USARSim: a robot simulator for research and education

Stefano Carpin
School of Engineering
University of California, Merced
USA

Mike Lewis Jijun Wang
Department of Information Sciences
and Telecomunications
University of Pittsburgh – USA

Stephen Balakirsky Chris Scrapper
National Institute of
Standards and Technology
USA

***Abstract*— This paper presents USARSim, an open source high fidelity robot simulator that can be used both for research and education. USARSim offers many characteristics that differentiates it from most existing simulators. Most notably, it constitutes the simulation engine used to run the *Virtual Robots Competition* within the Robocup initiative. We describe its general architecture, describe examples of utilization, and provide a comprehensive overview for those interested in robot simulations for education, research and competitions.**

I. Introduction

In the past years there have been continuous efforts to foster *robotics* at the earliest stages of education. Figures depicting a constant decrease in the number of students entering engineering and science programs in the USA stemmed a wide range of initiatives to revert this alarming trend. Due to its intrinsic interdisciplinary and *hands on* nature, robotics seizes the students' interest as few other topics can. The success of initiatives like *Robocup*, an annual student-centric competition that attracts thousands of enthusiastic practitioners from all over the world, testifies that robotics plays a vital role in getting students involved. It is worth noting that the basin of attraction spans from children involved in the *Robocup junior* competition, to doctoral level students immersed in soccer and rescue competitions offering formidable scientific challenges. However, it is also evident that robotics education requires a significant amount of resources in terms of dedicated equipment, specialized supporting personnel, lab space and so on. In this paper we illustrate Urban Search and Rescue Simulation (USARSim), a high fidelity multi-robot simulator that proved to be an excellent research tool within the Robocup community, and that has the potential to become a first-class tool in robotic education. We expect USARSim to be an attractive possibility to complement activities with physical robots when resources are scarce or heavily constrained. USARSim offers a set of interesting characteristics currently not matched by any other simulator available:

1) it builds upon a widely used and affordable state of the art commercial game engine. Every improvement driven by the ever improving gaming industry translates into USARSim's advantages. This is particularly true for hyper realistic rendering and physical simulation.
2) the simulator itself is available for free under the GPL terms.
3) it is highly configurable and extendible. Users can easily add new sensors, or model new robots. This has already proven to be a viable development path, since some of the robot models currently bundled with the simulator were contributed to by end users modeling their own custom designed robots.
4) USARSim can be interfaced with *Player* [1], a popular middleware used to control many different robots. It follows that code developed within USARSim can be transparently moved to real platforms without any change (and viceversa).
5) USARSim seemlessly interfaces with the Mobility Open Architecture Simulation and Tools framework (*MOAST*) [2][3], a fully functioning hierarchical control system. MOAST privides a fully functional modular control system that users can immediately use to control robotic platforms. The user may then create more capable robots by adding commands to a module's vocabulary or may experiment with novel algorithms by rewriting any individual module.
6) quantitative evaluations show a close correspondence between results obtained within USARSim and with the corresponding real world system or sensor.
7) USARSim has been chosen as simulation infrastructure for a recently started competition held within the Robocup initiative. This creates a basin of highly skilled users that eventually release their developed code to the scientific community for its widespread dissemination.
8) although USARSim was originally developed aiming to Urban Search And Rescue simulation (hence the name), it is a general purpose multi-robot simulator that can be extended to model arbitrary application scenarios.
9) USARSim is platform independent and runs on Windows, Linux and MacOS.

The paper is structured as follows: in section II we illustrate current alternatives in the field of robot simulators. Then, in section III we briefly report on USARSim's technical design and in section IV we summarize validation results that illustrate encouraging findings. The Robocup Virtual Robots competition is then presented in section V and conclusions are presented in VI.

