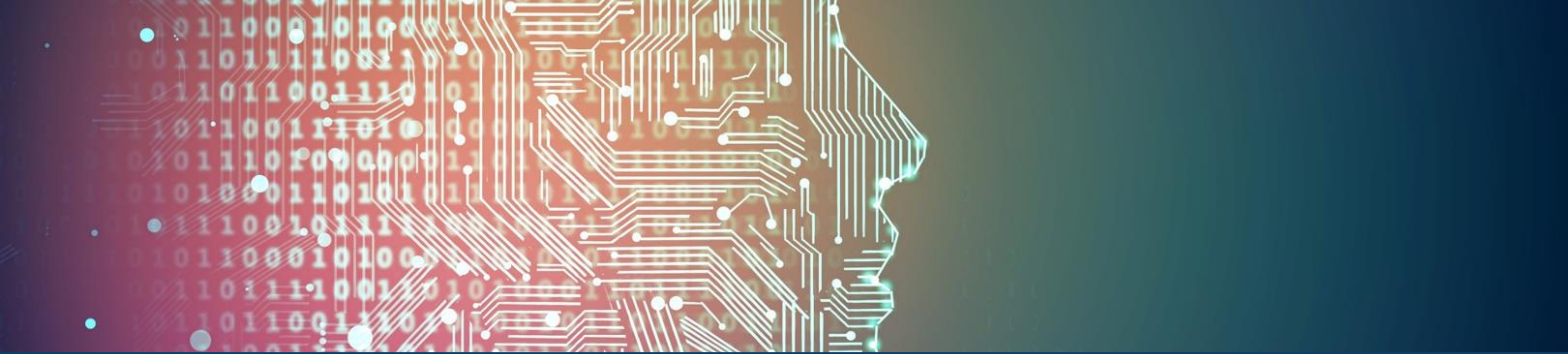


MATLAB과 함께하는 딥러닝 4주 완성 부트캠프

김종남 부장
Application Engineer @ MathWorks
calebkim@mathworks.com

세션4. MATLAB으로 시작하는 강화학습

MATLAB과 함께하는 딥러닝 4주 완성 부트캠프

김종남 부장
Application Engineer @ MathWorks
calebkim@mathworks.com

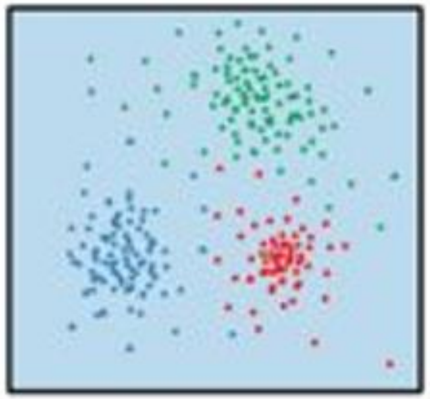
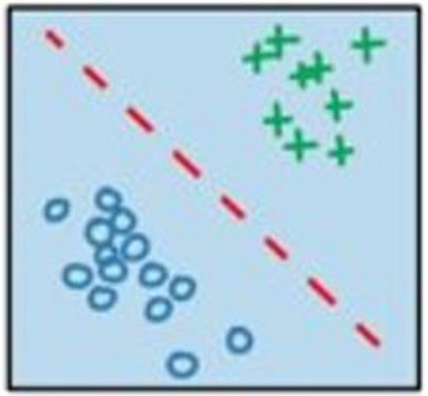
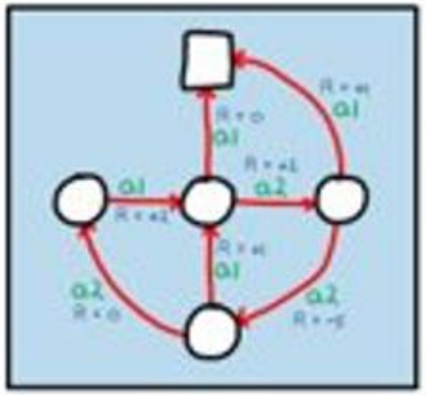
MathWorks®
Accelerating the pace of engineering and science

Agenda

- How to Train a Robot to Walk
  - What is reinforcement learning?
  - Overview of using a traditional controls approach
  - Applying the reinforcement learning workflow to train the robot with Reinforcement Learning Designer

Reinforcement Learning: A Subset of Machine Learning

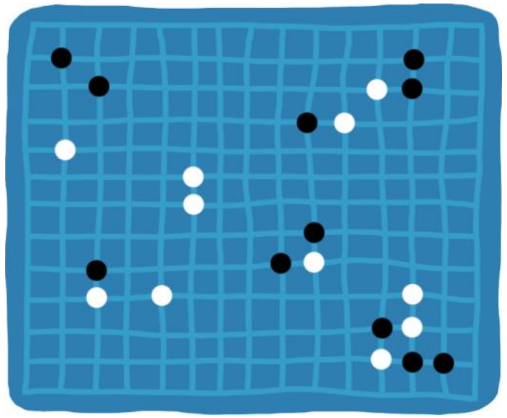
machine learning

unsupervised learning	supervised learning	reinforcement learning
[unlabeled data]	[labeled data]	[interaction data]
		
clustering	classification and regression	control and decision making

Reinforcement learning:

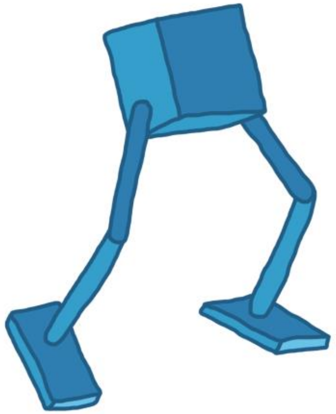
- Learning a **behavior** or accomplishing a **task** through trial & error
  [*interaction*]
- Complex problems typically need deep models
  [*Deep Reinforcement Learning*]

Reinforcement Learning Applications



video games

autonomous vehicles

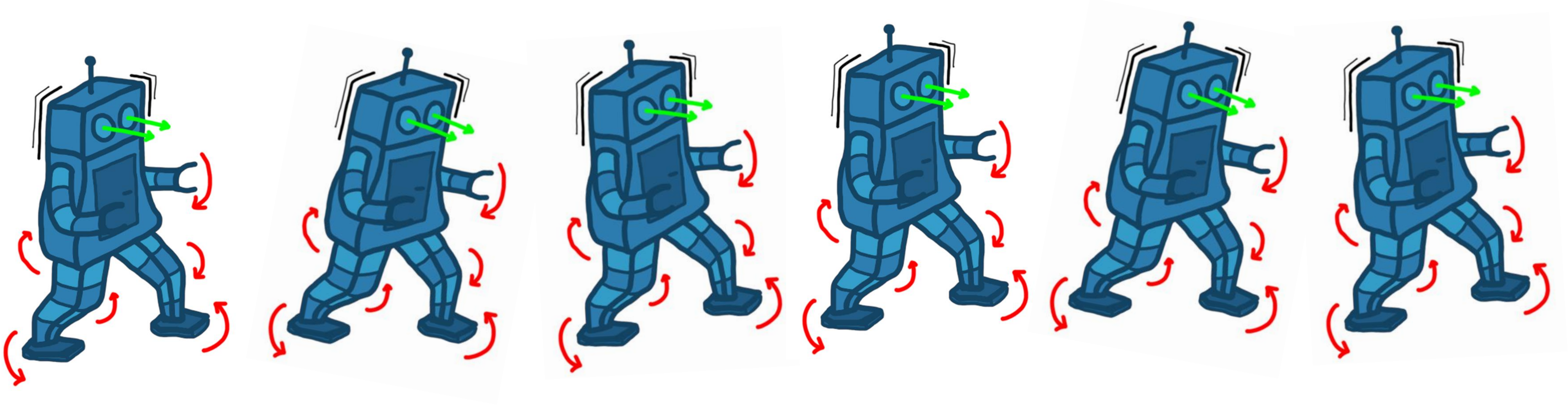


robotics

controls

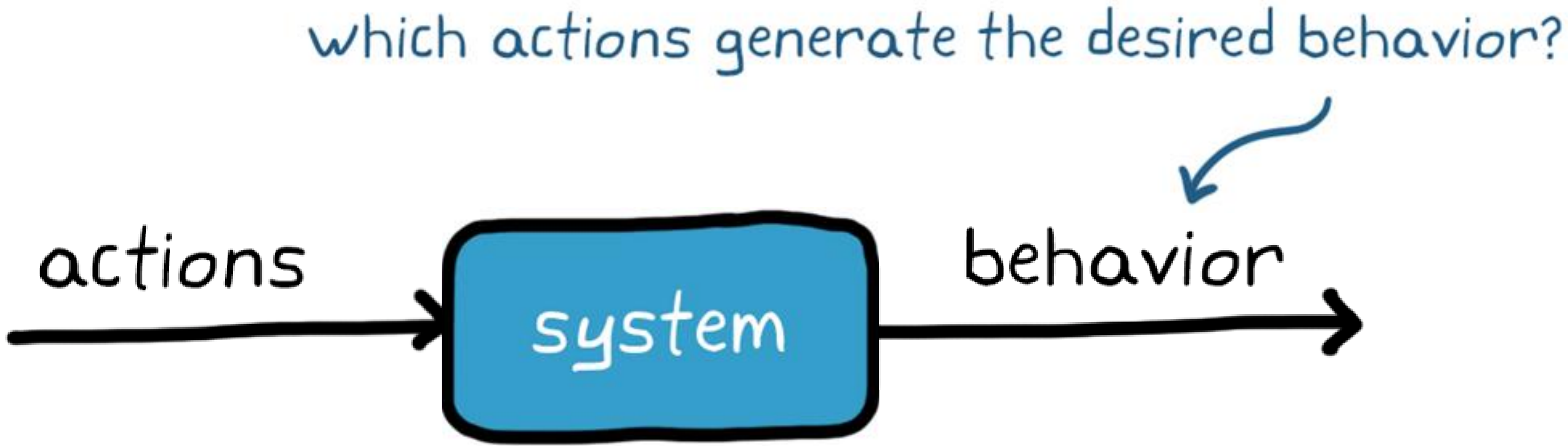
How do We Train a Robot to Walk

Goal: Train a robot to walk a straight line

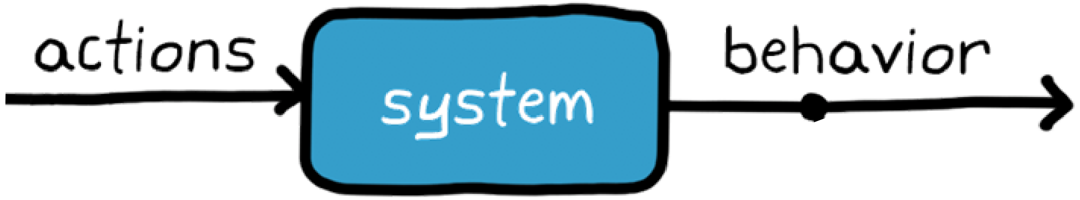


What sequence of motor commands do we need to make the robot walk?

