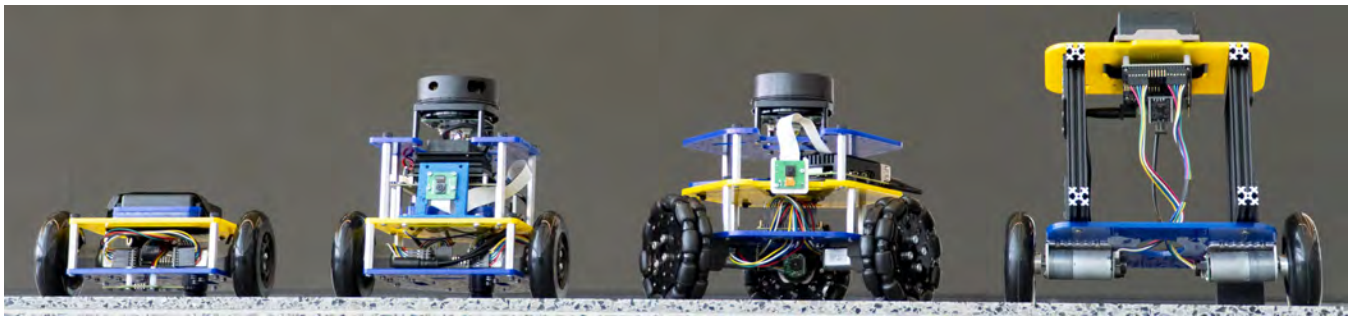
2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

MBot: A Modular Ecosystem for Scalable Robotics Education

Peter Gaskell Jana Pavlasek Tom Gao Abhishek Narula Stanley Lewis Odest Chadwicke Jenkins



***Abstract*— The Michigan Robotics MBot is a low-cost mobile robot platform that has been used to train over 1,400 students in autonomous navigation since 2014 at the University of Michigan and our collaborating colleges. The MBot platform was designed to meet the needs of teaching robotics at scale to match the growth of robotics as a field and an academic discipline. Transformative advancements in robot navigation over the past decades have led to a significant demand for skilled roboticists across industry and academia. This demand has sparked a need for robotics courses in higher education, spanning all levels of undergraduate and graduate experiences. Incorporating real robot platforms into such courses and curricula is effective for conveying the unique challenges of programming embodied agents in real-world environments and sparking student interest. However, teaching with real robots remains challenging due to the cost of hardware and the development effort involved in adapting existing hardware for a new course. In this paper, we describe the design and evolution of the MBot platform, and the underlying principals of scalability and flexibility which are keys to its success.**

I. Introduction

The growing popularity of robotics and AI has led to an increasing demand for new courses which integrate cutting-edge concepts at all levels of undergraduate and graduate studies. And what is a robotics course without real robots? The use of real robot platforms in robotics curricula has the potential to motivate and inspire students, while providing practical grounding for the course concepts. However, many commercially available platforms are prohibitively expensive and lack the flexibility to support the diverse needs of students and instructors. Open-source projects are often more customizable and affordable, but require a significant upfront development effort and consistent maintenance, making their integration into curricula impractical for many institutions.

The **MBot** is a low-cost, flexible platform for robotics education at the undergraduate and graduate levels designed at the University of Michigan. The MBot is designed with modularity in mind, and includes a number of possible robot configurations. The platform is the result of over a decade of teaching robotics and has served six courses and over 1,400 students since 2014. The platform has been used to teach courses on robot localization, control, planning, and introductory programming, from the first year of undergraduate study through graduate-level courses. The MBot has also been used as part of a distributed teaching initiative to offer robotics courses at collaborating institutions.

The MBot can be constructed from commercial, off-the-shelf components. The platform is driven by a simple, versatile backbone, called the *MBot Robotics Control Board*. The board can be used in a number of distinct configurations, from a low-cost version with basic sensing capabilities to a platform capable of advanced autonomy using a 2D Lidar and an RGB camera. The flexible design enables a configurable chassis and sensor suite with minimal core hardware modifications.

Central to the MBot's versatility as an educational platform is a comprehensive set of open-source software tools and a custom API for high-level programming, making it suitable for use in both advanced and introductory courses. The robot can be programmed through multiple modes depending on the needs of the user and on the application. The software supports advanced autonomy applications such as mapping, localization, and perception through message-passing frameworks. Alternatively, users can program the robot using the custom *MBot Bridge API*, which provides a simple, synchronous interface to the autonomy processes. The MBot platform also features a custom web application for remote control and visualization, which can be accessed from any personal computer.

