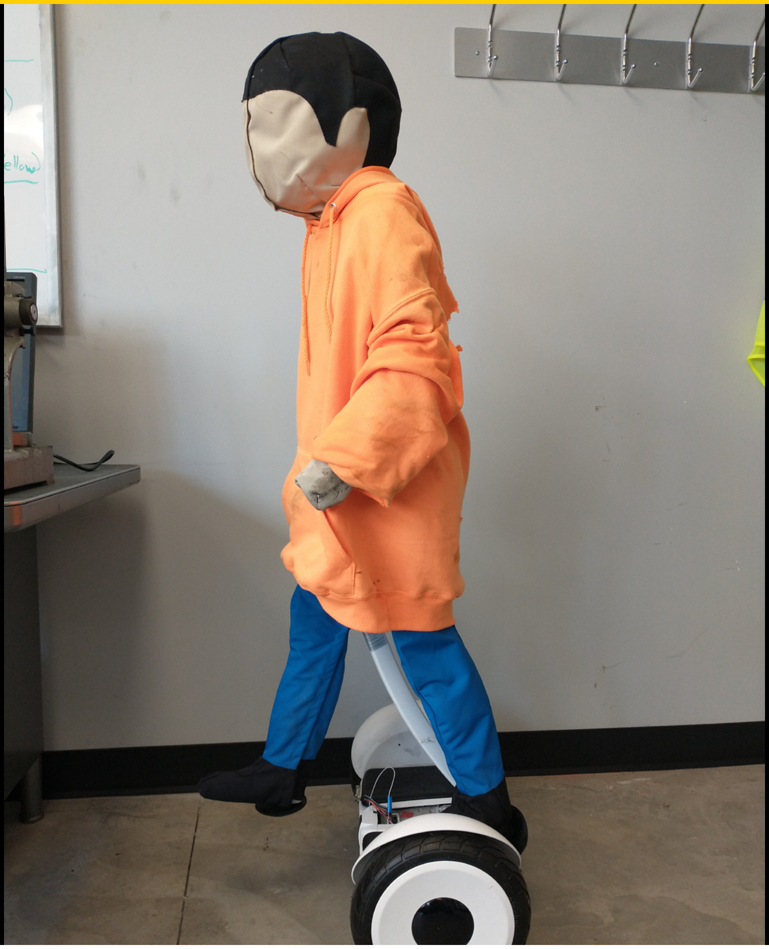
CENTER FOR CONNECTED AND
AUTOMATED TRANSPORTATION

Final Report #76
March 2024



xBOT — A Versatile Robot to Assist Testing of Autonomous-Connected Vehicles

Sridhar Lakshmanan, PhD
Weidong Xiang, PhD

DEARBORN

CENTER FOR CONNECTED
AND AUTOMATED
TRANSPORTATION

Report No. 76 *September 15, 2023*

Project Start Date: July 1, 2022

Project End Date: June 30, 2023

xBOT – A Versatile Robot to Assist Testing of Autonomous-Connected Vehicles

by
Sridhar Lakshmanan
Associate Professor
University of Michigan-Dearborn

CCAT | CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

DISCLAIMER

Funding for this research was provided by the Center for Connected and Automated Transportation under Grant No. 69A3551747105 of the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (OST-R), University Transportation Centers Program. The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Suggested APA Format Citation:

Lakshmanan, S. and Xiang, W. (2023). xBOT – A Versatile Robot to Assist Testing of Autonomous-Connected Vehicles. Final Report. Project 76.
DOI: 10.7302/22448

Contacts

For more information:

Sridhar Lakshmanan
4901 Evergreen Rd
Dearborn, MI 48128-1491
Phone: (313) 593-5516
Email: lakshman@umich.edu

Weidong Xiang
4901 Evergreen Rd
Dearborn, MI 48128-1491
Phone: (313) 593-5525
Email: xwd@umich.edu

CCAT
University of Michigan Transportation Research Institute
2901 Baxter Road
Ann Arbor, MI 48152
umtri-ccat@umich.edu
(734) 763-2498

CENTER FOR CONNECTED
AND AUTOMATED
TRANSPORTATION

Technical Report Documentation Page

1. Report No. Final Report 76	**2. Government Accession No.** Leave blank – not used	**3. Recipient's Catalog No.** Leave blank - not used

4. Title and Subtitle xBOT – A Versatile Robot to Assist Testing of Autonomous-Connected Vehicles DOI: 10.7302/22448	**5. Report Date** September 15, 2023 **6. Performing Organization Code** N/A
7. Author(s) Sridhar Lakshmanan, Ph.D., https://orcid.org/0000-0001-7387-3943 Weidong Xiang, Ph.D., https://orcid.org/0000-0001-6010-538X	**8. Performing Organization Report No.** N/A
9. Performing Organization Name and Address Center for Connected and Automated Transportation University of Michigan Transportation Research Institute 2901 Baxter Road Ann Arbor, MI 48109 University of Michigan-Dearborn 4901 Evergreen Rd Dearborn, MI 48128-1491	**10. Work Unit No.** N/A **11. Contract or Grant No.** Contract No. 69A3551747105
12. Sponsoring Agency Name and Address U.S. Department of Transportation Office of the Assistant Secretary for Research and Technology 1200 New Jersey Avenue, SE Washington, DC 20590	**13. Type of Report and Period Covered** Final Report (July 2022 - June 2023) **14. Sponsoring Agency Code** OST-R

15. Supplementary Notes

Conducted under the U.S. DOT Office of the Assistant Secretary for Research and Technology's (OST-R) University Transportation Centers (UTC) program.

