Programming and Multi-Robot Communications

A pioneering group forges a path to affordable multi-agent robotics

This article was coordinated by Robin Shoop, Director of the Carnegie Mellon Robotics Academy. Contributors to this piece, in addition to Robin, included:
Dr. Manuela Veloso, Computer Science Faculty CMU
Dr. Howie Choset, Robotics Faculty CMU
Robert Avanzato, Penn State Abington Engineering Faculty
Tim Friez, Robotics Academy Staff
Somchaya Liemhetcharat, Graduate Student
Ben Morse, Graduate Student

Robotic technologies are ubiquitous and are integrated into many modern devices yet most people do not recognize these devices as robots. The best example of a robotic system that millions of people have but they do not referee to as a robot is a cellphone. Today's cellphones have numerous sensors such as accelerometers, gyroscopes, cameras, multi-touch screens, Bluetooth, wireless modems allowing internet access and Global Positioning Systems integrated into them (all robotic technologies).

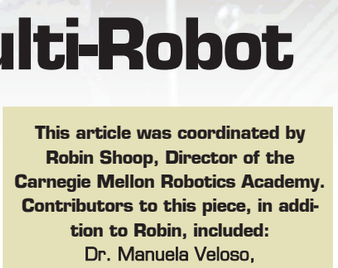
These sensors, when integrated into a system, can be used to provide a very accurate description of the location of your phone. In order to accomplish this location discovery, your phone is pre-programmed to use the correct combination of data from sensors, the Internet and a system of cellular networks to pin-point its location. Once the system has its location, it uses other pre-programmed algorithms and data to provide you with a set of directions to restaurants, gas stations or to a person's house. Programming and robotic technologies are finding their way into everyday devices and many people believe that the skill sets that students learn through CS and Robotics are "new basic skills" for future innovators. The Defense Advanced Research Projects Agency (DARPA) is providing support to researchers at Carnegie Mellon to develop inexpensive systems and training materials that will teach students the basics of multi-system communications. The developed materials from the research are being made available for free through CMU's Computer Science Student Network (CS2N).

Over the years, DARPA has worked to increase American innovation by funding research and technology development that not only has improved our military capabilities but also has significantly improved our quality of life (projects such as the Internet and GPS). Recently, DARPA is investing in computer science and robotic innovation with programs like the DARPA Grand Challenge, the DARPA Urban Challenge, and the CS2N. The current state of educational robotics is being outpaced by robotics innovation; schools are teaching students how to program single robots, yet tomorrow's robotic systems will require multi-agent communications. In the book "Outliers", Malcolm Gladwell repeatedly talks about the "10,000-Hour Rule", claiming that the key to success in any field is, to a large extent, a matter of practicing a specific task for a total of around 10,000 hours. If we want American's students to lead the world in innovation, then we need to develop an affordable system that will allow them to begin to put their first 10,000 hours in on thinking about multi-agent communications.

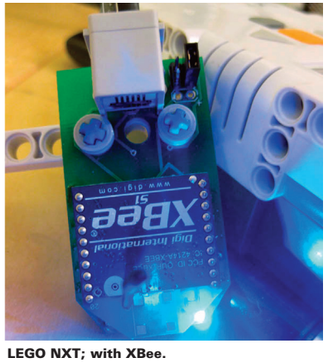
HARDWARE TESTING AND DEVELOPMENT

Our goal is to develop an inexpensive communications system that will not cost the end-user more than $150 per set; the cost assumes that the user already has a pair of robot controllers. We tested Arduino, LEGO, and VEX robot controllers using the low cost 1mW XBee Radio and an assortment of hardware from www.sparkfun.com. We were able to build a homebrew solution for less than $100 for a pair of robots. The homebrew solution requires the user to have a modest electronics background and to know what to purchase; a list of parts and building instructions can be found at our site. This summer, Dexter Industries, www.dexterindustries.com, developed a solution for the MINDSTORMS NXT called the NXTBee which costs $99 per pair. We used the Dexter solution all summer for training and found the NXTBee to be very reliable. CMU also worked closely with Dick Swan, the inventor of ROBOTC, to integrate libraries into ROBOTC to make multi-robot communications easy for students to implement. This story describes the types of lessons that CMU has developed and where you can find them to test them out.

VEX Cortex controller with XBee.



Arduino MEGA controller and XBee.



