20 Years of Reality Gap: a few Thoughts about Simulators in Evolutionary Robotics

Extended Abstract – SimER workshop

Jean-Baptiste Mouret
Inria Nancy – Grand Est, France
CNRS, Loria, UMR 7503, France,
Université de Lorraine, Loria, UMR 7503
jean-baptiste.mouret@inria.fr

Konstantinos Chatzilygeroudis
Inria Nancy – Grand Est, France
CNRS, Loria, UMR 7503, France,
Université de Lorraine, Loria, UMR 7503
konstantinos.chatzilygeroudis@inria.fr

ABSTRACT

Simulators in Evolutionary Robotics (ER) are often considered as a "temporary evil" until experiments can be conducted on real robots. Yet, after more than 20 years of ER, most experiments still happen in simulation and nothing suggests that this situation will change in the next few years. In this short paper, we describe the requirements of ER from simulators, what we tried, and how we successfully crossed the "reality gap" in many experiments. We argue that future simulators need to be able to estimate their confidence when they predict a fitness value, so that behaviors that are not accurately simulated can be avoided.

ACM Reference format:
Jean-Baptiste Mouret and Konstantinos Chatzilygeroudis. 2017. 20 Years of Reality Gap: a few Thoughts about Simulators in Evolutionary Robotics. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 4 pages.
DOI: http://dx.doi.org/10.1145/3067695.3082052

1 INTRODUCTION

Simulators are an integral part of the Evolutionary Robotics' literature because they allow researchers to explore their ideas more easily than evolution in physical systems. Given the importance of this topic, the community would benefit from a shared culture of simulators and from sharing their toolkits. In this short paper, we describe the requirements of ER from simulators, what we tried, and how we successfully crossed the "reality gap" [10, 11] in many experiments.

2 WHAT SIMULATOR DO WE NEED?

Evolutionary robotics imposes specific requirements on the simulators:

- **Fast:** Most ER experiments spend most of their time in the fitness function, that is, in running the simulator for many time-steps; it is therefore critical that the simulators are as fast as possible.
- **Short cold start:** The fitness function will be evaluated millions of times, therefore starting a new simulation should take as little time as possible.
- **Fully re-entrant and thread-safe:** A common strategy to reduce the duration of ER experiment is to parallelize the fitness evaluations (this strategy makes even more sense now that most computers have many cores); however, to do so, the simulators need to be programmed without any hidden state (e.g. static or global variables) and allow many instances to run at the same time (ideally in the same process).
- **Stable:** Evolution, like other optimization algorithms, tend to push the simulators to their limits and over-exploit "bugs" and instability; it is, for example, common in ER to observe simulated multi-body robots "explode" (break the joint constraint) because it allows the main body to travel a high-speed.
- **Accurate:** more accuracy is always better as it reduces the reality gap (see section 4); however, the level of accuracy depends strongly on the type of study (not all arguments/demonstrations need accurate physical simulations).
- **Open source:** It is critical for reproducibility that (1) other researchers can confirm and expand results without needing to buy a license for some closed-source software, (2) that simulators can be updated to run on new systems / operating systems even if the company that made the original software does not exist anymore, (3) it is possible to know exactly what computation is performed; for these reasons, scientific work in ER needs to focus on open source solutions.
- **Stable API:** As simulators are not the main research focus of the community, we would like to spend as little time as possible on maintaining them.

These requirements are often different from those of "classic" robotics [9]. For instance, simulators used in mechanical design do not need to be thread-safe and can take a long time to start; and they often favor accuracy over speed. On the contrary, simulators for video-games do not need physical accuracy (only visual accuracy) and are often primarily optimized for speed.

3 WHAT DID WE TRY?

Over the years, our team implemented several open source simulators by taking advantage of the available open source physics libraries; all of them have been interfaced with Sferes$_{v2}$ [20] (http:

