**Boston University**
**Electrical & Computer Engineering**
**EC463 Senior Design Project**

Second Prototype Test Report

# MOSS Wheelchair: Modular Open-Source Smart Wheelchair

by

Team 11
MOSS Wheelchair

Team Members

Sebastian Gangemi sgangemi@bu.edu
Aya Kassem ayak@bu.edu
Norbert Malik norbert@bu.edu
Maha Noor mahakdn@bu.edu

## Equipment and Setup

*Motor Controller:*

A. Hardware:
    - Wheelchair
    - 2 50 W 24 V Brushed Motors
    - 2 Motor controllers
    - 2 24 V AGM Batteries
    - 2 Single-ended quadrature wheel encoders
    - 2 Single-ended to differential quadrature signal converters
    - Joystick
    - Arduino
    - NVIDIA Jetson Nano

B. Software:
    - Simple Arduino script
        - Controls RPM of each motor via a PWM signal sent to its respective motor drive
    - Simple NVIDIA Jetson Nano Script
        - Uses PySerial library
        - Communicates with Arduino to send commands to move forward, backward, and stop

*Object Detection:*

A. Hardware:
    - NVIDIA Jetson Nano
        - Flashed microSD card
        - Power adapter (2.5 A 5 V)
        - WiFi card
        - WiFi antennas
    - Raspberry Pi SC0024 IMX219 NoIR Camera

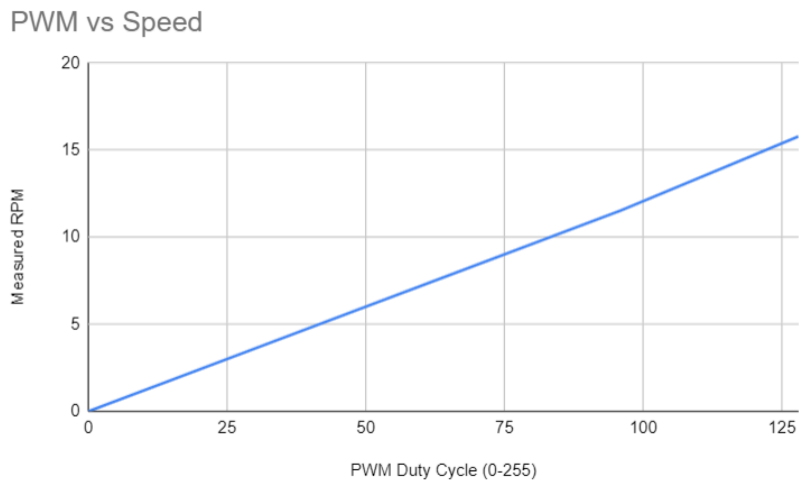
B. Software:
    - Python 3.6
    - Jetson Inference
    - PyTorch with CUDA support
    - TorchVision
    - GStreamer

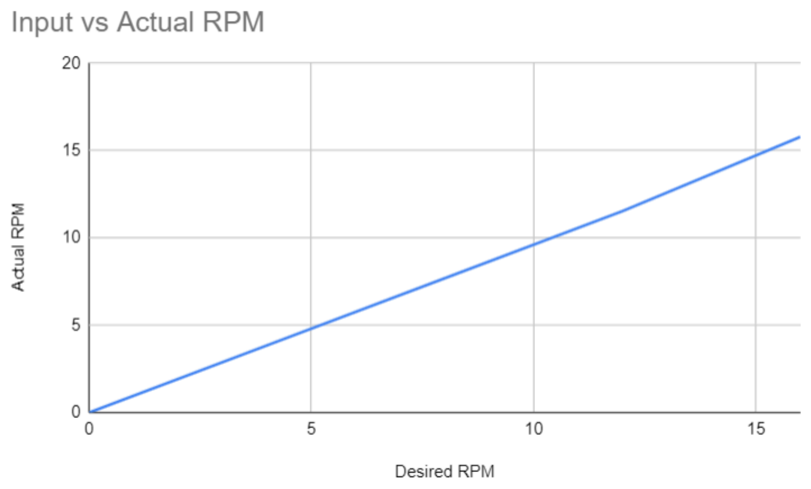
## Measurements Taken

*Motor Controller:*

-We recorded the relationship between the PWM signal sent by the Arduino, versus the time it takes for the wheel to do a full rotation to confirm that there is a proportional response from the motor for each inputted speed up to our target speed: 16 RPM ≈ 1 ft/sec