The MBot has built on this design to realize a low-cost mobile robot platform that has been used to train students in robotics and AI at the University of Michigan and our collaborating colleges. The MBot platform was designed for seamless adoption into higher education curricula. Specifically, the MBot aims to meet the needs of teaching robotics at scale to match the growth of robotics as a field and an

Authors are with the Robotics Institute and Robotics Department, University of Michigan, Ann Arbor, USA 48109-2106. [pgaskell, pavlasek, zimingg, abnarula, stanlew, ocj]@umich.edu.

Fig. 1: Ten years of teaching with the MBot.

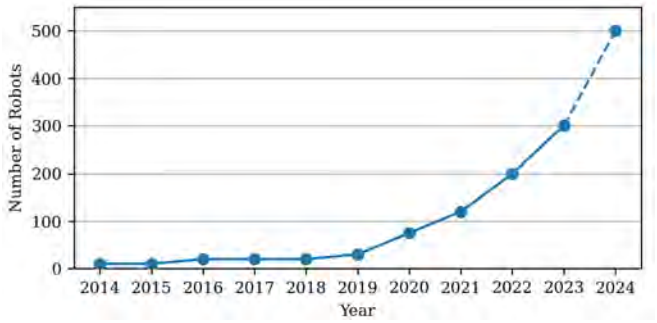


Fig. 2: Growth of MBot fleet over time to meet teaching needs for robotics classes at Michigan. The dotted line represents projected numbers.

TABLE I: MBot Bill of Materials

		MBot Version			
Component	Approx. Cost	Basic	Classic RPi	Classic Jetson	Omni RPi
Control board	$15	✓	✓	✓	✓
Pico	$5	✓	✓	✓	✓
Motors	$15	✓	✓	✓	✓
Battery	$25	✓	✓	✓	✓
Basic chassis	$20	✓	✓	✓	
Omni chassis	$40				✓
Basic wheels	$20	✓	✓	✓	
Omni wheels	$80				✓
RPi 4 kit	$85		✓		✓
Jetson kit	$175			✓	
Lidar	$100		✓	✓	✓
Total:		$100	$285	$375	$385

academic discipline [1], spanning all levels of undergraduate and graduate experiences. More information on the MBot platform is available at: https://mbot.robotics.umich.edu/.

II. The Development of the MBot

The MBot has grown into an ecosystem of platforms and tools suitable for instruction of a variety of courses in higher education in response to lessons learned over nearly a decade of teaching. Figure 1 shows the evolution of the platform. The Maebot platform [2] was originally developed for instruction of a final-year undergraduate course on autonomous robotics at the University of Michigan in 2014. In 2016, the MBot was designed as a lower-cost version of the original design and expanded to teach the graduate-level course *Robotic Systems Lab* as part of the University of Michigan's graduate degree in robotics.

Early generations of the platform focused on affordability and the incorporation of features suitable for supporting the learning objectives for advanced robotics courses, such as a 2D Lidar for mapping and navigation. The COVID-19 pandemic sheltering necessitated a new generation of the platform with enhanced tools suitable for remote learning efforts. Students in robotics lab classes were able to successfully achieve course learning objectives in their home environments with typical compute resources when provided only with a basic MBot kit of parts shipped to them and remote support by the teaching staff.

The founding of the University of Michigan Robotics undergraduate degree [1] in 2022 expanded the number of robotics courses offered, introducing a number of new courses with hands-on components. Growing enrollment introduced new constraints on the MBot platform, including that the platform support courses offered early in the degree with minimal or no programming prerequisites. This sparked the development of the latest generation of the MBot family of platforms which includes an ecosystem of tools for interacting with the robot at different levels, including a custom web application for visualization and an API. Multiple additional configurations of the robot, including a low-cost budget version and an omnidrive platform, were also introduced as part of the expanded robotics curriculum. Currently all courses are using the latest generation of MBot. Figure 2 shows the growth of the MBot fleet over time.

The MBot has also been used as part of a *distributed teaching collaborative* between institutions. Building off of lessons learned during the pandemic when robots were used in students' homes, the MBots have been deployed at Berea College, Howard University, and Morehouse College to teaching autonomous robotics to undergraduate students.