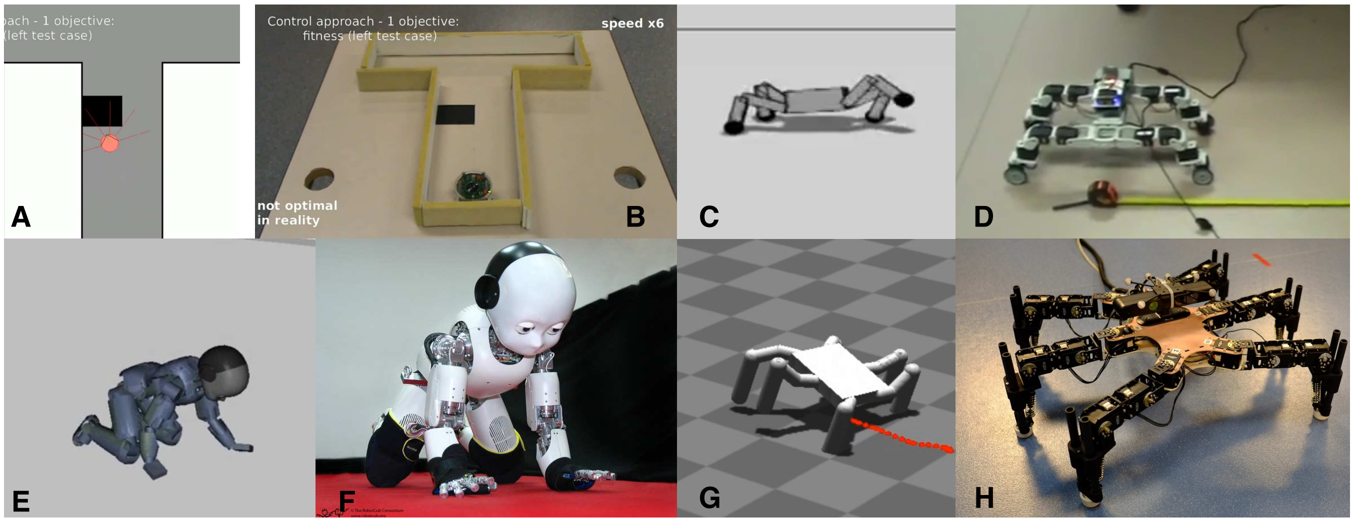

Figure 1: Examples of simulated and real robots we have used in our research. A. Fastsim (fast 2-D kinematic simulator) [16]. B. E-puck robot in the same maze as in A [16]. C. Quadruped robot simulated in RobDyn (Bullet) [14–16, 22]. D. Real quadruped robot based on Dynamixel actuators (AX-12) [14–16, 22]. E. iCub robot crawling simulated in our new Dart-based simulator [23]. F. Real iCub robot crawling. G. Hexapod robot simulated in RobDyn (ODE) [4–6, 13]. H. Real hexapod robot [3–6, 13].

//github.com/sferes2/sferes2), our framework for evolutionary computation:

- **Fastsim:** A 2-D kinematic simulator for differential-drive robots that is designed to be fast, simple and flexible; it is programmed in C++ – http://github.com/jbmouret/libfastsim.
- **RobDyn:** a simple wrapper of the Bullet physics library (http://bulletphysics.org/), that we later transformed to a wrapper of the ODE physics library (http://www.ode.org); it is designed to simulate the dynamics of legged robots (e.g. hexapods or quadruped) and common position-controlled servos (e.g., Dynamixel) – http://github.com/resibots/robdyn.
- **Dart-based simulator:** a wrapper around the Dart physics library (https://github.com/dartsim), which is faster in our experiments than ODE or Bullet, and which can load URDF files (Universal Robotic Description Format, which is the XML file format used in ROS to define the kinematic and dynamic properties of a robot) – https://github.com/resibots/robot_dart.

We tried 3 physics engines: ODE, Bullet, and Dart. ODE is well-documented and its API (in C) did not change in many years, but it does not include recent developments like a GPU-accelerated pipeline or the Featherstone's solver for multi-body dynamics [8, 18]. Bullet was not very documented when we tried it and its API was changing frequently, but it includes more modern ideas. Dart is a new physics library designed with robots and animation in mind. It has a modern C++11 API and is faster than ODE in our experiments; however, it is under intense developments and changes often (it seems that it is close to having a more or less fixed API and the developers are very responsive).

In addition, we investigated more advanced simulation environments like Gazebo [12] (http://www.gazebosim.org) and V-Rep [26] (www.coppeliarobotics.com), but they added too much overhead to be efficiently used in our experiments (their communication layer slows the simulation down for simple robots, they can be difficult/slow to restart, and they put a lof of constraints on the interface; it is also not easy to run many instances of Gazebo in parallel).

4 THE REALITY GAP

In our experiments, no simulator led to any notable boost of accuracy and all of them led to reality gap issues. As a consequence, instead of working on more accurate simulation, we explored many ideas to overcome or bypass the reality gap.

One of our most promising idea is the *transferability approach* [6, 13–16, 22]: instead of attempting to correct the simulator to make better predictions, a supervised learning algorithm uses tests on the real robot to learn the limits of the simulator, that is it learns a mapping between behavior descriptors, extracted from the simulator, and a prediction of the accuracy (or confidence) of the simulation. Intuitively, learning these limits is easier than correcting the simulator because (1) there are effects that cannot be easily modeled in a simulator (e.g. aerodynamic effects in a simulator that does not implement aerodynamics), , especially with a few data points (e.g. 20 experiments on the robot), and (2) many non transferrable ("cheating") behaviors are easy to dismiss (e.g. exploding robots that travel at high speed, unrealistic jumps, etc.).

