Cloud Simulations for RoboCup

Enric Cervera[1], Gustavo Casañ[1], and Ricardo Tellez[2]

[1] Robotic Intelligence Lab, Universitat Jaume I, 12006 Castelló de la Plana, Spain
[2] The Construct Sim LTD, 08007 Barcelona, Spain

Abstract. Possibly the most appealing aspect of RoboCup is working with real robots, specially for young people. Yet as the complexity of the task increases, the effort in software development becomes higher, and a simulation testbed can be a valuable tool for prototyping and testing software solutions prior to their implementation on a real robot. In fact, several RoboCup leagues feature both real and virtual competitions. In addition, the RoboCup community could benefit from the cooperation and sharing of experiences among users in an online worldwide platform. We present a simulation tool based on the cloud, which can model complex robots off the shelf by using only a web browser as the base system for learning robotics, and running competitions. Such a platform minimizes costs and the troubles associated with different operating systems, while providing a rich experience of testing, with the possibility of a straightforward transfer to a real robot. Moreover, users can easily share their simulations for cooperative learning.

Keywords: cloud robotics, simulation

1 Introduction

With the advent of powerful computers and graphic cards, 3D realistic simulators are becoming popular in RoboCup leagues. Yet programming and maintaining a simulator is complex, hard, and time consuming. A prototypical example is the RoboCup Rescue Virtual Robot Competition, launched in 2006, which uses a simulation software, USARSim (Unified System for Automation and Robot Simulation) [2], built on top of the Unreal game engine[1]. This engine has evolved along several versions (2004, UT3, UDK) that required to rewrite the simulation software from scratch. Initially maintained by the National Institute of Standards and Technology (NIST), but no longer supported, it recently switched to a different platform developed by the Open Source Robotics Foundation (OSRF) [15].

Another example is the RoboCup Junior Rescue CoSpace [7], with a simulator built on top of the Microsoft Robotics Developer Studio. This framework has not been updated since 2014, as a result there are increasing difficulties in fixing bugs, adding support for modern robots and sensors, or working with new versions of operating systems.

[1] https://www.unrealengine.com/

On the other hand, RoboCup Soccer uses an open source simulator, SimSpark [17], in the 3D simulation league. We can only wonder how much effort is replicated among those simulator platforms, which face similar problems (physical engine, graphical visualization) in different yet related domains.

In recent years we have been working on web-based laboratories for both real robots and simulators [4]. We have developed web interfaces for systems based on the ROS middleware [12, 3], which provides a hardware abstraction layer, enabling the user to share the same code between a real robot and its simulated model. Our aim is the development of a common platform for robotic simulations, suitable for any RoboCup virtual competition, and even for replicating the leagues that right now only use physical robots. Such platform would be based on a cloud infrastructure, enabling the users to run their simulations from their browsers, and to share their experiences with other users throughout the world.

The rest of the paper is organized as follows: Section 2 outlines our cloud simulation platform, along with some demonstration examples in competitions and education; in Section 3, we advocate for the adoption of a common, open-source, cloud simulation platform in RoboCup leagues; finally, Section 4 draws some conclusions and outlines some future lines of work.

2 Cloud Simulation Platform

RDS[2] (ROS Development Studio) is a web application for the simulation of robots in the cloud. The platform consists of Virtual Machines (VMs) [16] running in the cloud infrastructure provided by Amazon Web Services (AWS) [8]. Each user connects to a single, dedicated VM, running a full-featured distribution of Ubuntu Linux with all the necessary software already installed and configured: ROS, simulators, and development tools.

An advantage of VMs is that they can be mapped to different physical machines based on the power and memory requirements, e.g., a low-complexity simulation can run in single CPU with low memory, but a high-fidelity complex environment may use a multi-CPU machine with one or several additional GPUs and a larger amount of RAM. In any case, the user connects to the VM through a client machine, with no special power or system requirements, since only a browser is needed.

The user interface consists of a web page with a login and password, which gives access to all the tools through the web browser: no other software is needed in the client computer. Different windows in the browser are used for the components of the platform (notebook, simulation view, file editor, shell) as depicted in Fig. 1.

This interface works on top of any WebGL [9] enabled browser, like Safari, Chrome or Firefox. It can be used with any computer or device running any of those browsers, in any operating system, including Linux, Windows, Mac, or even tablets and smartphones.

