Making Reinforcement Learning Practical For Real World Developers

Praveen Veerath
Sr. AI Solutions Architect
Amazon Web Services

aws

THE AWS ML STACK

Broadest and deepest set of capabilities

AI Services

VISION			SPEECH		LANGUAGE		CHATBOTS	FORECASTING	RECOMMENDATIONS
REKOGNITION IMAGE	REKOGNITION VIDEO	TEXTRACT	POLLY	TRANSCRIBE	TRANSLATE	COMPREHEND & COMPREHEND MEDICAL	LEX	FORECAST	PERSONALIZE

ML Services

Amazon SageMaker

Ground Truth	Notebooks	Algorithms + Marketplace	Reinforcement Learning	Training	Optimization	Deployment	Hosting

ML Frameworks + Infrastructure

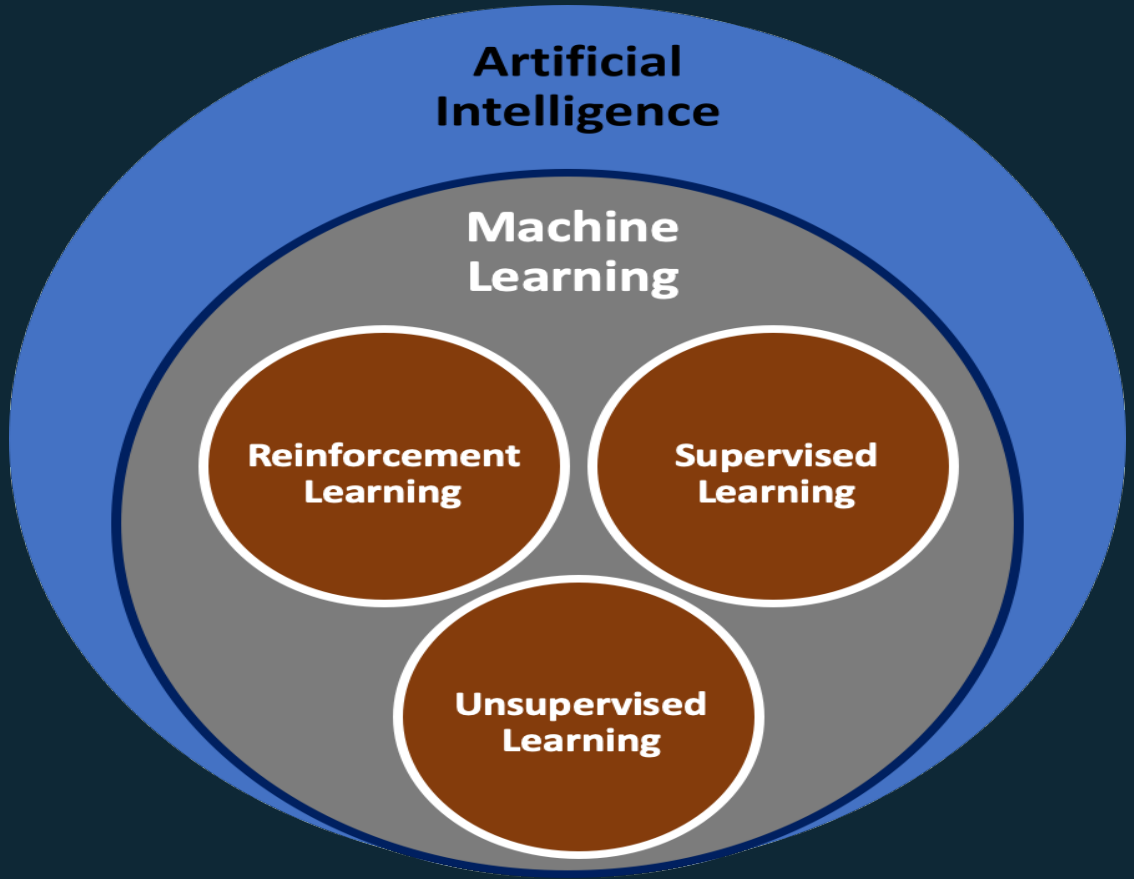
FRAMEWORKS	INTERFACES
TensorFlow, mxnet, PYTORCH	GLUON, Keras

INFRASTRUCTURE

EC2 P3 & P3DN | EC2 G3 | EC2 C5 | FPGAS | GREENGRASS | ELASTIC INFERENCE | INFERENTIA

aws

Reinforcement Learning in the broader AI context

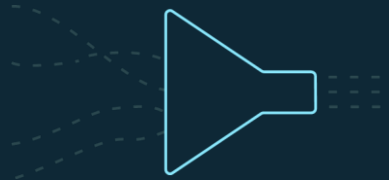


aws

Machine Learning Overview

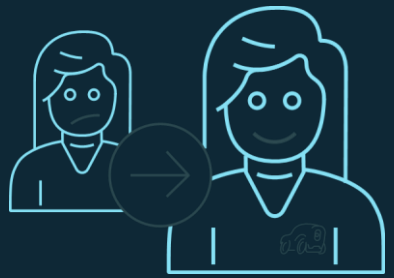
SUPERVISED

Example Driven Training – every datum has a corresponding label



UNSUPERVISED

No labels for training data



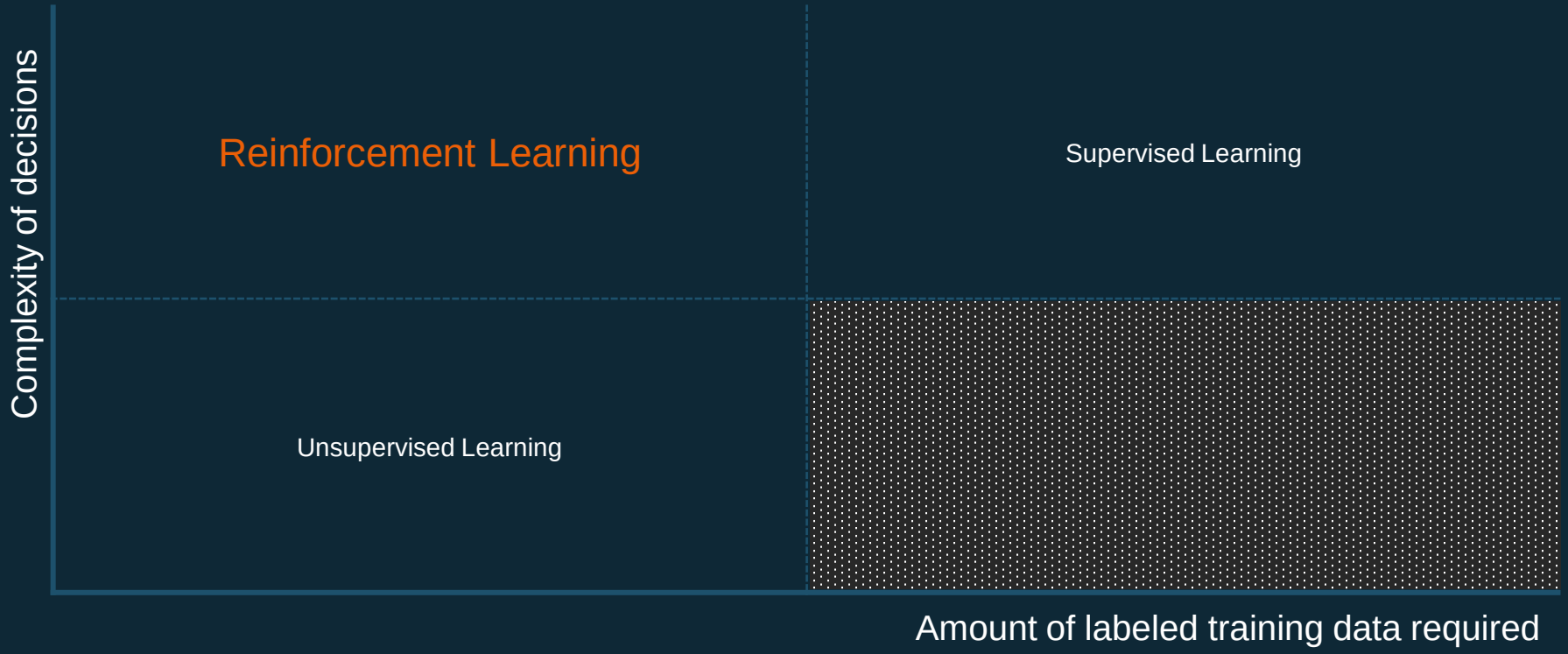
REINFORCEMENT

Learns through consequences of actions in a specific environment

aws

How do you build machine learning models that can make decisions when there is no labeled data?