When the evolutionary algorithm knows the limits of the simulation, it can select individuals with good fitness in simulation and high predicted transferability (e.g., with a multi-objective evolutionary algorithm or a constrained evolutionary algorithm). Put differently, the evolutionary algorithm avoids the parts of the search space that are not accurately simulated. Using this approach, we discovered that current physics engine (e.g. ODE) can be surprisingly accurate for some behaviors. For instance, on a quadruped robot, we observed a difference of fitness of less than 10% for high-performing behaviors evolved with the transferability approach, versus a difference of more than 100% when we transferred the best individual obtained in simulation to the real robot.

Using the same intuition but a different process (the MAP-Elites algorithm [19]), we generated thousands of potential solutions (e.g. gaits) in simulation, then selected online using a data-efficient algorithm (here Bayesian optimization [27]) the one that works best on the physical robot [4]. We performed our experiments with a damaged, 6-legged robot whereas the simulations were performed with an intact robot. The reality gap was, therefore very important since we added the issues of the simulator to the damage; nonetheless, we were able to find high-performing gaits for all the damage conditions that we tested in only a dozen of tests on the real robot [4]. In this process, the online algorithm (Bayesian optimization) searches for the solutions that transfer well, that is, that are high-performing both in simulation and in reality.

Combining the transferability approach with "Novelty Search with Local Competition" [17] (an illumination [19] or quality-diversity [24] algorithm), we were able to evolve *behavioral repertoires* so that a 6-legged could walk in any direction [7]. Overall, the results showed that only a few short experiments on the physical robot were needed by the algorithm in order to generate a repertoire of controllers that allows the robot to reach every point in its reachable space. In more recent work, we exploited the same idea of repertoire creation (in simulation only with the intact robot) and combined it with online adaptation (via Gaussian process regression [25]) and replanning (using Monte Carlo Tree Search [1]) to achieve fast recovery of a damaged physical 6-legged robot without any human intervention and while taking the environment into account [3].

5 THE MYTH OF THE PERFECT SIMULATOR

Overall, we do not think that simulators will ever be both fast enough for evolution and accurate enough to cross the reality gap easily. The alternative would be to propose a new generation of simulators that can predict a behavior (and therefore a fitness value) but also *estimate their confidence in this prediction.* This confidence measure is what is learned with the transferability approach, but it could be computed or estimated by the simulator [1]: after having simulated a behavior, we could retrieve the fitness value (e.g. the distance covered by a walking robot) and the confidence in the fitness. Typically, the confidence would be low for highly-dynamic behaviors or for behaviors that highly rely on some features of the environment (e.g. the friction coefficient), and high for more robust, static behaviors.

Such simulators would allow algorithms to avoid overfitting the simulator (e.g. using multi-objective evolution) and thus to greatly increase the transferability of behaviors evolved in simulation. At least two approaches could be investigated to add confidence levels in ER simulators: (1) Monte Carlo estimates, that is, running many simulations with variations of parameters (e.g. variations of the friction level) and measure the mean and variance, or (2) crowd-sourced estimates, that is, building an online database of experiments with a simulator and the result on the real robot (provided that there are enough users of the same simulators).

Last, not all ER experiments need accurate, physical simulators. For instance, our work on behavioral diversity [21] did not need such a simulator to make our point (which is, that diversity in behavior space needs to be encouraged). As a consequence, we do not *always* need an accurate simulation; instead, we need *flexible* simulators that can be fast or accurate, depending on the experiment and the stage of the project.

[1]This is what we did when we learned a "simulator" (a dynamic model) from scratch for data-efficient policy search in robotics [2].

6 ACKNOWLEDGMENTS

This work received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement number 637972, project "ResiBots").

REFERENCES

[1] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games* 4, 1 (2012), 1–43.

[2] Konstantinos Chatzilygeroudis, Roberto Rama, Rituraj Kaushik, Dorian Goepp, Vassilis Vassiliades, and Jean-Baptiste Mouret. 2017. Black-Box Data-efficient Policy Search for Robotics. *arXiv preprint arXiv:1703.07261* (2017).

[3] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. 2016. Reset-free Trial-and-Error Learning for Data-Efficient Robot Damage Recovery. *arXiv preprint arXiv:1610.04213* (2016).

