Aristotle University of Thessaloniki
Department of Informatics

DEEPBOTS: AN EASY-TO-USE DEEP REINFORCEMENT LEARNING FRAMEWORK FOR ROBOTICS

M. Kirtas, K. Tsampazis, N. Passalis and A. Tefas
{eakirtas, tsampaka, passalis, tefas}@csd.auth.gr

PRESENTATION OUTLINE

Deep Reinforcement Learning and Robotics

Needs in DRL using simulators

Deepbots Framework / Deepworlds

Conclusion

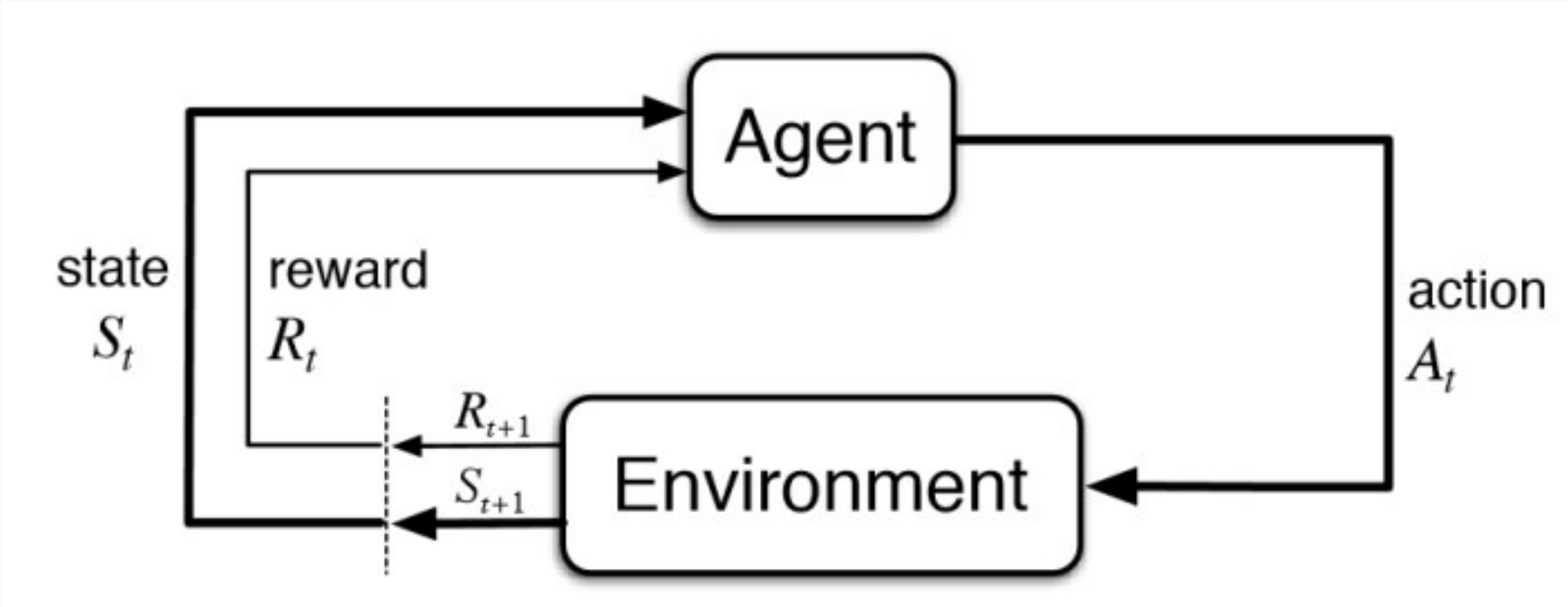
DEEP REINFORCEMENT LEARNING AND ROBOTICS

DEEP REINFORCEMENT LEARNING

Deep Reinforcement Learning is a fruitful domain of Machine Learning:

- **High impact** in the academic community and industry
- Employed in a great **variety of problems**
- Deals with **multidimensional and delicate tasks**

WHAT IS REINFORCEMENT LEARNING?



