Autonomous Robots 18, 1–25, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

The Robotic Autonomy Mobile Robotics Course: *Robot Design, Curriculum Design and Educational Assessment*

ILLAH R. NOURBAKHSH
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

KEVIN CROWLEY
Learning Research and Development Ctr, Univ. of Pittsburgh, Pittsburgh, PA 15260, USA

AJINKYA BHAVE AND EMILY HAMNER
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

THOMAS HSIU
Gogoco, LLC. Sunnyvale, CA 94086, USA

ANDRES PEREZ-BERGQUIST
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

STEVE RICHARDS
Acroname, Inc. Boulder, CO 80303, USA

KATIE WILKINSON
Learning Research and Development Ctr, Univ. of Pittsburgh, Pittsburgh, PA 15260, USA

UNCORRECTED PROOF

Abstract. Robotic Autonomy is a seven-week, hands-on introduction to robotics designed for high school students. The course presents a broad survey of robotics, beginning with mechanism and electronics and ending with robot behavior, navigation and remote teleoperation. During the summer of 2002, Robotic Autonomy was taught to twenty eight students at Carnegie Mellon West in cooperation with NASA/Ames (Moffett Field, CA). The educational robot and course curriculum were the result of a ground-up design effort chartered to develop an effective and low-cost robot for secondary level education and home use. Cooperation between Carnegie Mellon's Robotics Institute, Gogoco, LLC. and Acroname Inc. yielded notable innovations including a fast-build robot construction kit, indoor/outdoor terrainability, CMOS vision-centered sensing, back-EMF motor speed control and a Java-based robot programming interface. In conjunction with robot and curriculum design, the authors at the Robotics Institute and the University of Pittsburgh's Learning Research and Development Center planned a methodology for evaluating the educational efficacy of Robotic Autonomy, implementing both formative and summative evaluations of progress as well as an in-depth, one week ethnography to identify micro-genetic mechanisms of learning that would inform the broader evaluation. This article describes the robot and curriculum design processes and then the educational analysis methodology and statistically significant results, demonstrating the positive impact of Robotic Autonomy on student learning well beyond the boundaries of specific technical concepts in robotics.

Keywords:

1. Introduction

Robots have been playing an active role in education since the advent of the LOGO Turtle (Papert and Harel, 1991). Both as project foci in laboratory coursework and as team challenges in national contests, the process of designing, building and programming robots has served to excite students across a broad age range. The current field of robotic educational endeavors is extremely large and diverse; see Fong et al. (2002) and Druin and Hendler (2000) for an overview.

We have had two primary goals in designing and executing a new robotics course. First, we planned to explicitly evaluate the educational impact of robotics on secondary level students. We were particularly interested in quantifying lessons learned in service of robotics that are broadly applicable to learning in general. Second, we hoped to collect data covering a far longer span of time than can be afforded based on a weekend robotics course. Not only would the planned course need to fill a summer; but the students should be able to continue their explorations at home after course completion.

To enable our basic goal—the educational assessment of a long-term course of study in robotics—the authors and others developed, taught and evaluated *Robotic Autonomy*, a seven-week introductory hands-on robotics course as part of Carnegie Mellon West's NASA-Ames campus in Mountain View, California (RASC, 2003). The research surrounding this effort included robot design, curriculum design and ongoing, long-term educational evaluation. Although we and other authors recognize and study the role of robotics in education (Beer et al., 1999; Druin, 2000; Kumar and Meeden, 1998; Murphy, 2000; Nourbakhsh, 2000a, 2000b; Wolz, 2000), this work is notable in that all aspects of the robot mechanism, electronics, software and educational curriculum were subject to ground-up, coordinated design. A total of 30 Trikebot robots were built and used during this program. They continue to be used by graduates of the course at home.

This article begins with a brief overview of the Robotic Autonomy curriculum, then presents the educational robot design process in Section 3, including mechanical considerations, control electronics and the student robot programming interface. Section 4 then presents the educational analysis methodology in detail. A discussionof results follows, with statistically significant learning demonstrated over a number of coded themes, including Teamwork and Problem Solving, as well as an analysis of gender differences. The results strongly support the contention that robotics curriculum not only meets specific instructional goals but can also provide meaningful student engagement for general interest, skills and confidence for promoting future success in technology education.

UNCORRECTED PROOF

2. Course Overview

A sufficiently competent mobile robot is not available commercially at a reasonable price for long-term student robot interaction. Thus short-term robotic educational efforts often turn to Lego building blocks, usually designing curriculum both around robot morphology and construction as well as robot programming and interaction (Stein, 2002; Wolz, 2000). Another successful approach has been the integration of research robots and field robot prototypes into curriculum, where time with the robot is rare and therefore valuable (Coppin et al., 1999, 2002; Maxwell and Meeden, 2000). We were particularly interested in focusing on a course that would concentrate on robot behavior and robot algorithm rather than robot morphology. In order to provide every graduate of Robotic Autonomy with such a rich, programmable robot that would be robust to hundreds of hours of use, we chose to design and produce a new educational robot (Hsiu et al., 2003).

Robotic Autonomy was taught over a seven-week period in the summer of 2002 at the Carnegie Mellon West campus, located within NASA/Ames Research Center (Mountain View, California). The top-level goal for this course was straightforward: to provide selected high school students with an immersive exploration of mobile robotics using leading-edge technologies. Course graduation was intended to mark, not the completion of these educational activities, but a launching point: every student would take home a robust, programmable mobile robot system for continued exploration for months and years. Although robotics would be the focus of this curriculum, we hoped that lessons learned would encompass important concepts reaching well beyond just robotics.

2.1. Organization

The Robotic Autonomy course was aimed at students entering their senior year of high school, and specified one prerequisite: the successful completion of any introductory programming course. Following the application and acceptance process, student composition ultimately included 18 students attending under full scholarship and 10 paying students (see Fig. 1). The scholarship students were from various underprivileged backgrounds, and were primarily Hispanic. The course was comprised of 8 girls and 20 boys.

The course structure depended primarily on teamwork. Principles governing effective teamwork were explicitly discussed, as shown by the curriculum below. Students self-organized into teams of three during the first day, with the constraint that single-gender teams be created whenever possible. Based on previous experience teaching robotics courses at the undergraduate level, we felt that single-gender female teams would be more likely to encourage active participation by all members of the team, especially in the case of shy female students. Throughout the seven weeks, all team members shared joint responsibility to meet course challenges, with all members of the team receiving the same grade on each week's activities. In order to tackle weekly assignments, each team used just one of their three robots in early weeks, but by the first month's end made use of all three team robots in cooperative robot team exercises.

The course was taught by a single principal instructor and four teaching assistants. The teaching assistants ranged from graduate to undergraduate students. There were also two instructors who took the course as a way of becoming trained to teach it in the future. They provided some teaching assistance as well. One of the teaching assistants was female, and the rest of the instruction team was male.

2.2. Curriculum

Robot-based curricula is used today across diverse age groups and with a broad variety of purposes. At the informal learning extreme, after-school programs based on annual contests have become popular with the advent of national contests including Botball (2004) and Stein (2002) and US FIRST (Hobson, 2000; US FIRST, 2004; Yim et al., 2000). These contests demand that teams of students together design, fabricate and iterate to present robotic solutions that often perform in head-to-head exercises against competitors. Botball and US FIRST are foremost team-based physical design challenges, and there is evidence that such competitive design exercises draw upon cross-disciplinary skills in a powerful manner (Manseur,

UNCORRECTED PROOF

Figure 1. The Robotic Autonomy 2002 students.

2000). Yet the competitive aspect of these contests can be troubling in terms of gender disparity; for instance (Milto et al., 2002) utilizes student surveys to conclude that males find the contest format more interesting than women do.

Nevertheless contests have permeated a significant proportion of robotics curricula at all age levels. Kolberg and Orlev (2001) presents cumulative weekly exercises designed for 5th and 6th grade levels, building incrementally sophisticated behaviors and culminating in a final contest. Numerous courses at the undergraduate level, including (Archibald and Beard, 2004) present robotics in the framework of a simplified Robot Soccer contest, with students working in teams to design and build soccer-playing robots that then compete for glory. At the Swiss Technical Universities of EPFL and ETHZ (Siegwart, 2001) has documented robot design and fabrication courses for undergraduates also culminating in a final contest that changes annually.

A number of robot courses bring existing robot platforms to bear, shifting the intellectual emphasis from robot design to robot algorithm design, as in the undergraduate setting (Maxwell and Meeden, 2000; Billard and Hayes, 1997; Billard, 2003; Nourbakhsh, 2000b; Murphy, 2000; Kumar and Meeden, 1998). For example since 1994 (Nourbakhsh, 2000b) has taught a mobile robot programming course at Stanford and at Carnegie Mellon University in which students incrementally program Nomad mobile robots, culminating in a team-based final competition.

Common to virtually all of the robotics curricula surveyed is the focus on challenge-based, hands-on and bottom-up learning. The bottom-up approach, which maximizes exploration and self-discovery, is inspired by Constructionism (Papert and Harel, 1991). The curriculum designed for Robotic Autonomy is most similar in spirit to that of Kolberg and Orlev (2001), presenting a series of incremental exercises at a weekly period. However, Robotic Autonomy is a full-time, eight hour per day course, demanding a rich set of exercises that can stimulate students of varying robot aptitude. We wished to focus on robot algorithm development rather than physical robot design specifically because of the paucity of such coursework. Our hope was that sufficient attention to robot algorithms, mated with a highly competent robot platform, would lead to truly sophisticated robot projects by course's end. We also wished to avoid the potential gender bias of a contest-based focus, and indeed wanted to ensure that self-motivation and inquiry would be paramount because of the important of such skills for home exploration with the Trikebot. Thus our curriculum diverges from that of design-centered coursework such as Archibald and Beard (2004) and Siegwart (2001) and is more similar to the incremental programming curricula of Kumar and Meeden (1998), Nourbakhsh (2000b) and Kolberg and Orlev (2001).

Robotic Autonomy is designed around a one-week unit length, with an intra-week repeating structure to provide a familiar weekly trajectory. Each Monday and Tuesday is spent presenting new material and posing a new, open-ended challenge for each team to tackle. Wednesday is Challenge Day, including extensive testing of the challenge submissions of every team. In addition, a portion of this day is set aside for each team to document their weekly solutions, including source code, prose, pictures and videos to be placed on a specially configured team website. On Thursday morning, teams receive the details of an end-of-week contest, which apply the concepts learned for that week's challenge in an enjoyable and competitive format. Thus Thursday is spent preparing carefully for the next day's contest. Friday is Contest Day, with invited guests (parents, administrators and visitors) watching and cheering as team robots engage in games such as line-following races, bomb defusing contests, musical chairs, *et cetera* (see Fig. 2). While contests are thus not completely eliminated, Friday contests are designed carefully as significantly simpler tasks than the challenges due two days earlier, on Wednesdays. As the end of the course approached, Friday contests were replaced by team-designed Exhibitions, making the full transition from mediated learning to self-directed exploration.

