UT Austin Computer Sciences Technical Report AI09-01,
February 2009.

The UT Austin Villa 3D Simulation Soccer Team 2008

Shivaram Kalyanakrishnan, Yinon Bentor and Peter Stone

Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712-0233
{shivaram, yinon, pstone}@cs.utexas.edu

Abstract. This paper describes the research focus and ideas incorporated in the UT Austin Villa 3D simulation soccer team that participated in the RoboCup 2008 competitions held in Suzhou, China.

1 Introduction

In this paper, we describe the agent our team UT Austin Villa developed for participation at the RoboCup 3D Simulation Soccer competition 2008, which was held in Suzhou, China. The main challenge presented by the 3D simulation league is the low-level control of a humanoid robot with more than 20 degrees of freedom. The simulated environment is a 3-dimensional world that models realistic physical forces such as friction and gravity, in which teams of humanoid robots compete with each other. Thus, the 3D simulation competition paves the way for progress towards the guiding goal espoused by the RoboCup community, of pitting a team of 11 humanoid robots against a team of 11 human soccer players. Programming humanoid agents in simulation, rather than in reality, brings with it several advantages, such as making simplifying assumptions about the world, low installation and operating costs, and the ability to automate experimental procedures. All these factors contribute to the uniqueness of the 3D simulation league.

The approach adopted by our team UT Austin Villa to decompose agent behavior is bottom-up in nature, comprising lower layers of joint control and inverse kinematics, on top of which skills such as walking, kicking and turning are developed. These in turn are tied together at the high level of strategic behavior. Details of this architecture are presented in this paper, which is organized as follows. Section 2 provides a brief overview of the 3D humanoid simulator. In Section 3, we describe the design of the UT Austin Villa agent, and elaborate on its skills in Section 4. In Section 5, we draw conclusions and present directions for future work.

2 Brief Overview of 3D Simulation Soccer

2007 was the first year of the 3D simulation competition in which the simulated robot was a humanoid. The humanoid used in the 2007 RoboCup competitions in Atlanta, U.S.A., was the Soccerbot, which was derived from the Fujitsu HOAP-2 robot model [2]. Owing to problems with the stability of the simulation, the Soccerbot was replaced by the Aldebaran Nao robot [1] at the 2008 RoboCup competitions in Suzhou, China. The robot has 22 degrees of freedom: six in each leg, four in each arm, and two in the neck and head. Figure 1 shows a visualization of the Nao robot and the soccer field during a game. The agent described in the following sections of this paper is developed for the Nao robot.

Each component of the robot's body is modeled as a rigid body with a mass that is connected to other components through joints. Torques may be applied to the motors controlling the joints. A physics simulator (Open Dynamics Engine [3]) computes the transition dynamics of the system taking into consideration the applied torques, forces of friction and gravity, collisions, etc. Sensation is available to the robot through a camera mounted in its torso, which provides information about the positions of all the objects on the field every cycle. In the 2008 competitions, *noise-free* visual information was provided, with an *unrestricted* field of vision. The visual information, however, does not provide a complete description of state, as details such as joint orientations of other players and the spin on the ball are not conveyed. Apart from the visual sensor, the agent also gets information from touch sensors at the feet and gyro rate sensors. The simulation progresses in discrete time intervals with period 0.02 seconds. At each simulation step, the agent receives sensory information and is expected to return a 22-dimensional vector specifying torque values for the joint motors.

Since 2007 was the year the humanoid was introduced to the 3D simulation league, the major thrust in agent development thus far has been on developing robotic skills such as walking, turning, and kicking. This has itself been a challenging task, and is work still in progress. High-level behaviors such as passing and maintaining formations are beginning to emerge, but at this stage, they play less of a role in determining the quality of play when compared to the proficiency of the agent's skills.

Figure 1. On the left is a screenshot of the Nao agent, and on the right a view of the soccer field during a 3 versus 3 game.

3 Agent Architecture

At intervals of 0.02 seconds, the agent receives sensory information from the environment. The visual sensor provides distances and angles to different objects on the field from the agent's camera, which is located in its torso. It is relatively straightforward to build a world model by converting this information about the objects into Cartesian coordinates, particularly because visual sensation is complete and noise-free. In addition to the vision perceptor, the our agent also employs its force resistance perceptors at the feet to determine whether they touch the ground. At this point, we do not make use of information from the gyro rate perceptors and auditory channels.

Once a world model is built, the agent's control module is invoked. Figure 3 provides a schematic view of the control architecture of our humanoid soccer agent.

