# MaxQ

Manjunath Rao Chimbili

# AWS DeepRacer

AWS DeepRacer is an autonomous 1/18th scale race car designed to test RL models by racing on a physical track. Using **cameras to view the track** and a reinforcement model to control throttle and steering, the car shows how a model trained in a simulated environment can be transferred to the real-world.

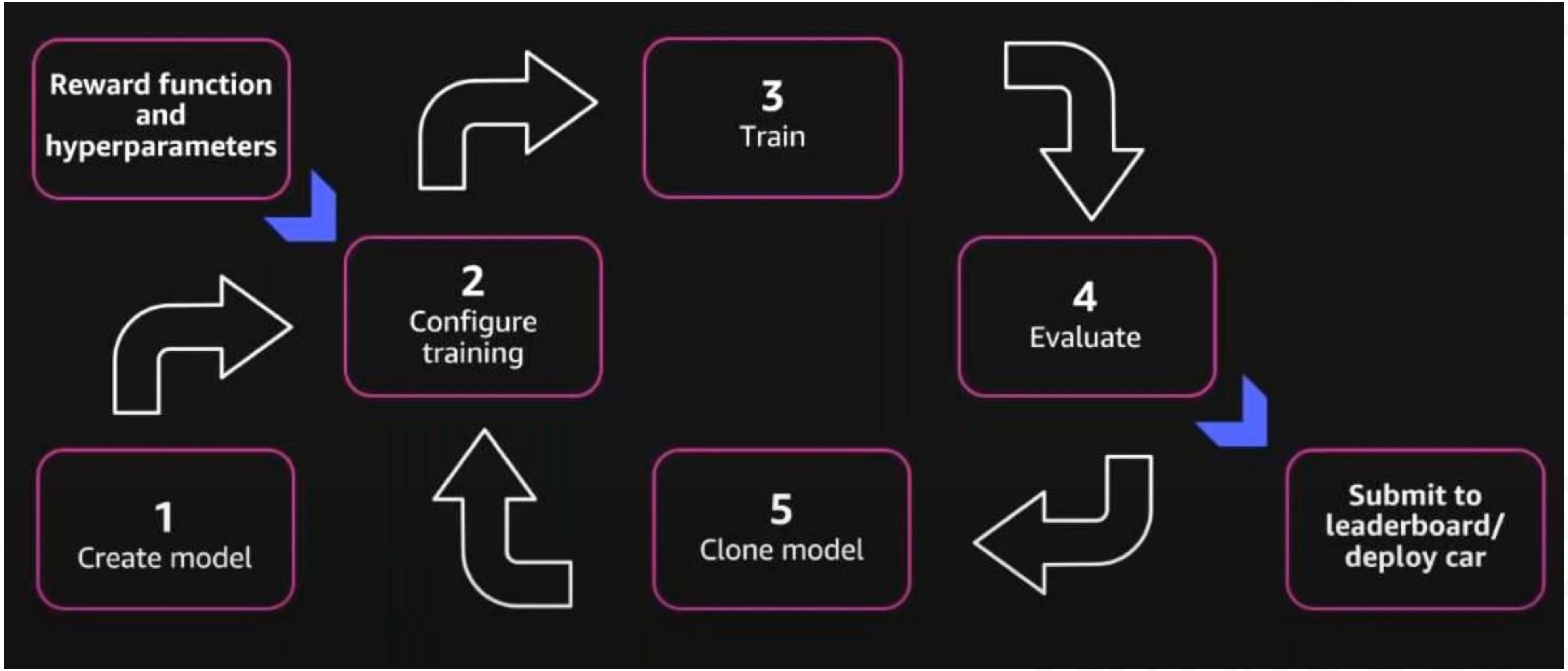
**State space:** The image fed through the camera mounted the vehicle's front

**Action space:** Discrete vs. continuous

**Reward function:** Custom rewards given based on type of action

# Input params considered:

| | |
|---|---|
| *all_wheels_on_track* | *-flag to indicate if the agent is on the track* |
| *track_width* | *-width of the track* |
| *distance_from_center* | *-distance in meters from the track center* |
| *waypoints* | *-list of (x,y) as milestones along the track center* |
| *closest_waypoints* | *-indices of the two nearest waypoints.* |
| *heading* | *-agent's yaw in degrees* |
| *speed* | *-agent's speed in meters per second (m/s)* |