III. The MBot Platform

The guiding principles for the design of the platform are as follows:

- a low-cost basic version,
- low barrier to entry for students and instructors,

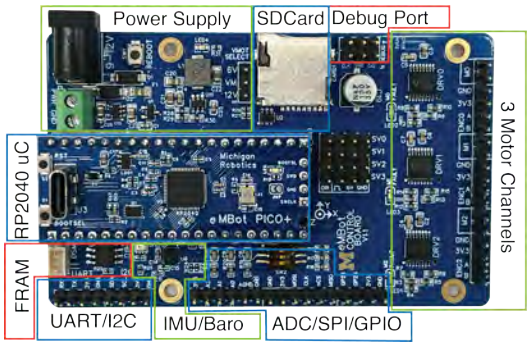


Fig. 3: Image of the Robotics Control Board and custom RP2040 microcontroller based off the Raspberry Pi Pico. The subsystems of the board are labeled. Each motor channel can drive a brushed DC motor and read a quadrature encoder.

- a modular, repairable, configurable & extensible design,
- compatibility with advanced robotics tools (e.g. ROS [3], [4]) for graduate education & research.

The cost of the three most common versions of the platform (the Basic, the Classic, and the Omni) can be found in Table I. The hardware design is described below.

A. Configurable Hardware Design

The Basic version of the MBot platform forms the foundation for all platforms in the MBot family. It is centered around the MBot control board and can use either 12V or 6V 20mm motors and comes complete with a chassis, motor mounts, wheels and magnetic encoders. The Basic model is powered by a 12V 3Ah Li-Ion battery pack, and uses a 7-segment line sensor for line following and navigational tasks. The BOM cost for the Basic model is under $100 USD as of Fall 2023. The chassis plates have mounting holes for adding additional sensors, and creating a new chassis can be done quickly with a laser cutter. Building on the Basic model, the Balance add-on provides the parts to turn the Basic model into a mobile inverted pendulum robot and includes a redesigned chassis, additional hardware, and larger 12V 25mm motors for enhanced speed and torque.

The Classic version is geared towards advanced robotic applications, specifically around visual processing, mapping, and navigation. It includes a single-board computer as its high-level computing module. Currently supported computing boards include the Raspberry Pi 4B+, Pi 5, and the NVidia Jetson Nano. The platform also supports additional sensors, including a 2D Lidar module to enable mapping and localization, and a camera allows for vision tasks such as visual odometry and object detection. The Classic can be configured with different drive types, including Omnidrive and Ackermann versions.

B. MBot Control Board

The foundation of the MBot platform is the MBot Control Board, shown in Figure 3. The board uses a RP2040 based microcontroller, and is compatible with the Raspberry Pi Pico family of microcontroller modules. It allows for control of three brushed DC motors with current feedback and relative position feedback using a quadrature encoder. Each motor driver (TI DRV8876) can deliver 1.3A continuous current. The motor voltage can be selected by an onboard jumper between the applied DC voltage and the onboard 6V, 4A voltage regulator. The control signals for the 3 motors, along with two additional motor control signals are broken out so the board can be used to drive four external motor drivers. Any of the the motor channels can be configured to drive a hobby servo motor instead.

The board contains various sensors and storage capabilities. A 9 degree of freedom MEMS Inertial Measurement Unit (Bosch BMI160B) with gyroscope, accelerometer, and magnetometer (Bosch BMM150) components performs data fusion to yield pose estimates. A barometer (Bosch BMP280) is also available to estimate altitude relative to an initial pressure reading. A nonvolatile memory chip is included to store calibration data and parameters onboard the controller, and a SD card connector enables data logging without a host computer.

To connect external sensors and other devices, the board has dedicated ports for communication over common serial protocols including I2C and SPI. The three analog to digital converters can be switched from measuring the current draw of the motors to measuring an external voltage. Additionally, the debug pins near the card reader allows for soft/hard resets of the board, loading of different programs onto the board, and software debugging connections to enable debugging tools such as the GNU Debugger.

C. MBot Firmware

The MBot provides an accessible and user-friendly firmware solution for the RP2040 microcontroller. It offers firmware solutions in both C and MicroPython, enhancing the ease of implementing new MBot configurations, regardless of the user's programming or embedded development expertise. The clear separation between low-level interface and communication protocols and user-space code allows the robot to be customized for specific learning objectives without specialized, device-specific knowledge. Meanwhile, experienced users can access the underlying driver codebase for further enhancements and modifications if desired and the community and support for the low-cost RP2040 microcontroller makes enhancing the codebase accessible. This distinct advantage positions the MBot Control Board as a preferred choice over alternative options such as ST Nucleo boards which require substantial embedded experience to master, or the higher cost Beaglebone Blue, which while a capable platform for robotics education [5], is overpowered for simple motor control and sensor interfacing tasks.