II. Related work

Almost every field in engineering makes extensive use of computer simulations. Robotics is no exception. However, the use of simulators in robotics has also been criticized at a philosophical level, because of the necessity to simulate not only robots, but *situated* robots, i.e. also the robot-environment interactions through sensors and actuators. We therefore restrict here our short survey on simulators offering this possibility. Stage [1] is the simulator backend to the Player middleware mentioned in the introduction. Stage simulates a two dimensional world where multiple robots can concurrently act. It is well suited for the study of cooperative tasks and transparently interfaces with its companion project Player, i.e. controllers written in Player can equally drive simulated or real robots. At the moment differential drive platforms are supported, as well as some sensors like odometry, sonars, and range scanners. Stage is an intrinsically two dimensional tool. Vision or other sensors that imply a third dimension are not offered. The whole Player/Stage project experiences significant popularity, mainly due to the great benefits that come from the use of Player which supports multiple robotics platforms. A project called *Gazebo* was started more recently and aims to provide a three dimensional simulation of outdoor environments. Physics within Gazebo is simulated using the popular Open Dynamics Engine [4]. Notably, all these components are available under the GPL license.

Webots [5] is a commercial simulator initially developed to aid software development for the Kephera robots. Meanwhile it offers support also for other platforms like the Sony Aibo, a humanoid platform, and several differential drive commercial robots. Custom designs can also be simulated using a dedicated authoring tool. Robots inside Webots can be controlled by writing a *webot controller*. Webots also uses ODE for physics simulation. In the authors' opinion, while Webots is the ideal choice when working with the Kephera and Koala robots (manufactured by the same company), its proprietary nature is the main limitation.

Delta3D [6] is a fully open source simulation engine whose concept is similar to the one used within USARSim. Currently, its main users seem to be the military, with many military applications developed and supported. Other applications and robotics scenarios are still scarce, or at least not well publicized. It nevertheless seems that Delta3D is capable of such simulations.

III. USARSim

A. General architecture

USARSim builds upon *Unreal Engine* 2.0, a commercially available game engine produced by Epic Games [7] to market first-person shooter games. Today's games achieve a level of complexity and realism so extreme that a single game would not generate enough revenue to financially counterbalance the development efforts. Game engines are general purpose simulation systems that can be used to implement multiple games based on the same foundation. They are therefore extremely open and customizable, and are thus excellent candidates to be used to develop robot simulators and other scientific investigations [8]. While the internal structure of the Unreal Engine is proprietary, there exists an interface called *Gamebots* [9][10] that allows an external application to exchange information in both directions with the engine. The Unreal Engine implements a so called *Unreal Virtual Machine*, a concept very similar to the Java Virtual Machine. Code written in the host language Unrealscript, a proprietary object oriented language with syntax resembling C++ and Javascript, is compiled into an intermediate platform-independent bytecode that can be executed by the Unreal Engine. Unrealscript code can retrieve information about objects in the environment, move parts around and so on. Figure 1 depicts USARSim's architecture.

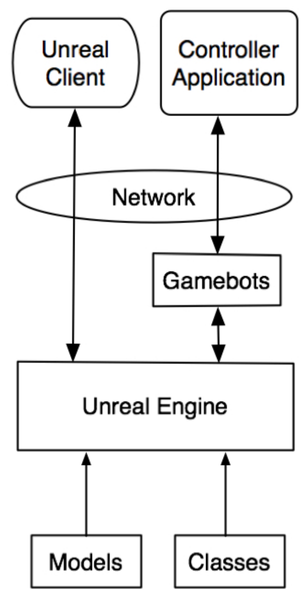

Fig. 1. USARSim's architecture

Upon startup, the Unreal Engine loads a set of geometrical models describing all the objects in the environment. These models can be easily created with the *UnrealEd* application, a geometrical modeling program shipped in bundle with the UnrealTournament game. UnrealEd can import and export models compatible with most commercially available modeling software packages. For each object one can specify shape, color, textures, and many other properties. The Unreal Engine also loads a set of classes of compiled scripts that govern the behavior of loaded models. USARSim in itself is a set of models and classes defining the simulation of robots, sensors and actuators. After the startup phase is completed, the system is ready to accept connections from client applications. Users can connect with the standard Unreal Tournament client. This spawns a disembodied agent that can move in the environment and observe how the simulation is evolving. Figure 2 exemplifies this view modality.