In summary, new concepts are largely presented early in the week, with the most difficult bar set by the Wednesday challenge. Following this intellectual apex, the Friday contest offers a chance for students to reuse lessons learned that week in an enjoyable and playfully competitive atmosphere. In addition to the direct lectures and challenges, weekly guest speakers are brought in on Mondays and Tuesdays to provide one-hour discussions on their areas of expertise. These speakers provide both an outside perspective on robotics and a window into the lifestyle of career roboticists.

The outline below shows the challenges and contests associated with each week of Robotic Autonomy, together with the underlying concepts learned in that week. Also noted are prepared speeches and guest speakers' topics. The complete curriculum for Robotic

UNCORRECTED PROOF

Figure 2. Several examples of Friday contests: Robot Theater (left); Bomb Squad (center); Outdoor Jogging (right).

UNCORRECTED PROOF

Autonomy as well as all student web sites are available for download (RASC, 2003).

Week 1

Challenge
- Stand-alone Java timer and calculator programs
- Build Trikebots with unique outfits

Contest
- Capture the Flag (remote-control operation)

Concepts
- Using hand tools
- Using buttons and textfields in the Java GUI: javac and java
- How joints, servos, and motors work
- Kinematics: the Instantaneous Center of Rotation
- Introduction to electronics: batteries, power, PWM motor control, servos, wiring, plugs, connectors, polarity
- Using the iPAQ to directly test the Trikebot
- Using the Java Trikebot UI for direct motor control
- How to use the iPAQ: network configuration

Talks
- Thomas Hsiu: talk on Mechanical design considerations

Week 2

Challenge
- Ded-reckoning primitives for timed robot moves
- Autonomous, choreographed, robot dance

Contest
- Robot theater (choreographed autonomy)
- Robot soccer (button-based remote operation only)

Concepts
- Physical robot sources of error: wheel-floor interactions, backlash, slippage
- Controlling robot speed and position using time
- Testing and tuning ded-reckoning, servo and speed calibration
- Trapezoidal speed profiles
- Programming the Trikebot
- Testing ded-reckoning error using geometric scripted motions
- Sequential scripted motions
- Website documentation
- iPAQ connection diagnostics: problem solving without instructor assistance
- Elements of a good robotic theater performance
- Designing and implementing functionality for GUI buttons in teleoperation

Talks
- Thomas Hsiu: talk on robotics in special effects & Hollywood

Week 3

Challenge
- Touch-free racing (signaling to the robot via the rangefinder)
- Autonomous wandering and exploring

Contest
- Escape (crossing an obstacle field)
- Musical chairs (mixed autonomous wander and remote operation)

Concepts
- Downloading firmware to the iPAQs without instructor assistance
- The role of sensing in autonomy
- Survey of rangefinding sensors
- Accessing the Sharp Rangefinder readings on the Trikebot using Java
- Creating sensor-driven robot control code

- Open sourcing robot software: how to make a code repository and why
- Adjustable autonomy: mixing autonomy with remote control
- Proprioception in humans and achieving this in robots
- Back-EMF based DC Motor speed sensing: principles and execution
- Motor acceleration and current: relationships
- Teamwork: evaluating the effectiveness of teams; communication; best practices
- Techniques for maintaining a sense of direction: sensing versus physical manipulation

Talks

- Tom Lauwers: talk on Starting a Robotics Club of your own
- Illah Nourbakhsh: talk on Innovative Mechanism: Gyrover, Bowleg hopper

Week 4

Challenge

- Martian Explorer (Video-based, high-latency teleoperation)
- Go Home (Teleoperation-based localization)
- Visual tracking challenge (maximum tracking distance)

Contest

- Bomb Squad (team-based bomb disposal)
- Outdoor visual control (vision-based cues for head-to-head races)

Concepts

- Human vision: anatomy, color sensing and object recognition in the brain
- CMOS-based vision sensors: background photonics and limitations
- CMUcam: electronics overview and introduction
- Attaching and using CMUcam (hardware, EE, UI)
- Intelligent Teleoperation: research survey
- Color tracking with CMUcam: pitfalls and representations of color
- Autogain, auto white-balance, and other visual feedback loops in CMUcam

Talks

- Anthony Rowe (CMUcam inventor): talk on designing and using the CMUcam

Week 5

Challenge

- Duckling (autonomous robot visual tracking and following)
- Outdoor jogging (high-speed visual following)

Contest

- Navigator (autonomous navigation using visual fiducials)
- Robot exhibition design

Concepts

- Designing your own team exhibition: starting the process
- Active visual tracking and following
- Picking good colors for CMUcam: using DumpFrame for diagnostics
- Providing tracking information as feedback to neck servos to center head
- Considerations for designing hardware and software for 1 robot to follow another
- Videotaping your robots: open-source value; pointers
- Localization and navigation: designing navigating systems; landmarks; heading, termination
- Adjustable autonomy; modes of interaction with semiautonomous navigators

Week 6

Challenge

- Mobot (a simple line-following race)
- Security Patrol (motion/intruder detection)

Contest

- Mazeworld (mixed teleoperation and autonomous line-following)
- Exhibition 2.0 (second chance to refine and practice custom exhibit)

Concepts

- Detecting and performing line-following: vision and feedback control, convergent control, oscillations, considerations for vision outdoors
- Robotics and Social Responsibility: broad discussion of technology and impact
- Control and state: Zero-state functional systems; reactive systems; state machines
- Navigation as map representation and state representation

Talks

- Liam Pedersen: talk on Robotics at NASA and NOMAD in Antarctica
- Jonathan Buford: talk on muscle wire on a robotic spider

Week 7

Challenge

- Final exhibition for parents, educators and invitees

UNCORRECTED PROOF

Concepts
Navigation: path planning techniques commonly used, a survey
Public speaking and presentations: pointers and tips
Robot demonstrations and exhibit design
Talks
Illah Nourbakhsh: talk on The Personal Rover Project
Steve Richards: talk on Acroname's robots; overcoming ded-reckoning error

3. Robot Design

The robot design goals are informed by the intended target audience for the educational course: high school students between their junior and senior year. Prerequisites include basic mechanical dexterity (e.g. simple assembly and fabrication) and knowledge of a programming language (e.g. *Introduction to Programming*). Significantly, each student would be slated to take a Trikebot home to program and use at will after course completion. Thus, the Trikebot would need to be designed not only for the beginning robotics student but for the continuing, sophisticated user. In other words the robot would need to have sufficient expressiveness and capability to serve as an educational and exploratory tool beyond the confines of a seven-week course.

Design and production of a new educational robot, the Trikebot, was a costly and time-consuming step in the execution of Robotic Autonomy. After surveying state-of-art educational robot hardware alternatives, we concluded that this design and fabrication process would yield a significantly more desirable solution; the reasons behind this decision are worth amplification.

The Robotic Autonomy course was an intensive but short-term course, providing students with seven weeks in a formal learning environment with the hope of sparking self-directed further exploration into *robot algorithms* with the robotic platforms after graduation. To this end six qualities were paramount in the selection of an educational robot for Robotic Autonomy.

1. *Mechanical Empowerment.* Even in an algorithm-focused course, a broad introduction to robotics demands inclusion of electromechanical aspects of robot design as well as robot programming. Furthermore, because students would keep the robots following graduation, they must be sufficiently empowered to be able to repair their robots in the near-certain case of eventual physical malfunction and breakage. Our strategy for meeting this need is to ensure that the educational robot arrives in kit form: students construct each robot, which has solely off-the-shelf life-limited parts, and are thereafter able to replace such parts.
2. *Behavioral Richness.* The desire for open-ended, project based exploration leads to a requirement for sufficiently rich robot-world and human-robot interaction as to engage students during weeks and months of programming. This relatively vague requirement is made concrete by way of two hard constraints: the robot platform must have visual competence (i.e. ability to track fiducials, follow lines, detect visual environmental changes) and must be richly programmable using a high-level programming language (e.g. C++, Java, etc.).
3. *Robustness.* As is the case with all electromechanical course products, an educational robot must be robust to the numerous accidents which occur with great frequency in the initial few days of a mobile robotics course. Furthermore, because we aimed for student projects for which robots may move for an hour or more autonomously, mechanical robustness should extend temporally over more than a few minute of run-time.
4. *Maneuverability.* In keeping with (2) Behavioral Richness and to engage and challenge students in their own natural world, we stipulate that the robotic platform should be capable of maneuvering at fast walking speed in both indoor and outdoor environments, including sidewalks, short grass and gravel. Such breadth of application environments opens the field in terms of team and individual robot programming challenges throughout classroom areas and the field.
5. *Wireless Scaling.* Robotic Autonomy planned for up to 40 students at one time, and therefore a hard constraint is that all wireless robot control be scalable to at least 40 simultaneous robots. This requirement alleviates the unnecessary logistical burden of time-sharing robot execution among multiple teams and robots.
6. *Price Point.* Given a fixed budget and the desire to award every Robotic Autonomy graduate with a high-competence mobile robot, we established a hard limit of $2,000 total cost per robot platform, with a bias for the least expensive possible solution.

A number of existing robotic platforms satisfy a subset of the criteria noted above. Table 1 provides comparison data for several popular educational robot

UNCORRECTED PROOF

Table 1. A comparison of educational robot platforms and kits in view of six Robotic Autonomy robot constraints. Rows include the basic Lego Mindstorms kit (Martin et al., 2000); the Lego kit augmented by a Handyboard microprocessor (Botball, 2004); Amigobot (ActivMedia, 2004); Khepera (K-Team, 2004); ER1 (Evolution Robotics, 2004); Garcia (Acroname, 2003); and the authors' Trikebot solution.

	Mechanism	Behavior	Robustness	Maneuvering	Scalability	Price point
Lego RCX	Y	N	N	N	N	$200
Lego-Handy	Y	Y	N	N	N	$1000
Amigobot	N	Y	Y	N	N	$2500
Khepera	N	Y	Y	N	N	$2000
ER1	Y	Y	Y	N	Y	$300
Garcia	N	Y	Y	N	Y	$1700
Trikebot	Y	Y	Y	Y	Y	$1200

UNCORRECTED PROOF

packages. Kits such as the Lego Mindstorm are the basis of a number of successful robot courses (Fagin, 2003; Gage and Murphy, 2003; Kumar, 2001; Schumacher et al., 2001; Wang, 2001; Wang and Wang, 2001). When used in conjunction with the Lego RCX, such a solution satisfies our price point and mechanical empowerment constraints only. Lack of RAM and ROM space on the RCX obviates rich programmability, as does a lack of vision-based on-board sensing. Addition of a more sophisticated microprocessor such as the Handyboard (Botball, 2004; Nagchaudhuri et al., 2002) allows for more sophisticated sensors, actuators and algorithms.

A second popular approach even eschews the mechanical modularity of Lego, preferring to empower students to fabricate robots of their own design using metal, wood, plastic and other rapid prototyping materials (Heer et al., 2002; Siegwart, 2001). While such robot design projects have real educational benefits, such a focus on robot construction is often at the expense of time spent exploring sophisticated robot programming.