The firmware of the MBot Control Board facilitates seamless communication with other devices through two virtual USB serial ports. One of these ports is dedicated to message-based communication with higher-level computing devices (e.g., a Raspberry Pi) using the ROSSerial protocol [6]. This capability enables the MBot Control Board to engage in communication using a publish/subscribe approach. When coupled with one of the serial communication service components of the MBot software ecosystem, the MBot Control Board can effortlessly publish and subscribe to messages used by higher-level computing devices. The second serial port

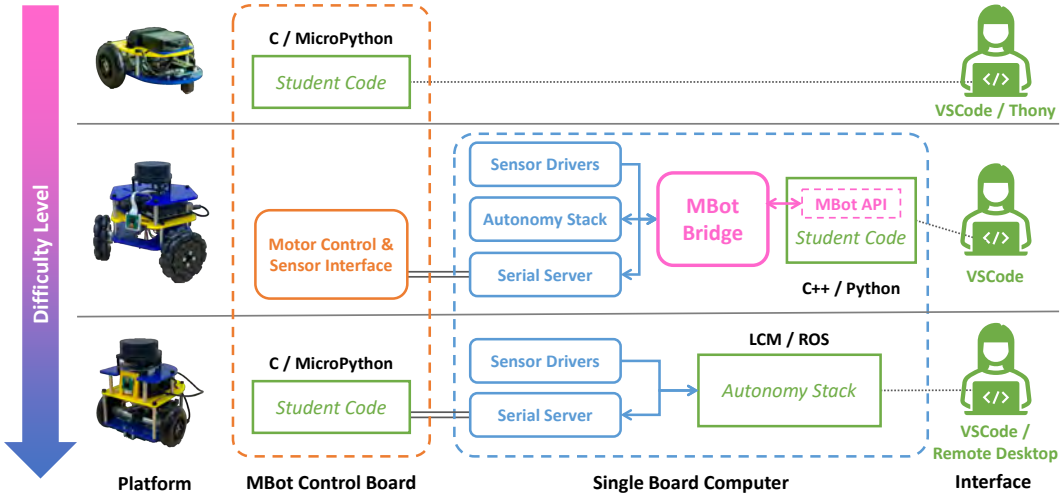


Fig. 4: Possible software configurations for different levels of robotics courses. The Basic configuration with no single board computer can be programmed directly for basic applications using C or MicroPython (top). Classic and Omni versions can be programmed through the MBot Bridge API, which provides a simple interface for programming the MBot (middle). Advanced applications can interface directly with the message passing framework (bottom).

is reserved for user-space code, allowing users to transmit human-readable data for purposes such as status reporting, ad-hoc data collection, or debugging.

Moreover, the MBot Control Board offers support for MicroROS [7], enabling DDS communications with ROS2 systems. As ROS2 continues to gain prominence in the field of robotics research, this feature not only future-proofs the MBot Control Board but also enhances its compatibility with existing robotics systems. This forward-thinking approach ensures that the MBot remains a versatile and adaptable platform for robotics education and research.

IV. The MBot Architecture

The MBot software architecture is designed to be both flexible and easy to use. The architecture enables users to select an interface based on the desired application and level of difficulty, making the platform suitable for introductory and advanced courses. The MBot comes equipped with a full asynchronous software stack, including sensor drivers, mapping and localization, and path planning, in addition to an easy-to-use synchronous API. The robot also has a custom web app which can be used for visualization and basic control. For advanced applications, the robot can be programmed by interfacing directly with the core software stack through an asynchronous message passing framework (e.g. LCM [8] or ROS [3], [4]). For single-threaded, synchronous applications, such as introductory programming assignments or quick prototyping, the *MBot Bridge API* provides a simple interface to the MBot software stack. The software for various configurations is shown in Figure 4.

The typical workflow for programming the MBot uses the single-board computer running a Linux-based operating system as the development environment. Users connect to the remote computer using a remote session on a local IDE (e.g. VSCode's Remote extension), or through a remote desktop. This enables the robots to be programmed with minimal configuration on a personal computer.

In the following sections, we describe the functionalities available in the MBot software stack as well as two key tools we have developed for easy interaction with the MBots: the MBot Bridge API and the MBot Web App.

