An Open-Source Simulator for Cognitive Robotics Research: The Prototype of the iCub Humanoid Robot Simulator

V. Tikhanoff, A. Cangelosi
University of Plymouth
Plymouth PL4 8AA
UK
vadim.tikhanoff@plymouth.ac.uk; acangelosi@plymouth.ac.uk

P. Fitzpatrick
Lira-Lab University of Genoa
Viale F.Causa, 13, Genova
Italy
paulfitz@liralab.it

G. Metta, L. Natale, F. Nori
Italian Institute of Technology
Via Morego 30, Genova
Italy
giorgio.metta@iit.it
lorenzo.natale@iit.it
francesco.nori@iit.it

***Abstract*— This paper presents the prototype of a new computer simulator for the humanoid robot iCub. The iCub is a new open-source humanoid robot developed as a result of the RobotCub project, a collaborative European project aiming at developing a new open-source cognitive robotics platform. The iCub simulator has been developed as part of a joint effort with the European project "ITALK" on the integration and transfer of action and language knowledge in cognitive robots. This is available open-source to all researchers interested in cognitive robotics experiments with the iCub humanoid platform.**

***Keywords*:** *Open-Source, Simulator, iCub humanoid robot, cognitive robotics*

I. INTRODUCTION

Computer simulations play an important role in robotics research. Despite the fact that the use of a simulation might not provide a full model of the complexity present in the real environment and might not assure a fully reliable transferability of the controller from the simulation environment to the real one, robotic simulations are of great interest for cognitive scientists [18]. There are several advantages of robotics simulations for researchers in cognitive sciences. The first is that simulating robots with realistic physical interactions permit to study the behavior of several types of embodied agents without facing the problem of building in advance, and maintaining, a complex hardware device. The computer simulator can be used as a tool for testing algorithms in order to quickly check for any major problems prior to use of the physical robot. Moreover, simulators also allow researchers to experiment with robots with varying morphological characteristics without the need to necessarily develop the corresponding features in hardware [1]. This advantage, in turn, permits the discovery of properties of the behaviour of an agent that emerges from the interaction between the robot's controller, its body and the environment. Another advantage is that robotic simulations make it possible to apply particular algorithms for creating robots' controllers, such as evolutionary or reinforcement learning algorithms [12]. The use of robotics simulation permits to drastically reduce the time of the experiments such as in evolutionary robotics. In addition, it makes it possible to explore research topics like the co-evolution of the morphology and the control system [1]. A simulator for the iCub robot magnifies the value a research group can extract from the physical robot, by making it more practical to share a single robot between several researchers. The fact that the simulator is free and open makes it a simple way for people interested in the robot to begin learning about its capabilities and design, with an easy "upgrade" path to the actual robot due to the protocol-level compatibility of the simulator and the physical robot. And for those without the means to purchase or build a humanoid robot, such small laboratories or hobbyists, the simulator at least opens a door to participation in this area of research.

The iCub simulator is currently being used by both the RobotCub and the ITALK project partners for preliminary experiments on the simulator robot, and subsequent testing with the physical robots.

II. ICUB SIMULATOR DEVELOPMENT

The iCub simulator has been designed to reproduce, as accurately as possible, the physics and the dynamics of the robot and its environment. The simulated iCub robot is composed of multiple rigid bodies connected via joint structures. It has been constructed collecting data directly from the robot design specifications in order to achieve an exact replication (e.g. height, mass, Degrees of Freedom) of the first iCub prototype developed at the Italian Institute of Technology in Genoa. The environment parameters on gravity, objects mass, friction and joints are based on known environment conditions.

A. Open-Source approach

The iCub simulator presented here has been created using open source libraries in order to make it possible to distribute the simulator freely to any researcher without requesting the purchase of restricted or expensive proprietary licenses.
The very first iCub simulator prototype was developed using the commercial Webots package [10,17], a professional robotic simulator which is widely used in academia and research. The Webots package is primarily designed for industrial simulations but used as a reliable tool for robotic research. Although a powerful software, the main disadvantages of the Webots package are its price, the computational heaviness of the package itself and the fact that, depending on the type of licence, there are limitations on the source code available in order to modify some properties of the actual simulator. Therefore the potential open source distribution of such a first prototype was quite limited.
Other open source simulators suitable for robotics research also exist. Amongst others we can find the Player/Gazebo project [6, 7], Simbad [16], Darwin2K [8], EvoRobot [4,12] and the OpenSim [14]. The Simbad, Darwin2K and Evorobot simulators have a strong focus on evolutionary algorithms and they have been mainly developed for scientific educational purposes. They are built to study AI algorithms and machine learning for multi-robot platforms. Gazebo is a powerful and complex multi-robot simulation in a 3D environment. OpenSim is a general multi-robot platform built in a similar way as the Gazebo package. However such systems use the same third party software and libraries.
Although the proposed iCub simulator is not the only open source robotics platform, it is one of the few that attempts to create a 3D dynamic robot environment capable of recreating complex worlds and fully based on non-proprietary open source libraries.

