SimRobot – Development and Applications*

Tim Laue and Thomas Röfer

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH,
Sichere Kognitive Systeme, Enrique-Schmidt-Str. 5, 28359 Bremen, Germany
E-Mail: tim.laue@dfki.de, thomas.roefer@dfki.de

Abstract. This paper describes SimRobot, a robot simulator which is able to simulate arbitrary user-defined robots in three-dimensional space. It includes a physical model which is based on rigid body dynamics. To allow an extensive flexibility in building accurate models, a variety of different generic bodies, sensors, and actuators has been implemented. Furthermore, the simulator follows a user-oriented approach by including several mechanisms for visualization, direct actuator manipulation, and interaction with the simulated world. To allow a more detailed simulation, algorithms for simulating image disturbances as well as for actuator parameter optimization have been added. During the past years, SimRobot has been used to simulate several different robots, of which some are presented in this paper.

1 Introduction

The SimRobot project has already been started in the 1990s [1] as a general, kinematic, single-user, single-workstation robot simulation (which it still is). The main application was the simulation of autonomous wheelchairs [2]. Over the years, SimRobot's functionality and performance have been extended, e. g., by integrating standard components for graphics (OpenGL) and rigid body dynamics (ODE) [3]. Up to now, a variety of legged and wheeled robots – especially in the RoboCup domain – has been successfully simulated by SimRobot. Being one simulator among many others within the universe of RoboCup simulators, SimRobot of course has many similarities to other applications as well as some features not yet implemented by others. These differences will be pointed out within the following sections.

This paper is organized as follows: Section 2 describes the overall approach and particular concepts of SimRobot. Recent developments increasing the simulation's quality are presented in Sect. 3. In Sect. 4, some applications of SimRobot are shown. The paper concludes with a short description of ongoing and future works in Sect. 5.

* This work has been partly funded by the Deutsche Forschungsgemeinschaft (DFG) in context of the priority program RoboCup (SPP 1125) and the EU project SHARE-it (FP6-045088).

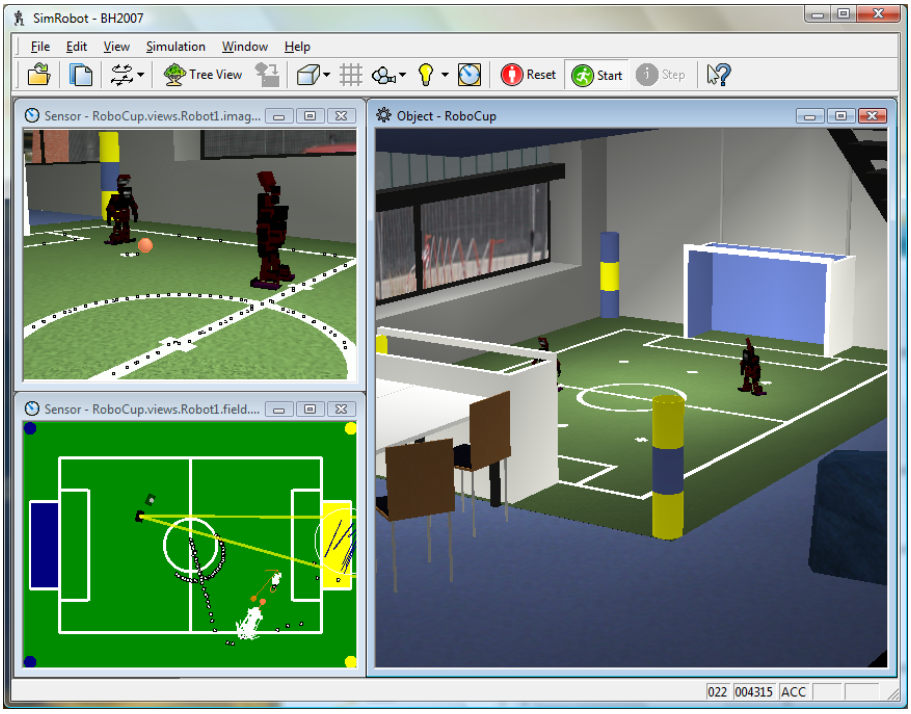

Fig. 1. SimRobot simulating the humanoid robots of team B-Human playing soccer in one of our labs, displaying the camera image and the world model of one of the robots.

2 Concept of SimRobot

As shown in Figure 2, SimRobot consists of several modules that are linked to a single application. This approach, which is different from many other client/server-based simulation concepts, has been chosen as it offers the possibility of halting or stepwise executing the whole simulation without any concurrencies. It also allows a more comprehensive debugging of the robot software executed. The main components of SimRobot are the *SimRobot core*, the *simulation scene*, the *user interface*, and the *controller*.

