RoboEdu: Technical Report

Franklin Volcic (Honors Capstone), Jana Pavlasek (Mentor),
Odest Chadwicke Jenkins (Advisor)

Introduction

Background

Robotics is a quickly growing field. By the year 2027, it has been estimated that robotics will be a $43 billion dollar industry [1]. With a growth in robotics comes a need for roboticists, and robotics education. Schools, such as the University of Michigan, are beginning to develop robotics coursework for students wishing to pursue a degree in robotics. Robotics coursework also brings a need for educational robots that students can use within their courses. For example, the University of Michigan has developed what is called the *M-Bot* (see figure 1) which is an educational robot that students at varying technical levels can use when learning robotics. The M-Bot robot gives students the ability to write their own code and run it on a real robot, giving them the opportunity to see their code come to life. Further, the M-Bot offers a number of different ways to interface with the robot. One method in specific is through a website hosted by the robot itself, providing a graphical user interface that allows students to control the robot in different ways, such as driving the robot, and in addition to robot control, allow students to visualize different pieces of data coming from the robot. While this system provides students, especially at the introductory level, a great way to interact with the robot, the current platform suffers from a number of challenges, many of which are outlined later in this report. To solve some of the problems that the current M-Bot web based interface suffers from, we've developed a new interactive web framework for the M-Bot robot platform. This technical report first introduces the new M-Bot web framework along with some goals of the project. The report then details some of the problems addressed by the M-Bot web application. After an introduction to the project and the problems addressed, the report details a technical overview of the project, detailing the tools provided by the framework. The report is then concluded with a discussion of the framework as a whole.

Introduction to the M-Bot Web Framework

The new M-Bot web application provides a number of advantages over the old web application, benefitting both educators and students. For students, the new M-Bot web application makes it easy to interact with the robot in a reliable and visual way, providing tools to do everything from driving your robot, to visualizing data that is coming from the robot, such as visualizing a map that has been created by the robot. For educators, the new M-Bot web framework provides a robust set of tools that empower them to supercharge the development of their own tools for the robot, which run on the M-Bot web application, while also providing a set of prebuilt tools that can be easily installed on their M-Bot web application.

Why the Project Matters

The M-Bot web framework was developed in an attempt to create an open-source web based robotics framework that educators can use to ease the process of developing robotics education. Developing robots for education is not an easy nor inexpensive task. While R1 universities, like the University of Michigan, may be able to fund the development of robotics coursework, institutions without the same level of funding, such as minority serving institutions and tuition free institutions, may struggle to develop adequate robotics coursework. One goal of the M-Bot robot as a whole is to lower the barrier to entry for schools looking to introduce robotics coursework to their students. The M-Bot web framework is a tool that aims to align with these goals, providing a free to use web framework that educators can use to allow students to interact with their robots in a graphical and simple way. In additional to providing free tools for universities that cannot afford to develop robotics coursework the same way R1 universities can, another goal of this project was to provide a highly modular web framework that allows educators to easily develop their own robotics tools that run on the M-Bot web application, with the hope of creating an open-source ecosystem with a variety of prebuilt tools that can easily be installed to a students robot, which they can begin using immediately.

Figure 1: The M-Bot Robot

Problems addressed by and Features Included with the M-Bot Web Framework

This section discusses some of the problems that the M-Bot web framework attempted to solve, and provide insight on how the framework solved these problems. In addition, this section touches on some of the high level features that the M-Bot Web Framework included during development.

Lower the Barrier to Entry for Educational Robotics

One issue that educational robotics face is a high barrier to entry in terms of the amount of capital required to fund development of robot systems along with the amount of time required to develop these robotic systems. The new M-Bot web application addresses this by providing an out of the box solution for web based robotics tools that requires minimal configuration to set up. This simple install, tied with easy to install extensions for the web application, allows instructors to focus on developing course material for their students, and spend less time developing robots. This makes robotics education

far more accessible for institutions that don’t have the same amount of funding places like the University of Michigan have.