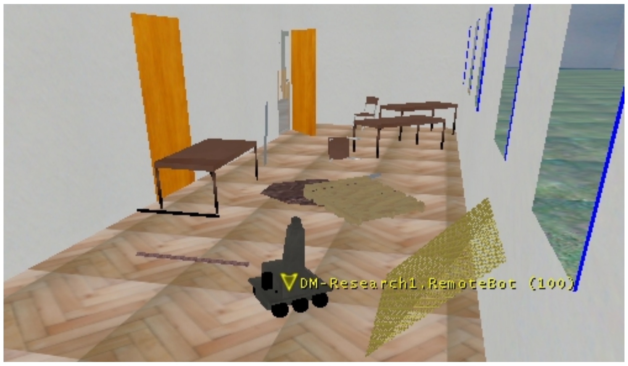

Fig. 2. In *spectator* mode a disembodied observer can move within the simulator without interacting with the robots, thus allowing a close observation of the performance from different points of view.

In addition, users can start their own controller application to control robots inside the simulated environment. Any language capable of reading and writing a TCP socket can be used. The format used to exchange information is a simple text based protocol documented in [11]. Users can directly use this information format, or use available Player, Pyro or MOAST compliant wrappers.

B. Sensors and actuators

USARSim currently features the simulation of multiple sensors and actuators. Sensors are implemented as classes coded in Unrealscript polling the internal status of the simulator upon request. USARSim internally implements a hierarchy of classes devoted to sensor simulation as depicted in figure 3. Users who need to introduce a new sensor therefore need not start from scratch, but rather have to create a suitable subclass from the existing ones.

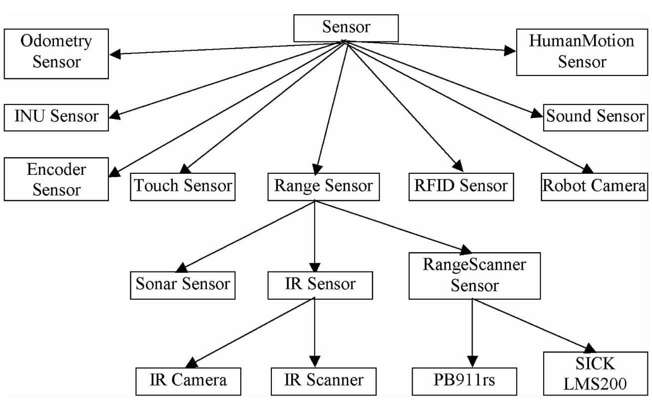

Fig. 3. Sensors hierarchy inside USARSim

Classes implementing sensors can model different noise sources and correspondingly alter returned values. Noise intensity and characteristics can be specified in a suitable configuration file. Users can therefore test their algorithms in noise free environments, or measure their robustness against increasing noise. This feature is extremely useful while designing new algorithms. As can be seen from figure 3, classes implementing the most commonly used sensors in mobile robotics are available. Adding a new sensor to USARSim is straightforward. Within Unrealscript one has access to all the physical entities present in the simulated environment, and a wide set of functions to extract information from them are available. After groundtruth data are extracted, adding noise and presenting them in a shape compatible with commercial sensor interfaces is a simple process. Due to its very different nature, vision sensors are implemented in a different way. An external application called *ImageServer* sets the point of view of a running Unreal Client so that it coincides with the exact pose of a robot's camera sensor. When the image has been rendered, it captures it, encodes in JPEG format and makes it available to the controller application. ImageServer can be set to deliver images with different resolutions and quality, and from different robots. Figure 4 shows an image obtained from a simulated camera mounted on a mobile robot.

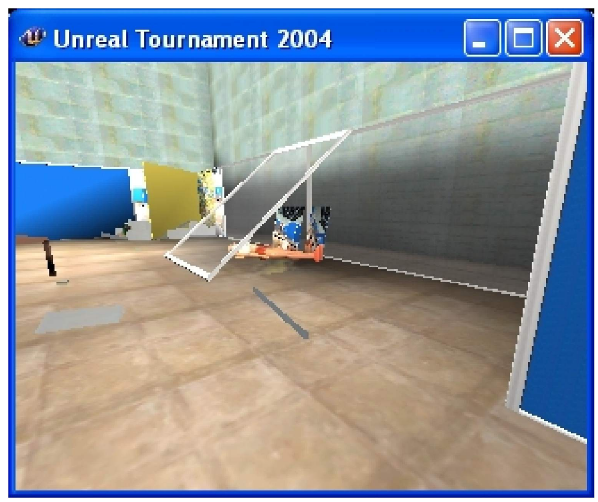

Fig. 4. An image obtained from a simulated camera mounted on a mobile robot exloring a maze-like environment.