[2] http://env.theconstructsim.com/RDS

Fig. 1. User interface: from left to right, notebook, simulator view, file editor (top) and shell (bottom).

The Robot Operating System (ROS) has become a de-facto standard among robotics researchers as an open source framework for robot programming and control [12]. Currently, two simulators are supported in the cloud platform, Gazebo [10] and Webots [11].

ROS supports several client libraries in different programming languages. The main supported libraries are written in C++ and Python, but there is also active development in Lisp, C#, Go, Haskell, Java, Javascript, Julia, Lua, Matlab, Pharo, R, and Ruby.

The main programming interface of the cloud platform consists of the Jupyter Notebook, an open-source web application for creating and sharing documents that contain live code, data visualization and explanatory text [13], which supports over 40 programming languages, including most of the languages supported in ROS.

A cloud platform provides the user with the infrastructure for creating a social network that encourages the interaction between the community members [1]. Users can share their simulations and code, collaborate or compete against each other. Such communities, e.g. the Scratch platform [14], have an incredible educational potential in computer and engineering disciplines.

In the following we present some working applications of the cloud platform: two online competitions, and an online course.

2.1 Online Competitions

One way to encourage students and robotics research is by doing contests where the participants have to compete against each other. Two competitions have been organized using the cloud platform and simulations of humanoid robots: a NAO robot race, and sumo fighting between two Darwin robots.

In the racing competition (Fig. 2), a Nao humanoid has to be programmed to walk 10 meters as fast as possible. Participants are given a standard walking controller, which can be modified and optimized for speed and robustness. Modifications are uploaded to each user's account, and the system performs the simulations and compares the results.

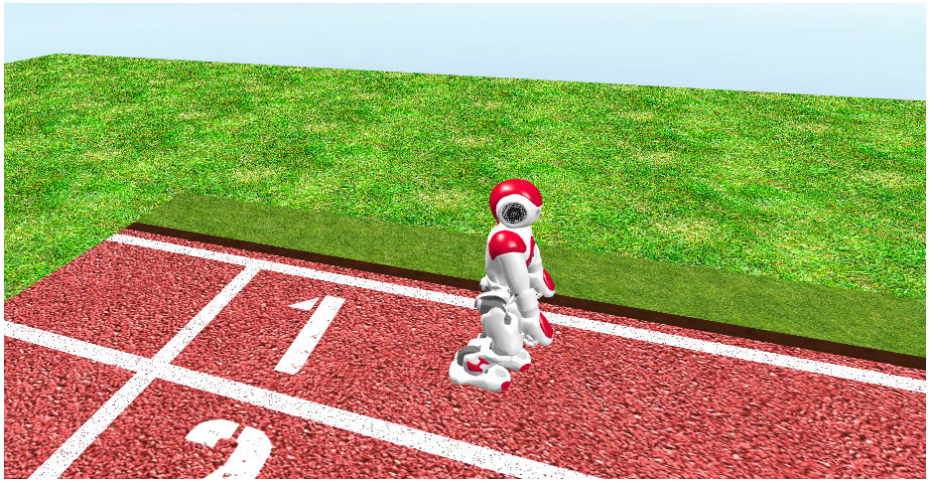

Fig. 2. Nao race: the robot is programmed for walking 10 meters as fast as possible, the time is measured by an automatic chronometer at the finish line.

The Sumo Challenge (Fig. 3) was a worldwide contest, consisting of two simulated Darwin humanoid robots that must fight against each other in a simulated sumo dojo. Participants had to build a controller for their robot, trying to knock out the opponent. The controllers were automatically taken each day and made to fight against the rest of participants in a Bubble Sort style league.

2.2 Online Courses

We have recently used the cloud platform in a MOOC (Massive Open Online Course) on Autonomous Mobile Robots, where we designed a simulation environment inspired in the RoboCup Junior Rescue competition (Fig. 4). In this world, the robot must be programmed first to follow a line with obstacles and intersections, then to find and pick some balls scattered around the room, and carry them to a destination area.

The simulation platform was used in combination with a Learning Management System based on Moodle [5]. The students worked on the lessons and examples, developed the exercises on the simulators, and submitted their code through Moodle workshops, with peer assessment activity, where students submit their own work and then receive a number of submissions from other students which they must assess according to the teacher's specifications [6].

Fig. 3. Sumo challenge: the robots are programmed to fight each other according to the rules, with an automatic referee for scoring the matches.

