ROBBIT: AN OPEN SOURCE SIMULATOR FOR EDUCATION IN ROBOTICS

Gianluca Antonelli[1], Filippo Arrichiello[1], Chitresh Bhushan[2], Stefano Chiaverini[1], Sayandeep Purkayasth[2]
[1]Università degli Studi di Cassino, Italy, [2]Indian Institute of Technology Kharagpur, India

Corresponding author: Gianluca Antonelli,
Dipartimento di Automazione, Elettromagnetismo,
Ingegneria dell'Informazione e Matematica Industriale (DAEIMI),
Università degli Studi di Cassino,
via G. Di Biasio 43, 03043 Cassino (FR), Italy
Email: antonelli@unicas.it

Abstract. An open source software, named ROBBIT, aimed at the simulation of multiple mobile robots has been developed at the Università degli Studi di Cassino, Italy together with students coming from the Indian Institute of Technology Kharagpur, India. The main project's objective was to develop a software platform with educational purposes in robotics, computer sciences, non-linear control, behavioral and cognitive sciences. Several potential applications have been identified in order to use this software as a teaching tool for graduate, master thesis and PhD students. ROBBIT is released under the GNU General Public License (GPL).

1 Introduction

Teaching of control theory and robotics to computer science students can take great advantage in using practical demonstrations, both in laboratory or via numerical simulations [9, 18]. In recent years, both the decrease of the hardware costs and the diffusion of the *open source* philosophy made accessible to most of the universities the set-up of an educational laboratory for robotics applications [14]. Together with the teaching of basic mathematics, physics and systems dynamics, a fundamental role to fully appreciate a robotic class is played by the practical aspects of design, implementing and tuning a controller. A well designed software tool, moreover, may be of crucial importance also in allowing the students to better understand the importance of the modelling and of the identification steps of the design procedure. Software tools may also be used to teach basic feedback control theory to high-level students [12].

Several *similar* experiences have been proposed in the literature such as [20], based on the LEGO Mindstorm hardware. The work [11] proposes an experience based on the control of a DC motor with the help of the commercial Matlab [15] software, all the design and testing procedure is kept within the same software environment thus saving the students time. In [16] the control software for a mobile robot has been explicitly designed to run on PocketPC hardware thus to increase the transportability of the set-up. The work [21] presents the customization of a commercially available mobile robot to be used as test-bed both for education and research within a Matlab environment. An object-based graphical engine is proposed in [19], that provides a graphical user interface and C library to produce graphical simulation of robotic systems under X Windows.

In this paper a new project, named ROBBIT, is presented. ROBBIT is a open-source software which provide a 3D simulation environment for multiple robot system. In particular, ROBBIT enables the users to write their own controllers, modify the environment and, in future release, use sensors in order to test control algorithm and visualize their effects on different types of robots. It is not designed to provide a real world simulation and it is kept simple, modular and extensible for educational purpose. If needed, the user can easily add obstacles, sensors or define a new robot.

Several educational experiences are possible through the use of ROBBIT. The user can design and implement a customized control algorithm or acquire insights into the control design by, e.g., modifying the control gains of existing controllers. An unicycle-like kinematics has been implemented, car-like and Dubin kinematics will follow. Moreover, the software is written for the general case of multiple robots acting on the same environment, thus allowing also its use to more advanced concepts such as coordinated control. At an higher control level, by resorting to the kinematic control implemented [13], it is possible to simulate navigation or exploration algorithms for single or multiple mobile robots. Advanced control strategies, such as, behavioral or cognitive control [10], fuzzy logic, network control or genetic algorithms may be implemented as well.

The project's home page is provided in [5], while the SVN server in [4]. ROBBIT is released under the GNU General Public License (GPL). Figures 1 and2 provide two snapshots of the main window.

