Memo

To: Professors Pisano, Osama, Hirsch, and Lagoy

From: Linden Adamson, Jason Calalang, Tara Gill, Zhilang Gui, Cole Resurreccion, Raina Yin

Team: Team 7 - Bike for Blind

Date: 4/10/2025

Subject: Final Project Testing

**Boston University**

**Electrical & Computer Engineering**

**EC464 Senior Design Project**

**Final Testing Report**

**Blind Bike**

****

**By**

**Team 7**

**Bike for Blind**

**Team Members**

**Linden Adamson** [lindena@bu.edu](mailto:lindena@bu.edu)

**Jason Calalang** [jasonc16@bu.edu](mailto:jasonc16@bu.edu)

**Tara Gill** [taragill@bu.edu](mailto:taragill@bu.edu)

**Zhilang Gui** [zgui@bu.edu](mailto:zgui@bu.edu)

**Cole Resurreccion** [coler@bu.edu](mailto:coler@bu.edu)

**Raina Yin** [ryin2@bu.edu](mailto:ryin2@bu.edu)

**Required Materials/Equipment**

Hardware:

* Jetson Orin Nano Super Developer Kit
* OAK-D RGBD Camera
* 2x 12V Trucklite 15200Y (steering and braking lights)
* 2x DS5160 60 kg\*cm Servo Motor
* PCA9685 Motor Driver
* NFP-36GP-555-EN 24V Geared Motor
* DC Motor Driver L298 Dual H Bridge Motor
* 19V 4.2A Barrel Jack Power Supply
* 2x Arduino Uno R3
* Arduino Uno R2
* Hilitand 4-Channel MOSFET PWM
* 433MHz RF transmitting button
* RXB6 433MHz Superheterodyne Wireless Receiver Module
* 90dB Piezo Buzzer
* 3x 3.3V ERM coin motors (haptics)
* 5x DC-DC Step-Down Converter 5A maximum
* 2x Weize 12V 12Ah SLA batteries connected in series

Software:

* Arduino Code
  + Brake and Steering script - flashed to Arduino Uno R3
  + Key Fob script- flashed to Arduino Uno R2
* Collision Avoidance
* Manual Stop Button
* Orin Nano ROS2 Nodes
  + Keyboard Operation (for control)
  + Braking Command
  + Steering Command
  + OAK-D ROS2 Nodes
  + Vectornav

**Set-Up**

First, every component must be wired according to the diagram below.

