

Reinforcement Learning for Particle Accelerators

An Introduction

Jan Kaiser and Oliver Stein
MT-ARD-ST3 pre-meeting ML workshop



Try Reinforcement Learning Yourself

Jupyter Notebook with code for examples from this presentation

The screenshot shows a Jupyter Notebook interface. On the left is a file browser with a search bar and a list of files and folders. The main area displays a notebook cell with the following content:

Part 3: Reinforcement Learning for Particle Accelerators

In this part of the workshop we will have a look at Reinforcement Learning (RL) and see how to apply it in practice, including a brief example looking at RL for accelerator optimisation.

Content

0. Introduction to RL
1. Practical example of applying RL on the well-known lunar lander task
2. Practical example of applying RL to a tuning task at the ARES particle accelerator
3. Further resources

0. Introduction to Reinforcement Learning

Reinforcement learning (RL) is a subfield of machine learning (ML), or more specifically a method of training ML models.

The idea of RL is to learn by trial and error, where one wants to perform good actions that give high rewards more often and bad actions that give low rewards less often.

What can RL do?

Here are some examples of what can be done with RL:

- Control humanoids in simulation
- [Play games with imperfect information and develop long-term strategies](#)
- [Control robot hands in the real world](#)
- [Control the plasma in a tokamak fusion reactor](#)

At the bottom of the notebook cell, there are two small images showing a game environment (likely StarCraft II) and a simulation of a tokamak fusion reactor.

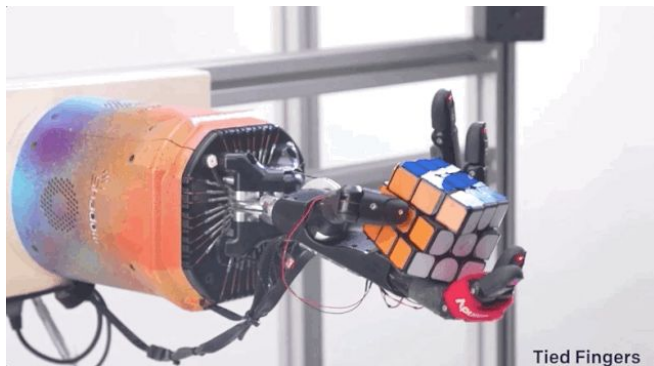
What can RL do?

Some examples

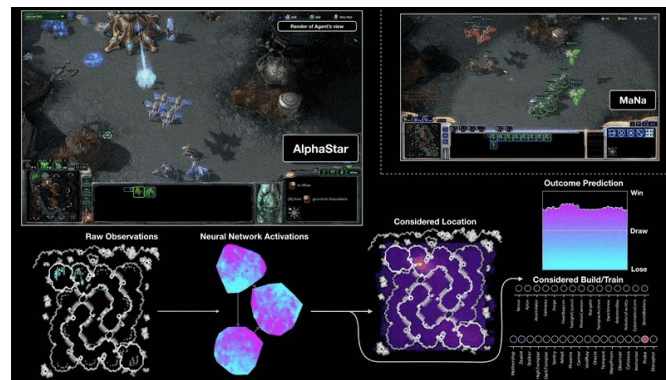
Control humanoid in simulation



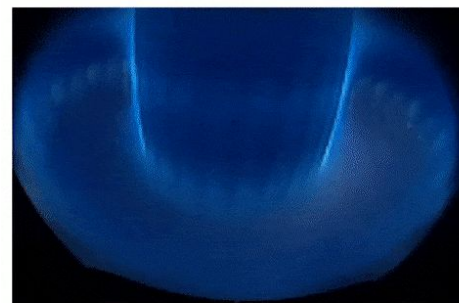
Control robot hands in the real world



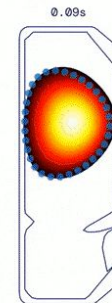
Play games with imperfect information and develop long-term strategies