16. Abstract

This project aims to develop a robot (xBOT) that is customized for testing autonomous-connected vehicles. xBOT itself would be an automated-connected robot that behaves, and be perceived, as a free-moving Pedestrian, Scooter, Bicycle, Motorbike, or full-sized Vehicle. The project's original intention was to build xBOT by modifying the Segway Ninebot©. As such, the objectives of the project were to ruggedize the platform, provide a user-friendly interface to easily program the platform's motion, integrate the platform into the Mcity OS, allow the platform to communicate, and be synchronized, with other static and dynamic traffic elements, and achieve all this under a complete life-cycle cost of $10,000 per robot. The pitfall in developing on top of the Segway Ninebot is that when the commercial manufacturer modifies the firmware, it requires a major update of our (developer's) side. This issue was only realized in July 2023, towards the end of the CCAT project. To circumvent this problem, an alternate platform is being pursued and the results of that effort will be reported in the parallel Mcity project.

Intention 17. Key Words Mobile Robot. Trajectory Control. Internet of Things. Vulnerable Road User. Proxy.	**18. Distribution Statement** No restrictions.

19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 20	**22. Price** Leave blank – not used

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Table of Contents

Table of Figures

B. Introduction

The purpose of the XBot project is to engineer an inexpensive, serviceable robotics platform. The primary application of this project is for autonomous vehicle testing, wherein the XBot platform serves as a human proxy for vehicular autonomous detection and decision making. The current end deliverables of this project are to achieve the following:

1. Design a robotics platform.
2. Allow for remote control.
3. Implement a method of programming a predetermined route, and utilize Differential GPS for path following.
4. Develop with a microcomputer running Linux OS for easy integration into various other operating systems and use-case environments.

In order to achieve the low cost and ease-of-service nature of this project, our team opted to repurpose an off-the-shelf Segway Ninebot (link). This helps cut down on the majority of manufacturing costs, as well as replacement parts as they can be readily taken from another Segway Ninebot.

As this project was designed with the human proxy use-case in mind, a soft-bodied dummy was fastened to the Ninebot throughout the course of development and testing.

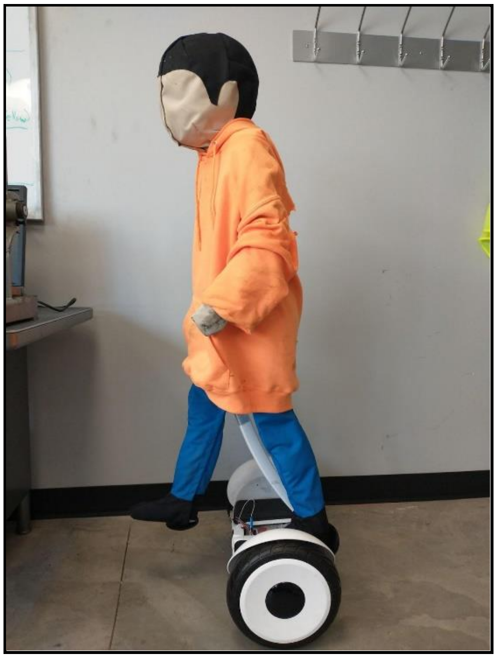


Figure 1: XBot Platform with Pedestrian Soft-Body Proxy

Please note that the initial method of motor control for the XBot platform encountered a setback in July of this year, our team is currently in the process of redesigning our motor control from the ground up. Further details of this setback and our current main objective is discussed in greater detail further below.

C. Findings

Initial Approach

Our team's initial approach for designing the platform was to utilize as much of the off-the-shelf functionality provided by the Segway Ninebot as possible. This mainly pertains to the self-balancing nature of the Ninebot, along with the analog motor control circuit and the low-level embedded motor drivers. The Ninebot is produced with the intent of a human rider who connects the platform with a handheld mobile device via Bluetooth Low Energy (BLE). Once a BLE connection between the mobile device and the Ninebot is established, the user is able to put the Ninebot in "Remote Mode," where a joystick GUI is displayed on the mobile device and the user is able to control the Ninebot from there. We opted to hack the Ninebot to spoof the digital signals on the platform as our mechanism for controlling the platform.

