



Trabajo 1: **AWS DeepRacer**

CE Inteligencia Artificial y Big Data
Modelos de Inteligencia Artificial
2024/2025

Daniel Marín López

Índice

1. Introducción	3
2. AWS DeepRacer	3
3. Creación del modelo	4
4. Entrenamiento	11
5. Conclusión	13

1. Introducción

En esta práctica se entrenará un coche utilizando **DeepRacer**, una aplicación de **Amazon Web Services (AWS)** que emplea técnicas avanzadas de aprendizaje automático, específicamente **aprendizaje por refuerzo**, para entrenar un coche de carreras autónomo. A través de esta herramienta, los participantes podrán diseñar, simular y ajustar un modelo de control que optimice el desempeño del coche en una pista virtual.

2. AWS DeepRacer

Primero, accedemos a la consola de **AWS** y, una vez dentro, nos dirigimos a la sección correspondiente buscando la aplicación **DeepRacer**. Si ya has interactuado previamente con esta herramienta, es probable que aparezca automáticamente en la sección de "Visitados recientemente", lo que facilita su localización. En caso de que no esté allí, se puede usar la barra de búsqueda en la parte superior de la consola para encontrarla rápidamente. Una vez localizada, haz clic en ella para acceder al entorno donde podrás explorar todas las funcionalidades que ofrece, como la creación de modelos, la configuración de carreras virtuales y la gestión de recursos asociados a tu proyecto.



Una vez hemos accedido a la aplicación, aparecerá la siguiente pantalla:

Machine Learning

AWS DeepRacer

The fastest way to get rolling with machine learning

Get hands-on with a fully autonomous 1/18th scale race car driven by reinforcement learning, 3D racing simulator, and global racing league.

Get started with reinforcement learning

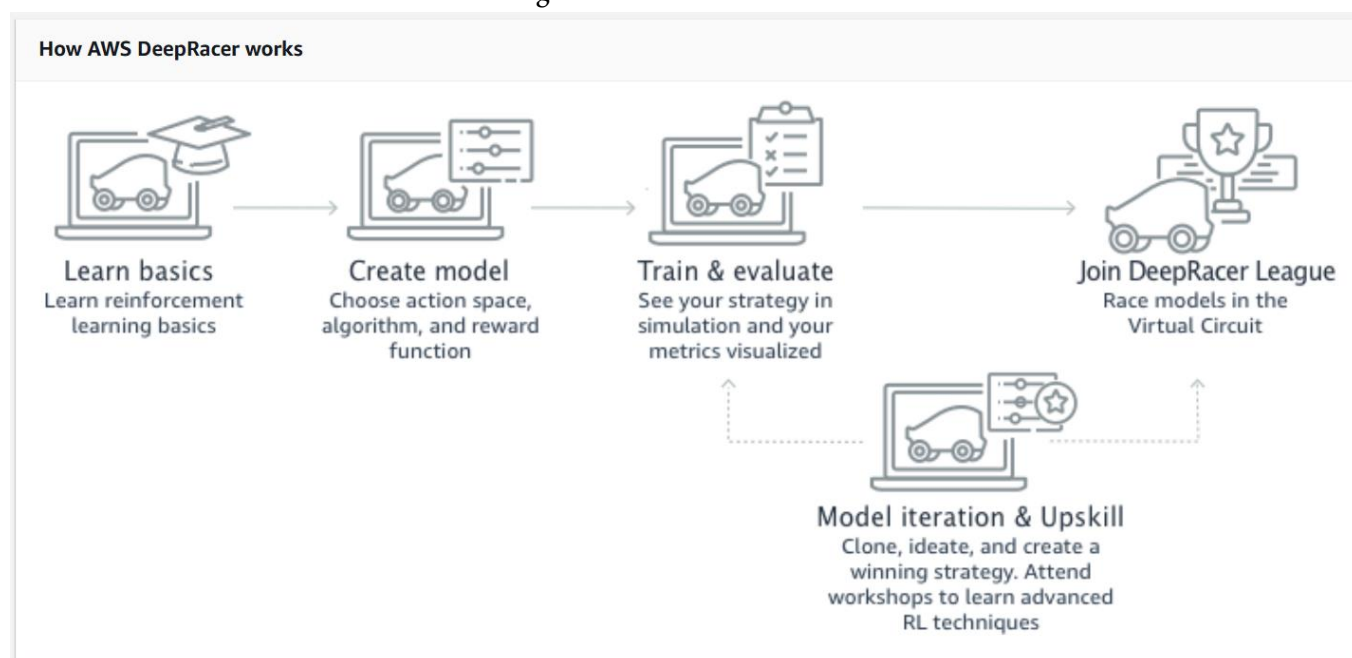
Build your model, evaluate its performance on a virtual track, and then compete in the AWS DeepRacer League.