Complexity vs Data



How do you learn in RL?

Learn by interacting with the environment

Trial and error

Observe results

Optimize learning strategy to maximize long term reward

Model learns how to make complex decisions

aws

What is an RL environment?

Real-world or a representation of the real world

Programmed to represent real world conditions

Enables interaction with user or a computer program

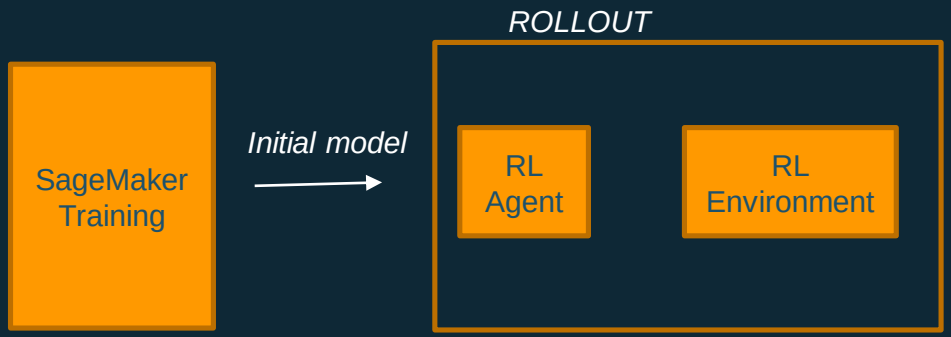
Dynamic and updates itself based on the interactions and programmed behavior

aws

How SageMaker RL works

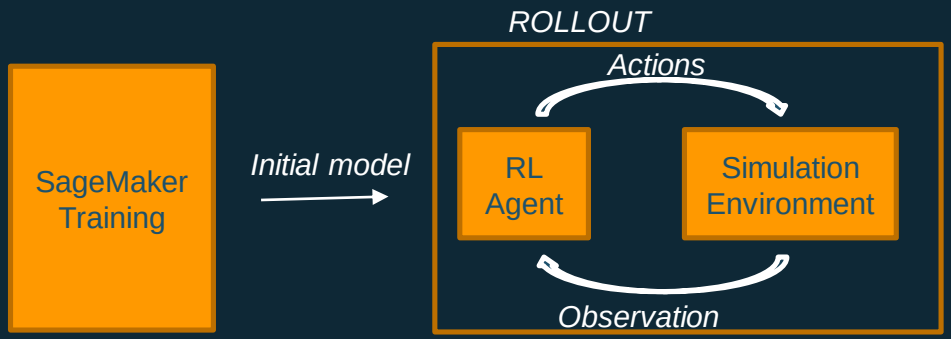
Let's train our humanoid, Harry, to walk

The players



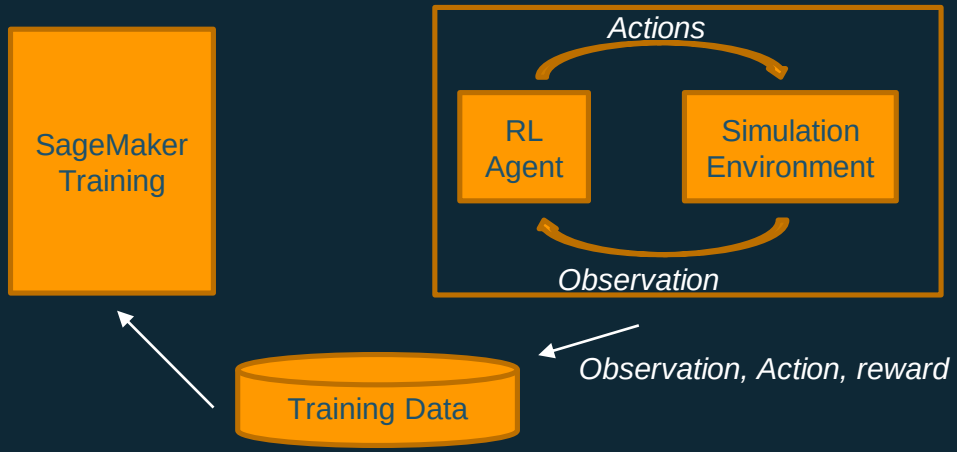
At first, Harry can’t even stand up

Actions and Observations

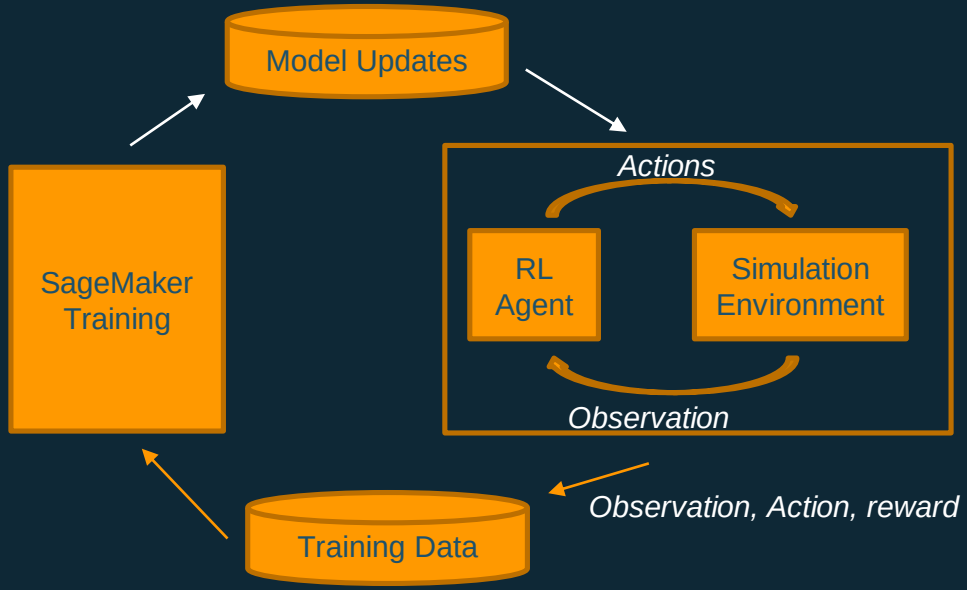


Learning from Harry’s actions and observations

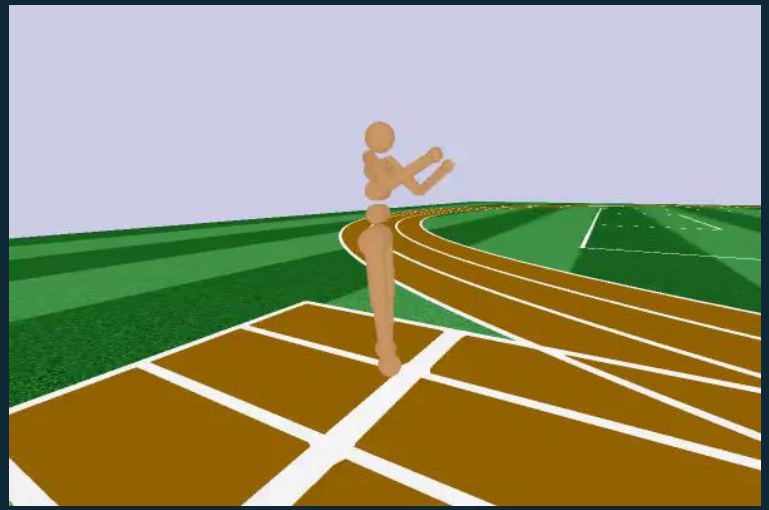
Interactions in the environment generate training data