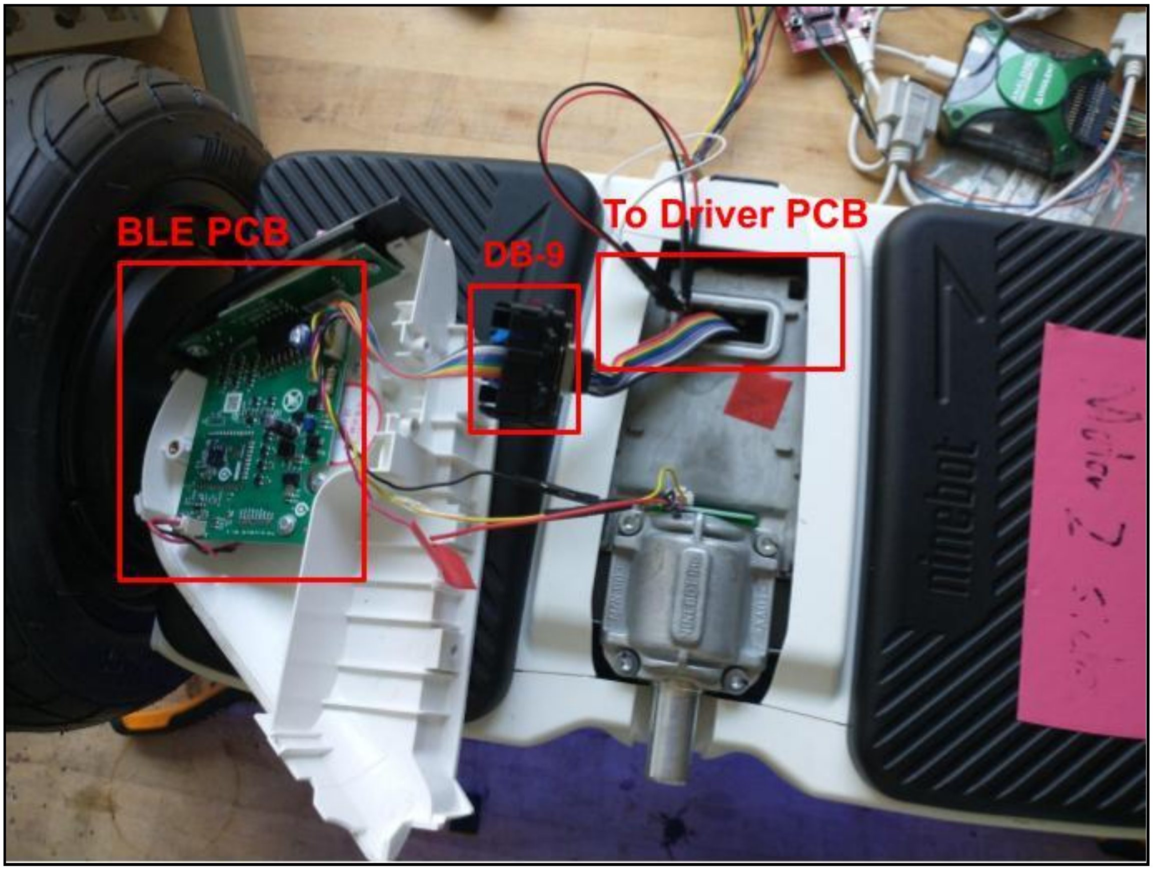


Figure 2: Ninebot PCBs and DB-9 Connector

Initial Approach: Bluetooth Low Energy Signal Analysis

Using a USB Bluetooth Low Energy sniffing device (link) and the Wireshark software (link), an open-sourced data packet tool, we were able to look at the digital data being sent back and forth between the mobile device and the Segway Ninebot. From here, our team scoped out the signals on the Ninebot's printed circuit boards (PCBs) and cross-referenced the digital data with the BLE data observed. After studying the PCBs, we found that the Ninebot system is composed of two separate microcontrollers. The first is housed on the main PCB that rests in the center of the Ninebot Chassis, which handles the firmware for on-board tilt sensors and control of the motor drivers. This microcontroller will henceforth be referred to as the "driver microcontroller" The second microcontroller is located on a PCB stationed in an external section of the Ninebot, and is fitted with a BLE transceiver for establishing BLE connectivity with the mobile device. This secondary microcontroller will henceforth be referred to as "BLE microcontroller." These two microcontrollers are able to communicate with one another via a DB-9 cable that connects the PCBs to one another.

After our team devised a way to log the digital signals on the Ninebots PCB and the digital BLE connection's packets, we conducted a series of experiments whilst the Ninebot was subjected to different test cases to find out how the Remote Mode of the Ninebot operated. This included:

- Logging data as Ninebot was supplied with voltage from a battery with various charge level thresholds.
- Logging data as Ninebot was turned off.
- Logging data as the Ninebot was turned on and stationary.
- Logging data as the Ninebot was turned on and moving in various environments.
  - Moving forwards/backwards, left-right, on flat surfaces/on an incline, etc.
- Logging data as Ninebot was disconnected from the mobile device.
- Logging data as Ninebot was connected to the mobile device.

We were able to understand the groundwork of the communication protocol between the Ninebot's two microcontrollers. We found that of the nine pins on the DB-9 Connector, two transferred digital data between the microcontrollers. Each microcontroller transmitted data on one data line, and received data on the other. From the digital signals of the various test cases, we were able to parse the BLE data packets and find the format of the data packets that contained various information such as battery charge level, motor status, and directional control commands.

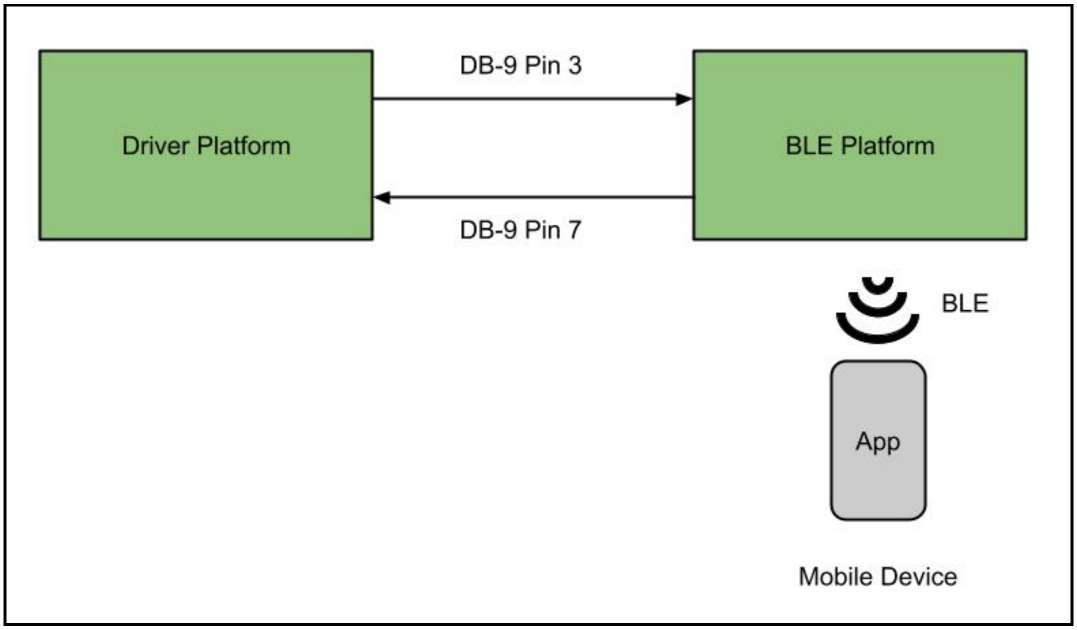


Figure 3: Segway Ninebot Communication Diagram

Initial Approach: Signal Spoofing

The Ninebot platform operates as a state-machine whose states differ depending on the following parameters:

- If the Ninebot is disconnected from a mobile device.

- If the Ninebot is connected to a mobile device.
- If the Ninebot is connected to a mobile device and in “Remote Mode.”
- Various in-between substates whilst the platform is configuring itself to operate in a different state.

From this point, our team had an idea as to what a spoofed system may look like. Our plan was to perform a “man-in-the-middle” attack, where we insert our own microcontroller in the DB-9 cable to read the data each microcontroller is sending, modify it if necessary, and transmit it forward to the next receiving microcontroller. As we wished to only modify data frames pertaining to the directional control data whilst letting other data through (battery life, error codes, etc.) unhindered. This was accomplished by printing our own custom PCB with an STM32F103 (link) microcontroller to perform the man-in-the-middle operation and route the other seven pins of the DB-9 cable through.

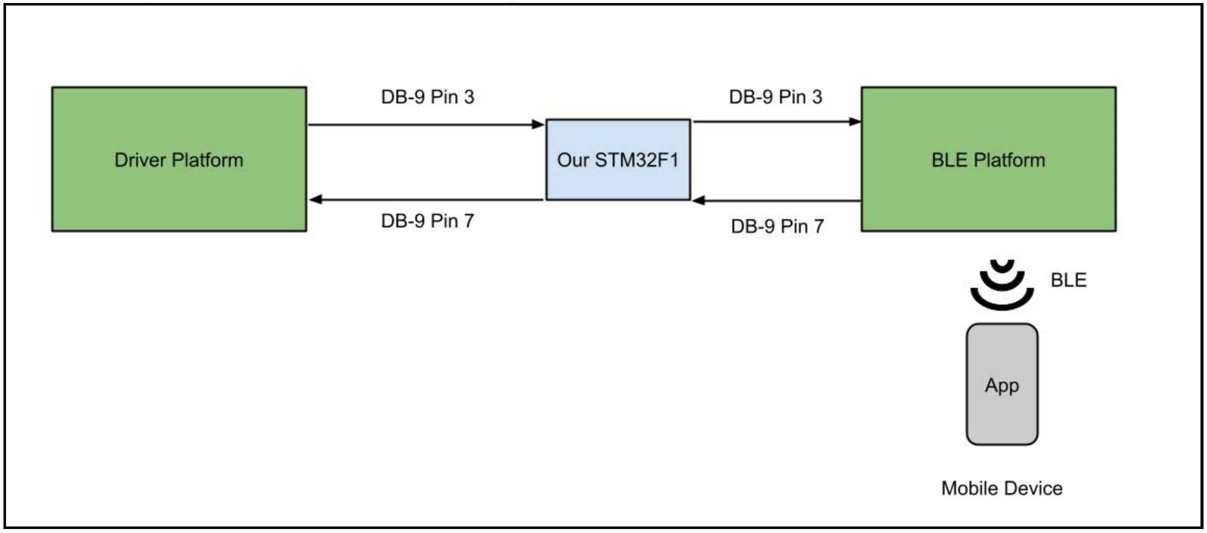


Figure 4: Main-In-The-Middle Diagram

From here, we were successfully able to control the directionality of the Ninebot with a preprogrammed code on our STM32F103 microcontroller whilst the Ninebot was connected to a mobile device, but were not yet able to control it in real time as there was no way to relay information to our microcontroller. From here, we opted to add a microcomputer to our setup, which would allow for real time control of the Ninebot.

Initial Approach: Weight Distribution

An important feature of the Segway Ninebot is the mechanical process in which it moves forward and backward. The Ninebot is a two-wheeled system, meaning it has to balance itself without tipping over. When a user is riding atop the Ninebot, this is accomplished by leaning forward or backward so the Ninebot begins to tip over. It then moves itself to “catch” the user, propelling itself in the direction the rider leans.

When operating in “Remote Mode,” it functions similarly to when there is a rider, but the tilt is accomplished by micro adjustments of the motors. It still needs to tilt, and have weight above the center of gravity of the chassis to move forward or backwards. This proved to be an issue that persisted over the course of designing and testing the system, as the human proxy dummy that was stationed atop the XBot had to be weighted so the center of gravity is at a specific height. Furthermore, the human proxy had to be securely fastened to the chassis, as if it slipped and began to lean in one direction, the XBot would move in that direction. This proved to be a persistent issue, and led to a series of problems with navigation in non-ideal environments such as wet pavement, up and down inclines, and outdoors in windy conditions.

