Graphical Simulation and High-Level Control of Humanoid Robots

James J. Kuffner, Jr. **Satoshi Kagami** **Masayuki Inaba** **Hirochika Inoue**

Department of Mechano-Informatics
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{kuffner,kagami,inaba,inoue}@jsk.t.u-tokyo.ac.jp

Abstract

Physically-based simulation software is commonly used for developing and testing low-level robot control algorithms. In order to facilitate the development and evaluation of higher-level robot behaviors, broader-based simulations are needed. Examples include software for simulating 3D vision, motion planning for obstacle avoidance, and integrating vision and planning. In addition to modeling the general interaction between the robot and its environment, the software can be used as a graphical user interface for directly controlling or interacting with a robot operating in the real world. This paper describes our current efforts toward building a large-scale software simulation framework for the development and testing of high-level behaviors for humanoid robots. We view this as a potential useful tool for the visualization and development of robotic systems, as well as an interactive, task-level programming interface for robots.

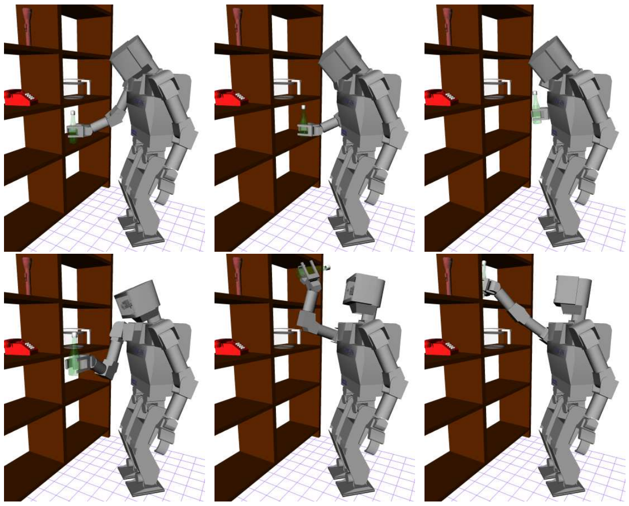

Figure 1: Manipulation planning for the 'H6' robot.

1 Introduction

Much research effort in robotics has been focused on designing high-level control architectures for autonomous robots. The primary aim of our simulation environment is to assist in the development and testing of such high-level control software. For humanoid robots, we are interested in developing software to solve complex tasks, such as navigating in an unknown environment while avoiding obstacles, and grasping and manipulating objects.

Advances in computing hardware have enabled increasingly sophisticated software simulations to be used in the design and testing of robotic systems. In particular, complex robots such as humanoid robots have benefited from dynamic simulation software. Such software has assisted in the realization of dynamic walking in several humanoid robots[5, 18, 14].

In comparison, much less attention has been devoted to simulating higher-level behaviors for humanoid robots. Successful simulation environments have already been developed for teleoperated space robots [1, 6, 15]. Similar kinds of software capabilities are needed for humanoid robotic systems, and some sophisticated systems are currently being developed[3]. Software libraries for 3D vision, motion planning for obstacle avoidance, and integrating vision and planning are needed. To facilitate the deployment of such software, we are currently developing a graphical simulation environment for the humanoid robots in our laboratory (Figure 2). The project builds upon a software framework that was originally developed for the high-level control of computer animated characters in 3D virtual environments[12]. Although its present capabilities are limited, it is our hope that through the

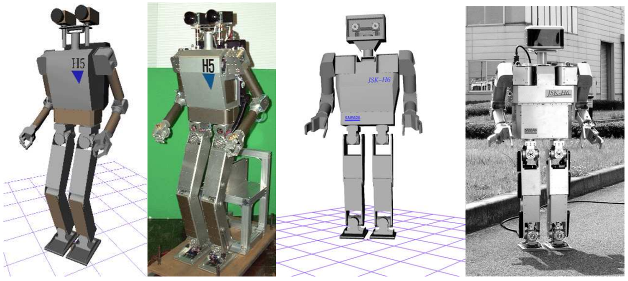

Figure 2: Virtual and real humanoids 'H5' and 'H6'

use of such interactive simulation software, the current and future capabilities of humanoid and other complex robotic systems can be improved.

