Teaching Robotics Software with the Open Hardware Mobile Manipulator

Marsette Vona, *Member, IEEE,* and Shekar NH

***Abstract*—The *open hardware mobile manipulator* (OHMM) is a new open platform with a unique combination of features for teaching robotics software and algorithms. On-board low- and high-level processors support real-time embedded programming and motor control, as well as higher-level coding with contemporary libraries. Full hardware designs and software source codes are available under free licenses enabling the community to adapt and update the design as the field (and available hardware) evolves. Digital manufacturing (3D printing and laser cutting) methods, now available as low-cost internet services, enable OHMM to be easily modified and to be built from scratch for a competitive price. Five hands-on curriculum modules are presented which use OHMM to teach robotics software and algorithms at the undergraduate and graduate levels, and results are presented based on student performance and feedback.**

***Index Terms*—Digital manufacturing, educational robot, embedded computing, mobile manipulation, open hardware, visual servoing.**

I. INTRODUCTION

FREE and open source software (FOSS [1]) is now extensively used in education [2]. Recent efforts have also brought to prominence open courseware (OCW [3], [4]), such as lecture notes and homework problems. This paper introduces the further evolution of these themes to a new open source hardware (OSHW [5]) platform specifically suited for teaching robotics software and algorithms. Though open hardware can be more logistically complex than software or courseware, the intention is to enable some of the same advantages: transparent and documented hardware and software designs enable students to explore and develop whole-system understanding; a common shared platform can make it possible for educators and students at disparate institutions both to learn together and to collaborate to improve the curriculum; and a free design can be adapted and updated by the community as the field of robotics—and available hardware such as motors, processors, and cameras—evolves.

This new platform is called the *open hardware mobile manipulator* (OHMM), Fig. 1. The domain of mobile manipulation is chosen as it supports teaching a broad cross-section of robotics software and algorithms: control, planning, obstacle avoidance, kinematics, sensing and perception, manipulation, etc. Full hardware designs and software source codes are available on the web at [6] under CreativeCommons and GNU GPL licenses. Though this is not the only OSHW robot [7]–[11], a combination of features make it unique:

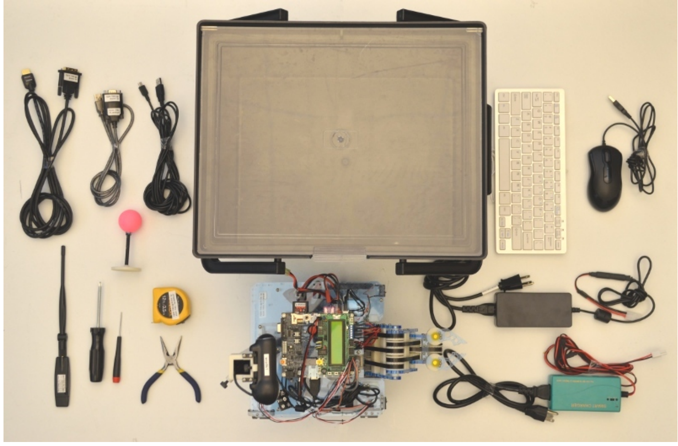

Fig. 2: The full lab kit fits in a 17 × 14 × 11 inch carry case (top center) with all necessary tools and accessories.

- **Low- and high-level processors** (LLP and HLP) so students can learn both bare-metal embedded programming and real-time motor control in C/C++ as well as system implementation using high-level languages—Java, Python, Scheme, and so on—and libraries like OpenCV [12] and ROS [13].
- **An arm and gripper** with three degrees of freedom (DoF), intelligent servos, and intrinsic compliance which can grasp, lift, and carry small objects.
- **A Mast-mounted camera** with a field of view that can be used both for obstacle avoidance and navigation as well as visual servoing and grasp planning. Both monocular RGB cameras and depth cameras like the Kinect are supported.
- **Able to fit in a portable case** for software labs that may have limited space. Enables students to "sign out" a kit.
- **Digitally-manufactured** parts. Unlike traditional machining, new 3D printing and laser cutting technologies require little expertise to design or modify and parts can be quickly fabricated via low-cost internet services.
- **Multiple configuration** options from basic mobility to full mobile manipulation with correspondingly scaled costs.