B. Physics engine

The iCub simulator uses ODE [13] (Open Dynamic Engine) for simulating rigid bodies and the collision detection algorithms to compute the physical interaction with objects. The same physics library was used for the Gazebo project and the Webots commercial package. ODE is a widely used physics engine in the open source community, whether for research, authoring tools, gaming etc. It consists of a high performance library for simulating rigid body dynamics using a simple C/C++ API. ODE was selected as the preferred open source library for the iCub simulator because of the availability of many advanced joint types, rigid bodies (with many parameters such as mass, friction, sensors...), terrains and meshes for complex object creation.

C. Rendering engine

Although ODE is a good and reliable physics engine, computing all the physical interaction of a complex system can take a good deal of processing power. Since ODE uses a simple rendering engine based on OpenGL, it has limitations for the rendering of complex environments comprising many objects and bodies. This can significantly affect the simulation speed of complex robotic simulation experiments. It was therefore decided to use OpenGL directly combined with SDL [15], an open source cross platform multimedia library. This makes it possible to render the scene with much more ease and to carry out computationally-efficient simulation experiments.

D. YARP protocol

As the aim was to create an exact replica of the physical iCub robot, the same software infrastructure and inter-process communication will have to be used as those used to control the physical robot. iCub uses YARP [5, 9] (Yet Another Robot Platform) as its software architecture. YARP is an open-source software tool for applications that are real-time, computation-intensive, and involve interfacing with diverse and changing hardware. The simulator and the actual robot have the same interface either when viewed via the device API or across network and are interchangeable from a user perspective. The simulator, like the real robot, can be controlled directly via sockets and a simple text-mode protocol; use of the YARP library is not a requirement. This can provide a starting point for integrating the simulator with existing controllers in esoteric languages or complicated environments.

E. Architecture

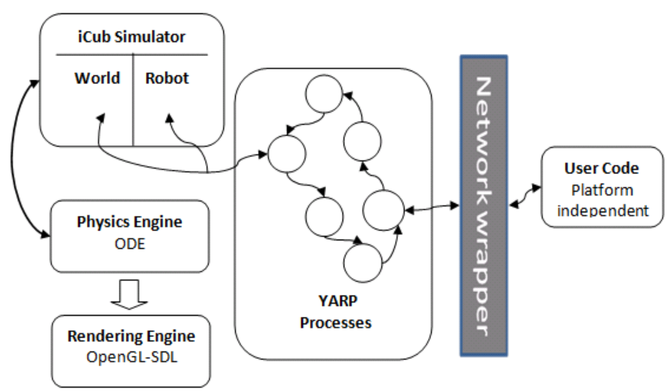

Fig 1: This figure shows the architecture of the simulator with YARP support. The User code can send and receive information to both the simulated robot itself (motors/sensors/cameras) and the world (manipulate the world). Network wrappers allow device remotization. The Network Wrapper exports the YARP interface so that it can be accessed remotely by another machine.

F. iCub body model

The iCub simulator has been created using the data from the physical robot in order to have an exact replica of it. As for the physical iCub, the total height is around 105cm, weighs approximately 20.3kg and has a total of 53 degrees of freedom (DoF). These include 12 controlled DoFs for the legs, 3 controlled DoFs for the torso, 32 for the arms and six for the head.

The robot body model consists of multiple rigid bodies attached through a number of different joints. All the sensors were implemented in the simulation on the actual body, such as touch sensors and force/torque sensors. As many factors impact on the torque values during manipulations, the simulator might not guarantee to be perfectly correct. However the simulated robot torque parameters and their verification in static or motion are a good basis and can be proven to be reliable [11].

