RoboPatriots: George Mason University 2010 RoboCup Team

Keith Sullivan, Christopher Vo, Sean Luke, and Jyh-Ming Lien

Department of Computer Science, George Mason University
4400 University Drive MSN 4A5, Fairfax, VA 22030 USA
{ksulliv2, cvo1, sean, jmlien}@cs.gmu.edu

1 Introduction

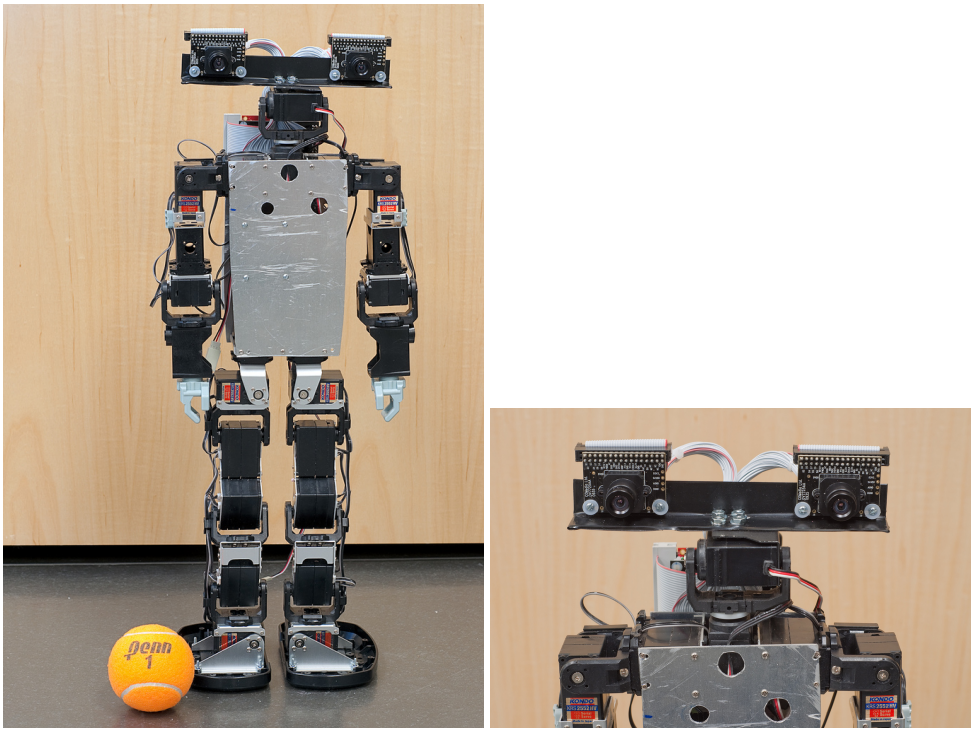
The RoboPatriots are a team of three humanoid robots sponsored by the Computer Science Department at George Mason University. Each robot is based on the Kondo KHR-3HV and a customized Surveyor SVS camera (see Figure 1(a)).

Our goals for RoboCup 2010 are to develop a durable hardware and software platform for future research activities into vision based localization, path planning in dynamic environments, and multi-robot coordination. In addition, we continued our research into evolving human-like motions [1].

2 Hardware

We are interested in embodied AI, so we choose commercially available hardware rather than fabricating our own. After the 2009 competition, we realized our robots contained too much plastic and that the hardware architecture was too complicated. So, we completely redesigned the robots with more metal and a simpler architecture. Figure 2 shows the hardware architecture and information flow between components. The robot base is the Kondo KHR-3HV with an extended aluminum torso. Each robot has 3 DOF per arm, 5 DOF per leg, and 2 DOF in the neck. The sixteen Kondo KRS-2555HV digital servos used in the arms and legs produce 14 kg-cm of torque at a speed of 0.14 sec / 60 degrees. The 2555HV servos communicate via a serial protocol over RS485 and are controlled by the Kondo RCB-4 servo controller board. In addition, two KRG-3 single axis gyros connect to the RCB-4. The two Kondo KRS-788HV digital servos used our pan/tilt mount in the neck produce 10 kg-cm of torque at a speed of 0.14 sec. / 60 degrees. These servos are controlled by the Surveyor SVS vision system via PWM. Each robot has a 11.1V 2200 mAh Lithium-Ion battery.