Though robotics can be taught from many different perspectives, the primary focus of the OHMM platform and curriculum is teaching *software and algorithms* to students with a basic computer science background (or at least some programming experience). Special attention is paid to a rich software-hardware interface, while pre-designed uniform hardware enables a curricular focus on software, but not on mechanical or electrical design and fabrication. An additional concern is the practical issue of portability, as many software teaching labs are not equipped with the same tools and workspaces as mechanical or electrical engineering labs.

The authors are with the College of Computer and Information Science, Northeastern University, Boston, MA 02115 USA e-mail: vona@ccs.neu.edu.

Manuscript received TBD; revised TBD.

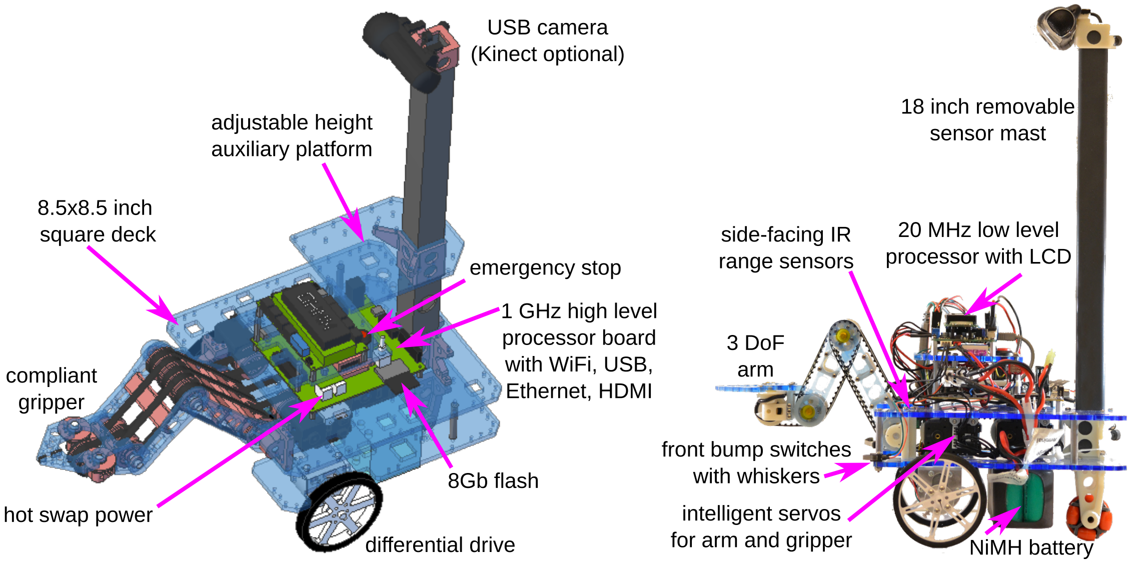

Fig. 1: The Open Hardware Mobile Manipulator (OHMM), a new platform for teaching robotics software.

To assess the usefulness of the OHMM platform within a Computer Science department, 12 kits, Fig. 2, have been constructed and used in undergraduate- and graduate-level applied robotics software courses. After reviewing related work, details are given on the hardware (Section II), software (Section III), and curriculum modules (Section IV). The paper concludes with results and analysis based on student feedback (Section V).

A. Related Work

The unique combination of features in OHMM compared to other open hardware platforms was presented above. But what about non-open platforms? A minimal version of OHMM omitting the arm, camera, and high-level processor can be built for as little as \$650; the full kit with all features costs \$1730[1]. Some currently available small mobile manipulators (e.g., Kuka YouBot [14]) are over ten times as expensive. Those available for a comparable price either would not fit in an easily portable case or do not directly allow access to low-level motor control (e.g., platforms like Bilibot [15]); others omit features like the arm, camera, or high-level processor[2] (e.g., CrustCrawler Nomad [16], Lego Mindstorms [17]). Also, compared to platforms based on construction kits—Vex, Mindstorms, Fischer Technic, etc.—for teaching software and algorithms, it can be preferable to limit hardware variability (the numerous components in such kits can also be a challenge [18]).