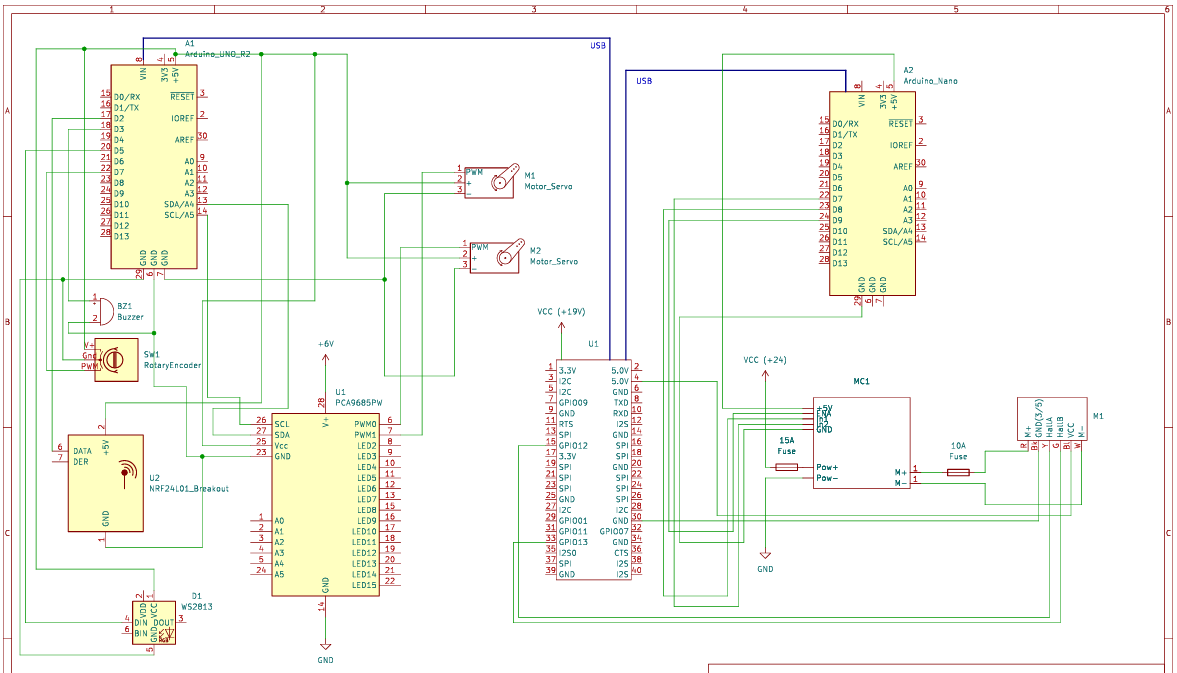


Figure 1: Circuit Schematic of Prototype System

Once all the components are appropriately connected, ensure no exposed wires contact the aluminum frame to cause a short. After confirming connections, turn on the battery switch to supply power to the necessary components.

* Arduino Nano and Uno should get power from the USBs provided by Orin.
* Connect the 6V buck converter output from the battery to the PCA9685 servo driver.
* Connect the 24V buck converter output from the battery to the motor controller and set the output to 24V.
* Power the Orin with the 19V barrel jack plug which is connected to the battery via buck converter.
* The braking and steering lights should be powered from the battery via a 12V buck converter.
* The OAK-D should be powered from an Orin USB port.
* The Vectornav should be powered from an Orin USB port.

For proper braking tests, both of the DS5160 servo motors should be mounted and attached to the trike.

For the braking system test, the Arduino Uno will be connected to the Jetson Orin. The brake light and PCA9685 motor driver will be connected to the Arduino Uno. The DS5160 servo motors will both be connected to the PCA9685 motor driver, with each servo motor mounted onto the trike’s front two brake pads.

For the key fob, a 433MHz RF receiver and a 90dB Piezo buzzer will be connected to the Arduino Uno R2.

For the steering system, the NFP is directly connected to the center axle of the trike controlling the steering. The NFP should be plugged into the controller and the encoder output should be attached to the Arduino.

For object detection, an OAK-D camera will be mounted onto the trike and connected to the Orin.

**Pre-testing Setup Procedure:**

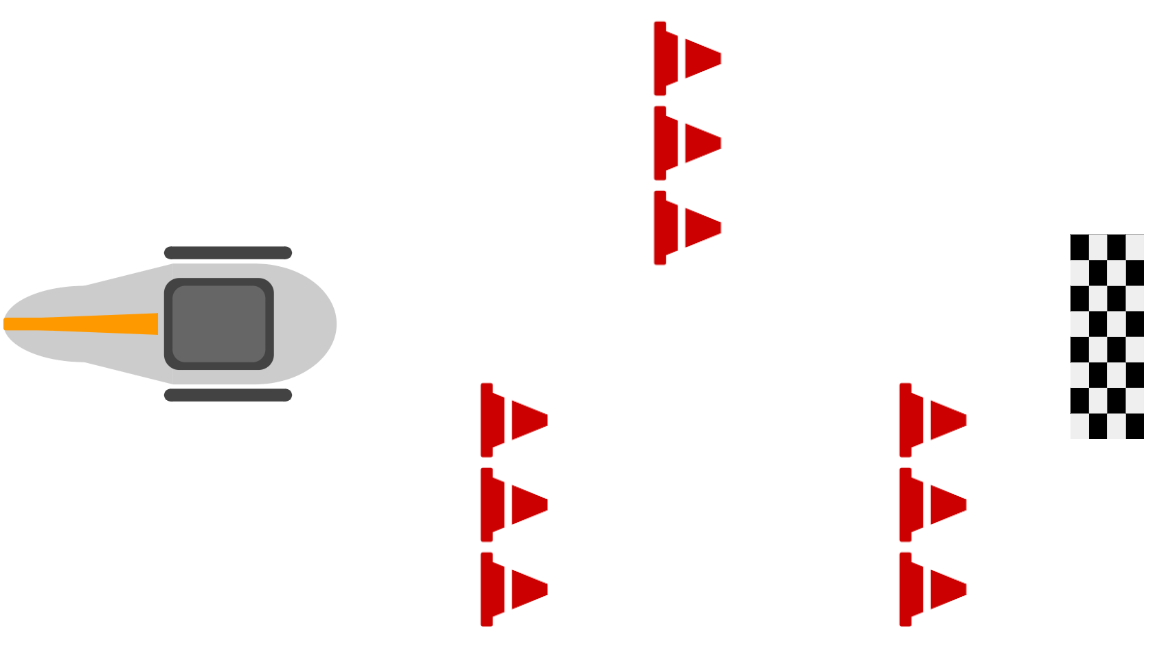
On Orin Nano:

