AWS DeepRacer Manual



**By: Ishaq Ahmed**

***Table of Contents:***

1. *Introduction to AWS DeepRacer*
2. *Setting up DeepRacer*
3. *Creating your first DeepRacer Model*
4. *Training your Model*
5. *Evaluating and Tuning your Model*
6. *Resources from community*
7. *Conclusion*
8. *Code provided by Ishaq.*

**1. Introduction to AWS DeepRacer**

AWS DeepRacer, the autonomous racing car platform driven by reinforcement learning (RL) is a place where you will learn about the basic concepts, such as how RL is used to train models for autonomous driving, and an introduction to the tools and services involved.

**2. Setting Up AWS for DeepRacer**

Before starting, ensure you have an active AWS account.

**3. Creating Your First DeepRacer Model**

In this section, you will be guided through navigating the AWS DeepRacer console and creating your first training model.

*3.1. Search for DeepRacer on the home console.*

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

*3.2. Navigating the AWS DeepRacer Console*

Once this page is loaded, click on **Create Model** and it will bring you to this page:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Note: Step 1 consists of a crash course on reinforcement learning and how it is used to optimize your vehicle/agent, so I highly recommend reading it if you’re a beginner.

*3.3. Creating a Training Job*A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

- This page consists of naming your model and giving a description of your first/current model, what has changed or what is new.

*3.4. Configuring the environment (track selection, action space, etc.)*

* You can select the track that you want your agent to train in based on the conditions you would like to select. It could be that you want your agent to go around the track fast or make sharp corners or even both.

*A screenshot of a computer

Description automatically generated*

* There are multiple tracks available, and you can find more when you click on **View more race tracks**.
* In Step 2 you can choose the race type and algorithms that are available to you. In this manual I will be doing a time trial and using the PPO state for the training algorithm.

A screenshot of a computer

Description automatically generated

* The hyperparameters are going to be set to the default choices but you can tweak it if you would like. A screenshot of a computer

  Description automatically generated
* Next step is to select the action space. I will be choosing the Continuous action space as it allows the agent to select an action available from the range of values for each state through each waypoint of a track.

A screenshot of a computer

Description automatically generated

* Now we define the continuous action space in which we choose the steering angle and the speed of the agent. I suggest leaving the steering angle at 30∘ so for full range of steering action.
* You may tweak the speed as that is going to determine how fast the agent can complete one lap in the training and evaluation.

A screenshot of a computer

Description automatically generated

**Note:** Speed does affect the training and evaluation so make sure that the reward function is set to speed up at the straights and slow down at the turns.

* Now we choose the agent shell and sensor configurationA screenshot of a computer program

  Description automatically generated

*3.5. Defining the reward function:*

* Now we will choose the reward function for the agent.

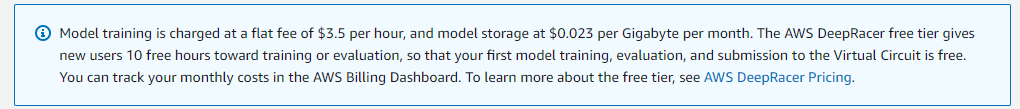
A screenshot of a computer program

Description automatically generated

* Remember to always validate the code you have written before executing the training.
* In this model we will use the example code provided.
* Define your stop condition. You can train your model for a minimum of 5 to 1440 minutes but remember you will be charged a fee of $3.5/hour to train your model.

A screenshot of a computer screen

Description automatically generated



- Proceed with creating model.

**4. Training Your Model**

- The model is initializing

**A screenshot of a computer

Description automatically generated**

*4.1. Setting Hyperparameters*

- These are the current hyperparameters that are set for this model. You may change them once the model training is complete or could change them before

A screenshot of a computer

Description automatically generated

*4.2. Best practices for training*

* You can watch the simulation video stream that is displayed while training to see what the agent is doing.

A screenshot of a graph

Description automatically generated

* If there are too many peaks and valleys in the graph, you can stop the training and tweak your code to have a steady growth in your reward, track completion and your evaluation.

**5. Evaluating and Tuning Your Model**

5.1. Evaluating performance

* Once done training is complete you can evaluate your model but placing it in another track to see how well it performs. Depending on the code and conditions provided by the training track should perform as well as it should on a test track.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

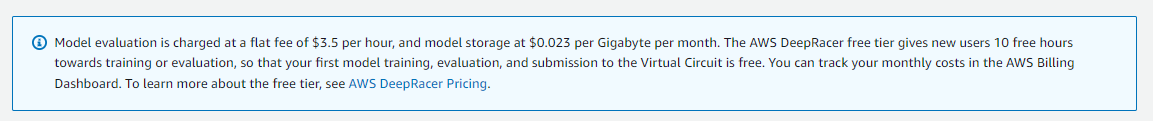
Description automatically generated

A screenshot of a computer

Description automatically generated

* You may choose the track you would like to choose for the Evaluation criteria.

**Remember:**



**6. Resources and community support**

In this section, you will find additional insights on resources and community support details that will help on your model building.

*6.1. AWS DeepRacer Developer Guide*

* This provides a detailed documentation of how to train and evaluate the model with more information on the parameters and concepts that can enhance your understanding of how the model works. This [link](https://docs.aws.amazon.com/deepracer/latest/developerguide/what-is-deepracer.html) takes you to the page for more information.

*6.2. Discord community*

* The discord community is a place where people come together to discuss their own models and is a place where you can ask questions regarding your model and gather help from them on the code that you would like to update or change altogether. Click this [link](https://discord.gg/RPm5wtZ2) to join their discord community.

*6.3. GitHub*

* The AWS DeepRacer community has a GitHub repository that contains all the data of the racetracks and certain data that you may use for your code. Click this [link](https://github.com/aws-deepracer-community) to view their community.

**7. Conclusion**

In summary, building and optimizing an AWS DeepRacer model is a rewarding process that combines practical machine learning with firsthand experimentation. Throughout this journey, we learned how reinforcement learning can be implemented to autonomous driving, from setting up the environment to tuning hyperparameters and refining reward functions. While there may be challenges along the way, such as tweaking the model to perform better on various tracks, the experience deepens your understanding of AI and its practical applications. Keep exploring, testing, and learning—each iteration brings you closer to mastering AWS DeepRacer.

**8. Code provided by Ishaq**

*8.1. Trial 1*

A screenshot of a computer program

Description automatically generated

*8.2. Trail 2*

*A screenshot of a computer program

Description automatically generated*

*8.3 Tral 3*

Implementation of waypoints

A screenshot of a computer program

Description automatically generated

...Continues to the next page…

A screenshot of a computer program

Description automatically generated

*8.4. Trail 4*

Best code with waypoints. Can implement waypoints for other race tracks and it follows the path fast. This [link](https://www.linkedin.com/pulse/aws-deepracer-free-student-workshop-run-faster-using-your-cheuk-lam/?published=t) can provide the steps on how to install the Jupyter notebooks to view all the tracks and waypoints.

A screenshot of a computer program

Description automatically generated

…Continues into the next page…

A screenshot of a computer program

Description automatically generated

*8.5. MK1*

First part of the final code that can be done through reinforcement learning without physical waypoints.

A screenshot of a computer program

Description automatically generated

*8.6. MK2*

Final code that can go around the track 100%. Tweaked the speed reward that was not working right in MK1.

A screenshot of a computer program

Description automatically generated

The reward graph for this code:

A screenshot of a computer

Description automatically generated

**Thank you!**