The SimRobot core is the most important part of the application. It models the robots and the environment, simulates sensor readings, and executes commands given by the controller or the user. Even most parts of the visualization are integrated into the core. The specification of the robots and the environment, i. e. the *simulation scene*, is modeled via an external XML file and loaded at runtime. Together with researchers from the Fraunhofer Institute for Autonomous Intelligent Systems, the specification language `RoSiML` (Robot Simulation Markup Language) [4] has been developed. It is a part of a joint effort to establish common interfaces for robot simulations. The aim is to exchange components between different simulators and to allow the migration of robot

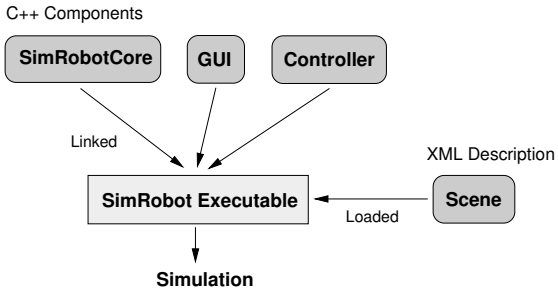

Fig. 2. The modules of SimRobot and their dependencies.

models among simulators without any complicated adaptations. The language by itself has been specified in XML Schema.

The user interface (cf. Fig. 1) of SimRobot has been designed to allow as much visualization and interaction as possible as well as to be flexible enough to handle simulations of different kinds of environments. Therefore a tree of all objects of the scene is the starting point for all user operations. Each node of that tree may be selected to open a view for that kind of object. In case of actuators (e. g. a hinge joint), a control for direct manipulation is opened. For sensors, several different visualization modes are implemented. Through this concept, it is also possible to open several views of arbitrary subsets of the scene graph. Furthermore, it is possible to interactively drag and drop and rotate objects inside the scene or to apply a momentum to an object (e. g. to let a ball roll). This is quite useful to arrange different settings while testing, e. g., a robot behavior. To add own views to a scene (e. g. the world model in Fig. 1), an interface for user-defined views has been implemented that allows the definition of drawings from inside the controller. Other elements of the user interface are a text editor for the scene description files and a console for text output from the controller.

The controller implements the sense-think-act cycle. In each simulation step, it is called by the simulation, reads the available sensors, plans the next action, and sets the actuators to the desired states. A controller which is suitable for the current scene loaded has to be provided by the user, i. e. it contains the control software of the simulated robots which usually is as much as possible identical to the software running on the real robots. Although the simulator itself is single-threaded, the programmer is free to implement a controller which allows a parallelization of the simulated software.

3 Recent Developments for a More Realistic Simulation

One characteristic trait of all simulations is that they can only approximate the real world, this inherent deficit is called *Reality Gap*. This affects all aspects of simulations: the level of detail as well as the characteristics of sensors and

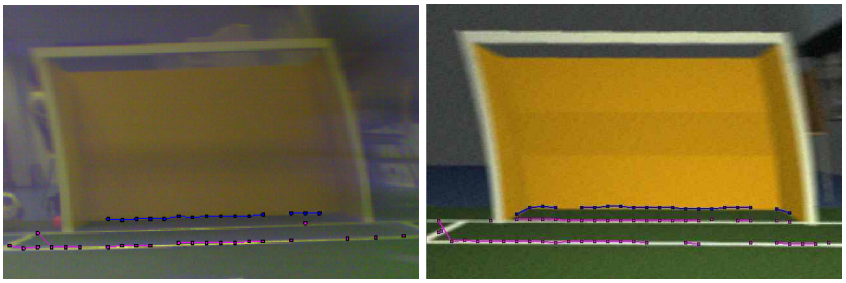

Fig. 3. Comparison of a real image (on the left) and an equivalent simulated image. The dots and lines represent percepts of the robot's vision system.

actuators. For many applications, the gap is of minor relevance as long as the simulated robot system performs in a reasonable way.

3.1 Camera Disturbances

Nevertheless, the characteristics of some sensors matter to an extent that leads to a significant mismatch between simulation and reality. Such a mismatch is relevant in two cases: the first and naïve case is that the robot control software is initially developed only in a simulator, and it does not handle sensor distortions that are not present in the simulation, but surely will be in reality. In the other case, software tuned for real sensor readings may perform worse or even fail if confronted with unrealistically simple sensor readings coming from a simulator. Hence, it is important that a simulation generates all the sensor distortions that are actively handled by the robot control software developed.

