The Northern Bites 2013 Standard Platform League Team

Eric Chown, Elizabeth Mamantov
Wils Dawson, Edward Googins, Benjamin Mende, Josh Zalinger
Josh Imhoff, Brian Jacobel, Ellis Ratner, Daniel Zeller
Department of Computer Science
Bowdoin College
8650 College Station
Brunswick, ME, 04011-8486, USA
echown@bowdoin.edu
`http://robocup.bowdoin.edu`

May 24, 2013

Abstract

This document serves as the team description paper for the Northern Bites entry into the RoboCup 2013 Standard Platform League World Championship competition in Eindhoven, the Netherlands.

1 Introduction

The Northern Bites is Bowdoin College's Standard Platform League RoboCup team. As a small undergraduate liberal arts college, Bowdoin is not the typical participant in international robotics competitions. The team was a late entry into what was formerly the Four-Legged League but has performed strongly since its inception. The Northern Bites won two games at the U.S. Open 2006, finished tenth at RoboCup 2006, and in 2007 took home first place in Atlanta. In 2008, the last year of Aibo competition, the Northern Bites placed third. In 2009 the team took second place in the new Standard Platform League (SPL) in Austria, and again made the quarterfinals in Singapore in 2010. The Northern Bites reached the semifinals in Istanbul in 2011 and again in Mexico in 2012, meaning that the team is one of only two SPL teams to make the final sixteen in every year since 2006.

The Northern Bites is a unique team because it hails from a small college—Bowdoin has fewer than 1600 students—with an even smaller computer science department, which has fewer than twenty senior majors and no graduate program. Hence, all team members are undergraduates. These students are not necessarily majoring in computer science; most do their work as an extracurricular activity without credit, while balancing a full course load. Our initial goal during our first two years of competition was to make RoboCup a positive educational experience as our team became functional. Because our primary goal was educational, we used little code from other teams, and with the exception of motion, we have continued to adhere to a policy of writing our major systems from scratch. This has served us well as we have not become bogged down with trying to understand other people's code or debug code we have not written. In recent years we have become increasingly research-focused.

This paper highlights some of the work currently being done by the Northern Bites as well as the features of the team that have made us successful.

2 Code Structure

Like most SPL teams, we divide our code up into several major sections, which include vision, localization, motion, comm, and behavior. The low-level, computation-intense systems are written in C++, while the behavior system is written in Python. This year we completely overhauled our architecture, in order to truly modularize our codebase [3]. We previously had a monolithic system with little overarching organization, so to improve the clarity and usability of our code, we converted our code base to a message-passing architecture where each major system is a module that can be treated as a black box in a data flow diagram. This type of improvement is especially important considering Bowdoin's unique position as an undergraduate team since new members generally join the team unprepared to handle a large academic code base and students have at most four years to learn about the system and contribute.

2.1 Message-Passing Architecture

The main advantage of our new architecture is that it removes the dependencies between main systems, so a student working on localization, for example, needs to know nothing about how data is produced and passed in from the vision system. From the point of view of each of our systems, all of the other systems are basically black boxes. Each system has *input portals* and *output portals*, through which it receives data from and provides data to other systems. A given system gets all of its input from its input portals and outputs its results through its output portals, and input portals can be "wired" to any output portal that produces the same type of information; the system module itself does not need to know where its data comes from or goes to.

Information is generally passed between portals in the form of *Google Protocol Buffer* messages, although our architecture can pass any C++ class as a message and we have implemented specialized message types for images. Since we transmit data not only between modules on the robot, but also over a network connection to other robots and, via a log file, to our offline debugging tool, we use Protocol Buffers for nearly all of our communication. Protocol Buffers are easy-to-use and feture-rich information holders that also provide efficient serialization and deserialization—Google relies on them and has optimized their performance since it has stringent efficiency requirements.

Modules are wired up to each other in a *diagram*, which is essentially a directed graph that determines the order in which to run the modules and handles the message passing and management. Since the core of the architecture has been implemented to be thread-safe, our system now consists of several diagrams, containing a variable number of modules, which run in parallel at different frame rates. We are currently working on an extension to this architecture where all of the wiring between modules can be specified as an XML file. This addition will allow us to edit the architecture in a GUI and also give us the ability to change the wiring while the robot is running. By simply changing the XML file, for example, we could have a robot switch from running on live input to using logged input or vice versa.