[4] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. 2015. Robots that can adapt like animals. *Nature* 521, 7553 (May 2015), 503–507. DOI: http://dx.doi.org/10.1038/nature14422

[5] Antoine Cully and Jean-Baptiste Mouret. 2013. Behavioral Repertoire Learning in Robotics. In *The 15th Annual conference on Genetic and evolutionary computation (GECCO'13).* ACM, Amsterdam, Netherlands, 175–182. DOI: http://dx.doi.org/10.1145/2463372.2463399

[6] Antoine Cully and Jean-Baptiste Mouret. 2016. Evolving a Behavioral Repertoire for a Walking Robot. *Evolutionary Computation* 24, 1 (2016), 59–88. DOI: http://dx.doi.org/10.1162/EVCO_a_00143

[7] Antoine Cully and J-B Mouret. 2016. Evolving a behavioral repertoire for a walking robot. *Evolutionary computation* 24, 1 (2016), 59–88.

[8] Roy Featherstone. 1984. *Robot dynamics algorithms.* Ph.D. Dissertation. University of Edinburgh Scotland.

[9] Serena Ivaldi, Jan Peters, Vincent Padois, and Francesco Nori. 2014. Tools for simulating humanoid robot dynamics: a survey based on user feedback. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on.* IEEE, 842–849.

[10] Nick Jakobi. 1997. Evolutionary robotics and the radical envelope-of-noise hypothesis. *Adaptive behavior* 6, 2 (1997), 325–368.

[11] Nick Jakobi, Phil Husbands, and Inman Harvey. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life.* Springer, 704–720.

[12] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, Vol. 3. IEEE, 2149–2154.

[13] Sylvain Koos, Antoine Cully, and Jean-Baptiste Mouret. 2013. Fast Damage Recovery in Robotics with the T-Resilience Algorithm. *International Journal of Robotics Research (IJRR)* 32, 14 (Dec. 2013), 1700–1723. DOI: http://dx.doi.org/10.1177/0278364913499192

[14] Sylvain Koos and Jean-Baptiste Mouret. 2011. Online Discovery of Locomotion Modes for Wheel-Legged Hybrid Robots: a Transferability-based Approach. In *14th International Conference on Climbing and Walking Robots (CLAWAR).* Paris, France, 70–77. DOI: http://dx.doi.org/10.1142/9789814374286_0008

[15] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. 2010. Crossing the Reality Gap in Evolutionary Robotics by Promoting Transferable Controllers. In *The 12th Annual conference on Genetic and evolutionary computation (GECCO'10).* ACM, United States, 119–126. DOI: http://dx.doi.org/10.1145/1830483.1830505

[16] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. 2013. The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transactions on Evolutionary Computation (TEC)* 17, 1 (2013), 122–145. DOI: http://dx.doi.org/10.1109/TEVC.2012.2185849

[17] Joel Lehman and Kenneth O Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 211–218.

[18] Brian Vincent Mirtich. 1996. *Impulse-based dynamic simulation of rigid body systems*. Ph.D. Dissertation. University of California at Berkeley.

[19] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909* (2015).

[20] Jean-Baptiste Mouret and Stéphane Doncieux. 2010. Sferes$_{v2}$: Evolvin' in the Multi-Core World. In *12th IEEE Congress on Evolutionary Computation (CEC'2010)*. France, 1–8. DOI: http://dx.doi.org/10.1109/CEC.2010.5586158

[21] Jean-Baptiste Mouret and Stéphane Doncieux. 2012. Encouraging Behavioral Diversity in Evolutionary Robotics: an Empirical Study. *Evolutionary Computation* 20, 1 (2012), 91–133. DOI: http://dx.doi.org/10.1162/EVCO_a_00048

[22] Jean-Baptiste Mouret, Sylvain Koos, and Stphane Doncieux. 2012. Crossing the Reality Gap: a Short Introduction to the Transferability Approach. In *Proceedings of the workshop "Evolution in Physical Systems"*. ALIFE.

[23] Vaios Papaspyros, Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. 2016. Safety-Aware Robot Damage Recovery Using Constrained Bayesian Optimization and Simulated Priors. In *Bayesian Optimization: Black-box Optimization and Beyond (workshop at NIPS)*. https://hal.inria.fr/hal-01407757

[24] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. 2016. Quality Diversity: A New Frontier for Evolutionary Computation. *Frontiers in Robotics and AI* 3 (2016), 40. doi: 10.3389/frobt.2016.00040.

[25] Carl Edward Rasmussen. 2006. Gaussian processes for machine learning. (2006).

[26] Eric Rohmer, Surya PN Singh, and Marc Freese. 2013. V-REP: A versatile and scalable robot simulation framework. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 1321–1326.

[27] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. 2016. Taking the human out of the loop: A review of bayesian optimization. *Proc. IEEE* 104, 1 (2016), 148–175.