



AWS DeepRacer: Get hands-on with machine learning



Agenda

- Introducing AWS DeepRacer
- Introduction to machine learning
- Reinforcement learning key terms
- The reward function
- AWS DeepRacer ML architecture
- Demo

Introducing AWS DeepRacer



**How can we put
machine learning in the
hands of all developers?**

Literally



A closer look

- 1:18 4WD scale car
- Intel Atom processor
- Intel distribution of OpenVINO toolkit
- Front-facing camera (4 megapixels)
- System memory: 4 GB RAM
- 802.11ac Wi-Fi
- Ubuntu 20.04 Focal Fossa
- ROS 2 Foxy Fitzroy

AWS DeepRacer Evo Expansion Pack

- Second front-facing camera (stereo cameras)
- 360-degree, 12-meter scanning radius Lidar sensor



OpenVINO™



Get racing...



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Get racing...



3D racing
simulator

Get racing...



3D racing
simulator



AWS DeepRacer
League

Get racing...



3D racing
simulator



AWS DeepRacer
League



Community
races

Get racing...



3D racing
simulator



AWS DeepRacer
League



Community
races



AWS DeepRacer

Introduction to machine learning

The broader context

Artificial
intelligence

building algorithms
which can take and
process information to
inform future decisions

Machine learning

teaching an algorithm how
to learn without explicitly
being programmed to do so

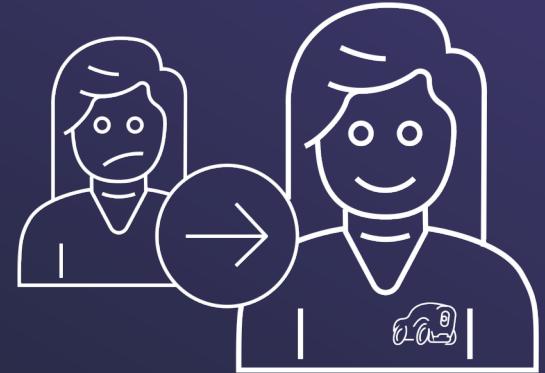
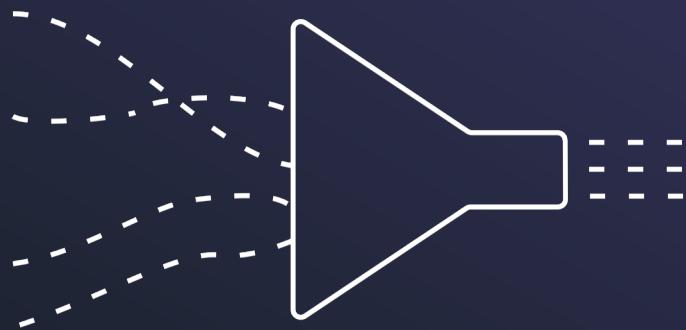
Machine learning model

Supervised
learning

Unsupervised
learning

Reinforcement
learning

ML overview



Supervised

Example-driven training;
every
piece of data has a
corresponding label

Unsupervised

No labels for
training data; useful for
clustering like data,
discovering patterns for
generative AI

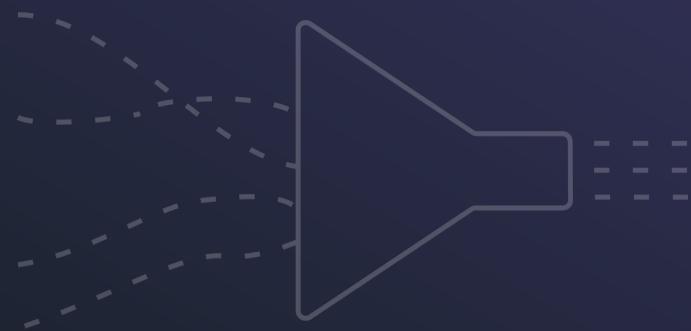
Reinforcement

Learns through
consequences of actions
in a specific
environment

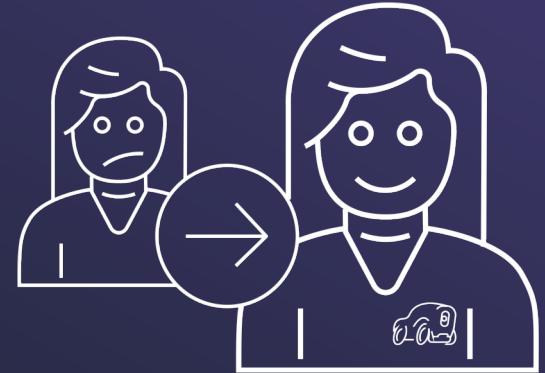
ML overview



Supervised
Example-driven training;
every
piece of data has a
corresponding label

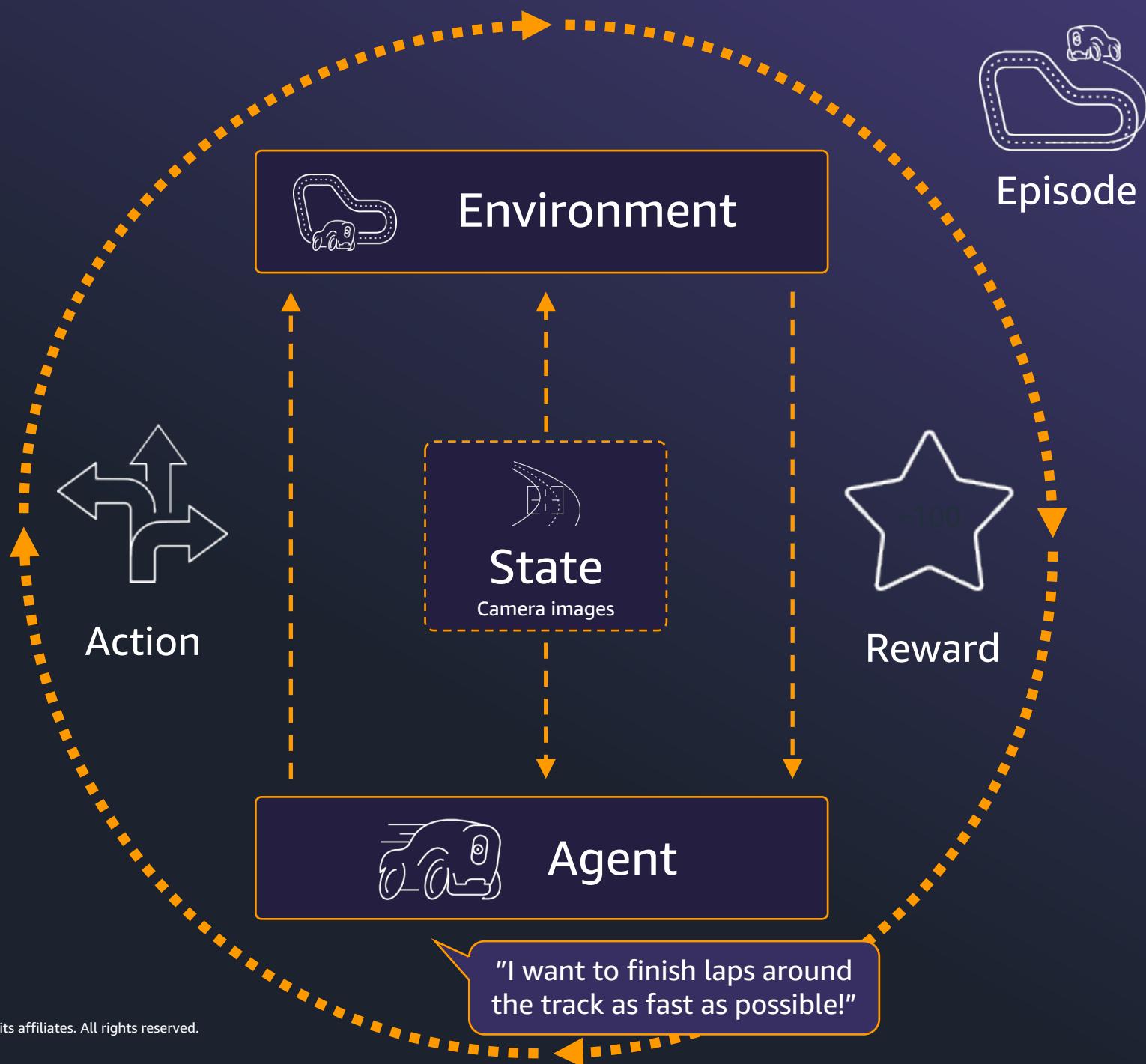


Unsupervised
No labels for
training data; useful for
clustering like data,
discovering patterns for
generative AI