The curriculum modules presented here are inspired by the one-term course Robotics Science and Systems I at MIT (6.141) [19], which one of the authors of the present paper (Vona) helped to develop in 2006. The strategy is to teach a sampling of algorithms and software development, starting with low-level motor control and culminating in visual-servo grasping and map-based navigation, while simultaneously assembling a robot from scratch. Thus students get the experience of building a full robot from primary components, both hardware and software. Successive curriculum modules use previously-developed capabilities, with the option for students to use either their own implementations or solution code. In addition to being open, the OHMM platform is significantly smaller and cheaper than the MIT lab kits, and could enable this type of course to be taught more broadly.

A number of other robotics software courses, such as Mobile Robotics at Washington University [20] and Introduction to Autonomous Robotics at Brown University [21] teach robot decision making, perception, and control using the Turtlebot [10]. This partly open platform is based on the iRobot Create base, which has some advantages, but does not offer direct access[3] to teach low-level programming for motor control. Further, it has no arm (though one can be built from third-party designs) and its side-facing Kinect, oriented for obstacle detection and mapping, does not have a good field of view for grasping local objects based on visual servoing.

II. HARDWARE

OHMM aims for portability and low cost while retaining key elements for teaching mobile manipulation software and algorithms. It is also intended to be accessible, not only via its open design, but also by using digital manufacturing—3D printing and laser cutting—instead of custom machining. Digital manufacturing equipment is increasingly common at educational institutions and is also now broadly available on the internet from services like Ponoko [22]. In addition to being lower-cost and faster for small part quantities, these technologies also have simpler design constraints than traditional machining. This should enable OHMM to be updated and adapted by a community including scientists and educators who may not be experienced in design for manufacture.

---

[1] These are current costs to purchase parts for one unit from primary component suppliers in the US.

[2] Specifically, a high-level processor where standard languages such as Python, Java, and Scheme can be used, in addition to major libraries like OpenCV and ROS.

[3] To the authors' knowledge, short of reverse-engineering.

To meet these goals OHMM combines off-the-shelf circuit boards, sensors, actuators, and mechanical components like wheels and belts; digitally-manufactured custom parts; and a few items that require basic hand tools such as the aluminum arm shafts which are cut with a hacksaw, Fig. 3. The full bill of materials with vendors, design files, and assembly instructions are provided on the OHMM website [6].

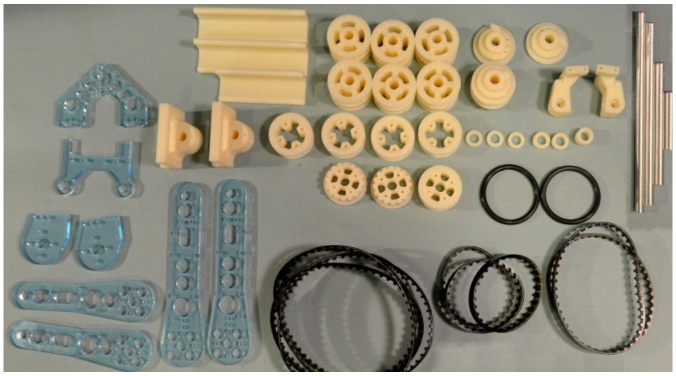

Fig. 3: Laser cut acrylic arm parts (left, blue), 3D printed ABS parts (top, white), hand-cut aluminum shafts (upper right), and standard off-the-shelf drive belts (lower right).

Table I summarizes major component categories, costs[4], and notes on availability and documentation. These considerations are important to make OHMM a useful open platform that will be adaptable as available hardware changes.

Category	Cost	Note	Category	Cost	Note
3D printed ABS	\$430	1–4	mast	\$4	3–5
laser cut acrylic	\$72	1–4	NiMH battery	\$70	3
drive belts	\$27	1–4	low level processor	\$100	4,5
fasteners	\$55	1–4	high level processor	\$205	4,5
arm shafts	\$12	1–4	castor wheel	\$15	4,5
proximity sensors	\$28	1–4	power hot swap	\$20	4,7
wiring, switches	\$88	1–4	camera	\$50	1,5,6
arm servos	\$180	1,5,6	drive motors	\$130	5,6

TABLE I: Breakdown of major component categories. Notes on availability and documentation: (1) multiple vendors; (2) multiple manufacturers; (3) drop-in alternatives available; (4) detailed documentation; (5) alternatives require moderate design changes; (6) partial documentation; (7) optional.