Reward function and hyperparameters

1 Create model

2 Configure training

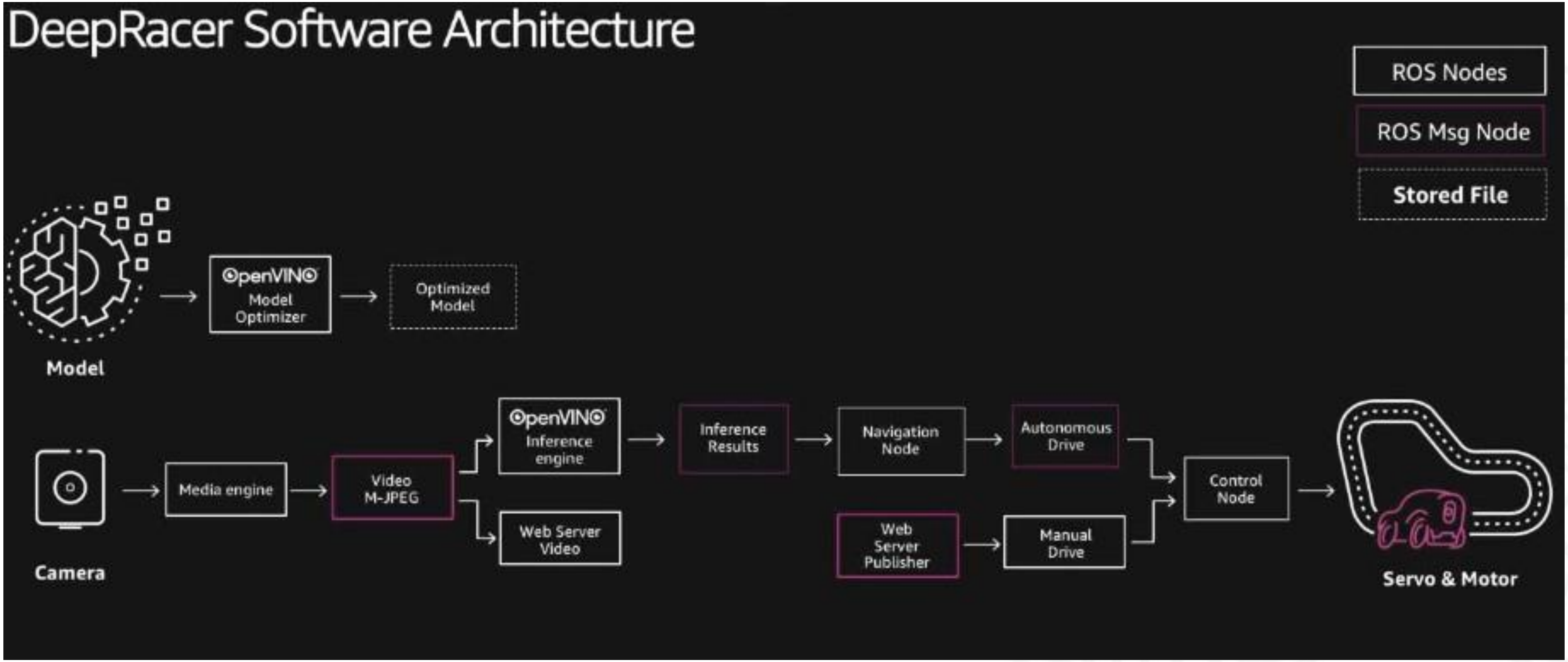
3 Train

4 Evaluate

5 Clone model

Submit to leaderboard/ deploy car

https://www.youtube.com/watch?v=X_Jm8bMklHc&ab_channel=AWSOnlineTechTalks



# DeepRacer Software Architecture

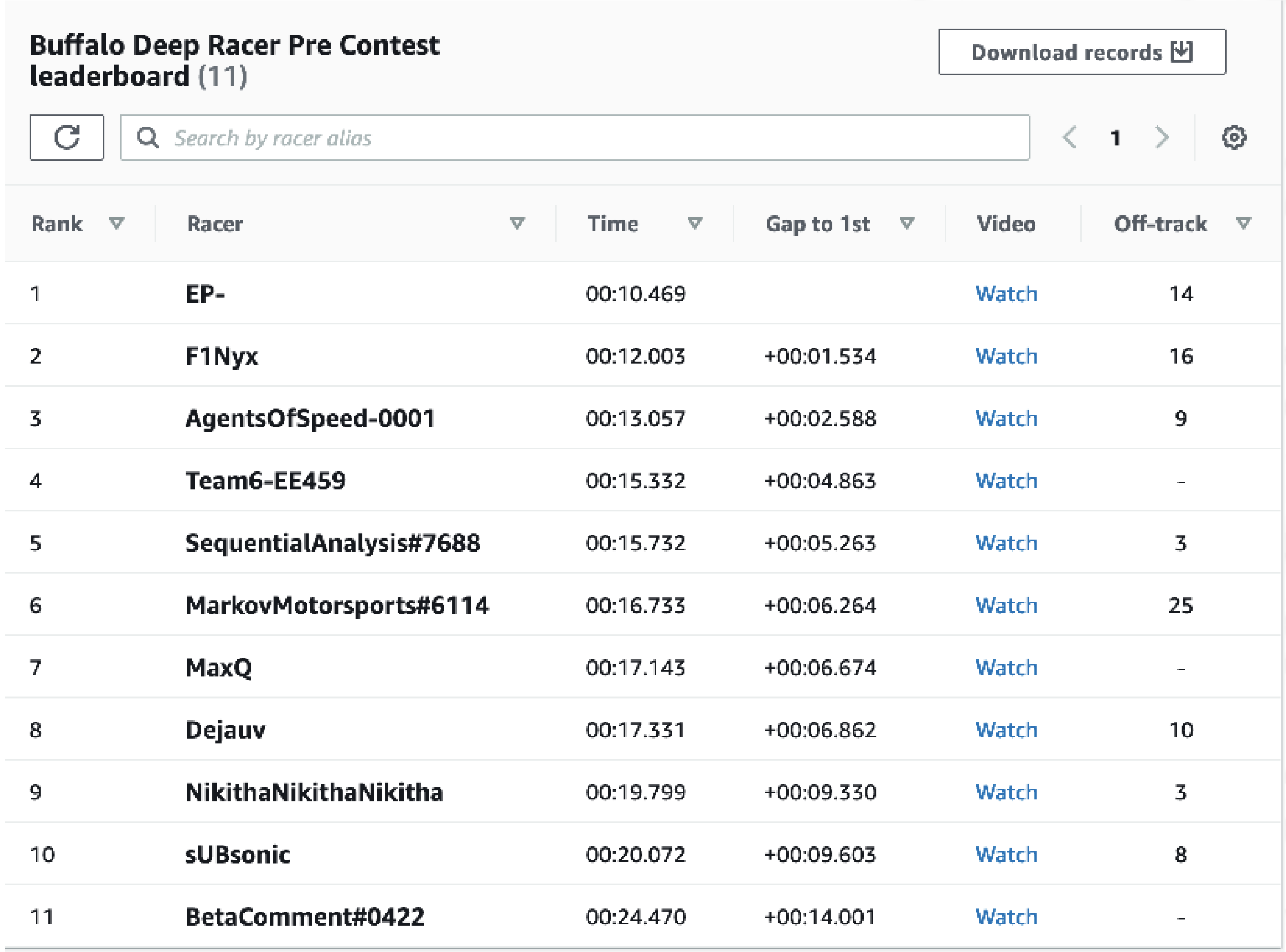
https://www.youtube.com/watch?v=X_Jm8bMklHc&ab_channel=AWSOnlineTechTalks

# Pre-Contest-Optimal Model

*Rank: **#7***

*Resets: **0***

*Best lap time: **17.143***



Buffalo Deep Racer Pre Contest leaderboard (11)

| Rank | Racer | Time | Gap to 1st | Video | Off-track |
|---|---|---|---|---|---|
| 1 | EP- | 00:10.469 | | Watch | 14 |
| 2 | F1Nyx | 00:12.003 | +00:01.534 | Watch | 16 |
| 3 | AgentsOfSpeed-0001 | 00:13.057 | +00:02.588 | Watch | 9 |
| 4 | Team6-EE459 | 00:15.332 | +00:04.863 | Watch | - |
| 5 | SequentialAnalysis#7688 | 00:15.732 | +00:05.263 | Watch | 3 |
| 6 | MarkovMotorsports#6114 | 00:16.733 | +00:06.264 | Watch | 25 |
| 7 | MaxQ | 00:17.143 | +00:06.674 | Watch | - |
| 8 | Dejauv | 00:17.331 | +00:06.862 | Watch | 10 |
| 9 | NikithaNikithaNikitha | 00:19.799 | +00:09.330 | Watch | 3 |
| 10 | sUBsonic | 00:20.072 | +00:09.603 | Watch | 8 |
| 11 | BetaComment#0422 | 00:24.470 | +00:14.001 | Watch | - |