Reinforcement
Learns through
consequences of actions
in a specific
environment

Reinforcement learning key terms



The reward function

The reward function

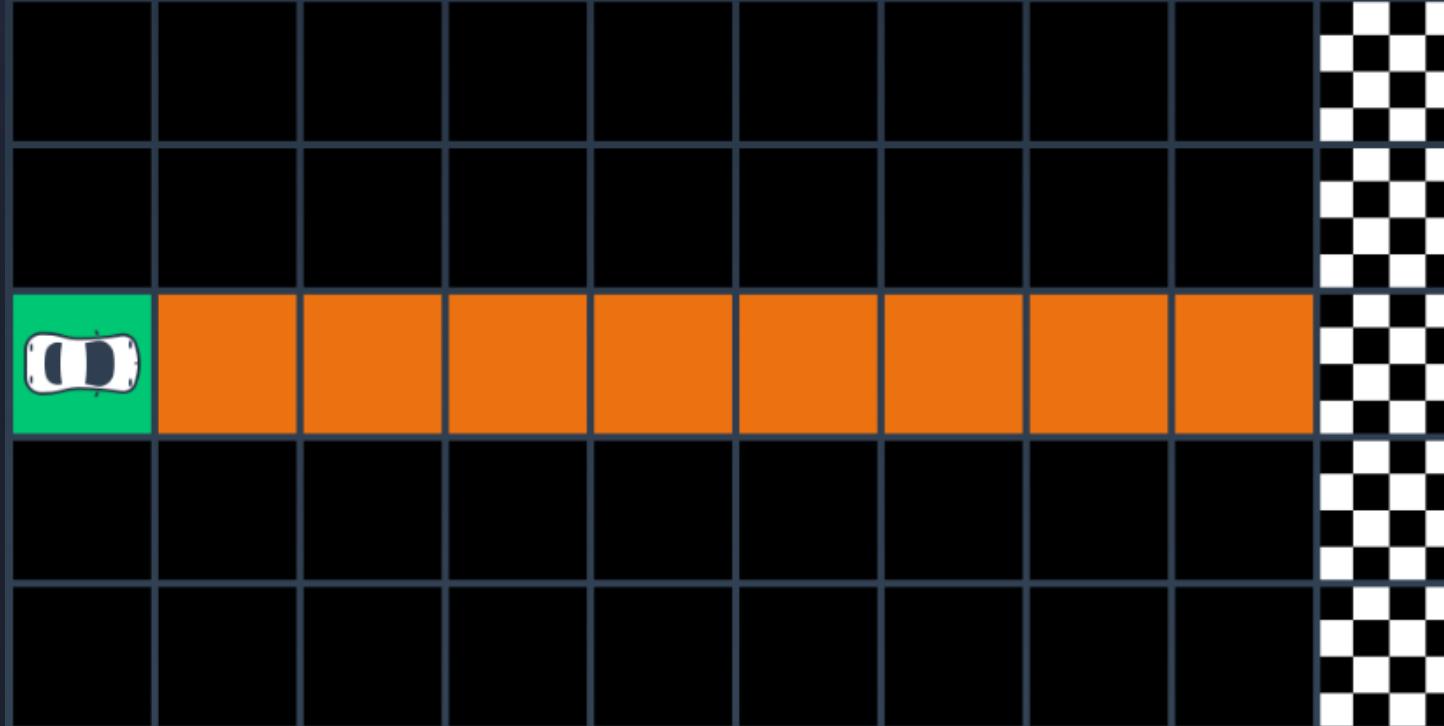


The reward function incentivizes particular behaviors and is at the core of reinforcement learning