ROBOTICS AND SIMULATIONS

Despite the potential of DRL in robotics, it is facing some **limitations**:

- **Enormous amount of time** to explore the environment
- **Endangering** expensive robot hardware
- **Difficult** to build complicated environment

Can be overcome by using robotics simulators with:

- **Realistic physics**
- **Photo-realistic** scenes
- **Easily adjustable** to user's needs
- Can simulate time at **accelerated speeds**

DRL IN ROBOTICS SIMULATORS

Simulators are powerful tools but it is **not straightforward to interface** with DRL methods:

- **Steep learning curve**
- DRL operates over a **higher level of abstraction**
- Introduces **development overhead** in DRL pipeline

Considerably **restricts their usefulness** for researchers and students.

NEEDS IN DRL USING SIMULATORS

BENCHMARKING AND STANDARIZATION

Dealing with DRL tasks in simulators requires:

- Interpretable results
- Benchmarking ability
- Problem-generic agents
- Easy interfacing

OPENAI GYM

OpenAI gym **provides**:

- Standardization
- Most RL agents interface with gym
- A variety of environments
- Easy use

However it comes with some **limitations**:

- Unrealistic scenarios
- Difficult to extend those scenarios
- Based on closed source MuJoCo simulator

WEBOTS SIMULATOR

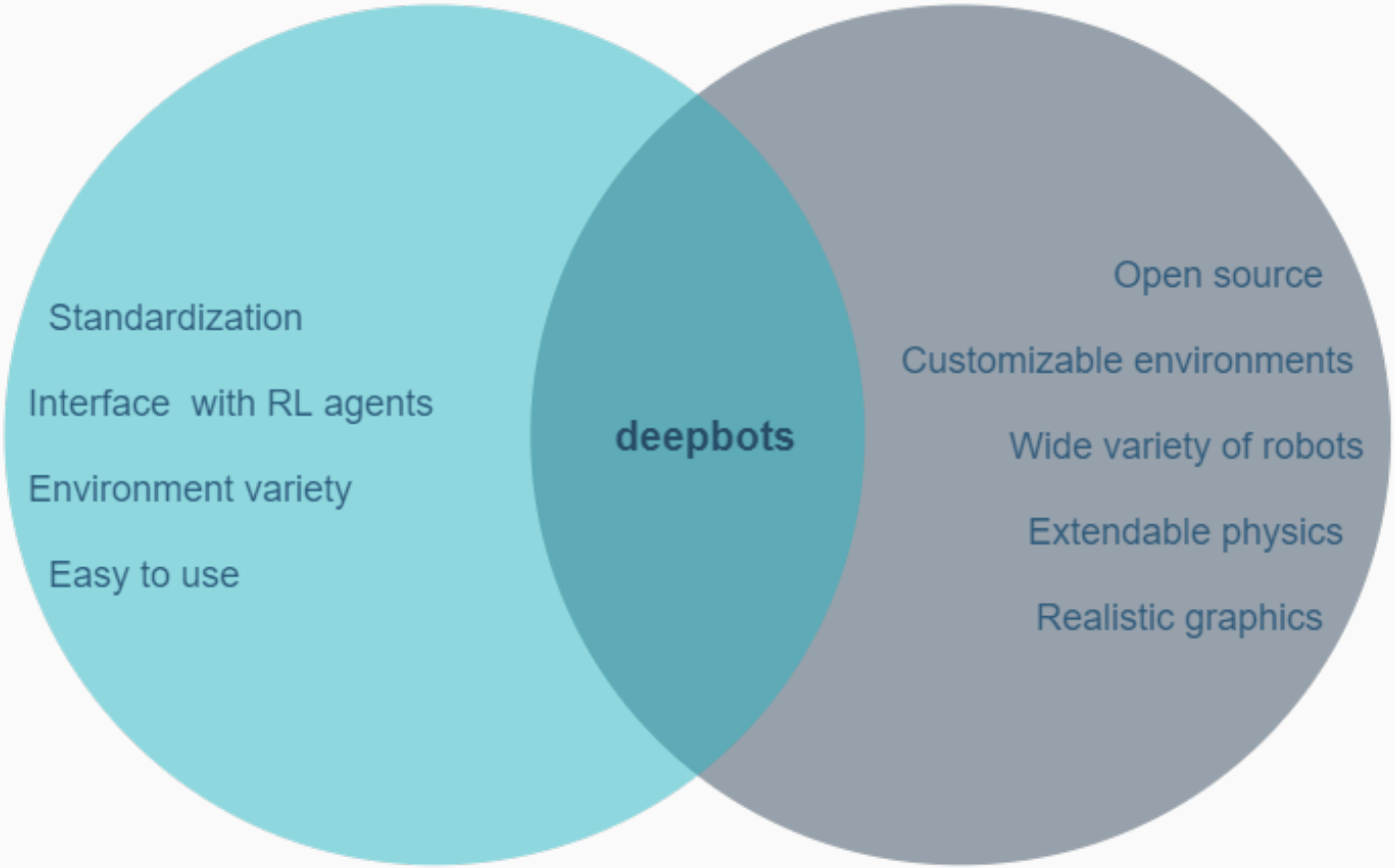
Webots is a state-of-the art robot simulator which, among others, **provides**