A. The MBot Software Stack

The software stack consists of drivers, utilities, and autonomy programs built on the asynchronous message-passing communication protocol Lightweight Communications and Marshalling (LCM) [8]. The available functionalities as of the time of writing include a driver for the Lidar, a serial interface to the MBot Control Board, a path tracker, and a SLAM node which employs Monte-Carlo Localization [9], [10]. The SLAM implementation enables on-the-fly mode switching between full mapping and localization, localization only, and idle modes, which can be controlled through the web app. Note that the SLAM system is not intended to be a state-of-the-art implementation, but rather is instead intended to represent a starting point for exploration and investigation. Its implementation and visualization is similar to the ROS navigation stack [11], but made as straightforward as possible to aid in student comprehension. We also note that while the MBot software uses LCM, the platform is also compatible with other frameworks such as the Robot Operating System (ROS) [3], [4].

For single-threaded use cases, the MBot software can be used as a tool for developing downstream applications. For example, students can use the MBot's SLAM for mapping and localization through the web app, then implement an autonomous navigation algorithm by accessing the localization information through the API (see the middle row of Figure 4). For advanced use cases, any process in the MBot's autonomy stack can be disabled and replaced with custom code. For example, for educational modules on mapping and localization, students can implement their own SLAM algorithm by interfacing directly with LCM or ROS.

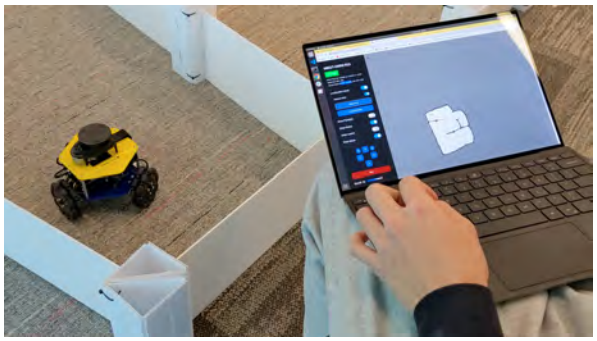


Fig. 5: The MBot Web App in action. The web app can be accessed through a browser on any device connected to the robot's network, and enables teleoperation of the robot, control over the SLAM state, and visualization.

B. MBot Bridge API

The MBot supports synchronous programming in C++ and Python through the MBot Bridge API. The API provides a simple interface for reading robot data and sending control commands in single-threaded programs. The API depends on the MBot Bridge Server, which manages incoming messages from the software stack and stores them in queues. The server exposes a websocket-based protocol using a custom JavaScript message definition inspired by ROS Bridge [12]. An added utility of the MBot Bridge server and API is that the websocket-based interface enables any device on the robot's network to communicate using the defined protocol. This means that the MBot Bridge can be used remotely from a personal laptop, and enables inter-robot communication within a robot fleet.

C. MBot Web App

Core to the philosophy behind the MBot system is that the robots should be user-friendly from the perspective of a typical undergraduate student. The MBot Web App provides visualization and control of the platforms. The app is hosted directly on the robot's single-board computer, and can be accessed from a browser from any device (e.g. personal computer or cell phone) connected to the same network as the robot. As such, interfacing with the robot through the web app requires no installation or technical prerequisites once installed on the robot, unlike other common visualization tools such as RViz.

The web app includes a driving interface for sending robot velocity commands either through a joystick or keyboard control. It also allows the user to toggle between idle mode, localization mode, or full SLAM (mapping and localization) mode, as well as reset the map. The web app displays the current map, the robot position in the map, the laser scan, and any published paths. A visualization of the web app is shown in Figure 5.

V. Teaching with the MBot

A summary of the courses which have been taught by the MBot thus far can be found in Table II. Below, we highlight two flagship courses based around the MBot for advanced (graduate or senior undergraduate) and beginner (first year undergraduate) students.

1) Robotics Systems Lab: The Robotics Systems Lab is one of the core classes in the University of Michigan Robotics graduate program. The course utilizes the MBot extensively in order to address several learning objectives, most notable those surrounding statistical inference, controls, and trajectory planning. Students implement a particle filter based Markov Random Field SLAM solution utilizing the onboard LiDAR and odometry sensors in order to autonomously explore, then subsequently navigate, a previously unseen labyrinth. For this class, students are exposed to the full breadth of the MBot software stack. Software is written in C/C++ using the Lightweight Communications and Marshalling (LCM) [8] messaging framework. The low-level command and control code bases are fully available for students to implement or modify as necessary. The configuration used is shown in the bottom row in Figure 4. As of this writing, 722 students have successfully completed the Robotics Systems Lab course, more than 50 of whom have done so as hybrid or fully remote students.

