EAGER

Release 0.1.0

Bas van der Heijden, Alexander Keijzer, Jelle Luijkx

Feb 10, 2022

USER GUIDE

EAGER is a physics engine abstraction layer for reinforcement learning. EAGER allows switching between simulators and reality with a single line of code, supports action and observation processing and controller switching for resets.

EAGER is built on ROS 1.

CHAPTER

ONE

INSTALLATION

1.1 Prerequisites

EAGER requires ROS Melodic or Noetic and python 3.6+ to be installed.

Note: Running in ROS Melodic is supported but may cause issues in some cases. Make sure you have installed all modules needed to run ROS on python 3.

In Melodic, the gym environment side of EAGER will need to be run in python 3 but the physics bridge side can be run in both python 2 or python 3. This should allow users to run robotics hardware that does not support Noetic and python 3.

1.2 Stable Release

At this time no stable version has been released yet.

1.3 Bleeding-edge version

Currently, only a bleeding-edge version is available. We provide two ways to install the bleeding-edge version, i.e. cloning the repository into a catkin workspace and installation with pip.

1.4 Direct installation from the repository

To install EAGER directly from the repository follow these steps:

If you do not have a ROS workspace yet create one:

```
mkdir -p catkin_ws/src
cd catkin_ws/src
```

Clone the eager repository into the `src` folder:

```
git clone https://github.com/eager-dev/eager.git
```

Note: At this time two requirements to run the UR5e cannot be retreived using rosdep. Please clone these into the `src` folder using the following commands:

```
git clone -b melodic-devel https://github.com/ros-industrial/universal_robot.git
git clone -b kinetic-devel https://github.com/ros-industrial/ur_modern_driver.git
```

Move to your workspace folder and run rosdep to install any requirements:

```
cd ..
rosdep install --from-paths src --ignore-src -r -y
```

Then build and source the packages:

```
catkin_make
source devel/setup.bash
```

1.5 Installation with pip

Installation with pip also provides the possibility to perform a custom installation rather than full installation with all EAGER packages. This can done via HTTPS or SSH.

- Using HTTPS, run:

```
pip install git+https://github.com/eager-dev/eager
```

- Using SSH, run:

```
pip install git+ssh://git@github.com/eager-dev/eager.git
```

Now install EAGER by running:

```
install_eager
```

The bash script `install_eager` will clone the repository and create a catkin workspace at desired locations. It also asks for input in order to create links to the desired packages in this workspace. Afterwards, it will build the workspace. In order to use EAGER, the only thing that is required is sourcing:

```
source [EAGER_WORKSPACE_LOCATION]/devel/setup.bash
```

It is possible to run `install_eager` multiple times in order to install additional packages later.

CHAPTER

TWO

GETTING STARTED

2.1 Initializing an Engine

Switching and initializing physics engines is very simple in EAGER. Currently the following physics engines are supported:

2.1.1 WeBots

2.1.2 PyBullet

2.1.3 Gazebo

2.1.4 Reality

In the codeblock below we import the engine class and create a default world configuration.

```
from eager_bridge_webots.webots_engine import WebotsEngine
engine = WebotsEngine()
```

The classes here do not actually do anything when initialized but only store parameters to be used when starting the engine.

2.2 Initializing Objects

To add objects to the world function create them using the `create` funcion in the `Object` class.

In the codeblock below we create a single robot of type `ur5e` from the `eager_robot_ur5e` package with the name `robot1`.

```
from eager_core.objects import Object
ur5e_robot = Object.create('robot1', 'eager_robot_ur5e', 'ur5e')
```

As long as objects do not have the same name as many as needed can be added to an environment. These objects do not do anything yet when initialized and will only start when passed on to an `EagerEnv`.

2.3 Creating an EagerEnv

Now an envirnment can be made with the created objects and the simulator. As soon as the `EagerEnv` is initialized the chosen physics engine will start and the objects will be added to the world.

In the codeblock below we create a default `EagerEnv` with the single robot we created.

```
from eager_core.eager_env import EagerEnv
env = EagerEnv(engine, objects=[ur5e_robot])
```

From this point on the `EagerEnv` works exactly as any other gym Env. The full observation and action space of all the objects of the world will be combined. To alter the behaviour of the environment and its observations and actions to be exposed to the reinforcement learning algoritms the `BaseEagerEnv` can be subclassed.

2.4 Full Example

In this example we create a WeBots environment with a single UR5e robot. Once started we step the environment with random actions from the action space and then close the environment.

```
#!/usr/bin/env python3

import rospy
from eager_core.eager_env import EagerEnv
from eager_core.objects import Object
from eager_bridge_webots.webots_engine import WebotsEngine

if __name__ == '__main__':

    rospy.init_node('ur5e_example')

    engine = WebotsEngine()

    ur5e_robot = Object.create('robot1', 'eager_robot_ur5e', 'ur5e')
    env = EagerEnv(engine, objects=[ur5e_robot])

    for _ in range(1000):
        env.step(env.action_space.sample())

    env.close()
```

CHAPTER

THREE

MODULE EAGER ENV

This is the main interface of EAGER. These classes extend `gym.Env` will run the simulators.

3.1 EagerEnv

3.2 BaseEagerEnv

CHAPTER

FOUR

MODULE OBJECTS

This module contains the classes to add objects to an EAGER environment.

4.1 Object

4.2 Sensor

4.3 Actuator

4.4 State

4.5 BaseRosObject

CHAPTER

FIVE

MODULE EAGER BRIDGE WEBOTS

This is interface to launch Webots from EAGER.

5.1 WebotsEngine

CHAPTER

SIX

MODULE EAGER BRIDGE PYBULLET

This is interface to launch Pybullet from EAGER.

6.1 PybulletEngine

CHAPTER

SEVEN

MODULE EAGER BRIDGE GAZEBO

This is interface to launch Gazebo from EAGER.

7.1 GazeboEngine

CHAPTER

EIGHT

MODULE EAGER BRIDGE REAL

This is interface to real robots from EAGER.

8.1 RealEngine

CHAPTER

NINE

CONTRIBUTING

N/A