| PWM duty load (0-255) | Deired RPM | Measured Time/Rev (Seconds) | Measured RPM |
|---|---|---|---|
| 32 | 4 | 15.6 | 3.8 |
| 64 | 8 | 7.8 | 7.7 |
| 96 | 12 | 5.2 | 11.5 |
| 128 | 16 | 3.8 | 15.8 |





*Object Detection:*

- For each object, we recorded:
    - Whether the model detected the presence of an object (even if it was labeled wrong). We were interested in this measurement because in the case of obstacle detection and collision avoidance during navigation, detecting the presence of an object, no matter what the object is, is often enough to avoid obstacles or collisions. For example, we can stop the wheels' motors as long as an object is detected in front of the camera sensors, no matter what the object is. If the presence of an object is not detected at all, however, the wheelchair might end up colliding with that object.
    - Whether the detected object was labeled correctly or not. In the case where we want to detect objects for everyday tasks such as picking objects up, pushing elevator buttons, opening doors, etc., the object must be detected correctly since the task executed by the wheelchair's robotic arm will be specific to the label of the object detected.
    - Whether the object was detected in less than 1s.

- The following results were collected:

| Object | Detected Object in Frame? | Labeled Object Correctly? | Label Given by Model (if mislabeled) |
|---|---|---|---|
| Person | Yes | Yes | N/A |
| Bottle | Yes | Yes | N/A |
| Backpack | Yes | Yes | N/A |
| Cellphone | Yes | Yes | N/A |
| Whiteboard | Yes | No | Airplane |

- The camera was at 20 fps, and objects were detected at approximately 300 ms

## Conclusions

*Motor Controller:*

As the motor itself is operating in RPM ranges >>500 RPM, in this prototype test we are seeing a much smoother response from the motor than our previous test, in which we only tested up to ≈ 600 RPM. The increase in RPM, as expected, has increased the reliability of the motor output significantly, with our data showing that the motor's output is consistently within 5% of the corresponding RPM input. While this level of accuracy is sufficient for the navigational components to begin integration with the motor controlling system, in the future we will be integrating EMF feedback from the motor drives to work in combination with the wheel encoders to even further improve the precision of the motor controlling system.

*Object Detection:*
The object detection test for our second prototype was similar to our first, but we have moved from detecting objects with a laptop to detecting them with just an NVIDIA Jetson Nano. Because the Jetson Nano was not compatible with our previous fine-tuned model from last semester, we only tested object detection with an Inception-V2 model pre-trained on the COCO dataset, which contains 80 classes of common objects. The test showed that the model detected the presence of an object in the frame 100% of the time and that it both detected an object and labeled it correctly 80% of the time. Despite these numbers being satisfactory for our testing plan's measurable criteria, the sample size of this test was quite small because of the time limit of our test. As part of our next steps, we would like to get a better estimate of the capabilities of our model by experimenting with a much bigger and more realistic sample size. This second prototype test also allowed us to test the Raspberry Pi camera that we connected to the Jetson Nano, along with the Inference library working on the Jetson Nano. Successfully detecting objects from the COCO dataset validated our method. For our next steps, we need to train our model by fine-tuning it on a custom-made dataset that contains instances of objects that we need to detect specifically for our project such as doors, door handles, and elevator buttons. After that, we will add more objects to our custom dataset (street signs, for example) and attempt to train our next model with more training epochs to see whether it would do an even better job at accurately detecting and labeling objects.