2) Hello, Robot! Introduction to Robotics and AI: As part of the University of Michigan's Robotics undergraduate degree, an introductory programming class based on the MBot for first-year students was developed. The course has no prerequisites and covers introductory programming through projects on the MBot Omni, including wall following, bug navigation, path planning, and image classification. The course uses the MBot Bridge API as a simple interface to the autonomy software. The configuration used is shown in the middle row in Figure 4. This course has been adapted and offered at Berea College, Howard University, and Morehouse College.[1]

VI. Related Work

The study of robotics and education has a long history of scientific exploration. This history dates back to the seminal work of Papert [13] in his book *Mindstorms*. This work established the foundational value proposition for computational literacy as a critical need for society. Through the invention of the LOGO Programming Language, Papert et al. [14] emphasized the importance of learning computer programming through robotics, where through "embodiment as the physical computer, computation opens a vast universe of things to do."

Research into robotics education has grown rapidly since these early beginnings, especially for undergraduate and graduate education. We refer the reader to the survey by Miller and Nourbakhsh [15] for an overview of robotics education up through the mid-2000s. There is a wealth of studies into methods and efficacy for robotics education across various levels [16], [17], and its effects and benefits for student diversity [18]. With their groundbreaking Robotics Engineering major at Worcester Polytechnic Institute (WPI), Gennert et al. [19] made the first major steps into explorations creating a whole curriculum around this discipline of robotics. Our design of the MBot and the Michigan Robotics curriculum

[1]Details of the course *Introduction to AI and Programming* are available at: https://hellorob.org/.

TABLE II: Courses taught with the MBot Platform.

Course	Institution	Years Offered	Student Level	Course Content
Robotics Systems Lab	UM	2014 – Pres.	Grad.	Control systems (Balance), localization & mapping, path planning (Classic)
Autonomous Robotics	UM	2014 – Pres.	Undergrad.	Localization & mapping, path planning, independent design project (Classic)
SLAM & Navigation	UM	2022 – Pres.	Undergrad.	Localization & mapping, path planning (Classic)
Intro to AI & Programming	UM	2021 – Pres.	Undergrad.	Intro. to C++ & Python, feedback control, autonomous navigation, image classification (Omni)
Intro to AI & Autonomous Systems	BC	2021, 2023	Undergrad.	Feedback control, autonomous navigation, image classification (Omni)
Robotics: Autonomous Navigation	HU	2023	Undergrad.	Intro. to programming, wall following, autonomous navigation (Omni)

UM = University of Michigan; BC = Berea College; HU = Howard University

drew considerable inspiration and insight from the Robotics Engineering program at WPI.

The era of scalable and programmable robot platforms was catalyzed by Martin's *Robotics Explorations* book [20] and the introduction of the LEGO Mindstorms, borrowing from the spirit from Papert et al. Following the LEGO Mindstorms were a number of highly impactful mobile robot platforms for education. These platforms included the Parallax Scribbler [21], Sony AIBO [22], and modified versions of the iRobot Roomba [23]–[26], as forerunners to the Willow Garage Turtlebot [27]. Robotics education also saw the development of its own robot programming environments, such as Tekkotsu [28], more amenable to undergraduate teaching than common robot middleware frameworks.

Robots available today for higher education tend to fall into two categories. There are those where the focus is on and interfacing of basic sensors and the low level control of motors so that the user may program the robot to perform simple tasks like line following. These platforms, like VEX [29], the Pololu 3Pi+ [30] or the SparkFun RedBot [31], usually provide a simple way to program the robots, like with a custom library in the Arduino IDE. There is great educational value in working with a system like this to learn about programming embedded systems, but the sophistication of behaviours students are able to program into the robot is typically limited. These robots are therefore best suited to high school or lower level undergraduate education as a part of a beginner curriculum.

On the other end of the spectrum are robots like Turtlebot [32], Duckiebot [33], JetBot [34], MIT RaceCar [35], and MuSHR [36], all using Raspberry Pi or Nvidia Jetson, which are capable of interfacing with multiple high bandwidth sensors, cameras and Lidar, and utilize mature and sophisticated robotics software and algorithms in ROS. These robots are typically suited for students and researchers with a strong background in computers and programming as part of an advanced curriculum or in research. Other platforms such as Chronos [37] and The Robotarium [38] are suited for similarly experienced students for specific robotic problems such as autonomous driving and remote access (respectively). The MBot was designed to meet needs across undergraduate and graduate levels of education and be accessible and adaptable for many types of institutions, students, and courses.