A. Low- and High-Level Processors

Particular attention was paid to the choice of on-board embedded processors to support teaching low- and high-level programming and hardware interfacing. Each includes full I/O port documentation and low-level sourcecode and can be programmed from students' computers using free tools.

The low-level processor (LLP) includes a 20MHz 8-bit AVR microcontroller, 128kB flash, 16kB RAM, pulse-width-modulated amplifiers for the drive motors, corresponding quadrature encoder counters, power management and emergency stop provisions, a liquid-crystal display, USB interface, analog and digital sensor inputs, low-level digital communications, and several user switches. It is programmed in C at the "bare metal" level (no formal operating system, though some libraries are provided, see Section III).

The high-level processor (HLP) includes a 1GHz ARM CPU, 8GB flash, 1GB RAM, four USB ports, Ethernet, WiFi, Bluetooth, RS-232 serial, and an HDMI display port. It runs GNU/Linux and can be programmed in C/C++, Java, Python, Scheme, and many other languages, with powerful open-source libraries like OpenCV, ROS, JavaCV [23], OpenKinect [24], and the Point Cloud Library (PCL) [25].

In the current design both processors are single-board-computers: the LLP is the Pololu Orangutan SVP 1284 [26] (2.2 × 3.7 inches) and the HLP is the Texas Instruments PandaBoard [27] (4 × 4.5 inches). There are other products with similar capabilities, price, and form factor, and OHMM's open design means that the community can adapt it for other choices.

B. Differential Drive Base and Arm

OHMM has a differential-drive mobility base with an 8.5 inch square deck, two DC drive motors with encoders, and a Nickel-metal hydride battery ($\sim$4h operation per charge). A hot-swap circuit enables switching to tethered power or swapping batteries rapidly, which can be important in time-constrained educational settings. The deck has mounting holes for peripheral sensors, nominally two forward bump switches with whiskers and two side-facing infra-red range sensors.

Unlike many other low-cost designs, OHMM's 3DoF manipulation arm situates all its actuators in the base and drives each joint with belts. This keeps the moving mass low so that relatively inexpensive servomotors can be used, even while holding outstretched poses. It also provides intrinsic compliance in the belts[5]. For low-load operations (up to about eight ounces payload) a simple calibration procedure is sufficient for repeatability within less than 0.25 inch. Maximum reach is about nine inches horizontal and eight inches vertical. The four intelligent servos communicate via a serial link with the LLP.

The current design uses motor and wheel assemblies from Pololu for the drive motors and Dynamixel AX-12A+ smart serial servos from Robotis for the arm. As for the processor boards, OHMM could be adapted for other alternatives.

C. Camera Mast

OHMM includes an 18 inch tall mast to hold a camera with a field of view covering both the arm workspace and potential obstacles. One compelling exercise is visual servoing to locate, approach, and collect an object, Section IV, Fig. 4. The HLP has enough computational power to use libraries like OpenCV and PCL, enabling a wide range of vision-based learning activities ranging from simple line following, Fig. 5, to cutting-edge object recognition.

The current design use a Logitech webcam, and also includes a bracket to mount the Microsoft Kinect, Fig. 6. By

---

[4] Full kit cost is more than the sum of these as it includes tools, the box, etc.

[5] This is only coarsely useful as only the servo positions, not the joint angles, are sensed; add-on joint sensors are under development.

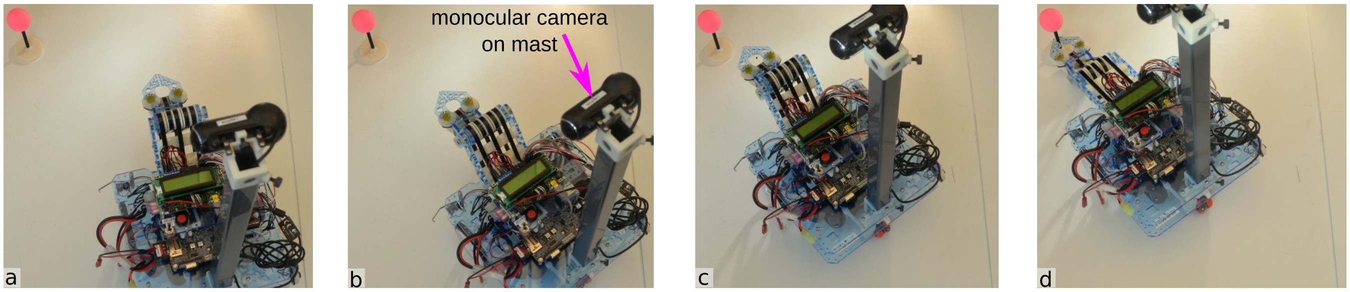

Fig. 4: Visual servoing to automatically approach and grasp an object.