All sensors are configurable, i.e. the end user can specify in a configuration file which sensors, how many, where, and the amount of noise that should be used for every robot. Within USARSim, actuators are divided into two categories: actuators dedicated to mobility like wheels and similar, and other actuators. We here describe the latter category, while mobility will be discussed in the next subsection. The *mission package* is a collection of controllable links, where each link consists of a joint and a static mesh of a robotic component. Links are represented through the use of *Denavit and Hartenberg* (DH) parameter notation [12]. Control of the links is accomplished through adjusting the value of the free varialbe of the link. These commands may take the form of a joint position, joint velocity, or joint torque. This representation allows for the consistent description of mission packages ranging from robotic arms and manipulators to pan/tilt systems.

C. Robots

USARSim was initially developed with a focus on wheeled robots, in particular differential drive systems. Due to increased interest and wide community support, the spectrum of available platforms significantly grew, and the currently available version offers multiple robots, including underwater vehicles, legged platforms, and humanoids. Like sensors and actuators, robots are implemented by specific classes, thus making it easier to develop new platforms that model custom designs. Three base classes model different kinds of wheeled locomotion, namely differential drives, omnidirectional vehicles and Ackerman steered vehicles. Figures 5 and 6 show some simulated robots inside USARSim.

Fig. 5. On the left: a simulated ATRVJr. On the right: a P2AT robot. Both robots mount a Sick laser in front, a PTZ camera and some sonars.

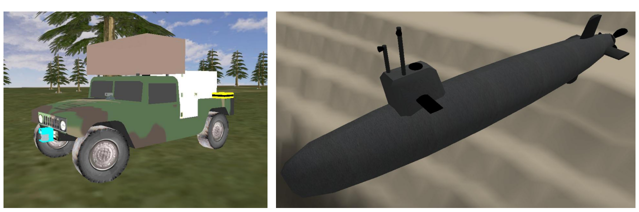

Fig. 6. A simulated HUMMV vehicle based on the Ackerman's steering, and a simulated submarine based on propeller speed, rudder, and stern plane position.

In addition to wheeled robots, some research groups have recently pushed USARSim forward to develop control algorithms for underwater vehicles, legged robots, and humanoid robots [13]. Figure 6 shows a submarine, while Figure 7 shows a simulated Sony AIBO and a humanoid robot.

For underwater robots, control is based on providing the propeller speed, rudder position, and stern plane possition. While the hydro-dymamics are greatly simplified, interesting exploration and mapping algorithms have been explored with these robots.

For legged robots, the authors acknowledge that the simulator accuracy currently does not allow to tune algorithms for dynamical gait balancing and similar tasks. However, the simulator has been successfully used to perfect the mechanism triggering behaviors based on visual input. Results obtained in simulation were then moved to physical robots.

Tracked vehicles have been implemented as well, although their simulation is at the moment not satisfactory from

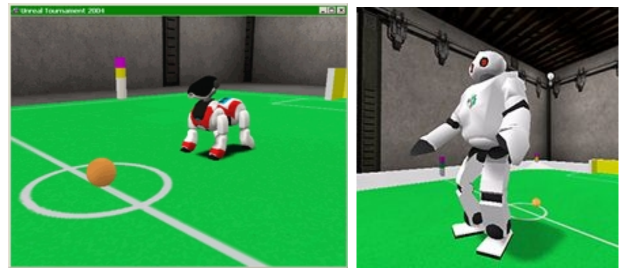

Fig. 7. On the left a simulated AIBO robot. On the right a humanoid

the point of view of track-soil interaction. The required high degree of accuracy turns out to be too demanding for Karma, the current physics engine currently supporting the UnrealEngine. However, the forthcoming advent of extension boards for hardware accelerated physics is expected to overcome this problem.