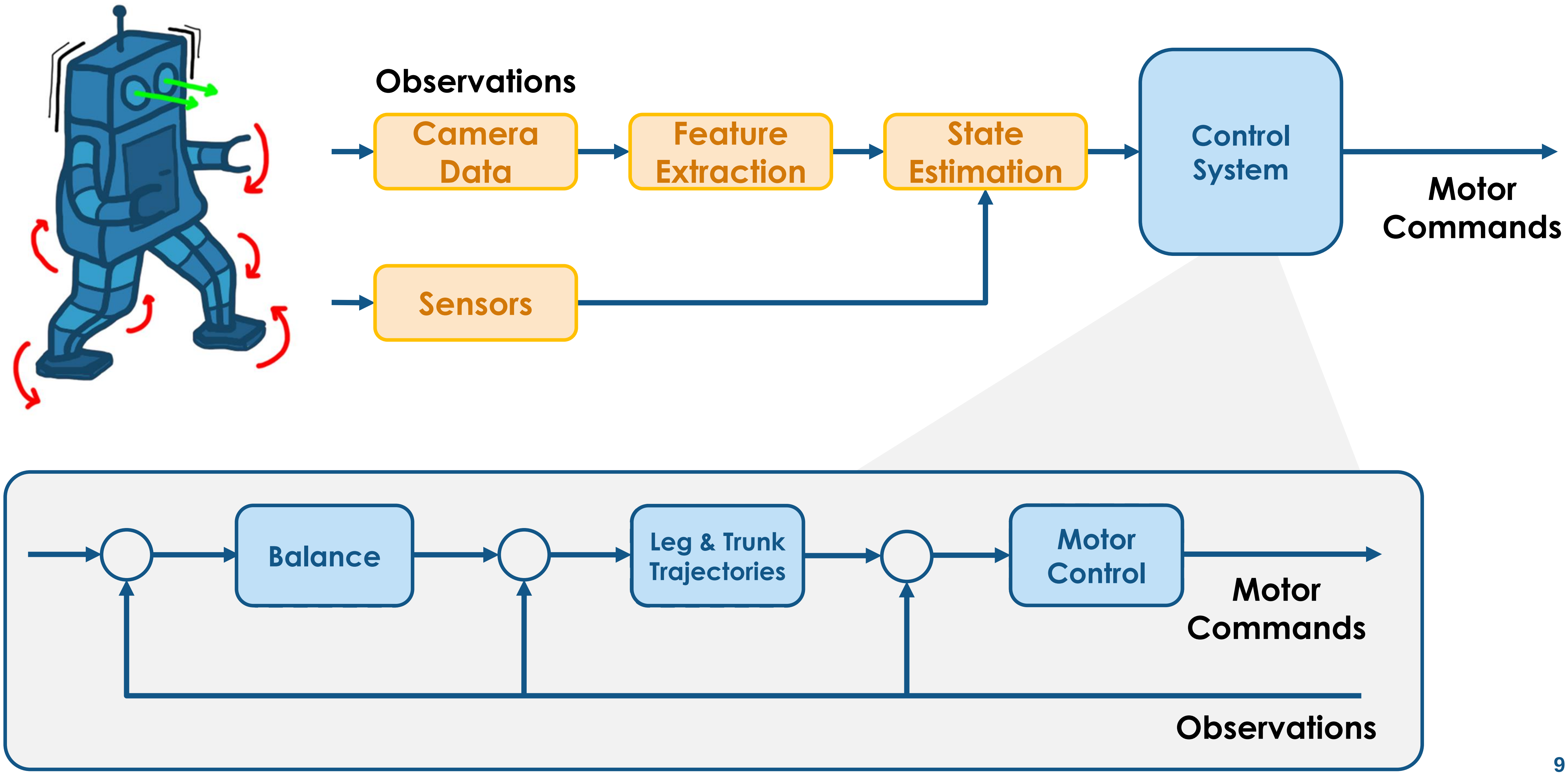
The goal of control



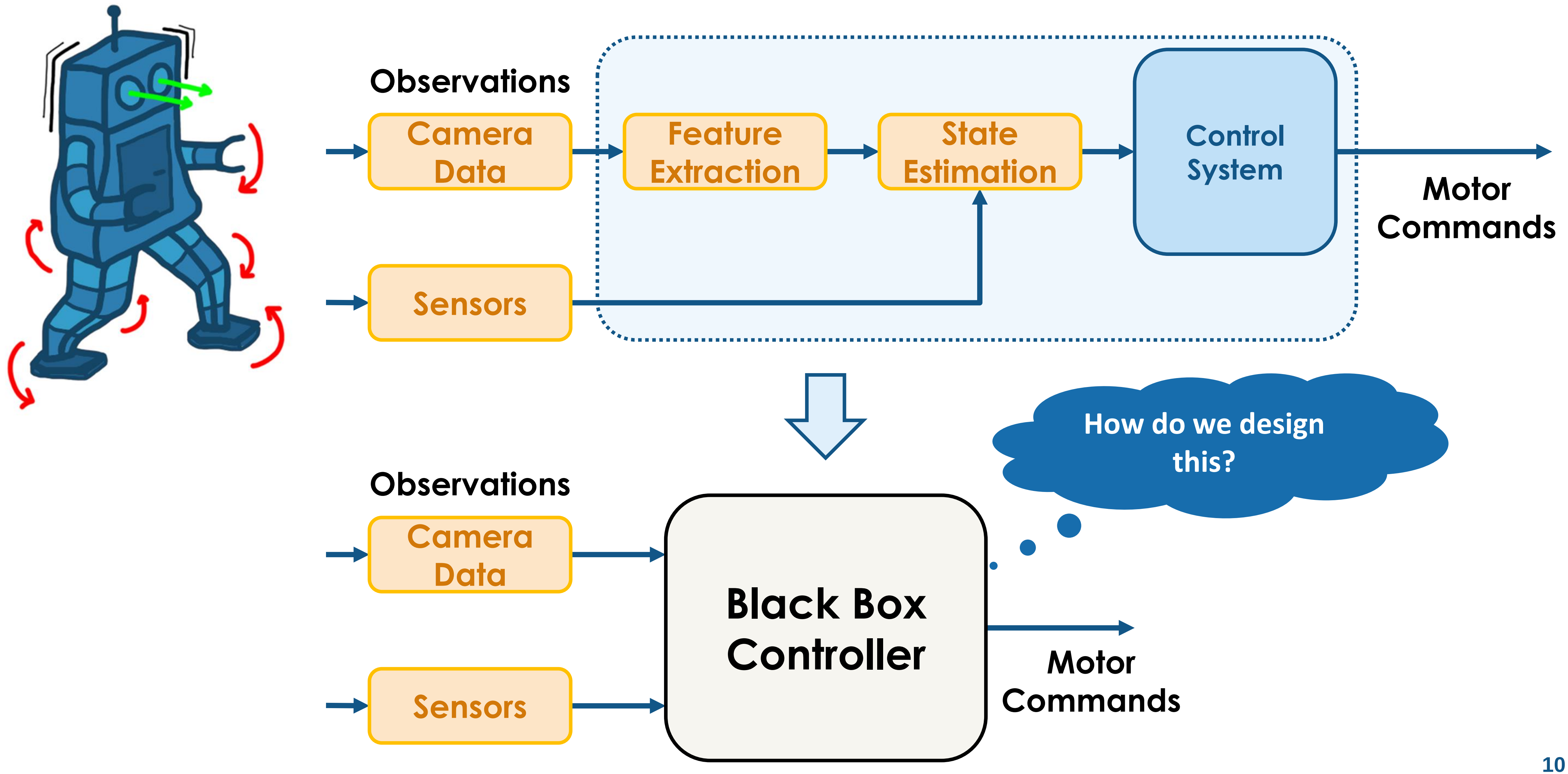
The goal of control



A walking robot – a traditional controls approach



A walking robot – an alternative approach



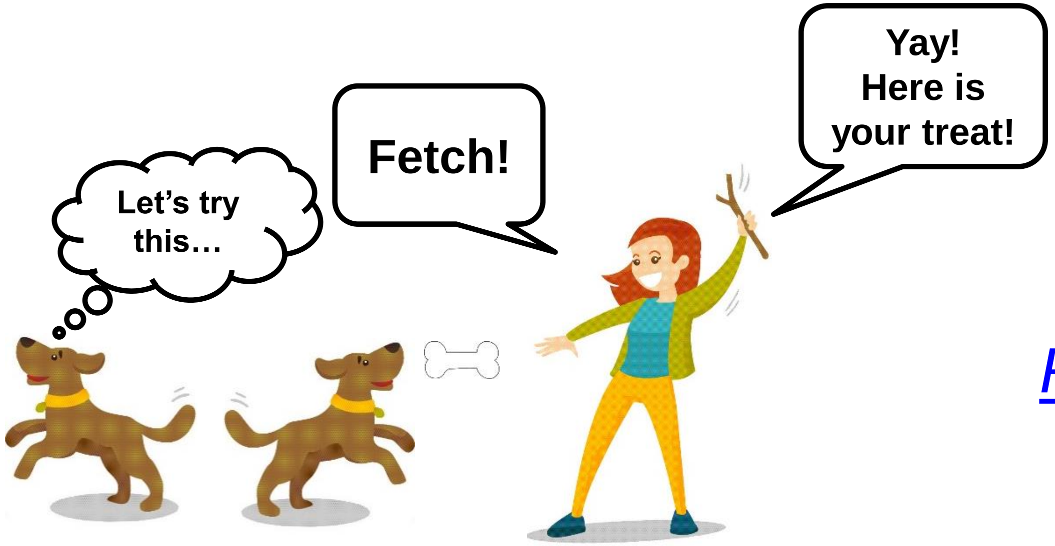
What Is Reinforcement Learning?

"

Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical reward signal.

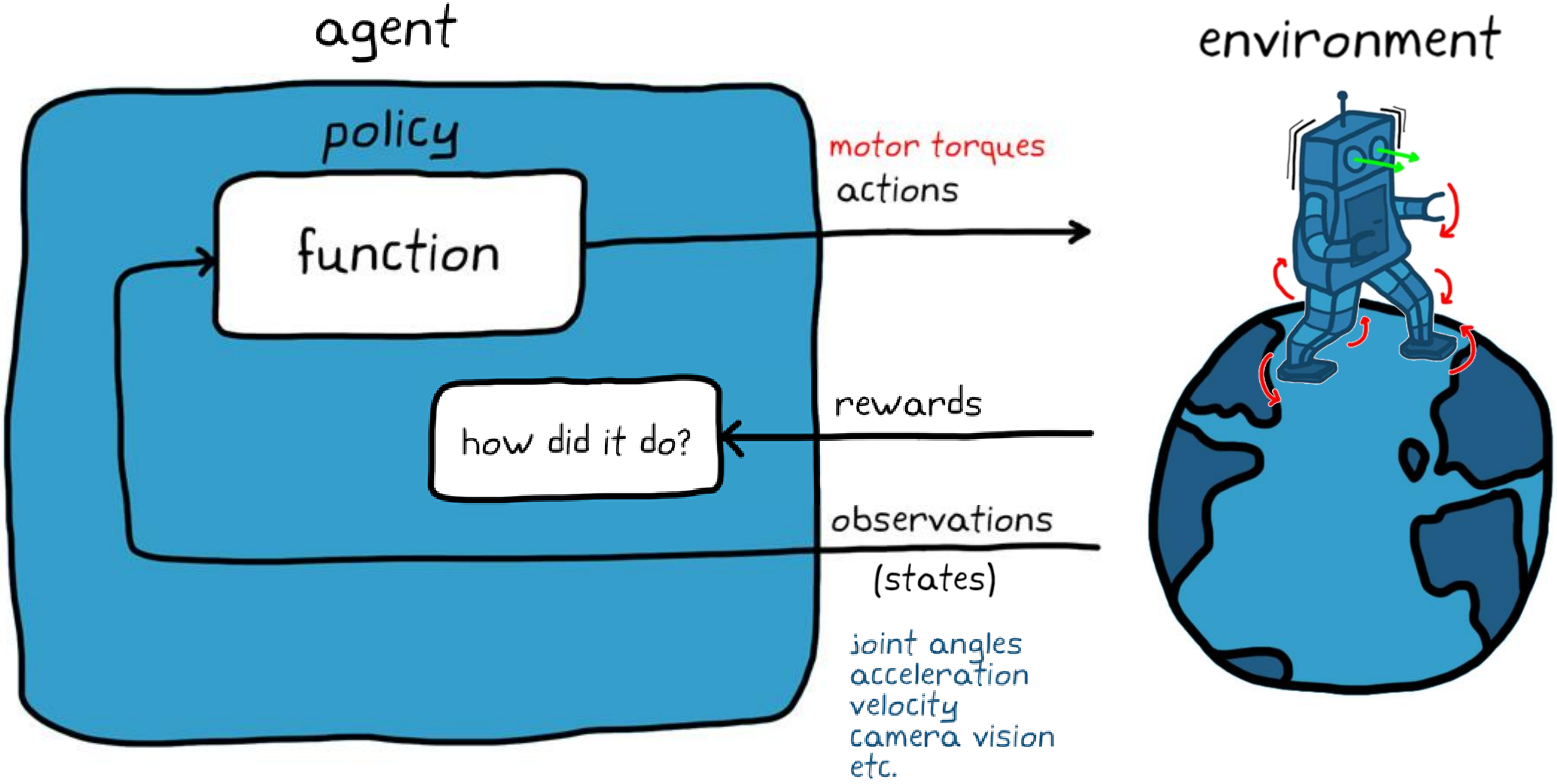
The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them.

"



Sutton and Barto,
Reinforcement Learning: An Introduction

Some Reinforcement Learning Terminology



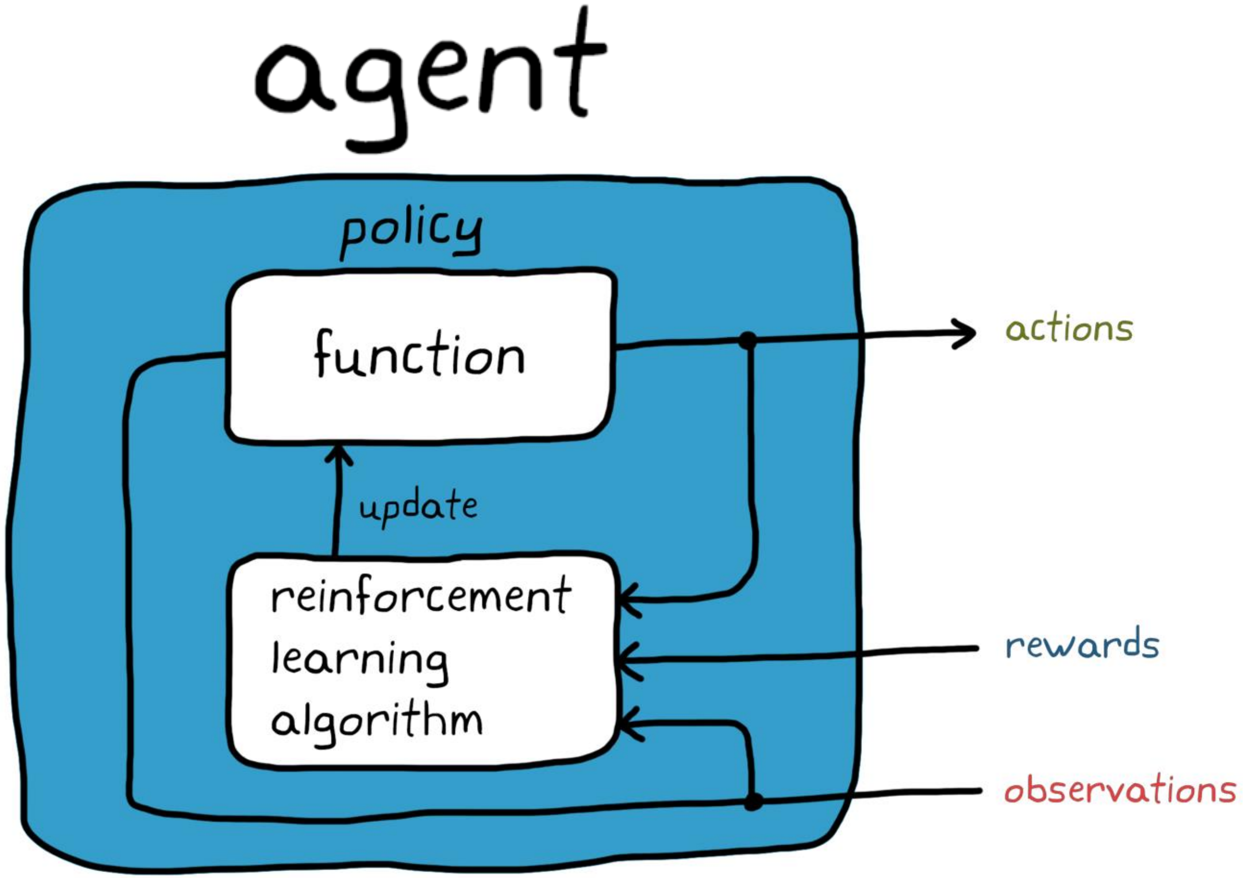
Learning the Optimal Policy

Policy

function that maps observations to actions

Reinforcement Learning Algorithm

optimization method used to find the optimal policy that maximizes accumulative long-term reward