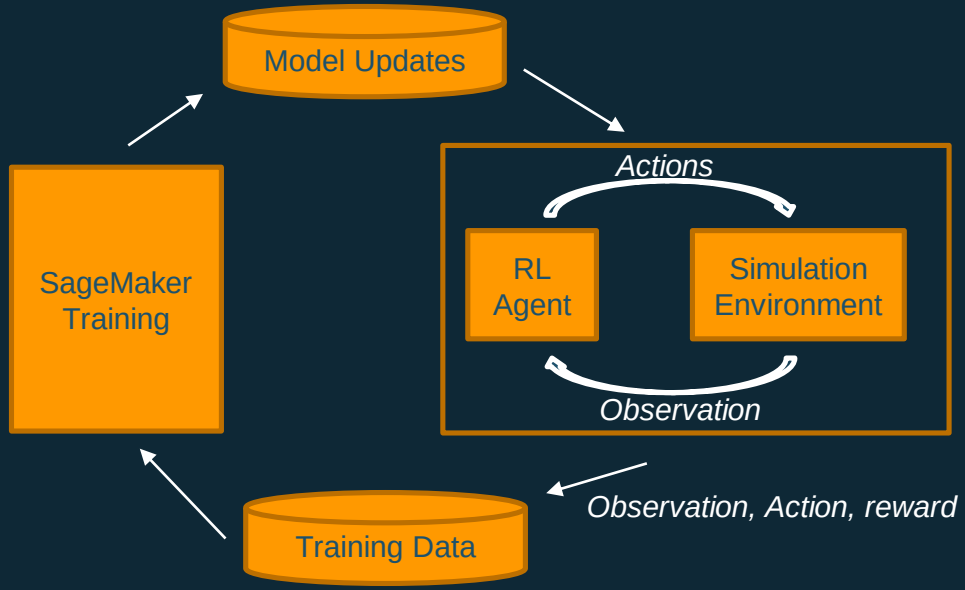
Training results in model updates



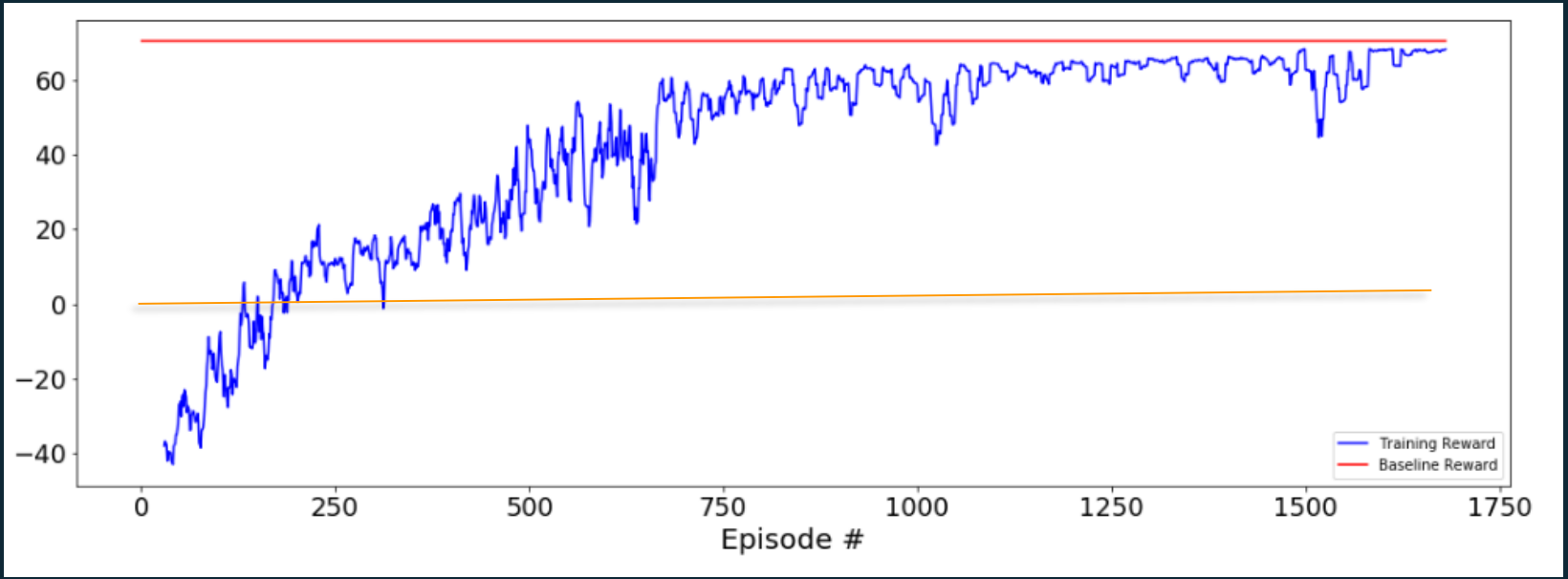
Harry learns to stand and step



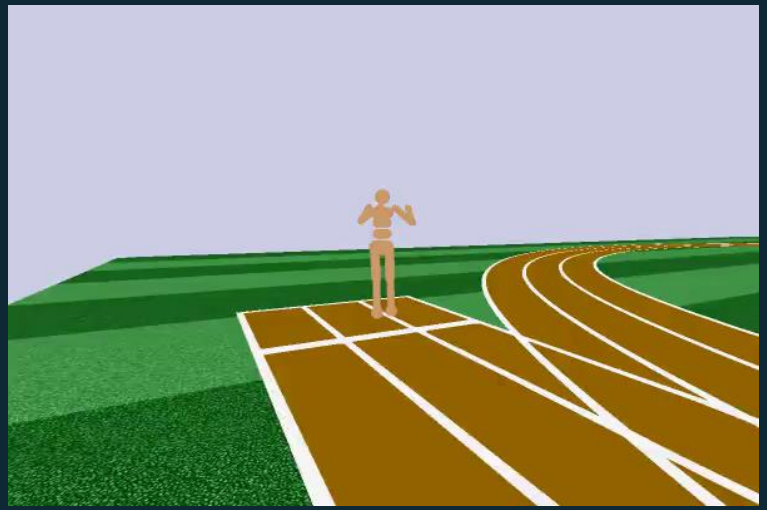
Multiple training episodes improve learning