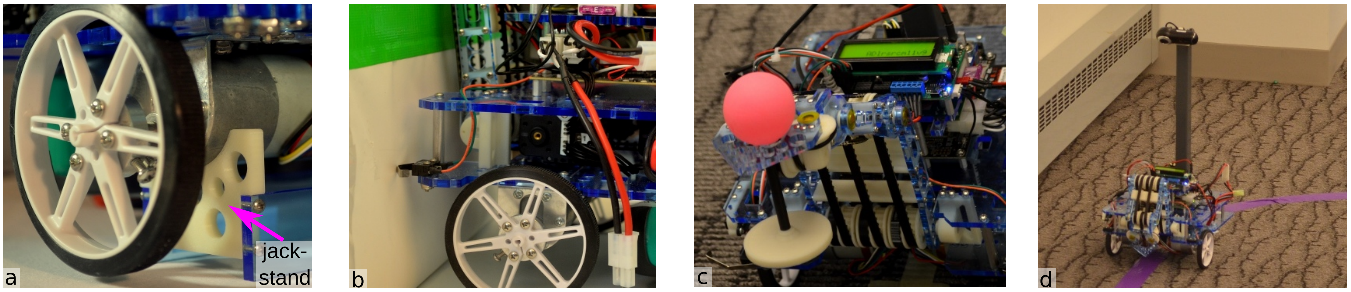

Fig. 5: Curriculum modules: (a) motor control; (b) obstacle avoidance and navigation; (c) kinematics and grasping; (d) integration (also see Fig. 7). An additional module on visual servoing is shown in Fig. 4.

adapting the mechanical mount a wide variety of small USB cameras could be used.

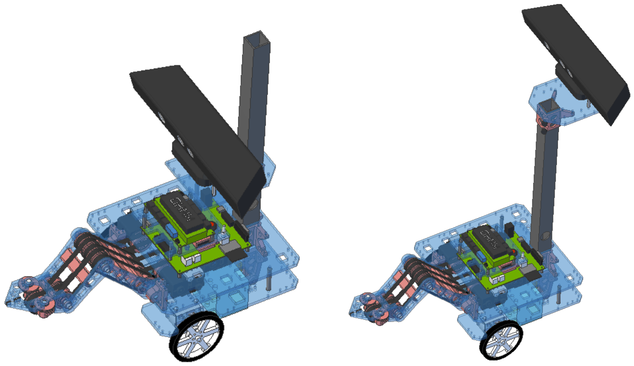

Fig. 6: Two options for mounting the Kinect for 3D perception.

D. *Assembly and Example Configurations*

A two-stage assembly process is expedient for teaching software and algorithms while still giving students an experience of constructing the robot from scratch. The course staff does a first stage of initial assembly (about four to six hours per robot), including some soldering and work with a few other basic tools. The remaining assembly is done by the students themselves during the course labs with a few simple and safe tools (screwdriver, pliers). At the end of the term those steps are reversed by the course staff to enable the hardware to be used again.

Most subsystems are sufficiently decoupled to enable a variety of useful configurations to save cost or reduce complexity. Consider the scenario where the students progressively assemble the robots through the term. First only the wheels, battery, and LLP are assembled. Then the whiskers, bump switches, and HLP are added, followed by the arm, and finally the mast and camera. The robot is functional and durable at each intermediate stage. Other possible configurations include everything except the HLP and camera; no arm; or just the arm and the LLP as a fixed-base manipulator.

III. SOFTWARE

The OHMM platform uses a variety of open-source software components (GNU/Linux, OpenCV, etc.) to help get students up and running and doing interesting projects quickly, and also to give them experience with some of the same libraries currently being used in many research and industrial projects.

OHMM also includes a new open-source Java library with over 200 functions to interface with the hardware. This runs on the HLP and communicates with a new open-source firmware for the LLP (written in C) with modules for task management, motor control, LCD, sensors, command processing, battery management, and manipulator arm. The firmware can also be used in an ASCII communications mode with a user command prompt for development and debugging using only the LLP, which may be connected by USB directly to a PC.