LEGO NXT; with XBee.

CURRICULUM DEVELOPMENT

Our goal is to develop a series of lessons to teach the concepts of multi-robot communication. The target audiences for these lessons are students with a basic understanding of programming single robots. Our initial lessons focus on robot setup and basic communications. This next section describes CMU's initial labs which are capable of being implemented using LEGO, VEX, or Arduino robots.

Setup. This lab provides a set of instructions that show people how to setup the Xbee radio? Plug it in, check for communications between the radio and the home robot, check to see if there is communications between the Xbee and another Xbee.

Basic Communications. These labs consist of several sample programs that can be loaded onto the robot and include instructions on how to modify the code. This set of lessons passes messages back and forth and displays them on the robots LCD screen.

Sending and Interpreting Instructions. This lab has one robot send information and another robot interpret the data and do something based on the message. (i.e. move 360 encoder counts)

Interpreting and Formatting Data. This next set of labs requires the student to develop functions that turn parameters like 90 into a 90 degree turning behavior, or a parameter like 1 or .5 into a behavior that allows the robot to travel one meter or one half meter.

In order to accomplish the introductory labs, we begin by teaching students a how to share messages between robots using strings. In computer programming a string is traditionally a sequence of characters, either as a literal constant or as some kind of variable. With each lesson we provide example code in ROBOTC that broadcasts a "string" out to the network – that is, it sends a "message" to all other robots that are listening. Next, we provided sample code on how to receive a string.

"A string is traditionally a sequence of characters, either as a literal constant or as some kind of variable."

Strings are the basic communication tools used to pass messages.

ROBOTC Code Example

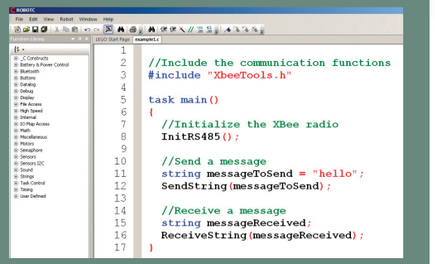
In the slide below you will see some of the reserve words that are incorporated into ROBOTC.

InitRS485();
• A Function to initialize the Xbee radio

SendString(message);
• Sends a string to every other robot in range

ReceiveString(message);
• Receives a string that has been sent from another robot

• Delays the program until the message is received



In the ROBOTC Multi-Robot Communication Libraries there are three key functions that are used. First is an initialization command which is a command specific for the NXT that is used to setup the NXT's sensor port 4 to communicate with the Xbee Wireless Radio. The Xbee Wireless Radio requires a serial (RS-232 or RS-485) connection to communicate from the robot controller. The NXT's sensor port 4 can be used as a high-speed connection which supports RS-485, so the command is named "InitRS485()". On the VEX platform, this command will setup one of the user controlled serial ports to communicate with the wireless radio.

The second command, "SendString", is used to send a string of data using the wireless radio to any other device that may be configured to be listening on the same network. This SendString command is used to broadcast out to everyone, rather than to a specific robot. Future implementations of SendString will have the ability to send to a specific robot or to groups of robots. Because ROBOTC uses standard C-Style commands, you will be able to send strings with more data (such as numbers and parameters) using various string manipulation commands found in the C-Language. The third command, "ReceiveString", is used to receive a string of data using the wireless radio from any other device that is configured to be listening on the same network. The ReceiveString command will loop internally until it receives data, although a timeout can be specified to have it continue on in the program if data isn't received within a certain period of time.

Together, these two snippets of code provide the basic framework for multi-robot communication. Also, it is common in multi-robot communication for one robot to wait until a specific message is received, and we provided a code sample of a function that does it. At the end of the introductory lessons, we created a list of programming challenged designed to give students practice having robots share and interpret messages.

MULTI-ROBOT LESSONS TESTED WITH HIGH-SCHOOL TEACHERS

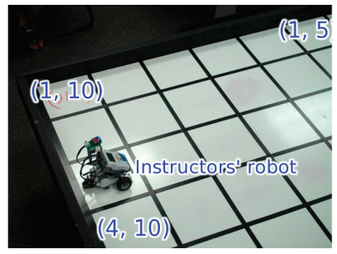
The initial Multi-Robot lesson development occurred during spring of 2011. The development team worked on the curricular materials over a three month period internally and then piloted the materials with a group of CS Undergraduate Women who visited CMU as part of an "Opportunities for Undergraduate Research in Computer Science" program. This section highlights a robotic challenge given to over 160 teachers who attended CMU Robotics Academy's one-week summer classes. The multi-robot programming portion of training started the fourth day of a five day program. The fourth day proved to be ideal because by then teachers had learned basic programming concepts like wait-states, loops, conditional statements, variables, functions, and how to pass parameters. These concepts are foundational and necessary to implement multi-robot communications concepts.