TABLE III: Comparison of available robot platforms

Platform	Cost	Camera or LiDAR	Drive	Compute
Pololu 3Pi+	$150	–	DD	μC
SparkFun RedBot	$140	–	DD	μC
Parallax Scribbler	$180	–	DD	μC
Vex V5 Kit	$789	–	Multi	V5
DuckieBot	$450	Camera	DD	Pi4
JetBot	$260	Camera	DD	Nano
TurtleBot 4 Lite	$1200	Both	DD	Pi4
MuSHR	$1525	Both	Ack	Nano
MIT Racecar	$2600	Both	Ack	TX2
AgileX Limo	$2900	Both	Multi	Nano
MBot (Basic)	$100	–	Multi	μC
MBot (Classic)	$285	Both	Multi	Pi4
MBot (Classic)	$375	Both	Multi	Nano

DD = Differential; Ack = Ackermann; Multi = Reconfigurable
μC = low-cost microcontroller; V5 = Vex V5 (CortexA9)
Pi4 = Raspberry Pi 4; Nano = Nvidia Jetson Nano
TX2 = Nvidia Jetson TX2

VII. Conclusion

We present the MBot ecosystem for robotics education at the undergraduate and graduate levels. The platform has been developed over nearly a decade of robotics education at the University of Michigan. It has proven successful at a number of courses across multiple institutions. Future work on the platform will involve working with partnering institutions to integrate it into more courses, and further expanding the suite of tools available.

VIII. Acknowledgments

This work was supported in part by Ford Motor Company, J.P. Morgan AI Research, Amazon Inc., Toyota Research Institute, the Sloan Foundation, an NSERC scholarship, and a generous donation from Roger Ehrenberg and Carin Levine-Ehrenberg. A full list of contributors is available is at: https://mbot.robotics.umich.edu/.

References

[1] O. C. Jenkins, J. Grizzle, E. Atkins, L. Stirling, E. Rouse, M. Guzdial, D. Provost, K. Mann, and J. Millunchick, "The Michigan Robotics undergraduate curriculum: Defining the discipline of robotics for equity and excellence," *arXiv preprint arXiv:2308.06905*, 2023.

[2] "MAEBot: An open-source robot platform for education and research," https://april.eecs.umich.edu/maebot/, accessed: 2023-01-13.

[3] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[4] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[5] T. Bewley, J. Strawson, S. Ostovari, and H. C. Briggs, "Leveraging open standards and credit-card-sized linux computers in embedded control & robotics education," in *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015, p. 0801.

[6] P. Bouchier, "Embedded ros [ros topics]," *IEEE Robotics & Automation Magazine*, vol. 20, no. 2, pp. 17–19, 2013.

[7] K. Belsare, A. C. Rodriguez, P. G. Sánchez, J. Hierro, T. Kołcon, R. Lange, I. Lütkebohle, A. Malki, J. M. Losa, F. Melendez, M. M. Rodriguez, A. Nordmann, J. Staschulat, and J. von Mendel, "Micro-ROS," in *Robot Operating System (ROS): The Complete Reference (Volume 7)*, ser. Studies in Computational Intelligence, A. Koubaa, Ed. Cham: Springer International Publishing, 2023, pp. 3–55.

[8] A. S. Huang, E. Olson, and D. C. Moore, "LCM: Lightweight communications and marshalling," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 4057–4062.

[9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[10] J.-S. Gutmann and D. Fox, "An experimental comparison of localization methods continued," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1. IEEE, 2002, pp. 454–459.

[11] "ros-planning / navigation," https://github.com/ros-planning/navigation, 2024.

[12] C. Crick, G. Jay, S. Osentoski, B. Pitzer, and O. C. Jenkins, "Rosbridge: ROS for non-ROS users," in *Robotics Research: The 15th International Symposium ISRR*. Springer, 2017, pp. 493–504.

[13] S. Papert, "Mindstorms: children, computers, and powerful ideas," 1980.

[14] C. Solomon, B. Harvey, K. Kahn, H. Lieberman, M. L. Miller, M. Minsky, A. Papert, and B. Silverman, "History of logo," *Proceedings of the ACM on Programming Languages*, vol. 4, no. HOPL, pp. 1–66, 2020.