Control the plasma in a tokamak fusion reactor



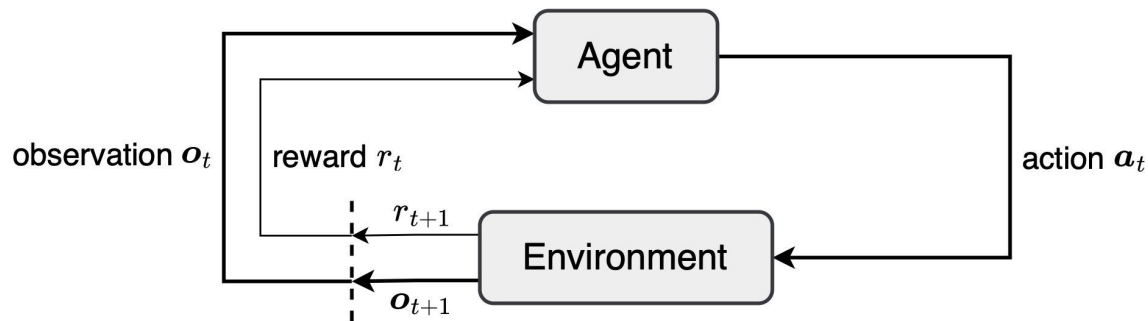
View from inside the tokamak



Plasma state reconstruction

Concepts of Reinforcement Learning

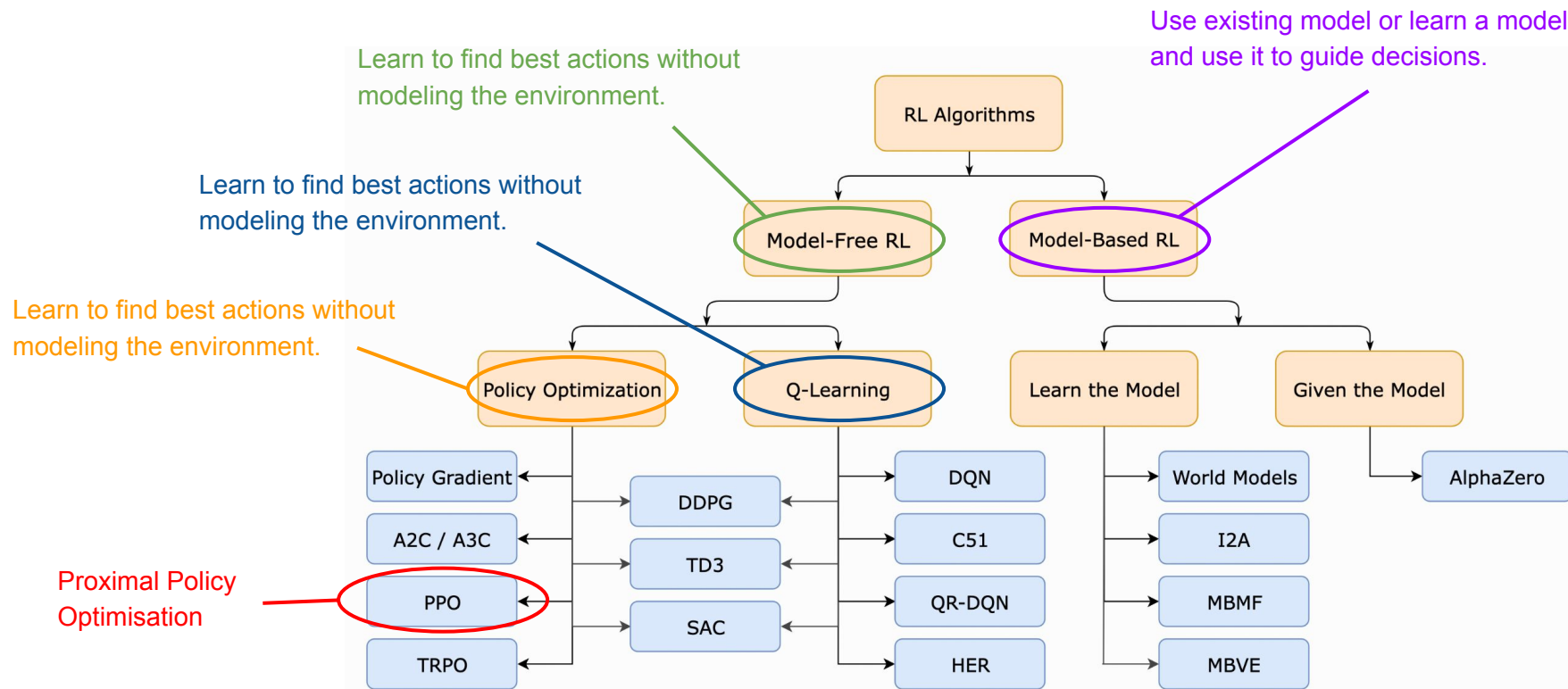
Some examples



- The **agent** (or **policy**) is the function we are trying to learn and tells us what to do to solve the task.
- The **environment** is the world that the RL agent lives in and defines the task.
- **Actions** are how the RL agent interacts with the environment.
- **Observations** are what the agent sees of the environment.
- The **reward** is returned by the environment after each action and describes the goodness of that action.
- The **return** is the cumulative reward over time. The goal of RL is to maximise the return.