The paper is organized as follows: Section 2 gives an overview of the simulation environment and its larger goals, along with a brief description of three components: simulated vision (Section 2.1), motion planning (Section 2.2), and balance compensation (Section 2.3). Section 3 shows some preliminary experimental results and Section 4 concludes with a summary discussion.

2 Simulation Environment

For simulation purposes, we are designing a system architecture similar to the *TeleSensorProgramming* architecture proposed in [1]. We begin by constructing a *virtual robot* in a virtual environment. The virtual robot comes equipped with *virtual sensors* and *virtual control inputs*. Software being developed for controlling a real robot can first be debugged, tested, and evaluated by connecting its inputs and outputs to the virtual robot.

The goals of the simulation environment are:

1. Visualize the robot model and movement in a simulated 3D environment.

2. Provide a testbed for the development and evaluation of robot control and behavior software.

3. Serve as an interactive, task-level user interface for controlling a real robot.

4. Provide a means for a robot to reason about the physical environment.

By "reasoning" about the environment, we mean that the robot has the ability to *compute the potential consequences of a set of possible actions*. This will

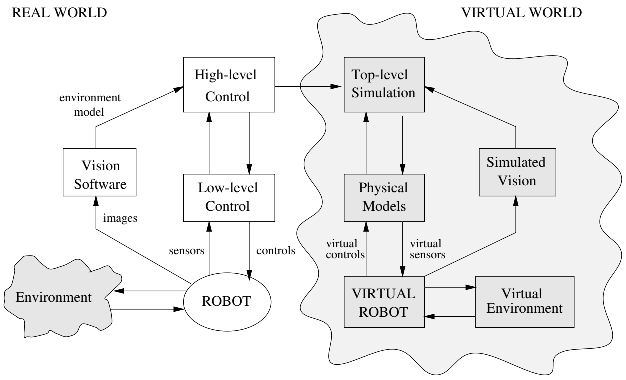

Figure 3: Simulation architecture overview.

hopefully allow the robot to make better informed decisions when attempting to solve complex tasks.

Clearly, a virtual robot has many advantages over a real robot. In particular, if one so desires, one can equip a virtual robot with "perfect" sensing and "perfect" control. However, this is typically not a realistic scenario. One of the primary advantages to using a flexible simulation environment is that one has the freedom to construct detailed models of sensor noise, or errors in control. By testing control strategies under different conditions, one can potentially distinguish between failure of the algorithm, or failure due to sensor noise or control errors.

The key point is that one can build software models that are better than current robot technology. Thus, while the designs and technology used in real robots will improve in the future, the software to control them can potentially be developed at the present time.

The simulation environment separates high-level and low-level control, and provides a model of interaction between the robot and the environment (Figure 3). If developed properly, the simulation could provide the robot with the means to anticipate the consequences of a set of possible actions. For example, suppose a humanoid robot is attempting to pick up a nearby object. If the robot has general knowledge of the physical properties of the object and its current situation in the environment, simulation can be used to help decide the best way to accomplish the given task. The criteria used might consist of the best way to grasp, lift, and transport the object given the physical limitations of the robot and the object's properties. In this way, a robot can attempt to select a plan of action that will minimize the chance of injury or error, while maximizing the realization of its goals and tasks.

In developing our humanoid robot simulation environment, we have initially focused on the specific tasks of using simulated vision for perception-based navigation, path planning for object grasping and manipulation, and balance compensation. The goal is to construct a series of abstraction layers between the fine-grained, low-level control of the motors[17], and the robot's high-level goals.

2.1 Simulated Vision

One of the long-term goals of the robotics research community is to build reliable software that will allow a robot to explore an unknown environment, and incrementally build its own internal model of the world through sensors. Toward this purpose, we are developing a simulated vision system for testing and evaluating experimental vision algorithms. By utilizing graphics rendering hardware, a virtual robot can acquire virtual camera images from a 3D scene model.