Current Solutions Lack Modularity and Customizability

There do exist some web based educational resources developed by the University of Michigan. While these tools work well for the exact context in which they were developed for, they lack the ability to be easily extended. This poses significant challenges when new features or tools are needed, as a large codebase which had been built for a specific use case needs to be changed. This was noted during the development of the new M-Bot web framework, and the framework was built using a package system, allowing tools to be created easily in isolation from other tools. This means instructors can easily build a custom tool and install it to the web application without jumping through numerous hoops like reworking a large codebase.

Open Source Development

There are a variety of tools that robotics educators may require between data visualizations, robot controls, and debugging tools. While a small team may be able to develop all of the necessary tools over a number of years, an open source community can help supercharge the development of different tools. This was a core idea that helped guide development of the M-Bot web framework. The framework has a number of built in tools that makes the open sourcing and distribution of tools built for the M-Bot web application a seamless process. The ultimate goal of this is to foster an open source community of contributors who build tools for the web application.

Technical Review of the M-Bot Web Framework

The M-Bot web framework comes with a variety of tools that make the lives of students, educators, and open source contributors easier. This section aims to provide an overview of these tools.

The M-Bot Frontend

One important aspect of this project was the user-facing application that students will interact with when using the M-Bot web application. The front-end is separated into two parts, application packages, and the app skeleton. On the M-Bot web application, a **package** is a web tool that is served by the M-Bot web application which is accessed via the app skeleton. The **app skeleton** is simply the application that manages displaying and serving the available packages. To help understand the difference, figure 2 showcases the app skeleton being used to navigate through different available packages, and figure 3 showcases an example package that allows a user to drive a robot. Currently, all packages use rosbridge to communicate with the robot. While it is not a requirement that packages use rosbridge, the current open sourced packages were designed assuming the user was using ROS or ROS2 to manage their robot, as is the case with the new M-Bot, which is shown in figure 1.

When a user installs the M-Bot web framework via the provided install command in the documentation, a drive control package is installed by default, alongside a settings package (see figure 4) which allows users to both install packages and uninstall packages all through the web application. This allows the user to begin testing out the web application out of the box.