1. Plug in the Orin Nano. The correct commands should automatically run. SSH to the Orin via *10.239.146.85* and run the command *ros2 run tester key\_op*

Power System

1. Connect the 2 12V batteries in series and press the on switch.

**Testing Course Setup:**

****

*Visual representation of the obstacle course to simulate a straight line with adaptive turning by weaving–distance between objects and sizes are not to scale. Driving distance between each cone segment should be at least 1.5m.*

**Testing Procedure:**

The user can access the trike with a key fob and begin pedaling when ready to go.

**Accessibility Feature: Key Fob**

1. Stand 10m away from the trike (powered off) and use the keyfob by pressing the RF button transmitter and hearing the sound from the buzzer to find the trike.

**Manual Control**

1. On the keyboard operation terminal:
   1. Type the character *s*. The brakes should engage and the LED brake lights should turn on.
   2. Type the character *w*. The brakes should disengage and the LED brake lights should turn off.
   3. Type the character *j*. The steering motor should turn left.
   4. Type the character *k*. The steering motor should stop.
   5. Type the character *l*. The steering motor should turn right.
2. Manual brake
   1. Engage the handlebar brake on the inside of the vehicle. The brakes should engage.
3. Remote steering and braking with obstacle
   1. Arrange an obstacle course using cones placed in a predetermined pattern along the test route.
   2. The user sits on the bike, ready to pedal, and prepared to control the steering and braking through the keyboard.
   3. Using the keyboard, the user maneuvers the bike carefully through the course, demonstrating precise manual steering.

**Collision Avoidance**

1. Activate emergency braking software
   1. Clear OAK-D camera’s sightline of any object 3m or closer. The brakes should not be engaged.
   2. Walk to within 3m of the OAK-D camera (2m from the nose of the trike). The brakes should engage. Haptic feedback on the seat must activate to signal to the user that braking is occurring.
   3. Leave 3m sightline of OAK-D camera. The brakes should disengage. The haptic feedback should return to idle, signaling to the user that the brake is no longer active.

**Navigation**

1. The bike should autonomously make a take a left turn on the preplanned path
   1. Set up the bike on Cummington Mall.
   2. User sits on the bike and starts pedaling, preparing to take a left turn from Cummington Mall away from the Photonics building
   3. As the turn is initiated, the trike’s steering system should engage automatically and maintain a smooth and precise transition through the turn.
   4. The IMU continuously records data and adjusts the trajectory as needed to ensure optimal path planning throughout the maneuver.
   5. During the turn maneuver, the corresponding turn signal lights should automatically illuminate to indicate the intended direction.
2. Repeat each of the above testing procedures 5 times to prove consistent accuracy of our system.