Figure 3 shows an example of a typical distortion, the image is taken by a CMOS camera that is part of the head of a humanoid robot. While the blurry impression is caused by motion blur, the distortion that bends and squeezes the yellow goal is produced by the so-called rolling shutter. Instead of taking complete images at a certain point in time, a rolling shutter takes an image pixel by pixel, row by row. Thus the last pixel of an image is taken significantly later than the first one. If the camera is moved, this results in image distortions. Compensating this effect becomes significantly important when working with robots which move their cameras fast or operate in a rapidly changing environment. In the RoboCup domain, this particularly affects robots with pan-tilt heads, e. g. in the Four-legged and the Humanoid league. To overcome problems which arise from these disturbances, different compensation methods have been developed, e. g. by Nicklin et al. [5] or by Röfer [6]. A necessity of simulating these disturbances can be derived from this explicit handling by the software of different teams.

Within the RoboCup community, a variety of robot simulators is currently used. Among the most advanced and established ones are, for example, Webots [7], Microsoft Robotics Studio [8], and the USARSim [9]. In general, they are

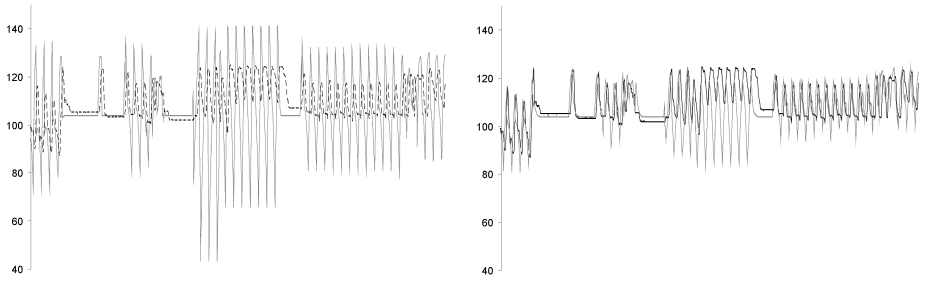

Fig. 4. Left: an extract from the controlled (solid) and the measured joint angle (dashed) of an AIBO's front right knee joint while playing soccer. Right: an extract from the real (dark) and the simulated joint movement (light) of the same joint after learning the (preliminary) maximum joint forces.

able to simulate camera images at a high level of detail, e. g., by simulating lights and shadows. But so far, none of them addresses the problem of image disturbances. In [10], an according Simrobot extension for simulating common image disturbances, i. e. the rolling shutter effect and motion blur, has been presented. By exploiting the features of modern graphics hardware, these disturbances can be simulated in real-time.

3.2 Optimizing Joint and Friction Parameters

Within the RoboCup domain, actuator performance is a crucial aspect. Through applying optimization algorithms, impressive results regarding robot velocities have been achieved during the last years, e.g. by Röfer [11] and Hebbel et al. [12] using the AIBO robot, or by Hemker et al. [13] using a humanoid robot. One common attribute of these algorithms is the strong exploitation of the environment's features, i.e. certain characteristics of the motors or the properties of the ground in this case. This leads to control trajectories that strongly differ from the resulting trajectories of the real robot joints, as shown on the left side of Figure 4. For robot simulations, especially when working with legged robots which have a high number of degrees of freedom, this requires a proper parametrization, i.e. to simulate actuators that behave close to real ones. Otherwise, the simulated robot might not only behave unrealistic but could fail completely.

All previously listed simulators [7–9] as well as the RoboCup Simulation League's fully dynamic robot simulation SimSpark, which additionally aims towards a closer cooperation with real robots [14], allow a detailed specification of the environment. But all of them demand the user to do this manually what might become an exhausting task given the high number of environmental parameters. Additionally, a once working parameter set is not guaranteed to be compatible with a different walking gait learned at a later point of time.

In [15], a general multi-staged process for which minimizes the reality gap between real and simulated robots regarding the behavior of actuators and their interaction with the environment has been presented. This optimization has

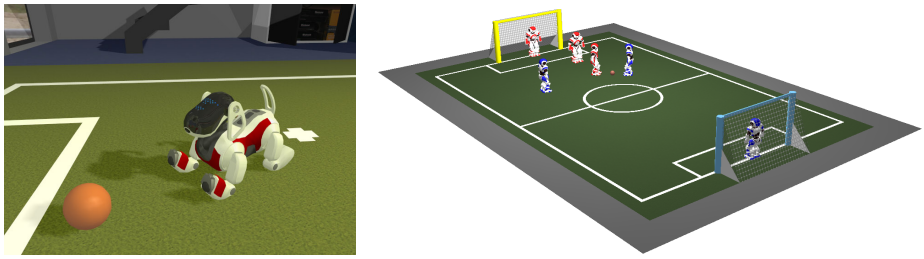

Fig. 5. Simulation of RoboCup Standard Platform League robots. Left: AIBO with all rendering features enabled. Right: 3 vs. 3 game in the Nao competition.