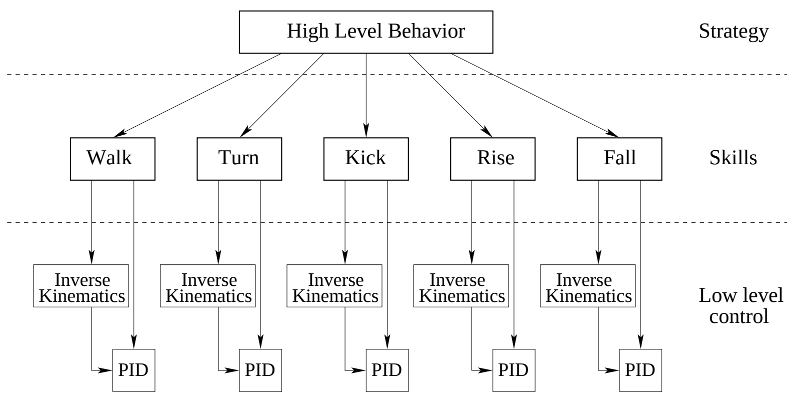

Figure 2. Schematic view of UT Austin Villa agent control architecture.

At the lowest level, the humanoid is controlled by specifying torques to each of its joints. We implement this through PID controllers for each joint, which take as input the desired angle of the joint and compute the appropriate torque. Further, we use routines describing inverse kinematics for the arms and legs. Given a target position and pose for the foot or the hand, our inverse kinematics routine uses trigonometry to calculate the angles for the different joints along the arm or the leg to achieve the specified target, if at all possible. The PID control and inverse kinematics routines are used as primitives to describe the agent's skills, which are described in greater detail in Section 4.

Developing high-level strategy to coordinate the skills of the individual agents is work in progress. Given the limitations of the current set of skills, we employ a simple high-level behavior for 3 versus 3 games. We instruct the goalie to remain standing a little in front of the goal and to intercept a ball that is heading towards the goal. Of the other two players, one player (the forward) intercepts the ball and kicks it towards the goal once intercepted, while the other heads to a position a few meters behind to act as cover if the opponent team gets past the forward player.

4 Player Skills

Our plan for developing the humanoid agent consists of first developing a reliable set of skills, which can then be tied together by a module for high-level behavior. Our foremost concern is locomotion. Bipedal locomotion is a well-studied problem (for example, see Pratt [9] and Ramamoorthy and Kuipers [10]). However, it is hardly ever the case that approaches that work on one robot generalize in an easy and natural manner to others. Programming a bipedal walk for a robot demands careful consideration of the various constraints underlying it.

We experimented with several traditional approaches to program a walk for the humanoid robot, including monitoring its center of mass, specifying trajectories in space for its feet, etc. Through a process of trial and error, we concluded that a dynamically stable walk is faster and more robust than a statically stable walk.[1] We achieved a reasonably fast dynamically stable walk by programming the robot to raise its left and right feet alternately (and perfectly out of phase) to a certain height above the ground, and then moving them forwards a small distance, and finally stretching them back to their initial configurations. This is implemented as a periodic state machine with four states. Inverse kinematics routines determine joint angles for the feet given the target position. Interestingly, it is more appropriate to describe this walk routine as a "run", since there occur points in time during its execution when both feet lose contact with the ground. By making minor changes to our walk routine, we were able to realize other useful skills for the robot. For getting the robot to turn, all that was required was to orient one of its hips at a slight angle while continuing to beat its feet up and down. Other slight variations of this basic pattern allowed for skills such as walking sideways and walking backwards.

Before invoking the kicking skill, we ensure that the agent is placed within a certain distance behind the ball and within some lateral displacement with respect to the direction in which the ball is to be kicked. From such a position, we use inverse kinematics to align the foot in the kicking leg behind the ball. The kicking motion is essentially a swing of the whole leg with respect to the hip joint: after swinging backwards by some angle, the leg is immediately swung forwards with a reasonably high gain for the motor torque at the hip until the foot impinges the ball. After this motion is completed, the kicking foot is brought back to return the robot to a standing position.

Two other useful skills for the robot are falling (for instance, by the goalie to block a ball) and rising from a fallen position. We programmed the fall by having the robot bend its knee, by virtue of which it would lose balance and fall to one side. Our routine for rising is divided into stages. If fallen face down, the robot bends at the hips and stretches out its arms until it transfers weight to its feet, at which point it can stand up by straightening the hip angle. If fallen sideways, the robot first gets to the face down position by rolling over. If fallen

[1] By "dynamically stable", we mean that the robot is balanced as long as it is moving; by "statically stable", we mean that the robot remains balanced even if it stops moving abruptly.

face up, the robot gets to a position in which it is fallen sideways, from which it gets to the face down position, before finally getting back on its feet.

Table 1 lists values of some of the relevant statistics for the set of skills implemented on our agent.[2] We collected times and distances for forward walking, backward walking, and kicking over 10 trials each, starting each run a newly instantiated agent. Side walking and turning times were collected over 20 trials, 10 for walking or turning left and 10 for right. The difference between left and right walking or turning speeds was not significant; the results are averages of distances or times in both directions.