D. Environments

As previously outlined, thanks to the UnrealEd applications, new environments can be easily developed. USARSim initially aimed at providing a high resolution reproduction of the NIST reference arenas. The reference arenas are a set of three test environments created to assess the effectiveness of robotic platforms designed for urban search and rescue missions [14]. These arenas, whose models are part of the core USARSim distribution, feature indoor environments with different degrees of difficulty. Some are maze like planar environments, while others offer three dimensional settings with slopes, stairs, rubble piles and more. Those interested in developing and tuning general purpose algorithms for tasks like navigation, mapping, localization, exploration and the like will find a wide range of different challenges. More recently, concurrently with the last Robocup competition, larger settings on the scale of a city block with both indoor and outdoor features were developed and released to the public domain.

IV. Validation

A central tenet of modern behavior-based robotics is that effective systems can best be designed by eliminating internal representations and focusing instead on the direct relation between stimulus and action. From this perspective a good simulation must simultaneously supply an accurate model of the robots geometry and kinematics, accurate models of sensors, an accurate model of the environment, and an accurate model of the robots interaction with that environment. If any one of these constituents breaks down the simulation can no longer provide an adequate model of the process being studied, and the results may not be generalizable to real robots. Simulation requirements were far more relaxed for an earlier generation of robots that relied on planning and many robot simulators still provide only schematic or 2D models of the environment and pay little attention to the physics of the interaction between robot and environment. USARSim, by contrast, provides detailed models of both the environment and the physics of interaction, making possible to learn about behavior-based robotics from simulation. This approach to robotics education, however, depends crucially on validation of the models used so that students and instructors have reasonable assurance that the problems they encounter and solutions they devise are representative of actual problems and solutions in robotics rather than simply artifacts of the simulation.

The level of effort devoted to validation has been a distinguishing feature of USARSim. Each of its major constituents: robot kinematics, interaction with the environment, sensors, and camera video, have been subjected to ongoing validation testing. Tests of a Hokuyo PB9-11 laser range finder and its simulation reported in [15], [16] compared the detection of walls and corresponding Hough transforms for real and simulated versions of NISTs Orange Arena.

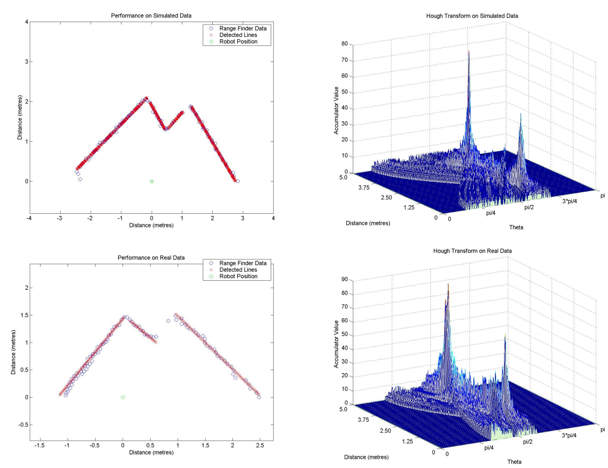

Fig. 8. Data from simulation at top, detected walls on left, Hough transform on right. Data from real systems are displayed in the second row.

As shown in figure 8 there is very close agreement between the real and simulated sensor. Robot performance and human-robot interaction (HRI) tests reported in [16], [17], [18] show equally close agreement for performance times, accuracy of control, and effects of surface roughness and task complexity. In these validation experiments for the Activemedia P2/3-AT and experimental PER participants repeatedly drove either a real or simulated robot using a simple interface showing camera video and a joystick used for teleoperation (direct) or waypoint selection (command).

Figure 9 shows average task times in seconds with pairs of bars contrasting the performance of real and simulated robots under each condition. Performance agreed closely under all but one condition.

Our most recent experiments [18] have compared information extraction from real and simulated camera video using four popular image processing algorithms. Figure 10 shows correlations between edges extracted from synthetic camera imagery and real camera images from well lit and poor lit instances of the same scenes.