Existing, commercially available educational robots, as shown in Table 1, satisfy only a subset of our constraints (K-Team, 2004; ActivMedia, 2004; Evolution Robotics, 2004; Acroname, 2003). To be fair, such commercial solutions achieve far higher levels of overall robustness than the Trikebot; however, when failures do occur, such systems can be extremely difficult to repair in the home due to their lack of off-the-shelf parts and their mechanical complexity. In terms of price point, existing commercial products appear to be more expensive than Trikebot. This is surprising given that such commercial products are created in higher volumes than the Trikebot. There are two reasons for this disparity. First commercial products must include sufficient markup for long-term viability while Trikebot's price essentially represents Cost of Manufacture. Second the Trikebot benefits in price from significant parts donations and price reductions by commercial vendors. Finally, in terms of the maneuverability feature, the level of terrainability desired for Robotic Autonomy is not available in existing products to our knowledge.

In comparison to existing robot platforms the Trikebot occupies a point in design space that is particularly well-suited to the nature of Robotic Autonomy. The chassis consists of durable plastic pieces fitted together via a slot and tab design. All degrees of freedom are actuated by off-the-shelf hobby servomotors available on-line. The on-board IPaq PDA is also off-the-shelf, and provides scalable 802.11b wireless connectivity to an off-board portable laptop, which itself enables high-level programming in Java. The CMUcam vision sensor, capable of line following and object tracking (via color) paves the way for relatively engaging and rewarding robot behavior, all of which can be executed both indoors and outdoors because of the Trikebot's large diameter wheels and ground clearance.

Sections 3.1, 3.2 and 3.3 describe design objectives and solutions for robot mechanism, control electronics and the student programming interface respectively.

3.1. Robot Mechanism

The Trikebot chassis has three primary functions. It is a camera platform for the CMUcam (Rowe et al., 2002), it provides mobility over a variety of indoor and outdoor terrains, and it can carry a relatively large payload. In addition to these functions, the Trikebot is meant to be assembled and serviced by students with few specialized tools. Most of the related design decisions were driven by these requirements.

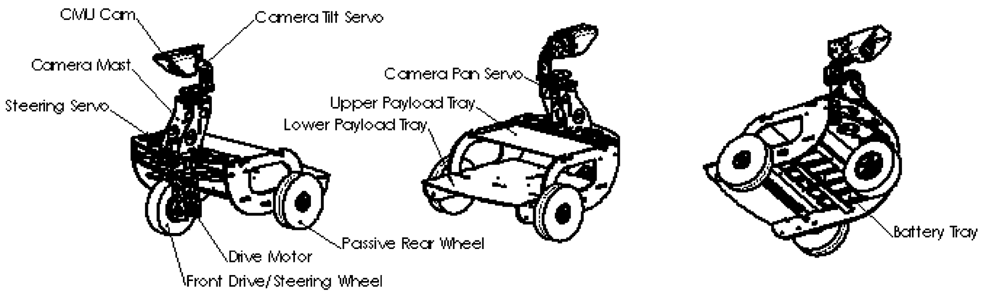


Figure 3. Trikebot chassis overview.

UNCORRECTED PROOF

As a camera platform, our goal was to place the camera at least 18 inches above the ground plane. This was part of a decision to make the Trikebot a floor-based robot with which students could interact more dynamically than a smaller table-top size robot. Putting the camera relatively high off the ground both gives the camera a wider effective field of view and encourages students to interact with the robot at an eye-to-eye level. Camera placement is important both for teleoperation modes of control and for autonomous robot operation. The pan and tilt mechanism is critical for *diagnostic transparency* and affection; it enables the robot to clearly indicate direction of gaze and widens the field of view further (Fong et al., 2003).

Because we expect the Trikebot to operate not only indoors but also on relatively flat outdoor areas such as parking lots, sidewalks and lawns, it must be able to overcome minor obstacles such as electrical cables, door thresholds and gravel. The robot's ground clearance and wheel size enable such locomotion. To facilitate mobility in closed quarters, we required Trikebot to turn in-place within a 24 inch circle. Finally, to encourage student-robot interaction, the top speed of the Trikebot was specified as comparable to a person's medium speed walk, roughly 30 in/sec.

As a worst-case payload requirement, the Trikebot is designed to carry a laptop computer, six 7.2 V Remote Control (RC) car battery packs and various onboard electronics. This payload objective would turn out to be an overestimate primarily because our final architecture enabled an off-board laptop to communicate via 802.11b with the Trikebot, as described in Section 3.2.

Being assembled and maintained by students in a general classroom environment required that the majority of the components of the Trikebot be assembled using simple hand tools and that they be robust enough to handle rough treatment. Of course, cost is always an issue, so appropriate manufacturing techniques were chosen for the quantities of parts used. This dictated the look and feel of the individual components designed.

Together, all of the above objectives influenced the final design of the Trikebot. The proceeding chassis overview is followed by descriptions of how the various elements of the Trikebot chassis meet these objectives.

The Trikebot chassis is a three-wheeled mobile robot base in a tricycle-like configuration, with a single driven steerable wheel and two fixed passive wheels. Its major physical features are a tall camera mast with a pan and tilt mechanism and two large, flat payload areas, one low in the chassis and another smaller shelf above it (Fig. 3). Altogether the Trikebot has 4 control degrees of freedom—drive motor, steering, camera pan and camera tilt.

The *tread width*, or distance between wheel centerlines as viewed from the front or back, is 15.8 inches and the *wheelbase*, or distance between wheel axes as viewed from the side, is 10.9 inches (Fig. 4). The wheels of the Trikebot are 6 inches in diameter, supporting a ground clearance of 2.2 inches. The nominal camera height is 18.3 inches and it can pan approximately $\pm 90^\circ$ and tilt $+90^\circ/-45^\circ$. Overall, the mechanical chassis alone, minus batteries and electronics but including servos and drive motor, weighs approximately 10.5 lbs.

Wheel Configuration. A tricycle configuration with a single driven steering wheel gives the Trikebot very good agility using a single gearmotor as its drivemotor and a single high power servo for steering. The servo can steer the driven wheel through 180° allowing the robot to turn nearly in place, well within a 24 inch circle. This allows the Trikebot to turn completely around within a confined space such as a doorway.

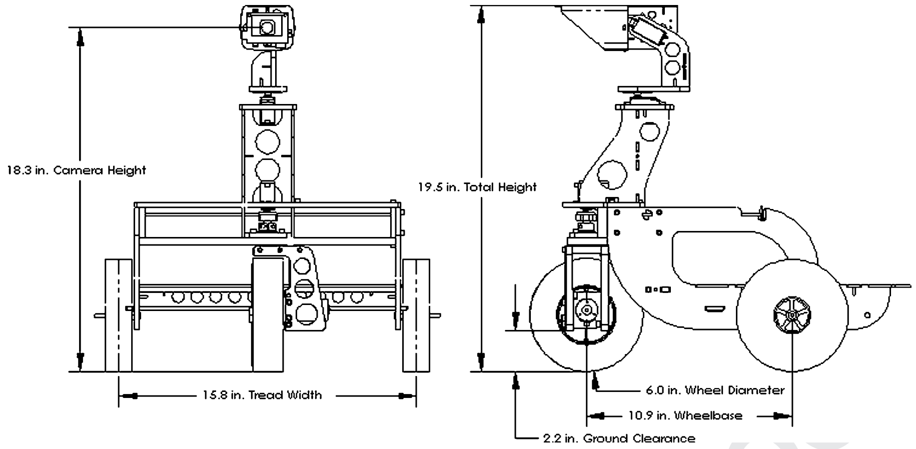


Figure 4. Trikebot chassis dimensions.

UNCORRECTED PROOF

We chose the tricycle design in lieu of the other common three-wheeled configuration, with two driven wheels and one trailing caster wheel, to avoid several problems generated by that configuration. One problem is that the trailing caster wheel can restrict the freedom of movement of the robot in certain situations. For instance, when reversing direction of travel, the action of the caster reversing to trail the direction of motion can force the robot to deviate its course or cause wheel slip in the robot's drive wheels. Furthermore, two driven wheels must match their speeds exactly in order for the robot to travel in a straight line. This generally requires additional motor encoders to achieve sufficient accuracy. The tricycle design eliminates both of these issues.

The single wheel forward arrangement was chosen for agility over obstacles. The driven wheel can more easily grip and climb over an obstacle at slow speeds, subsequently dragging the rear wheels over the obstacle. The tradeoff is less stability during maneuvers at high speeds, but it was expected that most of the Trikebot's maneuvers would not be at full speed.

One final advantage of a three-wheeled design is lowered torsional stress on the chassis. In a four wheeled chassis, a single wheel can be raised above the others when traversing uneven terrain. This causes torsional stress on the chassis which can twist the chassis (and its payload) unless it is strong enough to resist the twisting. A three wheeled chassis undergoes much less twisting, meaning the chassis can be both simpler and lighter.

Wheels. With wheel diameters of 6 inches and a ground clearance of 2.2 inches, the Trikebot can drive over obstacles such as power cords, uneven sidewalks, and even gravel paths. The traction element of the wheels consists of closed-cell foam rubber *tires*. These tires provide adequate stiffness and traction, yet are still light and help absorb shocks. The rear passive wheels and front wheel hub are stock RC model airplane parts and car parts, utilized to minimize cost.

The Drivetrain. The drivetrain consists of a 19.5:1 gearmotor directly coupled to the drive wheel. The gearmotor's output bearings are adequate for the loads expected to be delivered by the Trikebot and direct drive provided the simplest design. Together with a motor clamp and motor support structure, the drive wheel and gearmotor comprise a drive wheel assembly (Fig. 5).

The drive wheel assembly turns about a kingpin which is centered above the center of the drive wheel and mechanically fixed to the drive wheel assembly. By positioning the steering axis directly above the center of the drive wheel, no steering torque is generated when the drive motor is engaged. The kingpin is supported by two sets of ball bearings pressed into the main chassis. These bearings carry the load of the chassis on the drive wheel assembly, allowing the robot to steer

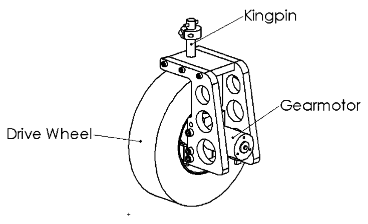


Figure 5. The drive wheel assembly.

with minimal friction. A high-torque RC servo directly drives the kingpin, providing steering control.

Camera Mast and Pan and Tilt. The camera mast incorporates a pan and tilt mechanism and elevates the camera to its desired 18 inch minimum height. The positioning of the mast to the front of the chassis allows the camera to scan slightly in front of the front wheel while looking down. This facilitates activities such as line following or object-in-path detection. The pan and tilt mechanism was also designed to maximize the camera's potential field of view. The camera is centered above the camera's pan axis and the camera's centerline passes through the tilt axis. This simplifies the analysis of the camera's view relative to the robot. One design compromise was to reduce the camera total pan angle from a panoramic 360° to 180°. This allowed both the pan and tilt to be directly controlled by stock RC servos, simplifying the design and reducing costs.

Payload Area. The Payload areas of the Trikebot are positioned low and to the rear of the camera mast in order to place the fully loaded robot's center of gravity as low as possible and roughly 1/3 of the wheelbase behind the front wheel. A low center of gravity maximizes the stability of the Trikebot and placing the center of gravity 1/3 of the way behind the front wheel helps provide traction to the front driven wheel. The battery racks are located below the lower payload tray, again to lower the center of gravity and to provide easy access to the batteries.