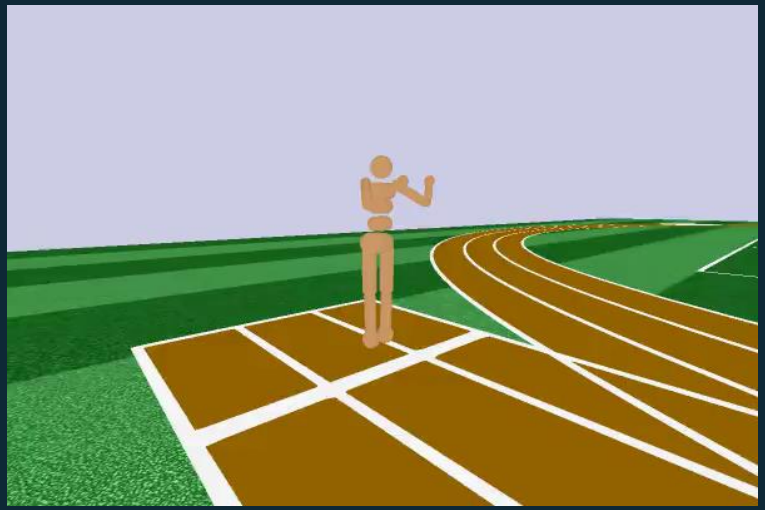


RL Agents try to maximize rewards

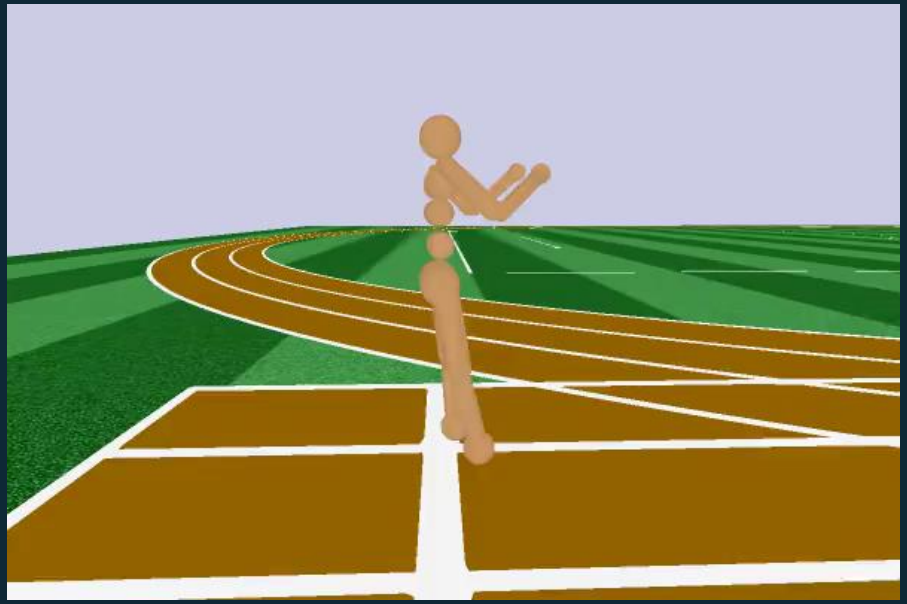


After many episodes of training





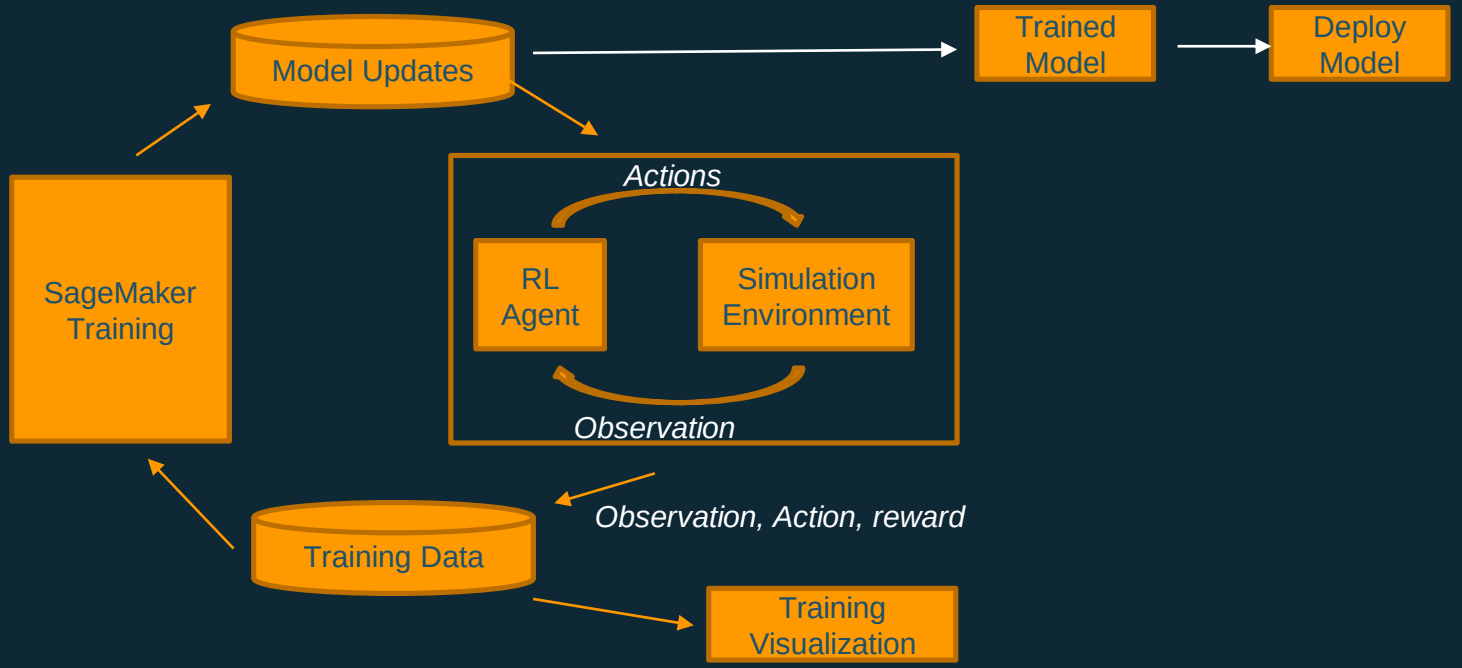
Eventually, Harry learns how to walk and run



You can continue training Harry to jump obstacles, play games, dance, and more

aws

Evaluate and deploy trained models



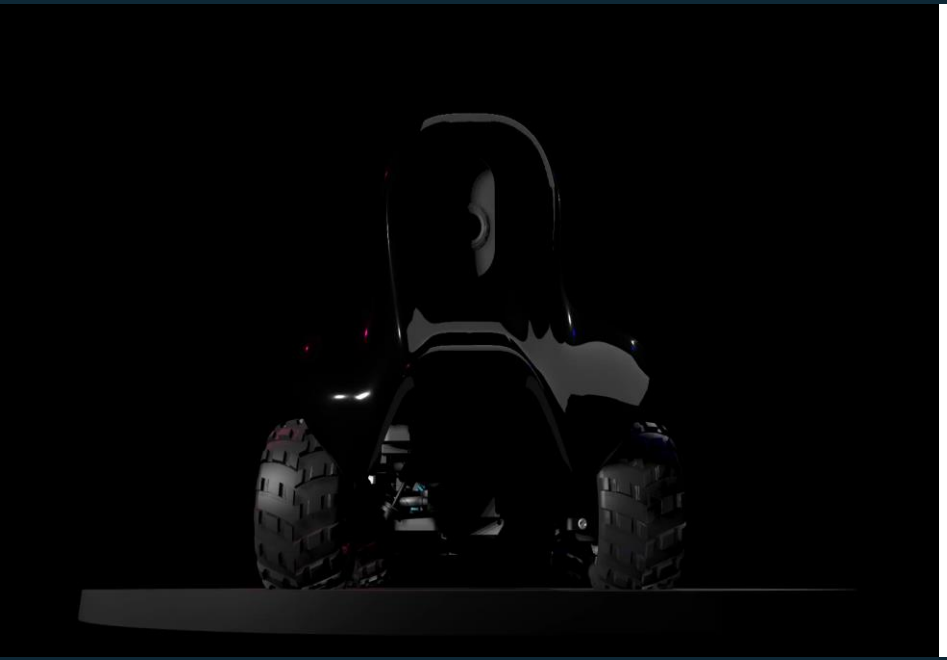
AWS DeepRacer Origin

How can we put Reinforcement Learning in the hands of all developers? *literally*

Our Objective: Teach developers RL

aws

AWS DeepRacer Car Specifications



Car: 1/18th scale 4WD with monster truck chassis

CPU: Intel Atom™ Processor

Memory: 4GB RAM

Storage: 32GB (expandable)

Wi-Fi: 802.11ac

Camera: 4 MP camera with MJPEG

Software: Ubuntu OS 16.04.3 LTS, Intel® OpenVINO™ toolkit, ROS Kinetic

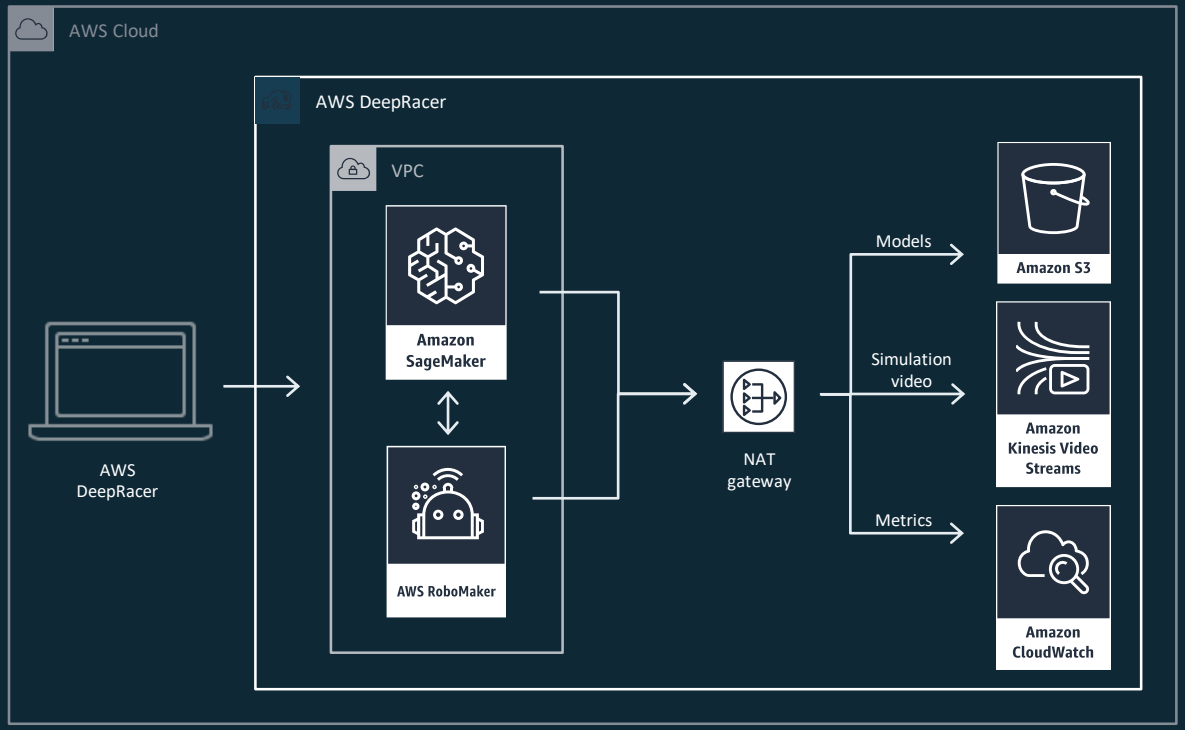
Drive battery: 7.4V/1100mAh lithium polymer