Create model

En donde ya tenemos la posibilidad de crear un nuevo modelo basado en aprendizaje por refuerzo que recorrerá una pista sin salirse de la misma en el menor tiempo posible.

3. Creación del modelo

A la hora de crear nuestro modelo se dan las siguientes instrucciones:



En este entorno digital, se desarrolla un fascinante ecosistema donde se crean, entrenan, clonan o modifican modelos únicos, diseñados específicamente para competir en emocionantes carreras online. Los usuarios deberán configurar sus modelos para superar las carreras. Además, el sistema de clonación o modificación permite a los usuarios optimizar sus modelos, adaptándolos a diferentes pistas y estilos de juego. Estas competencias online, cargadas de adrenalina, no solo ponen a prueba la destreza técnica de los usuarios, sino también su capacidad para diseñar y evolucionar sus modelos en un entorno dinámico y competitivo.

Lo primero que nos pregunta es el nombre del modelo, lo cual es fundamental para identificar de manera única el modelo que queremos entrenar o trabajar. Esto es especialmente útil si estamos manejando múltiples modelos en paralelo, ya que facilita la organización y el seguimiento de cada uno. Además, también nos pregunta en qué pista deseamos entrenarlo, refiriéndose al entorno o conjunto de datos específicos donde se llevará a cabo el proceso de aprendizaje. La elección de la pista es crucial, ya que determina el contexto, los parámetros y las reglas bajo las cuales el modelo adquirirá conocimiento y optimizará su desempeño. Este paso inicial es esencial para garantizar que las configuraciones sean precisas y que el modelo se ajuste a las necesidades del proyecto.

Training details

Model name

danielMarin-trabajo1

The model name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and hyphens (-). No spaces or underscores (_).

Training job description - optional

Log details for quick reference

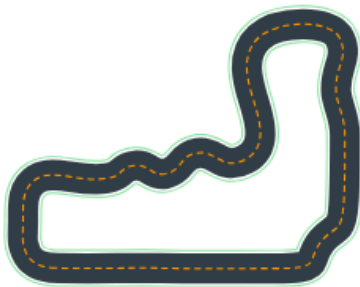
The model description can have up to 255 characters.

☐ Kumo Torakku Training

June's training track for the 2019 Virtual Circuit World Tour, the Kumo Torakku is complex and dramatic, featuring two 180-degree turns and a long straightaway.

Length: 22.63 m (74.25')
Width: 76 cm (30")

Direction: Counterclockwise

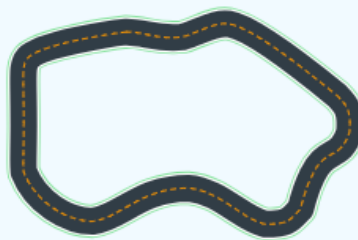


☒ London Loop Training

May's training track for the 2019 Virtual Circuit World Tour, the London Loop provides a variety of moderate curves and straightaways.

Length: 19.45 m (63.81')
Width: 76 cm (30")

Direction: Counterclockwise

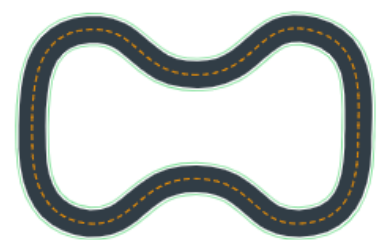


☐ Bowtie Track

The Bowtie offers a simple symmetrical track with a twist. It features shallow turns that you can use to experiment with different racing behaviors.

Length: 17.43 m (57.19')
Width: 76 cm (30")


Direction: Counterclockwise

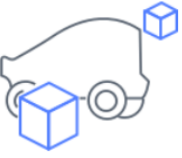



Una vez definida la pista, procedemos a seleccionar el tipo de carrera que queremos realizar. Entre las diversas modalidades disponibles, optamos por la clásica y desafiante prueba de **tiempo contra reloj**. Este modo pone a prueba no solo la velocidad del modelo, sino también nuestra precisión y estrategia, ya que el objetivo principal será realizar el recorrido de la mejor forma en el menor tiempo posible.