Our main sensor and computing unit is the Surveyor SVS (Stereo Vision System) [2]. The SVS consists of two OmniVision OV 7725 camera modules connected to two independent 500 MHz Blackfin BF537 processors. The two OmniVision camera modules are mounted on a pan/tilt mount with 10.5 cm seperation. See Figure 1(b). Each camera module operates at 640x480 resolution with a 90-degree field of view. The two processors are connected by a dedicated SPI bus, and the SVS provides a Lantronix Matchport 802.11g wireless unit.



(a) Complete RoboPatriot robot (b) Close up of robot's head

Fig. 1. 2010 RoboPatriot's robot and close up of stereo vision head and pan/tilt mount.

Two single axis RAS-2 dual-axis accelerometers connect to the SVS. A custom inverter board allows serial communication at 115200 bps between the RCB-4 and the SVS. By the competition, we will modify the SVS circuit board to fit with the robot's form factor.

3 Software

The goalie and attackers each run a state machine for high level decision making. The state machines include states for approaching the ball, orientation for kicking, and kicking towards the goal. As in past years, predefined motions are stored on the RCB-4 and are not dynamically modifiable. However, we can interrupt motions during execution and can run cyclic motions for an arbitrary length (e.g., we can execute N walking steps based on sensor feedback). The software architecture is split across the RCB-4 and SVS as follows:

- The RCB-4 handles gryo stabilization and execution of predefined motions.
- The SVS detects falls, performs vision related tasks (discussed below), runs the state machine, and responds to the referee box. These tasks are distributed across both processors of the SVS.

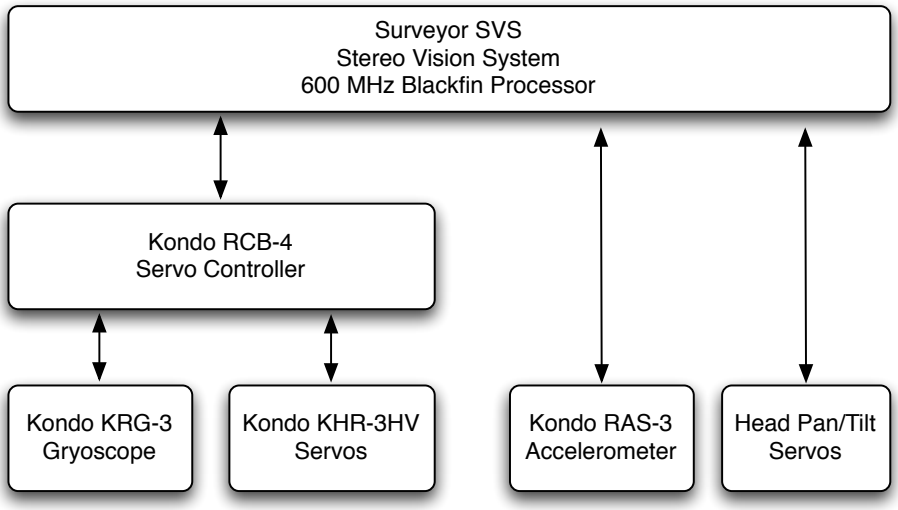


Fig. 2. The hardware architecture of the RoboPatriots, and the information flow between components.

Vision One camera module of the SVS handles basic color tracking, using first its own image, and then the image from the other camera module. If both camera modules detect the color, then stereo depth mapping combined with the camera pan/tilt position provides an approximate physical distance to the object of interest. In addition, ground-plane calibration allows us to ensure detected objects are on the floor, and shape detection ensures detection of appropriate objects such as the goals, the ball and field lines.

Localization Our localization module uses the distance information from the vision module combined with a priori knowledge about the field. The localization information is used for autonomous repositioning and path planning.

Path Planning Path planning is primarily used to move towards the ball. When moving towards the ball, the goal is to arrive at the ball approximately oriented towards the goal, thus requiring only small adjustments to kick the ball. Autonomous repositioning uses a simpler path planning algorithm since we are not as concerned about the final orientation. We use rapidly exploring random trees (RRT) [3] for both path planning modules.

4 Evolution of Motion

Using our custom simulator, we continued to explore using large scale evolutionary algorithms to discover stable motions. Figure 4 shows our simulator created

using the Open Dynamics Engines [4] to handle 3D physics. Unlike evolutionary robotics, we linked ECJ, our own evolutionary library [5], to the simulator, rather than perform evolution on the physical hardware to save wear-and-tear on the robots. Experience has shown that the evolved motions run on the real robots with minor corrections.