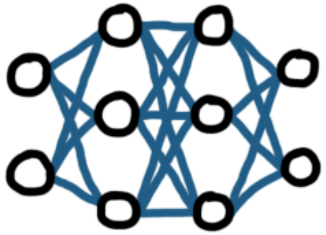


Reinforcement Learning Workflow

environment

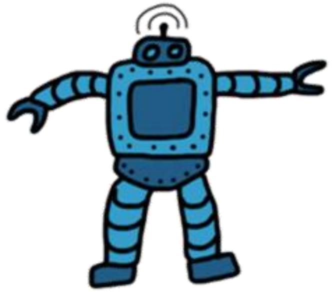
reward

policy



training

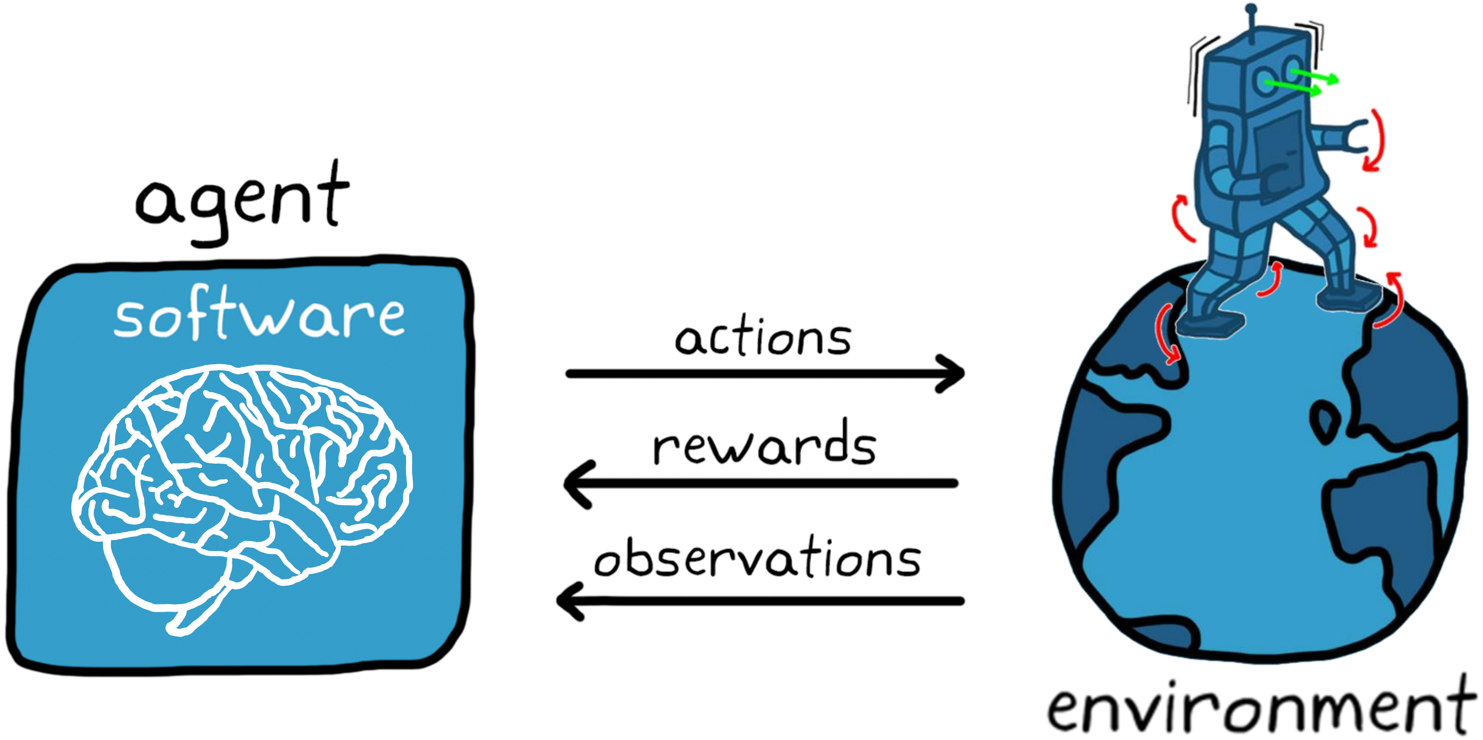
deploy



Reinforcement Learning Workflow

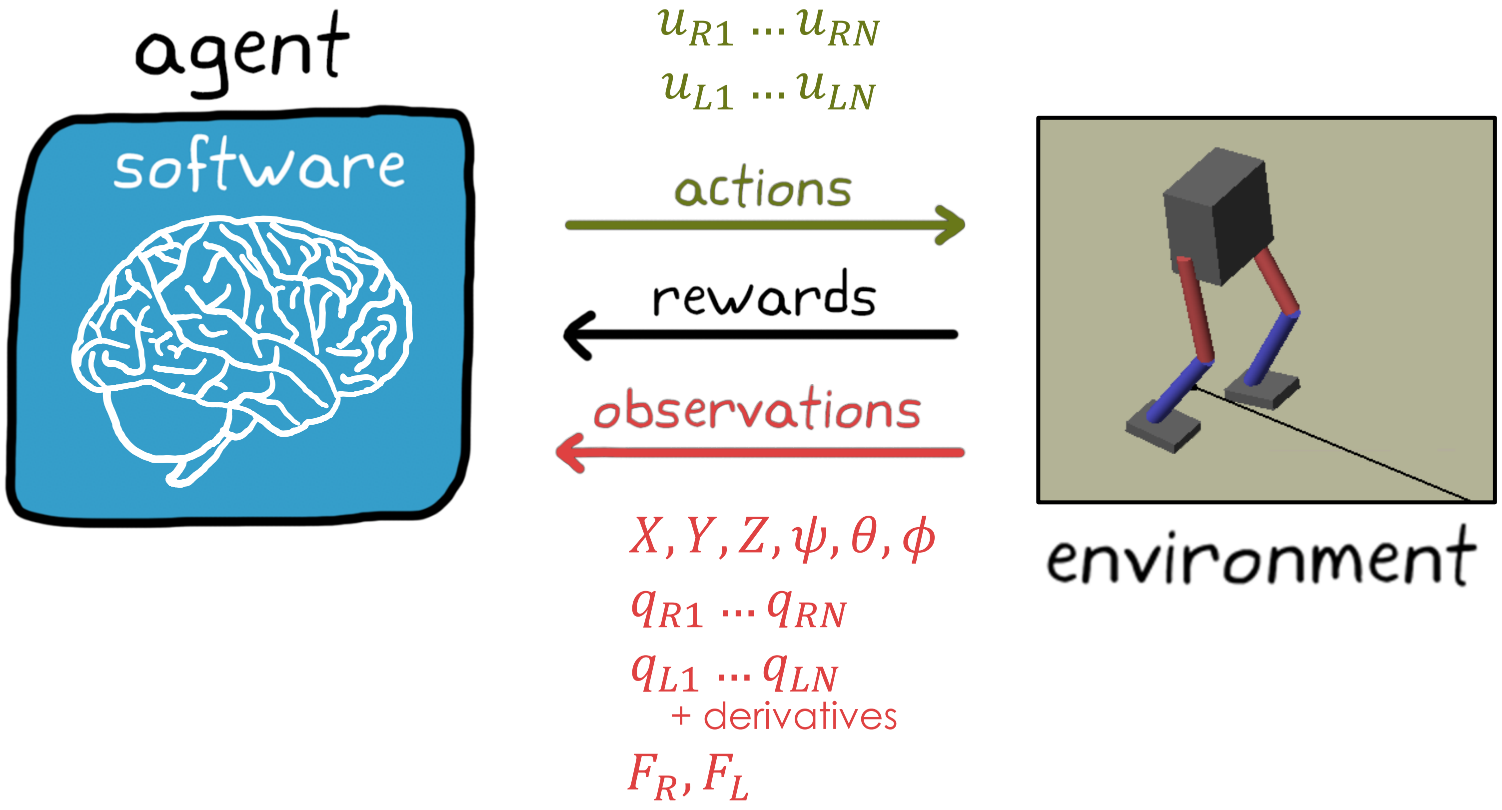
Environment

- Everything outside of an agent

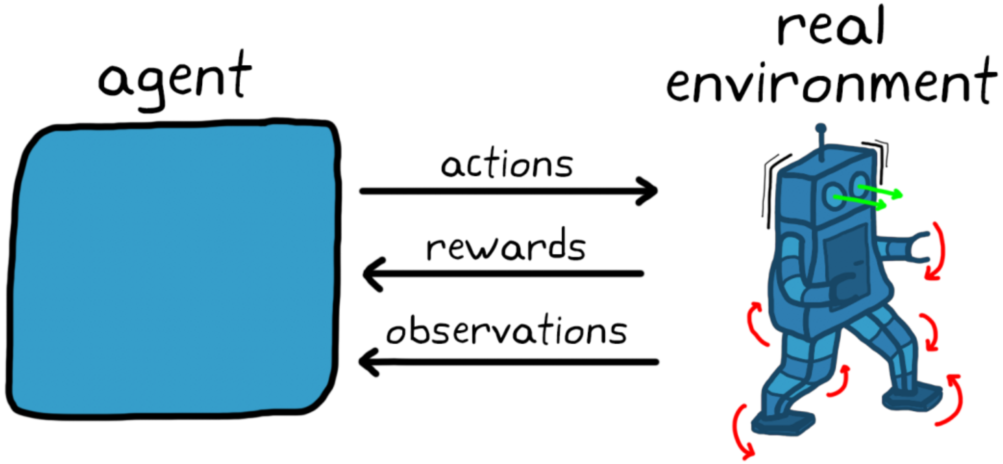


Environment

- Everything outside of an agent

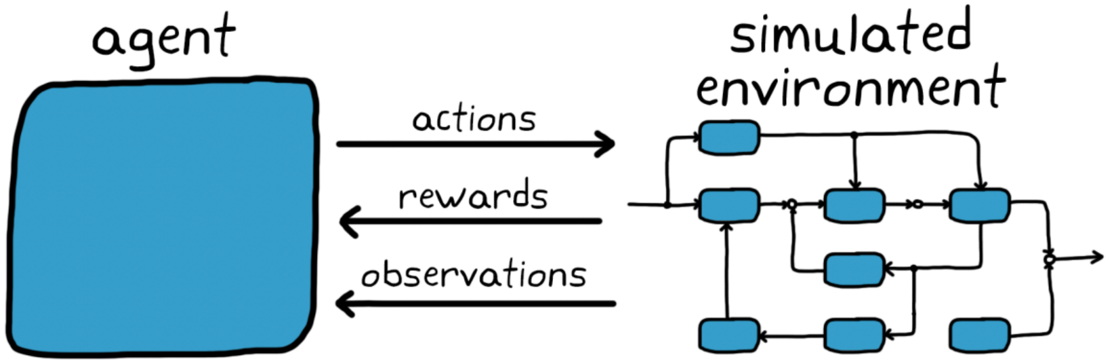


Real vs Simulated Environments



☺ Accuracy

☹ Risk



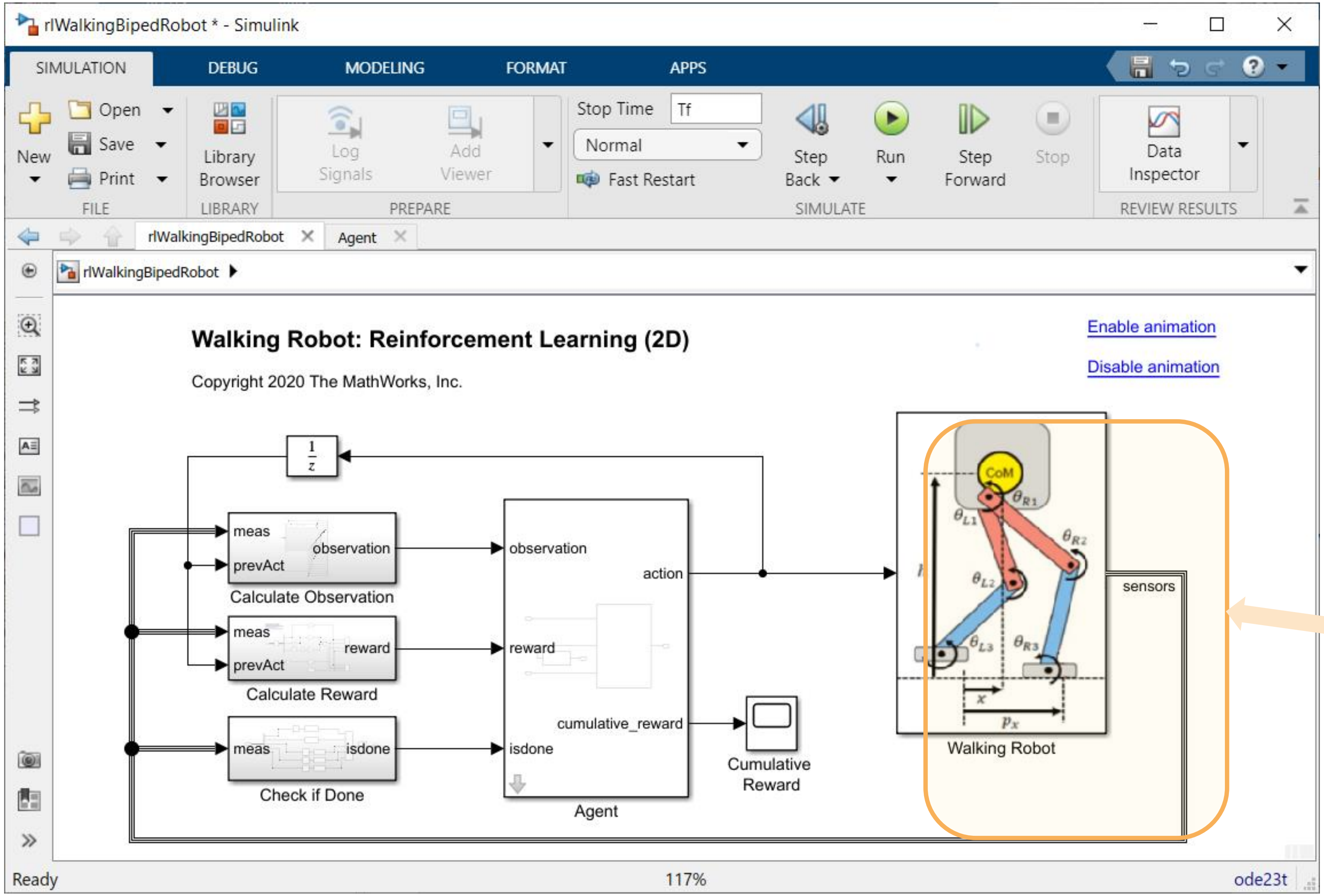
☺ Training speed

☺ Flexible simulated conditions

☺ Safety

☹ Model inaccuracies

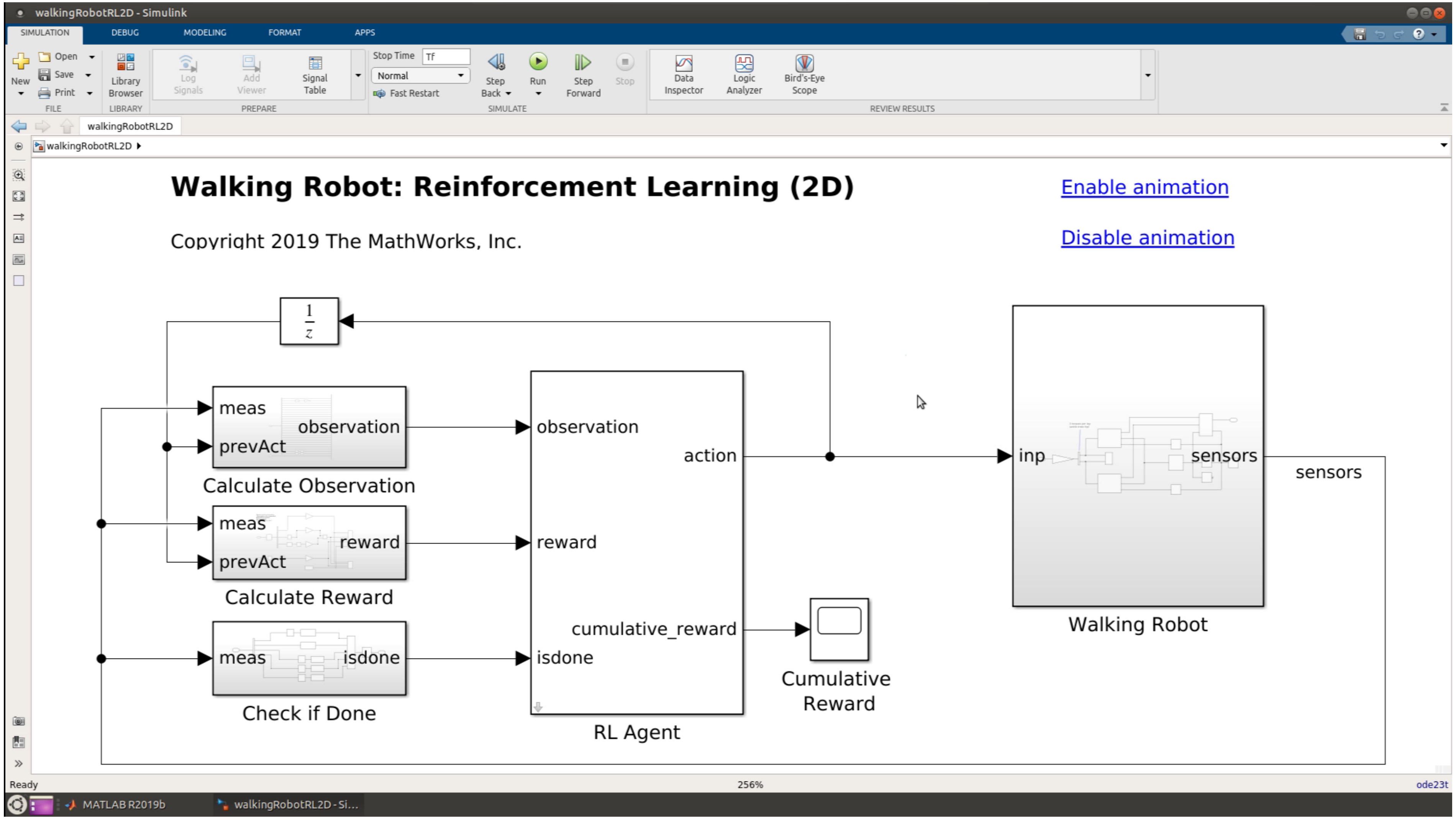
Define Simulated Environment



Physical modeling of robot dynamics and contact forces using Simscape

https://www.mathworks.com/help/reinforcement-learning/ug/train-biped-robot-to-walk-using-reinforcement-learning-agents.html