# Pre-Contest optimal model

**No change in hyperparameters**

**Speed: min:**0.5 & **max:** 1 m/s

**Action space:** Continuous

**Trained on:** re: Invent 2018

**Training hours:** 5 hours

**Reward formulation:**

On Track Reward

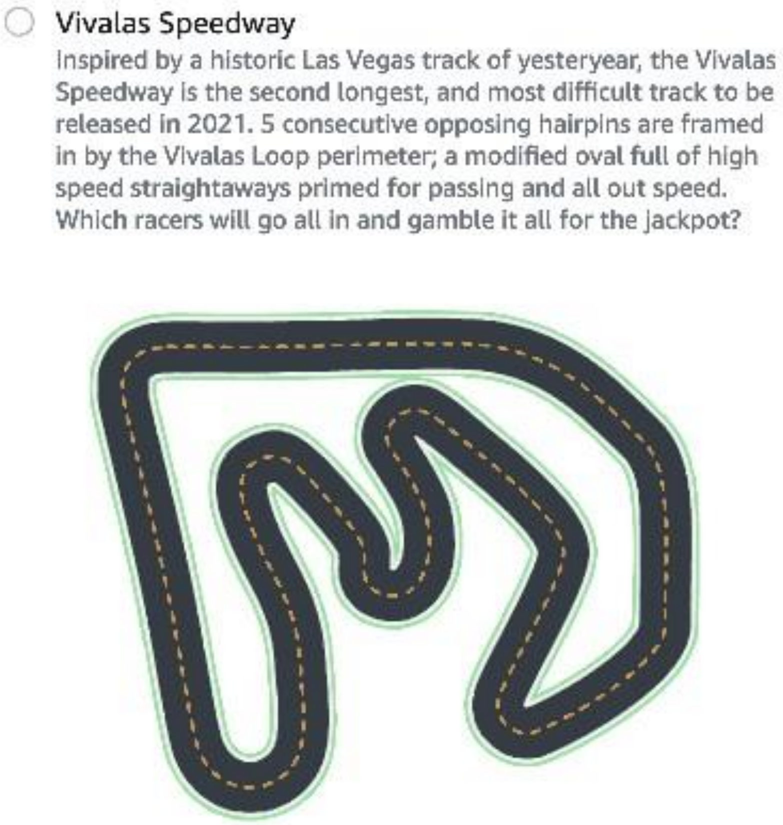
Center vehicle reward

Zig-zag steering penalty

Low-speed penalty

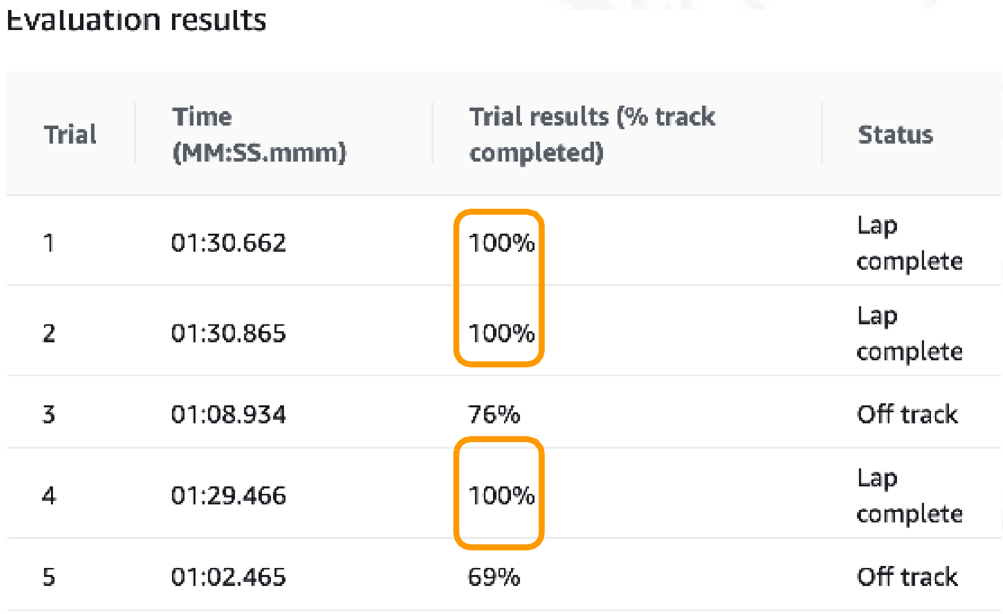
Corner Overspeed Penalty

# Performance on complex tracks



○ Vivalas Speedway

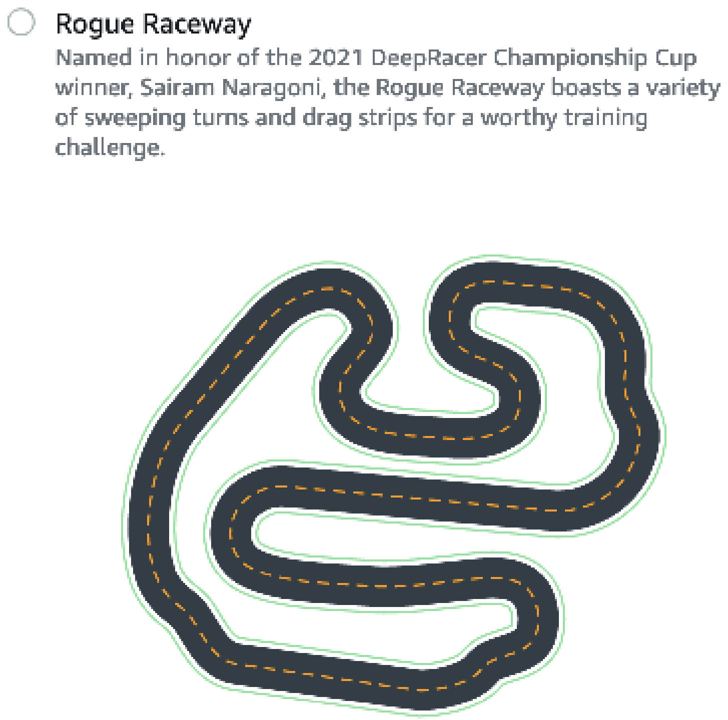
Inspired by a historic Las Vegas track of yesteryear, the Vivalas Speedway is the second longest, and most difficult track to be released in 2021. 5 consecutive opposing hairpins are framed in by the Vivalas Loop perimeter; a modified oval full of high speed straightaways primed for passing and all out speed. Which racers will go all in and gamble it all for the jackpot?



Evaluation results

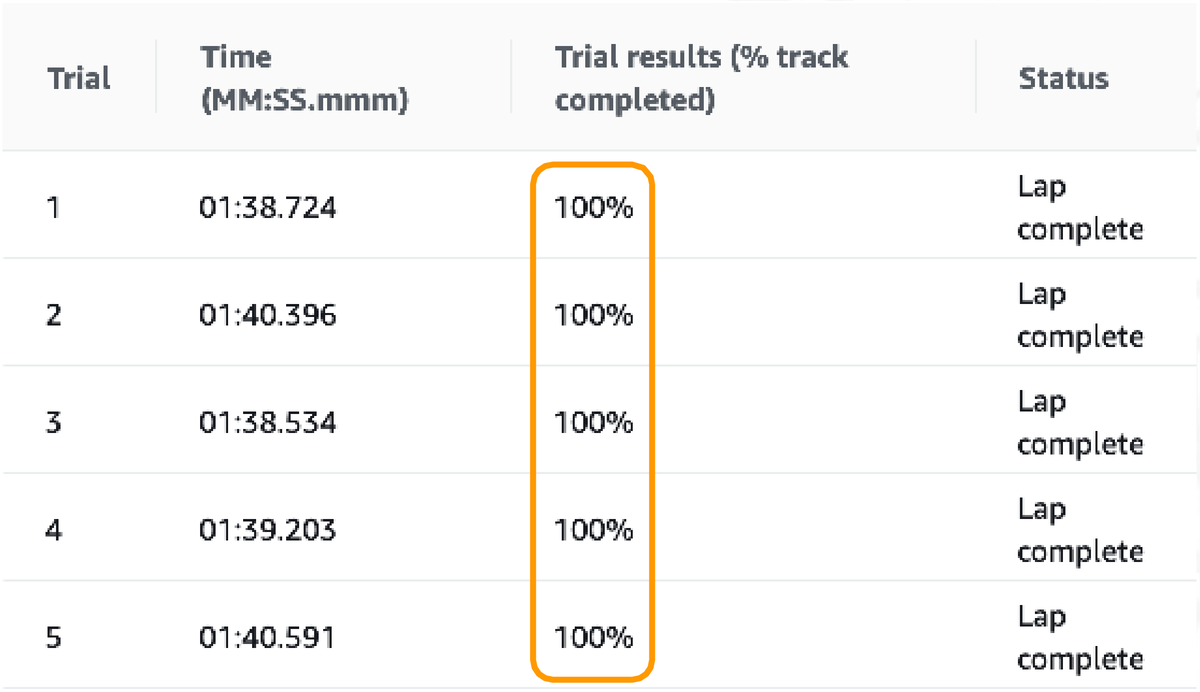
| Trial | Time (MM:SS.mmm) | Trial results (% track completed) | Status |
|---|---|---|---|
| 1 | 01:30.662 | 100% | Lap complete |
| 2 | 01:30.865 | 100% | Lap complete |
| 3 | 01:08.934 | 76% | Off track |
| 4 | 01:29.466 | 100% | Lap complete |
| 5 | 01:02.465 | 69% | Off track |

# Performance on complex tracks



○ Rogue Raceway

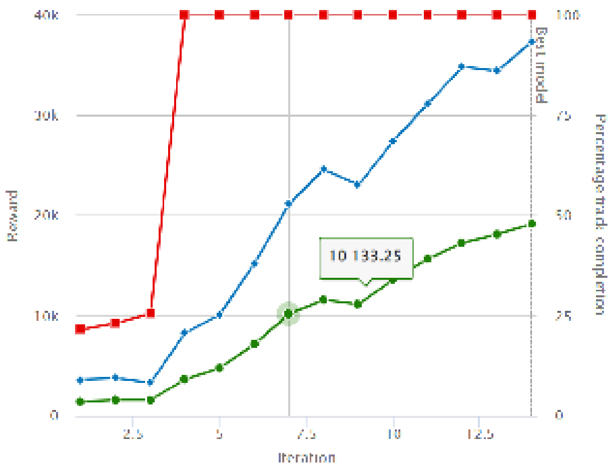
Named in honor of the 2021 DeepRacer Championship Cup winner, Sairam Naragoni, the Rogue Raceway boasts a variety of sweeping turns and drag strips for a worthy training challenge.