- Customizable environments
- Wide variety of robots
- Extendable physics engine
- High fidelity simulations with realistic graphics

However Webots is **not so friendly** for those with DRL background

DEEPBOTS FRAMEWORK / DEEPWORLDS

OVERVIEW

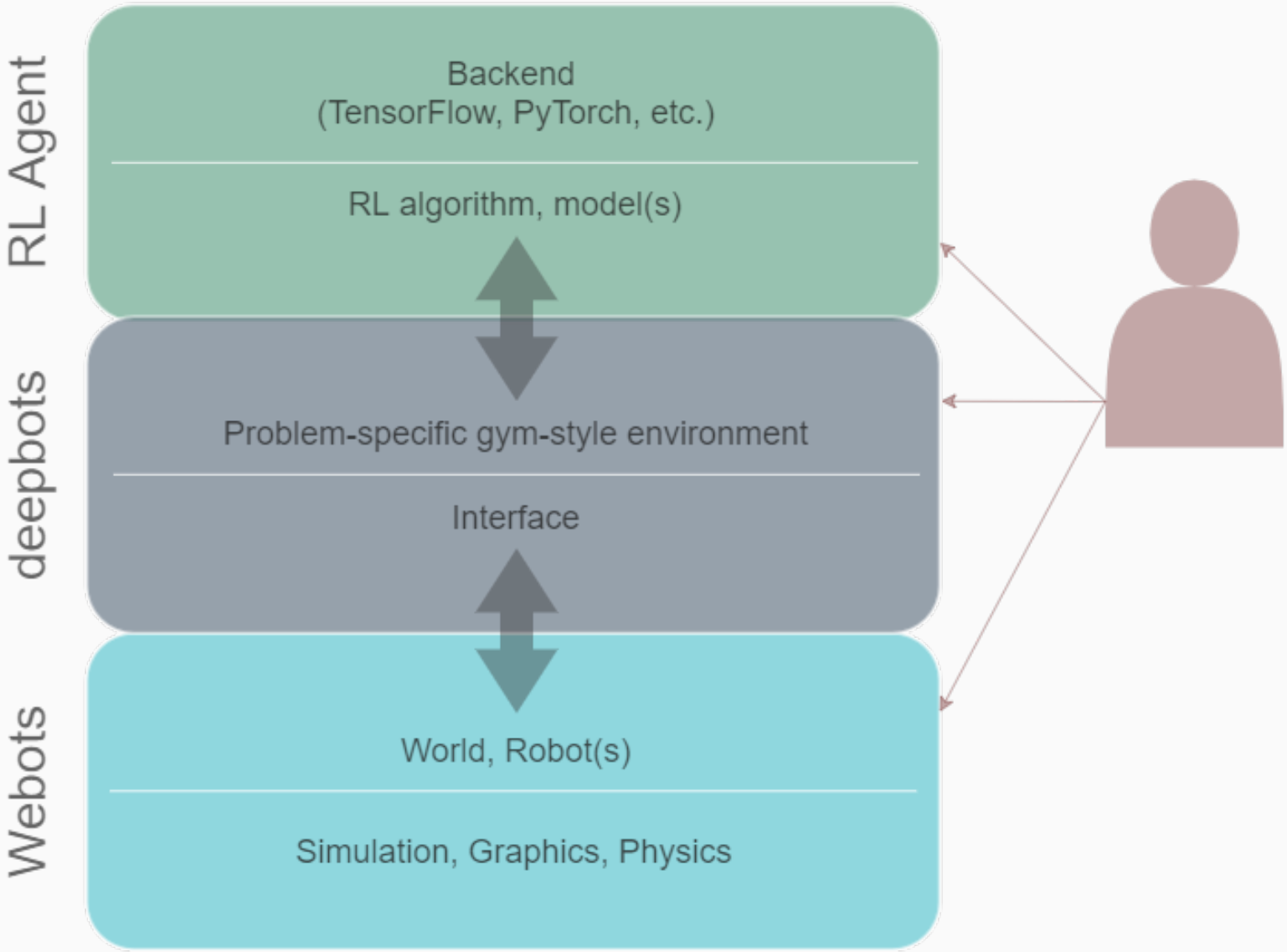

Standardization
Interface with RL agents
Environment variety
Easy to use
deepbots
Open source
Customizable environments
Wide variety of robots
Extendable physics
Realistic graphics
OpenAI gym
Webots

GOALS

The **deepbots framework** is an **open-source** project that aims to:

- **Facilitate** and **guide** development helping the user create a **gym-style environment** for their problem
- Act as an **interface** between the **environment** and **Webots**
- Hides standard Webots functions that are irrelevant to the problem enabling users to **focus on their task**
- Create an open-source ecosystem with ready-to-use environments for **benchmarking** RL algorithms

