Robocup 2009
Standard Platform League
Team BURST Description

Gal Kaminka, Eli Kolberg, Eran Polosetski,
Vladimir (Vova) Sadov, Alon Levy,
Asaf Shilony, Anat Sevet, Elad Alon, and Meytal Traub

Computer Science & Computer Engineering Departments,
Bar Ilan University, Israel
galk@cs.biu.ac.il
http://shwarma.cs.biu.ac.il/robocup/

Abstract. Team BURST is a newly-formed joint team which intends to compete in the RoboCup 2009 standard platform league. BURST is the first non-Junior RoboCup team from Israel, ever. Our main research interests within the scope of the SPL are *Robust, realtime vision*, *Machine learning for gait generation and adaptation* and *Architectures for humanoid decision-making*.

Key words: Humanoids, Nao, Robotic soccer.

1 Introduction

Team BURST (Bar-ilan University Robotic Soccer Team) is a newly-formed joint team which is competing in the RoboCup 2009 standard platform league. BURST is the first senior (non-Junior) RoboCup team from Israel, ever. The team is composed of two previously-independent RoboCup efforts at Bar Ilan University, lead by Drs. Kaminka (Computer Science department) and Kolberg (Computer Engineering department). Both team leaders have considerable experience in RoboCup (as participants, organizers, league chairs, and symposium co-chairs). All student members of the team have relevant experience in robotics. Some also bring to bear professional programming and team leadership experience.

We have a long history of publications, in and out of RoboCup forums, and in and out of robotic soccer (dozens of publications in the last three years, alone). We take RoboCup-based research very seriously, and have focused research goals that we hope to achieve by utilizing the robotic platform, and our participation in the league. Specifically, we are interested in: (i) anytime object-recognition (as a basis for anytime visual SLAM); (ii) using reinforcement learning to generate robust and speedy biped and quadped gaits; and (iii) decision-making architectures for humanoid robots.

We hope that by participating in the standard platform league, we would be able to attract Israeli fans and researchers to take a more active part in RoboCup in the coming years.

2 Software Architecture

Our software architecture is decomposed into vision (see 2.1), localization (see 2.2), actuation, communication and high level behavior (see 2.5) parts. They are implemented as follows:

- *Actuation, Low level Sensors, Walking* - We use the NaoQi client-server architecture (provided by Aldebaran) to access the lowest level modules and the actuation: namely, joint movement and walking. The actual motions we use were generated by the Choreographe tool and by reverse engineering the open loop walk engine of Aldebaran.
- *Communication* - A C++ module running in the NaoQi process exports all relevant variables to shared memory (we needed the speed for out of process on the same robot communication), while another module records these results to a gzipped file. We use ALMemory/Shared Memory/sockets to communicate, using the best means for each option.

2.1 Vision

We use vision code courtesy of Northern Bites [2, 3] with few changes, including improved color calibration tools and ball recognition. Basically this means the usual colorspace → colorset, run length encoding, blobing and object detection. Their implementation is a single module including both vision and localization code, which made it necessary for us to export the required variables through ALMemory / Shared memory.

2.2 Localization

We use localization code courtesy of Northern Bites [2, 3] with no planned changes.

2.3 Locomotion

Our ultimate goal is to create a robust closed-loop locomotion controller which is able to adjust itself to the changing terrain conditions, to switch gaits dynamically and prevent falls. To achieve this goal we have started from the basic NaoQi open-loop controller and generated several stable gaits variations with NaoQi built-in parameters. Using the resulting scripts we extracted command sequences for each joint involved in walking and built our own open-loop locomotion controller fully adjustable for our needs. We hope that FSR data, accelerometers/gyro sensors data and joints sensors position errors data will allow us to built a closed-loop controller on top of our current, open-loop locomotion controller.

2.4 Path Planning

A considerable effort has been made on studying motion path planning. Given starting coordinates (X,Y,Yaw), target coordinates (X,Y,Yaw) and set of all the gaits defined for the robot, we are building motion planner that will propose a combination of walking commands that minimizes total travel time. This project is performed by group of outstanding undergraduate students whom we are very proud to host in our lab.

2.5 Behaviors

Our high level behavior engine is a python script that can be run on a seperate host or on the robot itself. This fascilitates faster development as compared to a naoqi module, and by using shared memory we achieve comparable speeds.

Behaviors are written in python, a high level language that facilitates fast development, and using a framework written especially for the robocup event, which includes an event based programming model and a deferred mechanisem (the later is based on our team's experience with [4]).

References

1. Team BURST Homepage, `http://shwarma.cs.biu.ac.il/robocup/`
2. Northern Bites Homepage, `http://robocup.bowdoin.edu/blog/`
3. Northern Bites Source code Repository `http://github.com/northern-bites/`
4. Twisted Matrix `http://twistedmatrix.com/trac/`