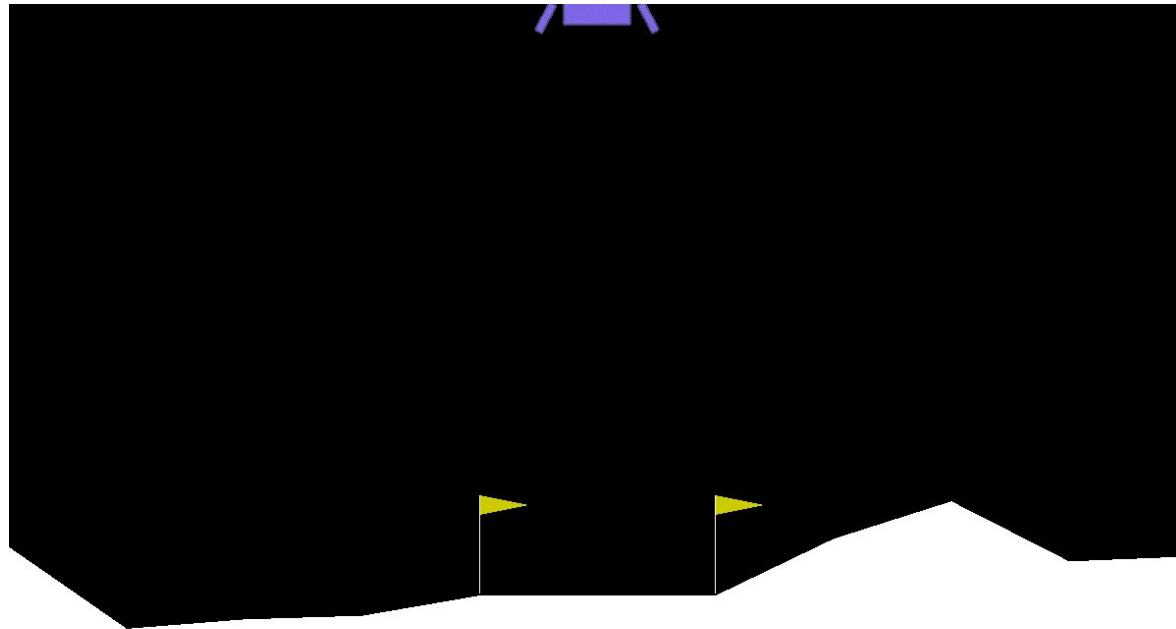
Taxonomy of Reinforcement Learning

A brief overview



Lunar Lander Example

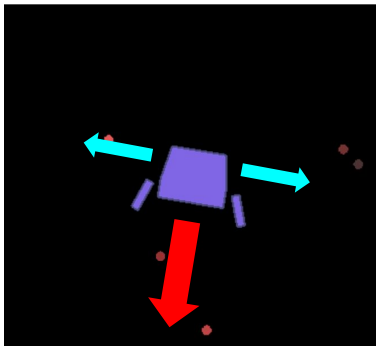
Introduction



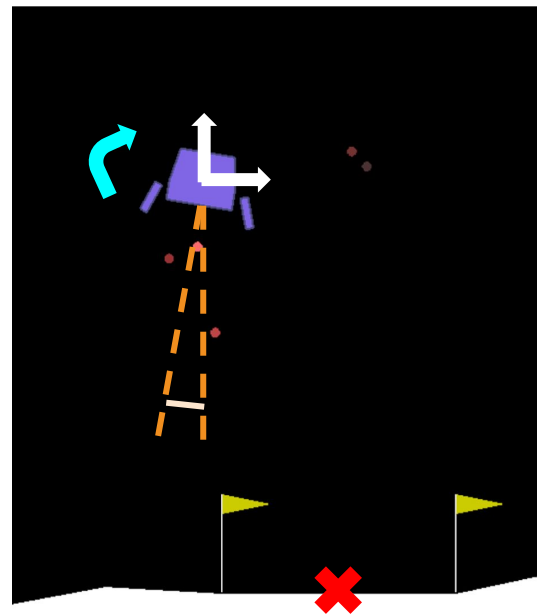
Lunar Lander Example

Actions and observations

Action

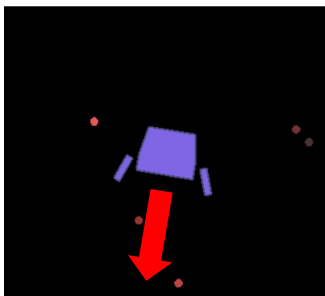


Observation

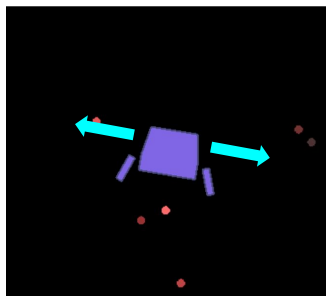


Lunar Lander Example