Race type

Choose a race type

☒ **Time trial**
The agent races against the clock on a well-marked track without stationary obstacles or moving competitors.


☐ **Object avoidance**
The vehicle races on a two-lane track with a fixed number of stationary obstacles placed along the track.


☐ **Head-to-head racing**
The vehicle races against other moving vehicles on a two-lane track.


AWS DeepRacer permite escoger el tipo de algoritmo de entrenamiento y configurar los hiperparámetros, lo que brinda flexibilidad y control sobre el proceso de aprendizaje automático. Entre las opciones disponibles, usaremos **PPO** (Proximal Policy Optimization), un algoritmo de refuerzo profundo que pertenece al campo del aprendizaje automático. Este algoritmo es ampliamente utilizado debido a su estabilidad y eficiencia en tareas de optimización de políticas en entornos complejos, como el de simulaciones autónomas. PPO permite que el agente aprenda a tomar decisiones mediante la interacción con el entorno, ajustando sus acciones para maximizar una recompensa acumulativa. Por esta razón, es especialmente adecuado para entrenar modelos en entornos simulados como los de AWS DeepRacer, donde se busca optimizar el desempeño del vehículo en pista.

Training algorithm and hyperparameters [Info](#)

☒ **PPO**
A state-of-the-art policy gradient algorithm which uses two neural networks during training – a policy network and a value network.

☐ **SAC**
Not limiting itself to seeking only the maximum of lifetime rewards, this algorithm embraces exploration, incentivizing entropy in its pursuit of optimal policy.

▼ **Hyperparameters**

Gradient descent batch size

☐ 32

☒ 64

☐ 128

☐ 256

☐ 512

Number of epochs

Integer between 3 and 10.

Learning rate

Real number between 0.00000001 (1e-8) and 0.001 (1e-3).

Entropy

Real number between 0 and 1.

Discount factor

Real number between 0 and 1.

Loss type

☐ Mean squared error

☒ Huber

Number of experience episodes between each policy-updating iteration

Integer between 5 and 100.

Cancel

Previous

Next

Después escogemos cómo será el espacio de acción, esto es debido a que nuestro modelo de aprendizaje por refuerzo recibirá una serie de recompensas en base a las decisiones o acciones que haya tomado nuestro modelo. Con acción podemos hablar del ángulo de giro o la velocidad a la que va. Tomaremos la opción de espacio de acción continuo y dejaremos los parámetros por defecto.

▼ Why is action space important?

In reinforcement learning, the set of all valid actions, or choices, available to an agent as it interacts with an environment is called an *action space*. In the AWS DeepRacer console, you can train agents in either a *discrete* or *continuous* action space.

When training an AWS DeepRacer model, the action space defines what speed and steering angle combinations are available to the agent. An *action* is a single speed and steering angle combination or choice an agent can make.



Select action space [Info](#)

Action spaces



Continuous action space

A continuous action space allows the agent to select an action from a range of values for each state.



Discrete action space

A discrete action space represents all of the agent's possible actions for each state in a finite set.

Define continuous action space [Info](#)

In a continuous action space setting, the agent learns to pick the optimal speed and steering values from the min/max bounds you provide through training. Providing a range of values for the model to pick from seems to be the better option but the agent has to train longer to learn to choose the optimal actions.

Steering angle

The steering angle determines the range of steering angles in which the front wheels of your agent can turn.

Left steering angle range

degrees

Values are between 0 and 30.

Right steering angle range

degrees

Values are between -30 and 0.

Speed

The speed determines how fast your agent can drive.

Min/max speed defines the range of speeds available to the agent while training.

Minimum speed

m/s

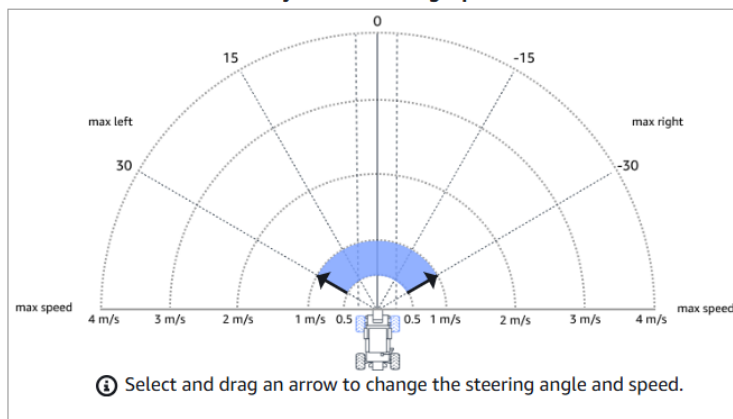
Values are between 0.1 and 4.