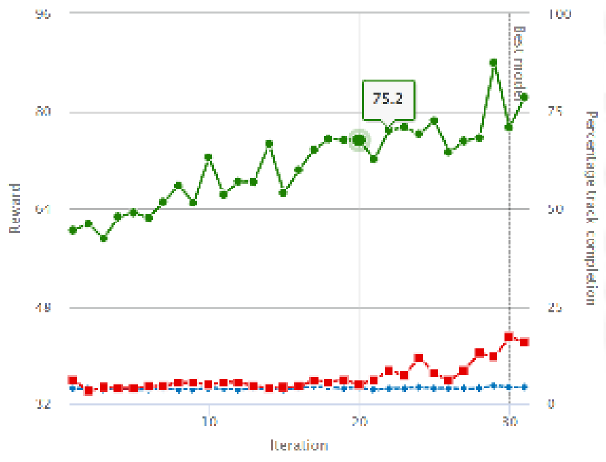


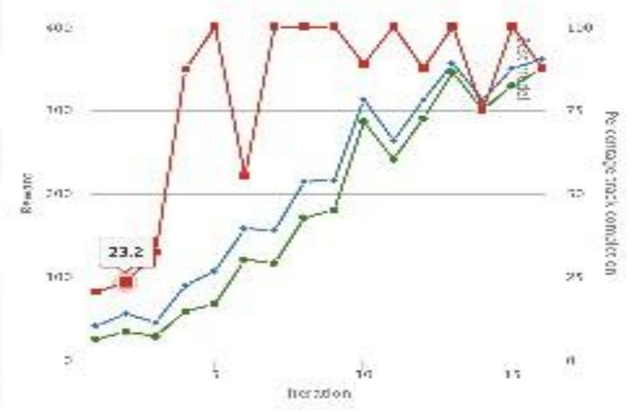
| Trial | Time (MM:SS.mmm) | Trial results (% track completed) | Status |
|---|---|---|---|
| 1 | 01:38.724 | 100% | Lap complete |
| 2 | 01:40.396 | 100% | Lap complete |
| 3 | 01:38.534 | 100% | Lap complete |
| 4 | 01:39.203 | 100% | Lap complete |
| 5 | 01:40.591 | 100% | Lap complete |

# Experimented models:

| | Action space | Speed | Training hours | Major changes |
|---|---|---|---|---|
| **Model 1:** | Continuous | [ 0.5 : 1.5 ] | 3 hours | # Decreased throttle while steering |
| **Model 2:** | Discrete[22 actions] | [ 0.5 : 3 ] | 1 hour | # Reward for **progress** parameter |
| **Model 3:** | Continuous | [ 0.5 : 1 ] | 4 hours | #Speed Incentive & Straightness |

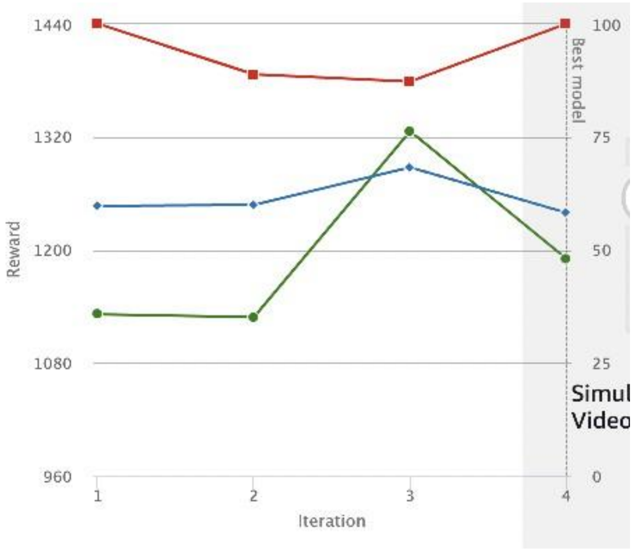


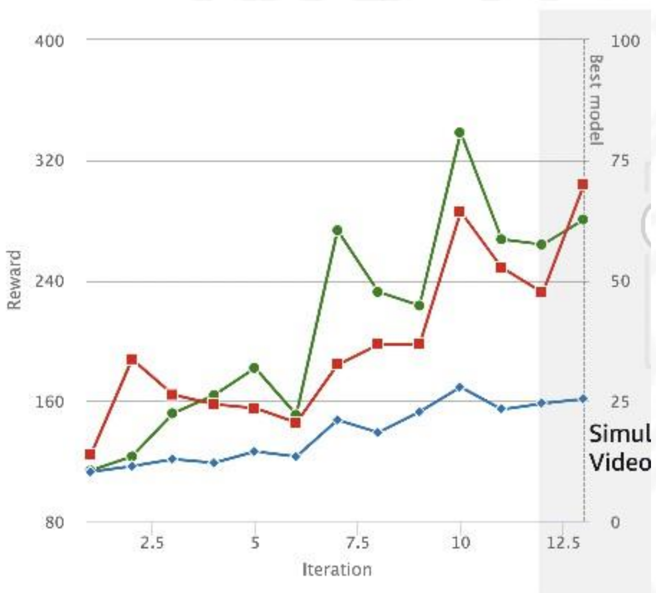




# Experimented models:

| | Action space | Speed | Training hours | Major changes |
|---|---|---|---|---|
| **Model 4:** | Discrete[15 actions] | [ 0.5 : 3 ] | 5 Hours | # closer to center line |
| **Model 5:** | Discrete | [ 0.5 : 1.5 ] | 2 Hours | # 15-20 actions space(_clone) |