been carried out by using an evolutionary algorithm. This approach is general and transferable to different kinds of legged robots, its application is shown in SimRobot using a model of an AIBO robot as example. One intermediate result of the optimization process is depicted on the right side of Figure 4. The final result of this optimization has been measured to be more precise than the model delivered with the commercial Webots simulator.

4 Applications

The *GermanTeam* [16] in the RoboCup Four-Legged League (now called *Standard Platform League*) has used the different versions of SimRobot [2, 17, 3, 10] since its foundation in 2001. The left half of Figure 5 shows the latest version, in which the 20 DOF model of the AIBO provided by Sony was enriched by physics, a camera, three infrared sensors, and touch sensors in the feet. As discussed in the previous section, a rather realistic simulation of the behavior of the legs during walking was achieved. The team *B-Human* [18] in the RoboCup Humanoid League uses a robot model based on the Bioloid kit. It has also been modeled using SimRobot, again with 20 DOF (cf. Fig. 1) It is equipped with a camera in the head, and three accelerometers and three gyros in the body. The real world walking, kicking, and getting-up motions also work in the simulator. The friction coefficients of the ball are tuned to give rather realistic simulation of the kicking distances. The new robot used in the RoboCup Standard Platform League, the Nao from Aldebaran Robotics, was converted from its Webots model and integrated in SimRobot as part of the software framework used by the team *BreDoBrothers* [19] (cf. Fig. 5 right). The simulated robot has 21 DOF, two gyros, three accelerometers, and a camera in the head. However, SimRobot is not limited to simulate legged robots. It is also able to simulate wheeled systems. For instance, the robots of the team *B-Smart* in the RoboCup Small Size League [20] have been simulated, including the omni-wheels with their passive rolls (cf. Fig. 6 left). Also, the autonomous wheelchair *Rolland* has been simulated based on a 3-D model of a *Champ 1.594* provided by its manufacturer Meyra (cf. Fig. 6 right). For that model it was important that the passive castor wheels behave realistically, because they cause many control problems on the real wheelchair.

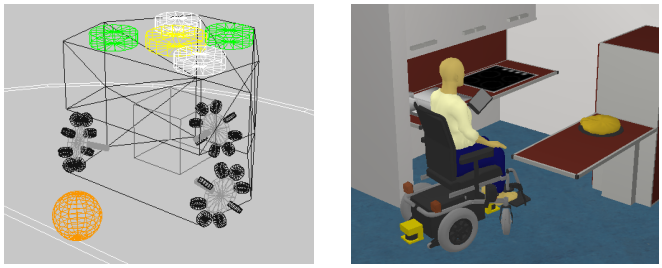

Fig. 6. Simulation of wheeled robots. Left: B-Smart robot with omni-wheels in wireframe view. Right: autonomous wheelchair Rolland, equipped with two laser scanners.

5 Ongoing and Future Works

Although being enhanced and used for a while, there does not exist any official software release of the current version of SimRobot. From its beginning, the simulator has been free software; on the official web page [21], some outdated versions are still available. In parallel to the publication of this paper, the most recent version of SimRobot becomes published as a part of the software release of the RoboCup team B-Human [22]. Nevertheless, this version still lacks a reasonable documentation which is intended to be provided in the future.

During the past years, SimRobot has been needed and developed further for the Microsoft Windows platform only. The aforementioned software release includes a new, partially incomplete, platform-independent version of SimRobot, which is based on the Qt toolkit.