1.1 Available simulation engines and platforms

Several simulation engines and platforms are currently available, most of them, however, are commercial products and do not satisfy our first requirement (see Sect. 2); an incomplete list is given in Table 1. In the following, only

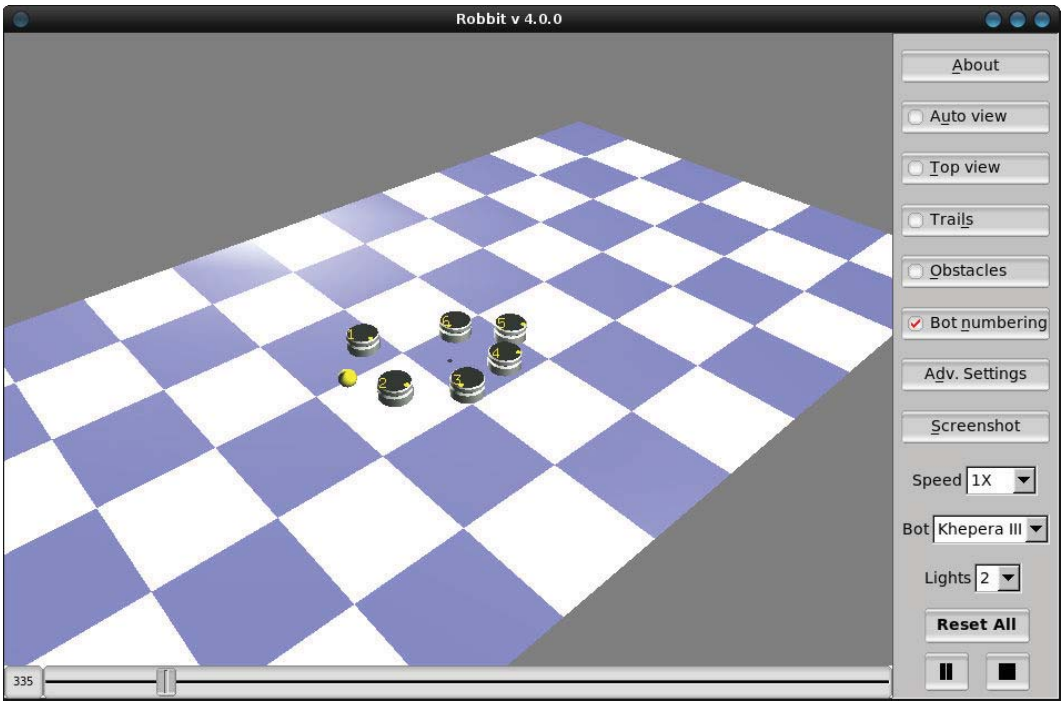

Figure 1: Snapshot of the main ROBBIT's windows simulating 6 mobile robots with a tennis ball as obstacle.

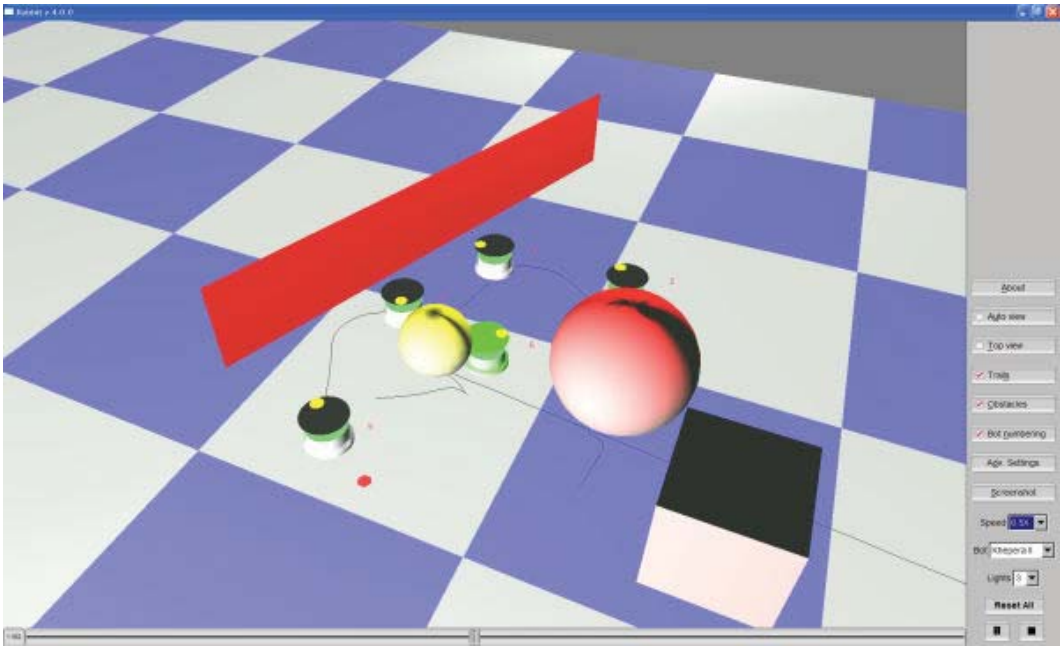

Figure 2: Snapshot of the main ROBBIT's windows simulating 5 mobile robots and several obstacles.