The reward function: Straight track race



Agent

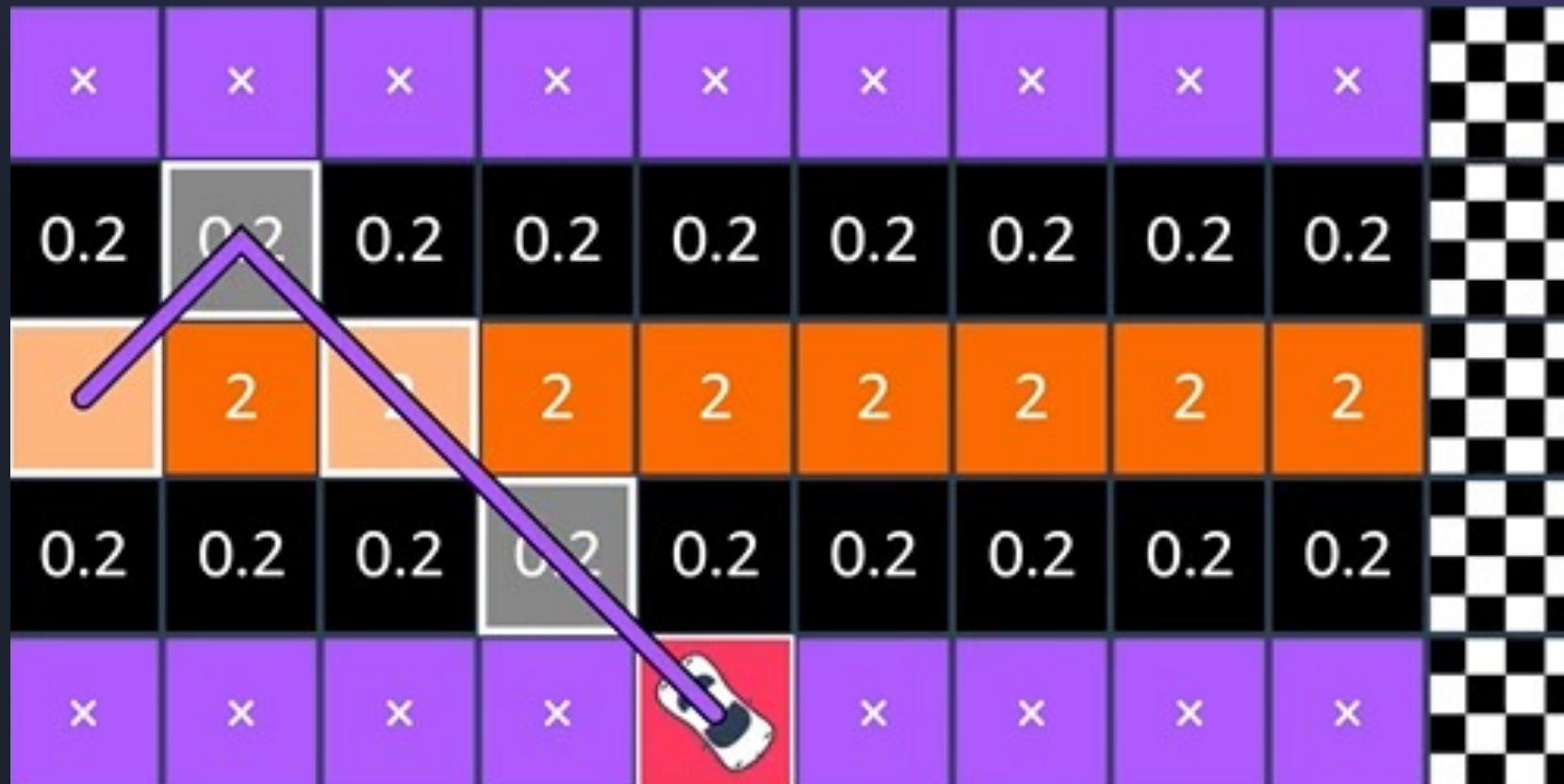


Goal

Rewards that incentivize center-line driving

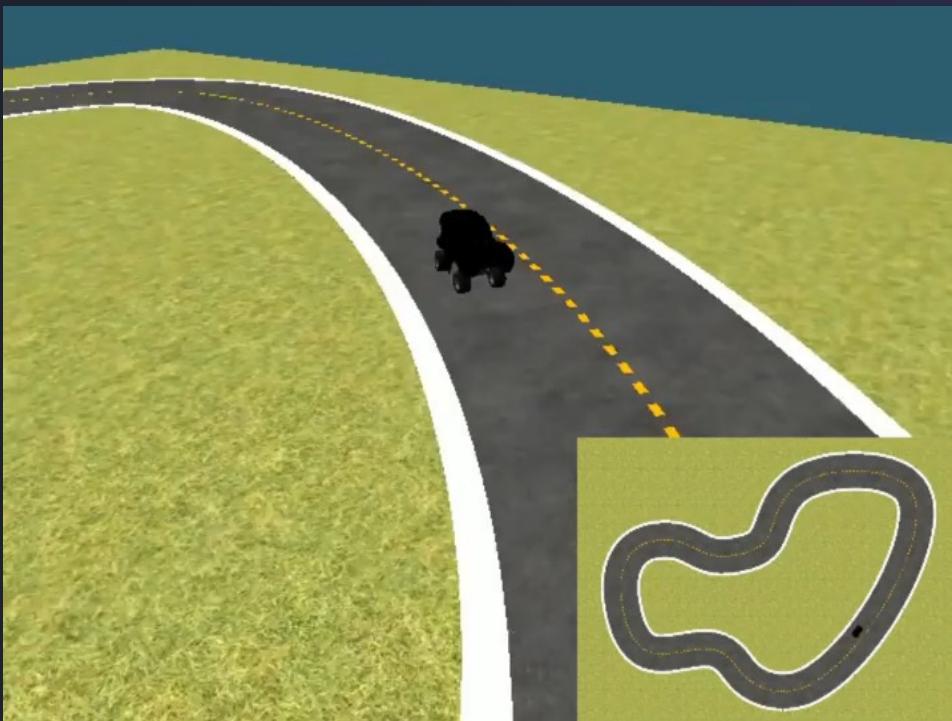
x	x	x	x	x	x	x	x	x	x	checkered
0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	checkered
car icon	2	2	2	2	2	2	2	2	2	checkered
0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	checkered
x	x	x	x	x	x	x	x	x	x	checkered

Exploration versus exploitation

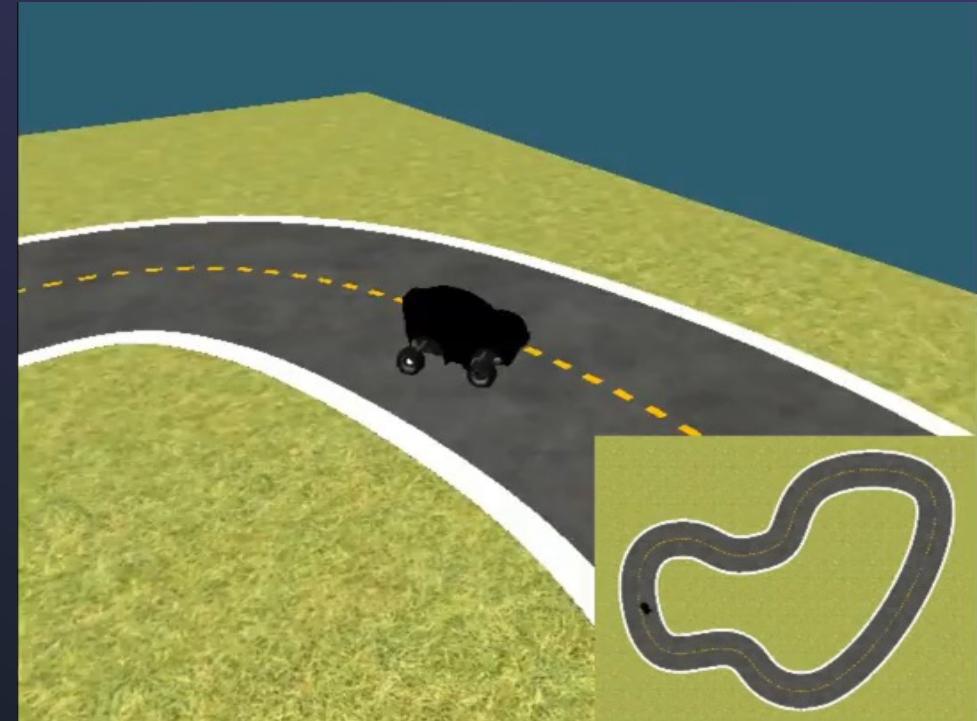


Exploration versus exploitation

Exploration



Exploitation



Convergence



Programming your own reward function

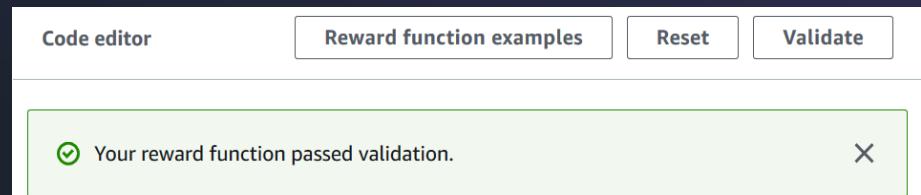
The screenshot shows the AWS Lambda console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, and 'Resource Groups' dropdown. Below the header, the title 'Reward function' is displayed with a 'Info' link. A descriptive text block explains that the reward function provides immediate feedback (score or penalty) when a vehicle moves from one position to another, aiming to encourage quick movement along the track. Below this is a 'Code editor' section containing a Python script for a reward function. The script uses parameters like 'track_width' and 'distance_from_center' to calculate rewards based on proximity to a center line. It includes markers at 0.1, 0.25, and 0.5 times the track width, with higher rewards for staying closer to the center. The code editor interface includes tabs for 'Code editor', 'Reward function examples', 'Reset', and 'Validate'. Two red circular markers are placed above the code editor area, connected by a red line to the validation message on the right.

```
1 def reward_function(params):
2     ...
3     # Example of rewarding the agent to follow center line
4     ...
5
6     # Read input parameters
7     track_width = params['track_width']
8     distance_from_center = params['distance_from_center']
9
10    # Calculate 3 markers that are at varying distances away from the center line
11    marker_1 = 0.1 * track_width
12    marker_2 = 0.25 * track_width
13    marker_3 = 0.5 * track_width
14
15    # Give higher reward if the car is closer to center line and vice versa
16    if distance_from_center <= marker_1:
17        reward = 1.0
18    elif distance_from_center <= marker_2:
19        reward = 0.5
20    elif distance_from_center <= marker_3:
21        reward = 0.1
22    else:
23        reward = 1e-3 # likely crashed/ close to off track
24
25    return float(reward)
```