Environment - Simulink



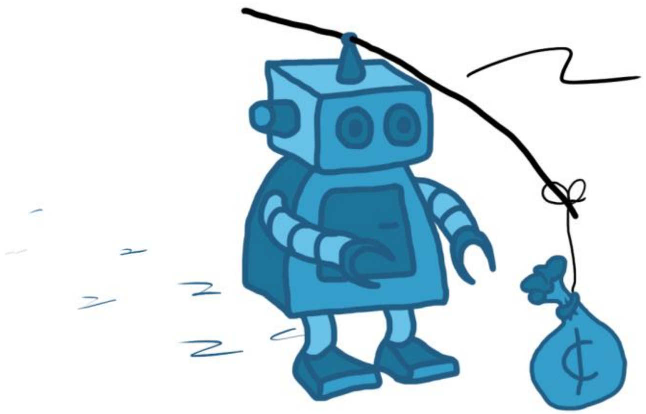
Reinforcement Learning Workflow

Reward

A function that outputs a **scalar number** that represents the immediate **"goodness"** of an agent being in a particular **state** and taking a particular **action**.

reward = function (state, action)

Defining the Reward



$$r_t = v_x - 3y^2 - 50\hat{z}^2 + 25\frac{T_s}{T_f} - 0.02\sum_i {u^i_{t-1}}^2$$

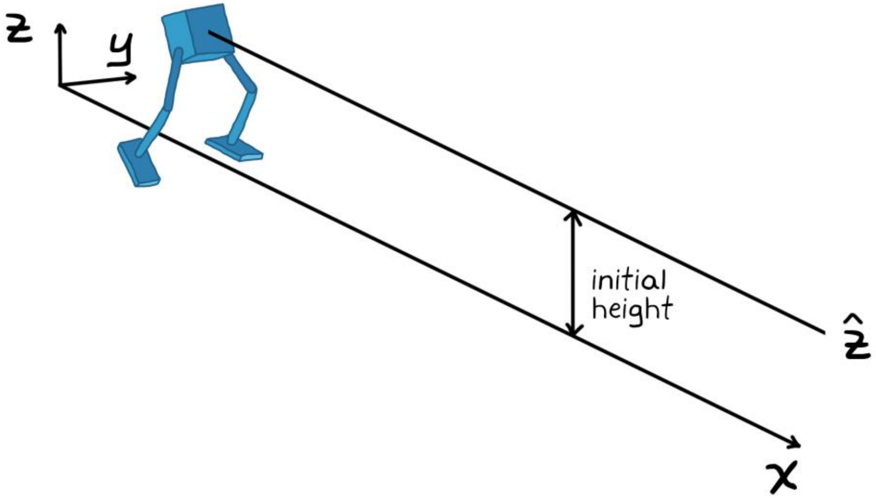
minimize actuator effort

walk as long as possible

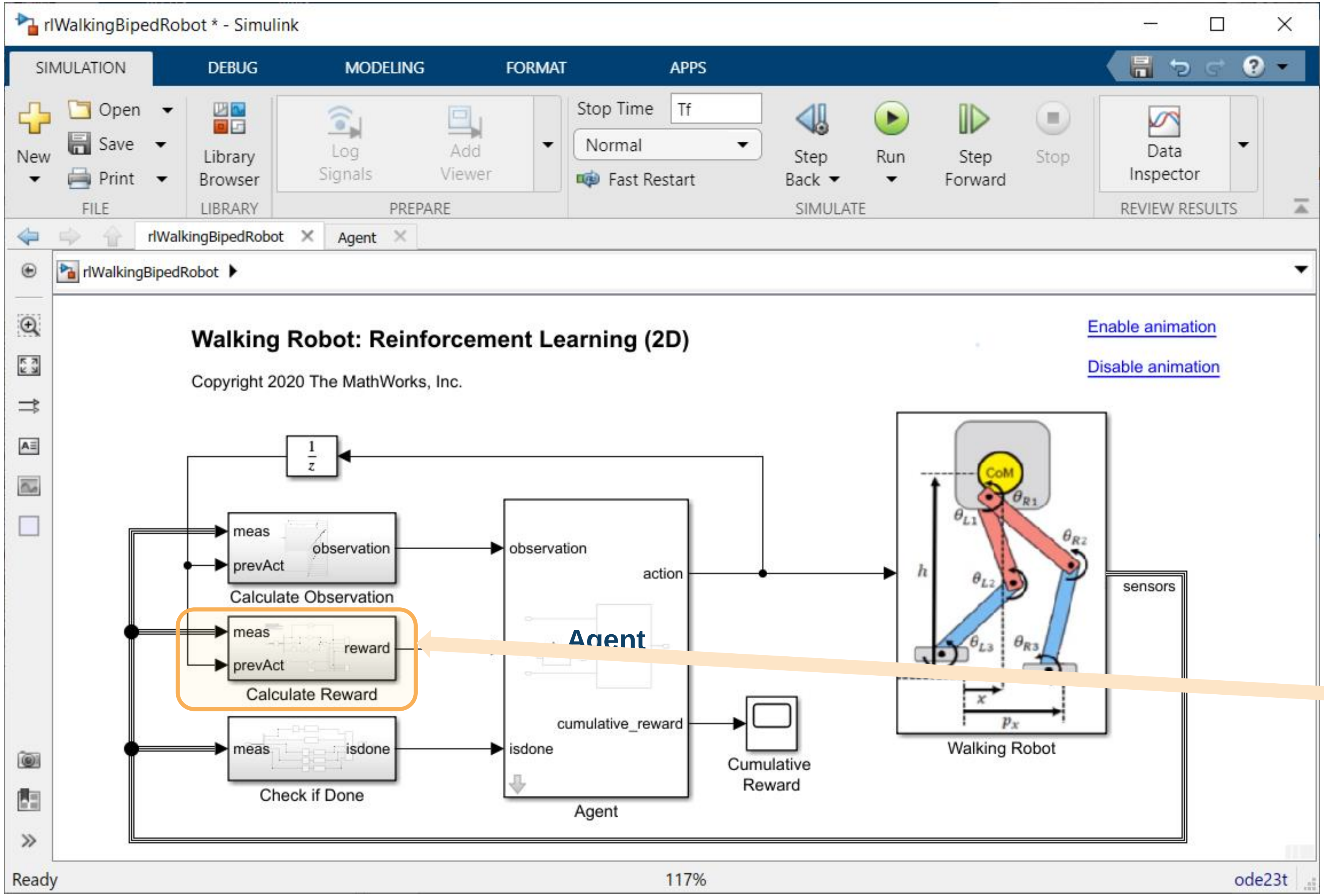
keep trunk at initial height

don't stray from path

forward velocity

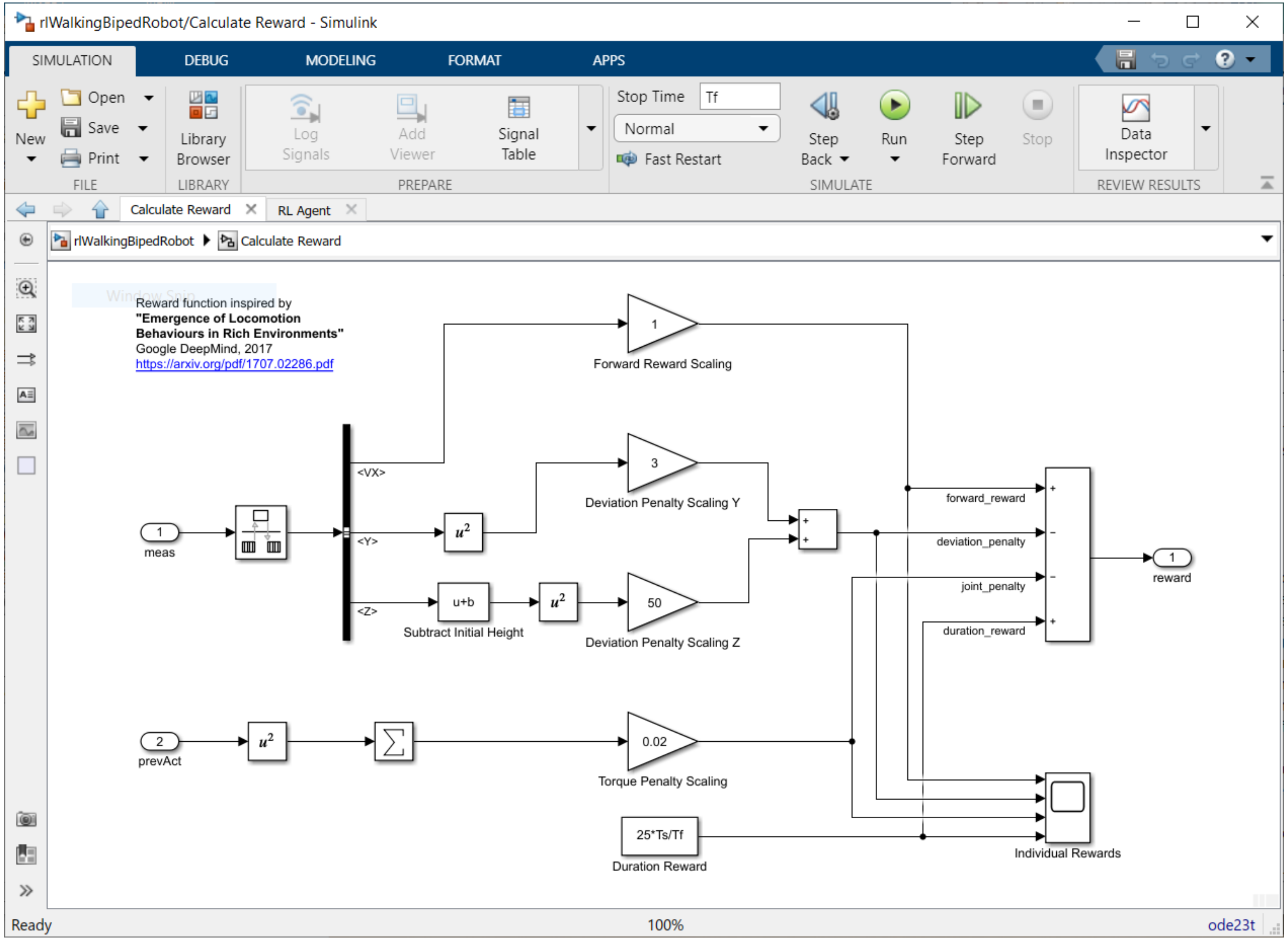


Defining the Reward



Reward defines task to learn

Defining the Reward



$$r_t = v_x - 3y^2 - 50\hat{z}^2 + 25\frac{T_s}{T_f} - 0.02\sum_i {u^i_{t-1}}^2$$