Computer graphics and computer vision are complementary disciplines: while the goal of graphics is to *generate images from a model*, the goal of vision is to *extract a model from images*. In order to evaluate the efficacy of an experimental vision technique, the following steps can be performed:

1. The virtual robot acquires image data from a scene (the scene is graphically rendered from the robot's point of view).

2. An internal model of the sensed virtual environment is constructed using whatever experimental vision technique is being tested.

3. The internal model is directly compared to the actual 3D model of the scene and an appropriate error metric is computed.

We are currently developing a simulated vision system for testing and evaluating a real-time 3D vision algorithm that has been used for legged robot navigation[8]. Similar simulated vision systems have also been successfully used in teleoperation applications [1].

A simulation environment can also be used for testing vision techniques other than those used for model-building. For example, evaluating image segmentation or object recognition techniques, the scene model can be initially divided up into a collection of small to medium-sized object models. Each object model is assigned a unique ID. A color table is initialized to represent a one-to-one mapping between object IDs and colors. To check which objects are visible to a virtual robot, the scene can be rendered from the robot's point of view, using flat shading and using the unique color for each object as defined by the object ID. If the resulting image pixels are scanned, a list of visible objects can be obtained from the pixel color information and compared to the output of the vision algorithm.

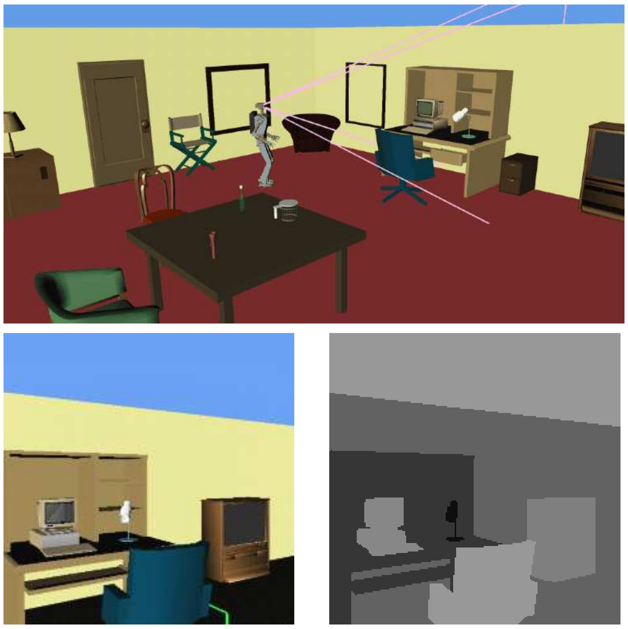

Figure 4: Simulated vision of a humanoid robot.

2.2 Motion Planning

Motion planning problems involve searching the system configuration space for a collision-free path connecting a given start and goal configuration[13]. For problems in low dimensions, the configuration space can often be explicitly represented and complete algorithms are known[16, 2]. For high-dimensional configuration spaces however, it is typically impractical to explicitly represent the configuration space. Instead, the space is often randomly sampled in order to discover free configurations[9].

We have initially focused on motion planning for two types of tasks: *global navigation* among obstacles, and computing collision-free *object manipulation* motions. For global navigation, the task of the robot is to move from a start location to a goal location while avoiding obstacles. For the specific case of navigating indoors, an explicit representation of the free space can usually be constructed and used to plan a coarse path connecting the start and the goal. The individual degrees of freedom are typically then controlled via a locomotion gait (e.g. walking, jogging, or crawling cycles) while attempting to closely follow the coarse path. For manipulation planning, the task is specified by identifying a target object to be moved and its new location. The motion planning software will then attempt to compute three trajectories:

- *Reach*: Position the robot to grasp the object.
- *Transfer*: After grasping, move the object to the target location
- *Return*: Once the object has been placed at the target location, release it and return the robot to its rest position.

In this case, the start and the goal are body postures that must be connected by a path in the configuration space. If a path at each phase is successfully returned by the path planner, the robot executes the motion, possibly guided by visual or force feedback.

There are many potential uses for such software, with the primary one being a high-level control interface for solving complex navigation or object manipulation tasks. Some basic experiments using path planning in a simulation environment for both manipulation and global navigation are described in Section 3.