2.2 Two Legged Motion

Last year we ported the B-Human motion engine for our team's use. Our port was successful enough that it has since been ported by several other teams, notably last year's champion, Austin Villa. Now that we have reimplemented our architecture into a truly modular system, it will be even easier for other teams to port the motion system for use in their own code bases. The SPL often discusses the idea of having standard modules available for teams, especially new teams, to plug into their code rather than starting from scratch, and we have made a significant step toward this goal by making B-Human's motion system a candidate for such a module.

2.3 Vision Processing

Our vision system has not seen any major changes this year; this is a transition year before our next major project, which will be to do a complete transition to a new, more modern, vision system next year.

2.4 Localization

This year we completed our transition from using an Extended Kalman Filter to a particle filter localization system [4]. We have found that our new system is substantially improved and should be a much better foundation for undergraduates wishing to work on localization, as it is considerably easier for new team members to understand and modify.

Additionally, we have transitioned our ball localization system to using a multi-model Kalman Filter in a manner inspired by B-Human. Essentially we now have two Kalman Filters for ball tracking: one that always assumes that the ball is moving and another that always assumes that the ball is stationary. The result has been a significant improvement in our ability to track and predict the ball's location and velocity.

In the meantime we are continuing to develop a completely different localization system based on human cognitive mapping in parallel. Humans appear to anchor spatial representations at key points in space called *gateways* [2]. Gateways serve as a way of organizing spatial reasoning; they divide large space into smaller regions and provide obvious points where stored representations of an environment should be made. We are exploring this concept in the context of the RoboCup soccer field environment. As a first step in this process, we have completely switched our goalie away from using standard probabilistic methods of localization to using a human-inspired visual feedback-based approach [5]. We first tested the cognitive goalie system in Mexico City in 2011 and continue to refine it. We have found the system works remarkably well for something so early in its development; our goalie reliably positions itself in the goal and generally performs better than it ever did with any of the team's past probabilistic localization systems. The system still has limitations, though, as the gateways we use are limited to key points related to the goalie's unique position, and we have yet to generalize the system so that the goalie can aggressively move away from its home position and be sure to find its way back.

2.5 Behavior

Our behavior system work this year has focused on transitioning to the new SPL rules: adding a fifth robot to the team and adapting to a larger field. We have also continued to develop behavioral solutions to the problem of how best to deal with a symmetric field; for example, the reliability of the goalie has allowed us to utilize its ball observations to determine on which side of the field the ball is located. Since our particle filter localization system has recently improved, we are also working on incorporating its information into behavioral decisions and balancing it with the visual sanity checks that we have previously used to guide kicking.

2.6 Cognition

A major goal of our team is to continue pushing our code base to become increasingly cognitive or at least influenced by cognitive models. We are currently working on two major projects in this area, one of which is the localization work described above.

The second project involves building a memory system that is cognitive in flavor. The spatial system is crucial to forming episodic memories in what Eric Chown has termed ”spatial prototypes” [1]. The key to forming such memories appears to be tracking the amount of change in the visual system. By noticing when the amount of change changes the most (essentially the second derivative of the visual stream) key moments in time can be extracted to form the basis for spatial memory. For example an "episode" might start when a ball is kicked—the new movement causes a spike in visual change—and end when it stops rolling, which will cause a drop in visual change. In turn, these episodes can be abstracted across many examples and then used predictively. As with our experimental localization system, our first version of this system will be implemented for our goalie. The episodes that the goalie will try and learn about and generalize will be episodes where balls are kicked in the goalie's direction.

In addition, this year we have implemented a complete human-inspired visual memory system [5]. This system is based on human short-term memory and is implemented essentially as a ring buffer. This new memory gives our cognitive system additional information to use in addition to the "state" information that is aggregated through the localization system and will thus support high-level reasoning that is impossible when only the current state of the environment is considered. So far our work has been mainly on implementation, and we are only now beginning to use the system as part of our behavior system.

References

[1] E. Chown. Spatial prototypes. In T. Tenbrink and C. Claramunt, editors, *Representing space in cognition: Interrelations of behavior, language, and formal models*. Oxford University Press, 2013. to appear.

[2] E. Chown, S. Kaplan, and D. Kortenkamp. Prototypes, location and associative networks: Towards a unified theory of cognitive mapping. *Cognitive Science*, 19:1–52, 1995.

[3] W. Dawson. Extensible continuous integration framework. Technical report, Bowdoin College, 2013.

[4] E.J. Googins. Robust localization with mobile robots. Technical report, Bowdoin College, 2013.

[5] E.C. Mamantov. Cognitive visuo-spatial reasoning for robotic soccer agents. Technical report, Bowdoin College, 2013.