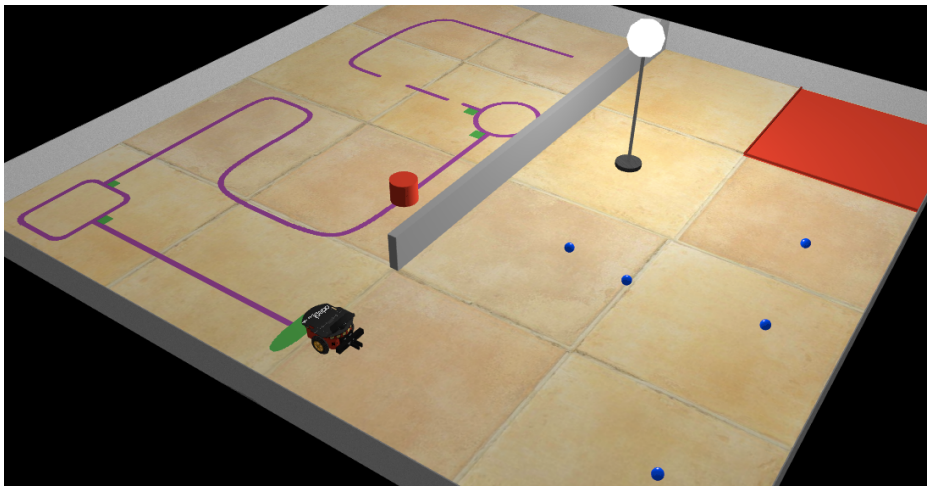

Fig. 4. MOOC simulation world inspired in the RoboCup Junior Rescue competition.

3 A Proposal for RoboCup

From its initial challenge in soccer, RoboCup has expanded into a wide range of domains. Though initially focused on the development of real robots, the advantages of simulators for fast prototyping and cost reduction has led to many of the leagues having either physical or simulated robots sub-leagues.

Each league has developed its own simulation software: while adapting to a particular domain may be an advantage, an undesirable consequence is the need of more resources for the development of software that could share a common base.

We advocate for the convergence of all the RoboCup simulators in a common open-source cloud platform. The advantages would be the optimization of the development resources and the availability of a cross-platform, powerful simulation engine, which can be run efficiently, no matter what operating system or hardware is used. The only requirements are a WebGL-enabled browser and an Internet connection. The participants would also have an easier time taking part in different competitions.

Based on the recent history of the RoboCup simulation competitions, we believe that the community is moving towards such a common, open-source platform. An example of this is the evolution of the RoboCup Rescue Simulation Platform: its was based initially on the Unreal Engine, but recently was moved to the Gazebo platform, benefitting from the progress made by the Open Source Robotics Foundation [15]. The maintenance of the simulation environment is now in hands of the open source community, since the simulator is no longer actively supported by the National Institute of Standards and Technology (NIST).

Moving this league to the cloud platform would be straightforward, since the tools (Gazebo and ROS) are already supported.

Two other different simulators are being used in RoboCup leagues: SimSpark for RoboCup Soccer Simulation [17], and Microsoft Robotics Developer Studio (MRDS) for RoboCup Junior CoSpace [7].

In this case, there are two ways for implementing the leagues in the cloud: one possibility is importing the world simulation files into Webots or Gazebo, the cloud supported simulators; the other alternative is to develop the WebGL and ROS interfaces for those simulators, which should be feasible for SimSpark since it already works in Linux, but quite problematic for MRDS, which only works in Windows.

It should be noted that supporting a specific simulator is a heavy task; in fact, MRDS has not been updated or patched since version 4.0, which was released in early 2012. Moreover, the Robotics Division of Microsoft Research was suspended in September 2014.

Other leagues that only used hardware platforms could be candidates to activate their simulated counterpart: in RoboCup@Work, the KUKA youBot platform is one candidate system for use, and this platform is readily available in the Webots and Gazebo simulators (Fig. 5). Participants who can't afford such a costly physical robot would definitely benefit from the availability of a simulated environment.