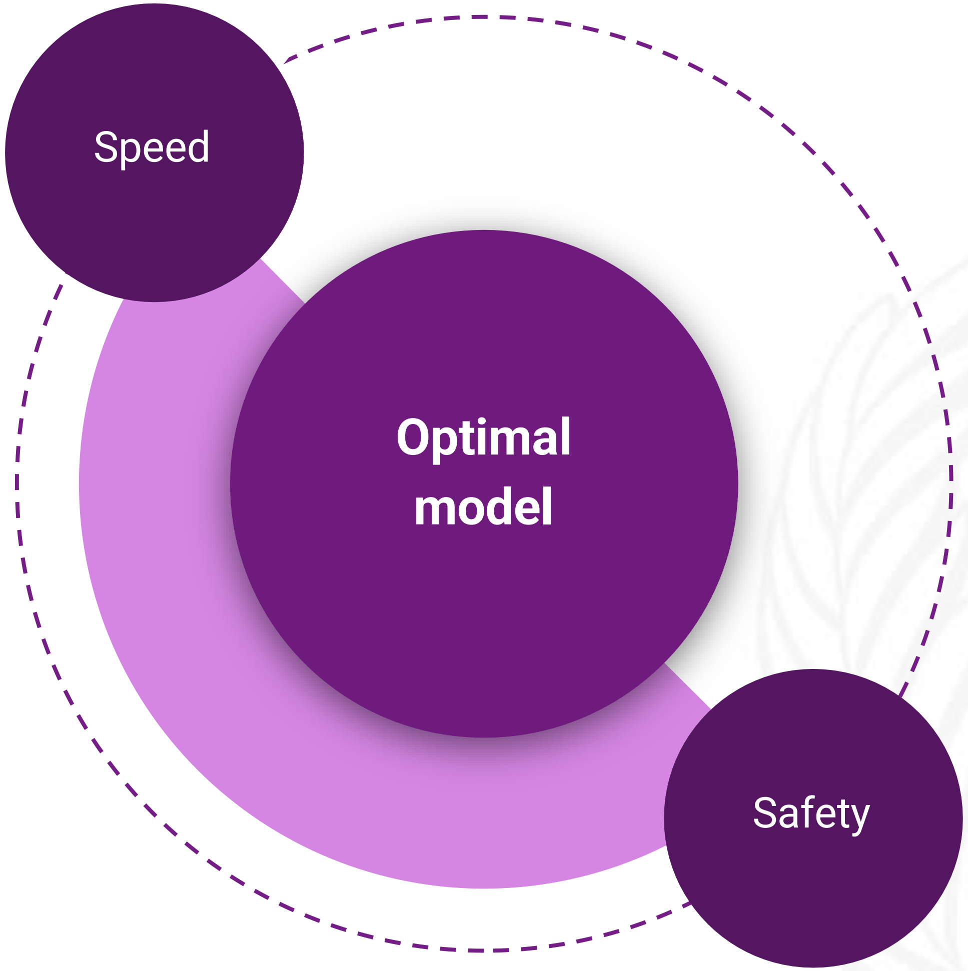


# Observations

1 *Continuous action space*

2 *Speed: 0.5 - 2 m/s*

3 *Corner prediction strategy*

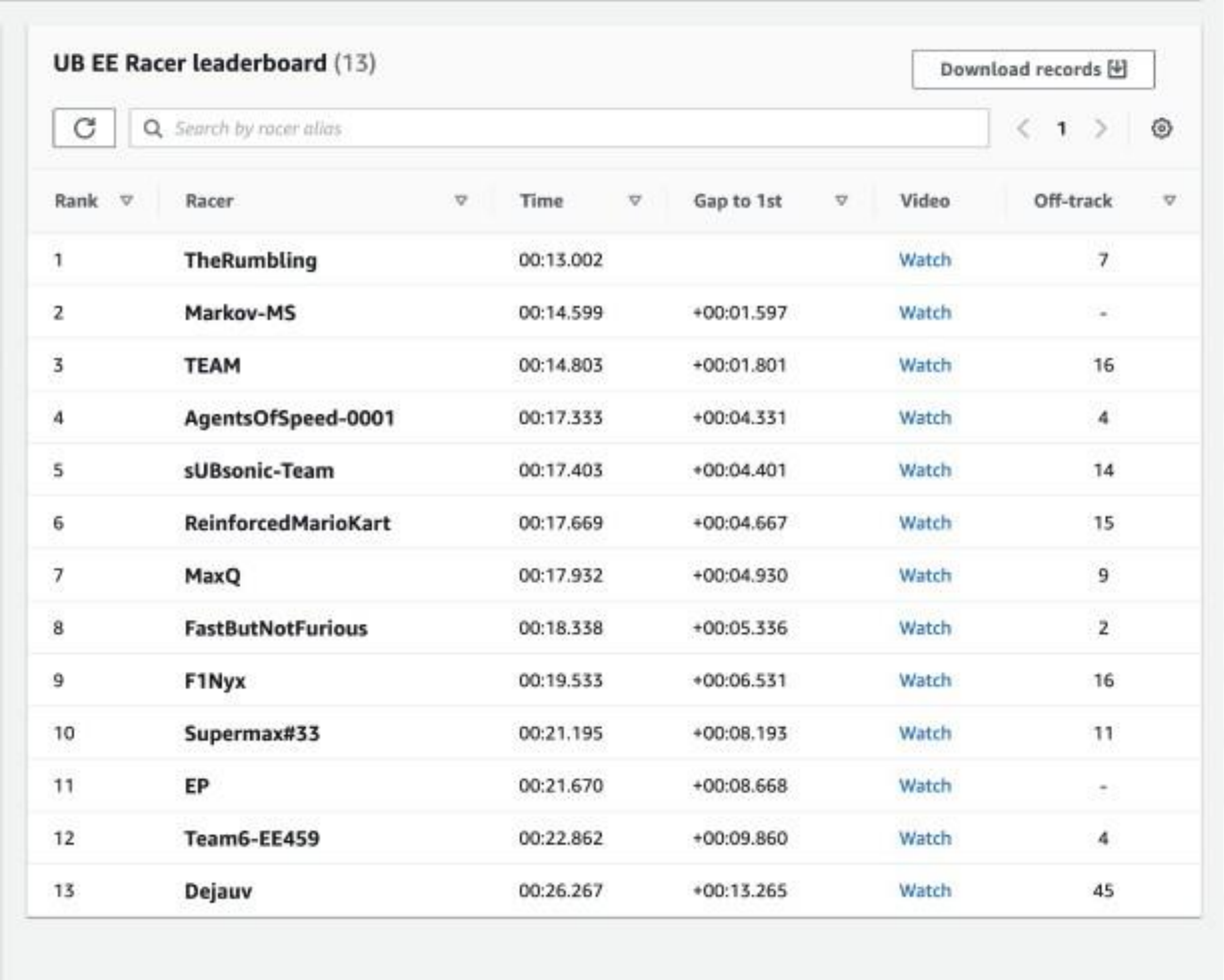

Speed
Optimal model
Safety

# Final Submission Model

*Rank: **#7***

*Resets: **9***

*Best lap time: **17.932***



**UB EE Racer leaderboard** (13)

| Rank | Racer | Time | Gap to 1st | Video | Off-track |
|---|---|---|---|---|---|
| 1 | TheRumbling | 00:13.002 | | Watch | 7 |
| 2 | Markov-MS | 00:14.599 | +00:01.597 | Watch | - |
| 3 | TEAM | 00:14.803 | +00:01.801 | Watch | 16 |
| 4 | AgentsOfSpeed-0001 | 00:17.333 | +00:04.331 | Watch | 4 |
| 5 | sUBsonic-Team | 00:17.403 | +00:04.401 | Watch | 14 |
| 6 | ReinforcedMarioKart | 00:17.669 | +00:04.667 | Watch | 15 |
| 7 | MaxQ | 00:17.932 | +00:04.930 | Watch | 9 |
| 8 | FastButNotFurious | 00:18.338 | +00:05.336 | Watch | 2 |
| 9 | F1Nyx | 00:19.533 | +00:06.531 | Watch | 16 |
| 10 | Supermax#33 | 00:21.195 | +00:08.193 | Watch | 11 |
| 11 | EP | 00:21.670 | +00:08.668 | Watch | - |
| 12 | Team6-EE459 | 00:22.862 | +00:09.860 | Watch | 4 |
| 13 | Dejauv | 00:26.267 | +00:13.265 | Watch | 45 |