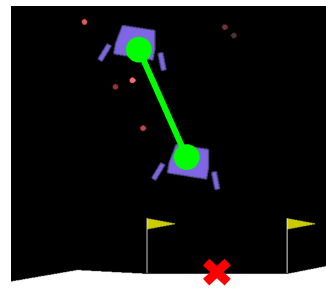
Rewards



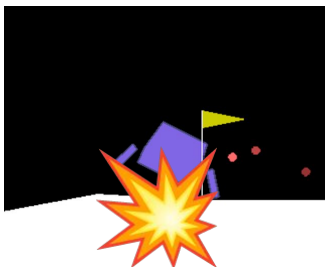
-0.3



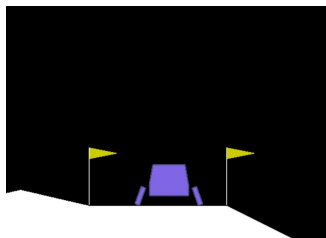
-0.03



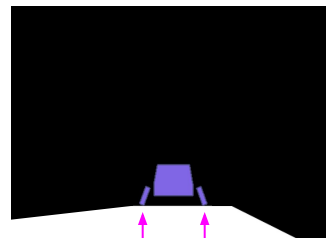
+/- d



-100



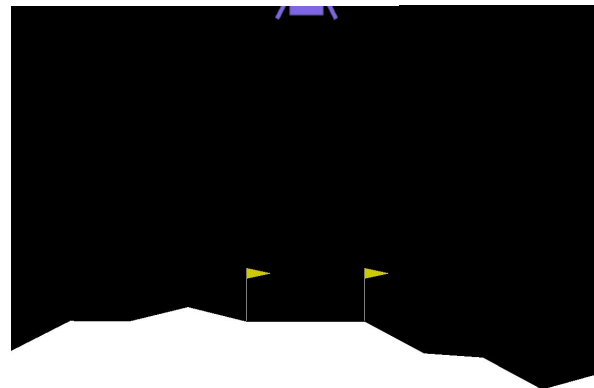
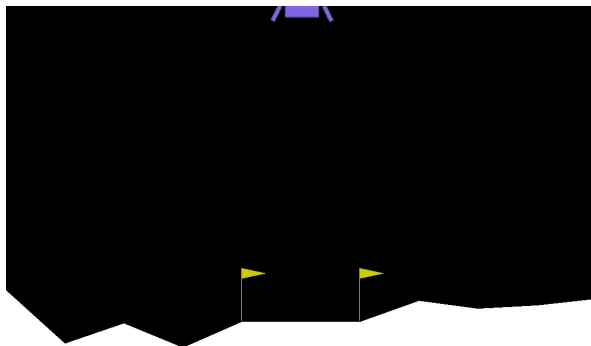
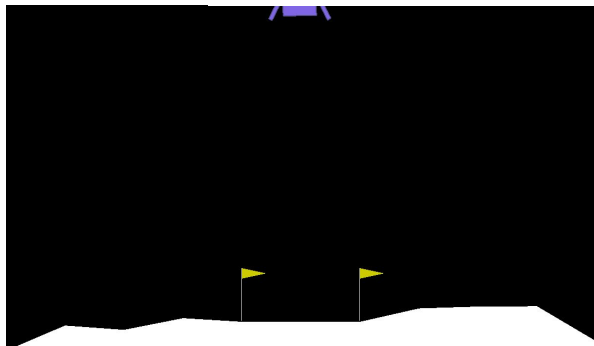
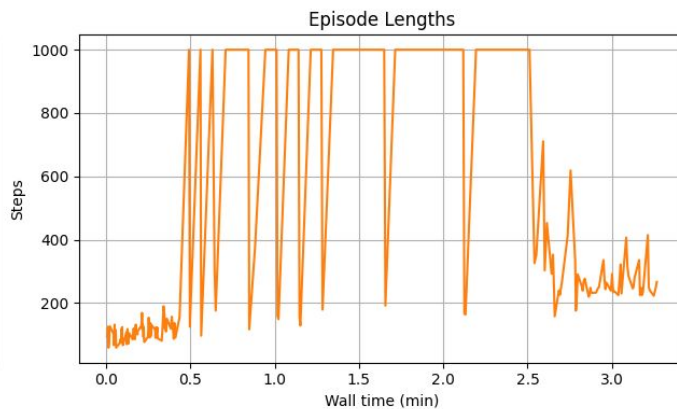
+200