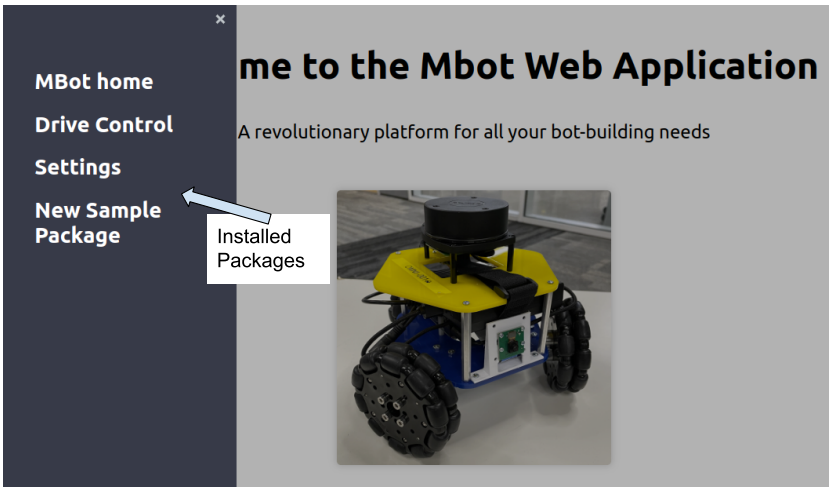

Figure 2: Navigating to different packages through the app skeleton

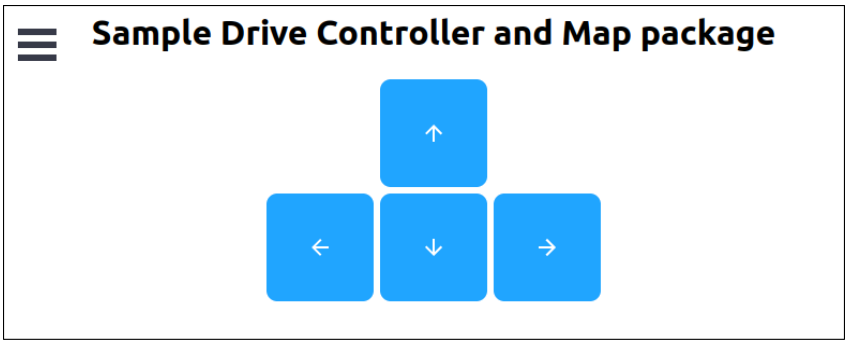

Figure 3: The default included Drive Control package

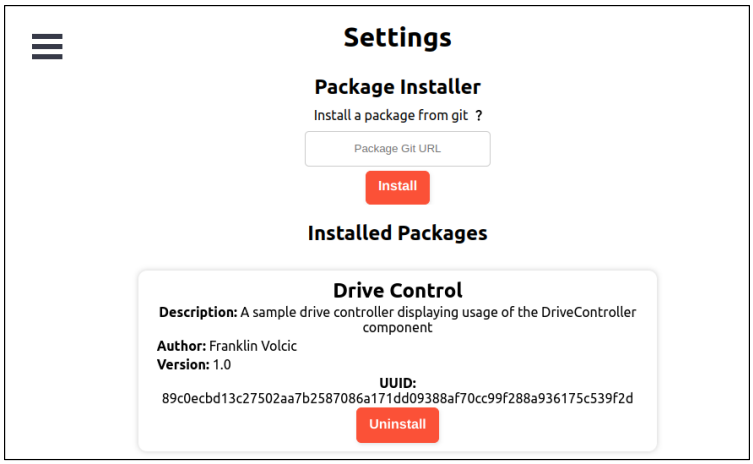

Figure 4: Default settings package

The M-Bot Backend

A backend system was required to power the user experience provided by the front-end. The M-Bot web application was built using a variety of tools to provide a robust system. The backend consists of two parts, the server, and the API. The server that hosts the web application is Nginx, which is a popular production server commonly used to serve static content to the web. The API was built using Flask, which is a Python-based microarchitecture. The Flask app is then served using a production WSGI server named Waitress. Waitress is a Python based WSGI server, meaning that the entire API can be served with Python being the only preinstalled tool required to run the API. This stack is easy to setup and maintain, while also having a low footprint, making it easy to run on low power devices such as a Raspberry Pi.

NPM Developer Packages

NPM, also known as Node Package Manager, is a popular tool for shipping Javascript source code to developers. The M-Bot web framework takes advantage of the simplicity of NPM to provide developers with a set of tools to aid the development and open source deployment of custom M-Bot packages.

The first of these tools is the M-Bot package template engine. The M-Bot package template engine is an application that can be executed through npx, and creates a template package for developers. The template provides the required starter code for a developer to create a React powered M-Bot package.

In addition to a template engine, the M-Bot web framework also has npm packages that provide javascript tools developers can use to speed up the development of their M-Bot packages. One such javascript tool provided with the M-Bot npm package is the getEndpoint function, which returns an endpoint that always points to the location of the M-Bot on the network, which is important as many students will be accessing the M-Bot via their IP addresses. Tools as such make developing an M-Bot package significantly easier for the developer.

Once a developer has completed a new package, they may wish to publish their application so others can use the package on their robot. Publishing a package to github can also be done through npx. Developers may run npx mbot-package-publisher, which will take care of compiling and deploying their package to github so others can use the package.

The M-Bot Web Framework Command Line Interface (CLI)

Included with the M-Bot web framework is a command line interface, which can run using the bash command “mbot-cli”. The CLI is installed by default when you install the framework via the installation script provided by in the install documentation. The CLI provides an easy to use interface to manage M-Bot packages. For instance, with the CLI, you may install a package from a folder, install packages from a git url, install

packages via a remote URL, generate a metadata file for a package, in addition to other operations. These operations aid both those who are operating a robot and need granular control over the packages on their robot, and also for developers who would need to test packages on the fly by allowing them to quickly install and uninstall development packages along with generating the necessary metadata required to install a package. Figure 5 highlights all the options that are available through the M-Bot CLI.