Name	type
breve [7]	Open Source & free
CLARAty	Open Source
Evolution Robotics ERSP	Commercial
iRobot AWARE	Commercial
Microsoft Robotics Studio [1]	Commercial/free
OpenJAUS	Open Source
OROCOS [6]	Open Source & free
Player, Stage, Gazebo [2]	Open Source & free
Simbad [17, 3]	Open Source & free
Skilligent	Commercial
URBI	Commercial
Webots	Commercial

Table 1: Some popular platforms and simulation engines.

the free software, or commercial one with free release for academics, will be briefly discussed.

Microsoft recently developed a robotics software platform named Microsoft Robotics Studio [1]. It provides both a simulation environment and a graphical service development toolkit. It implements behavior coordination by resorting to the concept of service arbiters. It is a commercial software available free for academic.

OROCOS [6] is the acronym for Open Robot Control Software project. The project's aim is to develop a general-purpose, free software, and modular framework for robot and machine control. It is not, thus a simulation engine but a platform. It has been designed with a special care for real-time applications and for Linux O.S. (Operative System) only. It does not have a graphical development neither a simulation environment.

Simbad [17, 3] allows the simulation of single or multiple robots in 3D and it has been developed with education purposes besides the scientific ones. As ROBBIT, It has to be interpreted as an open framework to test new ideas. It is written in Java (requires Java3d library) and runs on Mac OSX, Windows XP and some Linux distributions.

On the same philosophy of open source and free software is also Breve [7], written in Python. The user can develop her/his own controller also using a simple scripting language called *steve*.

The most interesting project, from the educational point of view, is given by the triad Player, Stage and Gazebo [2]. They are independent projects that are coordinated in order to be used jointly. Player is an open source TCP/IP-based hardware abstraction layer for several robotics hardware platforms. Stage and Gazebo implements accompanying simulation environments, they use ODE (Open Dynamic Engine - a rigid body dynamics simulator) to compute physical interaction with the environment. These projects are both Open Source and Free, they are probably among the most popular simulation engines in the research laboratories worldwide. They can be used on the following O.S.: Linux, Solaris, *BSD and Mac OSX but not Windows. They represent a powerful tool that needs some time to be learned.

2 ROBBIT overview

The choice to develop an open source software was natural after a long list of requirements:

- Free. The first requirement is probably the most serious: ROBBIT needs to be free (as free beer). One of the aims is that students can use ROBBIT using their home computers, moreover, low-budget academic groups should be able to use ROBBIT at no cost as well. It is well known that, for a private, the decision to use proprietary code versus open source code is given by the balance of the economical aspect versus the supposed larger time needed to work with open source SW. In this case, however, practicing with the code is one of the purposes of the project and thus the balance is polarized towards the open source choice;
- Portable. Open source code is usually more portable than closed one. There is no need to install hardware keys or install proprietary software in all the machines where the code needs to run. ROBBIT has been written in share mainly by the Authors of this article relying only to popular libraries;
- Platform independent. ROBBIT is available both for Windows and Linux. Most of the proprietary libraries are developed for Windows only;

O.S. version	Linux (kernel 2.6.24-19), Windows (XP)
C compiler	C, C++ (with Standard Template Library)
OpenGL	openGL Utility Toolkit v3.7.6 (Windows) / Freeglut 2.4.0 (Linux)
GUI toolkit	FLTK with FLUID v1.1.9
image library	libpng v1.2.31

Table 2: Main software requirements to run ROBBIT.

