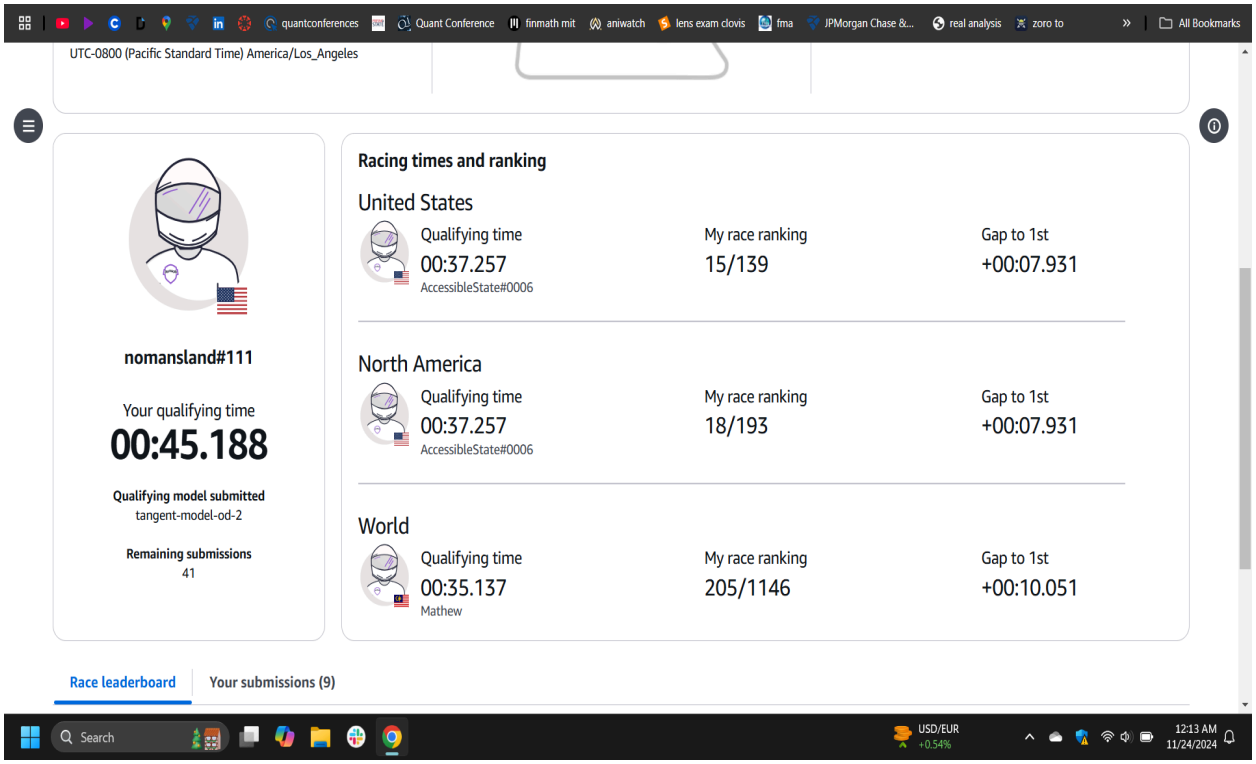


Reinforcement Learning for AWS DeepRacer

1. Introduction

Reinforcement Learning and AWS DeepRacer

When an agent learns to make decisions by interacting with an environment, it is called Reinforcement learning (RL) which is a type of machine learning. In this, the agent takes actions based on the feedback that it has received in the form of rewards or penalties. It also uses this feedback to improve its performance in future decisions. DeepRacer applies RL to autonomous driving, enabling agents, that is cars in this case, to learn how to navigate a track efficiently. It uses a custom reward function that guides their learning process.

Objective

In this lab our main objective is to explore reinforcement learning in AWS DeepRacer by training a series of models using different reward functions. Our goal was to improve the car's lap time and its stability to stay inside the Baja Turnpike track by testing and refining reward functions.

2. Methodology

Steps Taken to Train the Default Model

`Defojmodel` was the default model and it was trained using the "Stay Within Borders" reward function. This function rewards the car for staying on the track, without specifically optimizing for speed or turn control. It provides us with a baseline for performance metrics. It helped us understand it better that how the car behaves without additional modifications.