Maximum speed

m/s

Values are between 0.1 and 4.

[Reset to default values](#)

Dynamic sector graph

Cancel

Previous

Next

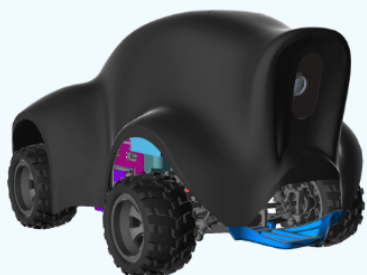
Luego nos pregunta el vehículo, solo hay uno por lo que no tenemos que pensar demasiado a la hora de escogerlo.

Vehicle shell with sensor configuration (1)

< 1 >

The Original DeepRacer

Sensor(s): Camera; Shell: DeepRacer



Lo siguiente que nos muestra es la función de recompensa, una herramienta clave de nuestro modelo que nos permite personalizar el comportamiento según nuestras necesidades. Si deseamos editarla, tenemos la posibilidad de añadir nuevas reglas para reflejar criterios adicionales, o bien, modificar alguna de las reglas existentes para ajustarlas a objetivos más específicos. Esta flexibilidad es esencial para adaptar el sistema a diferentes escenarios, optimizando su desempeño y asegurando que responda de manera adecuada a las demandas del contexto en el que se implemente.

Reward function [Info](#)

The reward function describes immediate feedback (as a score for reward or penalty) when the vehicle takes an action to move from a given position on the track to a new position. Its purpose is to encourage the vehicle to make moves along the track to reach its destination quickly. The model training process will attempt to find a policy which maximizes the average total reward the vehicle experiences. [Learn more](#) about the reward function and the reward input parameters you can use in your function.

```
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)
```

Por último, se preguntan las últimas configuraciones, entre ellas si deseamos inscribir nuestro modelo a las carreras online. Pero este modelo servirá como base para clonar otros que se puedan entrenar en el futuro.

Stop conditions [Info](#)

Set the conditions for your training job to stop. To avoid run-away jobs, you can limit the length of a job to within a maximum time period (**Maximum time**).

The training will stop when the specified criteria is met. When your model has stopped training, you will be able to clone your model to start training again using new parameters.

Maximum time

Maximum time must be between 5 and 1440 minutes.

Automatically submit to the DeepRacer race

☒ Submit this model to the following race after training completion.

Select DeepRacer League leaderboard to submit model to

There are no fees or costs associated with submitting a model to the DeepRacer League.

Email address for prize notifications - *new* [Info](#)

You must provide a contact email to claim prizes in the 2024 AWS DeepRacer League. The email you provide will **only** be used to send notifications for the prizes you earn. You can participate in the League without providing a notification email, but you risk forfeiting any prizes you earn. You can add a notification email at a later date in **Your profile**.

☒ I don't want to provide an email address. I understand that I risk forfeiting any prizes I earn.

☐ Email address for prize notifications

League requirements

Country/Area of residence

Select your country/area of residence to submit models in an AWS Virtual Circuit race.

Terms and conditions

I have read, understood, and unconditionally agree to the terms and conditions of the [2024 AWS DeepRacer league](#), including without limitation, the publicity rights in section 10 of the terms and conditions.

☐ I accept the terms and conditions and acknowledge that this is my country/area of residence during the 2024 league season.

4. Entrenamiento

Terminado el proceso de creación del modelo, se pone en preparación para el entrenamiento.

▼ Training [Info](#)