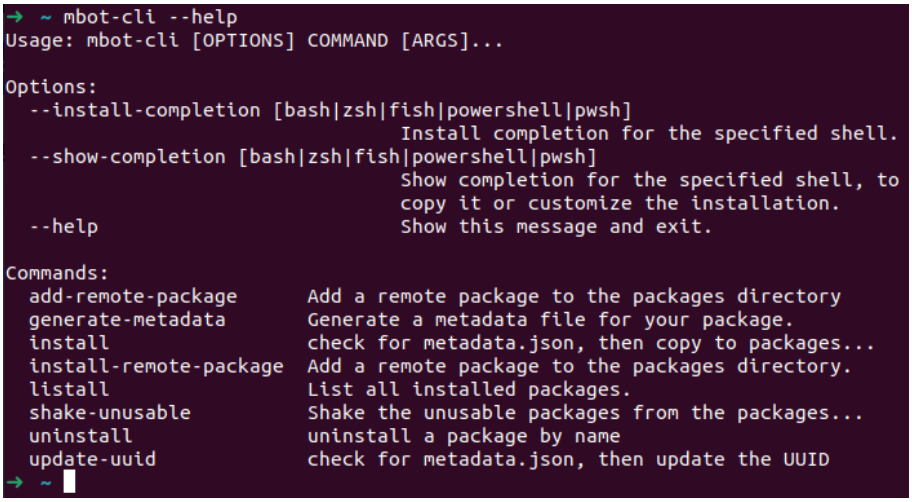

Figure 5: Commands provided by the M-Bot CLI

Git Powered Package Installation

As noted early, fostering an open source community was a major goal of this project. To help attain this goal, a git powered installation system was developed to make the installation of different packages easy on a device running the M-Bot web application. As a package has been published to Github using the npx mbot-package-publisher tool, users may use the git link to install a package through the settings page in the web application. Doing so takes care of the entire installation process for the user, and has the package up and running in seconds. Figure 4 highlights the settings page where you can see the git installation module.

Prebuilt M-Bot Packages Ready for Installation

While the framework as a whole is highly modular and ready for developers to start producing new packages, it is difficult to gain traction without some prebuilt packages that can be used right out of the box. As of now, there are a few prebuilt packages published on Github which can be installed to the M-Bot web application. Of these, the main packages are the Settings package which can be used to maintain your web application, the Drive Control package, allowing users to drive ROS powered robots, a basic ROS diagnostics package, which allows you monitor the status of ROS on your system, along with a few other basic example packages. These packages are easy to install, and require no setup to use.

Quick Installation Script

To ensure that the M-Bot web application is accessible to educators at all technical levels, installing the web application is as simple as copying one bash command into the robots terminal. The install script takes care of installing and configuring everything on the robot from Nginx and the API, to installing the default packages that are included with the M-Bot web application. The installation generally takes less than a minute, and once complete, the web application is ready to be used immediately.

Technical Review Concluding Remarks

This section highlighted a number of technical features and tools that are included with the M-Bot web framework. These tools and features included:

- React Frontend
- Nginx/Flask Backend
- NPM Tools for Package Development
- Command Line Interface for Package Management
- Git powered package installation
- Prebuilt Packages ready for installation
- Quick Installation Script

These tools and features were developed with the goal of making the web application useful out of the box for users of all technical skill levels, while also aiming to foster an open source community to make educational robotics tools more accessible for everyone.

Discussion / Conclusion

There are a number of challenges that educational robotics face, and among those are high cost and high barriers to entry. The M-Bot robotics platform aims to provide a cost effective solution to allow schools with lesser funding to provide high class robotics education. The framework as a whole is a web based graphical user interface giving students a simple way to connect to and interact with their robot. To ensure that the needs of all educators are met, the framework was designed to be as modular and expandable as possible. The M-Bot web framework provides educators who need specific robotics packages an extensive set of development tools to ensure that developing their own M-Bot packages is as simple as possible. A long term goal of this project is to have a thriving open source community of educators who develop and maintain their own set of open source robotics tools, giving way to a plethora of free to use and easy to install robotics packages, so that all needs that an educator may have on their robot can be met with a simple package installation. While the current set of tools and packages is small at the moment, there is a great opportunity moving forward, and hopefully the new M-Bot web framework is taken advantage of by many.

References

[1] “Robotics - worldwide: Statista market forecast,” *Statista*. [Online]. Available: https://www.statista.com/outlook/tmo/robotics/worldwide. [Accessed: 23-Apr-2023].