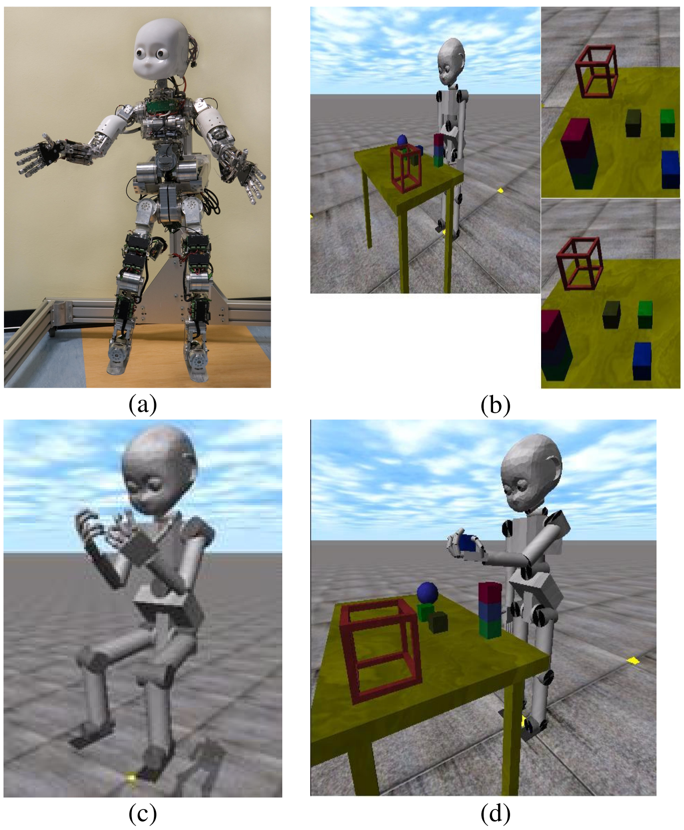

Fig 2: Photo of real iCub (a), of simulated iCub and the binocular view (b) The simulated iCub moving all four limbs as part of a demo (c) and the simulated iCub looking at and manipulating an object in its environment.(d)

All the commands sent to and from the robot are based on YARP instructions. For the vision we use cameras located at the eyes of the robot which in turn can be sent to any workstation using YARP in order to do develop vision analysis algorithms.

The system has full interaction with the world/environment. The objects within this world can be dynamically created, modified and queried by simple instruction resembling those that YARP uses in order to control the robot.

III. CONCLUSION

The current version of the iCub simulator has been used for preliminary testing by partners in the RobotCub and ITALK project. In addition to being used for experiments on the development of controllers for the iCub robot, some groups have used the simulator to create a mental model [2] used by the robot to represent the current state of the environment.

Future plans on the simulator development will mostly involve the design of functionalities to model and interact with the physical environment. For example, this will allow the users to modify the objects in the world where the iCub resides, in order to allow different types of experiments. Finally, further work will focus on the systematic testing and replication of simulation studies with the physical robot

SOFTWARE REPOSITORY

The latest version of the iCub simulation is available open source in the RobotCUb/iCub repository at http://www.robotcub.org/

ACKNOWLEDGEMENTS

This work was supported by the European Commission FP6 Project RobotCub (IST-004370) and FP7 Project ITALK (ICT-214668) within the Cognitive Systems and Robotics unit.

REFERENCES

[1] J.C. Bongard & R. Pfeifer (2003). Evolving complete agents using artificial ontogeny. In Hara, F. & R. Pfeifer, Eds., Morpho-functional Machines: The New Species (Designing Embodied Intelligence) Springer-Verlag, pp. 237-258

[2] PF Dominey (2007) Sharing Intentional Plans for Imitation and Cooperation: Integrating Clues from Child Developments and Neurophysiology into Robotics, Proceedings of the AISB 2007 Workshop on Imitation.

[3] European commission, unit e5, home page. http://cordis.europa.eu/ist/cognition/index.html, 2007.

[4] Evorobot http://laral.istc.cnr.it/evorobot/simulator.html

[5] P. Fitzpatrick, G. Metta, L. Natale: Towards Long-lived Robot Genes, Robotics and Autonomous Systems, 56(1):29-45, 2008

[6] B. Gerkey, R. T. Vaughan and A. Howard. The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. Proceedings of the 11th International Conference on Advanced Robotics, 2003.

[7] N. Koenig and A. Howard. Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japan, 2004.

[8] C. Leger, Darwin2K: An Evolutionary Approach to Automated Design for Robotics. Kluwer Academic Publishers, Aug 2000.

[9] G. Metta, P. Fitzpatrick & L. Natale. YARP: Yet Another Robot Platform. International Journal on Advanced Robotics Systems, 3(1):43–48, 2006.

[10] O. Michel, “Webots: Professional mobile robot simulation,” International Journal of Advanced Robotic Systems, vol. 1, no. 1, pp.39–42, 2004.

[11] N. Nava,V. Tikhanoff, G. Metta, G Sandini, Kinematic and Dynamic Simulations for The Design of RoboCub Upper-Body Structure ESDA 2008

[12] S. Nolfi & D. Floreano (2000). Evolutionary Robotics: The Biology, Intelligence and Technology of Self-Organizing Machines Cambridge, MA: MIT Press/Bradford Books
[13] Open Dynamics Engine http://opende.sourceforge.net/.
[14 OpenSim Simulator http://opensimulator.sourceforge.net/.
[15 SDL – Simple DirectMedia Layer http://www.libsdl.org/
[16] Simbad Simulator - http://simbad.sourceforge.net
[17] Webots - http://www.cyberbotics.com/
[18] Ziemke T. (2003). On the role of robot simulations in embodied cognitive science, *AISB Journal*, 1(4), 389-99