In front and side walking trials, we placed a single agent in front of the right goal and instructed it to immediately begin walking. For back walking trials, we instead placed the agent in the center of the field. Each agent walked for approximately ten seconds and recorded its position and time at each cycle. For right and left turning, agents completed a full turn and reported their times at the beginning of the turn and again when they were within approximately 1 degree of their initial heading.

To assess times to rise from forward and backward falls, we started the agent at quarter-field, clear of any obstacles, and instructed it to fall. We recorded the last time stamp before fall was detected (based on thresholding the "lean angle") and again after the agent returned to an upright state. As described earlier, agents that fell backwards flipped themselves over before attempting to rise, accounting for nearly double times-to-rise.

Videos of our agent's skills are available at a supplementary website [4].

Table 1. Performance statistics for skills performed by the UT Austin Villa agent. The mean and one standard error are reported.

Skill Statistic	Performance Value
Forward walking: linear velocity	(144.27 ± 1.22) mm/sec
Side walking: linear velocity	(62.80 ± 2.00) mm/sec
Backward walking: linear velocity	(150.42 ± 4.53) mm/sec
Turning: angular velocity	(19.96 ± 3.15) deg/sec
Kicking: distance reached by ball after kick	(3122.68 ± 14.47) mm
Rising: Time to rise after falling forwards	(10.21 ± 0.94) sec
Rising: Time to rise after falling backwards	(23.14 ± 0.81) sec

5 Conclusions and Future Work

The simulation of a humanoid robot opens up interesting problems for control, optimization, machine learning, and AI. The initial overhead for setting up the infrastructure is bound to be overtaken by the progress made through research on this important problem in the coming years. While the main emphasis thus far has been on getting a workable set of skills for the humanoid, it is conceivable that soon there will be a shift to higher level behaviors as well. A humanoid soccer league with scope for research at multiple layers in the architecture offers a unique challenge to the RoboCup community and augurs well for the future.

[2] We express thanks to Hugo Picado from the FC Portugal team for sharing with us a list of relevant statistics for the robot.

There are numerous vistas that research in the 3D humanoid simulation league is yet to explore; these provide the inspiration and driving force behind UT Austin Villa's desire to participate in this league.

UT Austin Villa has been involved in the past in several research efforts involving RoboCup domains. Kohl and Stone [8] used policy gradient techniques to optimize the gait of an Aibo robot (4-legged league) for speed. Stone *et al.* [11] introduced Keepaway, a subtask in 2D simulation soccer [5, 7], as a test-bed for reinforcement learning, which has subsequently been researched extensively by others (for example, Taylor and Stone [12], Kalyanakrishnan *et al.* [6], and Taylor *et al.* [13]). We are keen to extend our research initiative to the 3D simulation league. Our initial focus for the 2009 competition will be on optimizing our set of skills, to realize faster walks, more powerful and accurate kicks, etc. Banking on a reliable set of skills, we will seek to develop higher level behaviors such as passing and intercepting balls.

References

1. Aldebaran Humanoid Robot Nao. `http://www.aldebaran-robotics.com/eng/`.
2. Fujitsu Humanoid Robot HOAP-2. `http://jp.fujitsu.com/group/automation/en/services/humanoid-robot/hoap2%/`.
3. Open Dynamics Engine. `http://www.ode.org/`.
4. UT Austin Villa 3D Simulation Team. `http://www.cs.utexas.edu/~AustinVilla/sim/3Dsimulation/`.
5. M. Chen, E. Foroughi, F. Heintz, Z. Huang, S. Kapetanakis, K. Kostiadis, J. Kummeneje, I. Noda, O. Obst, P. Riley, T. Steffens, Y. Wang, and X. Yin. Users manual: RoboCup soccer server — for soccer server version 7.07 and later. *The RoboCup Federation*, August 2002.
6. S. Kalyanakrishnan, Y. Liu, and P. Stone. Half field offense in RoboCup soccer: A multiagent reinforcement learning case study. *Proceedings of the RoboCup International Symposium 2006*, June 2006.
7. H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. RoboCup: A challenge problem for AI. *AI Magazine*, 18(1):73–85, 1997.
8. N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2004.
9. J. Pratt. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots*. PhD thesis, Computer Science Department, Massachusetts Institute of Tehcnology, Cambridge, Massachusetts, 2000.
10. S. Ramamoorthy and B. Kuipers. Qualitative hybrid control of dynamic bipedal walking. *Robotics: Science and Systems*, II, 2007.
11. P. Stone, R. S. Sutton, and G. Kuhlmann. Reinforcement learning for RoboCup-soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.
12. M. E. Taylor and P. Stone. Behavior transfer for value-function-based reinforcement learning. In F. Dignum, V. Dignum, S. Koenig, S. Kraus, M. P. Singh, and M. Wooldridge, editors, *The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 53–59, New York, NY, July 2005. ACM Press.
13. M. E. Taylor, S. Whiteson, and P. Stone. Comparing evolutionary and temporal difference methods for reinforcement learning. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1321–28, July 2006.