2.3 Balance Compensation

Humanoid robots are much more difficult to control than wheeled robots, as it is much harder to preserve dynamic stability. Software that can adjust the individual degrees of freedom in order to maintain balance is required for building successful humanoids.

We have incorporated an online balance compensation algorithm[17, 7] within our simulation environment. We allow the user to interactively modify individual joint values of a humanoid robot, and adjust the other joints in real-time in order to compensate for any potential loss of balance. The algorithm acts as a filter function, transforming a purely kinematic input motion into a dynamically-stable output motion. Results of the current implementation are described in the following section.

3 Experiments

This section presents some preliminary results using a prototype simulation environment. Although the current capabilities are limited, our goal is to incrementally build upon the existing software framework for future experiments.

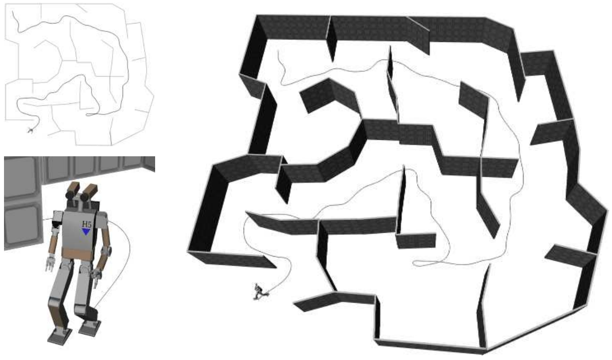

Figure 5: Exploring an unknown maze environment.

3.1 Perception-Based Navigation

Robust sensor-based navigation algorithms are vital for the development of humanoid robots that can coexist with real humans. Figure 4 depicts a humanoid robot navigating in a virtual office. An outline of a representative portion of the robot's viewing frustum is shown in each. The bottom two images show the scene rendered from the robot's point of view, the left image being full-color, and the right image an object-based segmentation image.

Using the simulated vision technique described in Section 2.1 and a fast navigation path planner[12], the behavior of a virtual robot was tested while exploring an initially unknown maze environment (Figure 5). Information about obstacles in the environment is incrementally added to the robot's internal model of the world as the robot moves. The robot is given the task of navigating from the upper-left corner to the bottom-left corner of the maze. The final path taken by the robot is solely the result of the interaction between path planning based in the robot's current internal model and the visual feedback obtained during exploration.

3.2 Object Grasping and Manipulation

In the examples shown in Figure 6 and Figure 7, the synthetic vision module is used in order to verify that a particular target object is visible to a virtual humanoid prior to attempting to grasp it. If the object is visible, a manipulation planner is invoked to plan a collision-free path to grasp the object. If the target object is not visible, the humanoid will attempt to reposition itself, or initiate a searching behavior in an attempt to find the missing object.

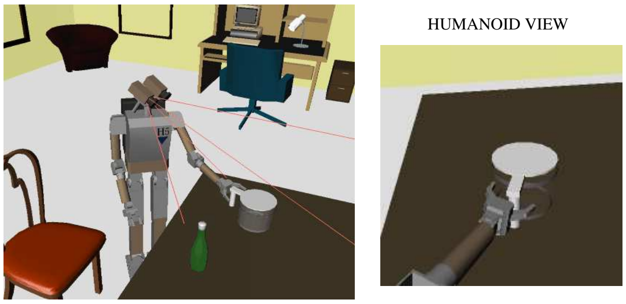

Figure 6: Reaching and grasping a coffee pot.

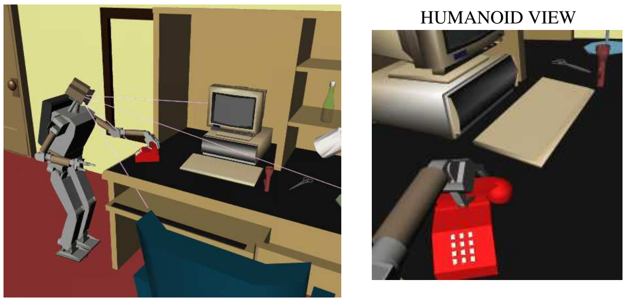

Figure 7: Answering the telephone.