[15] D. P. Miller and I. Nourbakhsh, "Robotics for education," in *Springer handbook of robotics*. Springer, 2016, pp. 2115–2134.

[16] C. A. Berry, S. L. Remy, and T. E. Rogers, "Robotics for all ages: a standard robotics curriculum for k-16," *IEEE Robotics & Automation Magazine*, vol. 23, no. 2, pp. 40–46, 2016.

[17] M. Anderson, A. McKenzie, B. Wellman, M. Brown, and S. Vrbsky, "Affecting attitudes in first-year computer science using syntaxfree robotics programming," *ACM Inroads*, vol. 2, no. 3, pp. 51–57, 2011.

[18] M. Gini, J. Pearce, and K. Sutherland, "Using the sony aibos to increase diversity in undergraduate cs programs," in *9th International Conference on Intelligent Autonomous Systems, IAS 2006*, 2006, pp. 1033–1040.

[19] M. A. Gennert and C. B. Putnam, "Robotics as an undergraduate major: 10 years' experience," in *2018 ASEE Annual Conference & Exposition*, 2018.

[20] F. G. Martin, *Robotic explorations: A hands-on introduction to engineering*. Prentice Hall PTR, 2000.

[21] J. Summet, D. Kumar, K. O'Hara, D. Walker, L. Ni, D. Blank, and T. Balch, "Personalizing cs1 with robots," *ACM SIGCSE Bulletin*, vol. 41, no. 1, pp. 433–437, 2009.

[22] M. M. Veloso, P. E. Rybski, S. Lenser, S. Chernova, and D. Vail, "Cmrobobits: Creating an intelligent aibo robot," *AI magazine*, vol. 27, no. 1, pp. 67–67, 2006.

[23] B. C. Dickinson, O. C. Jenkins, M. Moseley, D. Bloom, and D. Hartmann, "Roomba pac-man: Teaching autonomous robotics through embodied gaming." in *AAAI Spring Symposium: Semantic Scientific Knowledge Integration*, 2007, pp. 35–39.

[24] M. Conbere and Z. Dodds, "Toys and tools: Accessible robotics via laptop computers." American Association for Artificial Intelligence, 2007.

[25] M. Lapping-Carr, O. C. Jenkins, D. H. Grollman, J. Schwertfeger, and T. Hinkle, "Wiimote interfaces for lifelong robot learning." in *AAAI Spring Symposium: Using AI to Motivate Greater Participation in Computer Science*, 2008, pp. 61–66.

[26] B. Tribelhorn and Z. Dodds, "Evaluating the roomba: A low-cost, ubiquitous platform for robotics research and education," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 1393–1399.

[27] R. Amsters and P. Slaets, "Turtlebot 3 as a robotics education platform," in *Robotics in Education: Current Research and Innovations 10*. Springer, 2020, pp. 170–181.

[28] D. S. Touretzky and E. J. Tira-Thompson, "Tekkotsu: A framework for aibo cognitive robotics," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, no. 4. Citeseer, 2005, p. 1741.

[29] "VEX Robotics," https://www.vexrobotics.com/, accessed: 2023-09-12.

[30] "Pololu 3pi+," https://www.pololu.com/category/300/3pi-plus-2040-robot, accessed: 2023-09-15.

[31] "Sparkfun RedBot," https://www.sparkfun.com/products/12649, accessed: 2023-09-15.

[32] "TurtleBot 4: Robotics learning platform," https://clearpathrobotics.com/turtlebot-4/, accessed: 2023-01-13.

[33] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang *et al.*, "Duckietown: an open, inexpensive and flexible platform for autonomy education and research," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1497–1504.

[34] "Jetbot," https://jetbot.org/master/, accessed: 2023-09-15.

[35] "MIT Racecar," https://racecar.mit.edu/, accessed: 2023-09-15.

[36] S. S. Srinivasa, P. Lancaster, J. Michalove, M. Schmittle, C. Summers, M. Rockett, J. R. Smith, S. Chouhury, C. Mavrogiannis, and F. Sadeghi, "MuSHR: A low-cost, open-source robotic racecar for education and research," *CoRR*, vol. abs/1908.08031, 2019.

[37] A. Carron, S. Bodmer, L. Vogel, R. Zurbrügg, D. Helm, R. Rickenbach, S. Muntwiler, J. Sieber, and M. N. Zeilinger, "Chronos and crs: Design of a miniature car-like robot and a software framework for single and multi-agent robotics and control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 1371–1378.

[38] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1699–1706.