The robotic challenge followed a series of lessons designed to explain the various parts of multi-robot communications. Each lesson consisted of several short exercises (send and receive messages) designed to provide the new learners with the confidence they would need to complete the overall challenge. The teachers were grouped in pairs to solve the problems; each pair had two Xbee enabled robots to allow messaging. The labs were the same labs that we developed during the spring: Setup, Basic Communications, Sending and Interpreting Instructions, and Interpreting and Formatting Data.

On Friday, the teachers participated in a class-wide exercise that involved all the teacher groups and showcased "broadcast communications" between 7 robots.

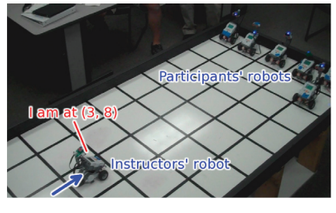
The instructors programmed the primary robot which acted as the primary waypoint, and all of the other robots had to translate that information and decide on a location to "surround" the primary robot. In order to facilitate this, the first lesson that we taught was the concept of operating in a common world. All 7 robots were placed in a grid environment, where black lines were used to demarcate the grid of the world. Using a grid system, we could emulate a Cartesian coordinate system of the robot's world to form the basis of the common language between the robots. The grid-based world allowed the robots to "know where they were" based on their initial start positions and orientations, we used wheel encoder feedback to localize their position on the board at any given time.

The instructors' robot autonomously moved from the initial starting position to another randomly chosen location and communicated its coordinates to the other



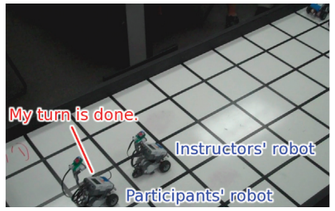
robots. Thanks to the common language (variable names and functions) created by all the teachers as a group, every robot understood the message and knew where the instructors' robot was located.

Another multi-robot concept that was taught was "turn-taking" among a team of robots. Because there are 6 robots that need to navigate from their starting position to their final destination, the teachers needed to create a predetermined order so that robots would know when to move. Once the instructors' robot sent its location to the other robots, each

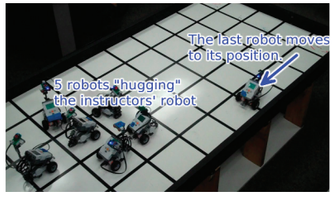


robot would store the position and then the first robot in the predetermined order would start moving. Once that robot reached its destination, it would send a message to the rest of the team informing them that its movement was complete and where it was located. The next robot would then start its turn and move to its position; each robot needed to store the message, and based on the space left determine where it was to go.

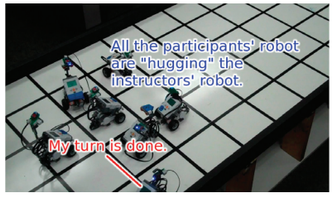
The remaining robots programmed by the teachers then moved to their respective positions around the instructors' robot, one after another by taking turns to move and communicate.

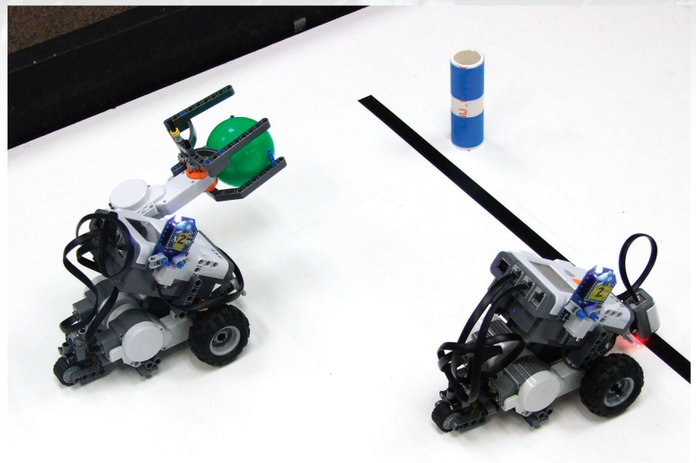


When the last robot reached its desired position, it also announced that its turn was over, and the robots now utilized another concept of the lessons – coordinated actions. In this case, the teacher's robots were programmed to do a dance, all at the same time.