DESIGN

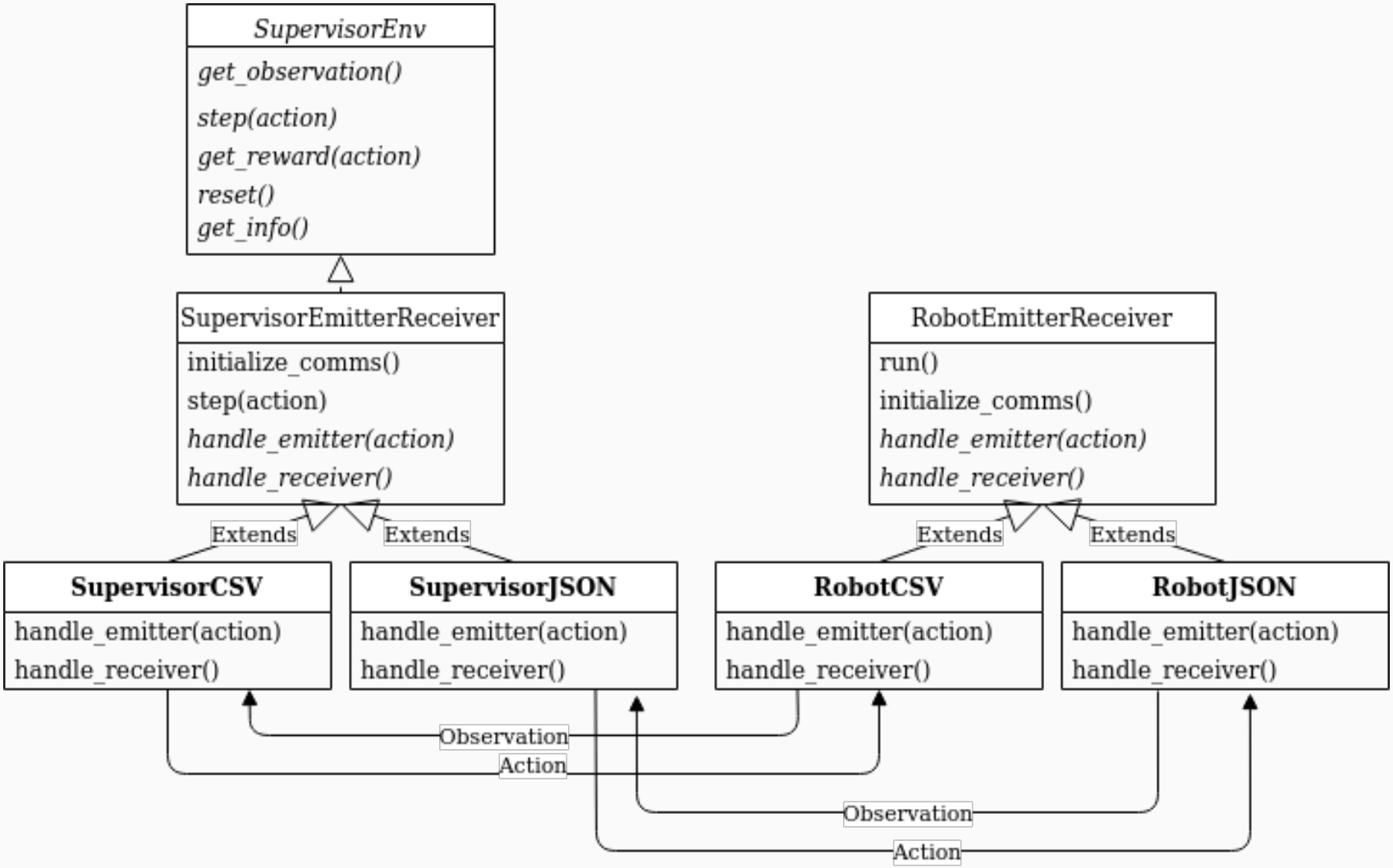

RL Agent
Backend
(TensorFlow, PyTorch, etc.)
RL algorithm, model(s)
deepbots
Problem-specific gym-style environment
Interface
Webots
World, Robot(s)
Simulation, Graphics, Physics

ARCHITECTURE

Deepbots uses a hierarchy of abstract or partially abstract classes to:

- Combine a gym-style environment with functionality needed by Webots
- Provide implementations of common functions
- Be customizable depending on the use-case by overriding or adding functionality to methods