minimize actuator effort

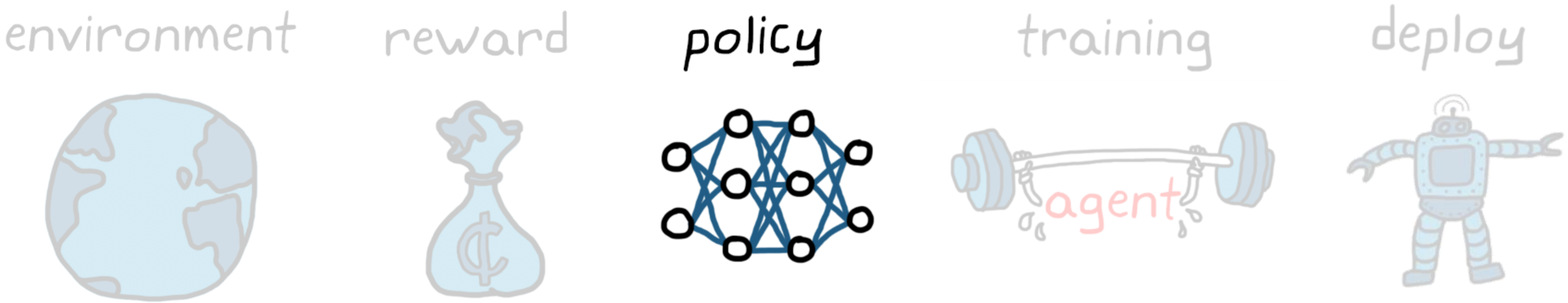
walk as long as possible

keep trunk at initial height

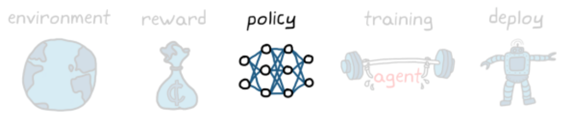
don't stray from path

forward velocity

Reinforcement Learning Workflow



The Agent


environment
reward
policy
training
agent
deploy


agent
software
actions
rewards
observations
environment

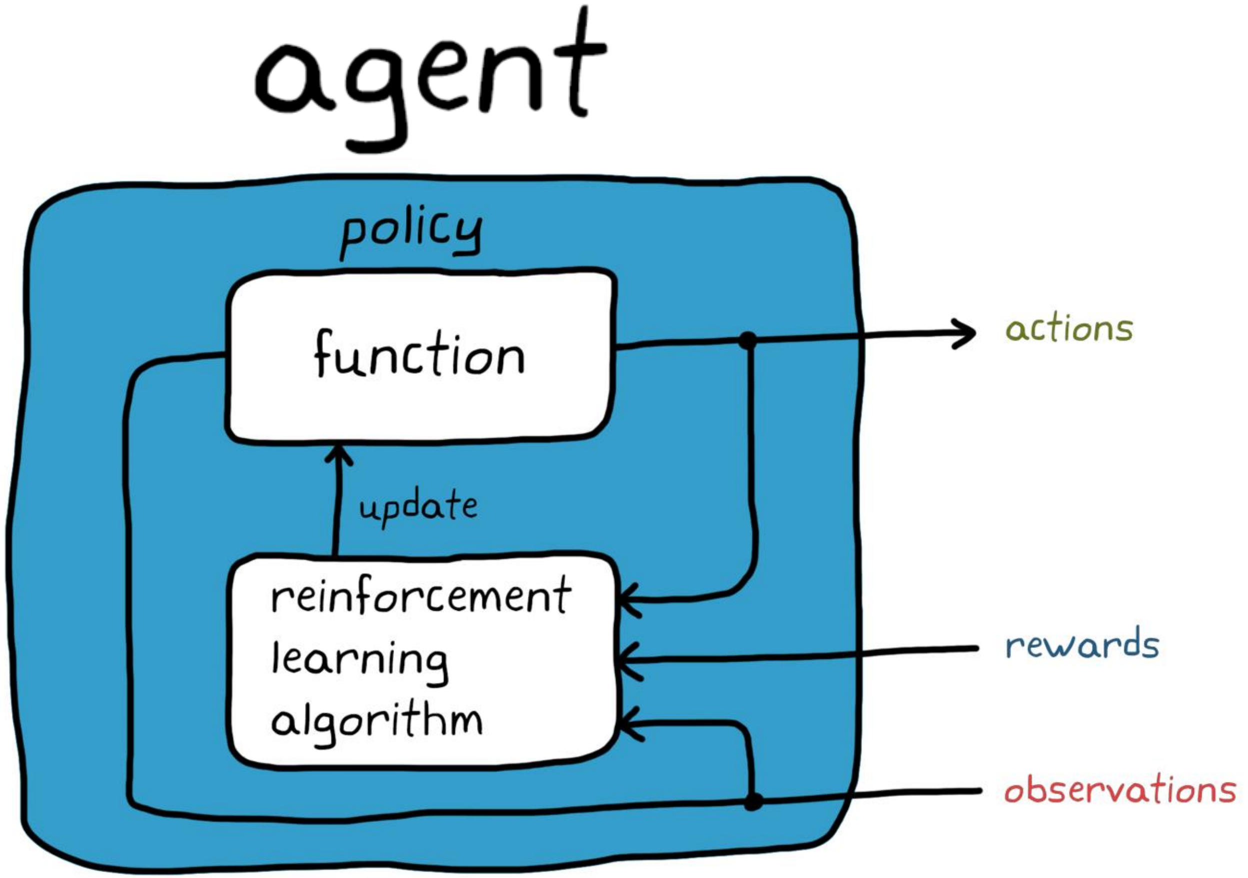
Learning the Optimal Policy

Policy

function that maps observations to actions

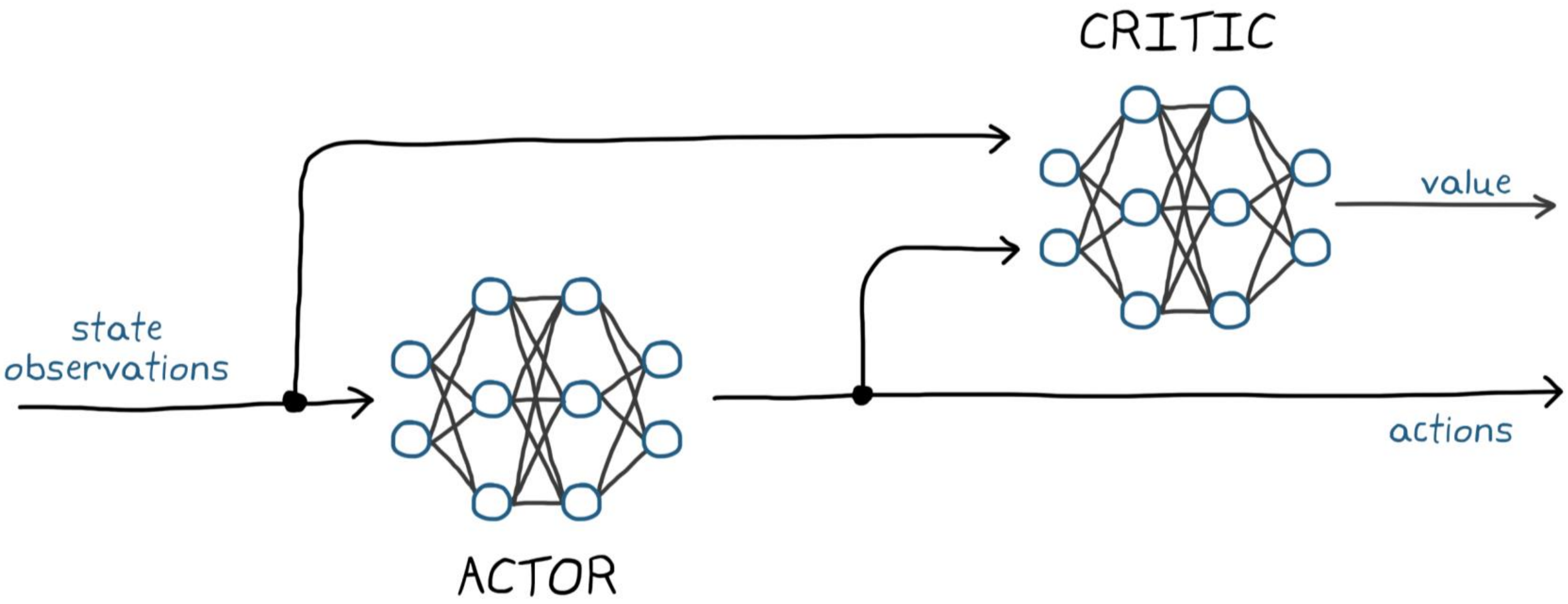
Reinforcement Learning Algorithm

optimization method used to find the optimal policy



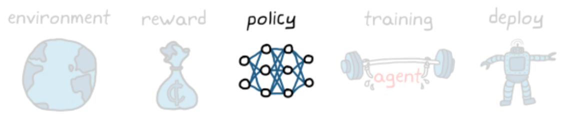
Actor-Critic Methods

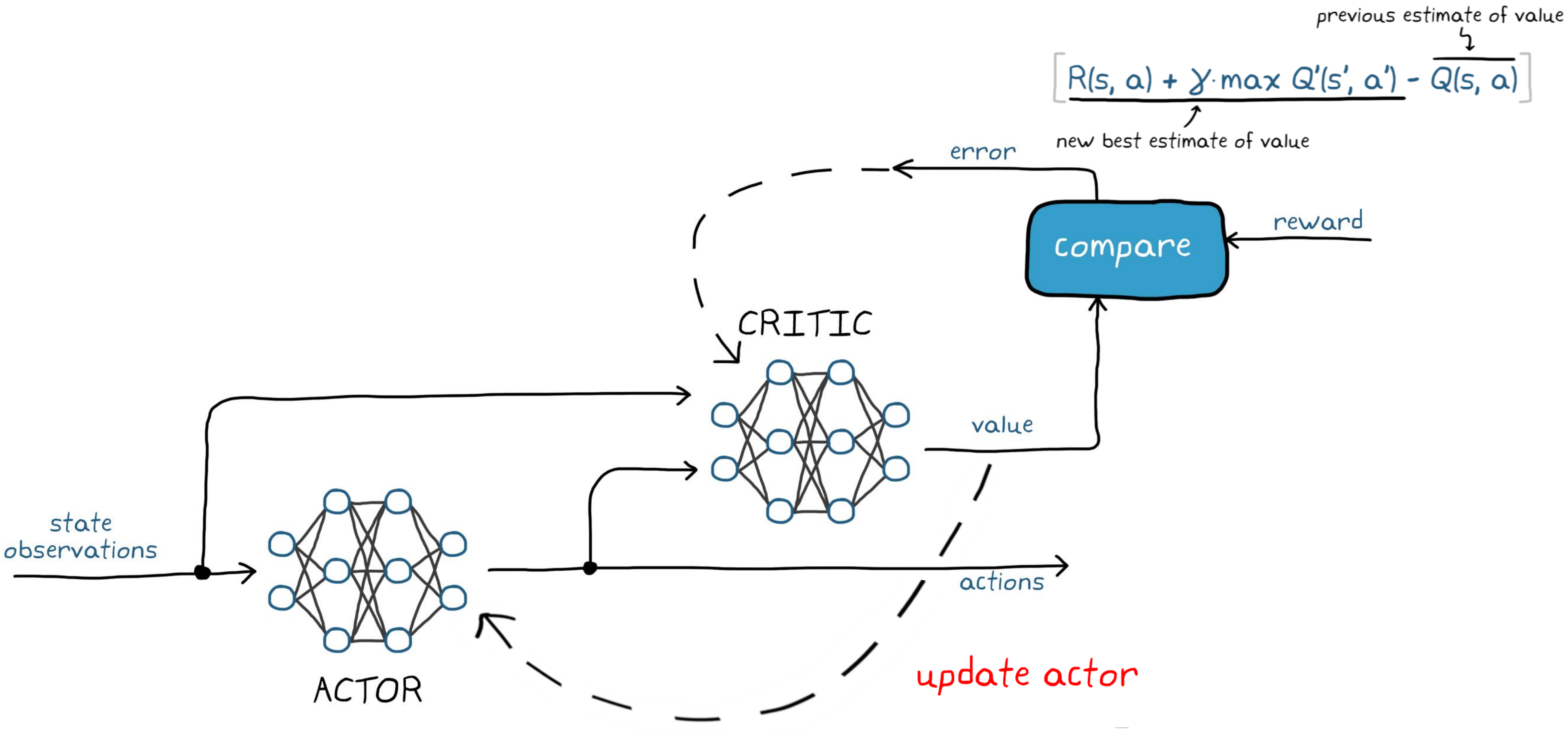
- **Two** neural networks (typically) are **simultaneously tuned** during training:
  - The actor tries to learn the best action at each state
  - The critic tries to
    - estimate the value of each state/state & action the actor takes
    - critique/guide the actor's choices



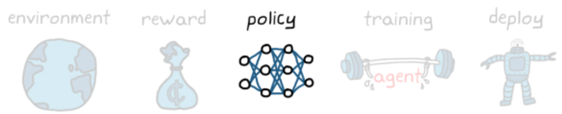
value the **total reward** an agent expects to receive **from a state and onwards into the future**