These OHMM libraries are copiously documented so students can use, study, and extend any part of them. For example, in the first curriculum module below students build on the provided motor control C code on the LLP and add new functions for whole-robot differential drive control and odometry; in later parts of the curriculum they make heavy use of the Java library on the HLP as a basis for implementing high-level algorithms including navigation and visual servoing.

IV. CURRICULUM MODULES

The OHMM platform has been used for five curriculum modules in one-term introductory courses for undergrad and graduate students. The robots are progressively assembled and programmed so that by the end of the term the students have constructed an entire autonomous mobile manipulator from primary components.

Descriptions of the progression of the five modules are given below. Each lasts for approximately two weeks, includes several scheduled in-lab sessions with the course staff, and culminates in a live demonstration of the students' work.

Differential drive control and odometry, Fig. 5: After assembling the two front drive wheels, motors, encoders, the rear wheel, the battery, and the low-level processor, students work to develop embedded C code on the LLP both for differential drive velocity and position control as well as for odometry to estimate the robots pose from encoder feedback. "Jackstands" included in the kit are important to help the students develop these algorithms. At the end of this module the robot can report its current pose (x, y, θ) and linear and angular velocity $(\mathbf{v}, \omega)$, maintain requested velocities under disturbances, and execute sequences of position commands such as "drive forward 2.5m, then turn left 90deg, then drive backward 0.75m."

Obstacle avoidance and map-based navigation, Fig. 5: The students add the high-level processor, forward bump switches with whiskers, and side-facing infra-red distance sensors. Two major software challenges are presented: first, the students implement a version of the classic Bug2 algorithm [28] for reactive obstacle avoidance while traveling along a nominal straight-line trajectory to a specified goal location. Second, they implement a global planner of their choosing (e.g. Dijkstra's algorithm on a visibility graph in configuration space) to navigate to a goal location based on a map of obstacles given at run time. Both parts of this module make use of the differential drive control and odometry capabilities from the first module; students are given the choice to use either their own implementation, other students' implementations (with permission), or provided solution code.

Arm kinematics and grasping, Fig. 5: The 3DoF manipulator arm and gripper (supplied to the students partially assembled) and four intelligent servos are added. After a calibration and forward kinematics exercise, the students implement analytic inverse kinematics in the vertical plane and code to transform Cartesian end-effector commands to joint commands. Treating the turn-in-place capability of the differential drive base as a fourth "yaw" DoF, the students extend their IK code so as to be able to grasp small objects at known locations within a full 3D workspace.

Visual servoing, Fig. 4: The mast and a monocular USB camera are added to the robot and a grippable object with a known height and distinct color is supplied. The object may be placed on the ground anywhere in the camera's field of view and the students must develop a visual servoing system to automatically locate the object, drive the robot to it, grasp it, and then return to the the starting location. The task is broken down into hue-based blob detection using OpenCV and closed-loop visual servoing using the mobility and manipulation capabilities from prior modules.

Integration, Figs. 5 and 7: The course culminates with a module that integrates capabilities from multiple prior modules. For undergraduates this is an autonomous object-gathering challenge: a walled environment is constructed with obstacles and objects. A map is supplied at runtime, but due to odometric drift the students still need to devise on-line strategies for localization and object detection. Graduate students chose a project which connects to recently published work, e.g., [29], [30]. Additional hardware including the Kinect, and software like ROS, are made available to the students for this final module.

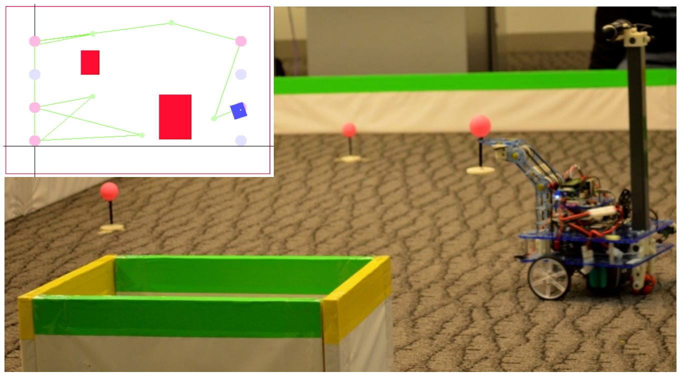

Fig. 7: A final autonomous object-gathering challenge for undergraduates requires integration of all prior modules.