The path planning algorithm used in these examples was *RRT-Connect* [11], which was originally developed to plan collision-free motions for animated characters in 3D virtual environments[12]. Combined with an inverse kinematics algorithm, the planner facilitates a task-level control mechanism for planning manipulation motions. Through a graphical user interface, an operator can click and drag an object to a target location and issue a *move* command.

Figure 1 shows snapshots of a planned motion for the 'H6' humanoid repositioning a bottle from the lower shelf to the upper shelf. Figure 8 shows the H5 humanoid model playing chess. Each of the motions necessary to reach, grasp, and reposition a game piece on the virtual chessboard were generated in less than one second on average on a 270 MHz SGI O2 workstation. Each robot arm comprises a 6-DOF kinematic chain, and each scene contains a total of over 10,000 triangle primitives. The 3D collision checking software used for these experiments was the RAPID library based on OBB-Trees developed by the University of North Carolina[4].

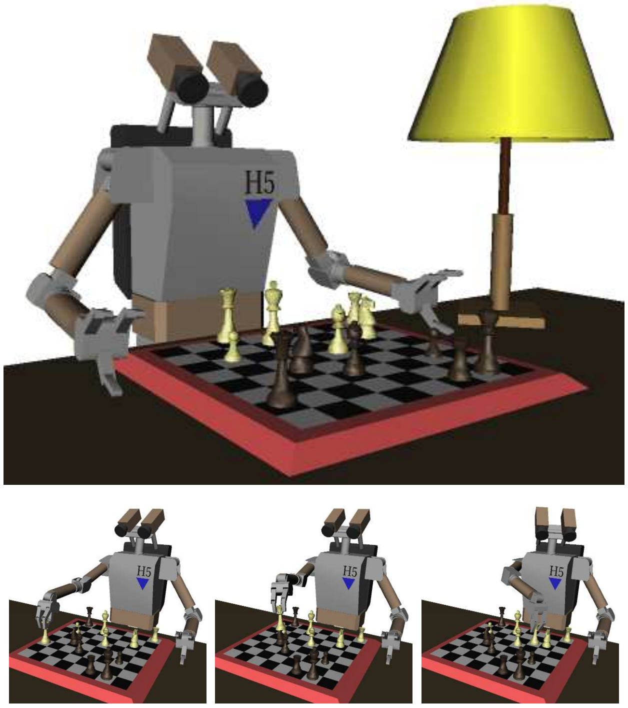

Figure 8: The 'H5' humanoid robot playing chess.

3.3 Interactive Balance Compensation

Figure 9 shows a series of stable postures for the 'H5' humanoid robot. In the current implementation of our simulation environment, the user can interactively modify individual joint values, as well as specify either dual-leg or single-leg supports. The balance compensation software simultaneously adjusts the remaining joint values in order to satisfy balance constraints.

This software has been incorporated into a motion planner that automatically generates full-body dynamically-stable motions[10]. In the future, we envision this software being used as a "virtual puppet" interface for interactively controlling a humanoid robot's posture *safely* and intuitively, and without the need for any special hardware aside from a graphic display.

4 Discussion

This paper presents a software simulation environment as a tool for building future humanoid robotic systems. Through simulation, the overall behavior of the robot system can be visualized and tested under

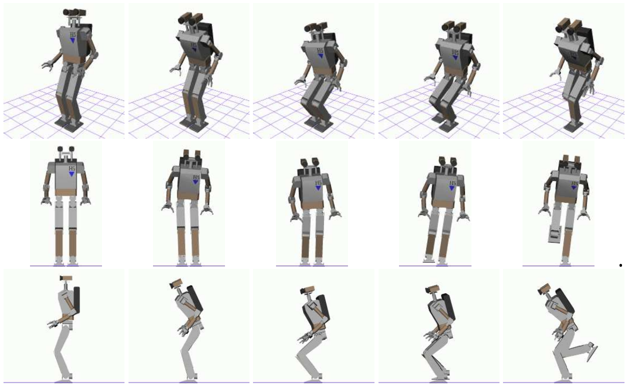

Figure 9: Dual and single-leg stable postures for 'H5'.

a variety of circumstances in many different environments. This should help to both shorten the development cycle time, as well as reduce the costs and safety risks involved during the initial phases of testing. We believe well-constructed simulations will greatly assist in the development of robust high-level behavior control software for complex robotic systems such as humanoids.