References

1. Siems, U., Herwig, C., Röfer, T.: SimRobot, ein System zur Simulation sensorbestückter Agenten in einer dreidimensionalen Umwelt. Number 1/94 in ZKW Bericht. Zentrum für Kognitionswissenschaften. Universität Bremen (1994)
2. Röfer, T.: Strategies for Using a Simulation in the Development of the Bremen Autonomous Wheelchair. In Zobel, R., Moeller, D., eds.: Simulation-Past, Present and Future, Society for Computer Simulation International (1998) 460–464
3. Laue, T., Spiess, K., Röfer, T.: SimRobot - A General Physical Robot Simulator and Its Application in RoboCup. In Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y., eds.: RoboCup 2005: Robot Soccer World Cup IX. Number 4020 in Lecture Notes in Artificial Intelligence, Springer (2006) 173–183
4. Ghazi-Zahedi, K., Laue, T., Röfer, T., Schöll, P., Spiess, K., Twickel, A., Wischmann, S.: Rosiml - robot simulation markup language (2005) http://www.tzi.de/spprobocup/RoSiML.html.
5. Nicklin, S.P., Fisher, R.D., Middleton, R.H.: Rolling shutter image compensation. In Lakemeyer, G., Sklar, E., Sorrenti, D., Takahashi, T., eds.: RoboCup 2006: Robot Soccer World Cup X. Lecture Notes in Artificial Intelligence, Springer (2007)
6. Röfer, T.: Region-based segmentation with ambiguous color classes and 2-D motion compensation. In Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F., eds.: RoboCup 2007: Robot Soccer World Cup XI. Lecture Notes in Artificial Intelligence, Springer

7. Michel, O.: Cyberbotics Ltd. - WebotsTM: Professional Mobile Robot Simulation. International Journal of Advanced Robotic Systems **1**(1) (2004) 39–42
8. Jackson, J.: Microsoft robotics studio: A technical introduction. Robotics and Automation Magazine **14**(4) (2007) 82–87
9. Wang, J., Lewis, M., Gennari, J.: USAR: A game based simulation for teleoperation. In: Proceedings of the 47th Annual Meeting of the Human Factors and Ergonomics Society. (2003)
10. Pachur, D., Laue, T., Röfer, T.: Real-time Simulation of Motion-based Camera Disturbances. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII. Lecture Notes in Artificial Intelligence, Springer
11. Röfer, T.: Evolutionary Gait-Optimization Using a Fitness Function Based on Proprioception. In: RoboCup 2004: Robot Soccer World Cup VIII. Number 3276 in Lecture Notes in Artificial Intelligence, Springer (2005) 310–322
12. Hebbel, M., Nistico, W., Fisseler, D.: Learning in a high dimensional space: Fast omnidirectional quadrupedal locomotion. In: RoboCup 2006: Robot Soccer World Cup X. Lecture Notes in Artificial Intelligence, Springer (2006)
13. Hemker, T., Sakamoto, H., Stelzer, M., von Stryk, O.: Hardware-in-the-loop optimization of the walking speed of a humanoid robot. In: CLAWAR 2006: 9th International Conference on Climbing and Walking Robots, Brussels, Belgium (September 11-14 2006) 614–623
14. Mayer, N.M., Boedecker, J., da Silva Guerra, R., Obst, O., Asada, M.: 3D2Real: Simulation League Finals in Real Robots. In: RoboCup 2006: Robot Soccer World Cup X. Lecture Notes in Artificial Intelligence, Springer-Verlag (2007)
15. Laue, T., Hebbel, M.: Automatic Parameter Optimization for a Dynamic Robot Simulation. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII. Lecture Notes in Artificial Intelligence, Springer
16. Becker, D., Brose, J., Göhring, D., Jüngel, M., Risler, M., Röfer, T.: GermanTeam 2008 - The German National RoboCup Team. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII Preproceedings, RoboCup Federation (2008)
17. Röfer, T., Brunn, R., Dahm, I., Hebbel, M., Hoffmann, J., Jüngel, M., Laue, T., Lötzsch, M., Nistico, W., Spranger, M.: Germanteam 2004. In: RoboCup 2004: Robot World Cup VIII Preproceedings, RoboCup Federation (2004)
18. Röfer, T., Laue, T., Burchardt, A., Damrose, E., Müller, M.F.J., Rieskamp, A.: B-Human - Team Description for RoboCup 2008. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII Preproceedings, RoboCup Federation (2008)
19. Czarnetzki, S., Hebbel, M., Kerner, S., Laue, T., Nistico, W., Röfer, T.: BreDoBrothers - Team Description for RoboCup 2008. In Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C., eds.: RoboCup 2008: Robot Soccer World Cup XII Preproceedings, RoboCup Federation (2008)
20. Kurlbaum, J., Laue, T., Penquitt, F., Weirich, M.: Bremen Small Multi Agent Robot Team (B-Smart) - Team Description for RoboCup 2005. In Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y., eds.: RoboCup 2005: Robot Soccer World Cup IX Preproceedings, RoboCup Federation (2005)
21. Röfer, T.: SimRobot Website (2005) http://www.informatik.uni-bremen.de/simrobot/.
22. B-Human: Team Website (2008) http://www.b-human.de.