# Final optimal model

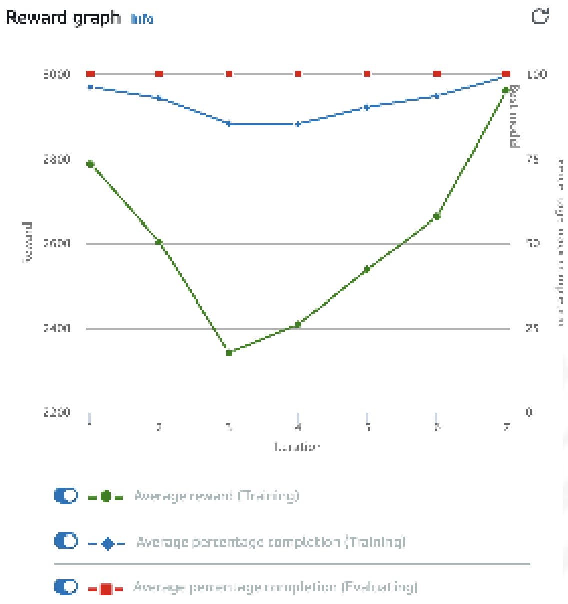
**No change in hyperparameters**

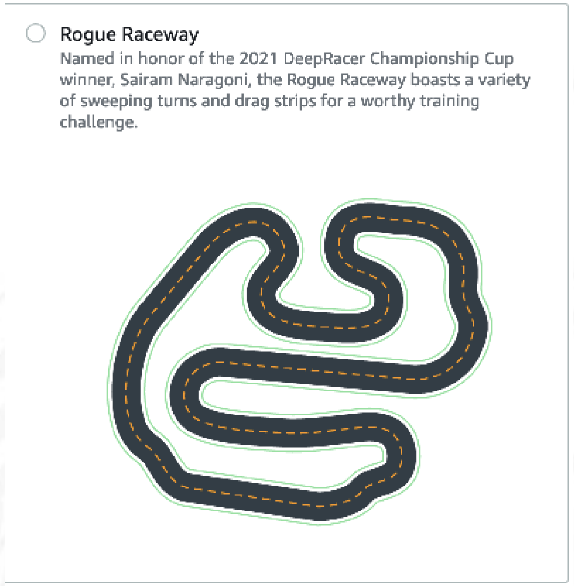
**Speed: min:**0.5 & **max:** 2 m/s

**Action space:** Continuous

**Trained on:** re: Invent 2018

**Training hours:** 12 hours



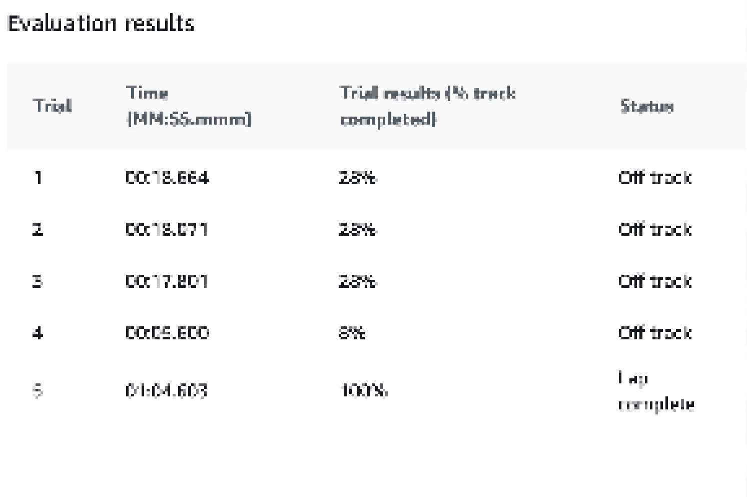


○ Rogue Raceway

Named in honor of the 2021 DeepRacer Championship Cup winner, Sairam Naragoni, the Rogue Raceway boasts a variety of sweeping turns and drag strips for a worthy training challenge.

# Method of evaluation

- We  made decisions on completion rate on challenging tracks and not community races .
- The best lap time will always be the lap completed without resets.

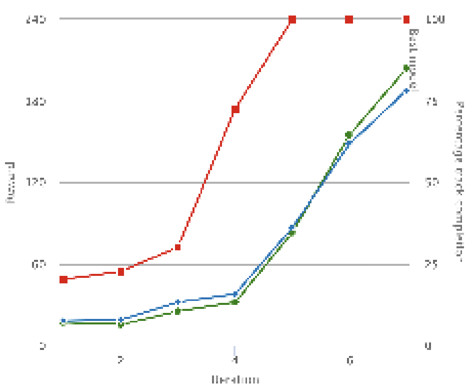


Evaluation results

| Trial | Time (MM:SS.mmm) | Trial results (% track completed) | Status |
|---|---|---|---|
| 1 | 00:18.664 | 28% | Off track |
| 2 | 00:18.071 | 28% | Off track |
| 3 | 00:17.801 | 28% | Off track |
| 4 | 00:05.600 | 8% | Off track |
| 5 | 01:04.603 | 100% | Lap complete |

# Final Model Development

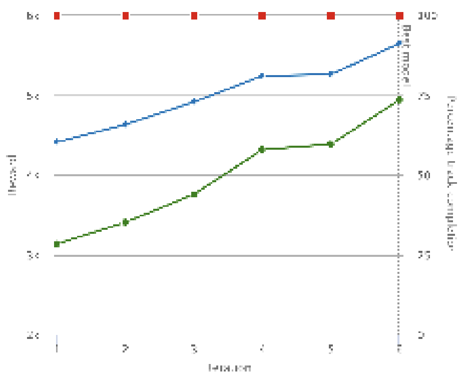
1m/s



1hr

- Basic Reward Function
- Only Line Markers

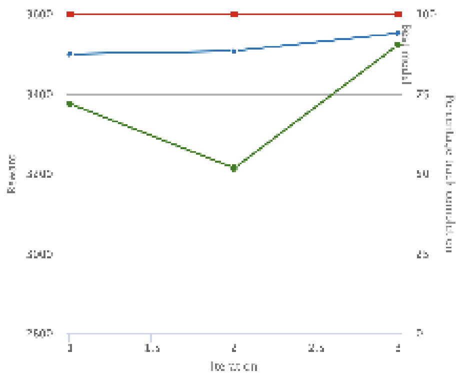
1m/s



4.5hrs

- Added zig-zag steering penalty
- Added low speed penalty
- Added corner perception
- Evaluation on multiple complex tracks gives 100% percent completion

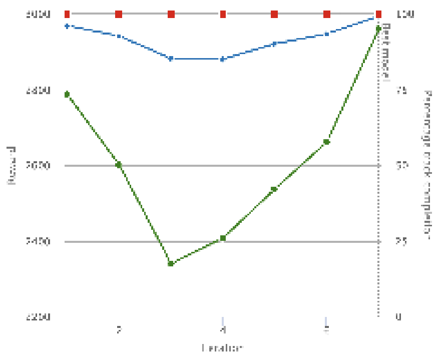
1.5m/s



2.5hrs

- Adjusted thresholds.
- Increased speed to 1.5m/s
- Evaluations on few complex tracks gives 100 % completion result

2m/s

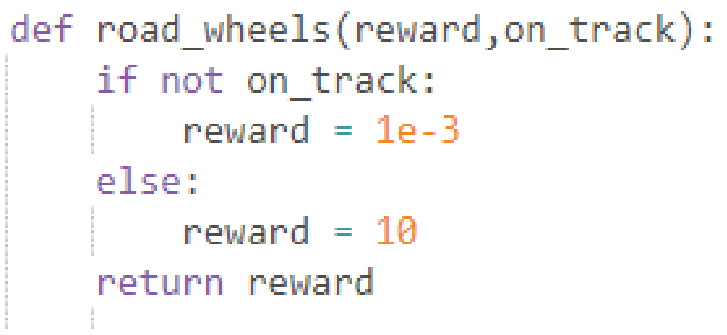


4.5hrs

- Increased speed to 2m/s
- Evaluation shows higher incompletion rate on even moderate tracks.