Descriptions of Each Model

1. **`defojmodel`**: This baseline model used a simple reward function to encourage the car to stay within the track borders. It demonstrated stability but lacked speed optimization, resulting in slow lap times and difficulty navigating sharp turns.
2. **`model2oj`**: This model aimed to build upon the default approach but struggled with sharp turns and straight sections due to conservative exploration and lack of speed incentives. It provided insights into the limitations of a purely stability-focused reward function.
3. **`model2oj-clone`**: A duplicate of `model2oj`, this model confirmed that the lack of exploration and speed optimization led to subpar performance. The lap times remained consistent with `model2oj`, highlighting the need for aggressive modifications.
4. **`tangent-model-od`**: This model introduced tangent-based steering, improving the car's ability to navigate turns effectively. While it reduced lap times significantly compared to earlier models, it still struggled with stability on straights, leading to frequent steering corrections.
5. **`tangent-model-od-2`**: A refinement of `tangent-model-od`, this model aggressively optimized speed incentives for straight sections while retaining effective turn control. It emerged as the top performer, demonstrating balanced speed and stability.
6. **`ojas-best-tangent`**: This model focused on improving stability while maintaining high performance in navigating turns. Though slightly slower than `tangent-model-od-2`, it showcased strong consistency and smoother steering behavior, particularly on straights.

Rationale Behind Reward Function Modifications

As the models progressed, the reward functions were modified to emphasize critical behaviors:

- **Speed Incentives**: Aggressive speed targets were introduced for straight sections to minimize lap times.
- **Turn Control**: Modifications to handle sharp turns by dynamically adjusting speed thresholds and steering guidance.
- **Racing Line Alignment**: Using the centerline as a proxy for an optimal racing line to encourage smoother trajectories.
- **Reduced Steering Adjustments**: Penalties for excessive steering helped prevent zig-zag movements, particularly on straight sections.

Each subsequent model built upon previous iterations to refine behaviors and address specific challenges observed during training.

3. Results

Performance Metrics Comparison

The following table presents the performance metrics of all the models:

Model Name	Qualifying Time	Best Lap Time	Average Lap Time	Status	Observations
defojmodel	2:09.994	2:09.994	2:10.152	Submitted	Stable but slow. Lacked speed incentives and struggled with sharp turns.
model2oj	2:25.465	2:25.465	2:30.465	Submitted	Even slower. Minimal exploration led to difficulty handling complex turns.
model2oj-clone	2:20.594	2:20.594	2:31.085	Submitted	Similar to `model2oj`. Lacked speed optimization and struggled on straights.
tangent-model-od	1:04.332	1:04.332	1:14.093	Submitted	Significant improvement. Tangent-based steering helped handle turns better.
tangent-model-od-2	0:45.188	0:45.188	0:45.974	Submitted	Top performer. Balanced speed and turn control using tangent guidance.
ojas-best-tangent	0:47.328	0:47.328	0:51.907	Submitted	Close second. Strong stability, but slightly slower on straights.

4. Discussion

Impact of Modifications

The reward function modifications significantly impacted the car's performance:

1. **Speed Optimization**: Introducing aggressive speed rewards for straight sections dramatically reduced lap times, as seen in `tangent-model-od-2`.
2. **Turn Handling**: Dynamic steering incentives and speed adjustments for sharp turns improved performance in models like `tangent-model-od`.

3. **Stability Improvements**: Penalizing excessive steering reduced zig-zagging, improving stability in `ojas-best-tangent`.

Challenges Faced

1. **`defojmodel`**: The car was stable but slow, as the reward function lacked incentives for speed. It struggled with navigating sharp turns and wasted time on straights due to lack of dynamic guidance.
2. **`model2oj`**: This model performed worse than the baseline due to overly conservative exploration and minimal speed optimization. It failed to adapt to sharp turns, often drifting off the optimal path.
3. **`model2oj-clone`**: Similar to `model2oj`, this model lacked significant changes and displayed identical weaknesses. It highlighted the need for aggressive reward function modifications to improve both speed and turn handling.
4. **`tangent-model-od`**: The tangent-based steering approach improved turn handling but introduced excessive steering adjustments on straights. These corrections slowed the car, preventing it from achieving consistent lap times.
5. **`tangent-model-od-2`**: Although this model achieved excellent lap times, training revealed difficulty maintaining stability during sharp turns. The speed incentives, while effective on straights, occasionally caused overshooting in tight curves.
6. **`ojas-best-tangent`**: Stability was a key strength of this model, but it sacrificed some speed on straights compared to `tangent-model-od-2`. The car's performance highlighted the trade-off between speed and consistency.