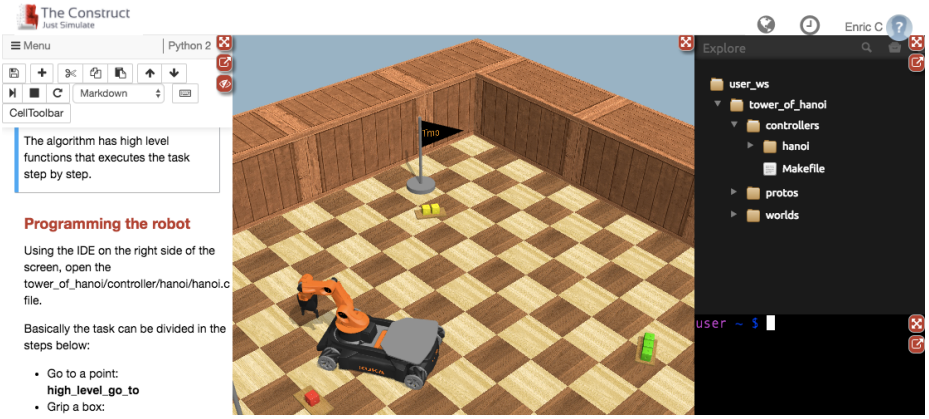

Fig. 5. Simulated environment of a KUKA youBot platform, a candidate system for the RoboCup@Work competition.

Competition based on low-cost platforms, like RoboCup Junior, can also benefit from the access to a simulation platform. It would allow the users to test their algorithms on standard platforms, or even simulate their own specific hardware design. We have already implemented the environment of the line sub-league of RoboCup Junior rescue (Fig. 6).

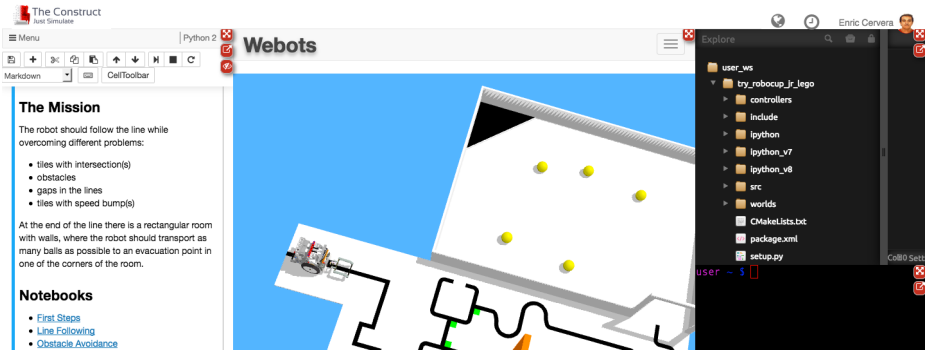

Fig. 6. Simulated environment for the RoboCup Junior Rescue competition.

3.1 Advantages of cloud simulators

These are the main advantages of using a web based simulator:

1. They do not need installation nor maintenance. Such operations are performed by the web portal that provides the simulator. Hence, neither the school nor the students have to deal with this unrelated task.

2. Students cannot break the simulation program by making errors or miss use of the simulation. If students make mistakes and something goes wrong crashing the simulation, they just can relaunch the web simulation again and start again from the initial conditions.
3. Different simulators available which allows entrance at different level of complexity. A web simulation system can provide different simulators at the same portal, each one with a different level of abstraction (for high school students, for university students, etc).
4. Students can use any type of computer. Only a WebGL-enabled browser is needed.
5. Students and teachers can work from anywhere. Since the only requirement is to have access to internet, teachers and students can actually be anywhere while doing their simulations.
6. Students can cooperate with their mates while working on the simulation. Web simulation is by nature a collaborative thing, at the contrary of desktop simulations. Hence, differently from desktop simulators, web simulators allow the work of different people at the same time on the same simulation.

3.2 Drawbacks

As any system, web simulations have also drawbacks:

1. Users need a fast enough internet connection. Web simulations are run through an internet connection, and since the simulations tend to be complex, it is required to have an internet connection with fast speed.
2. By being on the cloud, web simulation inherits all the security problems that any cloud system has. It is very difficult to ensure that the data stored in the cloud will not be accessed by unauthorized people.

4 Conclusion and Future Work

We have presented a cloud simulation platform suitable for RoboCup Virtual Competitions. An open-source, common simulation platform has an important advantage over the current scattered development of specific simulators for each RoboCup league: programming and maintaining a simulator application is hard and time consuming. In addition, the cloud platform allows any user, anywhere, to run a simulation with any device equipped with a WebGL-enabled browser, be it a laptop, desktop, tablet, or even a smartphone.