For each quartet of bars the first shows the autocorrelation

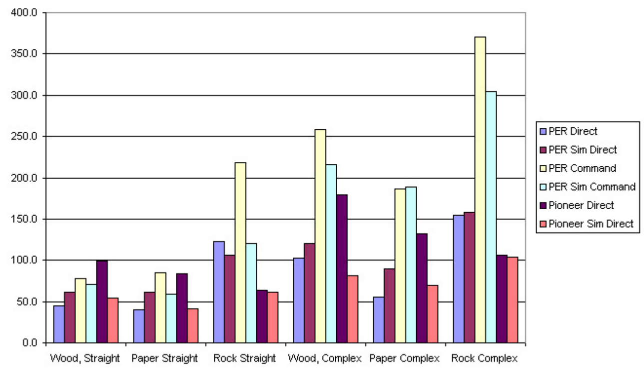

Fig. 9. Task times for real and simulated robots

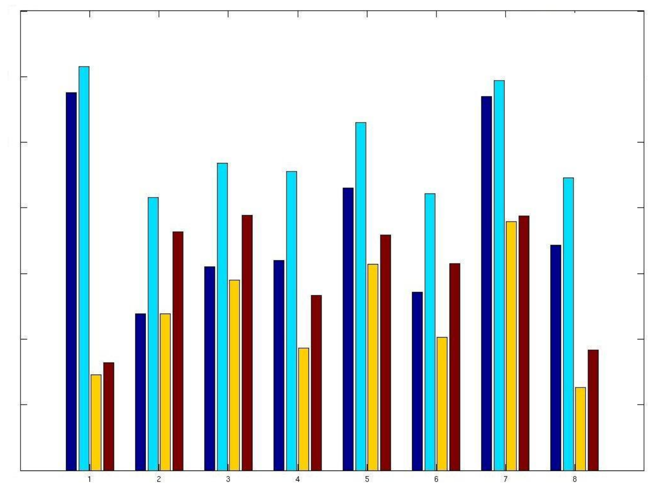

Fig. 10. Correlations between detected edges for synthetic and real camera images

for the synthetic image, the second correlation between synthetic and well lit camera, the third synthetic and poorly lit camera and the fourth, correlation between the cameras. Across these scenes the highest agreement was found between edges extracted from simulated and well lit imagery demonstrating that USARSims simulated camera video is not only suitable for HRI [16], [15], [18] but machine vision [18] as well.

V. The Virtual Robots Competition

The Robocup initiative, initially born with the goal of developing a team of robots playing soccer able to prevail in the world champions by 2050, has recently been extended to other domains. Among them, the Robocup Rescue league has gained significant attention. This competition aims at the development of robotics artifacts that could effectively help human rescuers in the aftermath of disasters like earthquakes, terroristic attacks, and similar extreme situations. Roughly speaking, the competition goal is to explore, map, and produce as much information as possible about an unknown and deeply unstructured environment. The league initially developed two competitions: a Real Robots Competition and a Simulation Competition. In the Real Robots Competition, robots currently operate in a indoor environment that represents a single rubble pile. Current crisp research topics are advanced mobility, system integration, and sensor fusion. The Simulation competition initially started with an Agent competition where rescue actors with high level capabilities operate on a city scale scenario. Research themes include distributed decision making, real time learning, and more. In the long term these two communities should eventually meet and merge their research efforts that are currently at the very opposite ends of the spectrum. In order to make this process easier [15][17], a new competition named *Virtual Robots Competition* was started abreast the Agents competition [19]. After a demo stage during Robocup 2005, a the regular tournament took place during Robocup 2006, with the participation of eight teams from four continents. This competition provides a meeting point between the different research communities involved in the other rescue competitions by providing scenarios on the scale of a city block (see figure 11) that emphsise realistic mobility challenges such as uneven terrain, maze-like areas, stairs, etc. and multi-agent cooperation challenges such as large exploration areas, joint mapping, resource utilization, etc.

Another goal of the competition is to lower the entry barries for newcomers. The possiblity of testing and developing control systems using platforms and modules developed by others makes the start-up phase easier. With this goal in mind, the open source strategy already embraced in the other competitions is fully supported. The code that was used on this year's winning platforms has already been posted on the web.