ARCHITECTURE: EMITTER - RECEIVER SCHEME



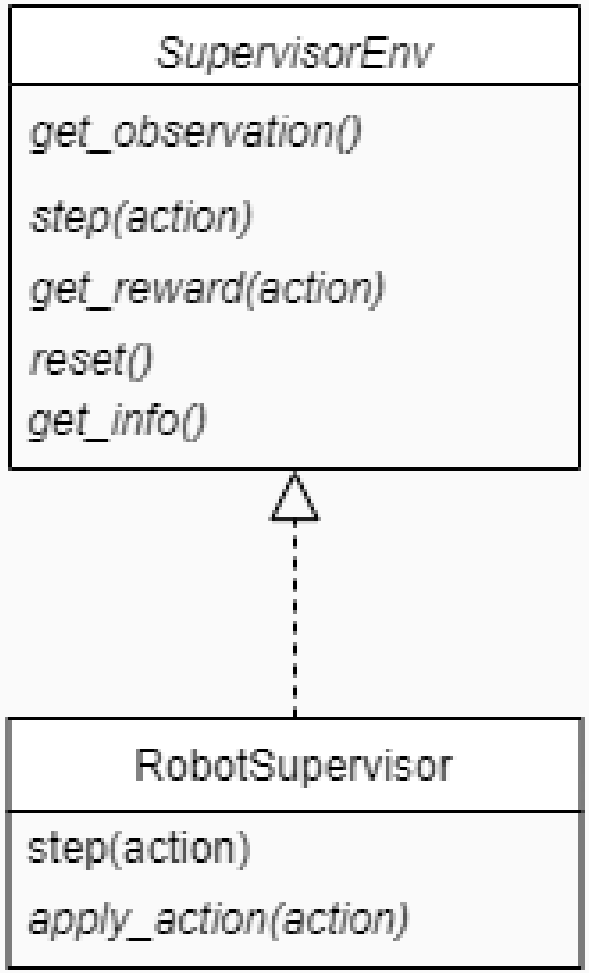
ARCHITECTURE: EMITTER - RECEIVER SCHEME

By separating the Supervisor from the Robot, the first scheme offers the following advantages:

- Flexibility to fit any use case
- Support for distributed experience acquisition by several Robots controlled by a centralized agent
- Robot is highly realistic, without having access to any additional information other than from its sensors

However, emitter-receiver communication introduces an overhead, both in execution speed and development time.

ARCHITECTURE: COMBINED SCHEME



ARCHITECTURE: COMBINED SCHEME

By combining the Supervisor and the Robot on a single class, the second scheme offers the following advantages:

- Overcomes the emitter-receiver communication overhead
- Can be used with high-dimensional or long data, e.g. camera images
- Easier to implement, no communication methods to implement

However, Robot needs supervisor privileges becoming unrealistic and is limited to one-to-one Robot-Supervisor setups.

USABILITY

- Deepbots is distributed as an easy-to-install Python 3 package: **pip install deepbots**
- Accompanying repository, **deepbots-tutorials**, contains tutorials to help new users: `https://github.com/aidudezzz/deepbots-tutorials`
- Currently contains a step-by-step tutorial for recreating the well-known Cartpole problem

EXTENSIBILITY

Deepbots is designed to be extensible via environment wrappers.

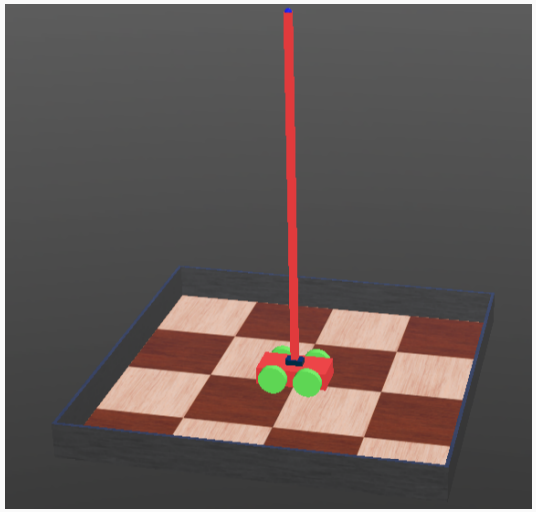
- Basic wrappers are provided, like a tensorboard logger wrapper
- More wrappers with additional features can be contributed by the community

DEEPWORLDS

The deepworlds repository contains

- Webots worlds implementing well-known problems that act as benchmarking environments, much like gym's environments
- Example worlds with solved problems, using established RL algorithms