The lower payload tray is sized to accommodate a standard laptop computer with the screen closed and the upper payload tray tilts up to allow easier access to the front of the lower tray.

General Construction. Most of the Trikebot chassis is constructed of lasercut acetal (Delrin) sheets (Fig. 6). Laser cutting provides maximum flexibility for the relatively small number of parts produced for this project while being more economical than traditional machining. Aluminum machined parts were used for a few items, such as the drive hubs and motor clamps, but machining was minimized as it costs over ten times to produce comparable parts over lasercutting. However lasercutting has its drawbacks, allowing only cuts perpendicular to flat sheets of material like paper or plastic (or metal for higher power laser cutters). To accommodate this, the Trikebot's parts fit together with tabs and slots, not unlike paper or cardboard models. Self-tapping screws wedged into slots hold the plastic parts together. While these fastening methods increase the design time, they minimize secondary machining operations such as drilling and tapping holes, ultimately saving cost. This also allows most of the Trikebot to be assembled by the students themselves using hand tools. When mechanical repairs or adjustments are needed, the students have been able to perform these tasks themselves. Using rapid manufacturing technologies such as lasercutting, combined with using stock parts such as RC servos and wheels, enables the Trikebot to be produced economically and quickly in the quantities required for this project, while fulfilling the desired design objectives.

3.2. Control Electronics

The role of the control electronics was to create a clean interface between the physical robot layer and the high-level Java programming interface the students would use to program the robot. The electronics abstract away most of the communication overhead, interface control

Figure 6. The unassembled components of one Trikebot; 30 assembled Trikebots (right).

UNCORRECTED PROOF

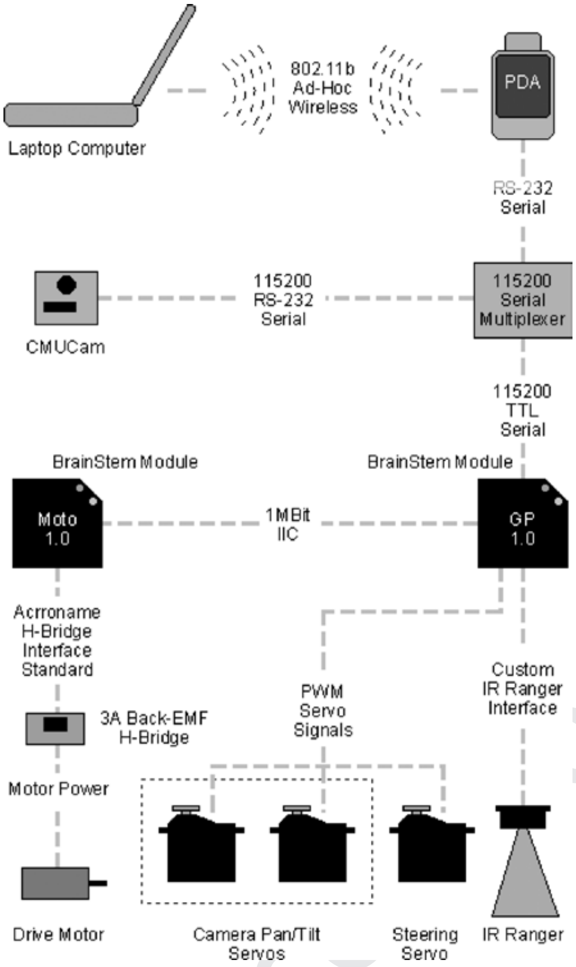


Figure 7. The Trikebot control electronics' connectivity.

UNCORRECTED PROOF

and motion control aspects of the Trikebot. Our solution accomplished this abstraction while allowing flexibility for expansion, lower level control and design modularity.

Figure 7 depicts the connectivity of the Trikebot's control electronics. An iPAQ 3650 serves as the 802.11b wireless link between the robot electronics and the students' laptops. This ARM processor has sufficient power to be the Trikebot's main server computer but lacks an interface for easy programmability by the student. Laptop to iPAQ communication is achieved over TCP/IP, with the resulting serial stream multiplexed between the CMUcam, which provides visual perception services, and the Brainstem network, which provides motion and sensing control.

3.2.1. Brainstem^{TM} Architecture. In the Trikebot, the BrainStem network is primarily a slave controller. The student's laptop performs high-level decision making and sequencing, in turn requesting control outputs and inputs from the Brainstem network using a Java API. The BrainStem architecture offers rich I/O capabilities in slave mode but can also function independently via TEA (tiny embedded application) programs which use ANSI C syntax to run on small virtual machines located within the BrainStem module's controller (Acroname, 2003). Writing TEA programs or setting up reflexes offers more control capacity and can serve as an expansion option for Trikebots. TEA can also offer failsafe behavior handling when the wireless link or iPAQ encounters trouble.

The Trikebot's steering and camera pan/tilt servos are driven by the BrainStem GP 1.0 module. This board also supports the Sharp GP2D02 IR distance ranger. Both of these tasks are managed by the GP 1.0 module which encapsulates the serial clocking of data from the digital IR sensor, dampens the motion input to the servos, and manages the servo ranges and offsets. Once configured, simple commands can be sent to the GP 1.0 module for neck position, steering and distance ranging.

This GP 1.0 module also acts as a serial to I2C router to communicate with the other BrainStem Module, the Moto 1.0 board. This approach allows all commands to be sent to the BrainStem I2C network via a single serial connection. The Moto 1.0 module handles the closed-loop motion control of the Trikebot's motor. This motion control is performed using PWM (pulse width modulation) output to an H-Bridge daughterboard.

3.2.2. Back-EMF Based Speed Control. One unique ability of the H-Bridge and Moto 1.0 module used in the Trikebot is Back-EMF speed measurement. This approach uses the natural characteristics of a spinning motor to derive a feedback voltage that is linearly proportional to the speed of the motor.

Most precision robotics applications use motors with optical or magnetic encoders offering quadrature position sensing. This approach is effective but the combination of the precision encoders and quadrature decoding chips on the motion controller make this approach expensive. Using Back-EMF control allows feedback-based PID speed control while using a simple gearmotor with no encoder.

The basic idea behind Back-EMF speed control is that while a motor is being driven, the H-Bridge windings that actually offer the connection to the drive current for the motor can be "floated" or left disconnected. When this occurs, the induction developed in the windings of the motors quickly collapses and the motor transitions to a generator of current due to the residual inertia in the mechanical drive system. This takes place in only a few milliseconds. Once this transition has occurred, the output voltage from the motor is directly related to the speed of the motor.

The Back-EMF circuit built into the 3A H-Bridge used in the Trikebot's Moto 1.0 board measures the voltage from the motor and converts it to a logic voltage centered at 2.5 volts. When the motor is running full speed in one direction, the voltage drops to ∼0.0 volts and when the motor is running full speed in the other direction, the voltage rises to ∼5.0 volts. This is read by a 10-bit analog input on the Moto 1.0 module and used as the feedback for the PID equation driving the duty cycle of the motor. Once the A/D measurement is taken, the motor is switched back on and driven via the PWM output.

3.2.3. iPAQ Robot Server. The iPAQ ARM-based processor serves as both an 802.11b to serial bridge and a real-time sensorimotor controller on-board the robot. Together with the Brainstem components, this unit completes the on-board electronics of the Trikebot (Fig. 8). There are a number of reasons to avoid placing the student laptop directly onboard the Trikebot. First, reducing the payload requirements enables a longer running time for the robot and reduces the chances of robot damage in the case of collisions. By the same token, the laptop is kept out of harm's way while providing direct diagnostic feedback to the student, even during program execution. Finally, an off-board laptop can serve as a teleoperation input device. Given the NASA collaboration in this project, such teleoperation was particularly relevant for curriculum exercises involving simulation of Mars Rover type activities.

Figure 8. Control electronics located on the Trikebot.

The fundamental problem of removing the laptop and thus the high-level control program from the Trikebot concerns communication latency. Even in the best of cases, roundtrip communication latency via 802.11b can easily exceed 150 ms. Although this is acceptable for high-level commands involving steering and speed decisions for the Trikebot, this is unacceptable for fast-feedback control loops such as visual pan-tilt tracking of moving objects using the Trikebot's CMUcam.

Figure 9 summarizes the functional layers of the iPAQ firmware. Using a checksum-based message-passing protocol, the off-board laptop communicates high-level vision commands and robot I/O commands to the iPAQ. The iPAQ controls the serial multiplexer state and formats and handles dialogue with both the CMUcam and the Brainstem Architecture.

In addition to providing communication services to each downstream electronic device, the iPAQ serves three other functions. When the laptop requests an image dump from CMUcam, the iPAQ acts as an intermediate image buffer to collect and send that information. Because CMUcam can take up to 2 sec. to deliver a complete image at 115200 baud, this process must be asynchronous and thus the image data is transmitted back to the laptop via a dedicated TCP/IP image socket. Second, the iPAQ can serve as a pan-tilt feedback controller, utilizing CMUcam to measure the visual displacement of a tracked object, then commanding the pan and tilt servos via Brainstem to visually center the object being tracked. Once again, this feedback loop must be performed asynchronously and provides feedback to the laptop regarding the tracked object and the Trikebot's neck position using a separate TCP/IP socket. Third, the iPAQ takes advantage of the fact that all three servoed joints in the Trikebot are statically stable to save power. By running separate timers for each joint, the iPAQ is able to power down each servo once it has reached the commanded position (for the steering servo this is plausible only when the robot is not moving). This strategy significantly lowers power requirements when the Trikebot is idling.

3.3. Programming Interface

As the interface between student and robot, the laptop environment is critical for students to learn successfully and enjoyably. One objective is that the environment enable the student to directly control the robot's

UNCORRECTED PROOF

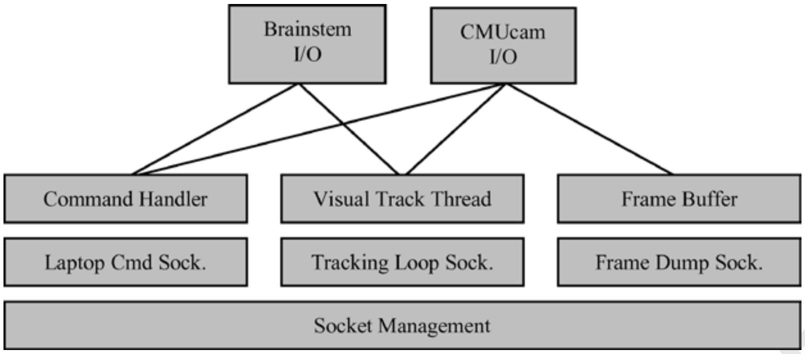


Figure 9. The functional layers of the iPAQ firmware.