Obviously, an Internet connection is needed, and the cost of the cloud infrastructure needs to be taken into account. We have presented a platform built on top of AWS, which can be used initially for free. More powerful computing configurations with an increasing number of CPUs and GPUs are available at an additional cost.

We expect that the RoboCup community opens a debate about the adoption of this or a similar platform for future editions of the virtual leagues. In the

near future, we plan to offer cloud versions for all the leagues that currently use simulators, and develop simulation versions of some other leagues that use physical robots. The cloud allows to share simulations and code, enabling the development of a lively community of RoboCup users, which would become a complement and reinforcing tool for the successful RoboCup events throughout the world.

Acknowledgement

Support of IEEE RAS through the CEMRA program (Creation of Educational Material for Robotics and Automation) is gratefully acknowledged. This paper describes research done at the Robotic Intelligence Laboratory. Support for this laboratory is provided in part by Ministerio de Economia y Competitividad (DPI2015-69041-R), by Generalitat Valenciana (PROMETEOII/2014/028) and by Universitat Jaume I (P1-1B2014-52).

References

1. Beetham, H., Sharpe, R.: Rethinking pedagogy for a digital age: Designing for 21st century learning. Routledge (2013)
2. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: USARSim: a robot simulator for research and education. In: Robotics and Automation, 2007 IEEE International Conference on, pp. 1400–1405. IEEE (2007)
3. Casañ, G.A., Cervera, E., Moughlbay, A.A., Alemany, J., Martinet, P.: ROS-based online robot programming for remote education and training. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on, pp. 6101–6106. IEEE (2015)
4. Cervera, E., Martinet, P., Marin, R., Moughlbay, A.A., Del Pobil, A.P., Alemany, J., Esteller, R., Casañ, G.: The robot programming network. Journal of Intelligent & Robotic Systems **81**(1), 77 (2016)
5. Cole, J., Foster, H.: Using Moodle: Teaching with the popular open source course management system. O'Reilly Media, Inc. (2007)
6. Dooley, J.F.: Peer assessments using the Moodle workshop tool. In: ACM SIGCSE Bulletin, vol. 41, pp. 344–344. ACM (2009)
7. Eguchi, A., Shen, J.: Student learning experience through cospace educational robotics: 3D simulation educational robotics tool. In: Cases on 3D Technology Application and Integration in Education, pp. 93–127. IGI Global (2013)
8. González-Martínez, J.A., Bote-Lorenzo, M.L., Gómez-Sánchez, E., Cano-Parra, R.: Cloud computing and education: A state-of-the-art survey. Computers & Education **80**, 132–151 (2015)
9. Hennig, M., Gaspers, D., Mertsching, B.: Interactive WebGL-based 3D visualizations for situated mathematics teaching. In: Information Technology Based Higher Education and Training (ITHET), 2013 International Conference on, pp. 1–6. IEEE (2013)
10. Koenig, N., Howard, A.: 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, pp. 2149–2154. IEEE (2004)

11. Michel, O.: Cyberbotics Ltd. Webots^{TM}: professional mobile robot simulation. International Journal of Advanced Robotic Systems **1**(1), 5 (2004)
12. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. In: ICRA workshop on open source software, pp. 1–6 (2009)
13. Ragan-Kelley, M., Perez, F., Granger, B., Kluyver, T., Ivanov, P., Frederic, J., Bussonier, M.: The Jupyter/IPython architecture: a unified view of computational research, from interactive exploration to communication and publication. In: AGU Fall Meeting Abstracts, H44D-07 (2014)
14. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., et al.: Scratch: programming for all. Communications of the ACM **52**(11), 60–67 (2009)
15. Shimizu, M., Koenig, N., Visser, A., Takahashi, T.: A realistic RoboCup rescue simulation based on Gazebo. In: L. Almeida, J. Ji, G. Steinbauer, S. Luke (eds.) RoboCup 2015: Robot World Cup XIX, pp. 331–338. Springer (2015)
16. Xu, L., Huang, D., Tsai, W.T.: Cloud-based virtual laboratory for network security education. IEEE Transactions on Education **57**(3), 145–150 (2014)
17. Xu, Y., Vatankhah, H.: Simspark: an open source robot simulator developed by the RoboCup community. In: S. Behnke, M.M. Veloso, A. Visser, X. R. (eds.) RoboCup 2013: Robot World Cup XVII, pp. 632–639. Springer (2013)