V. Results and Analysis

The above curriculum modules have so far been used with six undergraduate student groups and two graduate student groups, with two to three students per group. (The OHMM lab kits are designed to support up to four students each.) While most students were computer science majors, the groups also included majors in digital art, music technology, cognitive psychology, and bioengineering. All students had prior programming experience with high-level languages like Java, but many had no prior experience in C. Nevertheless, all groups but one were successful with the differential drive control module programmed in embedded C on the LLP. All groups successfully implemented the arm kinematics and visual servoing modules; all but one were successful with local obstacle avoidance and the final integrative module; and all but three completed map-based global navigation. That final result may suggest that the local and global navigation topics should be separated into two modules, as some groups spent too much time on the former and sacrificed the latter. Assuming that split, eight groups participated in six modules each with a total of about 88% success. Though variation is to be expected across student populations, this result suggests that the platform and curriculum can motivate and support the teaching of a broad set of robotics algorithms to students with diverse backgrounds.

At the end of the semester students were asked to participate in Likert scale and short answer surveys. With an overall response rate of 69%, 88% of responses agreed that the course was intellectually challenging, and 81% agreed that the assignments based on the new platform helped them to learn. One interesting result was that even though the lack of all-hours access to the lab space was mitigated by the portable take-home kits, 50% of all students still felt the need for more scheduled lab time. Students also commented that the course helped them build skills which could increase their employment opportunities, and some of the undergraduates said that it motivated them to consider applying for graduate study in robotics.

VI. CONCLUSION AND FUTURE WORK

The open hardware mobile manipulator is a new open platform for teaching robotics software and algorithms that uniquely combines low- and high-level processors, a 3DoF arm and gripper, a mast-mounted camera, portability, digital manufacturing, and the possibility to be incrementally assembled in different configurations. Open design combined with digital manufacturing enables the community to use and update the platform as needed. Alternatives exist should any of the key components become unavailable, and the design can be updated for future advances in the field.

The educational efficacy of this platform has been assessed, with a relatively diverse group of undergraduate and graduate students achieving nearly 90% success in transforming robotics algorithms—differential drive control and odometry, obstacle avoidance and map-based navigation, inverse kinematics and grasping, visual servoing for object collection, and an integrative project—from concept to working code.

Next steps include implementing some refinements to the platform (such as arm joint sensing), and new curriculum modules for teaching topics such as the combination of visual perception and compliant contact (e.g., locating, grasping, opening, and entering a small sliding door). The platform is also intended to be used in additional undergraduate and graduate courses and in a day program to motivate robotics research by introducing visual servoing to middle- and high-school students (the latter is in collaboration with the John's Hopkins University Center for Talented Youth, c.f. [31]).

ACKNOWLEDGMENT

The College of Computer Science at Northeastern University funded the materials for the courses. This work was supported in part by NSF CAREER award 1149235.

REFERENCES

[1] The Free Software Foundation, "What is free software?" http://www.gnu.org/philosophy/free-sw.html, 2012.

[2] M. Lytras and W. Scacchi, Eds., *IEEE Transactions on Education, Issue on Open Source Software for Education*, vol. 50, no. 4, 2007.

[3] The OpenCourseWare Consortium, "What is opencourseware?" http://www.ocwconsortium.org/en/aboutus/whatisocw, 2012.

[4] A. M. Dollar, D. Rus, and P. Fiorini, "Robotics courseware," http://roboticscourseware.org, 2012.

[5] E. Möller, B. M. Hill, A. Beesley, M. Garlick, and E. Stark, Eds., *Open Source Hardware (OSHW) Statement of Principles 1.0*, 2012.

[6] M. Vona, "The open hardware mobile manipulator (OHMM) project," http://www.ohmmbot.org, 2011.

[7] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, "The iCub humanoid robot: an open platform for research in embodied cognition," in *PerMIS: Performance Metrics for Intelligent Systems Workshop*, 2008.

[8] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, C. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proc. 9th Conference on Autonomous Robot Systems and Competitions*, 2009, pp. 59–65.