UNCORRECTED PROOF

motion (i.e. teleoperation) as easily and quickly as possible "out of the box." A second objective is that, assuming basic knowledge of some programming language, a student should be able to program the Trikebot for autonomous motion with as shallow a learning curve as possible. The goal is thus to rapidly surmount the obstacles of learning machine-specific programming and compilation details, instead devoting the majority of intellectual effort to exploring the space of autonomous and interactive robot behaviors. Finally, closing the loop, the third objective is that the interface provide maximal diagnostic transparency during program execution so that the student is empowered to improve the performance of the Trikebot (Nourbakhsh, 2000c).

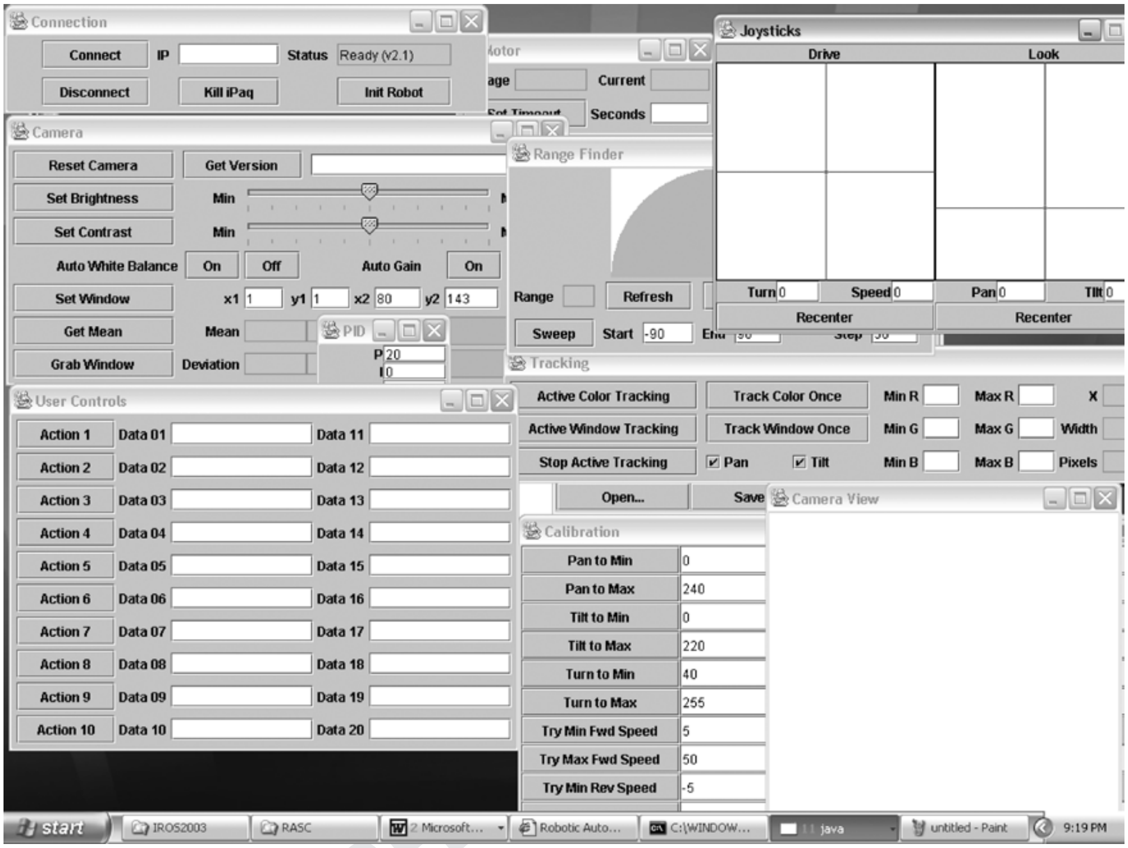
The Trikebot UI is both the teleoperative interface and the program execution and monitoring interface and is described in Section 3.3.1. The subsequent section describes the programming interface, through which the students write JAVA code to control the Trikebot interactively and autonomously.

3.3.1. Control and Diagnostic UI. The Trikebot UI, shown in Fig. 10, enables direct teleoperation of the Trikebot. Direct teleoperation is not only important as a novelty; it is critical to the ongoing diagnostic process of students being able to shift their point of view to that of the robot. By dumping images from the Trikebot's CMUcam, for example, students can visually inspect the quality of the video signal on which they are attempting computer vision operations. By manually moving the robot using a keyboard *joystick*, students disambiguate the locomotive limitations of the robot from the behavioral limitations of their programs.

The UI is subdivided into multiple windows, both for screen real estate adaptability and to logically separate functionality so that each individual form of human-robot interaction is focused and simple. At the control level, the UI enables the student to drive the Trikebot directly, control the head's pan/tilt position and dump images from CMUcam. During each of these control operations, the interface displays and continuously updates the same sensor values that students use during programming: motor speed and current values and rangefinder distance readings. Coupling this sensor feedback to the teleoperation screen further reinforces a student's ability to operate the Trikebot from the robot's point of view, observing and reacting to sensor measurements directly.

The *Tracking* window within the Trikebot UI (Fig. 11) enables students to launch, observe and modify the same high-level visual tracking routines in the Trikebot's iPAQ that they can use programmatically. This window is critically important when using CMUcam because it enables students to select, test and fine-tune vision parameters to ensure that the Trikebot will successfully track its visual targets.

The Trikebot UI was implemented outside of any high-overhead IDE, ensuring that the finished product can be compiled and executed using simple command-level calls in Java 1.4 or beyond. This ease of compilability is key to the *User Controls* window that is also part of the UI (Fig. 10). This window provides the student with a series of buttons and input/output textfields so that, without spending any time on GUI development, the student can launch their programs, observe Trikebot state during program execution and even halt their programs from the UI. This coupling of the teleoperation and control UI to the buttons and fields used to interact with student code is a critical aspect of the success of the Trikebot as an educational, programmable robot. The complete JAVA source fileset is available at TRIKEBOT (2003).



Status Ready (v2.1)

x1 1 y1 1 x2 80 y2 143

P 20
I 0

Turn 0 Speed 0 Pan 0 Tilt 0

Start -90

☑ Pan ☑ Tilt

Pan to Min 0
Pan to Max 240
Tilt to Min 0
Tilt to Max 220
Turn to Min 40
Turn to Max 255
Try Min Fwd Speed 5
Try Max Fwd Speed 50
Try Min Rev Speed -5

Figure 10. The Trikebot laptop UI.

☑ Pan ☑ Tilt

Figure 11. The CMUcam Tracking window enables iPAQ visual tracking code to be launched.

3.3.2. Programming Interface. Although the JAVA client for the Trikebot UI spans a large number of source files, the *User Controls* panel is implemented as a separate source file. In order to change the labels of buttons and text fields, students modify only a single contiguous block of one file. In order to write the JAVA functions that are triggered when those buttons are pressed, the students modify a second contiguous block in one other file (Fig. 12). Students are thus able to program the Trikebot by making direct modifications to two files using a text editor such as JEXT (2003), then compiling and executing from a command line using javac and java. This programming process removes the complexity of teaching students how to use an elaborate IDE such as FORTE.

The use of JAVA as the programming language of choice deserves some discussion. In previous work the authors have taught robot programming using LISP, C, C++ and JAVA. The most effective language among this list was LISP, not only because of its functional nature, similar in spirit to more recent robot programming languages such as GRL (Horswill, 1999), but also because of the existence of a Listener Window and, thus, the interactive ability to call any parameterized,

```
private void Action1() //dumb wander
 {
  int refreshes = 0;
  String Debugging;
  theWindow.quit = false;
  // Set up anything before the control loop
  while (theWindow.quit == false)
  {
   trikebot.RefreshState();
    //get the state variables
   if(trikebot.state.Range() <= 150) {
     trikebot.Drive(20,0);
    //wander forward at a slow speed
   } else {
     trikebot.KillMotor();
   }
  }
  // Do any needed cleanup
 }
```

Figure 12. An example of a student code fragment from the summer 2002 course.

defined function at will. This ability to execute a portion of the robot code in order to diagnose surprising robot behavior is extremely important in robotics, and this same purpose is served somewhat by the Trikebot UI's teleoperation capabilities.

The least effective languages in robot programming tend to be those which open up the field of possible programming errors unrelated to the robot. For this reason, C is particularly poor due to the virtually unbounded ability of a novice programmer to wreak memory space havoc via innocently written C code. JAVA serves as a practical, modern compromise in that it offers greater safety than C in a package that is relatively popular as compared to languages such as PASCAL and LISP.

UNCORRECTED PROOF

4. Educational Analysis Methodology

We assessed impact of the course experience on two levels. First, we conducted a broad evaluation of all the students' experiences in Robotic Autonomy. This evaluation was intended to provide both formative and summative information about whether the course was connecting with students at the appropriate level and making progress toward the broad instructional goals. Second, we conducted an in-depth study of one week of the course. This study, focusing on the experience of two teams of students, was intended to identify some of the micro-genetic mechanisms of learning that might inform patterns of change described in the broader evaluation.

4.1. Data Collected: Whole-Course Evaluation

At the broadest level, four classes of data were used to evaluate the educational effectiveness of the Robotic Autonomy class. First, students completed anonymous surveys about what they were learning throughout the course. On the first day of class, students completed an initial survey of 14 questions covering their technological backgrounds, their expectations for what they would learn in the course, and their plans for college and beyond. Each Monday throughout the course, students also completed a written survey asking them to reflect on the prior week's activities. Students rated their team's performance, described any discoveries they had made or hard problems they had encountered, and indicated how useful they had found specific course activities. During the last week of class, students completed a final survey that included similar content to the initial survey, but also asked specific questions about whether and how students had learned about the core themes and content of the course. The survey forms can be downloaded at RASC (2003).

Second, in addition to the weekly written feedback, an on-site ethnographer conducted on-camera interviews with each team. These interviews usually lasted about ten minutes and were flexible in format. The teams were asked about their progress on the assignments and whether anything particularly notable had occurred that week. A total of 9 hours of weekly team interviews were collected, with approximately 1 hour of interview time per team. Interviews were conducted at different times throughout the week, although an attempt was made to do most of the data collection mid-week.

Third, students were required to open-source and document their challenge programs on the class website. The format included an explanation of what the program did and how to use it, an analysis of its performance and limitations, suggestions for future improvements, and photographs and videos of the robot performing *in situ*. Each team created seven open-source robotics websites to fulfill this requirement. Also associated with each weekly challenge was a grade assigned by the instructor using both quantitative and qualitative grading criteria. The student documentation and grades enabled us to analyze the "output" of student learning over the span of the course.

Finally, after completion of the course, follow-on data was collected in the form of monthly online surveys (RASC, 2003). These surveys asked students about their attitudes toward robotics, science, and engineering; their activities with respect to robotics over the past month; and their future robotics and career plans. In the first 6 months following the end of class, monthly surveys were consistently collected from more than two-third of course graduates.

4.2. Data Collected: One Week in-Depth Evaluation

In addition to the overall evaluation of the Robotics Autonomy class, an intensive, one-week study of two of the nine teams was conducted to develop a more detailed description of the learning and problem solving that occurred in the course on a minute-to-minute basis. The in-depth study focused on the fifth week of class. This week was particularly interesting because teams had mastered the basics of working with the robots and were, for the first time, learning how to work with true robot autonomy. Prior to the fifth week, students used remote control and ded-reckoning to navigate the robot. In week five, the core problem for students was how to enable the robot to do its own navigating through color tracking. Based on his experience teaching robotics, the instructor considered this transition to autonomous navigation to be one of the hardest challenges for students to overcome.