- Possibility to switch versus hardware-in-the-loop tests. For this project, the hard real-time issue has not been considered. Using (also) Linux, however, makes it easier to upgrade to a real-time Linux kernel and thus to the possibility to use ROBBIT for hardware-in-the-loop tests;
- Free (as free speech). Our educational role should be also to promote the wider diffusion of the expertise we learned trying, whenever possible, to avoid artificial threshold. Notice that the slogan was thought for the *Free Software* and some differences arise with respect to the Open Source philosophy.

Finally, it is necessary to take into account that several, independent, researches demonstrated that Open Source software and Operative Systems are more reliable and performing than the closed counterparts. Moreover, the open-source-community is generally more motivated.

Several other requirements, independent from the Open Source choice, have been imposed to ROBBIT such as the modularity and the simplicity of use.

3 How to use it

ROBBIT is released under the GNU General Public License (GPL) and it is available both for Windows and Linux platform. It can be downloaded at the webpage in [4]. Before the installation it is requested to install the list of free packages given in table 2.

When running, ROBBIT looks like a graphical interface that allows the user to select different options. At first, the user has to select the source of simulation data to visualize. The user can chose a sample log file from the archive or he can chose his self-developed log file. Then, ROBBIT reads the data from the chosen file and shows the motion simulation in the graphical window.

From the interface the user can choose the visualization parameters like camera position or light sources. Moreover he can chose to take some snapshots during the simulation run.

The user have to select the kind of robots used for the simulation. At the actual stage model for both Khepera II and Khepera III robots (manufactured by K-Team) are available. However, the user can define its own robot models.

The interface allows the user to visualize or not the obstacles in the environment, whose position is read from a text file. On the basis of the chosen robots and on their size, ROBBIT recognize the occurrence of collisions with obstacles (if selected) or among the robots. The collisions are evidenced by a temporary change of the robots' color.

The user can chose to visualize some variable useful to understand the behavior of their controller, e.g. paths of the robot, path of the robots' centroid, safety region for obstacle avoidance.

ROBBIT can be easily interfaced with the program containing the control law through text files that contain information about position and orientation of the robots.

4 A case study

As a case study on the use of ROBBIT, the research results presented in [8] are briefly summarized. In this paper a coordinate control strategy for multiple robot escorting an autonomous target is presented. After the theoretical development of the approach, the first simulations were run under the ROBBIT environment. ROBBIT allowed to debug the control code, test the validity of the control law and tune the control gains before doing the experiments. In particular, ROBBIT was used to simulate the motion of a team of khepera II mobile robots; at this purpose a realistic kinematic model of the khepera II robots was used to test the developed control law for non-holonomic vehicle with unicycle-like kinematics. Finally, the same code use for the simulation under ROBBIT was used to perform the experiments with a team of 7 khepera II. The use of ROBBIT gave a reasonable guarantee that the code was debugged and properly working; then, a fine tuning of the control gains was done on the basis of the real dynamics of the system. Figure 3 shows the software architecture.