Like our previous work, we evolve a direct representation of servo controls due to the nature of our hardware. With the direct representation, the search space is huge, and we need a seed to bias the search towards plausible motions (i.e., stable motions without self-colision). However, this year, instead of seeding the evolutionary algorithm with human capture data, we use a poor, hand tuned motion as the seed. We discovered that the human capture data was unreliable to magnetic interference, and that retargeting the data to the robot was problematic due to the higher DOF of a human versus our robot.

This year we focused on modifying the evolutionary algorithm to increase exploration of the search space. The three techniques we used were *steady state selection*, *elitism*, and *dynamic crossover rate*.

Steady State Selection In steady state selection, after a child is produced it replaces the worst individual in the population, rather than a randomly chosen one ala tournament selection. Steady state selection reduces selection pressure and ensures that the average population fitness always increases. Thus, steady state selection should produce children of increasing fitness. Experiments showed that steady state selection performed slightly better than a generational approach (i.e., tournament selection).

Elitism In the generational model, we added elitism to improve convergence. Elitism keeps the best N individuals from one generation to the next without applying genetic operators, thus, ensuring some good genetic material in future generations. Elitism guards against randomly losing good individuals, and ensures the minimum population fitness does not decrease.

Dynamic Crossover Rate Due to the representation, crossover is very destructive and highly likely to produce unfit children. So, we introduce a discount factor to the rate of crossover: at each generation we reduce the rate of crossover by a constant factor α. Changing the crossover rate decreases exploration as the EA runs, and focuses more on exploitation (mutation) in the future. In other words, early on, the EA will explore widely different areas of the search space, and then will then exploit theses areas similar to hill climbing or simulated annealing.

Acknowledgments

We would like to thank Brian Hrolenok, James O'Berine, Faisal Abidi, Zoran Duric, Nathalia Peixoto and Surveyor, Inc. for their assistance.

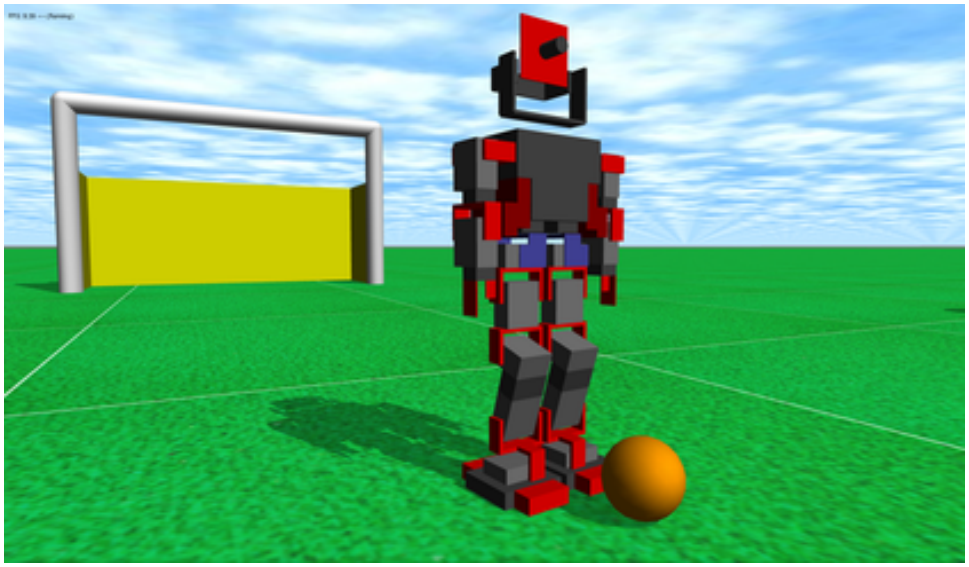


Fig. 3. Our custom robot simulator used in evolutionary algorithm experiments

References

1. Sullivan, K., Vo, C., Hrolenok, B., Luke, S.: RoboPatriots: George Mason University 2009 RoboCup team. In: Proceedings of the 2009 RoboCup Workshop. (2009)
2. : Surveyor stereo vision system. `http://www.surveyor.com/stereo/stereo_info.html` (2010)
3. LaValle, S.M.: Planning Algorithms. Cambridge University Press (2006)
4. Smith, R.: Open dynamics engine. `http://www.ode.org` (2009)
5. Luke, S.: ECJ 18: A Java-based evolutionary computation research system. `http://cs.gmu.edu/~eclab/projects/ecj/` (2008)