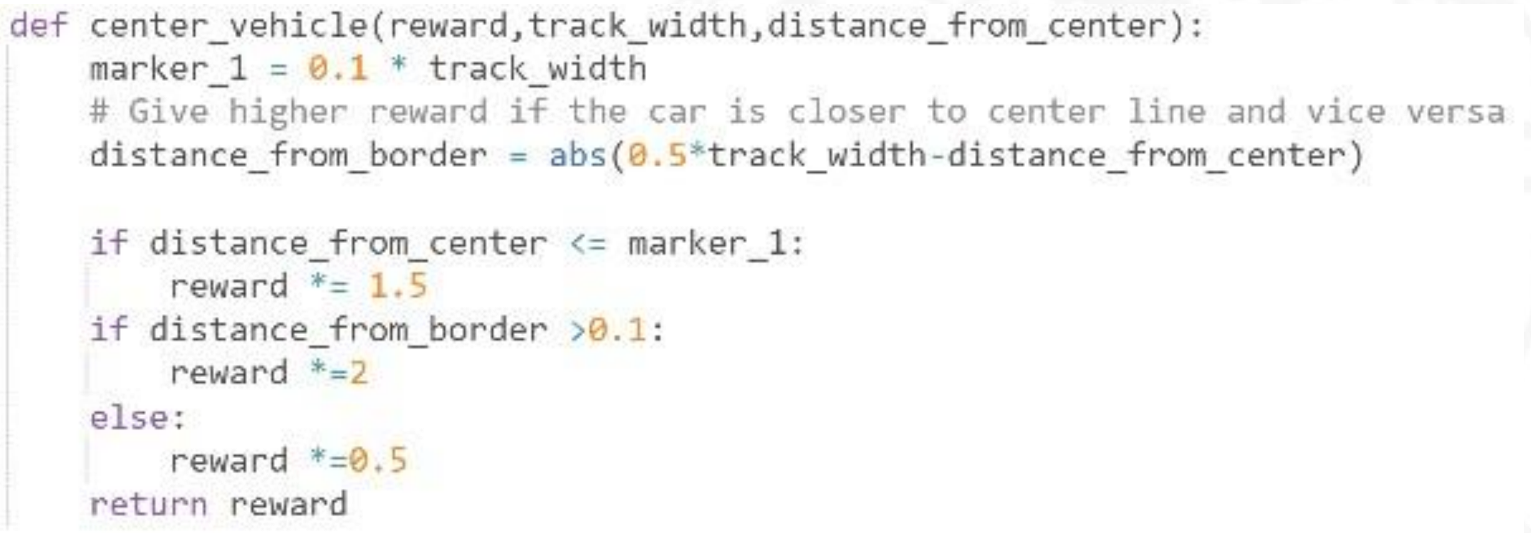
# Elements of the reward function

## On Track Reward



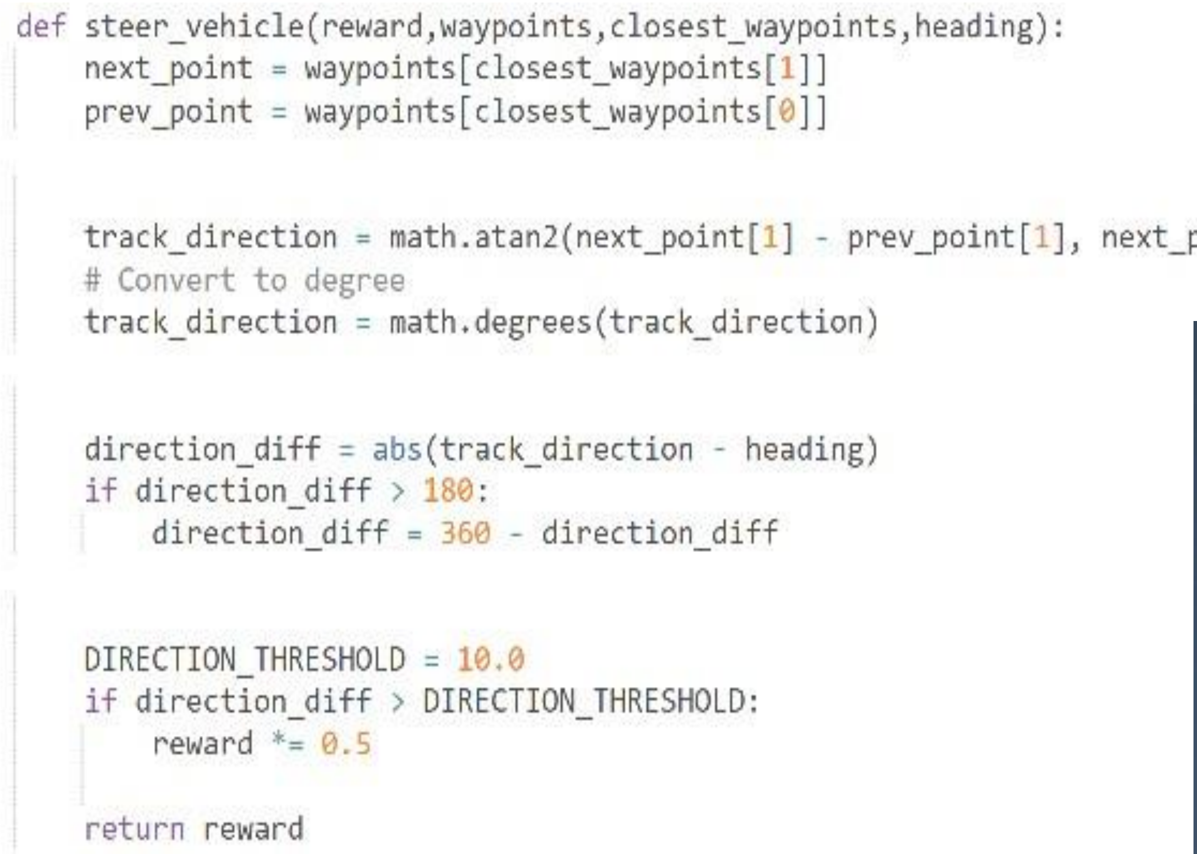
```python
def road_wheels(reward,on_track):
    if not on_track:
        reward = 1e-3
    else:
        reward = 10
    return reward
```

## Center vehicle Reward

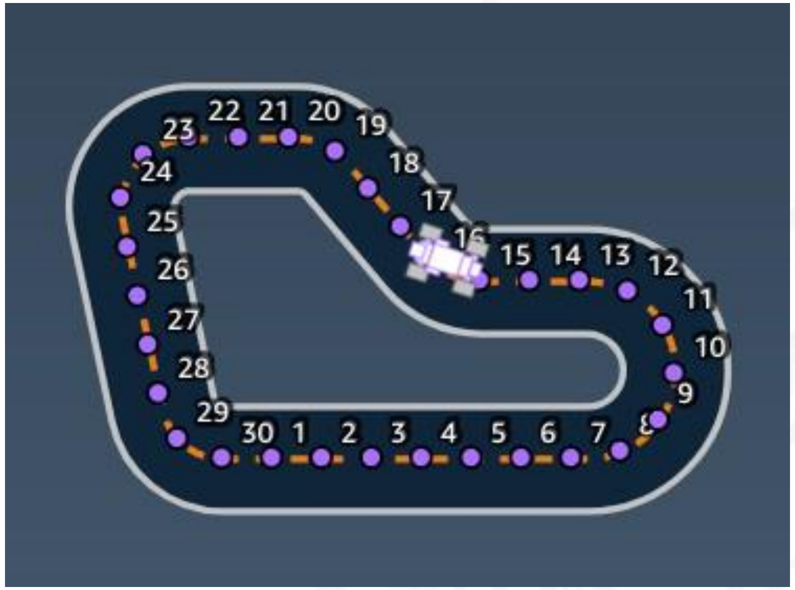


```python
def center_vehicle(reward,track_width,distance_from_center):
    marker_1 = 0.1 * track_width
    # Give higher reward if the car is closer to center line and vice versa
    distance_from_border = abs(0.5*track_width-distance_from_center)

    if distance_from_center <= marker_1:
        reward *= 1.5
    if distance_from_border >0.1:
        reward *=2
    else:
        reward *=0.5
    return reward
```

# Zig-zag steering penalty

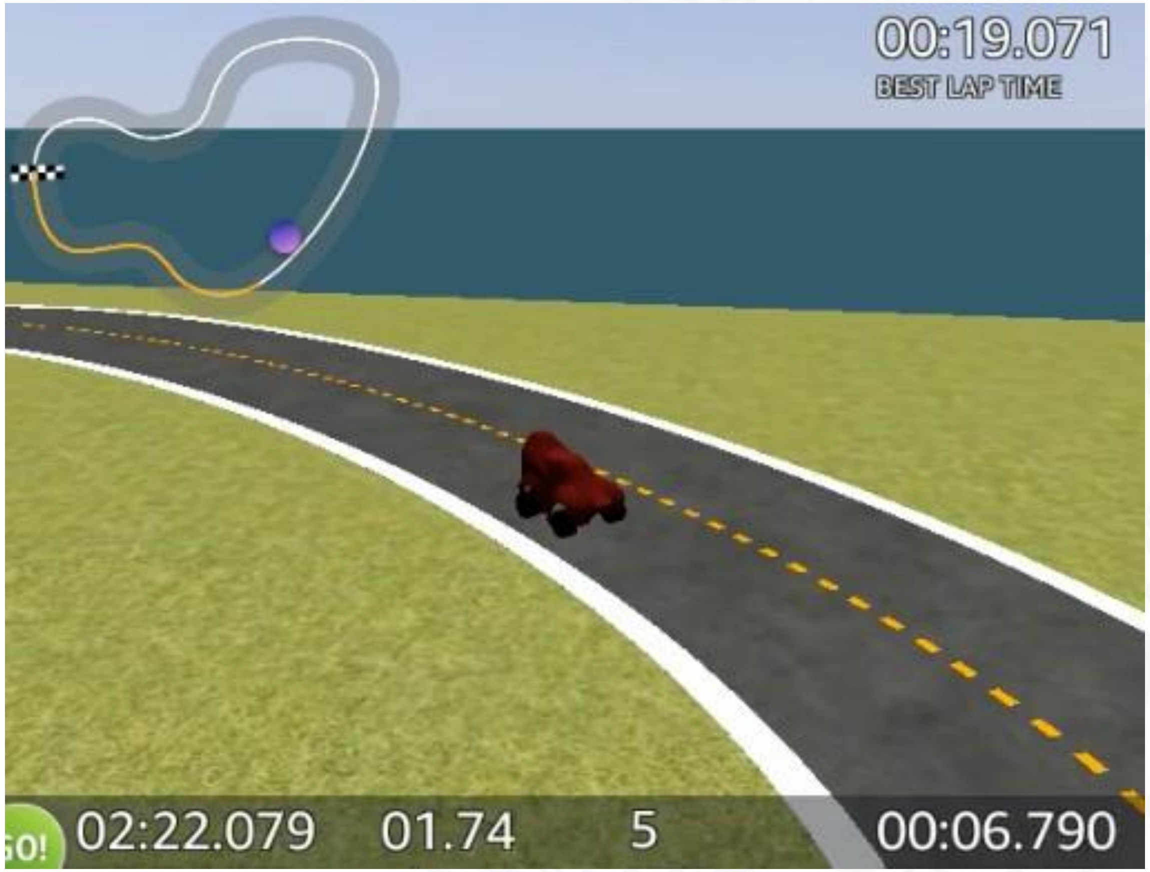


```
def steer_vehicle(reward,waypoints,closest_waypoints,heading):
    next_point = waypoints[closest_waypoints[1]]
    prev_point = waypoints[closest_waypoints[0]]


    track_direction = math.atan2(next_point[1] - prev_point[1], next_point[0] - prev_point[0])
    # Convert to degree
    track_direction = math.degrees(track_direction)


    direction_diff = abs(track_direction - heading)
    if direction_diff > 180:
        direction_diff = 360 - direction_diff


    DIRECTION_THRESHOLD = 10.0
    if direction_diff > DIRECTION_THRESHOLD:
        reward *= 0.5

    return reward
```