Download logs

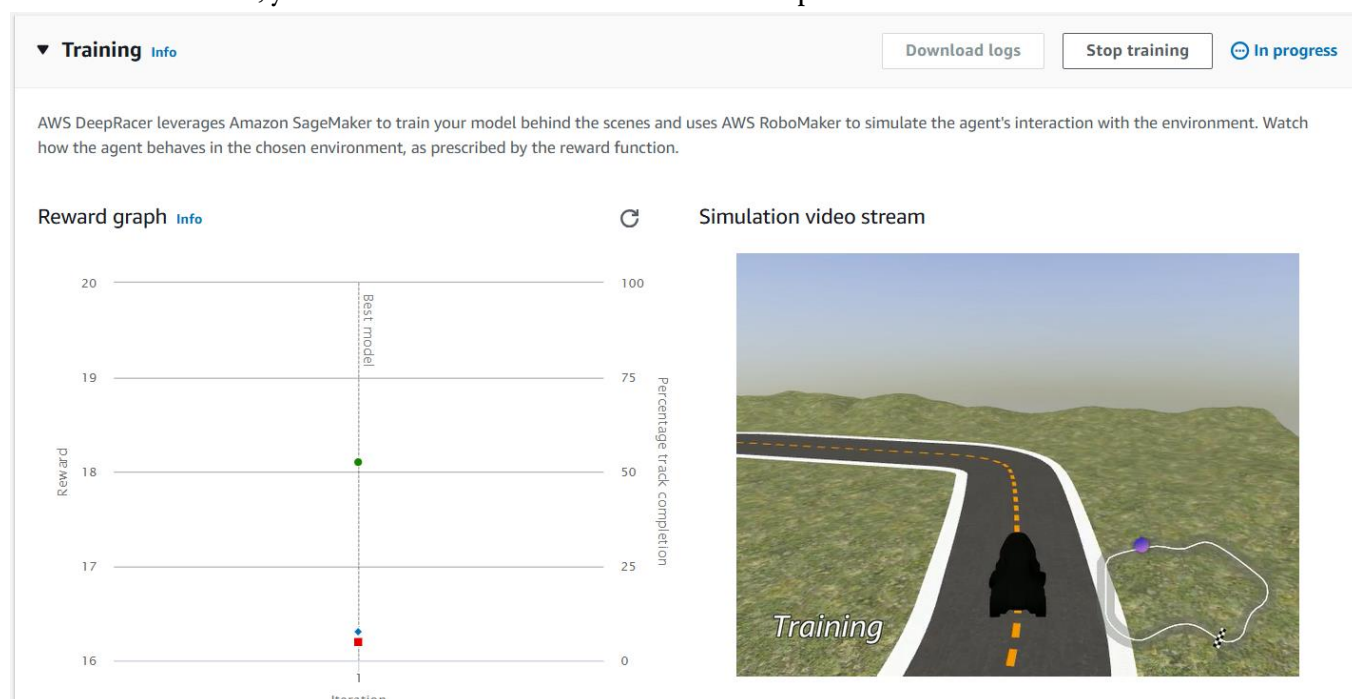
Stop training

Initializing

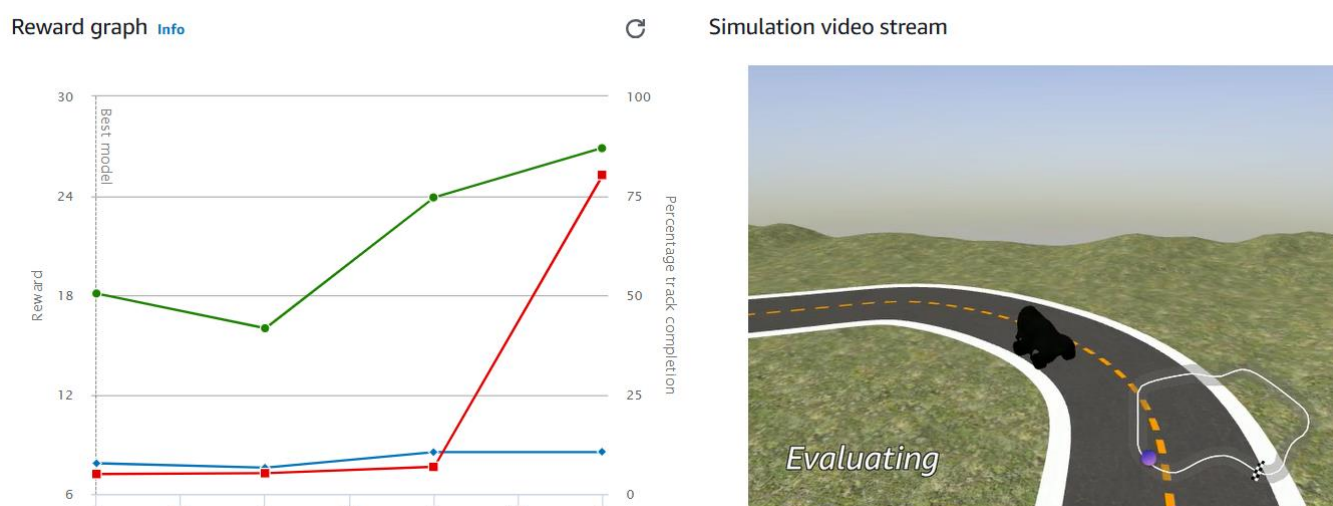
Inspect training after the training instance is provisioned and the training job is initialized.