Code editor: Python 3 syntax

Three example reward functions

Code validation via Lambda



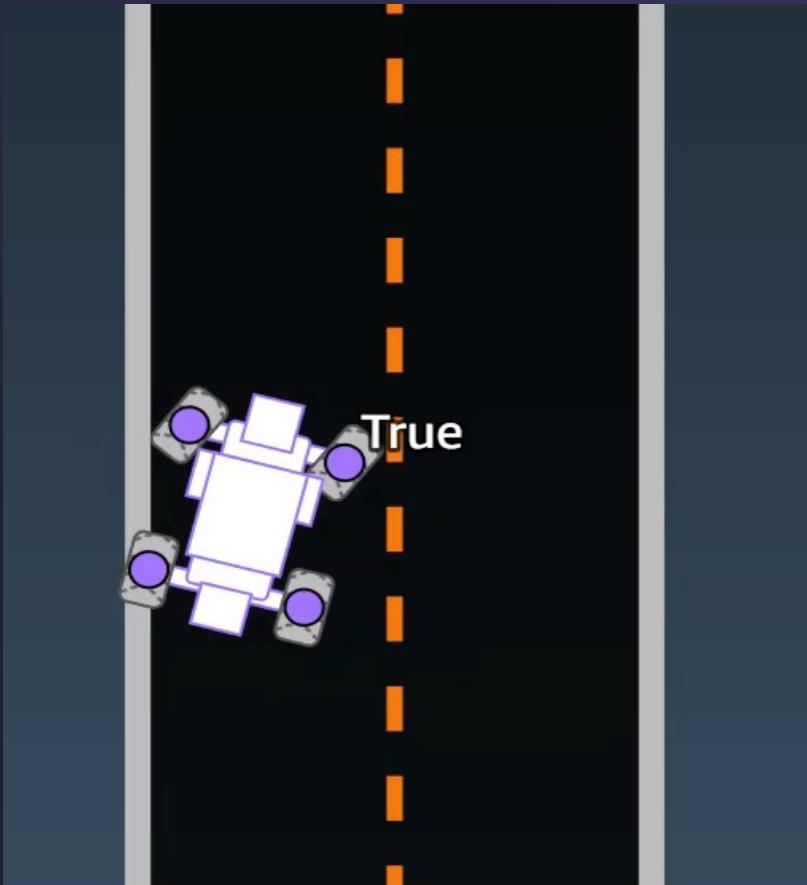
Reward function input parameters

```
{  
    "all_wheels_on_track": Boolean,                      # flag to indicate if the agent is on the track  
    "x": float,                                         # agent's x-coordinate in meters  
    "y": float,                                         # agent's y-coordinate in meters  
    "closest_objects": [int, int],                       # zero-based indices of the two closest objects to the agent's current position of (x, y).  
    "closest_waypoints": [int, int],                      # indices of the two nearest waypoints.  
    "distance_from_center": float,                       # distance in meters from the track center  
    "is_crashed": Boolean,                               # Boolean flag to indicate whether the agent has crashed.  
    "is_left_of_center": Boolean,                         # Flag to indicate if the agent is on the left side to the track center or not.  
    "is_offtrack": Boolean,                             # Boolean flag to indicate whether the agent has gone off track.  
    "is_reversed": Boolean,                            # flag to indicate if the agent is driving clockwise (True) or counter clockwise (False).  
    "heading": float,                                    # agent's yaw in degrees  
    "objects_distance": [float, ],                      # list of the objects' distances in meters between 0 and track_length in relation to the starting line.  
    "objects_heading": [float, ],                        # list of the objects' headings in degrees between -180 and 180.  
    "objects_left_of_center": [Boolean, ],               # list of Boolean flags indicating whether elements' objects are left of the center (True) or not (False).  
    "objects_location": [(float, float),],                # list of object locations [(x,y), ...].  
    "objects_speed": [float, ],                          # list of the objects' speeds in meters per second.  
    "progress": float,                                   # percentage of track completed  
    "speed": float,                                     # agent's speed in meters per second (m/s)  
    "steering_angle": float,                            # agent's steering angle in degrees  
    "steps": int,                                       # number steps completed  
    "track_length": float,                             # track length in meters.  
    "track_width": float,                            # width of the track  
    "waypoints": [(float, float), ]                     # list of (x,y) as milestones along the track center  
}
```

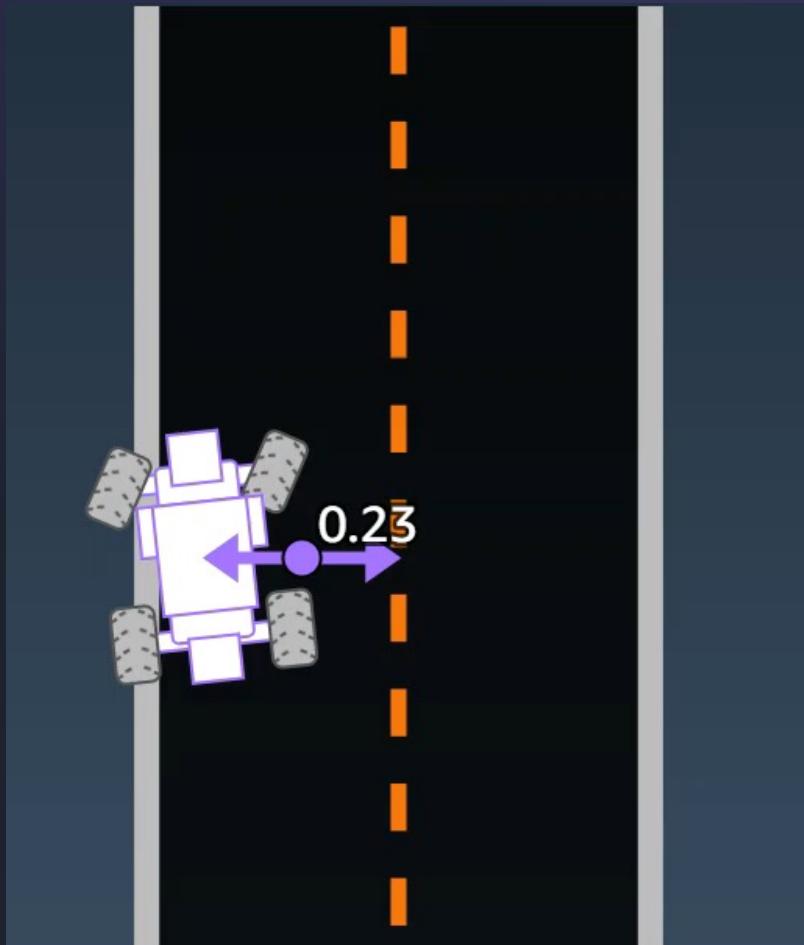
Example parameter: heading



Example parameter: `all_wheels_on_track`



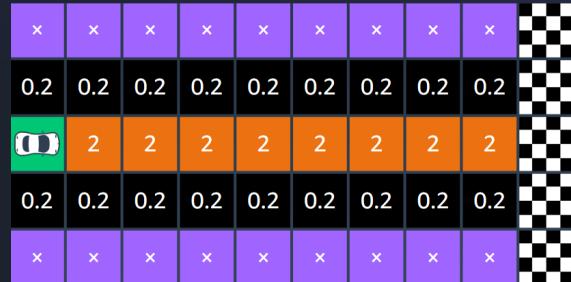
Example parameter: *distance_from_center*



Rewards that incentivize center-line driving

x	x	x	x	x	x	x	x	x	x	checkered
0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	checkered
car icon	2	2	2	2	2	2	2	2	2	checkered
0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	checkered
x	x	x	x	x	x	x	x	x	x	checkered

