







# RobotDART: a versatile robot simulator for robotics and machine learning researchers

Konstantinos Chatzilygeroudis <sup>1,2¶</sup>, Dionis Totsila <sup>3,4</sup>, and Jean-Baptiste Mouret <sup>3</sup>

<sup>1</sup> Laboratory of Automation and Robotics (LAR), Department of Electrical & Computer Engineering, University of Patras, Greece <sup>2</sup> Computational Intelligence Lab (CILab), Department of Mathematics, University of Patras, Greece <sup>3</sup> Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France <sup>4</sup> Computer Engineering and Informatics Department (CEID), University of Patras, Greece ¶ Corresponding author

DOI: [10.21105/joss.06771](https://doi.org/10.21105/joss.06771)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Sébastien Boisségault](#)  

## Reviewers:

- [@c-joly](#)
- [@bstanciulescu](#)

Submitted: 03 May 2024

Published: 16 October 2024

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

Robot simulation plays a pivotal role in robotics and machine learning research, offering a cost-effective and safe means to develop, validate, and benchmark algorithms in various scenarios. With the growing complexity of robotic systems and the increasing demand for data-driven approaches in machine learning, there is a pressing need for versatile and efficient robot simulators that cater to the diverse requirements of researchers. In response to this demand, we introduce RobotDART, a high-performance and versatile robot simulator designed to empower researchers in robotics and machine learning with a powerful and flexible simulation environment.

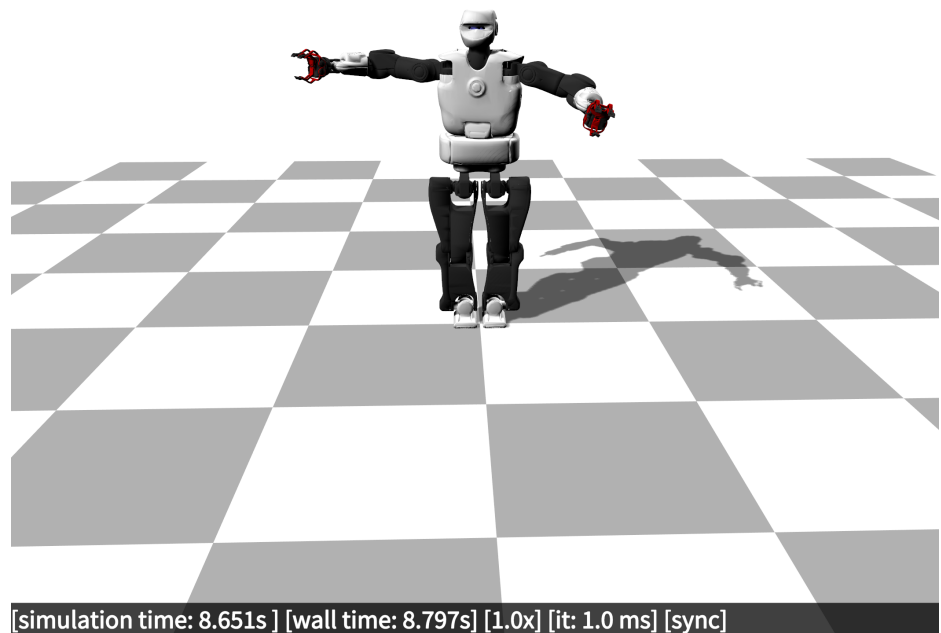


Figure 1: Talos Humanoid robot ([Stasse et al., 2017](#)) in RobotDART.

## Statement of need

Simulators play a crucial role in robotics by providing a safe environment for testing without the risk of damaging real robots. Within the robotics community, existing simulators primarily prioritize seamless deployment on physical robots, which is evident in their design philosophy. One of the best examples is [Gazebo](#) ([Koenig & Howard, 2004](#)), which is typically accessed through ROS, as an abstraction level, so that users can seamlessly transition between simulated and real robots. Because of this design choice, Gazebo: (1) is mostly asynchronous, which means that the simulator operates on its own clock and time-step analogous to real-world robots, (2) introduces a significant overhead, including network communication and serialization to topics/services/messages, and (3) is not designed for running multiple simulations concurrently or even sequentially. At a lower abstraction level, the robotics community has developed various physics libraries such as Mujoco ([Todorov et al., 2012](#)), Bullet ([Coumans & Bai, 2016](#)), and ODE ([Smith & others, 2005](#)). However, these libraries typically lack sensor abstractions, particularly for cameras, and do not come with pre-validated robot models, necessitating users to program their own simulators.

With the emergence of reinforcement learning for robot control and more generally robot learning ([Ravichandar et al., 2020](#)) ([Chatzilygeroudis et al., 2019](#)) ([Kroemer et al., 2021](#)), there arises a need for simulators that can address novel requirements not adequately addressed by current simulators or “low-level” physics libraries. Specifically, simulators designed for robot learning must encompass the following features:

- beyond real-time simulation speed: Given the necessity to execute thousands or even millions of simulations, both controllers and simulators stand to benefit from operating at speeds faster than real-time;
- overhead-free and fast reset: with the imperative to conduct numerous simulations efficiently, minimizing time introduced by the communication overheads and the initialization processes becomes paramount;
- synchronous operation: ensuring reproducible learning runs and effectively managing faster-than-reality simulations necessitates synchronization of simulation and controller time-steps;
- thread safety: modern computers with their many of cores can be leveraged to execute many simulations in parallel but the simulator needs to be designed in accordance;
- camera and depth sensor simulation: deep reinforcement learning has demonstrated promising results in learning vision-based policies through end-to-end learning; however, this demands the simulation of both depth and color cameras;
- ease of use, flexibility and versatility: the field of machine learning is characterized by rapid development, making it challenging to anticipate the diverse needs of researchers; for instance, techniques like privileged reinforcement learning ([Joonho Lee et al., 2020](#)) and domain randomization ([Tobin et al., 2017](#)) necessitate direct access to various low-level components of the simulator to enable experimentation and innovation.