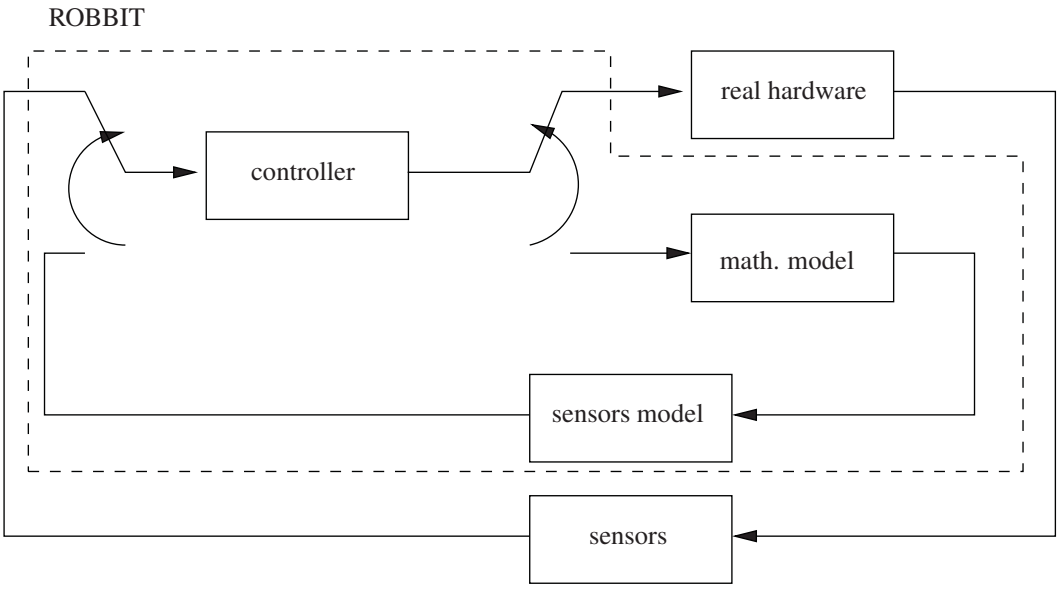

Figure 3: Sketch of ROBBIT as used in an experimental set-up.

Figure 4 reports a sketch of the Laboratorio di Automazione Industriale at the University of Cassino where ROBBIT is used as a tool for education and research and where the results presented in [8] have been run. The video of those experiences can be found at `http://webuser.unicas.it/lai/robotica/video`.

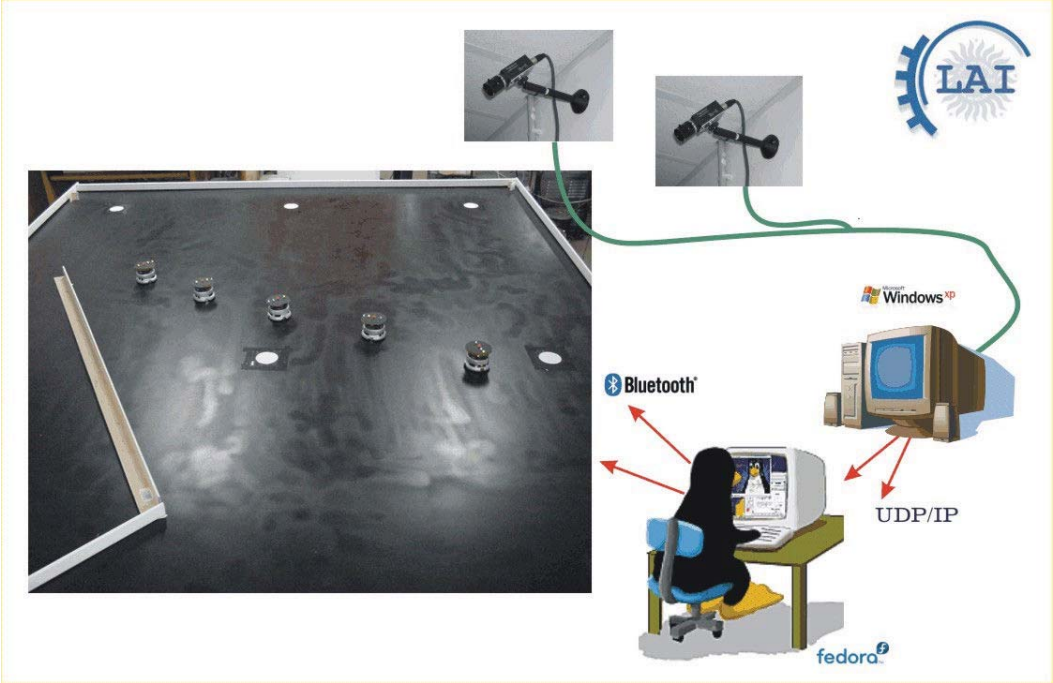

Figure 4: Sketch of the mobile robots experimental facility available at Laboratorio di Automazione Industriale at the University of Cassino where ROBBIT is used as a tool for education and research.

5 Future developments

ROBBIT is seen as an open project, where the user can develop her/his own modules depending on the necessity. Some scheduled developments concern:

- Decentralization of the control software for multiple robots. The control code currently run on a single machine and the decentralization of the controllers is obtained by software filtering of the available information. It is of interest to actually decentralize the controllers by running separate asynchronous threads.
- The current version of ROBBIT only allows to define simple objects to be used as obstacles during the

simulations. It is of interest to easily import and modify office or maze-like environments;

- A library of exteroceptive sensors needs to be developed such as sonar, laser and a simulated vision system;
- The communication among the robots is currently modelled as a simple time delay, better mathematical model may improve this aspect;
- Currently, only the kinematics of unicycle-like or point mobile robots is implemented, it is of interest both to extend the library to additional kinematics as well as to include the dynamics into the simulation engine;
- It is of interest to integrate the software with existing engines of numerical computation to make it easy to, e.g., plot the graphics of a simulation/experiment. Several free softwares might be of interest such as Octave and Scilab.