Out of the nine teams in the Robotic Autonomy course, we chose to follow two teams—one all female group, Powerpuff Girls, and one all male group, Snagglepuss. We purposely did not choose the highest or lowest performing groups, aiming instead for groups who were making progress but were still likely to face substantial challenges in making the transition to working with adjustable autonomy. We based our selection of the two groups on the students' online descriptions of their challenge programs, weekly team video interviews, and teacher opinions of the teams. The Powerpuff Girls were chosen over the other all female group, the FemmeBOTS, because FemmeBOTS contained a college freshman majoring in Electrical and Computer Engineering and it was thought that she might provide a disproportionate advantage. The instructors also thought that the Powerpuff Girls worked together more effectively as a team. Snagglepuss was chosen because the team had a good group dynamic and also appeared to be very creative. All three members of Snagglepuss and two members of Powerpuff Girls attended the Robotic Autonomy program through scholarships from National Hispanic University.

Each team spent approximately four hours a day engaged in group work leading up to the contest and challenge problems. The one-week ethnographer videotaped these problem-solving sessions. As there was only one ethnographer, every moment the group spent together was not recorded. However, each group was videotaped for about 10 hours, including several two to three hour problem-solving blocks. No set schedule of data collection was followed; a team was videotaped until they seemed to come to the end of a problem solving session or were all working independently. Snagglepuss frequently divided the problem into parts and worked independently more often than did the Powerpuff Girls. Also one member of Snagglepuss was absent for medical reasons for two and a half days of the five day data collection. Class lectures during the focus week were also videotaped.

To support the interpretation of the tapes, the ethnographer wrote nightly reflections detailing her impression of the day's activities and how students worked together as a group. Each reflection began with a general impression about how successful the day had been for the class as a whole. Then, for each team, the ethnographer recorded impressions of the team as a whole, and then each member of the team individually. In constructing these interpretations we explicitly sought to expand on areas that would help to interpret the activity she had recorded, aided by written notes that she had taken while videotaping.

4.3. Development of Learning Themes and Definitions of all Six Themes

In order to facilitate the evaluation of learning in the students, it was important to partition expected learning into a set of learning themes for which data would then be quantitatively coded. We hypothesize that six learning themes were particularly well suited to the learning taking place in an interdisciplinary program such as Robotic Autonomy. The themes chosen were: Mechanics, Programming, Teamwork, Problem Solving, Robot Point of View (Robot POV), and Self-Identification with Science and Technology (ID with Technology). The first two themes, Mechanics and Programming, encompass obvious lessons garnered from direct interaction with building and programming robots.

The remaining four themes represent important additional opportunities for learning. These themes (Teamwork, Problem Solving, Robot POV, ID with Technology) represent the types of broader learning

UNCORRECTED PROOF

UNCORRECTED PROOF

goals popular in curriculum design. Although popular as design goals, such broad categories rarely yield demonstrable gains, particularly in short-term programs such as Robotic Autonomy.

Mechanics

Sensors, *motors*, *iPAQ*, *back-EMF*, *wiring*, *Trikebot*, *etc.*

Mechanics embodies the interrelationship between various kinematics substructures of the robot and the kinematics of the overall robot. This includes an understanding of mechanical components and the manner in which all these components function together as a deterministic whole system. Basic mechanisms (servos, motors, chassis, suspension, bearings) and electronics (motor controllers, microprocessors, range-finding sensors, the vision system, the iPAQ) comprise this category. Because Robotic Autonomy students began the course by constructing the Trikebot rover using a fast-build kit, we hypothesized significant learning in the area of Mechanics, particularly in the early weeks of the course.

Programming

Java, *debugging*, *documenting*, *compiling*, *etc.*

Programming includes learning how to write commands and scripts that control the robot using, in this case, the Java programming language. The programming skills learned extend well beyond robotics, encompassing code generation/code writing, debugging, documenting, and commenting. Because the Robotic Autonomy challenges posed to the students were primarily challenges for the behavior of the Trikebot, we anticipated that a great deal of the direct learning with respect to overcoming daily challenges would fall in the category of Programming.

Teamwork

Communication, *importance of teamwork*, *etc.*

Learning how to work effectively in teams is a crucial ingredient for success in many endeavors. Specific skills within teamwork include generating and vetting new ideas; assigning roles and responsibilities; and co-constructing knowledge through observation, imitation, conversation and other socio-cognitive processes. Thus learning progress relative to teamwork would be an important focus of any educational evaluation. In Robotic Autonomy all students worked in teams of three on every phase of project completion. The Robotic Autonomy teams were formed in the first week and left intact throughout the seven-week curriculum.

Problem Solving

Patience, *perseverance*, *learning a new method of problem solving*, *etc.*

Robots such as the Trikebot are extremely complex machines. As such, the process of understanding and refining solutions using the Trikebot requires mastery of problem solving methodologies. Such skills include developing effective strategies for solving the problems that arose throughout the course: setting appropriate subgoals, using feedback from the robot to effectively identify weaknesses in current strategies, knowing when to abandon ineffective approaches, etc.

Robot Point of View

Autonomy, *integration of hardware and software*, *control of robot with programming*, *robot diagnosis*, *etc.*

This relatively focused learning theme relates to a critical skill in the understanding of a robot's operating sphere of influence. Robots are extremely limited, in that their sensory and effectory systems are highly constrained relative to that of a human. By *robot point of view* we mean the ability to "see" through the robot's eyes and thus understand the sensor limitations and action constraints under which the robot must operate. It is only by assuming an appropriate robot point of view that a robot designer can begin to discern the space of possible behaviors that are feasible from those that are impractically ambitious.

Self-Identification with Science and Technology

Self-confidence, *robotics community*, *career/experience*, *ethics/open sourcing*, *etc.*

This extensive learning theme encompasses broad empowerment with respect to science and technology. This includes developing an interest in technology, confidence in one's ability to work with technology, and interest in pursuing education and future careers in science and technology. In short, this theme considers students coming to see themselves as people who enjoy and are capable of technological explorations.

UNCORRECTED PROOF

4.4. *Theme Coding Process*

Two reviewers collaborated to code the learning themes. Each of the six themes was divided into general and specific subcategories. For example, for the Programming learning theme, a response that simply said "programming" would be put in the General Programming subcategory, while a response that said "programming in Java" would be coded under the specific subcategory of Java or Other Programming Language.

The following written survey questions were coded for the six learning themes:

Initial Survey:
What is this course about?
What made you want to take this course?
What do you expect to learn in this course?
Final Survey:
Five things I learned from this course were:
What was your favorite part of this course? Why?
What was the one thing you most wanted to change about this course? Why?
Please describe three plans you've made to work with your Trikebot.
Please write any additional comments that you have for us.
Weekly Surveys:
This week I made a big discovery or leap. (yes or no) What was is?
There was something that took me a long time to get or that I missed. (yes or no) What was it?

What students would change about the course and the additional comments were initially coded but were not used in the final learning theme analysis, because the majority of the responses were unrelated to student learning. For example, most of the additional comments were about how much the students liked the class, professor, and teaching assistants. The majority of the responses about what students would change said "nothing" or were a comment on a specific course challenge or contest. These two questions were however used for overall evaluation of the course.

Of the 452 responses coded in the Initial, Final, and Weekly surveys, only 5 did not fit into the learning themes. That 98.9% of the responses fit the learning themes supportes the validity of the coding scheme.

Once the themes were coded we calculated the proportion of times each student said each specific category. The formulas are below:

Initial Survey
Teamwork and Problem Solving: Number of times mentioned in "What is the course about" and "What do you expect to learn" questions $\div$ 2.
Programming, Mechanics, ID with Technology, and Robot POV: Number of times mentioned in all three initial questions $\div$ 3.
Final Survey
Teamwork and Problem Solving: Number of times mentioned in "Five things learned" and "Favorite part of class" questions $\div$ 2.
Programming, Mechanics, ID with Technology, and Robot POV: Number of times mentioned in "Five things learned", "Favorite part of course", and "Three plans for your Trikebot" questions $\div$ 3.

The same proportions were calculated for the whole class using first a sum of the total mentions of a theme and then a count of the number of students who mentioned a theme. Since the sums and counts turned out to be very similar, counts were used for the statistics so that the percentage of students that said something could be extrapolated. ANOVAs for each subcategory were run. Few differences were seen, so we ran theme totals (collapsing all categories) as well as specific theme vs. general theme.

5. Whole Course Evaluation Findings

To describe student experiences in the course, we first present analyses of the initial surveys, weekly surveys, and final surveys. The surveys were used in two ways: to track the success of the course, and also to track what students thought they were learning about each of the six core themes in the course.

5.1. *Overall Success*

In terms of success, responses indicated that the course kept the students' interest and that the curriculum sequence was effective. Every week students were asked to anonymously rate how much they enjoyed the week on a scale of 1 to 5, with 5 being the highest. All weeks except for the fifth week were given a mean rating of 4 or above. Ratings for the fifth week, which was the week when autonomous navigation was presented, averaged 3.4. Consistent with the overall ratings of enjoyment, students found the contests and challenges

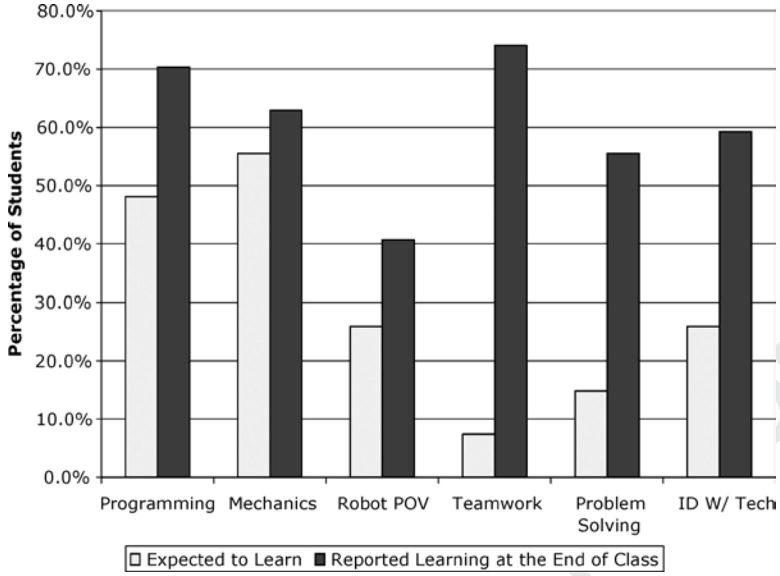


Figure 13. Student self-reports of learning opportunities for each of the core themes in the course. Students were coded for a theme if they mentioned it at least once in response to the survey question.

UNCORRECTED PROOF

to be increasingly motivating and engaging. On each weekly survey, students were asked whether the challenges and contests for that week were their favorite so far in the course. At least 33% of the students each week reported that it had been their favorite week thus far. As the course progressed, students consistently reported high mean levels of learning each week (3.7 and above).