+10 each

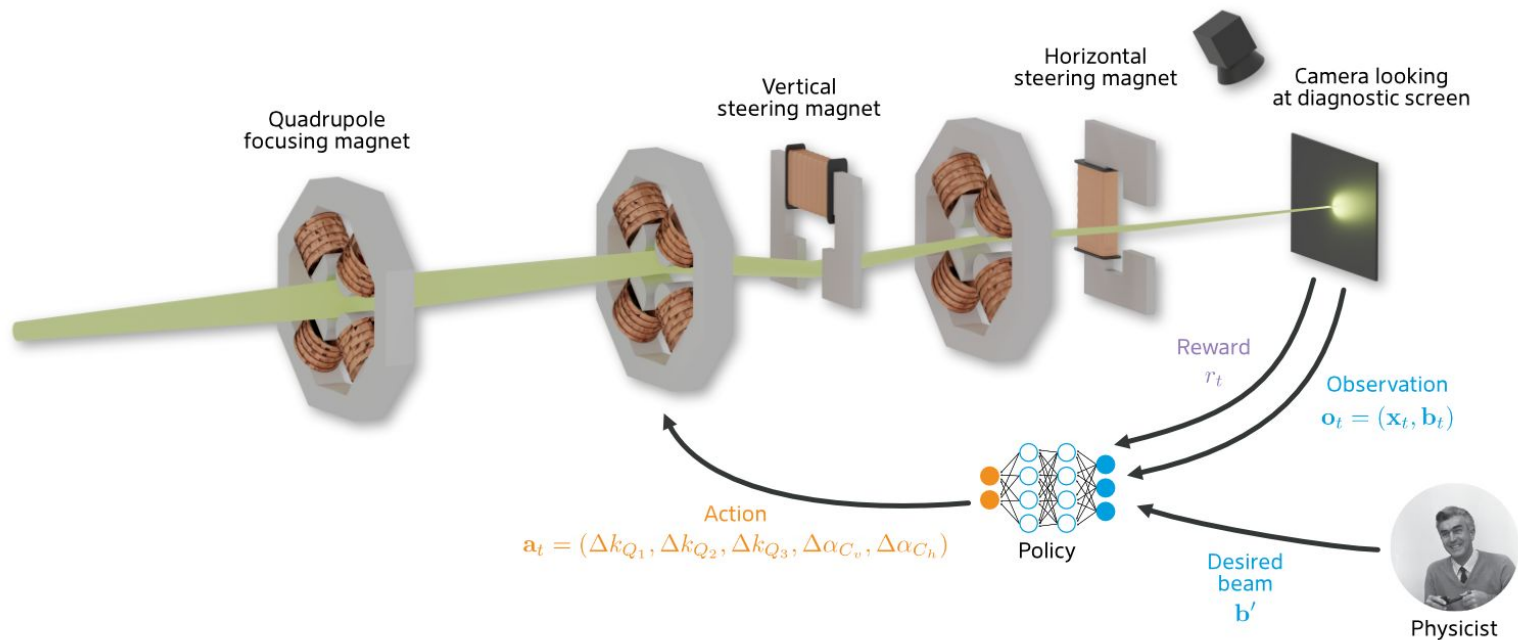
Lunar Lander Example

Training results



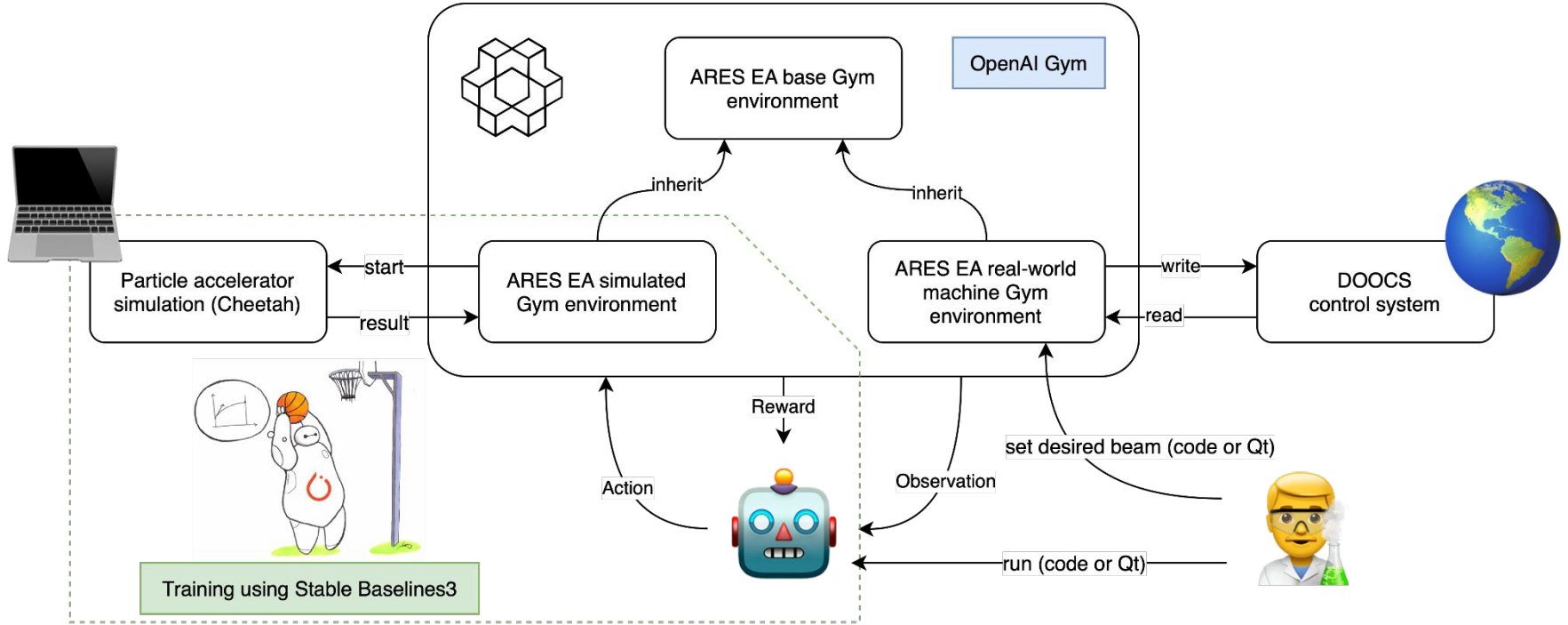
Accelerator Example