Initial Approach: Issue with Ninebot Firmware

In July of 2023, our team encountered a major setback in the motor control method we had previously been using. As previously discussed, controlling the platform was accomplished by modifying the directional control packets as they were transmitted through the DB-9 cable of the Ninebot. However, the Ninebot itself still needed to be connected to a mobile device and set to “Remote Mode.” In order to migrate away from this necessity, our team was attempting to spoof all the signals passing between the two PCBs, essentially tricking the Control microcontroller into thinking that the BLE microcontroller was operating in “Remote Mode.” One day during testing, our team established a BLE connection to the Ninebot and the mobile app auto updated the firmware. This changed all the digital data being sent through the DB-9 cable, effectively undoing a large portion of the work accomplished. Fearing that this would be a problem we would encounter again in the future, our team opted to migrate away from the previous method of motor control, and create our own motor driver PCB with a microcontroller capable of running our own firmware.

Microcomputer Integration

The microcomputer selected to establish data transfer between our microcontroller and an external host computer was the Raspberry Pi Zero W (link), a commonly used microcomputer capable of running a Linux-based operating system. This allowed us to establish a Secure Socket Shell (SSH) connection to the Raspberry Pi via an external computer. From here, a Python script was developed to initialize a Python script to run Raspberry Pi as a wireless hotspot and accept user input for directional control. This data was passed to our STM32F103 microcontroller via UART, allowing for real time directional control.

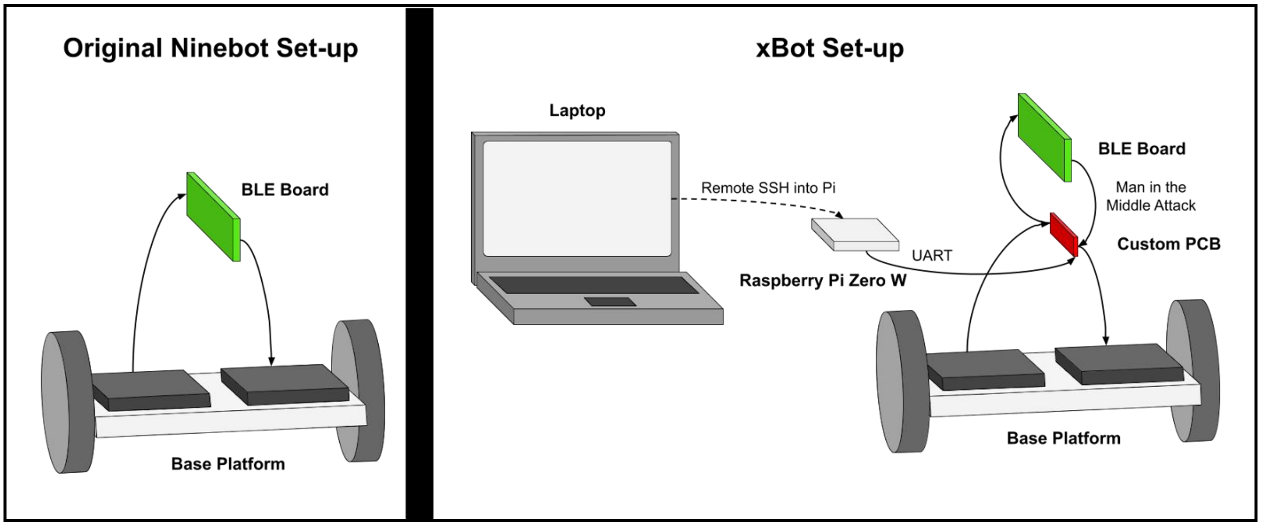


Figure 5: Original Ninebot Set-Up Compared to XBot Set-Up

Differential GPS

To properly follow a predetermined route, the XBot was fitted with the Swift Navigation Piksi Multi differential GPS system (link) for instantaneous position feedback. Briefly, a differential GPS system utilizes a stationary GPS base station to calculate its exact location, then use this information to derive any instantaneous local error in GPS positioning. This correction data is then transmitted to the GPS module stationed on the XBot platform via a radio transmitter. Standalone GPS, that is non-differential GPS such as those found in mobile devices and vehicular GPS systems, have a position error of approximately 5 meters. A differential GPS system is capable of achieving position error of approximately 10 centimeters, providing a level of accuracy necessary for robotic route tracing.

The GPS module on the XBot platform outputs position data, along with various other necessary packets of information such a magnetometer, accelerometer, and gyroscopic data, via a USB port. This was connected to the Raspberry Pi Zero W and a Python script was created to parse the incoming data as a format known as the Swift Binary Protocol.

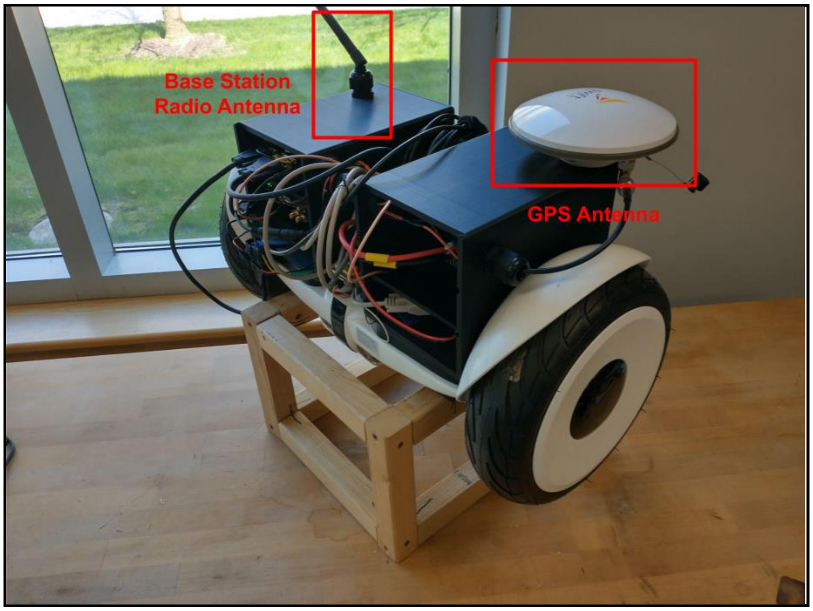


Figure 6: Xbot with Differential GPS Antennas Highlighted

Current Approach

The motor driver PCB on the Ninebot was studied and the physical signals were observed on an oscilloscope in order to find how the in-hub wheel motors are driven. Given that the motors are OEM parts and not available for individual sale, there is not data sheet available for us to work off or troubleshoot with. The printed circuit board we designed is shown below.