**Measurable Criteria**

1. Key fob will activate the buzzer at least 10 meters away from the trike with 80% accuracy.
2. Brake lights and brakes engage within 0.1s of receiving brake command from the Orin Nano with 100% accuracy.
3. Brakes will engage when an object is within 3m of OAK-D camera sightline (2m from the nose of the trike), and disengage when there is no object within 3m of OAK-D camera sightline with 90% accuracy.
4. Manual handlebar brake will engage the brakes with 100% accuracy. The system should confirm full brake engagement within 0.2 second. It should also override the braking command from the emergency brake.
5. Brake servo motors rotate enough to engage the brake pad with 100% accuracy. The response time for this and the haptic feedback should not exceed from command issuance to brake engagement must not exceed 0.2 second.
6. The steering motor will rotate in response to the motor command, and the rotation angle must match the command input within the tolerance of 2 degrees to ensure accurate steering.
7. The steering motor restricts itself from over-rotating/stalling.
8. During the obstacle course evaluation, the bike must not collide with the cones throughout the maneuver. In repeated tests, the bike should navigate the course without any impact on the cones in at least 95% of the trials.
9. Upon receiving the command, the steering motor should rotate the front 2 wheels to achieve a minimum turning angle of 20 degrees within 0.5 seconds and with 90 percent accuracy.
10. The trike must follow the planned route closely, deviating no more than 30 centimeters from the intended path. This performance should be observed in at least 90 percent of tests.

**Equipment and Setup**

Our setup went mostly as expected. We rolled the trike outside and turned on the power switch, where the trike successfully powered. When trying to run our tests, however, we struggled to ssh into the Orin Nano. This was because the internet connection was not strong outdoors. We moved the trike to a different location outside and were able to connect, however, the connection was still weak. Once connected, we inputted steering and braking commands to ensure our system was wired correctly.

**Measurements Results**

| **Test** | **Result** |
| --- | --- |
| Key fob will activate the buzzer at least 10 meters away from the trike with 80% accuracy. | The key fob works at 90% accuracy at 5 meters away from the bike. |
| The response time of brake lights, haptic motors, and brakes engage within 0.25s of receiving brake command from the Orin Nano with 100% accuracy. | * The brake lights turned on within 0.25s 100% of the time when the brakes were engaged. * The brakes engage within 0.25s 100% of the time when receiving brake command from Orin, but only when connected to the internet * Haptic feedback matches this signal accuracy rate. Haptic feedback is a simultaneous response with the servo motor signal. |
| Brakes will engage when an object is within 3m of OAK-D camera sightline (2m from the nose of the trike), and disengage when there is no object within 3m of OAK-D camera sightline with 90% accuracy. | When coasting towards an object (person/wall), the bike stops faster than a second at a distance of 3m from the camera sightline. The same results occur when a user is operating the bike and stops pedaling on the haptic feedback signal. If a person walks in front of the bike, from any angle, and enters the 3m camera sightline zone, the brakes engage in faster than a second. |
| Manual handlebar brake will engage the brakes with 100% accuracy. The system should confirm full brake engagement within 0.25 second. It should also operate independently from the braking command from the emergency brake. | Using normal grip pressure on the manual brakes (~15 lbs), Pressure to the brake pads were applied instantly and gradually increased to a stopping force within 0.25 seconds. This test was conducted while both the emergency brakes and remote brakes were disengaged. |
| Brake servo motors rotate enough to engage the brake pad with 100% accuracy. | The servo motors controlling the brakes act synchronously in engaging the brakes 100% of the time. |
| The steering motor will rotate in response to the motor command, and the rotation angle must match the command input within the tolerance of 2 degrees to ensure accurate steering. | When connection was established, sending a motor command had a 100% accuracy of rotating the motor and the rotation angle |
| The steering motor restricts itself from over-rotating/stalling. | When the duration of the motor control signal is remotely extended–causing the steering motor controller to turn the steering mechanism longer–the encoder sends a signal to stop turning the motor, this is confirmed in the test conducted by monitoring current and reading no spike in power usage (which would indicate stall current). |
| During the obstacle course evaluation, the bike must not collide with the cones throughout the maneuver. In repeated tests, the bike should navigate the course without any impact on the cones in at least 95% of the trials. | In each iteration of the test, remote control was able to maneuver the bike around the cones and no impact with cones occurred. The cones were spaced in a pattern so that the operator must weave between cones which were 1.5m apart in distance. |
| Upon receiving the command, the steering motor should rotate the front 2 wheels to achieve a minimum turning angle of 20 degrees within 0.5 seconds and with 90 percent accuracy. | When sending the steering command, the front two wheels rotate appropriately to the command, ensuring a 20 degree turn. Turns are accurate but do not respond with the same timing desired: (0.8 seconds/20°) |
| The trike must follow the planned route closely, deviating no more than 30 centimeters from the intended path. This performance should be observed in at least 90 percent of tests. | Due to lack of conditional navigation, the trike did not complete successful path following. However, no collisions were recorded in any tests. |