Actor-Critic Training Cycle



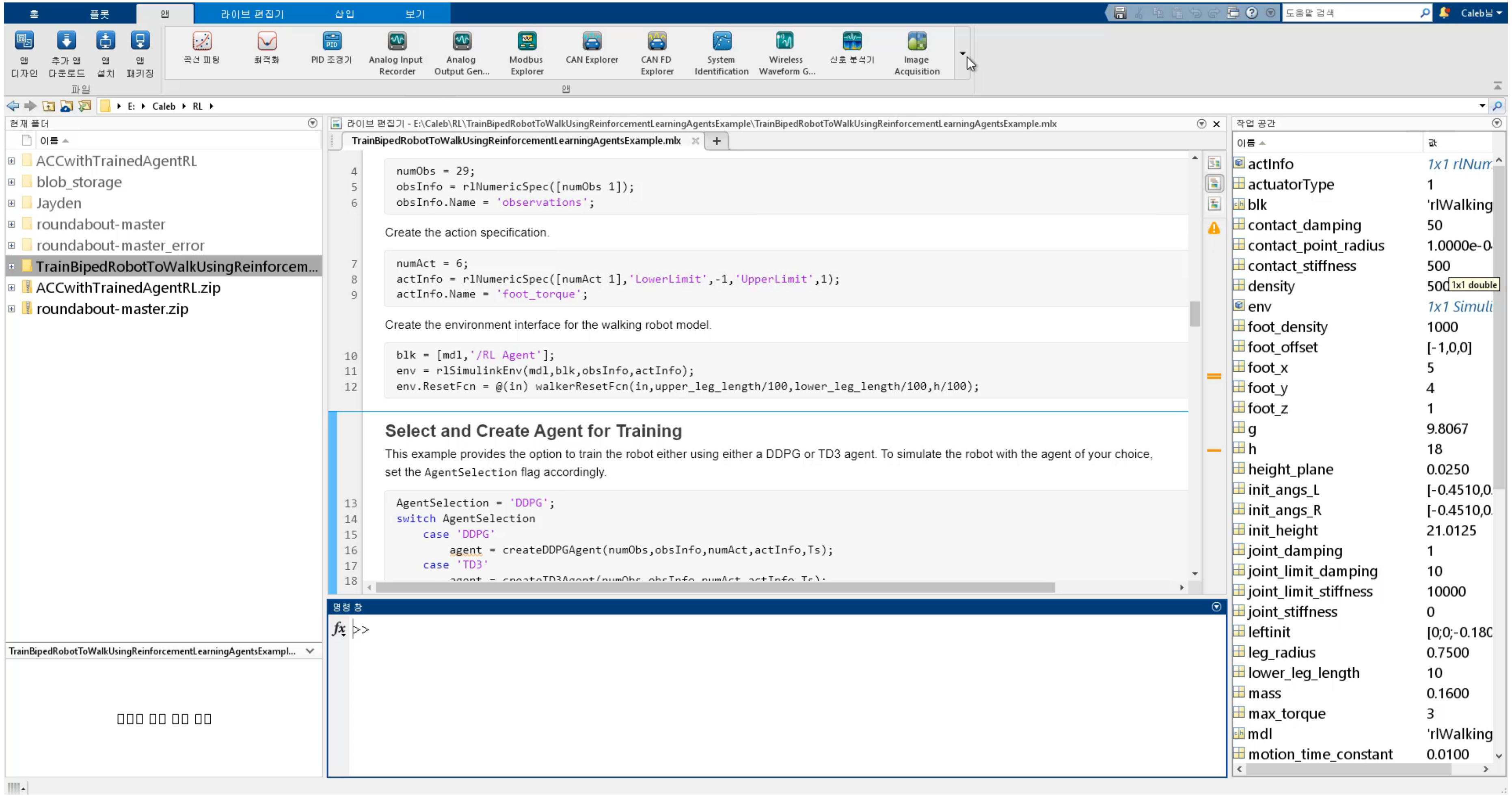


Creating the Agent

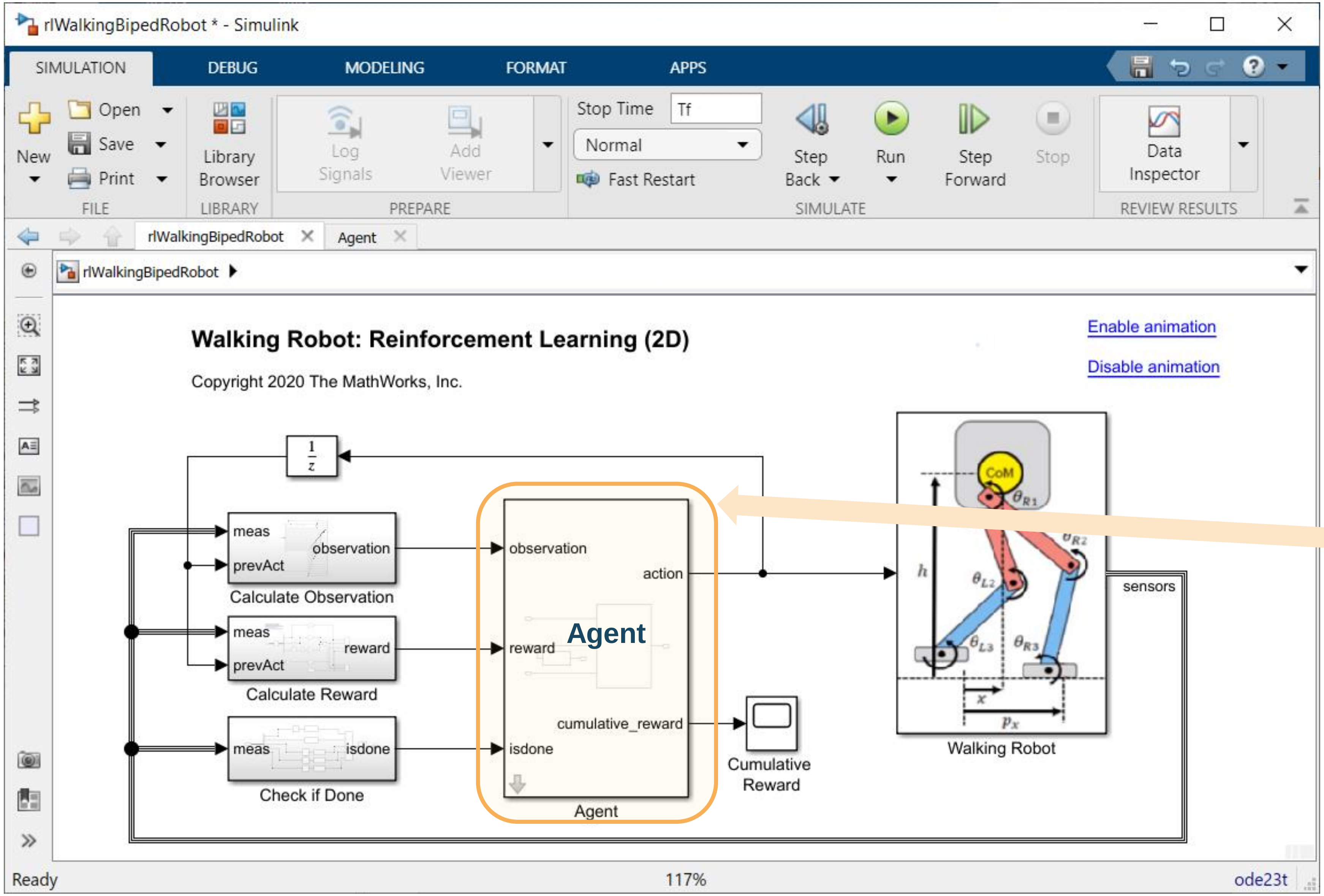


- Constructing a DDPG (Deep Deterministic Policy Gradient) Agent
  - Create the critic network
  - Create the policy network
  - Create the agent with actor and critic network and set hyperparameters

Creating the Agent with Reinforcement Learning Designer



Defining the Agent

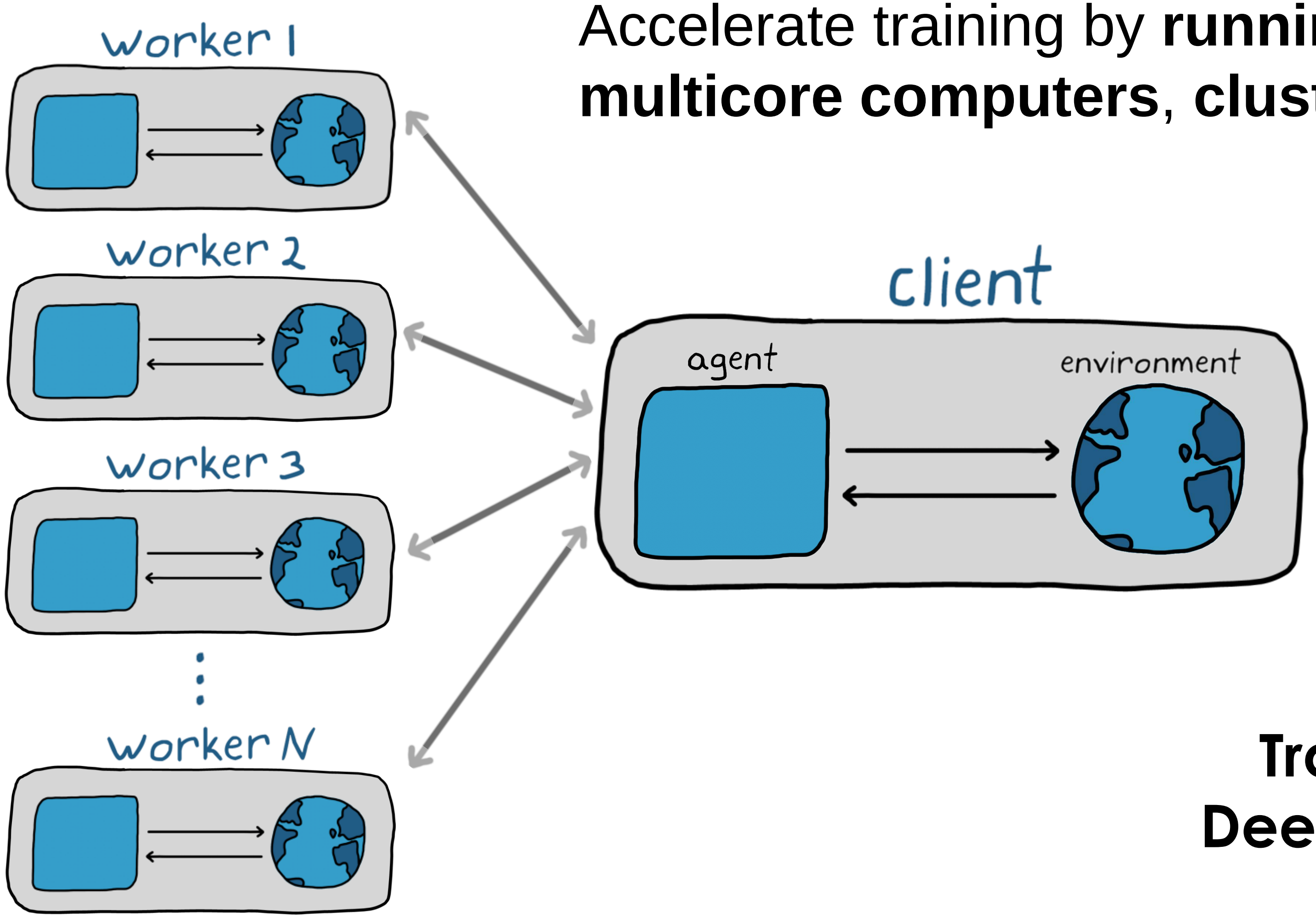


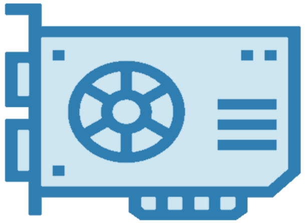
Define the agent

Reinforcement Learning Workflow

Training Our Deep Reinforcement Learning Agent

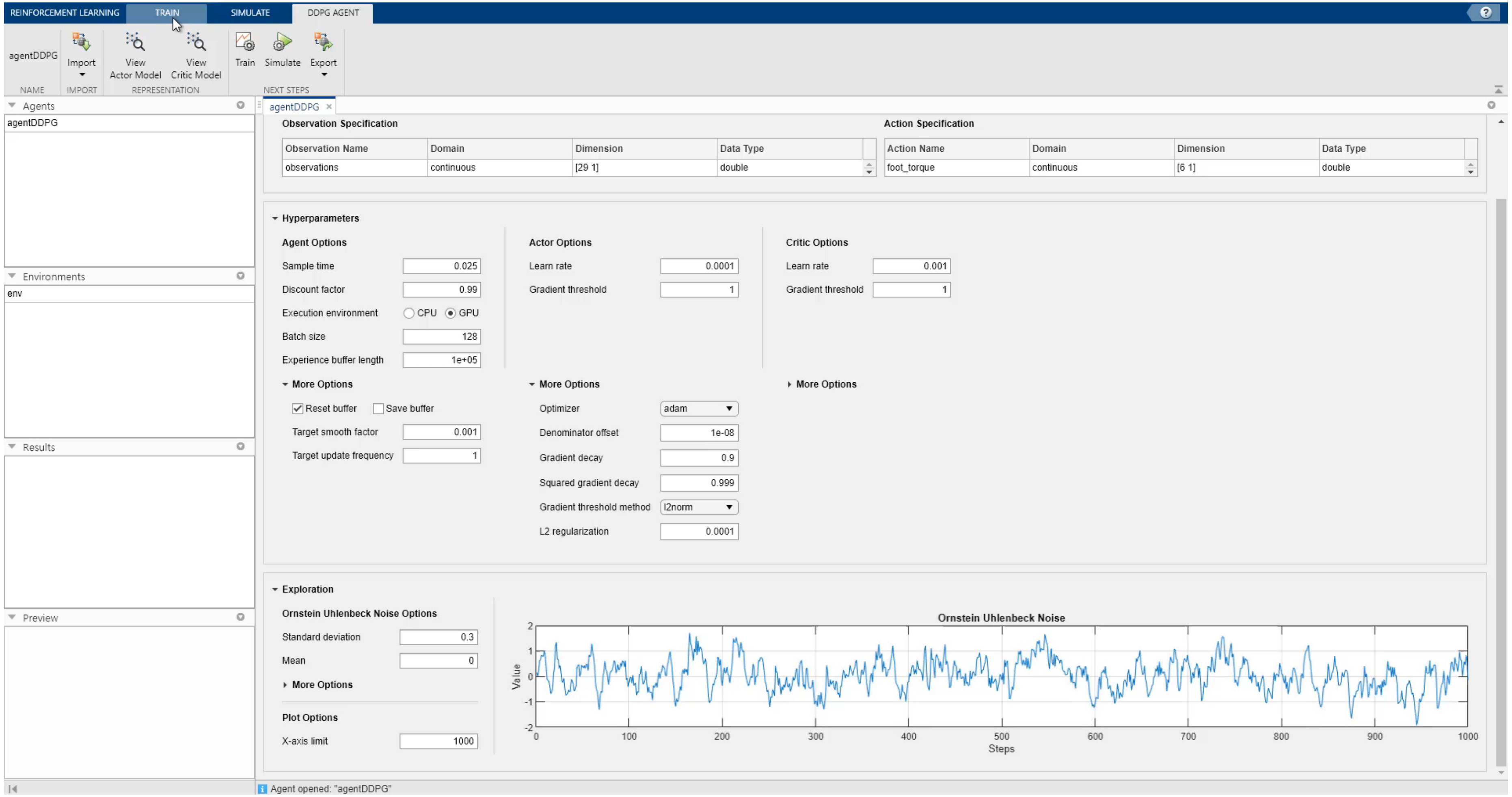
Accelerate training by **running simulations in parallel** on **multicore computers**, **clusters** or the **cloud**



