Gym-Ignition: Reproducible Robotic Simulations for Reinforcement Learning

Diego Ferigo[1,2], Silvio Traversaro[1], Daniele Pucci[1]

***Abstract*— This paper presents Gym-Ignition, a new framework to create reproducible robotic environments for reinforcement learning research. It interfaces with the new generation of Gazebo, part of the Ignition Robotics suite. The modular architecture of Ignition Gazebo allows using the simulator as a library, simplifying the interconnection between the simulator and external software. Gym-Ignition enables the creation of environments compatible with OpenAI Gym that are executed in Ignition Gazebo. The Gym-Ignition software architecture can interact natively with programs developed in low-level languages. The advantage is twofold: it permits direct interaction with robots in real-time, and it simplifies the embedding of existing software such as low-level robotic controllers.**

I. INTRODUCTION AND BACKGROUND

Simulations have always been a key component in robotics. Over the years, their accuracy and efficiency constantly improved, and nowadays there are numerous valid physic engines and simulators. They became part of every roboticist toolbox and always collected great contributions from the community.

Reinforcement Learning (RL) algorithms, to solve their decision making problems, need to operate on an environment. Classical benchmarks used in this research field typically involve grid-worlds or simple toy problems. However, the advent of Deep Learning and its combination with RL, allowed machines to solve complex decision making tasks that have been out of their reach until now. New benchmarks involving harder tasks have been developed, mainly originating from the gaming realm.

The new boosted capability of RL attracted much interested from many other research topics. Robotics is one of those that can benefit at most from an effective application. Examples range from manipulation to locomotion, both affected by a very high complexity which generally demands a tedious heuristic tuning.

Analogously to classic robotic research, also the application of RL to robotics has always tried to take advantage of simulated scenarios. The motivations, in this case, are even more critical since the system to control are costly and delicate. The intrinsic need for exploration during the training phase might be dangerous to the robots and their surrounding. Moreover, even if safe constraints are used while training, collecting enough experience only using real-world interaction is not always enough. Simulations can generate synthetic experience that can be fed into the policy, overcoming from one hand the limitations of real-time collections, but introducing from the other a bias caused by unavoidable modeling approximations.

Many advances have been done in the direction of exploiting more realistic simulations. The community already developed many frameworks that either target or can be adapted for robotic applications. Most of the solutions, though, only support a single physics engine. Furthermore, they either do not provide enough flexibility about the robot and the scene [1], are not deterministic[1], present limitations that affect the speed of experience collection [2], are not open source [3], or are not based on tools familiar to roboticists [4]. Few of these limitation are intrinsic of the architecture of the framework. Due to this reason, we decided to develop a new solution that targets specifically robotics exploiting existing native robotic tools.

In this work we present Gym-Ignition[2], a framework to create reproducible reinforcement learning environments for robotic research. Gym-Ignition exploits the new generation of the Gazebo simulator [5] part of the Ignition Robotics[3] suite. This project aims to narrow the gap between reinforcement learning and robotic research. It permits roboticists to create environments using familiar tools like Gazebo, SDF, and URDF files to model both the robot and the scenario. These environments are then exposed to the Python language wrapped by the common OpenAI Gym interface [6], making them compatible with the multitude of libraries developed by the reinforcement learning community.

To our knowledge, this is the first work that integrates with the new robotic suite developed by Open Robotics[4]. We believe that it will progressively take over the current generation of the simulator, providing new features, enhanced performance, and improved user experience.

II. ARCHITECTURE

The architecture of this project permits the same Python agent to learn on both simulated and real robot, without changing its code. This transparency is achieved mainly exploiting three levels of abstraction, illustrated in Figure 1:

- **Environment:** The environment is the interface exposed to the agent. The agent can set an action, and gather the observation and the associated reward. Actions and observations are samples belonging to a specific space.
- **Task:** The task is the interface that contains the logic about how to process the action received from the agent, and to create the observation sample. It also

[1]Italian Institute of Technology, Dynamic Interaction Control, Via San Quirico 19d, Genova, 16163, IT

[2]University of Manchester, Machine Learning and Optimisation, Oxford Road, Manchester, M13 9PL, UK

[1]http://wiki.ros.org/openai_ros

[2]https://github.com/robotology/gym-ignition

[3]https://ignitionrobotics.org

[4]https://www.openrobotics.org

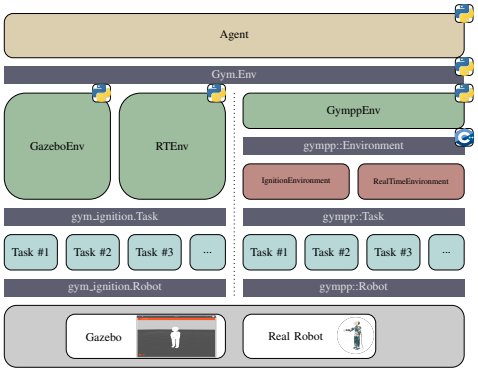

Fig. 1. Gym-Ignition software architecture, highlighting the abstraction layers. The two columns show the Python and C++ components.

calculates the reward. The task has typically access to more complete information about the state with respect to the agent.