Compute battery: 13600mAh USB-C PD

Ports: 4x USB-A, 1x USB-C, 1x Micro-USB, 1x HDMI

Sensors: Integrated accelerometer and gyroscope

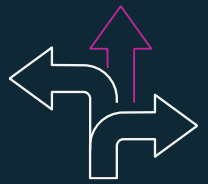
AWS DeepRacer Architecture



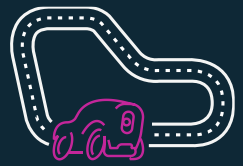
Reinforcement Learning with DeepRacer

MODEL

AGENT



ACTION

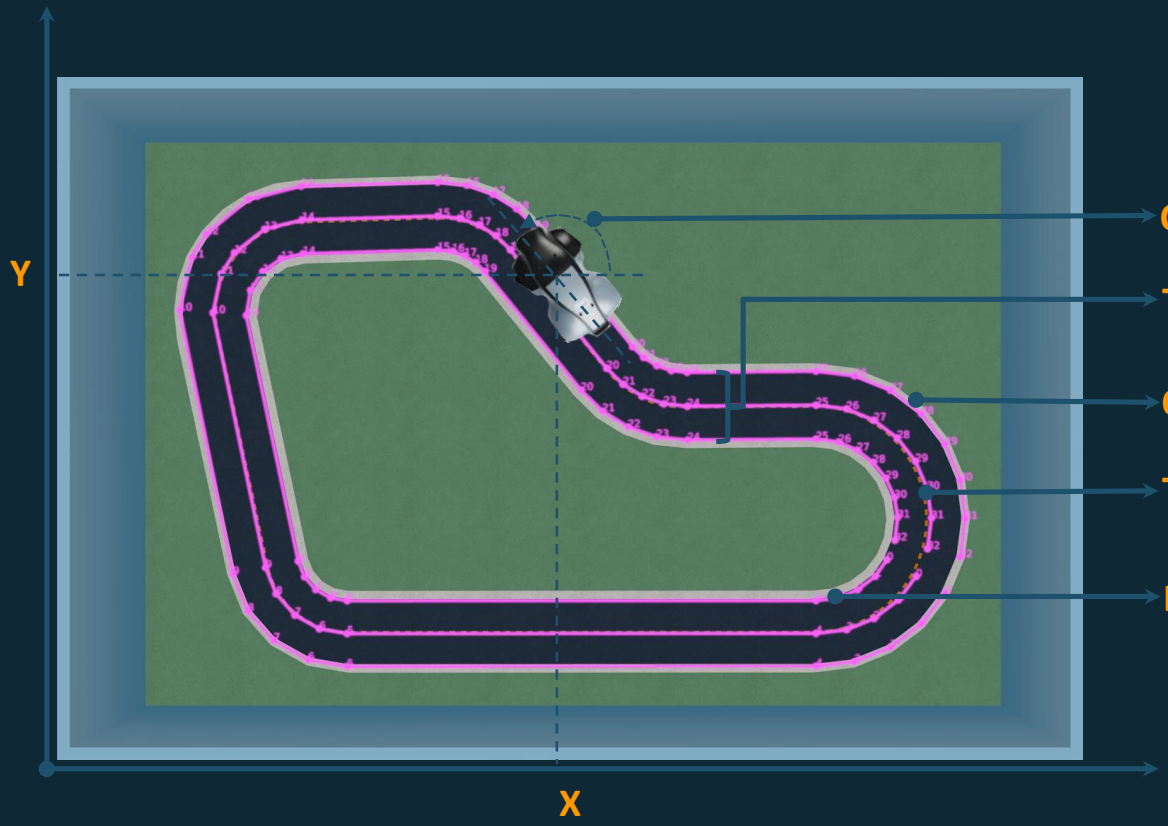


ENVIRONMENT

GOAL

aws

Coordinate system and track waypoints



CAR YAW

TRACK WIDTH

OUTER BOUNDARY WAYPOINTS

TRACK CENTER WAYPOINTS

INNER BOUNDARY WAYPOINTS

aws

How to train a reinforcement learning model.

Training an RL model

Training is an iterative process. In a simulator the agent explores the environment and builds up experience. The experiences collected are used to update the neural network periodically and the updated models are used to create more experiences.

With AWS DeepRacer, we are training a vehicle to drive itself. It can be tricky to visualize the process of training, so let's take a look at a simplified example.

Previous

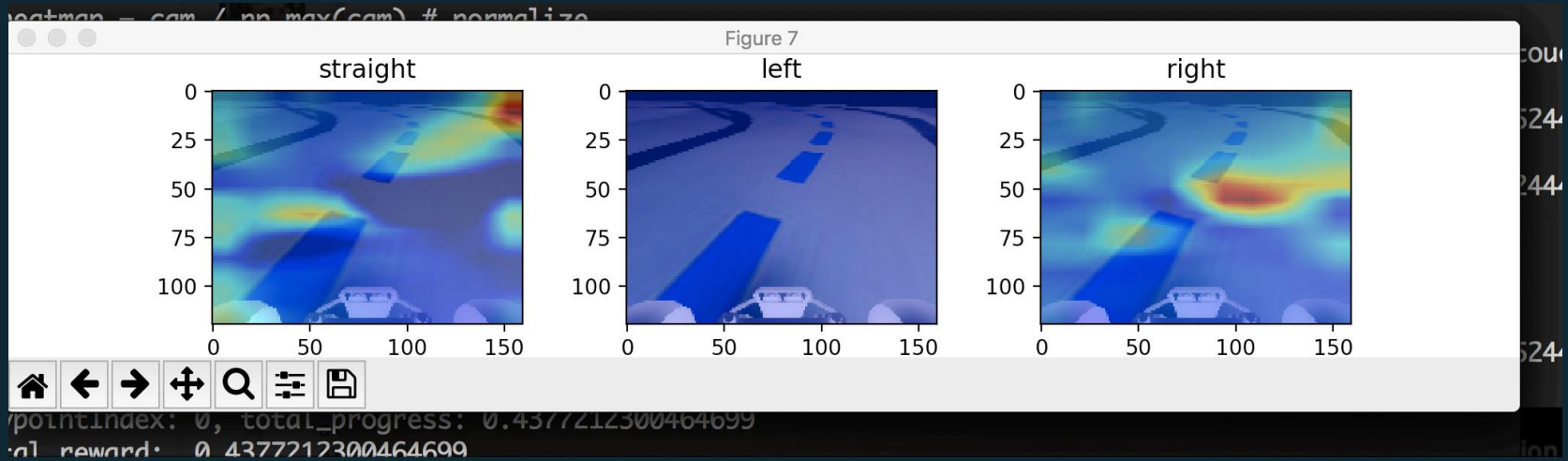
Next

Reward Function Example

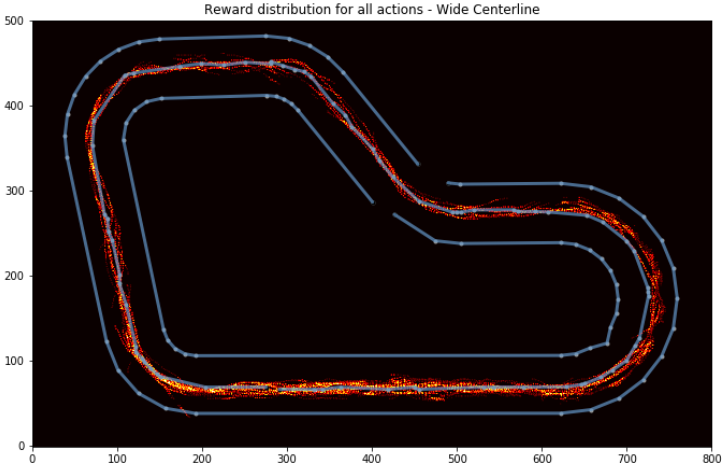
```python
def reward_function(params):
    '''
    Example of rewarding the agent to follow center line
    '''

    # Read input parameters
    track_width = params['track_width']
    distance_from_center = params['distance_from_center']

    # Calculate 3 markers that are at varying distances away from the center line
    marker_1 = 0.1 * track_width
    marker_2 = 0.25 * track_width
    marker_3 = 0.5 * track_width

    # Give higher reward if the car is closer to center line and vice versa
    if distance_from_center <= marker_1:
        reward = 1.0
    elif distance_from_center <= marker_2:
        reward = 0.5
    elif distance_from_center <= marker_3:
        reward = 0.1
    else:
        reward = 1e-3  # likely crashed/ close to off track

    return float(reward)
```