Positioning and focusing in the ARES Experimental Area



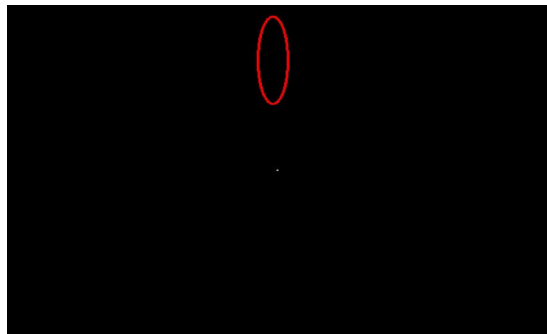
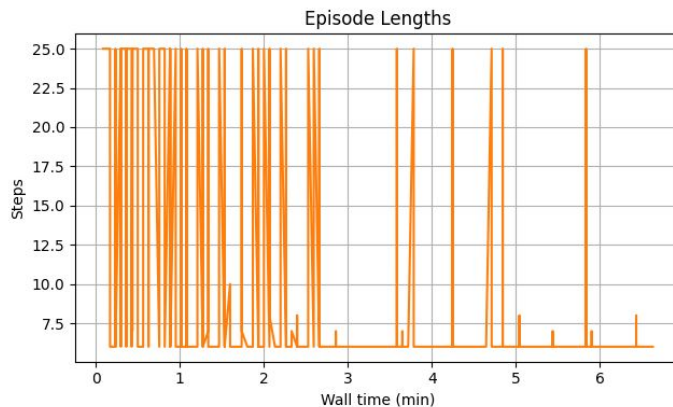
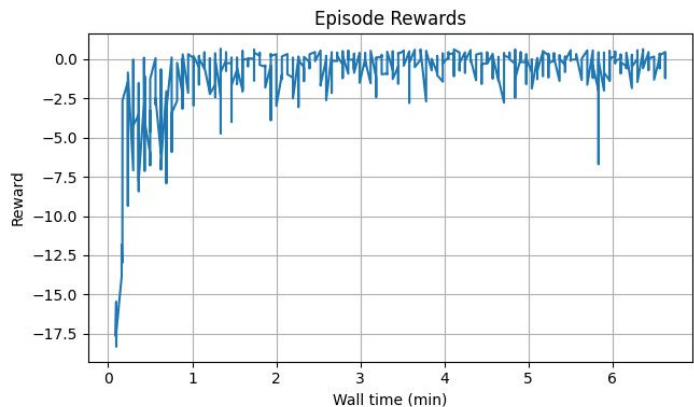
Accelerator Example