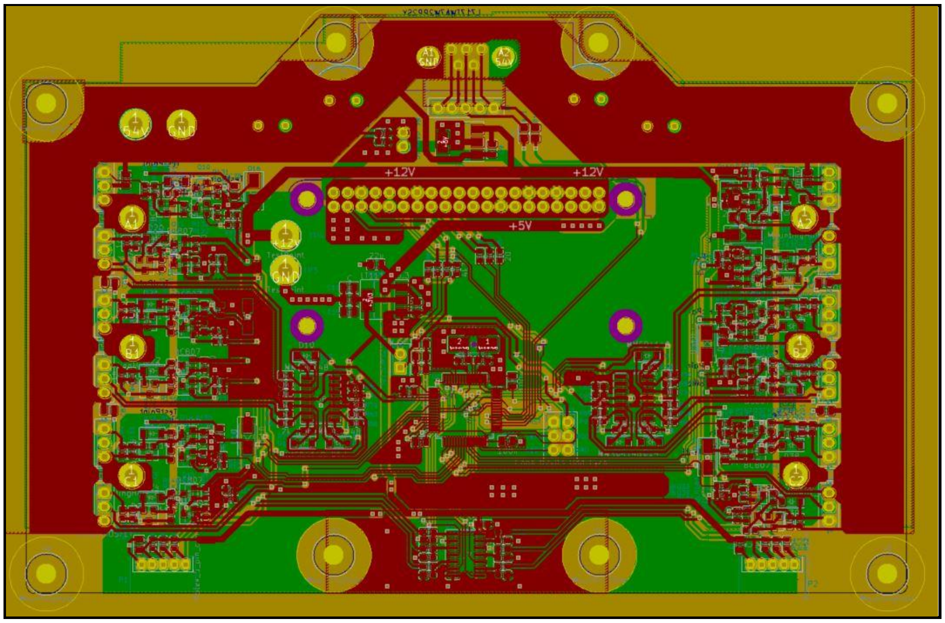


Figure 7: Custom XBot Motor Driver PCB Schematic

The PCB was designed to house all the necessary components for components for battery input, motor control signals, and interfacing with the Raspberry Pi. It was designed in the open source KiCad PCB design software (link), then sent to JLCPCB (link), a custom PCB printing fabrication house. Our team is currently working on soldering the components to the PCB, writing the code for the motor driver, and debugging the circuit. An image of the custom PCB with the components for driving a single motor is shown below.

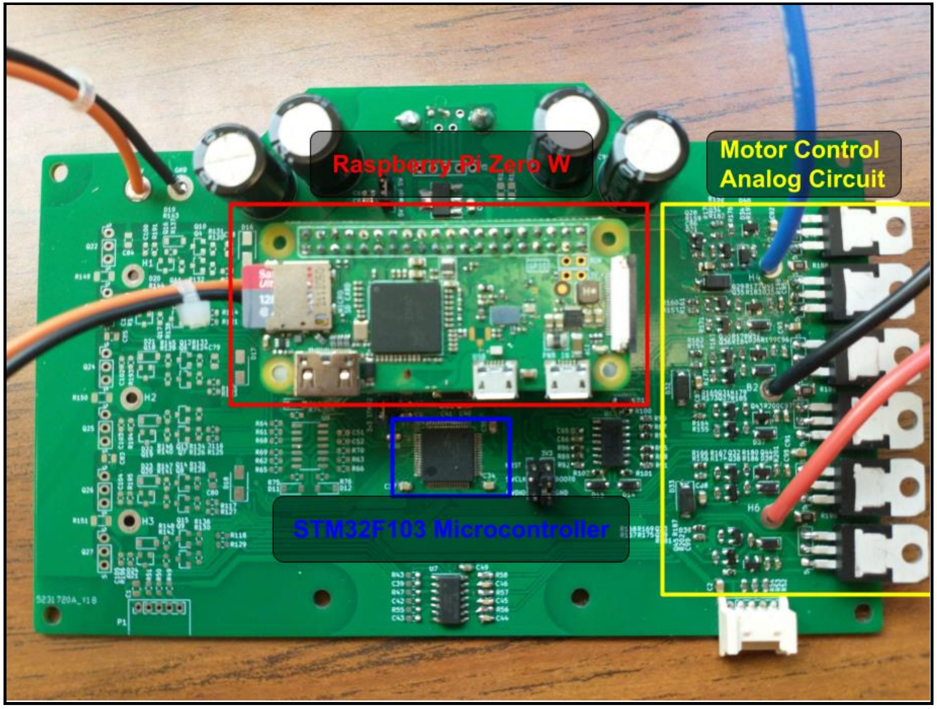


Figure 8: Custom XBot Motor Driver PCB

Newly Designed Platform Chassis

Due to the issues with balancing in non-ideal conditions, our team opted to migrate away from the two-wheel nature of the platform and redesign the chassis to have four wheels. The two additional wheels are simple caster wheels, fastened to arms attached to the chassis of the Ninebot. The aim of this is to provide greater mechanical stability to the system as a whole, allowing for improved performance in windy and uphill conditions. Furthermore, this simplifies the design for the control system, as there is no reliance on tilt sensors and balancing the weight. A graphic depicting the newly proposed chassis is shown below:

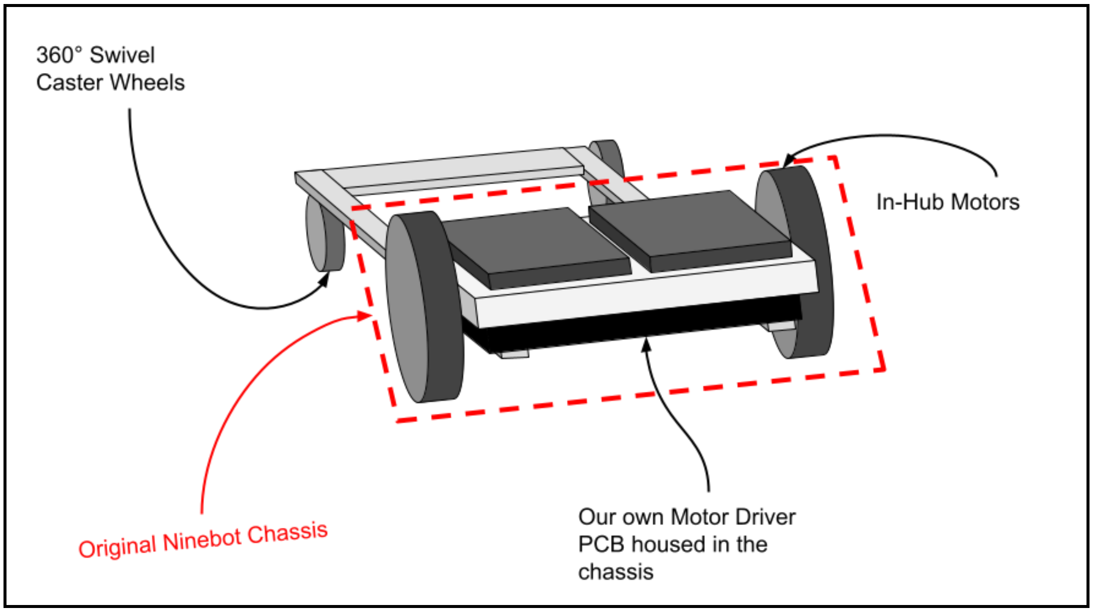


Figure 9: New Platform Chassis Graphic

Custom Controller and Path Routing

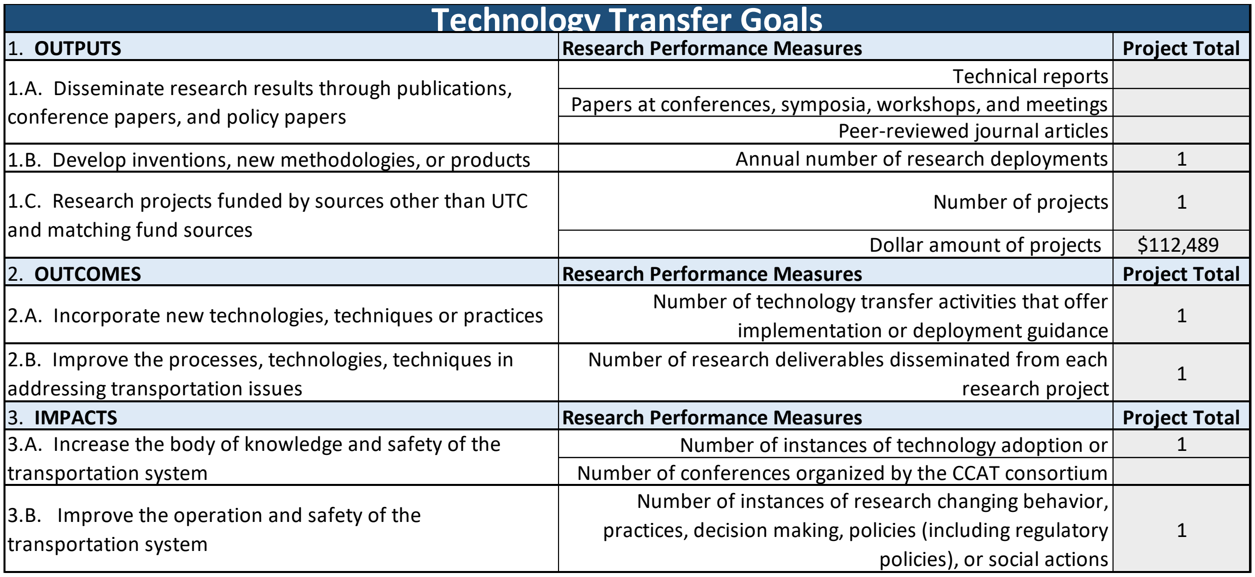
As the XBot runs with a microcomputer running a Linux OS Distribution, throughout development it was primarily controlled by connecting via SSH in a host computer's Command Window. Because the XBot is being developed as a breadcrumb GPS coordinate follower, SSH was deemed not efficient for plotting GPS points. Our team opted to develop our own mobile app, capable of steering the robot, then logging the position coordinates as a user controls the XBot. These positions can then be set as a series of targets, run again by the robot after given a start command. The UI of the app also provides a rudimentary map of the path, along with real time kinematic information such as velocity and direction relative to true north. This data is provided by the onboard sensors of the Swift Navigation Piksi Multi GPS unit.