Rewards that incentivize center-line driving



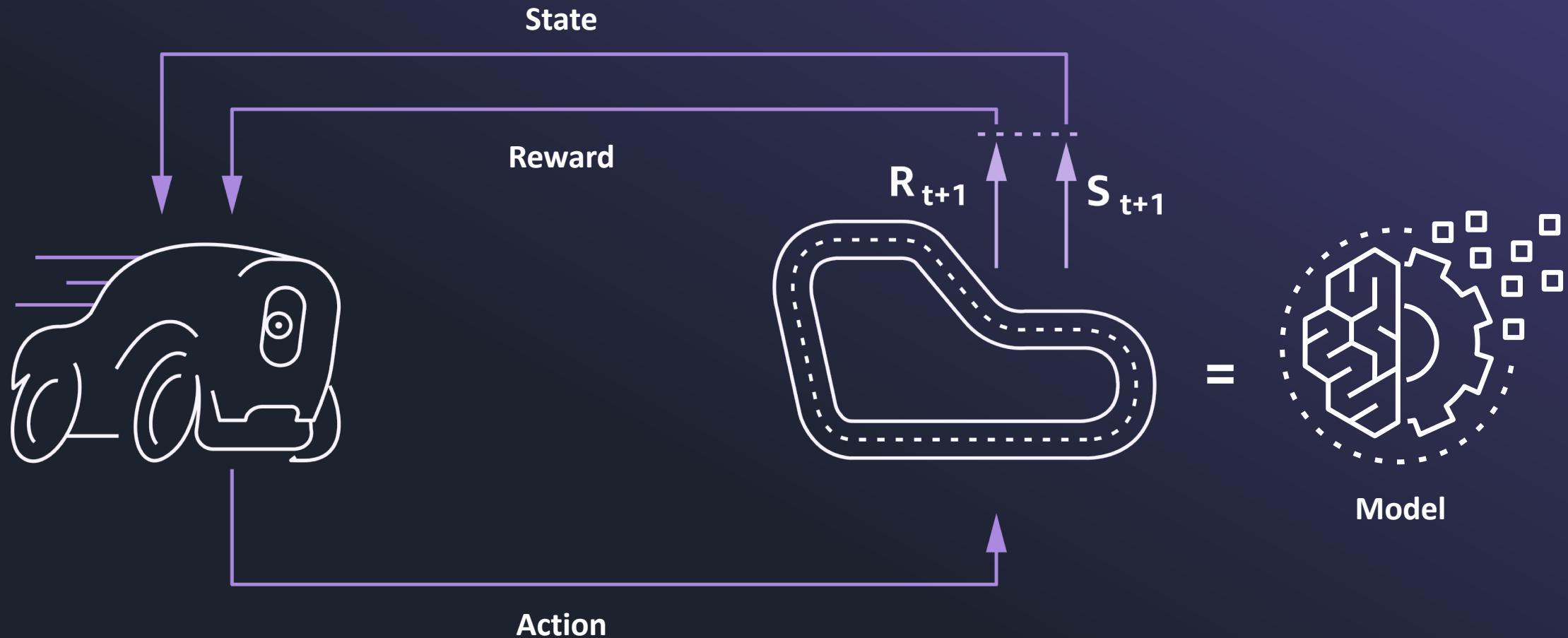
```
def reward_function(params):  
  
    track_width = params['track_width']  
    distance_from_center = params['distance_from_center']  
  
    marker_1 = 0.1 * track_width  
    marker_2 = 0.5 * track_width  
  
    if distance_from_center <= marker_1:  
        reward = 2.0 # Close to center line  
    elif distance_from_center <= marker_2:  
        reward = 0.2 # Further from center line  
    else:  
        reward = 1e-3 # Off track  
  
    return float(reward)
```

Sample reward functions

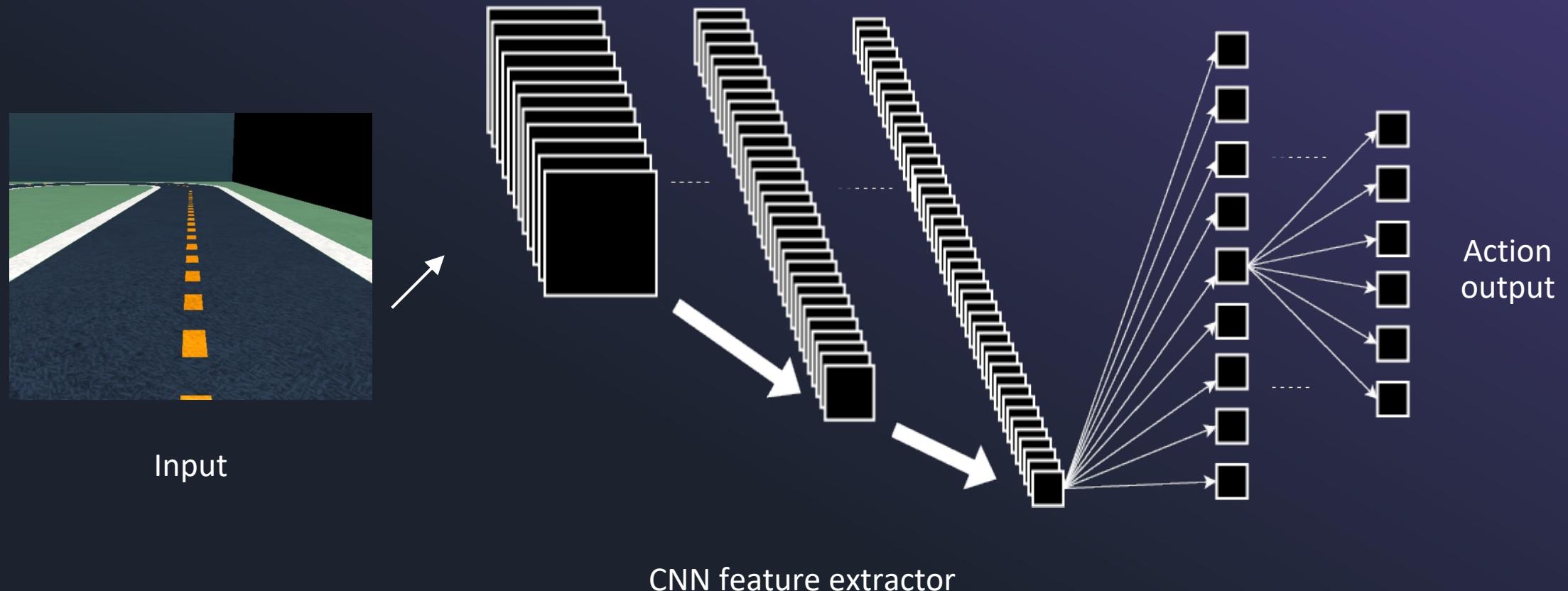
```
1 def reward_function(params):
2     """
3         Example of rewarding the agent to stay inside the two borders of the track
4     """
5
6     # Read input parameters
7     all_wheels_on_track = params['all_wheels_on_track']
8     distance_from_center = params['distance_from_center']
9     track_width = params['track_width']
10
11    # Give a very low reward by default
12    reward = 1e-3
13
14    # Give a high reward if no wheels go off the track and
15    # the agent is somewhere in between the track borders
16    if all_wheels_on_track and (0.5*track_width - distance_from_center) >= 0.05:
17        reward = 1.0
18
19    # Always return a float value
20    return float(reward)
```

AWS DeepRacer ML architecture

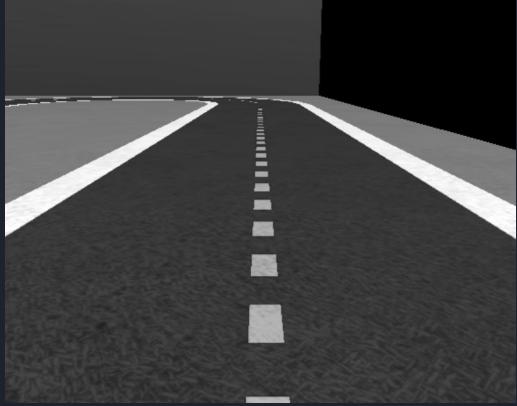
How does the learning (training) happen?