DEEPWORLDS EXAMPLES



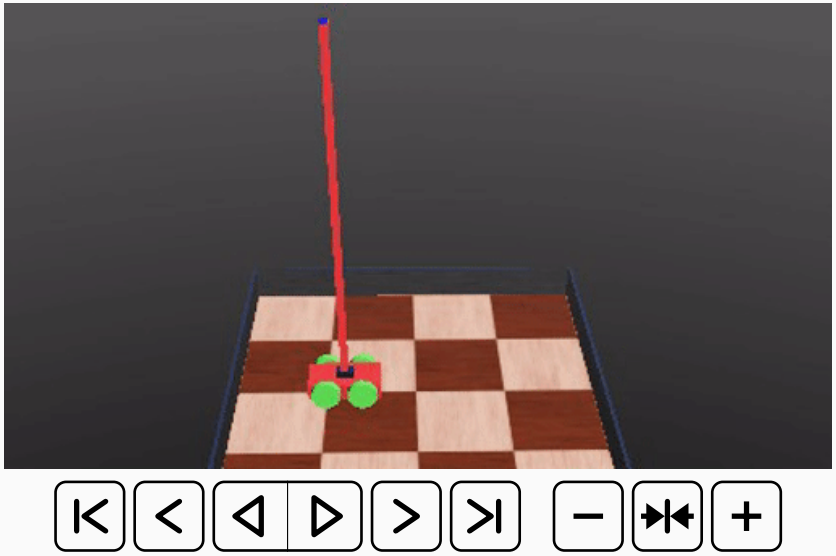
(a) CartPole

(b) PitEscape

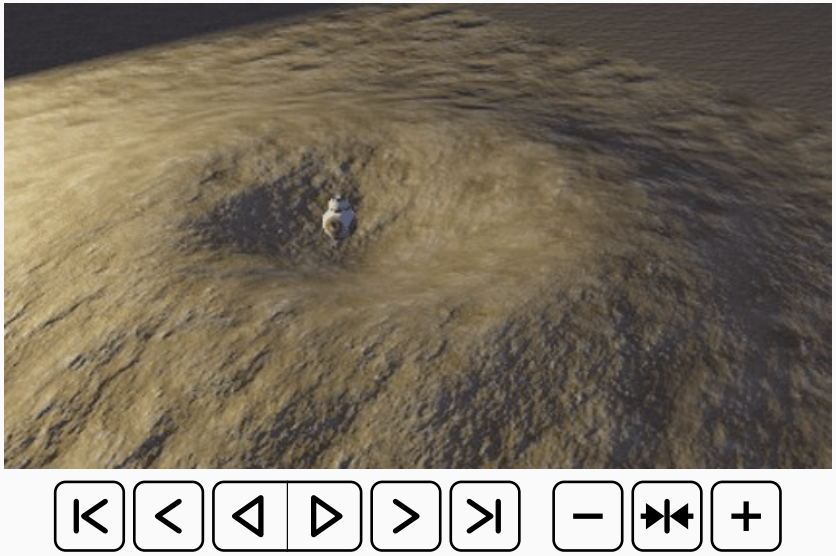
(c) FindTarget

These are solved using the Proximal Policy Optimization (PPO) and Deep Deterministic Policy Gradient (DDPG) algorithms, using PyTorch as a backend.