D. Recommendations

The pitfall in developing on top of the Segway Ninebot is that when the commercial manufacturer modifies the firmware, it requires a major update of our (developer's) side. This issue was only realized in July 2023, towards the end of the CCAT project. To circumvent this problem, an alternate platform is being pursued and the results of that effort will be reported in the parallel Mcity project.

E. Outputs, Outcomes, and Impacts

Synopsis of Performance Indicators

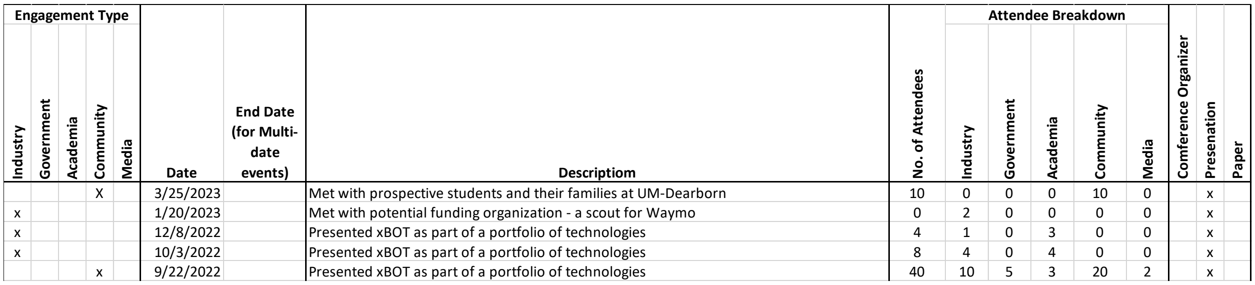


Technology Transfer Goals		
1. **OUTPUTS**	**Research Performance Measures**	**Project Total**
1.A. Disseminate research results through publications, conference papers, and policy papers	Technical reports	
	Papers at conferences, symposia, workshops, and meetings	
	Peer-reviewed journal articles	
1.B. Develop inventions, new methodologies, or products	Annual number of research deployments	1
1.C. Research projects funded by sources other than UTC and matching fund sources	Number of projects	1
	Dollar amount of projects	$112,489
2. **OUTCOMES**	**Research Performance Measures**	**Project Total**
2.A. Incorporate new technologies, techniques or practices	Number of technology transfer activities that offer implementation or deployment guidance	1
2.B. Improve the processes, technologies, techniques in addressing transportation issues	Number of research deliverables disseminated from each research project	1
3. **IMPACTS**	**Research Performance Measures**	**Project Total**
3.A. Increase the body of knowledge and safety of the transportation system	Number of instances of technology adoption or	1
	Number of conferences organized by the CCAT consortium	
3.B. Improve the operation and safety of the transportation system	Number of instances of research changing behavior, practices, decision making, policies (including regulatory policies), or social actions	1

Figure 10: Synopsis of Performance Indicators

The project outcomes consists of one patent application and one physical device (xBOT) for enhancing the testing of automated-connected vehicles. The project impacts consists of one physical device (xBOT) being developed with Mcity, and that one device is being intergated into the Mcity Operating System (Octane). Three presentations of xBOT were made - A2 TechTrek (9/2022), Toyota CSRC (10/2022), and Public High Schools (12/2022).

Summary of Outreach



Engagement Type									Attendee Breakdown							
Industry	Government	Academia	Community	Media	Date	End Date (for Multi-date events)	Descriptiom	No. of Attendees	Industry	Government	Academia	Community	Media	Comference Organizer	Presenation	Paper
			X		3/25/2023		Met with prospective students and their families at UM-Dearborn	10	0	0	0	10	0		x	
x					1/20/2023		Met with potential funding organization - a scout for Waymo	0	2	0	0	0	0		x	
x					12/8/2022		Presented xBOT as part of a portfolio of technologies	4	1	0	3	0	0		x	
x					10/3/2022		Presented xBOT as part of a portfolio of technologies	8	4	0	4	0	0		x	
			x		9/22/2022		Presented xBOT as part of a portfolio of technologies	40	10	5	3	20	2		x	

Figure 11: Summary of Outreach

Relevant Notes

1) There have been no publications or conference papers out of this project
2) Multiple presentations and demonstrations of XBot have been made
3) The primary output of this project is the robotics platform itself. Although it has not been completed, it has seen a great deal of progress since its kickoff and inception one year ago. Up until the setback we encountered two months ago, the project was initializing a field testing phase. Once the motor control is finalized, we are going to resume field testing of features in development.
4) Autonomous vehicles are a burgeoning technology that promises to revolutionize the transportation sector in a variety of ways, but prior to its widespread adoption, it must undergo vigorous testing for performance and safety. Amongst the most critical considerations are autonomous vehicles' behavior around pedestrians. Testing facilities are currently limited to a few less than ideal options for testing this behavior. The first is using actual pedestrians, which proves dangerous to the pedestrians if the autonomous vehicle does not sense them or behaves erratically. The second is to utilize rudimentary dummies, essentially non-robotic dummies on wheels which are mechanically pulled by a string or rope. The final is to employ expensive, top of the line robotic systems which generally cost more than US$100,000. These systems are not easily integrated into other automated

testing facility environments, difficult to service, and prove too expensive for many autonomous vehicle testing facilities. The main impact of the XBot project is to reduce these costs, and provide a more flexible testing environment, such as multiple human proxies following their routes in unison simulating a crowd in a densely populated city.

5) The primary technical challenges were the weight distribution balancing problem and critical motor control setback discussed in this paper. The lack of mechanical control of the platform essentially brought the project to a standstill. However, these issues should be resolved once the new four wheeled platform design is implemented.