**Conclusion**

With our final testing plan, we were able to electrify a trike and add L2 autonomy, as defined below, and incomplete L3 autonomy, and accessibility features. One major problem we faced was a loss of control with an unstable internet connection. To fix this, we will convert all control of the trike to be possible without an internet connection. This will be done by creating both a manual and autonomous mode controlled on the bike, similar to how cars have drive, reverse, neutral, and park modes. We plan to have a park, manual, and autonomous mode. The park mode will engage the brakes, the manual mode will use a set of controls on a trike, and the autonomous mode will use the ROS2 Nav2 stack.

**L2 Autonomy**

L2 autonomy is defined as having automated braking, steering, and acceleration. Our client did not want automated velocity control, but our trike is able to both brake and steer. The steering was accomplished by using an upside down motor for direct control on the revolute steering joint. Since the trike was able to navigate with motor commands quickly and accurately, we determined that this solution was mechanically robust. One concern was that direct drive would draw stall current on the motor, but with our intuitive angle monitoring using the built-in encoder, power usage remained safe throughout all our tests and use cases. The electric braking is done with servo motors using nylon Micro Cord to pull on the brake pads. From our tests we concluded that the force from our braking servos (on battery power) were able to pull the trike to a stop faster than our expectations, and restrict some pedaling from the user. The automated braking is done using the OAK-D stereo camera, which will send a brake command if there is an object within 3m of the camera. In our testing, we found our danger range of 3m to be too large, considering how fast our trike brakes. A further development is to use the current velocity to create a dynamic range for the braking.

**Accessibility Features**

The key fob works at a shorter range when the RF receiver is placed inside the bike. We attribute this to electromagnetic interference as our bike is fully made of metal, and the receiver is placed near all of the other data and electrical wires in our circuit. We also successfully implemented a haptic motor that successfully activates when the brakes are engaged. Success was determined by surveying a third party user who verified that haptic feedback was noticeable. This notifies the rider that the brakes will be engaged. The product we have created thus far is generally usable for someone who is visually impaired, however, we still aim to add voice activated features in time for ECE day.

**L3 Autonomy**

Our initial method for L3 autonomy was using a transformer neutral network known as ViNT. Unfortunately, in our simulation testing, ViNT failed to produce a model capable of conditional navigation. For this reason, we decided to switch to more traditional, physics-based methods. This switch, however, came very late in our project and was not able to be fully completed by the final test. Using ROS2 Nav2, we created an URDF file to represent our trike and tested using Nav2 on a map of Cummington in RViz in simulation. From our tests, it is clear that this deliverable is what we need to spend most time working on for ECE day. Before ECE day, we plan to implement SLAM with a LiDAR and integrate navigation with GPS waypoints. We also plan on expanding our emergency braking collision avoidance system into a “smarter” collision prediction and avoidance system, which would give the autonomous trike the ability to traverse around objects in its way, and/or move out of the trajectory of other moving objects. We have a working prototype in simulation, but need to improve the runtime performance for our real-time use case by ECE day.