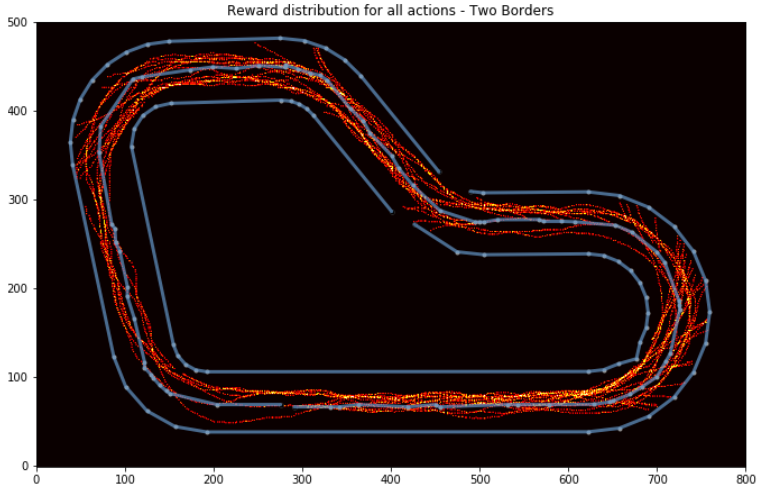

aws

Understand the model: Grad-CAM

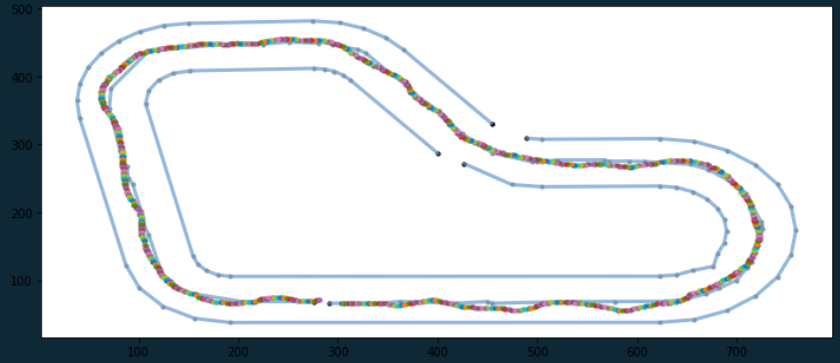


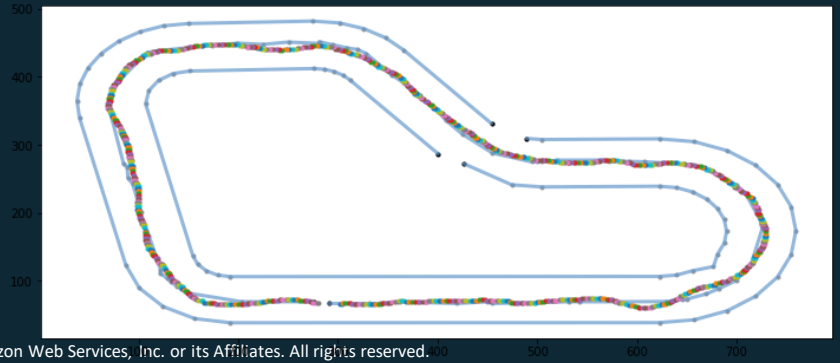
Reward debugging

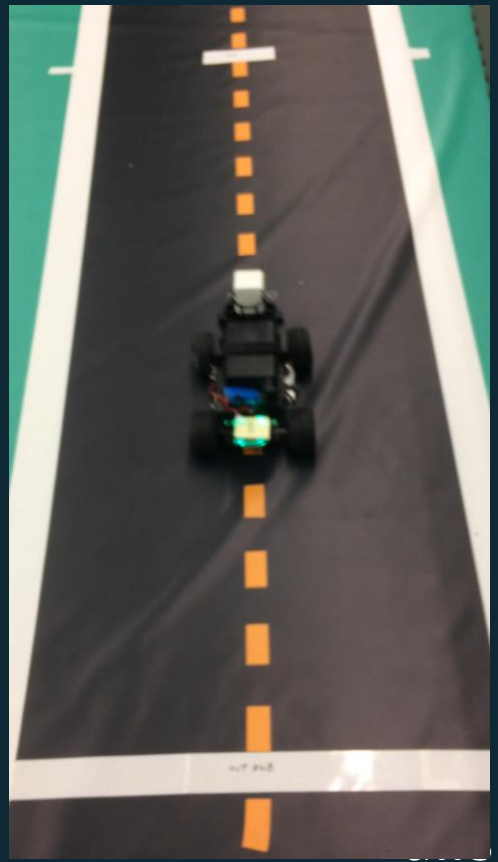




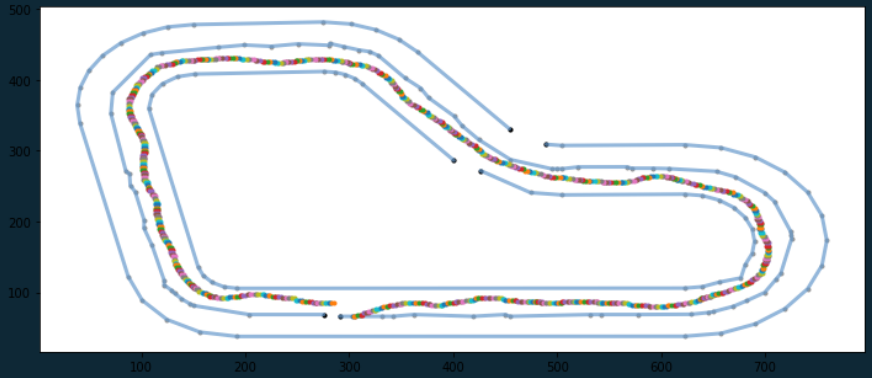
Reward Function – Center Lane

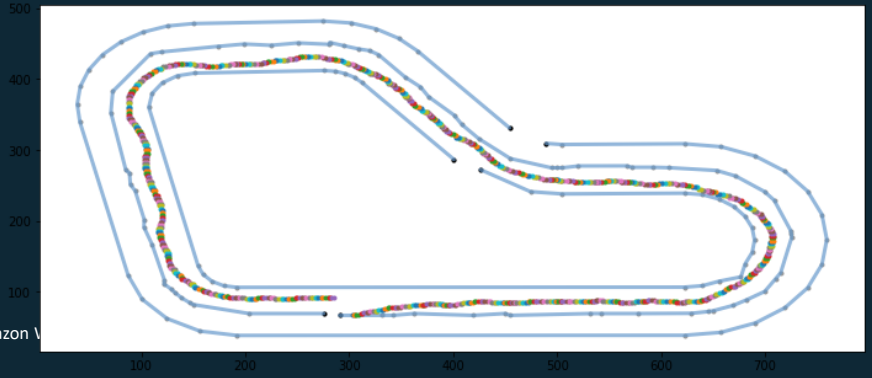


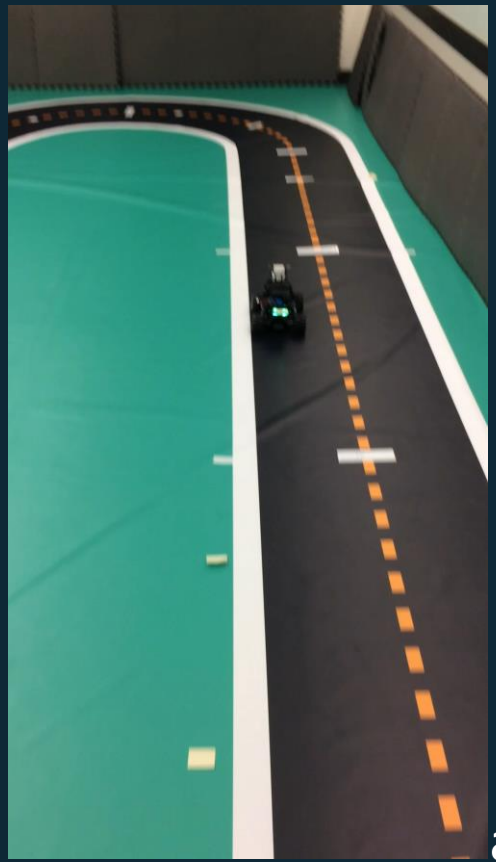