Overcoming Challenges
We followed a mixed approach of adapting speed threshold and reduced steering penalties. The use of tangent based alignment, addressed the previously mentioned challenges effectively. The introduction of racing line alignment in later models further improved performance.

5. Conclusion

Key Learnings

- The reward function plays a crucial role in reinforcement learning for autonomous systems. Small changes can significantly affect performance.
- Balancing speed incentives with stability is critical for achieving faster and more consistent lap times.

Reflection
The assignment highlighted the importance of tailoring reward functions to the specific characteristics of a track. By repeatedly modifying the reward function, it was possible to achieve a significant improvement in lap times and stability, which demonstrated the effectiveness of reinforcement learning in optimizing autonomous behaviors.

Supplements

A. Code Snippets

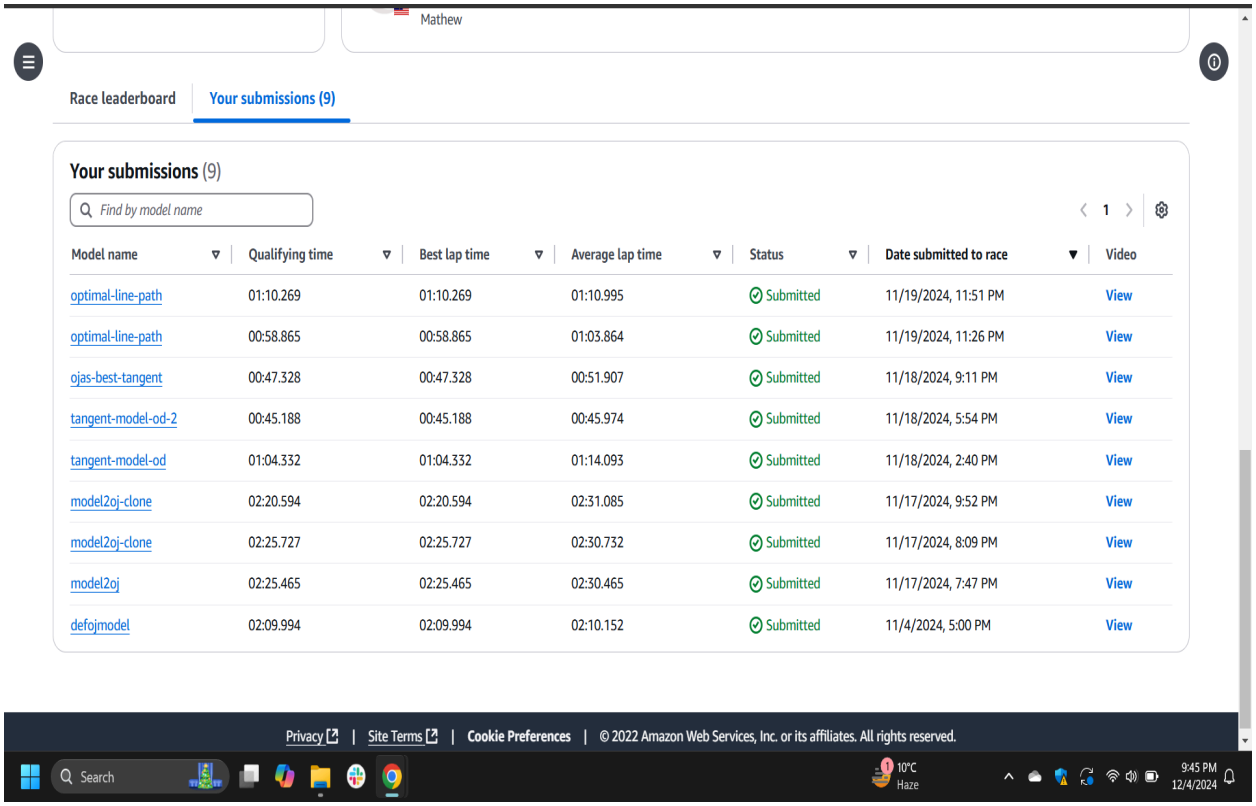
Best model - **`tangent-model-od-2`**

```
import math


def reward_function(params):
    # Read input parameters
    all_wheels_on_track = params['all_wheels_on_track']
    speed = params['speed']
    track_width = params['track_width']
    distance_from_center = params['distance_from_center']
    steering = params['steering_angle']  # Keep steering sign to detect left or right turns
    heading = params['heading']
    waypoints = params['waypoints']
    closest_waypoints = params['closest_waypoints']

    # Base reward
    reward = 1e-3

    if all_wheels_on_track:
        # Calculate center line direction (tangent) based on the closest waypoints
        next_point = waypoints[closest_waypoints[1]]
        prev_point = waypoints[closest_waypoints[0]]
```

```
    center_line_direction = math.degrees(math.atan2(next_point[1] - prev_point[1], next_point[0] - prev_point[0]))

    # Calculate the difference between the car's heading and the center line direction
    direction_diff = abs(center_line_direction - heading)

    # Identify if the section is a straight or turn based on direction difference between waypoints
    is_straight = direction_diff < 5.0  # 5 degrees threshold for considering a section straight

    # Minimize side-to-side movement on straight sections by rewarding minimal steering
    if is_straight:
        # Reward for keeping minimal steering on straight sections
        if abs(steering) < 5:  # Small steering adjustment allowed on straights
            reward += 1.0
        else:
            reward *= 0.7  # Penalize large steering on straight sections
    else:
        # For turns, use the center line slope to determine correct turn direction
        if center_line_direction - heading > 0:
            # Center line indicates a right turn
            if steering < 0:  # Negative steering means turning right
                reward += 1.0
            else:
```

```
                reward *= 0.5  # Penalize incorrect left turn
        else:
            # Center line indicates a left turn
            if steering > 0:  # Positive steering means turning left
                reward += 1.0
            else:
                reward *= 0.5  # Penalize incorrect right turn

    # Progress reward for steady movement along the track
    progress = params['progress']
    reward += progress * 0.1

    # Speed reward with control for turns
    SPEED_THRESHOLD = 3.0 if is_straight else 2.0
    if speed >= SPEED_THRESHOLD:
        reward += 0.5

    # Penalize for being too close to the track edges
    if distance_from_center > 0.4 * track_width:
        reward *= 0.5

return float(reward)
```