In the end, this multi-robot communications activity proved to teach many basic multi-robot communications concepts. Teachers started with a code base that allowed them to send and store locations; this was necessary due to the time constraints. Each team had to create their own encoder-based movement and path





planning code and to apply multi-robot communications code that allowed them to send and receive messages, take turns, store variables, and autonomously path plan based on prior robots messages. We conducted pre and post surveys of teachers involved in the program. The survey's goal was to help us to learn more about the appropriateness of the level of the challenge, the curriculum's ability to work in the teacher's classroom, and the teacher's perception of the importance of teaching multi-robot communication in a robotics classroom. The survey results were encouraging, Darlene Cook, an elementary school teacher from Austin Texas said "I thought the multi-robot challenge and technology was awesome. It was a week of intense information. A little overwhelming at times for a technology challenged person. However, I got it! It was fast pace. At times when we achieved the goal or challenge, there was another challenge immediately following. There wasn't enough time for the previous challenge to soak in and/or to reflect and think about how we achieved it. However, the info was superb!"

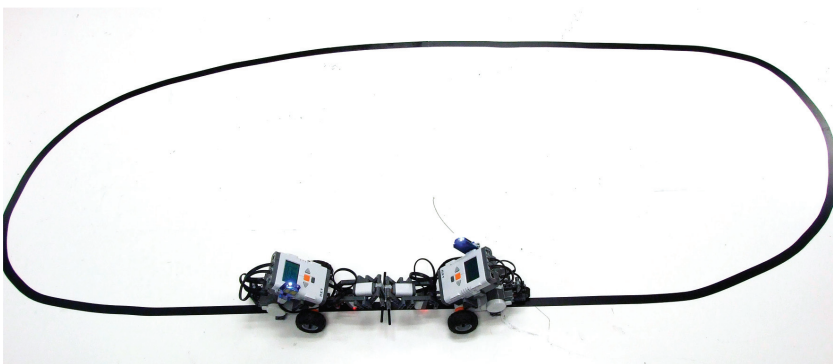
MULTI-ROBOT CURRICULUM EXTENSIONS

In the spring of 2011 Carnegie Mellon received a request from Robert Avanzato, an engineering faculty member at Penn State Abington near Philadelphia. Avanzato was looking to join our research staff for the fall term as part of his sabbatical. CMU was pleased to accommodate Avanzato who has a long history of helping to organize and coordinate middle school through college level robotics competitions at the Penn State Abington campus. Avanzato believes, "These types of events are especially important in urban areas such as the Philadelphia region to promote interest in computer science and STEM." Avanzato is leading the development of formalizing a set of laboratory exercises that build on and provide scaffolding for the initial lessons developed by CMU. He plans to integrate the laboratories and projects into his undergraduate introductory robotics course at Penn State Abington in the spring of 2012. He believes that the combination of the ROBOTC development platform and the inexpensive LEGO and VEX robots can be used to teach introductory concepts at all levels. The technology offers accessibility to middle and high school students, but also offers enough sophistication to be useful in a college level introductory robotics classroom.

In this next section we will describe a series of multi-robot challenges that Avanzato is collaboratively developing with staff here at the Robotics Academy. All lessons will be posted for free access at our website. The lessons assume that the user has a basic understanding of ROBOTC and are designed to develop multi-robot communications skills and techniques starting from the beginners level. When completed the hope is that the lessons are scaffolded in ways and include enough information so that a motivated student can teach themselves, or they can be used by a practicing teacher for use in their classroom. Each lesson includes: a project description, the hardware and soft-