- **Robot:** The robot is the interface that abstract both real and simulated robots. For robotics applications, the task needs to gather data from the robotic platform. The implementation of the robot interface provides kinematic, dynamic, and sensory data to the task implementation.

Taking a common toy problem in reinforcement learning, the robot can be a cartpole robot (either simulated or real), tasks can be either *pole balancing* or *swing-up*, both accepting either discrete or continuous forces applied to the cart.

Gym-Ignition is a project that aims to connect reinforcement learning libraries containing algorithms to common tools used in robotic research and industry. These two domains are historically grounded to Python and C++ languages, respectively. Gym-Ignition allows implementing environments in both languages through the following three main components:

- **`gympp`:** A C++ port of Python OpenAI Gym that provides the same environment and spaces APIs. It also contains the interface to abstract the robot.
- **`gympp-gazebo`:** A C++ library containing `GazeboWrapper` and `IgnitionEnvironment`. The former is a class that wraps the simulator and simplifies the interfacing with its features. The latter is an implementation of the `gympp` interface to create pure C++ simulated environments.
- **`gym-ignition`:** A Python package containing OpenAI Gym environments created with the Ignition Robotics libraries. Environments can be implemented either in C++ as Ignition plugins exposed through `IgnitionEnvironment`, or in Python using `GazeboWrapper` to communicate with the simulator.

This architecture allows taking the best from the two domains, represented by Python and C++. In fact, from one hand, the majority of the open source machine learning frameworks and the libraries of reinforcement learning algorithms typically target Python. They are mature and well documented tools, and the community behind their development is very active. On the other hand, instead, C++ is the main language of robotics. The majority of the robotic middleware is implemented in C++, and most of the robotic laboratories have their entire infrastructure implemented in this language. The possibility to interact and natively interface with such infrastructure might accelerate system integration.

Most of the alternative solutions to create robotic environments involve network communication to the simulator. This approach has both benefits and limitations. The network acts as an abstraction layer, and Python code does not need to be directly interfaced with low-level code. However, scaling up simulations with distributed and parallel environments might require complex network segmentation. Ignition Gazebo can be used as a library, and hence integrated into the same component of the agent. As a consequence, obtaining a reproducible simulation becomes considerably easier.

III. LIMITATIONS, FUTURE WORK, AND CONCLUSIONS

This work presents Gym-Ignition, a novel framework to create robotic environments for reinforcement learning applications. It currently supports simulated execution in the new Ignition Gazebo simulator, and it is being extended for real-time applications. We described its main features, primarily related to the current development status of the chosen simulator.

Most of the limitations for reinforcement learning research are related to the recentness of Ignition Gazebo, that did not yet reach feature parity with the current generation. Today, the simulator has full support of the DART physic engine and will gain the Bullet support in the next releases. When the simulator maturity will increase, Gym-Ignition might become one of the few projects that provide environments agnostic from the physic engine, that can be switched on-the-fly. Common sensors such as IMUs and cameras are already implemented, and few more such as force-torque sensors are already scheduled. Ignition Gazebo already has partial support of the Nvidia OptiX Engine[5], which provides realistic rendering capabilities that can be exploited for applications like visuomotor end-to-end learning.

Another notable feature under development is Ignition Fuel[6], a database containing 3D models and worlds. These resources can ease the creation of structured environments where the robot operates and interacts.

Finally, we dedicated much effort to support scaling up the simulations. Both Python and C++ environments can run accelerated and be safely distributed on multiple processes and machines. The support of cloud computation with tools like Google Colaboratory[7] is already possible in a preview stage.

ACKNOWLEDGMENTS

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No. 731540 (An.Dy).
The content of this publication is the sole responsibility of the authors. The European Commission or its services cannot be held responsible for any use that may be made of the information it contains.

[5]https://developer.nvidia.com/optix
[6]https://app.ignitionrobotics.org/dashboard
[7]https://colab.research.google.com

REFERENCES

[1] E. Coumans and Y. Bai, *Pybullet, a python module for physics simulation in robotics, games and machine learning*, 2016.

[2] N. G. Lopez, Y. L. E. Nuin, E. B. Moral, L. U. S. Juan, A. S. Rueda, V. M. Vilches, and R. Kojcev, "gym-gazebo2, a toolkit for reinforcement learning using ROS 2 and Gazebo," 2019.

[3] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," 2012.

[4] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A General Platform for Intelligent Agents," 2018.

[5] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," 2004.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," 2016.