AWS DeepRacer neural network architecture

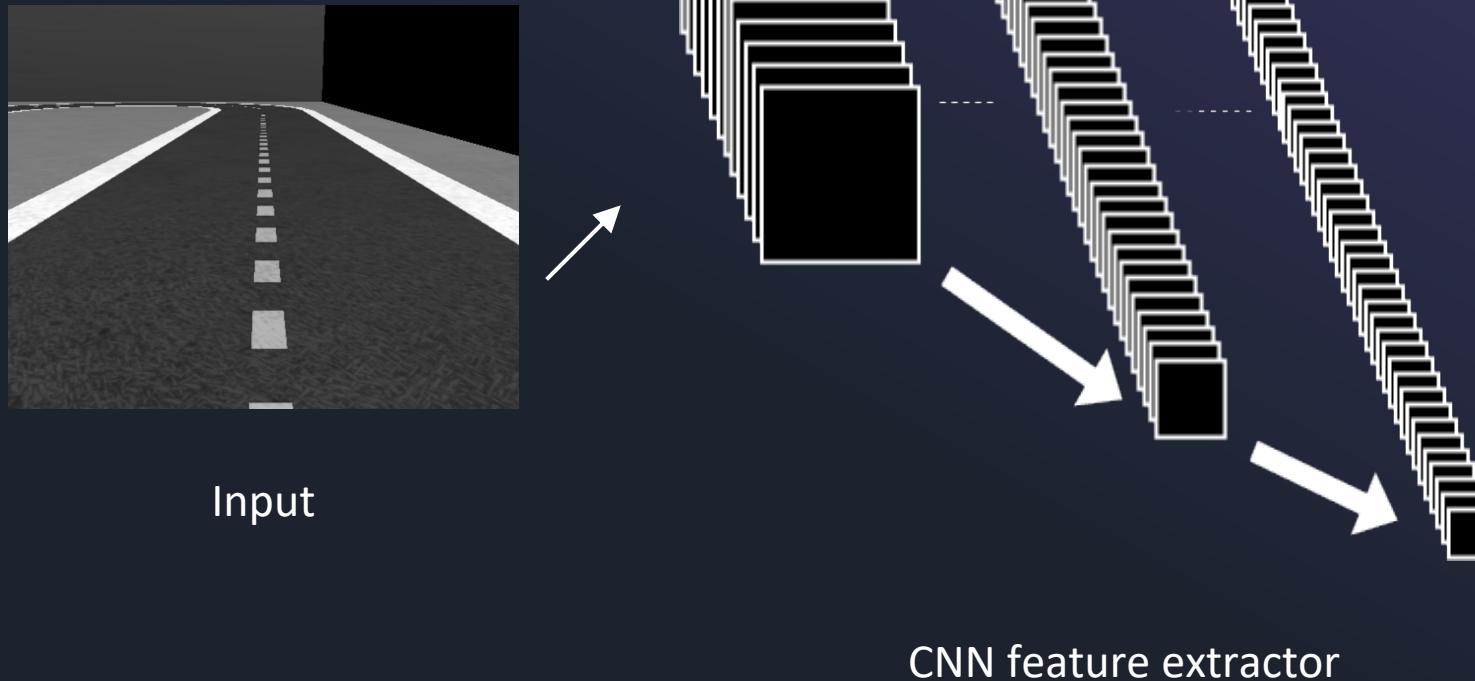


AWS DeepRacer neural network architecture

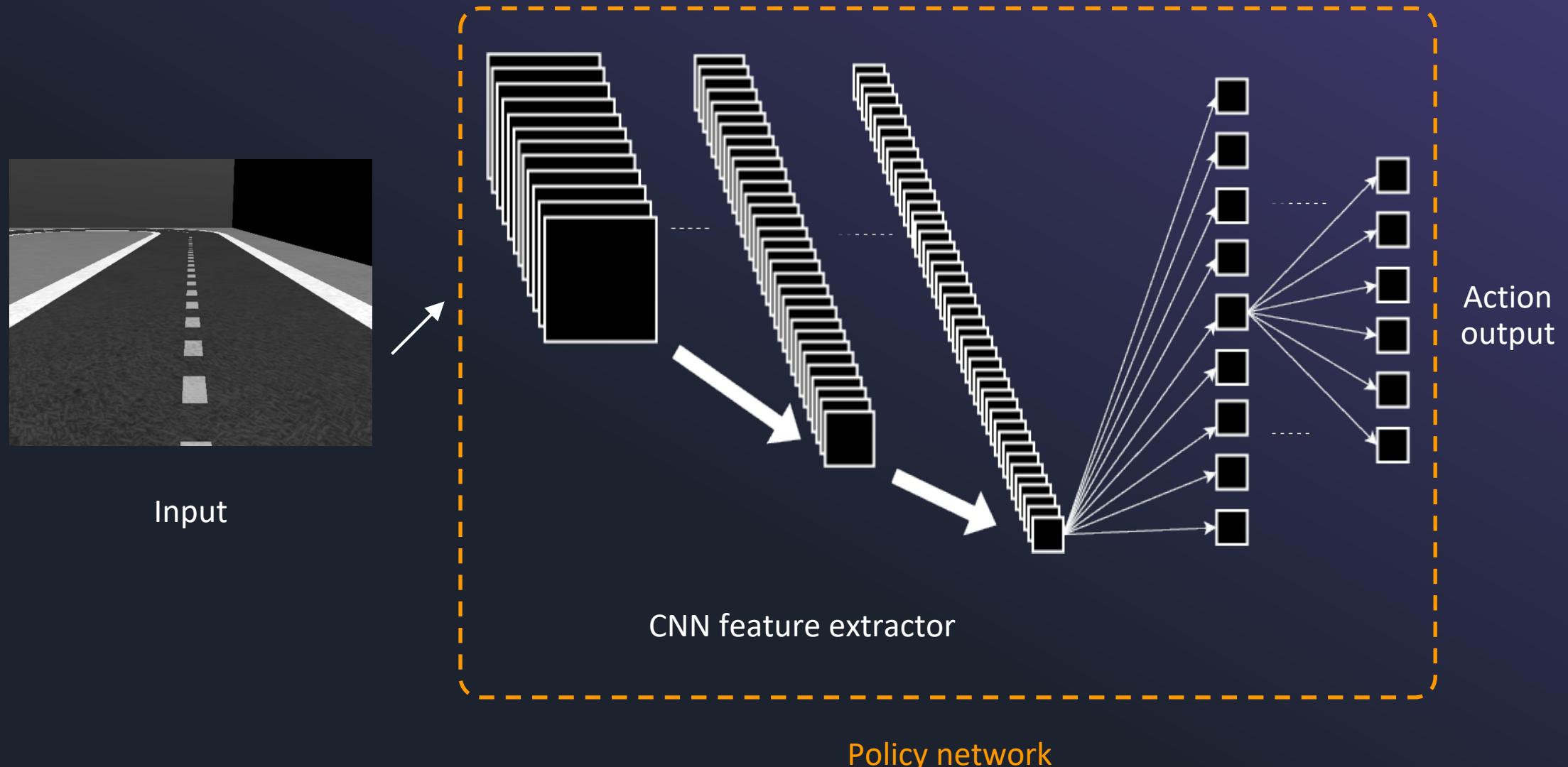


Input

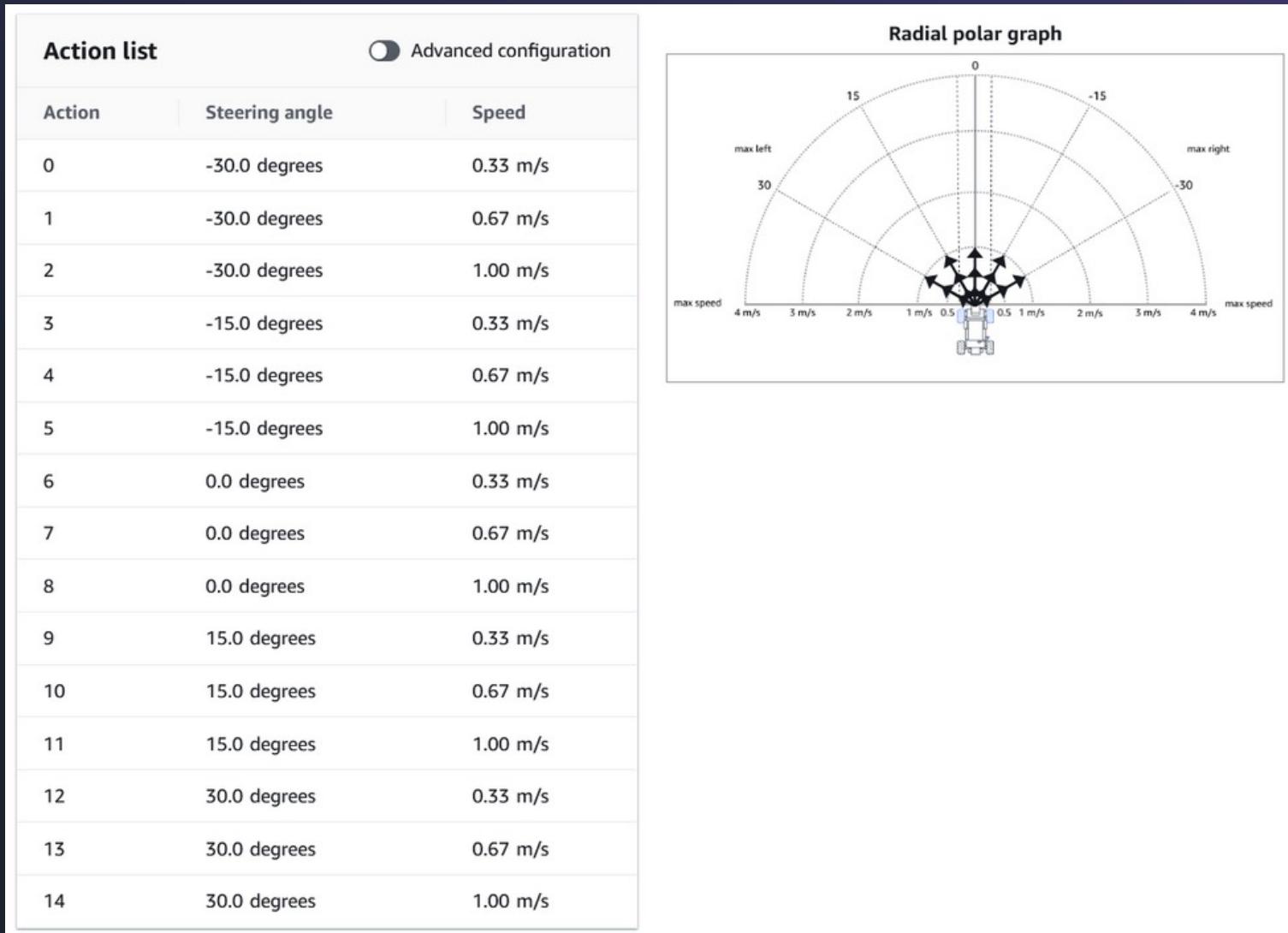
AWS DeepRacer neural network architecture



AWS DeepRacer neural network architecture



Action space



Demo – Create your Student League Account

Sing-Up



AWS DeepRacer Student

i Apply to the AWS AI & ML Scholarship and compete in the AWS DeepRacer Student League.

Sign up

Enter your email address and choose a password to create your [AWS Player account](#).

Email address

 ⚠ Required

Password

 ⚠ Required

Passwords must contain at least 8 characters uppercase, lowercase, number, and symbol.

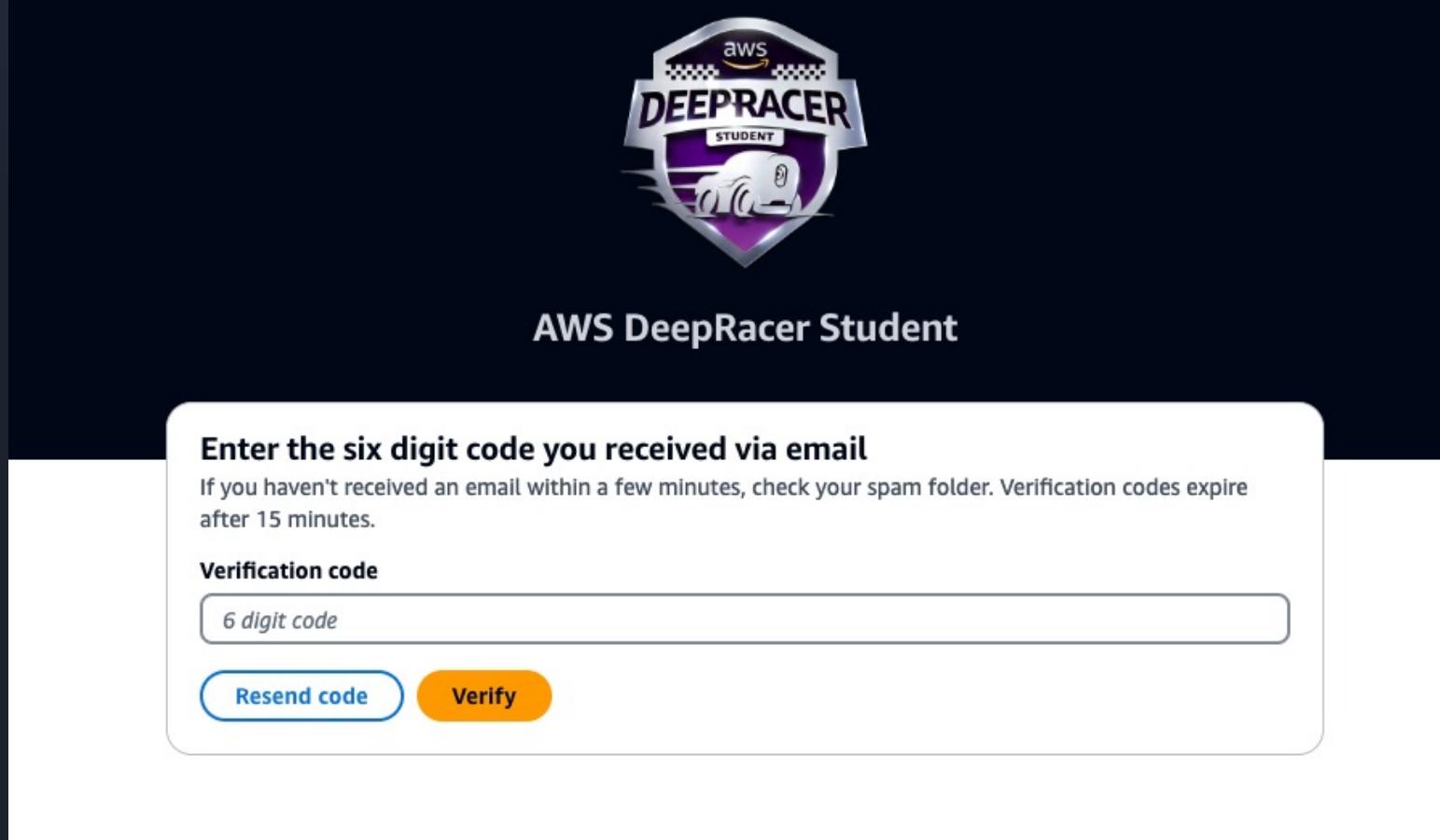
Show password

[Sign up](#)

Already have an AWS Player account? [Sign in](#)

By creating an account and using AWS DeepRacer Student, you agree to the [AWS Customer Agreement](#) ("Agreement"), [AWS Service Terms](#), [AWS Privacy Notice](#), and [AWS Acceptable Use Policy](#). Your AWS DeepRacer Student account is considered an AWS account for purposes of the Agreement. If you already have an Agreement with AWS, you agree that the terms of that agreement govern your use of this product.

Provide the confirmation Code



After Sign-In complete Sign-Up

Middle name - *Optional*

Middle name

Can't find your major on the list? Choose **Other** to type.

▼

Current or prospective major

Can't find your major on the list? Choose **Other** to type.