Technical overview

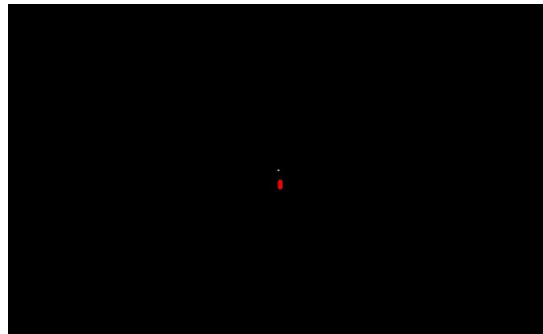


Accelerator Example

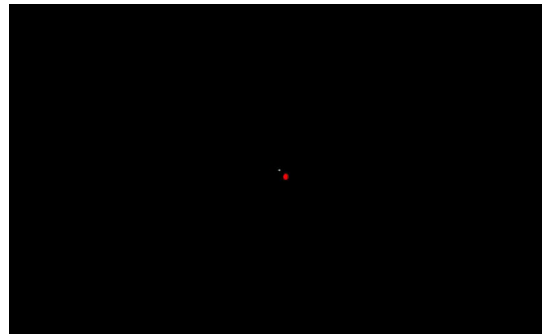
Training results



Q1=10.00 Q2=-10.00 CV=0.00 Q3=10.00
CH=0.00
mx=-0.07 sx=0.24 my=1.65 sy=0.66



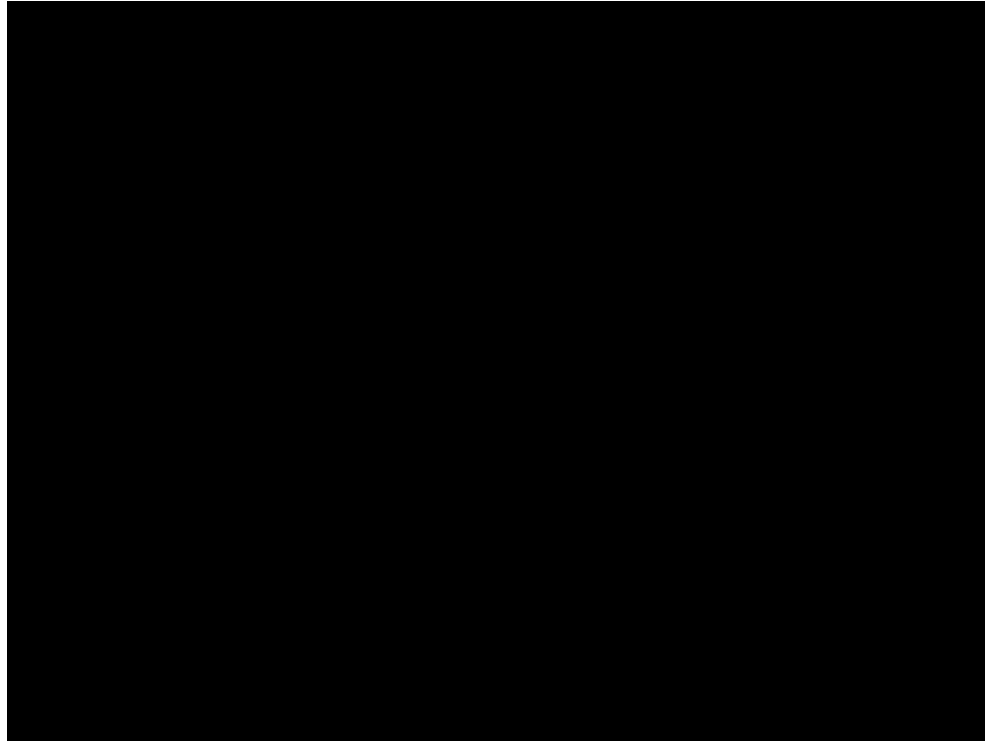
Q1=6.31 Q2=-16.51 CV=0.00 Q3=16.98
CH=0.00
mx=0.03 sx=0.02 my=-0.22 sy=0.06



Q1=4.59 Q2=-14.39 CV=0.00 Q3=19.12
CH=0.00
mx=0.10 sx=0.02 my=-0.10 sy=0.03

Accelerator Example

Running on the real accelerator



Further Resources

Where to start if you want to get into reinforcement learning

Getting started in RL

- [OpenAI Spinning Up](#) - Very understandable explanations on RL and the most popular algorithms accompanied by easy-to-read Python implementations.
- [Reinforcement Learning with Stable Baselines 3](#) - YouTube playlist giving a good introduction on RL using Stable Baselines3.
- [Build a Doom AI Model with Python](#) - Detailed 3h tutorial of applying RL using *DOOM* as an example.
- [An introduction to Reinforcement Learning](#) - Brief introduction to RL.
- [An introduction to Policy Gradient methods](#) - Deep Reinforcement Learning - Brief introduction to PPO.
- [Real-time artificial intelligence for accelerator control: A study at the Fermilab Booster](#) - Regulation of a gradient magnet power supply using RL and real-time implementation of the trained agent using field-programmable gate arrays (FPGAs).