[9] D. Hong, D. Lee, and J. Han, "DARwIn-OP: An open platform, miniature humanoid robot for research, education and outreach," http://www.romela.org/main/DARwIn_OP:_Open_Platform_Humanoid_Robot_for_Research_and_Education, 2012.

[10] Willow Garage, "TurtleBot," http://www.willowgarage.com/turtlebot, 2011.

[11] CMU Tekkotsu Project, "Calliope mobile manipulation platform," http://chiara-robot.org/Calliope, 2012.

[12] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal*, 2000.

[13] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *Open-Source Software workshop at IEEE ICRA*, 2009.

[14] Locomotec UG, "Kuka youBot," http://youbot-store.com/, 2012.

[15] G. Gallagher, "The bilibot project," http://www.bilibot.com, 2011.

[16] Crustcrawler, "Nomad," http://www.crustcrawler.com, 2011.

[17] A. Valera, M. Vallés, L. Marín, A. Soriano, A. Cervera, and A. Giret, "Application and evaluation of lego NXT tool for mobile robot control," in *IFAC World Congress*, 2011.

[18] A. Gage and R. R. Murphy, "Principles and experiences in using LEGO to teach behavioral robotics," *ASEE/IEEE Frontliners in Edu.*, 2003.

[19] D. Rus (Course Coordinator), "Robotics: Science and systems I (6.141)," http://courses.csail.mit.edu/6.141, 2011, MIT EECS.

[20] B. Smart, "CSE 550: Mobile Robotics," http://classes.engineering.wustl.edu/cse550/, 2012, Washington University.

[21] B. C. Dickinson, O. C. Jenkins, M. Moseley, D. Bloom, and D. Hartmann, "Roomba Pac-Man: Teaching autonomous robotics through embodied gaming," in *AAAI Symposium on Robot and Robot Venues: Resources for AI Education*, 2007.

[22] "Ponoko: the world's easiest making system," http://www.ponoko.com.

[23] S. Audet, "javacv: Java interface to OpenCV and more," http://code.google.com/p/javacv/, 2012.

[24] O. Project, "OpenKinect," http://openkinect.org, 2012.

[25] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *IEEE International Conference on Robotics and Automation*, 2011.

[26] Pololu, Inc., "Orangutan SVP-1284 robot controller," http://www.pololu.com/catalog/product/1327, 2012.

[27] Texas Instruments, "PandaBoard," http://pandaboard.org, 2012.

[28] V. Lumelsky and A. Stepanov, "Dynamic path planning for a mobile automaton with limited information on the environment," *IEEE Trans. Automatic Control*, pp. 1058–1063, 1986.

[29] H. R. Chitsaz, S. M. LaValle, D. J. Balkcom, and M. T. Mason, "Minimum wheel-rotation paths for differential-drive mobile robots," *Int. Journal of Robotics Research*, vol. 28, no. 1, pp. 66–80, 2009.

[30] A. Cherubini, F. Chaumette, and G. Oriolo, "Visual servoing for path reaching with nonholonomic robots," *Robotica*, vol. 29, no. 7, pp. 1037–1048, 2011.

[31] D. Rus, M. Vona, and K. Quigley, "Eye-in-hand visual servoing curriculum for young students," *IEEE Robotics and Automation Magazine*, vol. 17, pp. 116–117, 2010.

Marsette Vona [M '06] completed his B.A. at Dartmouth College in 1999, where he won the CRA Outstanding Undergraduate Researcher award for his work with Daniela Rus on self-reconfiguring robots. He completed his M.S. and Ph.D. in EECS at MIT in 2001 and 2009. He spent 2001-2003 at NASA/JPL building 3D user interfaces for the Mars Exploration Rover mission, for which he was a recipient of the NASA Software of the Year award in 2004. In 2010, he joined the faculty at the College of Computer and Information Science at Northeastern University as Assistant Professor, where he teaches robotics, geometric algorithms, and graphics. He founded the Geometric and Physical Computing group and recently received the NSF CAREER award to study 3D perception and compliant contact.

Shekar NH received his Bachelor's degree in Electronics and Communication in 2011 from M.S. Ramaiah Institute of Technology, India. He is currently a candidate for the M.Sc. in Information Assurance at Northeastern University. He is strongly interested in artificial intelligence, robotics, and cryptography.