Many issues remain to be solved. Although simulation software has increased in both sophistication and speed, the results are often highly dependent on the accuracy of the models used. For example, precise dynamic simulation generally requires accurate models of friction. While the use of more complex models improves accuracy, this typically comes at an higher computational cost. However, as the computing hardware continues to improve dramatically, we expect to see increasingly sophisticated simulations of robotic systems in the near future.

Acknowledgments

This research is supported in part by a Japan Society for the Promotion of Science Postdoctoral Fellowship for Foreign Scholars in Science and Engineering. Many thanks to collaborators Fumio Kanehiro, Koichi Nishiwaki, and Kei Okada.

References

[1] B. Brunner, K. Landzettel, B.-M. Steinmetz, and G. Hirzinger. Telesensorprogramming - a task-directed programming approach for sensor-based space robots. In *Proc. of Int. Conf. on Advanced Robotics (ICAR)*, 1995.

[2] J.F. Canny. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, MA, 1988.

[3] Y. Nakamura *et. al.* V-HRP: Virtual humanoid robot platform. In *Proc. of First IEEE-RAS Int. Conf. on Humanoid Robots*, September 2000. *To appear.*

[4] S. Gottschalk, M. C. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In *SIGGRAPH '96 Proc.*, 1996.

[5] Kazuo Hirai. Current and future perspective of honda humanoid robot. In *Proc. of 1997 IEEE/RSJ Int. conf. on Intelligent Robots and Systems*, pages 500–508, 1997.

[6] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl. Sensor-based space robotics - rotex and its telerobotic features. *IEEE Transactions on Robotics and Automation*, 9(5), October 1993.

[7] S. Kagami, F. Kanehiro, Y. Tamiya, M. Inaba, and H. Inoue. AutoBalancer: An online dynamic balance compensation scheme for humanoid robots. In *Proc. of Fourth Int. Workshop on Algorithmic Foundations on Robotics*, 2000.

[8] S. Kagami, K. Okada, M. Inaba, and H. Inoue. Large design and development of onbody realtime disparity image generation system. In *Proc. of 4th Robotics Symposia 1999*, pages 177–182. Robotics Society of Japan, 1999.

[9] L. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration space. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, 1996.

[10] J. Kuffner, S. Kagami, M. Inaba, and H. Inoue. Dynamically-stable motion planning for humanoid robots. In *Proc. of First IEEE-RAS Int. Conf. on Humanoid Robots*, September 2000. *To appear.*

[11] J.J. Kuffner and S.M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'2000)*, San Francisco, CA, April 2000.

[12] J.J. Kuffner Jr. *Autonomous Agents for Real-time Animation.* PhD thesis, Stanford University, 1999.

[13] J. C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston, MA, 1991.

[14] K. Nagasaka, M. Inaba, and H. Inoue. Walking pattern generation for a humanoid robot based on optimal gradient method. In *Proc. of 1999 IEEE Int. Conf. on Systems, Man, and Cybernetics*, 1999.

[15] M. Oda, K. Kibe, and F. Yamagata. ETS-VII - space robot in-orbit experiment satellite. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, April 1996.

[16] J. T. Schwartz and M. Sharir. On the 'piano movers' problem: Ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in applied Mathematics*, 4:298–351, 1983.

[17] Y. Tamiya, M. Inaba, and H. Inoue. Realtime balance compensation for dynamic motion of full-body humanoid standing on one leg. *Journal of the Robotics Society of Japan*, 17(2):268–274, 1999.

[18] J. Yamaguchi, S. Inoue, D. Nishino, and A. Takanishi. Development of a bipedal humanoid robot having antagonistic driven joints and three dof trunk. In *Proc of 1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'98)*, pages 96–101, 1998.