Most of these features are not implemented in the mainstreams robot simulators like Gazebo. Even if some of these features are implemented, they require too much work from the researcher. For example, to effectively use Gazebo for multi-core parallel simulations, we need to create multiple Gazebo servers with different ip addresses which need to be unique in the LAN and propagated to the correct thread; as a result, the researcher spends a lot of time in boilerplate code to handle this instead of focusing in their main research avenue. RobotDART aims at filling this gap by providing a flexible simulator for data-driven robotics while being open-source and as simple as possible.

## Design and Implementation

RobotDART is built upon the Dynamic Animation and Robotics Toolkit (DART) ([Jeongseok Lee et al., 2018](#)), a robust open-source physics engine known for its accuracy and efficiency in simulating rigid body dynamics and articulated systems.

One of the key strengths of RobotDART lies in its versatile architecture, which allows researchers to easily customize and extend the simulator according to their specific research needs. RobotDART provides a collection of pre-built robot models, sensors, and environments, covering a diverse range of scenarios commonly encountered in robotics research. Researchers can seamlessly integrate their own robot models, sensors, and environments into RobotDART, thanks to its flexible library design and intuitive Python API.

Unlike other simulation frameworks, like Gazebo, RobotDART runs headless by default, and there is the possibility of rendering the scene (e.g., through a camera sensor) without opening a graphics window. All RobotDART's code is thread-safe (including graphics and camera sensors), and thus enables its users to use their code in parallel jobs in multicore computers. RobotDART is intended to be used by Robotics and Machine Learning researchers who want to write controllers or test learning algorithms without the delays and overhead that usually comes with other simulators.

### Main features:

- High-performance simulation engine based on the DART physics engine.
- Support for various robot platforms, including manipulators, mobile robots, and humanoid robots.
- Library of sensors (cameras, lidars, IMUs).
- Modular environment system for creating custom simulation scenarios.
- Intuitive Python API for easy integration with machine learning frameworks.
- Modular graphics API for easily creating custom render pipelines.
- Thread-safe code that enables parallelization.

## RobotDART in Scientific Works

RobotDART has been actively used in many scientific research projects and publications within the domains of robotics, robot learning, reinforcement learning, evolutionary computation and artificial intelligence.

### Research Projects

1. H.F.R.I. project [NOSALRO](#), PI: Konstantinos Chatzilygeroudis, 2022-2025
2. UKRI project [RoboHike](#), PI: Dimitrios Kanoulas, 2022-2025
3. EPSRC project [REcoVER](#), PI: Antoine Cully, 2020-2022
4. ERC Starting Grant project [Resibots](#), PI: Jean-Baptiste Mouret, 2015-2020
5. H2020 project [AnDy](#), PI: Daniele Pucci, 2017-2021

### Scientific Publications

Type	#	Indicative Venues
Conference	>20	ICRA, IROS, GECCO, NeurIPS, AAAI
Journal	>10	RA-L, ACM TELO, IEEE SMCA, IEEE TEVC

## Conclusion

RobotDART is a versatile and efficient robot simulator tailored to the needs of robotics and machine learning researchers. By combining the accuracy of the DART physics engine with a modular and extensible design, RobotDART provides researchers with a powerful tool for prototyping, testing, and validating algorithms in a simulated environment. We believe that RobotDART will facilitate advancements in robotics and machine learning research by providing a flexible and accessible platform for exploring new ideas and pushing the boundaries of innovation.

## Acknowledgements

The work was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the “3rd Call for H.F.R.I. Research Projects to support Post-Doctoral Researchers” (Project Acronym: NOSALRO, Project Number: 7541).

## References

- Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S., & Mouret, J.-B. (2019). A survey on policy search algorithms for learning robot controllers in a handful of trials. *IEEE Transactions on Robotics*, 36(2), 328–347. <https://doi.org/10.1109/TRO.2019.2958211>
- Coumans, E., & Bai, Y. (2016). *PyBullet, a Python module for physics simulation for games, robotics and machine learning*. <http://pybullet.org>.
- Koenig, N., & Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3, 2149–2154. <https://doi.org/10.1109/IROS.2004.1389727>
- Kroemer, O., Niekum, S., & Konidaris, G. (2021). A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of Machine Learning Research*, 22(30), 1–82.
- Lee, Joonho, Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), eabc5986. <https://doi.org/10.1126/scirobotics.abc5986>
- Lee, Jeongseok, X. Grey, M., Ha, S., Kunz, T., Jain, S., Ye, Y., S. Srinivasa, S., Stilman, M., & Karen Liu, C. (2018). Dart: Dynamic animation and robotics toolkit. *The Journal of Open Source Software*, 3(22), 500. <https://doi.org/10.21105/joss.00500>
- Ravichandar, H., Polydoros, A. S., Chernova, S., & Billard, A. (2020). Recent advances in robot learning from demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 297–330. <https://doi.org/10.1146/annurev-control-100819-063206>
- Smith, R., & others. (2005). *Open Dynamics Engine*.
- Stasse, O., Flayols, T., Budhiraja, R., Giraud-Esclasse, K., Carpentier, J., Mirabel, J., Del Prete, A., Souères, P., Mansard, N., Lamiroux, F., & others. (2017). TALOS: A new humanoid research platform targeted for industrial applications. *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 689–695. <https://doi.org/10.1109/HUMANOIDS.2017.8246947>
- Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 23–30. <https://doi.org/10.1109/IROS.2017.8202133>

Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 5026–5033. <https://doi.org/10.1109/IROS.2012.6386109>