DEEPWORLDS EXAMPLES



(a) CartPole



(b) PitEscape

CONCLUSION

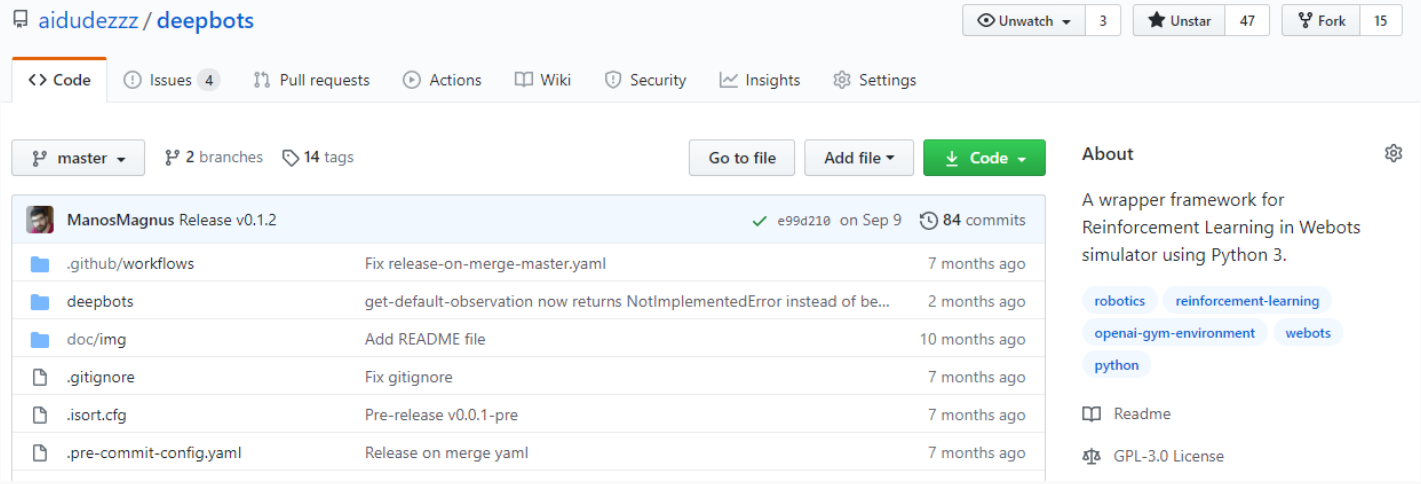
CONCLUSION

The deepbots framework:

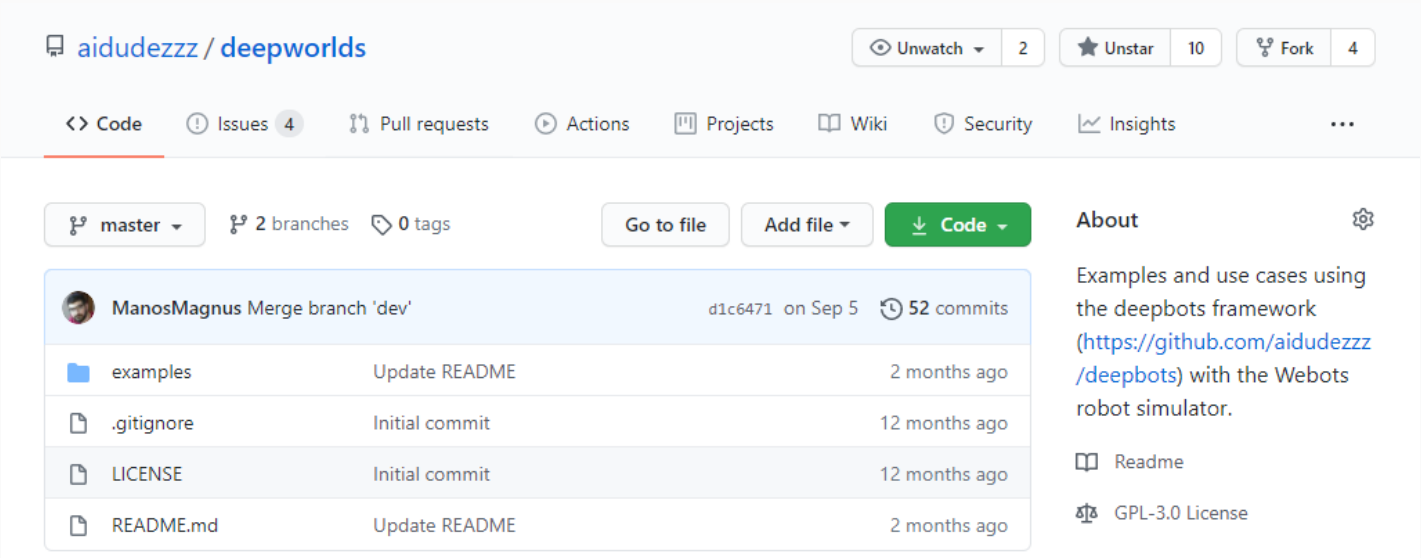
- Aims to bring the Deep RL and Robotics community together in a standardized ecosystem
- Bring a highly customizable simulator closer to the DRL community
- Help students delve into DRL, or DRL in robotics, easily

CONCLUSION

www.github.com/aidudezzz/deepbots



www.github.com/aidudezzz/deepworlds



ACKNOWLEDGMENTS

This work was supported by the European Union's Horizon 2020 Research and Innovation Program (**OpenDR**) under Grant 871449. This publication reflects the authors' views only. The European Commission is not responsible for any use that may be made of the information it contains.

Project Site: `https://opendr.eu`

CONCLUSION

Thank you!

Questions?