Fig. 11. A screenshot taken from outside area of the competition world used during Robocup 2006. The inside of the burning building on the left was completely modeled as well.

VI. Conclusions

We have provided an overview of USARSim, a high fidelity robot simulator that is enjoying significant popularity within and beyond the Robocup community. We envision that USARSim can be a very effective tool for teaching as well, because of its open source approach and the modest cost of the supporting game engine. The choice of founding USARSim upon a state of the art game engine places the simulator at the frontier of high accuracy rendering and physical simulation. Moreover, USARSim inherits its intrinsic extendibility. The adoption of USARSim as simulation system supporting the Virtual Robots competition creates a community of skilled users releasing to the public code for different robotic tasks, thus lowering the entry barriers for new comers. The simulator can be freely downloaded from [20].

Acknowledgments

The authors thank the many research groups and individuals that have developed and contributed extension to the initial USARSim core.

References

[1] "Player/stage project," http://playerstage.sourceforge.net, 2005.
[2] C. Scrapper, S. Balakirsky, E. Messina, "MOAST and USARSim - A Combined Framework for the Development and Testing of Autonomous Systems," in *Proceedings of the 2006 SPIE Defense and Security Symposium*, 2006.
[3] "MOAST project, " http://moast.sourceforge.net, 2005.
[4] R. Smith, "Open dynamics environment," http://www.ode.org, 2006.
[5] O. Michel, "Webots^{TM}: Professional mobile robot simulation," *International Journal of Advanced Robotics Systems*, vol. 1, no. 1, 2004.
[6] Delta3D, http://www.delta3d.org/, 2006.
[7] Epic games, "Unreal engine," www.epicgames.com, 2005.
[8] M. Lewis and J. Jacobson, "Game engines in scientific research," *Communications of the ACM*, vol. 45, no. 1, pp. 27–31, 2002.
[9] A. Adobbati, A. Marshall, A. Scholer, S. Tejada, G. Kaminka, S. Scaffer, and C. Sollitto, "Gamebots: a 3d virtual world test-bed for multi-agent research," in *Proceedings of the second international workshop on Infrastructure for agents, MAS, ans scalable MAS*, 2001.
[10] Gamebots, http://gamebots.sourceforge.net/.
[11] J. Wang, "USARSim v2.0.2 - a game based simulation of the nist reference arenas," http://sourceforge.net/projects/USARSim.
[12] J.J. Craig, "Introduction to Robotics: Mechanics and Control," Addision-Wesley Pub., Reading, Ma, USA, 3rd edition, 2005.
[13] M. Zaratti, M. Fratarcangeli, L. Iocchi. "A 3D simulator for multiple legged robots based on USARSim", in *Robocup 2006: Robot Soccer World Cup X*, Springer LNCS, (to appear).
[14] A. Jacoff, E. Messina, J. Evans. "Experiences in deploying test arenas for autonomous mobile robots", in *Proceedings of Performance Metrics for Intelligent Systens*, 2001.
[15] S. Carpin, J. Wang, M. Lewis, A. Birk, A. Jacoff. "High fidelity tools for rescue robotics: results and perspectives", in *Robocup 2005: Robot Soccer World Cup IX*, Springer, LNAI Vol. 4020, Springer, 2006, pp. 301-311.
[16] J. Wang, M. Lewis, S. Hughes, M. Koes, and S. Carpin. Validating usarsim for use in hri research, in *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting*(HFES05), pp. 457-461, 2005.
[17] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, C. Scrapper. "Bridging the gap between simulation and reality in urban search and rescue", in *Robocup 2006: Robot Soccer World Cup X*, Springer LNCS, (to appear).
[18] S. Carpin, T. Stoyanov, Y. Nevatia, M. Lewis, J. Wang. "Quantitative assessments of USARSim accuracy," in *Proceedings of PerMIS 2006*, 2006.
[19] The Virtual Robots Competition. www.robocuprescue.org
[20] USARSim, http://sourceforge.net/projects/usarsim