- [Magnetic control of tokamak plasmas through deep reinforcement learning](#) - Landmark paper on RL for controlling a real-world physical system (plasma in a tokamak fusion reactor).

Papers

- [Learning-based optimisation of particle accelerators under partial observability without real-world training](#) - Tuning of electron beam properties on a diagnostic screen using RL.
- [Sample-efficient reinforcement learning for CERN accelerator control](#) - Beam trajectory steering using RL with a focus on sample-efficient training.
- [Autonomous control of a particle accelerator using deep reinforcement learning](#) - Beam transport through a drift tube linac using RL.
- [Basic reinforcement learning techniques to control the intensity of a seeded free-electron laser](#) - RL-based laser alignment and drift recovery.

Literature

- [Reinforcement Learning: An Introduction](#) - Standard text book on RL.

Packages

- [Gym](#) - Defacto standard for implementing custom environments. Also provides a library of RL tasks widely used for benchmarking.
- [Stable Baselines3](#) - Provides reliable, benchmarked and easy-to-use implementations of the most important RL algorithms.
- [Ray RLlib](#) - Part of the *Ray* Python package providing implementations of various RL algorithms with a focus on distributed training.

Questions or remarks?

Contact

DESY. Deutsches
Elektronen-Synchrotron

www.desy.de

Jan Kaiser & Oliver Stein
Machine Beam Control (MSK)
jan.kaiser@desy.de | oliver.stein@desy.de