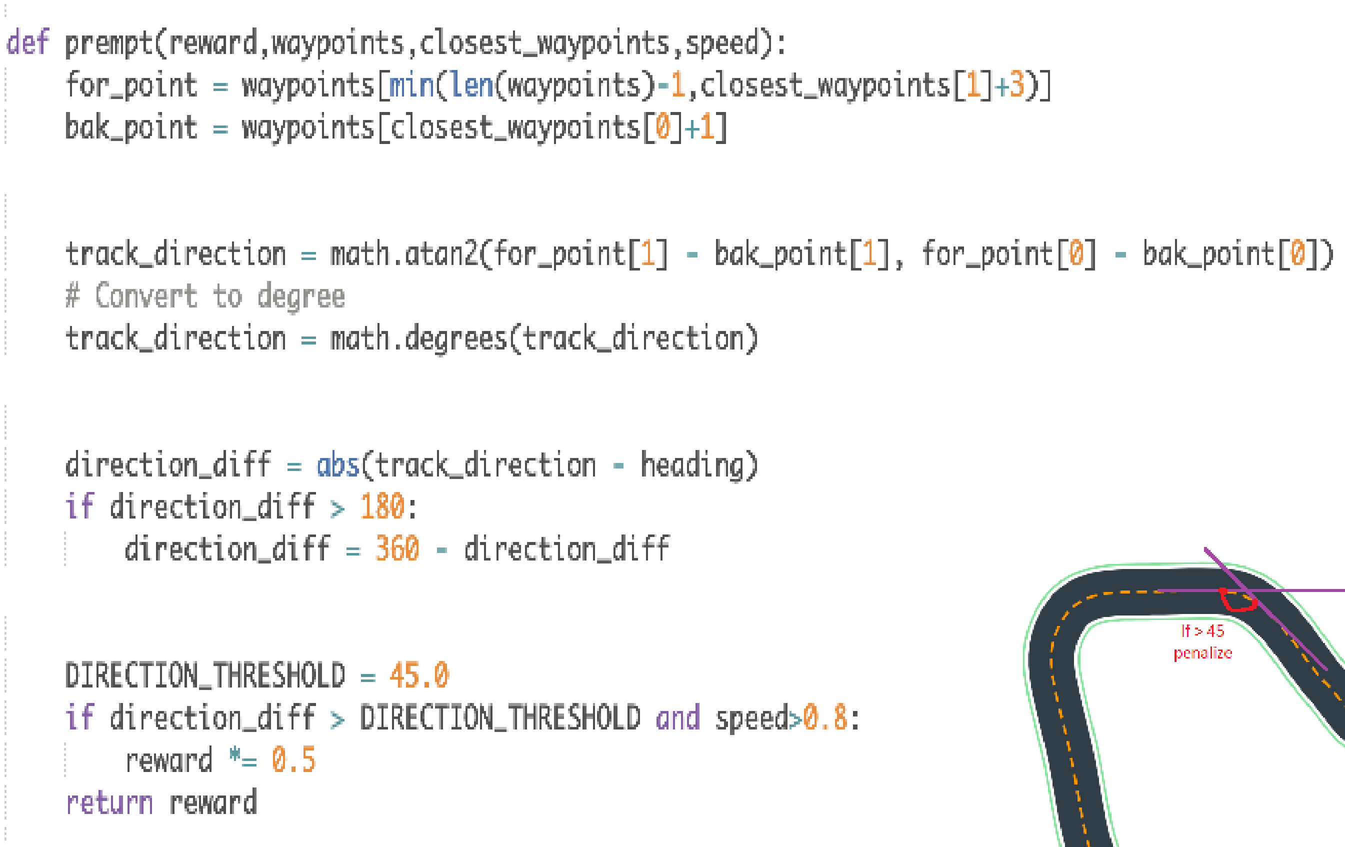


# Low speed penalty

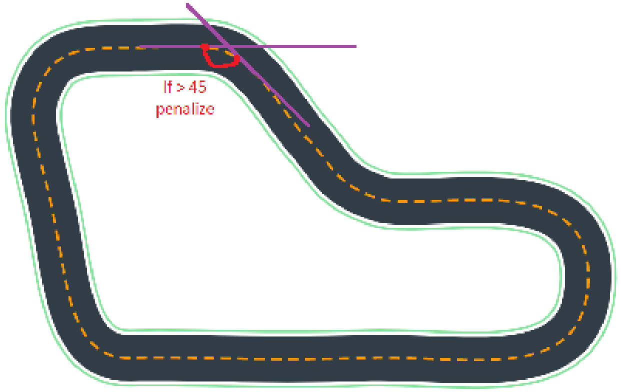
```
def accel(reward,speed):
    speed_fraction = speed/MAX_SPEED
    reward /= (1/(speed_fraction))
    return reward
```



# Corner Overspeed Penalty



```
def prempt(reward,waypoints,closest_waypoints,speed):
    for_point = waypoints[min(len(waypoints)-1,closest_waypoints[1]+3)]
    bak_point = waypoints[closest_waypoints[0]+1]


    track_direction = math.atan2(for_point[1] - bak_point[1], for_point[0] - bak_point[0])
    # Convert to degree
    track_direction = math.degrees(track_direction)


    direction_diff = abs(track_direction - heading)
    if direction_diff > 180:
        direction_diff = 360 - direction_diff


    DIRECTION_THRESHOLD = 45.0
    if direction_diff > DIRECTION_THRESHOLD and speed>0.8:
        reward *= 0.5
    return reward
```



# Reasons for choosing continuous action space.

- For small action spaces , we were able to create a safe model that complete challenging tracks 100% of the time.

- It is better option to incentivize desired behavior for specific points on any challenging track .

- Smoother changes in speed and steering ,accurately depicting real life conditions.

# Disadvantages of choosing continuous action space.

- Takes a big amount of time to train good models when you increase action space.

# Improvements:

- Decrease focus on completion rate on challenging tracks and focus on speed on easier tracks
- Maximum speed could be increased to 4 m/s
- Different steering range, lowering the steering angle could lessen the amount of resets.
- Discrete action space with lower steering range, could be explored to lessen training time.

*Thank you*