Your major

Planned year of graduation

Enter the year that you will graduate.

Your year of graduation

I am a student enrolled in high school, college, or university.

residency

f residency

about the AWS AI & ML Scholarship?

Scholarship program, in partnership with Udacity, provides students with funding, career mentorship programs, and nanodegree scholarships.

Welcome to AWS DeepRacer Student



Use AWS DeepRacer Student to learn the foundations of machine learning (ML) and compete for prizes with your friends in the AWS DeepRacer Student League.

Almost there! We need a few more details before you're off to the races.

[I will do this later. Sign out for now](#)

[Complete sign-up](#)

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Sign-Up competition

Complete sign-up for AWS DeepRacer Student [Info](#)

To start using AWS DeepRacer Student, we need a few more details.



Add your personal information to create your AWS DeepRacer Student account.

First name <input type="text" value="Your"/>	Middle name - Optional <input type="text" value="Middle name"/>	Last name <input type="text" value="Name"/>
School Can't find your school on the list? Choose Other to type. <input type="text" value="Other"/>	Current or prospective major Can't find your major on the list? Choose Other to type. <input type="text" value="Computer Science & Programming"/>	Planned year of graduation Enter the year that you will graduate. <input type="text" value="2025"/> Year of graduation must be between 2022 and 2030.
Enter the name of your school <input type="text" value="Your School Name"/>		
<input checked="" type="checkbox"/> I certify that I am a student enrolled in high school, university, or community college or an educator or event organizer for students in high school, university, or community college.		
Country/Area of residency <input type="text" value="Germany"/>		

Demo – Training using the AWS DeepRacer console

Next Steps...



AWS DeepRacer
League



AWS DeepRacer
Student League

Summary

- Introducing AWS DeepRacer
- Introduction to machine learning
- Reinforcement learning key terms
- The reward function
- AWS DeepRacer ML architecture
- Demo

General Points

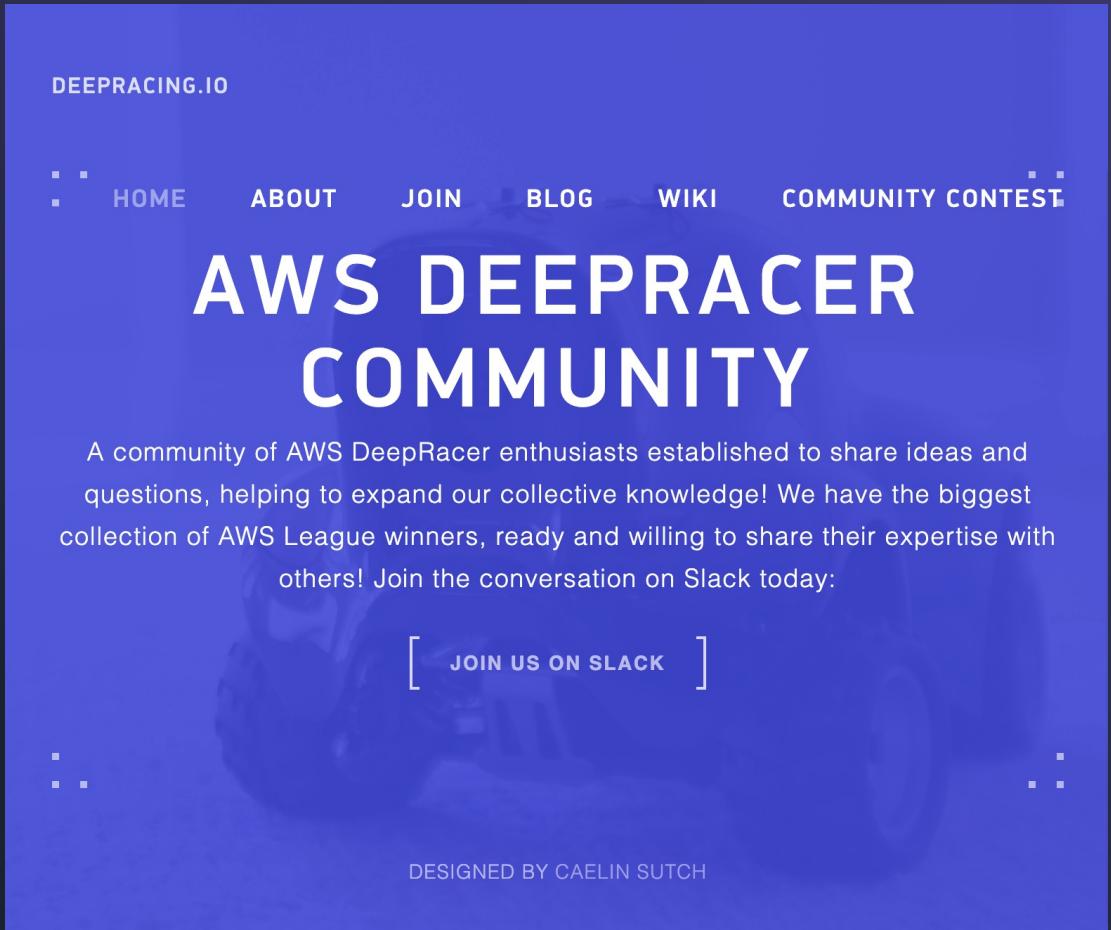
- Qualifier via Student Accounts
<https://student.deepracer.com/raceToken/s6FunlWBQSOxcptCp7bXkQ>
- Track is re:Invent 2018
- Fastest Lap wins
- Direction is counter clockwise
- Models for Live Race can be trained via Student Accounts, regular Accounts or DeepRacer 4 Cloud

Race Day – Racing Rules

- Track is re:Invent 2018
- Direction is counter clockwise
- 2 minutes per run
- Fastest Lap wins
- Unlimited resets
- Race as much runs as you like and we have time. After a run you have to requeue
- You can race during the whole day

Most Important Ressource

<https://deepracing.io/>



DeepRacer for Cloud

[View on GitHub](#) 

deepracer-for-cloud

Creates an AWS DeepRacing training environment which can be deployed in the cloud, or locally on Ubuntu Linux, Windows or Mac.

Introduction

Provides a quick and easy way to get up and running with a DeepRacer training environment in AWS or Azure, using either the Azure [N-Series Virtual Machines](#) or [AWS EC2 Accelerated Computing instances](#), or locally on your own desktop or server.

Resources I

DeepRacer Community:

<https://deepracing.io/>

Blogs:

<https://towardsdatascience.com/an-advanced-guide-to-aws-deepracer-2b462c37eea>

<https://blog.gofynd.com/how-we-broke-into-the-top-1-of-the-aws-deepracer-virtual-circuit-573ba46c275>

<https://towardsdatascience.com/penalizing-the-discount-factor-in-reinforcement-learning-d672e3a38ffe>

<https://towardsdatascience.com/proximal-policy-optimization-tutorial-part-2-2-gae-and-ppo-loss-fe1b3c5549e8>

<https://awstip.com/exponentially-improving-at-aws-deep-racer-b07ec116087>



Resources II

Videos:

<https://youtu.be/v7SXfYSWvBU> - Log Analysis with Breadcentric

<https://www.youtube.com/@BoltronRacingTeam> – Bolton Racing Team Channel

DeepRacer for Cloud:

<https://aws-deepracer-community.github.io/deepracer-for-cloud/>

<https://docs.google.com/document/d/1m14DWQPau6ee093i8ebV8iE7wiNwXWy9ALII6dZrlZY/edit#>

DeepRacer Log Guru

<https://github.com/aws-deepracer-community/deepracer-log-guru>

Resources III

Github:

<https://github.com/cdthompson/deepracer-k1999-race-lines> Raceline calculation

https://github.com/aws-deepracer-community/deepracer-race-data/tree/main/raw_data/tracks Tracks

Thingiverse:

<https://www.thingiverse.com/thing:5653115> Suspension Tuning part



Let's hit the road



© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.