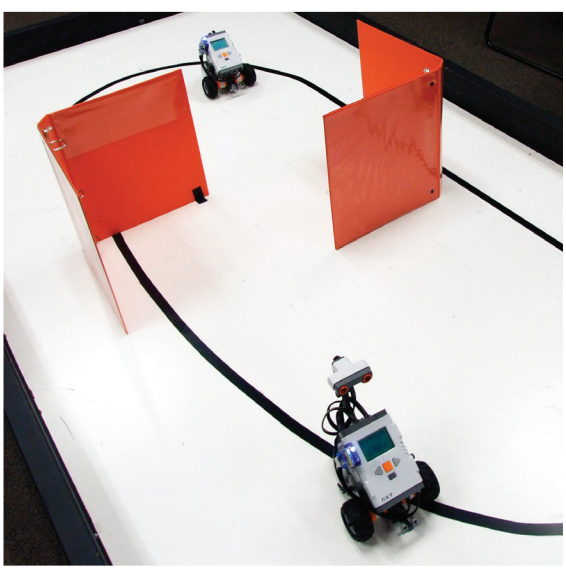
ware required, fully commented working code, teacher notes, the lesson setup, pictures and videos of working solutions, and multiple extension activities for students looking for harder challenges.

Mirror Robot Lab. In this lab, two robots are placed in position with the same starting point facing forward. A ball is positioned on a black line at an unknown distance in front of both robots. The leader robot has a light sensor and the follower robot has only a gripper (no sensors). The leader robot will drive forward and detect the line



then transmit this distance to the follower robot. Using the data received from the leader robot, the follower robot will then be able to drive up and collect the ball using its gripper.

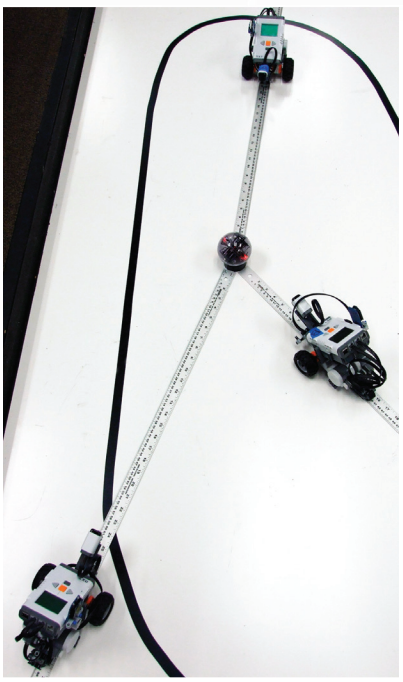
Measurement Lab. Two robots are equipped with a light sensor for line following, encoders to measure distance, and a touch sensor to detect a bump.



The robots start at opposite ends of the path and use line-following to navigate around a closed path. The robots use encoders to measure the distance they have traveled. When the robots collide, they transmit their measured distances (exchange data) and calculate and display the total perimeter of the path. This multi-robot project demonstrates how robots can share a task and combine results to improve efficiency and complete a task more quickly.

The Leader Follower Lab. This project demonstrates communication and coordination between two NXT robots although the lab can be accomplished using either VEX or the Arduino platform. The NXT leader robot uses a sonar sensor to navigate an obstacle course. While the robot is moving through the maze it is also storing data that it will transmit to the follower robot. Once the leader robot finishes the course it sends a message back to the NXT follower robot. The follower robot represents a specialized robot such as a payload-carrying robot. The follower robot is required to navigate the maze based on the communication from the leader, scout robot. This multi-robot project demonstrates how one robot can share sensor data with another robot to accomplish navigation.

Proximity Lab. In another lab, three NXT robots are each equipped with IR sensors and are facing an IR emitting ball. The three robots are positioned at different distances relative to the ball and the challenge is to have the robot team decide which robot is closest and have that robot closest approach the ball. In this activity,



the team leader robot first queries each robot to wirelessly report its distance from the ball. The team leader robot then determines the minimum distance, and transmits a command to have the appropriate robot move towards the ball. This lab demonstrates intelligent decision making based on feedback.

MOVING FORWARD

This project has made significant progress. Our goal was to develop an inexpensive system for education that can be used to teach basic multi-robot communications concepts. It is critical that we give students practice thinking about multi-agent communication in the networked world that they grow up in. To learn more about the project log into the Computer Science Student Network at www.cs2n.org where all of the lessons are posted for free.

Links
Carnegie Mellon Robotics Academy,
www.education.rec.ri.cmu.edu, (412) 681-7160

Computer Science Student Network,
www.cs2n.org

For more information, please see our source guide on page 89.