
    elif all_wheels_on_track and distance_from_center <= marker_3:
        reward = 0.2
    elif all_wheels_on_track and distance_from_center <= marker_4:
        reward = 0.1
    elif all_wheels_on_track and distance_from_center <= marker_5:
        reward = 0.001
    else:
        reward = 1e-4  # likely crashed/ close to off track
    return float(reward)
```

Reward Graph