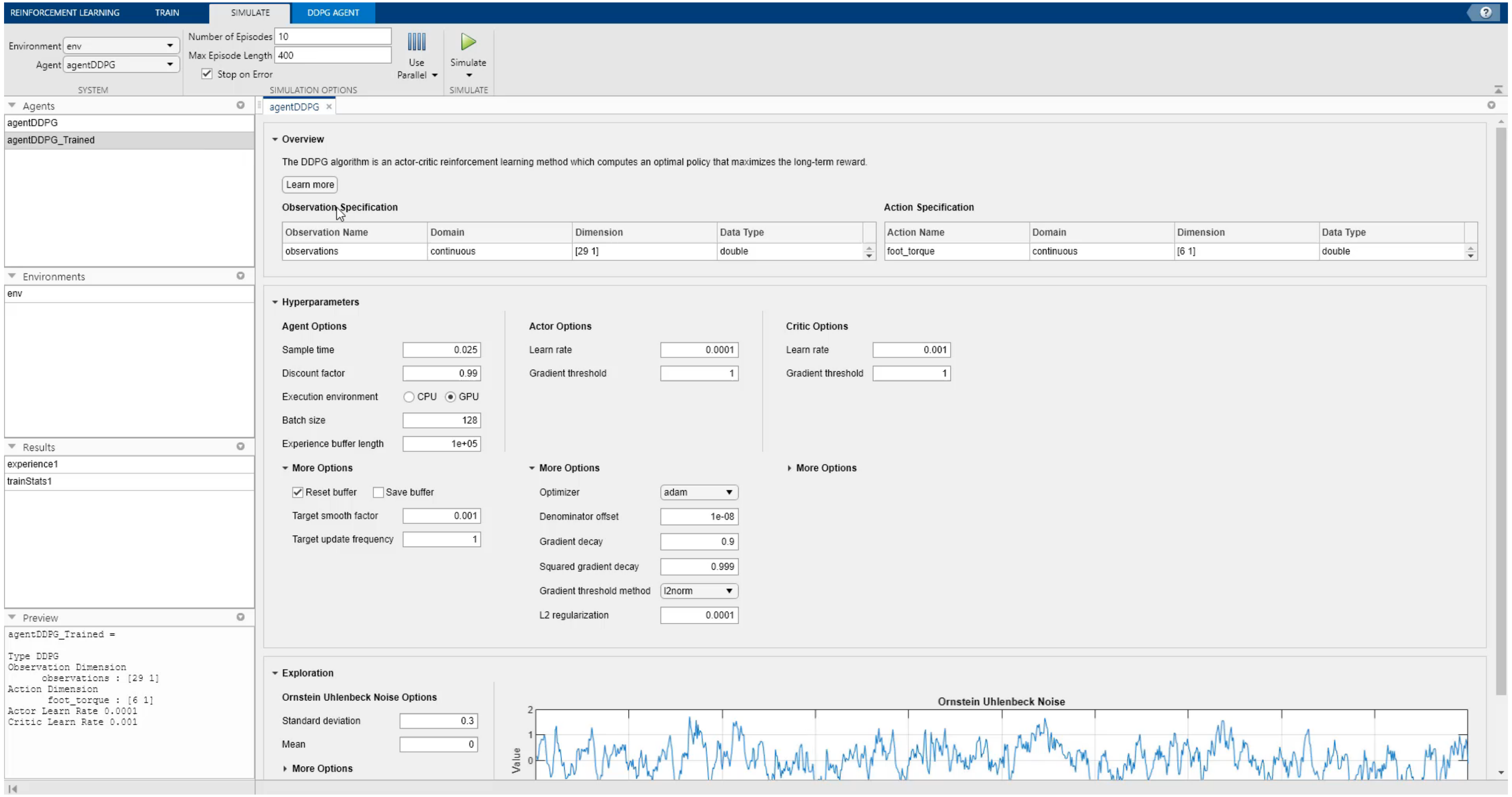


Train with the GPU when using **Deep Neural Networks** for Actor or Critic representations