B. Screenshots


Mathew
Race leaderboard
Your submissions (9)
Your submissions (9)
Find by model name
1
Model name
Qualifying time
Best lap time
Average lap time
Status
Date submitted to race
Video
optimal-line-path 01:10.269 01:10.269 01:10.995 Submitted 11/19/2024, 11:51 PM View
optimal-line-path 00:58.865 00:58.865 01:03.864 Submitted 11/19/2024, 11:26 PM View
ojas-best-tangent 00:47.328 00:47.328 00:51.907 Submitted 11/18/2024, 9:11 PM View
tangent-model-od-2 00:45.188 00:45.188 00:45.974 Submitted 11/18/2024, 5:54 PM View
tangent-model-od 01:04.332 01:04.332 01:14.093 Submitted 11/18/2024, 2:40 PM View
model2oj-clone 02:20.594 02:20.594 02:31.085 Submitted 11/17/2024, 9:52 PM View
model2oj-clone 02:25.727 02:25.727 02:30.732 Submitted 11/17/2024, 8:09 PM View
model2oj 02:25.465 02:25.465 02:30.465 Submitted 11/17/2024, 7:47 PM View
defojmodel 02:09.994 02:09.994 02:10.152 Submitted 11/4/2024, 5:00 PM View
Privacy | Site Terms | Cookie Preferences | © 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.
Search
10°C Haze
9:45 PM 12/4/2024