Reward Function – Left Lane

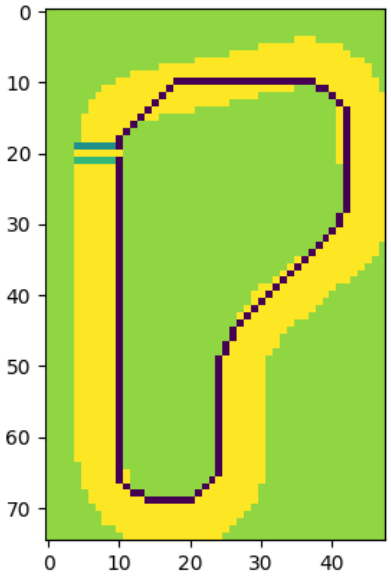




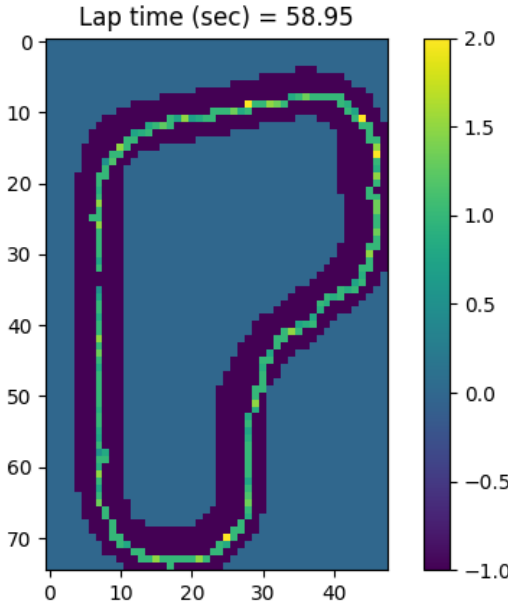


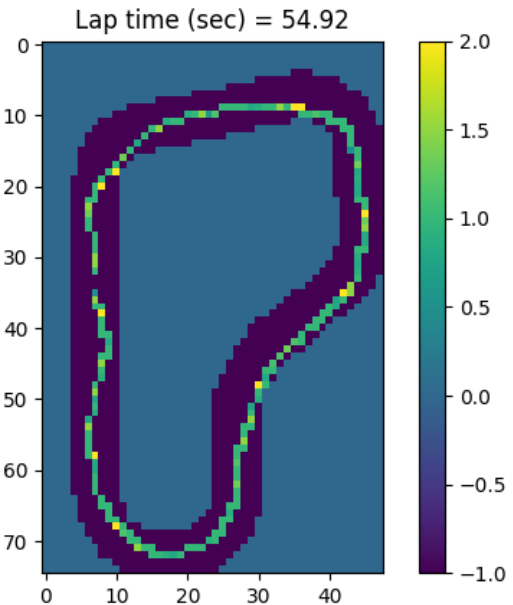
aws

Sim2Real Debug



Optimal Path

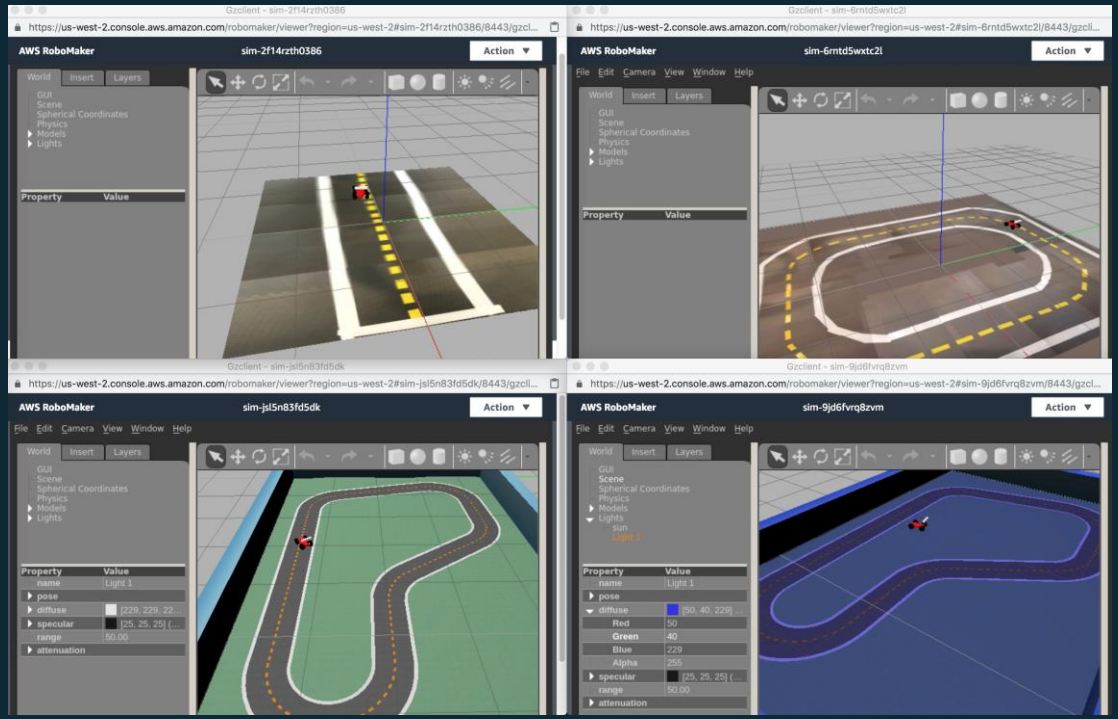




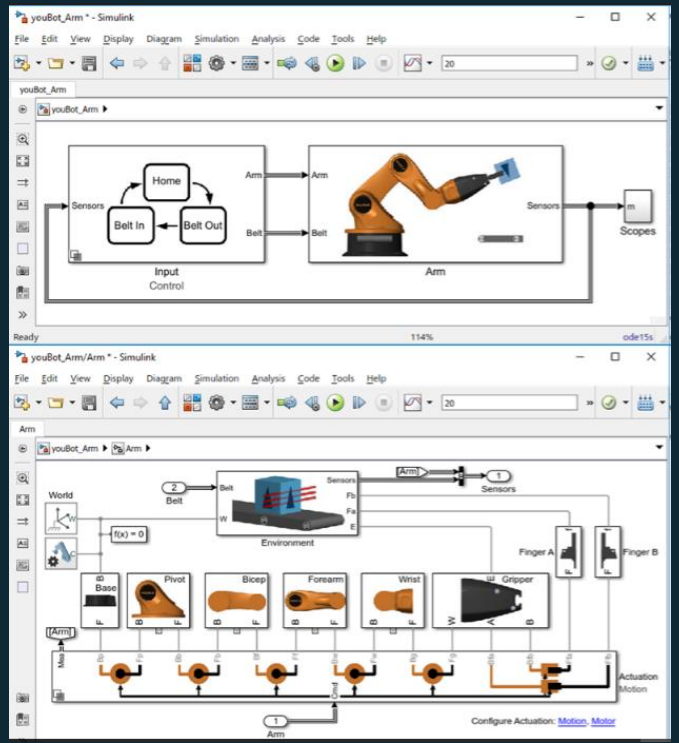
Converged path

aws

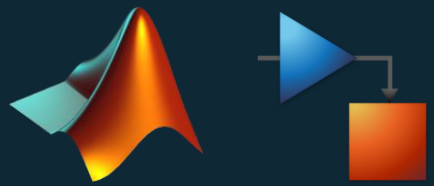
SageMaker RL + AWS RoboMaker accelerate learning for AWS DeepRacer