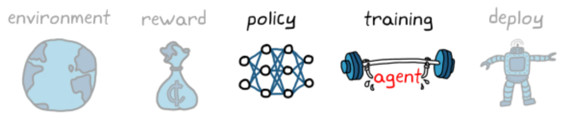
Training the Agent

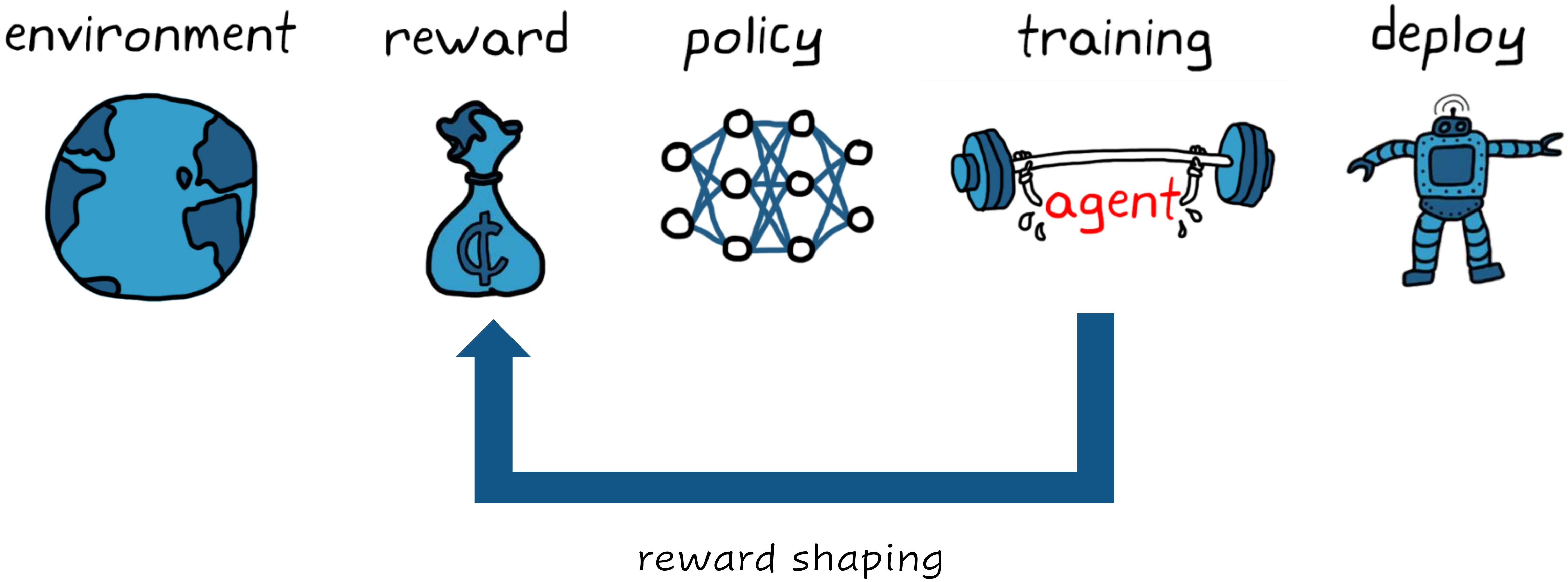


Testing the Agent

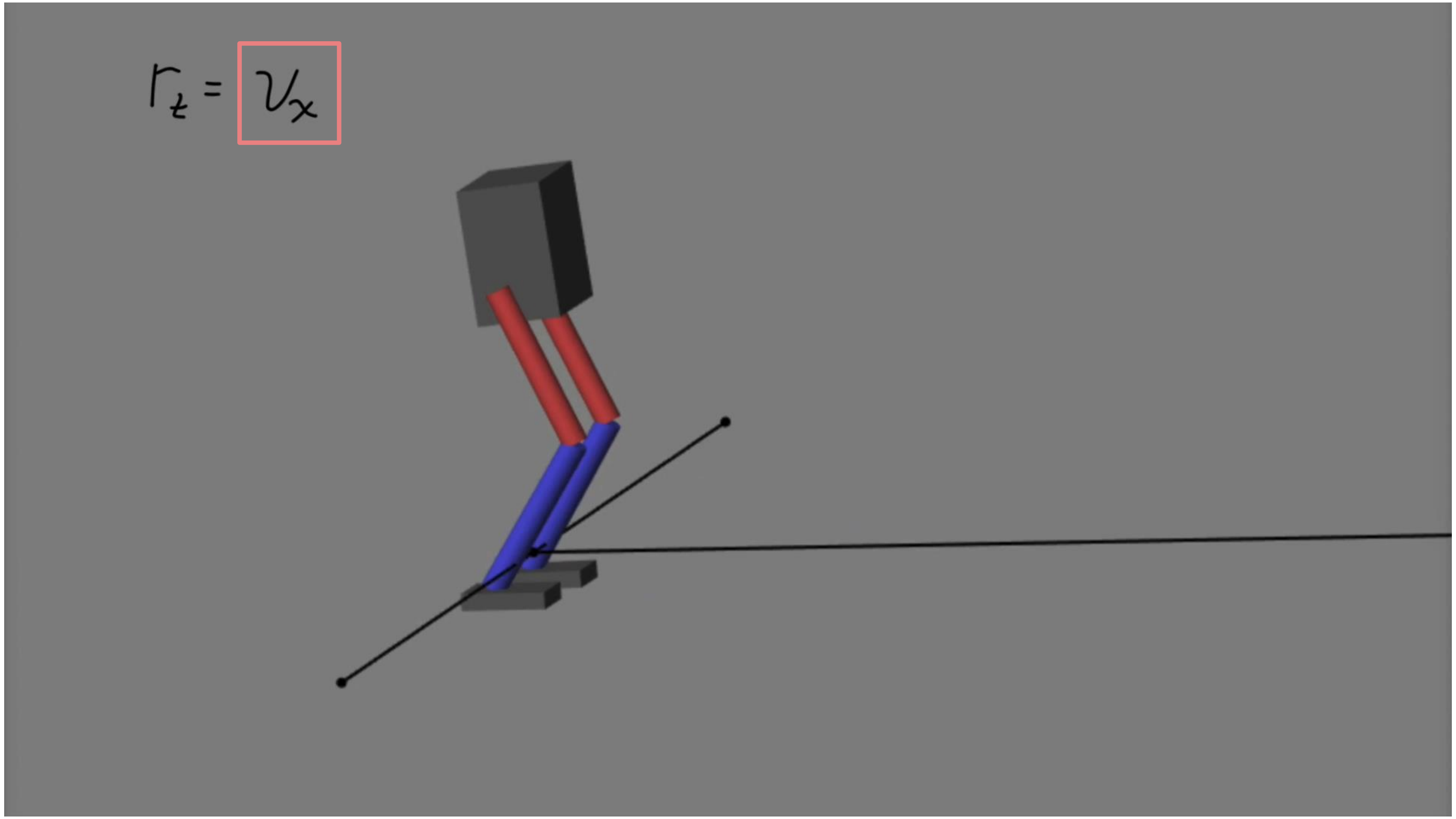


Reinforcement Learning Workflow



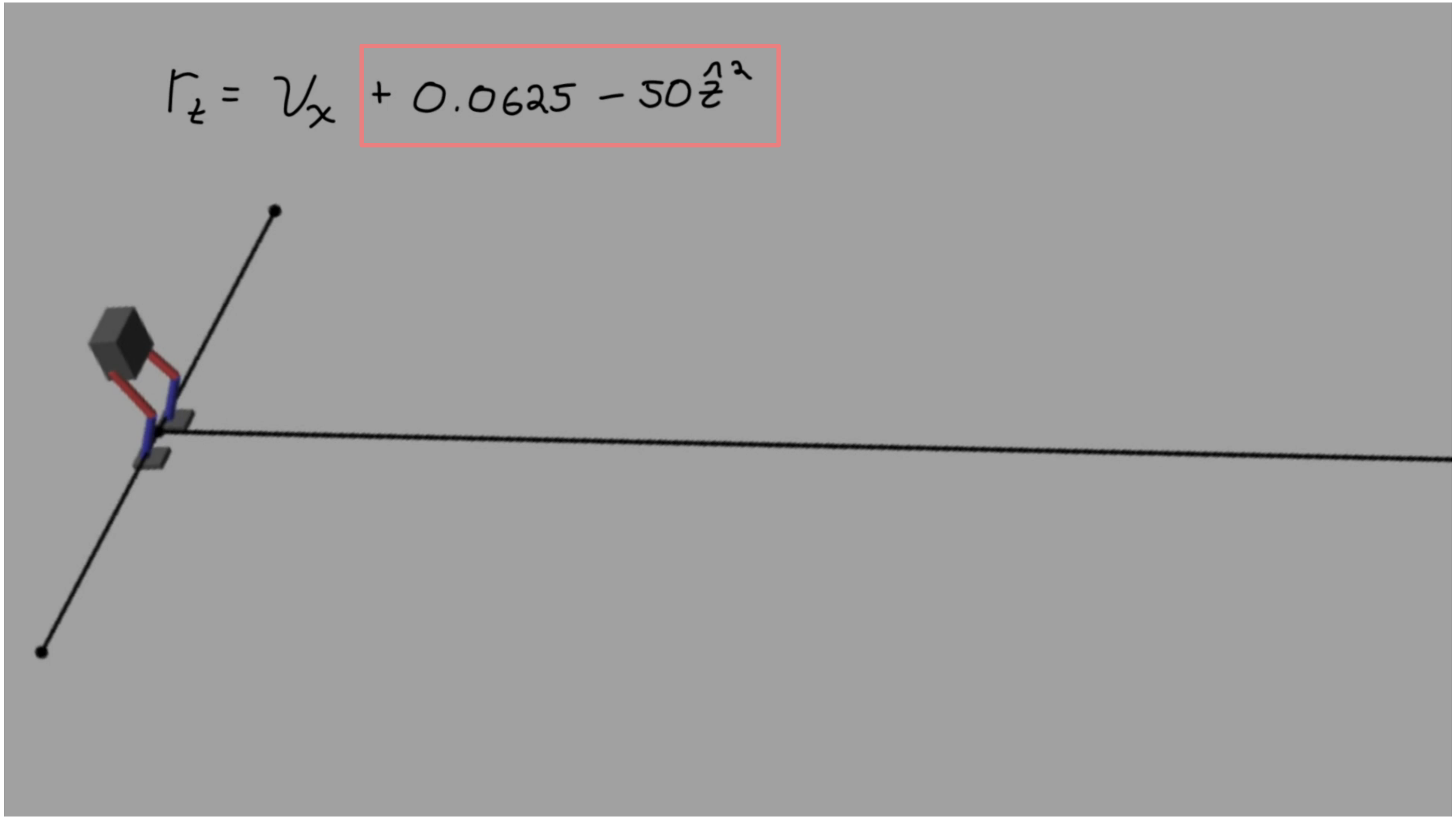


Reward Function Design Matters



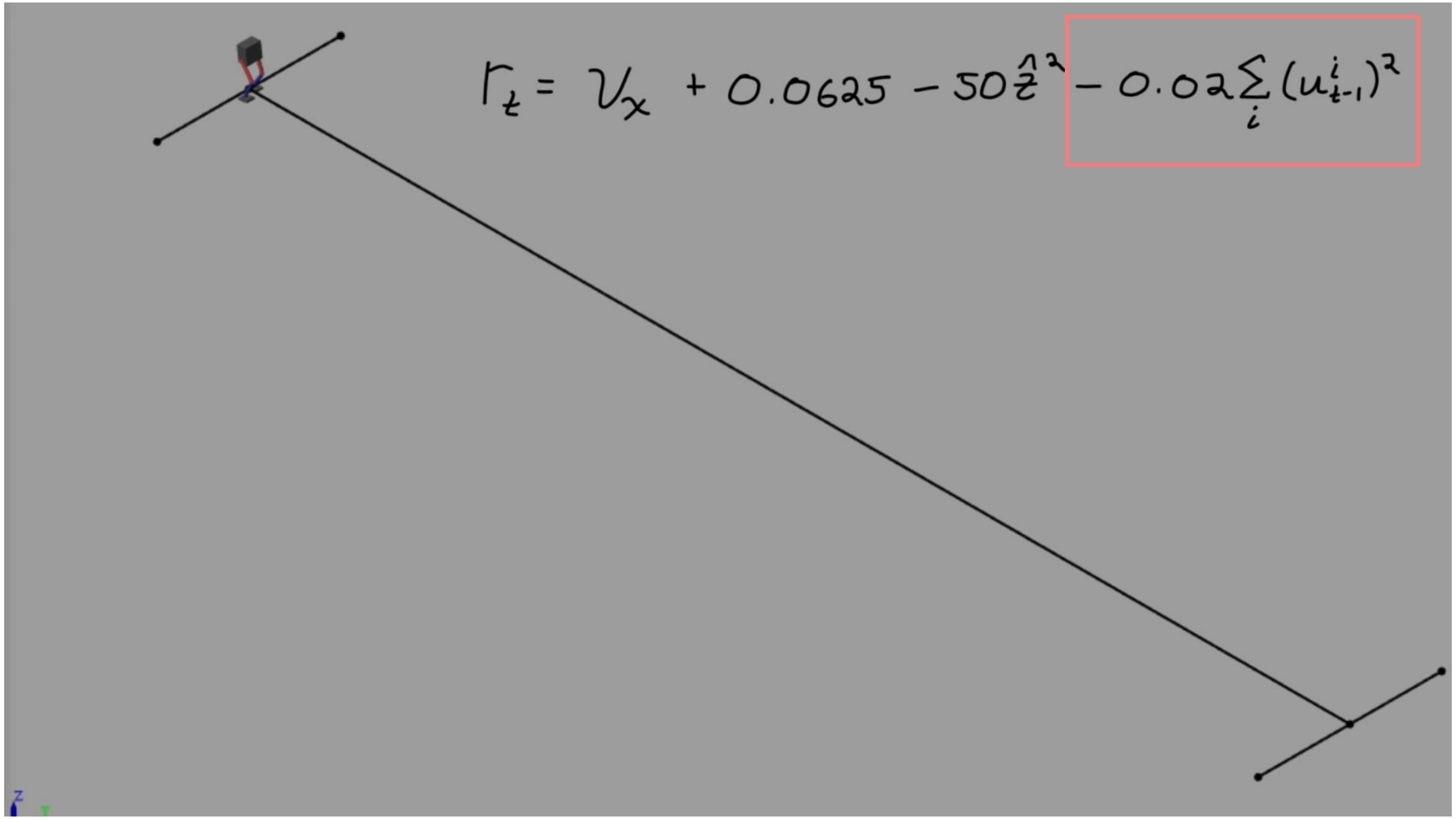
I want my robot to walk forward. Let's set the reward to be the robot's forward velocity.

Reward Shaping to Improve Learning



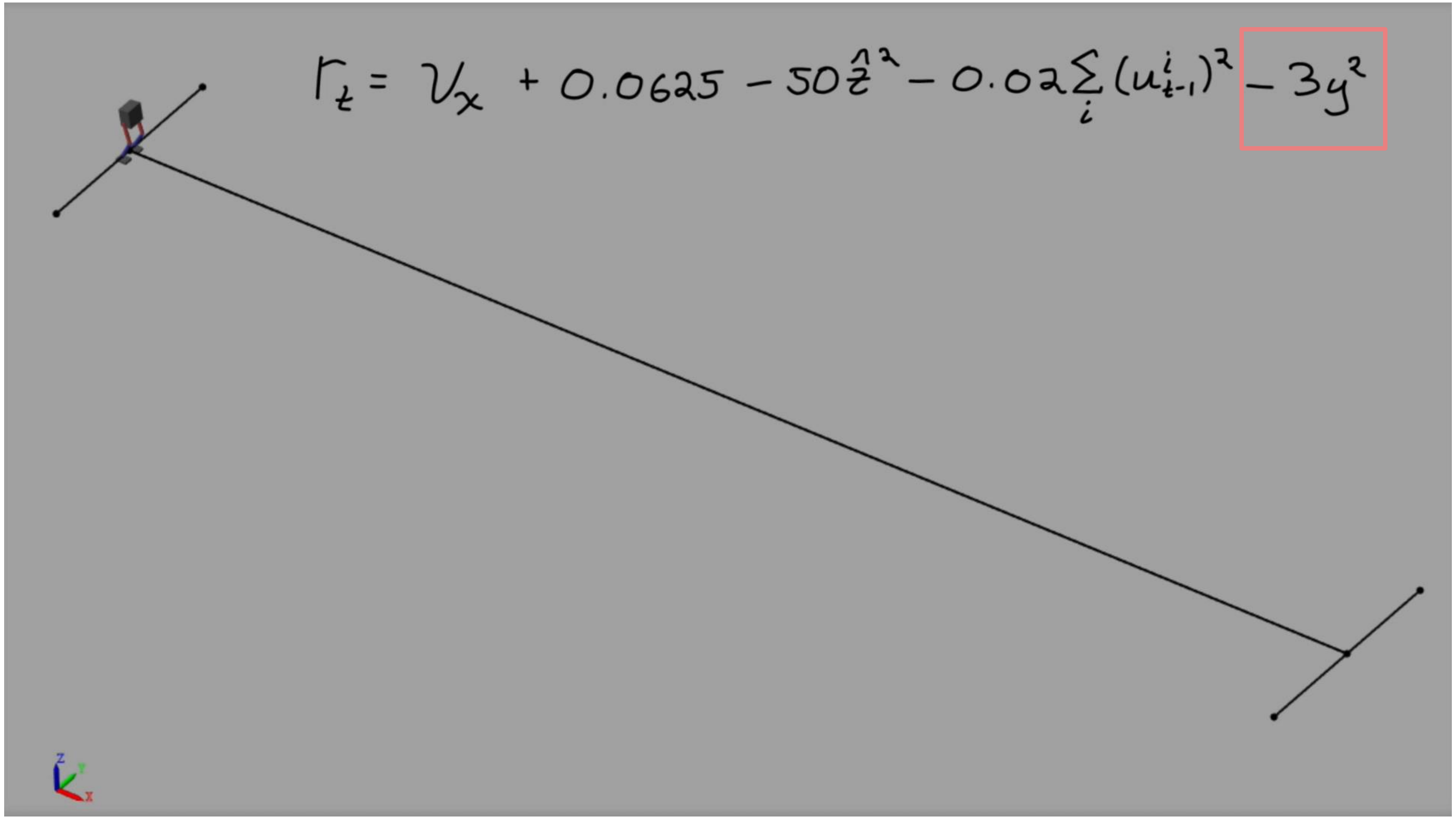
Let’s add a reward term for each time step it remains upright and a penalty for not maintaining a torso height

Reward Shaping to Improve Learning



Let’s add a penalty for the energy used for actuation

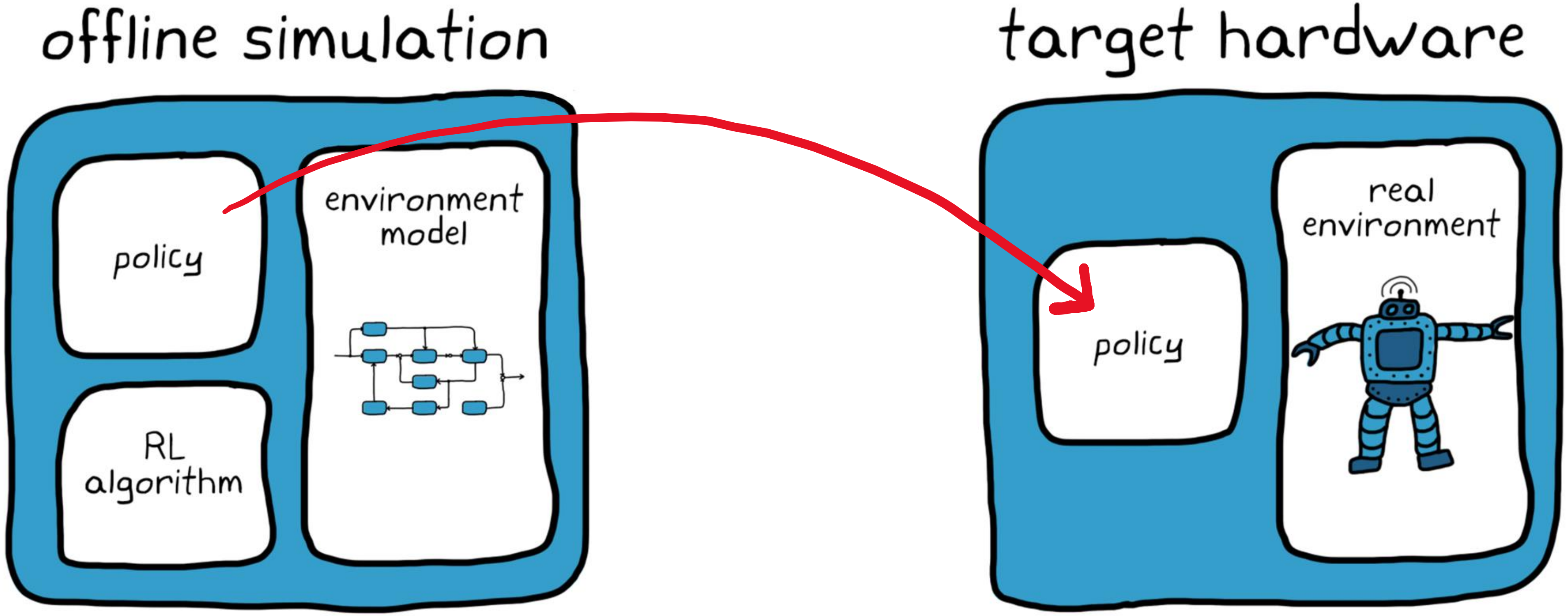
Reward Shaping to Improve Learning



Let's add a penalty for deviating from the line

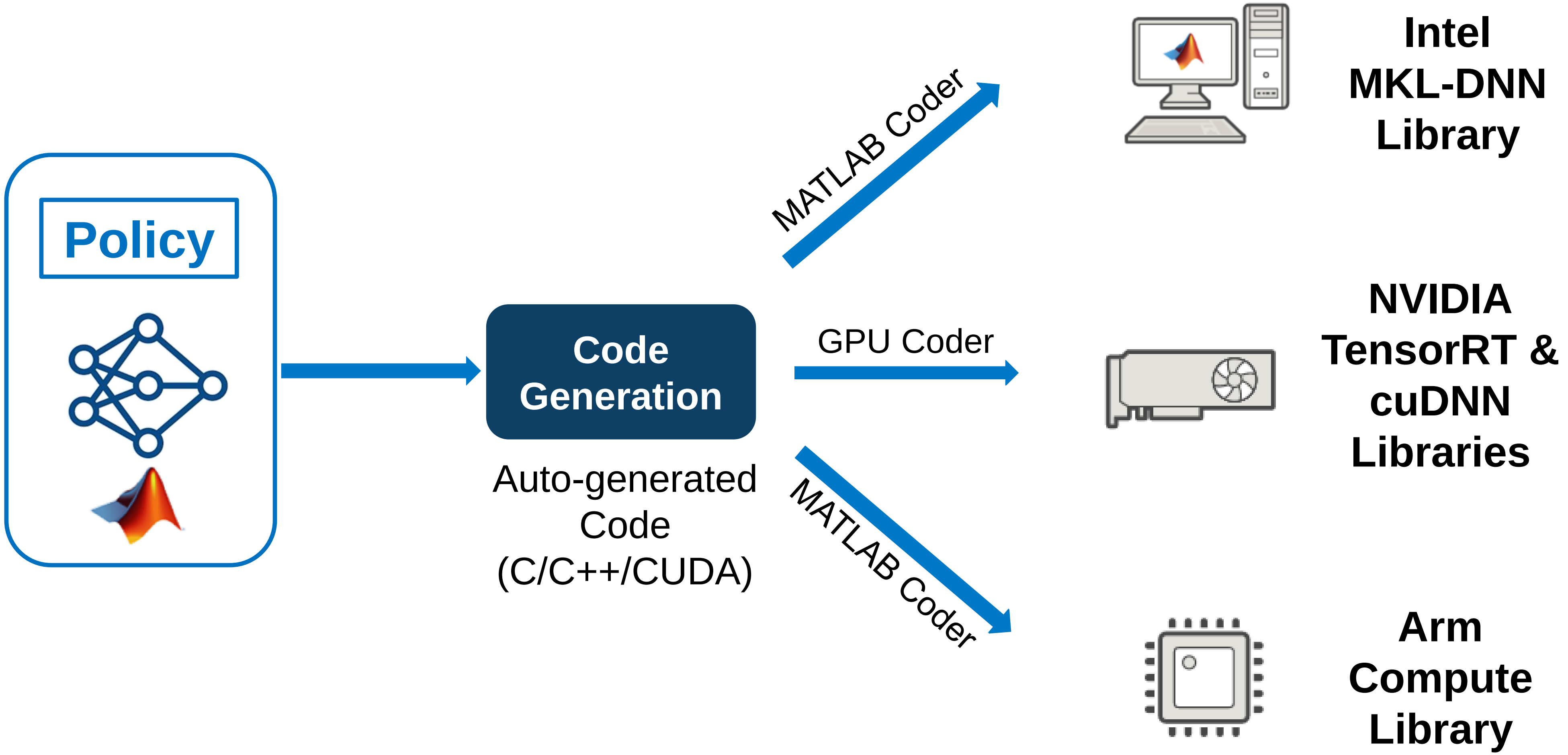
Reinforcement Learning Workflow

Deploy policy to the target hardware



Automatically generate C/C++ or **CUDA** code
to run the **policy** on an **embedded system**

Policy Deployment to Hardware Platforms



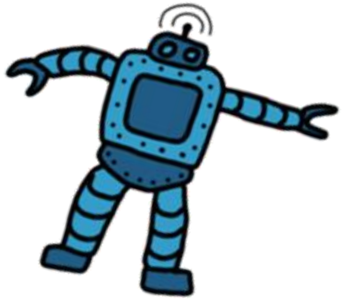
Intel® is a trademark of Intel Corporation

NVIDIA® and TensorRT® are registered trademarks of NVIDIA Corporation

Arm® is a registered trademark of Arm Limited (or its subsidiaries)

Key takeaways

- **Reinforcement Learning** can solve complicated control and decision problems
- **Reward Shaping** can improve learning outcomes
- **MATLAB and Simulink** provide a **complete workflow for Deep Reinforcement Learning**



감사합니다