On the final survey student responses also suggested that they had been engaged appropriately by the overall course experience. Students rated instructor effectiveness at a mean of 4.9 on a 5-point scale. Students thought the pacing of the course had been appropriate, rating pacing at 3.6 on a scale from 1 (“Too Slow”) to 5 (“Too Fast”). The guest speakers were appreciated (4.7 out of 5) with every student agreeing that speakers should be included if the course is taught again.

When asked on the final survey what should be changed about the course when it is offered again, 11 of the 27 students said that nothing should be changed, 6 students wanted the course to be longer or cover more material, and 5 students gave random responses, such as the course should be held at a better location. Only 5 students wrote down a specific course criticism, for instance that a certain contest should be redesigned or that the course should have allowed more mixed gender student teams.

5.2. Learning the Core Themes

We first asked the question of how students’ understanding of their own learning changed from the beginning to the end of the course. Students’ expectations for their learning of each of the six themes were coded from their responses to the initial survey question: *What do you expect to learn in this course?* On the final survey, students understanding of their learning of each of the themes was coded from their responses to a question that asked them to list the main things they had learned in the course.

As shown in Fig. 13, students developed different ideas about learning opportunities from the beginning to the end of the course. First, consider what students reported about the three themes that are the most specific to the technical aspects of robotics. At the beginning of the course, 56% of students expected to learn about Mechanics while, at the conclusion of the course, 63% reported Mechanics as one of the important things they learned. Similarly, 48% of students expected to learn about Programming and 70% reported that they had, in fact, done so. These findings do not strike us as remarkable; after all, a course about autonomous robots would certainly include the mechanical and programming aspects common to all robotics.

What are more interesting are the larger differences seen in self-reported learning of Teamwork, Problem

UNCORRECTED PROOF

Solving, and ID with Technology. While 7% of the students initially expected to learn about Teamwork, that theme turned out to be the most commonly reported learning outcome at the end—74% of the students listed it as something they had learned. Similarly, Problem Solving and ID with Technology were commonly reported as learning outcomes at the conclusion of the course, although they had been infrequently mentioned as possible outcomes at the beginning. These findings suggest that the course was successful at meeting the deeper goals of developing domain-general interest and skills that would prepare students for success in broader technology and science education in college.

A caveat deserves mention regarding the results shown in Fig. 13. The initial survey question preambled a single blank block for an answer; and therefore many students responded with a single learning expectation. The final survey offered five blank lines for answers to the same question, and therefore students always responded with many themes. Although this structural difference has impact on the absolute response frequency, distribution data across themes is informative; it is in this change in distribution that the increased emphasis on Teamwork, Problem Solving and ID w/ Tech can be seen.

In addition to coding whether students mentioned learning opportunities for each of the themes, we coded relevant questions from the initial and final surveys to track how much specific detail students reported when they described learning opportunities around specific themes. Although students mentioned Mechanics and Programming a similar number of times in the initial and final surveys, they provided significantly more specifics about each theme on the final survey. For instance, while students mentioned vague statements about "robot technology" on the initial survey, they were more likely to mention specific technologies such as "IR sensors" or "back-EMF" on the final survey, F(1, 52) = 5.47, $p < .05$. While they mentioned "learning to program" on the initial survey, they were more likely to talk about "states in programming" or "Java" on the final survey, F(1, 52) = 8.61, $p < .01$. Thus, student descriptions of their own learning became more specific and grounded in the curriculum content.

How students talked about the themes of Teamwork and Problem Solving also changed to include more specifics by the end of the course. Students originally said they would "learn teamwork" or "work in teams of three". In the final surveys comments like "Teamwork is hard especially with varying levels of skill and different personalities, [it] can be rewarding only through compromise" and "teamwork leads to victory" were more common, F (1, 52) = 15.91, $p < .001$. Similarly, from a few general statements about "learning how to solve problems" on the initial survey, student statements changed to specific observations such as learning to "really pay attention to what I am doing and try to solve it first before asking for help", F (1, 52) = 12.00, $p < .001$.

We now turn to an analysis of the weekly surveys students completed each Monday. Two of the key questions on the survey asked students to reflect on whether they had, in the preceding week, made a breakthrough or discovery and whether they had struggled to understand anything. Responses for all weeks and students were summed for analysis. There was a possibility for 162 responses to each question, but not every student reported a struggle and breakthrough every week. For all six surveys given there were 51 reported struggles, between five and thirteen per week, and 87 breakthroughs, between nine and seventeen per week.

As shown in Fig. 14, student struggles were mostly around two themes: Programming and Mechanics. This is not surprising, because those topics are most directly tied to the challenges. Typical responses are shown below.

"Our program had a bug which turned out to be a missing zero."
"There were long time delays between commands."
"Robots need to be tested in the same conditions as where they will perform."

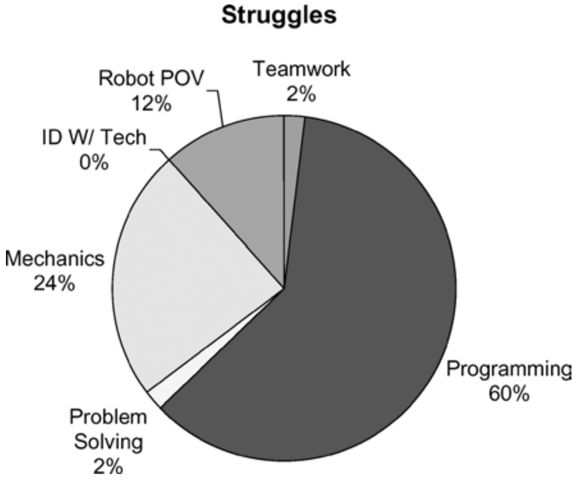


Figure 14. Percent of reported struggles by learning theme.

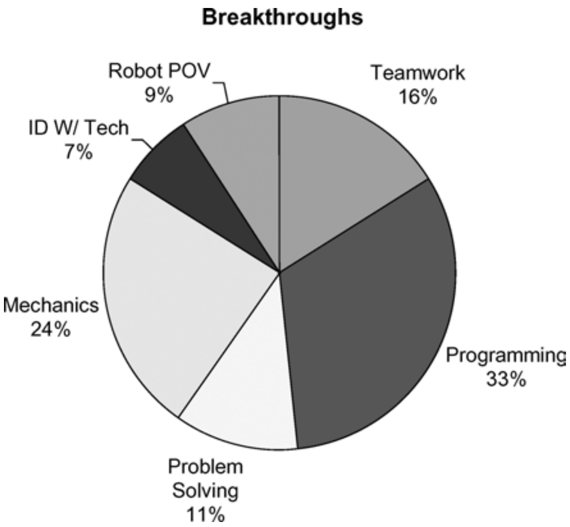


Figure 15. Percent of reported breakthroughs by learning theme.

In contrast, student breakthroughs occurred widely among the six themes. Mechanics and Programming were still mentioned most often, but breakthroughs coded as involving Teamwork, Problem Solving, Robot POV, and ID with Technology were also common (see Fig. 15).

Examples of these breakthroughs include:

Programming: "New programming languages are easier to understand than I thought."
Mechanics: "Understanding how sensors are so wonderful and yet so error prone."
Teamwork: "The big discovery was that if I try hard, by working with my teammates, we could make a lot of things happen."
Problem Solving: "Don't ever leave anything at the end or else you will be struggling to finish it on time."
Robot POV: "Robots are babies."
ID with Technology: "I made the discovery that building a robot could be very exciting instead of hard."

5.3. Gender Differences

Finally, we analyzed the student self-report data for potential gender differences. Although we began the project with no particular expectations that girls and boys would have different experiences, we were sensitive to the historical problem that computer science has had in attracting girls to engage in advanced study. We were also acutely aware of the fact that the majority of the students were boys, all of the outside speakers were men, and that the instructor and all but one teaching assistant were men. As the robot course was one of the first intensive advanced technology experiences for most of the students, we were aware that it had the potential to work against or in support of existing stereotypes regarding girls and technology. Thus, we were particularly interested in whether the experience was successful and positive for the eight girls enrolled.

For most of our findings, there were no differences between girls and boys, suggesting that the course provided a supportive and interesting environment for both. We did observe three differences. First, on the weekly surveys girls were more likely to report having struggled with Programming, $F(1, 25) = 9.12$, $p = .01$. Second, girls also entered the class reporting less confidence with technology than boys, $F(1, 25) = 9.72$, $p = .01$. Third, girls' confidence with technology increased more than boys' by the end of the course, $F(1, 25) = 14.58$, $p = .001$. Thus, despite our initial concern, the course appeared to welcome and support the participation of girls.

In summary, findings on student reported learning suggest that the course was successful in meeting its specific instructional goals of teaching the technology of autonomy and also its general goals of supporting meaningful student engagement with technology to build general interest, skills, and confidence that could promote future success with technology education.

5.4. Post-Course Survey Results

Educational evaluation of the Robotic Autonomy course has identified learning mechanisms and patterns within the scope of the seven-week course timeline. We implemented a periodic follow-up survey with course graduates in order to comprehend the longevity of those learning results. The follow-up survey was implemented as a web-based form sent to all course graduates once per month for six months. The web form was comprised of twenty topic questions, designed to probe ongoing self-identification with technology, quantitative self-reports regarding time spent with the Trikebot at home, and future career plans.

Student participation in the monthly survey was initially high, with 15 or more results each month for months one through four, with a significant drop-off in participation thereafter. While the dataset size obviates statistical evaluation, some instructive qualitative trends and results can still be developed, as discussed below.

UNCORRECTED PROOF

UNCORRECTED PROOF

Self-identification with technology is a significant theme based on analyses of Robotic Autonomy learning patterns and further because it has potential lifelong impact on attitudes and focus on technology literacy. The follow-up survey scored “I am familiar with robotics” and “I am comfortable with technology” (among twenty total questions) using five-point scales in order to establish the longevity of increased comfort and identification with technology during Robotic Autonomy. Familiarity with robotics consistently scored 3.60 or higher, with zero survey respondents trending lower month over month. Familiarity with technology as a whole scored much more highly, 3.90 to 4.60, and 18% of respondents trended higher month over month. The stability of these self-report results is encouraging because this suggests that gains in technology confidence over the course of Robotic Autonomy were not sacrificed in the months following graduation. The particular strength of “comfort with technology” suggests that, at the broad topic level, gains are not undermined following course graduation and may even be more tenacious than specific technological lessons such as robotics. This is further evidenced by a downward trend on “I will use my Trikebot at least once next month,” which trended down month over month for 33% of respondents (the remaining 66% of respondents reported the same score month over month). As the specific robotics tool is pushed to the background, we are encouraged that the broad technology literacy lesson lives on.

The theme of teamwork was tracked via a general question, “I like working in teams” and a specific communication question, “I will keep in touch with my RASC classmates.” Results again showed robustness of general learning in spite of narrow loss of interest, natural with the passage of time. With regard to remaining in contact with Robotic Autonomy classmates, 31% of respondents trended downward month over month, with an average overall score of 3.40. Yet with regard to enjoying working in teams, there was no downward trend, with an average score of 4.40. While specific social relationships with Robotic Autonomy peers fades due to the passage of time, we hypothesize that team problem-solving skills gained during the course can be retained through other activities.