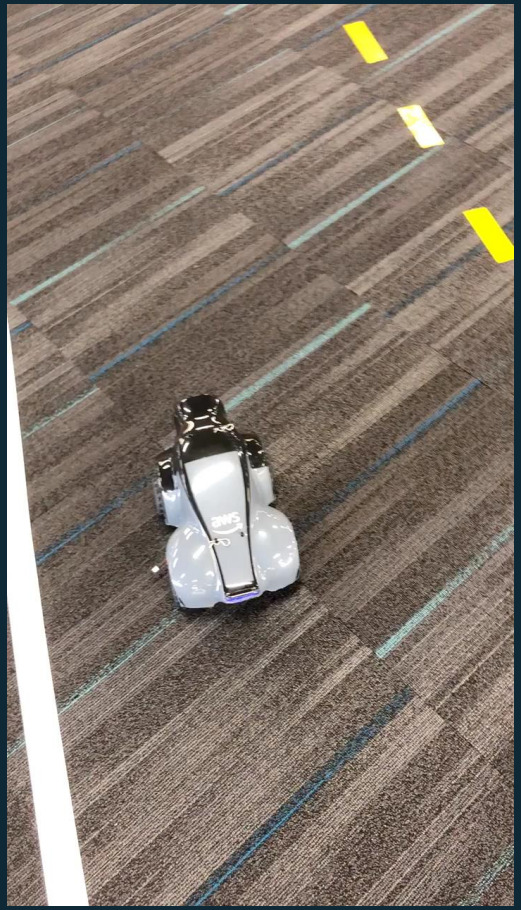
You can use commercial simulation environments



MATLAB® and Simulink® for modeling and simulation



aws



Customers are using Amazon SageMaker RL

GE Healthcare

HONDA

SIXT

mixi

amazon

SyntheticGestalt

Scientific Research by Artificially Intelligent Agents

Tradelegs

aws

DeepRacer Links

DeepRacer Simulator:

https://github.com/aws-robotics/aws-robomaker-sample-application-deepracer

SageMaker RL:
https://github.com/awslabs/amazon-sagemaker-examples/tree/master/reinforcement_learning/rl_deepracer_robomaker_coach_gazebo

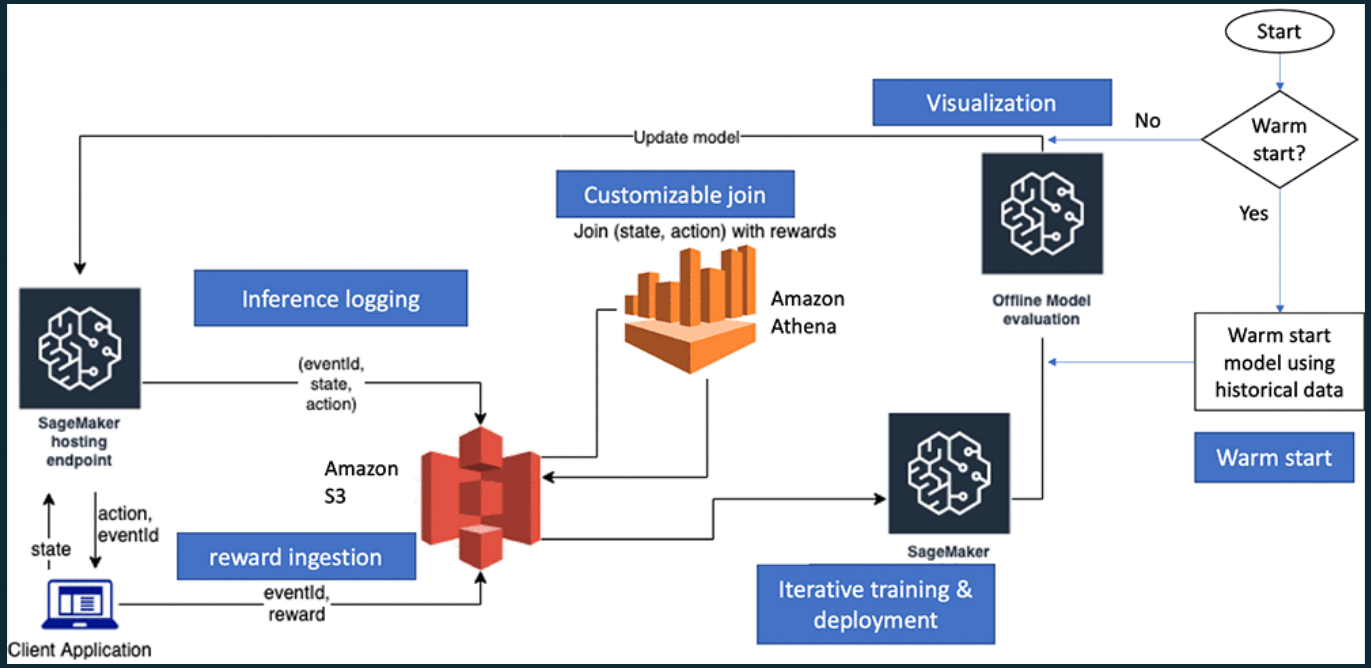
aws

Let’s look at some other examples

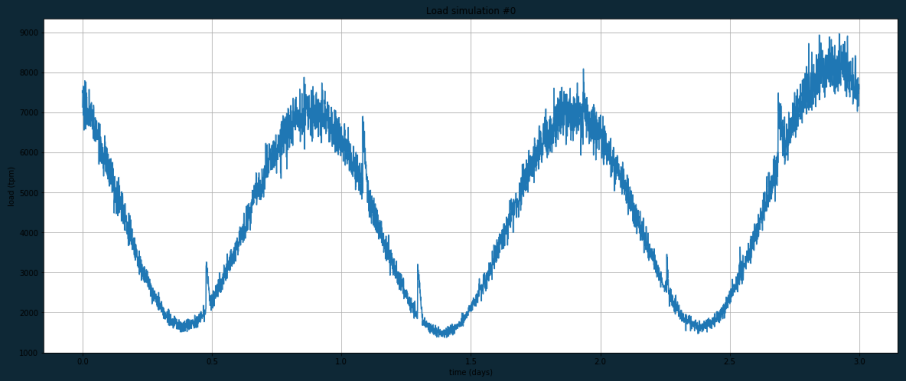
These examples and many others are available today in Amazon SageMaker RL

Amazon SageMaker RL contextual bandits solution

personalized web services (content layout, ads, search, product recommendations, etc.) are continuously faced with decisions to make, often based on some contextual information



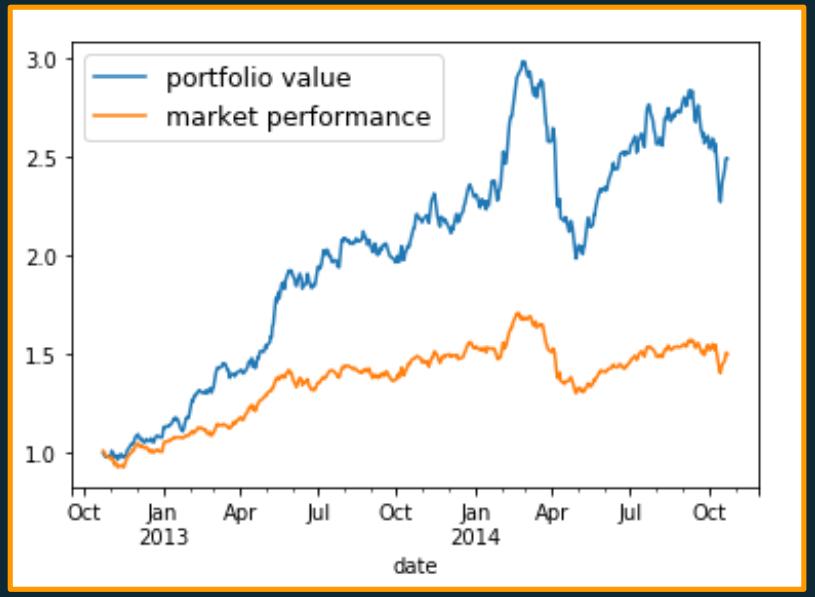
Autoscaling



Objective	Adapt instance capacity to load profile
State	Current load, failed jobs, active machines
Action	Remove or add machines
Reward	Positive for successful transactions
	High penalty for losing transactions

https://aws.amazon.com/blogs/aws/amazon-sagemaker-rl-managed-reinforcement-learning-with-amazon-sagemaker/

Financial portfolio management



Objective	Maximize the value of a financial portfolio
State	Current stock portfolio, price history
Action	Buy, Sell stocks
Reward	Positive when return is positive
	Negative when return is negative

https://github.com/awslabs/amazon-sagemaker-examples/tree/master/reinforcement_learning/rl_portfolio_management_coach_customEnv

aws

Thank You