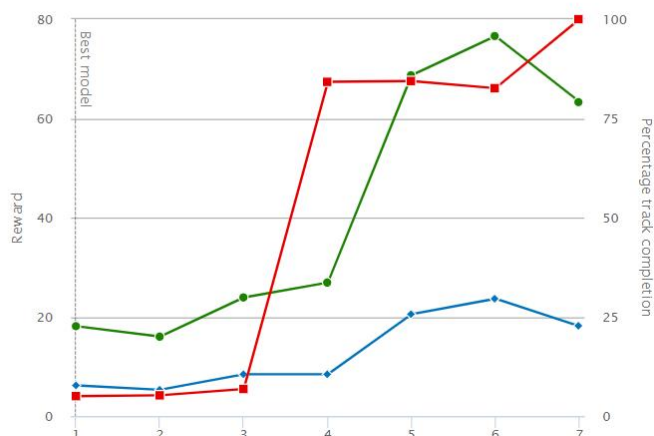
Watch training in progress to see how your agent attempts to complete the track in the simulator.

Tras esperar un rato, ya aparece nuestro modelo. Este aparece con una gráfica que muestra el entrenamiento y la evaluación del mismo, y a la derecha un video de nuestro coche en la pista seleccionada.



Por lo que simplemente dejaremos que el modelo vaya entrenando tranquilamente hasta que realice el circuito sin salirse de él en ningún momento. Este entrenamiento tiene de máximo 1 hora. A continuación se muestran capturas del entrenamiento realizado.



Reward graph [Info](#)

Simulation video stream



Una vez nuestro modelo consigue no salirse de la pista, paramos el entrenamiento.

Stop training

Are you sure you want to stop the training of your reinforcement learning model?

Once stopped, the training cannot be continued. It takes approximately 4 minutes to stop the training process.

Cancel
Stop training

Tras parar el entrenamiento, ya tenemos un modelo entrenado que posteriormente podremos clonar y usar en carreras online.

Training configuration

Race type
Time trial

Environment simulation
London Loop Training - Counterclockwise

Reward function
Show

Sensor(s)
Camera

Action space type
Continuous

Action space
Speed: [0.5 : 1] m/s
Steering angle: [-30 : 30] °

Framework
Tensorflow

Reinforcement learning algorithm
PPO

Hyperparameter	Value
Gradient descent batch size	64
Entropy	0.1
Discount factor	0.99
Loss type	Huber
Learning rate	0.0003
Number of experience episodes between each policy-updating iteration	20
Number of epochs	10

AWS DeepRacer > Your models

Models (0/2)							Import model	Actions ▼	Create model
<input type="text" value="Search models"/>							< 1 > ⚙		
<input type="checkbox"/>	Name ▼	Description ▼	Status ▼	Algorithm ▼	Sensors ▼	Creation time ▼			
<input type="checkbox"/>	danielMarin-trabajo1		Ready	PPO	Camera	Sat, 07 Dec 2024 17:34:31 GMT			
<input type="checkbox"/>	danielMarin-model1	Modelo 1 de Daniel Marín	Ready	PPO	Camera	Thu, 28 Nov 2024 18:15:10 GMT			

Aquí se muestran todos los modelos creados hasta el momento.

5. Conclusión

Hemos aprendido cómo usar DeepRacer de AWS, comprendiendo cada etapa del proceso: desde la creación de un modelo ajustando diversos parámetros clave, hasta el entrenamiento del agente para que logre recorrer un circuito sin salirse de la pista. Este proceso no solo nos ha permitido adentrarnos en la configuración técnica de un sistema basado en aprendizaje por refuerzo, sino que también nos ha brindado una comprensión más profunda de cómo funcionan los agentes inteligentes. Hemos explorado cómo estos agentes son capaces de interactuar con su entorno, evaluar diferentes situaciones y tomar decisiones basadas en los parámetros y recompensas establecidos, lo que les permite mejorar su desempeño de manera autónoma con cada iteración. Además, esta experiencia nos ha ofrecido una perspectiva práctica sobre la inteligencia artificial aplicada, mostrando cómo las tecnologías avanzadas pueden simular y optimizar comportamientos complejos en entornos dinámicos, lo que abre la puerta a una amplia gama de aplicaciones en la vida real.