In quantitative terms, average reported hours of Trikebot usage per student per month for months one through five were: 13.5 hours; 4.8 hours; 7.3 hours; 8.3 hours; 2.3 hours, respectively. We hypothesize that as senior year high school responsibilities grew, time for robotics exploration decreased by December. Yet between 30% and 57% of respondents reported participation in other robotics activities each month, with a slight upward trend month over month. This is once again encouraging because skills acquired during Robotic Autonomy, particularly confidence with technology, will be of value in enabling participation in such projects.

In summary the follow-up survey, while not yielding statistically significant conclusions, supports the contention that lessons learned during Robotic Autonomy are not transient, and that comfort with technology and a willingness to participate in technology-related projects may be the key long-term benefits of such an educational robotics program. A surprising quantitative result is that each respondent used their Trikebot robot at home for an average of 34 total hours in the four months following Robotic Autonomy graduation.

6. Conclusions

The overarching goal of this work has been the demonstrate end-to-end design and execution of a mobile robotics educational course. The educational focus of this assessment has been to characterize the impact of this hands-on robotics course using formal techniques. Our prior experiences with robotics education suggested that relatively broad forms of learning may be demonstrable, and this hypothesis has been validated. Learning about the coded themes of Mechanics and Programming is to be expected in a robotics course. Quantitative results based on self-reports supported this expectation. More surprising were large jumps from expectation to reported learning along the themes of Problem Solving, Teamwork and ID with Technology. This suggests that the course was able to meet deeper goals of developing domain-general interest and skills that can prepare students for broad success in technology and science education.

Coding for the level of detail in student comments regarding learning themes led to statistically significant increases in specificity. Significant trends were measured for “robot technology,” Programming, Teamwork and Problem Solving. These results suggest that students learned concrete lessons for each theme, digging below the surface of abstract concepts to a functional level of detail.

Evaluation of self-reported struggles and breakthroughs supported the above conclusions. Student

UNCORRECTED PROOF

struggles were reported mainly around two themes: Programming and Mechanics. But, student breakthroughs were reported across a broad range of themes, including Teamwork, Problem Solving, Robot POV and ID with Technology. Once again the inclusion of non-technological themes reported as breakthroughs suggests that, during the course, learning extended beyond the content of technical challenges and into broader scientific and social lessons.

Finally, analysis of student self-report data for gender differences was intended to identify the effect of this advanced technology course on existing stereotypes regarding girls and technology. Thus a critical question would be the degree to which Robotic Autonomy was a positive and successful experience for the girls enrolled. Three significant results summarize conclusions on this query. First, girls were more likely to struggle with Programming. Second, girls entered the course reporting less confidence with technology than boys. But third, girls' confidence in technology increased throughout the course significantly more quickly than the boys'. Thus the course appeared to support the participation of the girls and was able to compensate somewhat for the initial differences between girls' and boys' comfort with technology.

Acknowledgments

The Robotic Autonomy program was funded in part by NASA/Ames Research, National Hispanic University and Intel Corporation. The back-EMF speed control solution was first proposed by Randy Sargent and demonstrated at the Swiss EPFL AMR-DST in the laboratory of Roland Siegwart. Thanks for the following individuals who made significant contributions to Robotic Autonomy: Adriana Cardenas, Daniel Clancy, John D'Ignazio, Maylene Duenas, Neeti Malhotra and Raj Reddy. Thanks also to the anonymous reviews and editorial staff of Autonomous Robots for their valuable suggestions that have improved this article significantly.

References

Acroname. 2003. Web reference: http://www.acroname.com.

ActivMedia. 2004. Web reference: http://www.mobilerobots.com.

Archibald, J.K. and Beard, R.W. 2004. Goal! robot soccer for undergraduate students. *Robotics & Automation Magazine, IEEE*, 11(1).

Beer, R., Chiel, H., and Drushel, R. 1999. Using autonomous robots to teach science and engineering. *Communications of the ACM*.

Billard, A. 2003. Robota, clever toy and educational tool, Special Issue on *Socially Interactive Robots, Robotics and Autonomous Systems.*

Billard, A. and Hayes, G. 1997. Learning to communicate through imitation in autonomous robots. In *Proceedings of the 7th International Conference on Artificial Neural Networks*.

Botball. 2004. Web reference: http://www.botball.org

Cooper, M. et al. 1999. Robots in the classroom—tools for accessible education. In *Proceedings of the 5th European Conference for the Advancement of Assistive Technology*.

Coppin, P. and Wagner, M.D. 2002. EventScope: A telescience interface for internet-based education. In *Proceedings of the Workshop on Telepresence for Education, IEEE International Conference on Robotics and Automation*.

Coppin, P., Morrissey, A., Wagner, M.D., Vincent, M., and Thomas, G. 1999. Big Signal: Information interaction for public telerobotic exploration. In *Proceedings of the Workshop on Current Challenges in Internet Robotics, IEEE International Conference on Robotics and Automation.*

Druin, A. 2000. *Robots for Kids: Exploring New Technologies for Learning*. The Morgan Kaufmann Series in Interactive Technologies, Morgan Kaufmann, pp. 158–164.

Druin, A. and Hendler, J. 2000. *Robots for Kids: Exploring New Technologies for Learning*. The Morgan Kaufmann Series in Interactive Technologies, Morgan Kaufmann.

Evolution Robotics. 2004. Web reference: http://www.evolution.com

Falcone, E., Gockley, R., Porter, E., and Nourbakhsh, I. 2003. The personal rover project, Special Issue on *Socially Interactive Robots, Robotics and Autonomous Systems*.

Fagin, B. 2003. Ada/Mindstorms 3.0. Robotics & automation magazine. *IEEE*, 10(2).

Fong, T., Nourbakhsh, I., and Dautenhahn, K. 2003. A survey of socially interactive robots. Robotics and autonomous systems, special issue. *Socially Interactive Robots*, 42(3/4), 143–166, in print.

Fong, T., Nourbakhsh, I., and Dautenhahn, K. 2002. A survey of socially interactive robots: Concepts, design and applications. Technical Report CMU-RI-TR-02-29, Carnegie Mellon University, Pittsburgh, PA, USA.

Gage, A. and Murphy, R.R. 2003. Principles and experiences in using legos to teach behavioral robotics. In *Proceedings of the 33*rd *Frontiers in Education Conference*, Vol. 2.

Heer, D., Traylor, R., and Fiez, T.S. 2002. Tekbots/sup TM/: Creating excitement for engineering through community, innovation and troubleshooting. In *Proceedings of the 32nd Frontiers in Education Conference*, Vol. 2.

Hobson, R.S. 2000. The changing face of classroom instructional methods: Service learning and design in a robotics course. In *Proceedings of the 30th Frontiers in Education Conference*, Vol. 2.

Horswill, I. 1999. Functional programming of behavior-based systems. In *Proceedings IEEE International Symposium on Computational Intelligence in Robotics and Automation.*

Hsiu, T., Richards, S., Bhave, A., Perez-Bergquist, A., and Nourbakhsh, I. 2003. Designing a low-cost, expressive educational robot. In *Proceedings of IROS 2003*, Las Vegas, USA.

JEXT. 2003. Web reference: http://www.jext.org

K-Team. 2004. Web reference: http://k-team.com

Kolberg, E. and Orlev, N. 2001. Robotics learning as a tool for integrating science technology curriculum in K-12 schools. In *Proceedings of the 31st Frontiers in Education Conference*, Vol. 1.

Kumar, A.N. 2001. Using robots in an undergraduate artificial intelligence course: An experience report. In *Proceedings of the 31st Frontiers in Education Conference*, Vol. 2.

Kumar, D. and Meeden, L. 1998. A robot laboratory for teaching artificial intelligence. In *Proc. of 29th SIGCSE Symposium on Computer Science Education*.

Manseur, R. 2000. Hardware competitions in engineering education. In *Proceedings of the 30th Frontiers in Education Conference*, Vol. 2.

Martin, F., Mikhak, B., Resnick, M., Silverman, B., and Berg, R. 2000. *Robots for Kids: Exploring New Technologies for Learning*. The Morgan Kaufmann Series in Interactive Technologies, Morgan Kaufmann, pp. 10–32.

Maxwell, B. and Meeden, L. 2000. Integrating robotics research with undergraduate education. *IEEE Intelligent Systems*.

Milto, E., Rogers, C., and Portsmore, M. 2002. Gender differences in confidence levels, group interactions, and feelings about competition in an introductory robotics course. In *Proceedings of the 32nd Frontiers in Education Conference*, Vol. 2.

Murphy, R. 2000. *Introduction to AI Robotics*. MIT Press.

Nagchaudhuri, A., Singh, G., Kaur, M., and George, S. LEGO robotics products boost student creativity in precollege programs at UMES. In *Proceedings of the 32nd Frontiers in Education Conference*, Vol. 3.

Nourbakhsh, I. 2000a. When students meet robots. Essay in *IEEE Intelligent Systems and Their Applications*, 15(6), 15.

Nourbakhsh, I. 2000b. Robotics and education in the classroom and in the museum: On the study of robots, and robots for study. In *Proceedings Workshop for Personal Robotics for Education*, IEEE ICRA.

Nourbakhsh, I. 2000c. Property Mapping: A simple technique for mobile robot programming. In *Proceedings of AAAI 2000.*

Papert, S. and Harel, I. 1991. *Situating constructionism, In Constructionism.*, Ablex Publishing Corp.

RASC. 2003. Web reference: http://www.cs.cmu.edu/∼rasc

Rowe, A., Rosenberg, C., and Nourbakhsh, I. 2002. A low cost embedded color vision system. In *Proceedings of IROS 2002*.

Schumacher, J., Welch, D., and Raymod, D. 2001. Teaching introductory programming, problem solving and information technology with robots at West Point. In *Proceedings of the 31st Frontiers in Education Conference*, Vol. 2.

Siegwart, R. 2001. Grasping the interdisciplinarity of mechatronics. *Robotics & Automation magazine, IEEE*, 8(2).

Stein, C. 2002. Botball: Autonomous students engineering autonomous robots. In *Proceedings of the ASEE Conference*.

TRIKEBOT. 2003. Source code download site. Web reference: http://www.cs.cmu.edu/∼rasc/RA/TrikeBackg.htm

US FIRST. 2004. Web reference: http://www.usfirst.org

Wang, E. 2001. Teaching freshman design, creativity and programming with LEGOs and Labview. In *Proceedings of the 31st Frontiers in Education Conference*, Vol. 3.

Wang, E. and Wang, R. 2001. Using Legos and RoboLab (LabVIEW) with elementary school children. In *Proceedings of the 31st Frontiers in Education Conference*, Vol. 1.

Wolz, U. 2000. Teaching design and project management with Lego RCX robots. In *Proc. SIGCSE Conference*.

Yim, M., Chow, M., and Dunbar, W. 2000. Robots for kids: Exploring new technologies for learning. The Morgan Kaufmann Series in Interactive Technologies, Morgan Kaufmann, pp. 245–290.

UNCORRECTED PROOF