6 Conclusions

An open source software, named ROBBIT, aimed at the simulation of multiple mobile robots has been jointly developed by the Università degli Studi di Cassino, Italy and students coming from the Indian Institute of Technology Kharagpur, India. The main project's objective was the development of a software platform with educational purposes in robotics, computer sciences, non-linear control, behavioral and cognitive sciences. Several potential applications have been identified in order to use this software as a teaching tool for graduate, master thesis and PhD students. ROBBIT is released under the GNU General Public License (GPL).

7 References

[1] `http://msdn.microsoft.com/en-us/robotics/default.aspx`. Microsoft Robotics Studio.

[2] `http://playerstage.sourceforge.net/`. Player, Stage, Gazebo.

[3] `http://simbad.sourceforge.net/`. Simbad.

[4] `http://sourceforge.net/projects/robbit/`. ROBBIT.

[5] `http://webuser.unicas.it/robbit/`. ROBBIT.

[6] `http://www.orocos.org`. OROCOS.

[7] `http://www.spiderland.org/`. Breve.

[8] G. Antonelli, F. Arrichiello, and S. Chiaverini. An experimental study of the entrapment/escorting mission for a multi-robot system. *IEEE Robotics and Automation Magazine (RAM). Special Issues on Design, Control, and Applications of Real-World Multi-Robot Systems*, 15(1):22–29, March 2008.

[9] DS Bernstein. Control experiments and what I learned from them: a personaljourney. *Control Systems Magazine, IEEE*, 18(2):81–88, 1998.

[10] C. Buiu. Hybrid educational strategies for a laboratory course on cognitive robotics, accepted for publication. *IEEE Transactions on Education*, 2007.

[11] Y.C. Chen and JM Naughton. An undergraduate laboratory platform for control system design, simulation, and implementation. *Control Systems Magazine, IEEE*, 20(3):12–20, 2000.

[12] J. CORTES and W.B. DUNBAR. A high school-level course in feedback control: A matlab-based introduction requiring only algebra and trigonometry. *IEEE control systems*, 27(3):79–89, 2007.

[13] A. De Luca, G. Oriolo, and M. Vendittelli. Control of wheeled mobile robots: An experimental overview. *RAMSETTE: Articulated and Mobile Robots for SErvices and TEchnology*, 270:181–226, 2001.

[14] P. Fiorini. Encouraging robotics to take root [teaching tool]. *Robotics & Automation Magazine, IEEE*, 12(3), 2005.

[15] M.R. Guide. The MathWorks. *Inc., Natick, MA*, 1998.

[16] A. Howell, E. Way, R. McGrann, and R. Woods. Autonomous Robots as a Generic Teaching Tool. In *Frontiers in Education Conference, 36th Annual*, pages 17–21, 2006.

[17] L. Hugues and N. Bredeche. Simbad: an Autonomous Robot Simulation Package for Education and Research. *Proceedings of The Ninth International Conference on the Simulation of Adaptive Behavior (SABâĂŹ06), Roma, Italy*.

[18] M. Mansour and W. Schaufelberger. Software and laboratory experiments using computers in controleducation. *Control Systems Magazine, IEEE*, 9(3):19–24, 1989.

[19] DW Marhefka and DE Orin. XAnimate: an educational tool for robot graphical simulation. *Robotics & Automation Magazine, IEEE*, 3(2):6–14, 1996.

[20] VM Olivera, JMG Barahona, JC Gonzalez, and P. de las Heras Quiros. Libre software environment for robot programming. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 1, 2000.

[21] S. Piperidis, L. Doitsidis, C. Anastasopoulos, and NC Tsourveloudis. A low cost modular robot vehicle design for research and education. In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pages 1–6, 2007.