Boston University
Electrical & Computer Engineering
EC463 Senior Design Project

First Prototype Testing Plan

MOSS Chair: 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

I. Required Materials

1. *For Motor Controller:*
    A. Hardware
        - Brushed DC Motor (24V)
        - Electronic Motor Speed Controller
        - Microcontroller (Arduino)
        - AC-DC Plug-in Power Supply (24V)
    B. Software
        - Ardunio script
            - Controls the rpm of the motor through the speed controller

2. *For Object Recognition Software:*
    A. Hardware:
        - Laptop with webcam
    B. Software:
        - Python >= 3.8
        - Ultralytics Python Library
        - Pre-trained Object Detection YOLOv8 Model

3. *For Robotic Arm*
    A. Hardware
        - 3D printed inmoov bicep, forearm, and hand parts
        - Strings to attach and move the arm and fingers
        - Adhesives and fasteners such as screws and bolts

II. Set-Up

The equipment and setup is divided into 3 parts: The motor controller, the object recognition software, and the robotic arm.

Motor Controller:
We input the max RPM we wish to achieve into the Arduino using an analog PWM signal ranging from 0-5 V. Along with another analog signal for the direction(clockwise /counterclockwise), the Arduino linearly increases the RPM input signal being brought into the electronic speed controller, which accordingly regulates the amount of current brought from the power source into the motor. This should achieve close to a linear increase in RPM in the motor, which corresponds to constant acceleration.

Object Recognition:

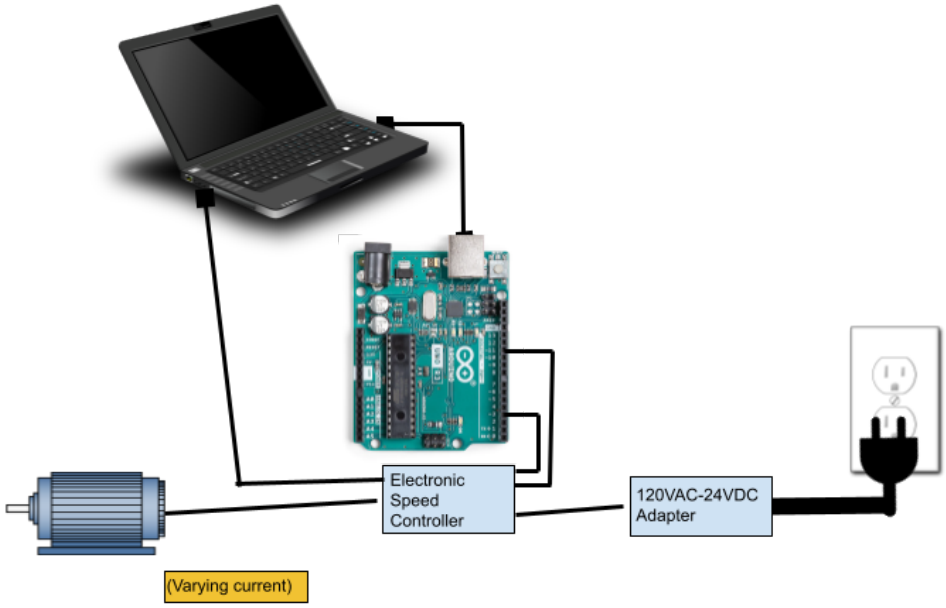
The Ultralytics library that we are using already contains a pre-trained YOLOv8 model trained on Microsoft's COCO dataset, which contains multiple instances of common objects. This pre-trained model was then fine-tuned on a combination of other datasets to allow it to detect more specific objects that we would need to detect in the case of our project. The model was set to train for 300 epochs or until there was no improvement for 30 epochs and exported for future use. Today, we will import this model and use it to detect objects in real time from a laptop's webcam and classify whether these objects were detected correctly or not.

Robotic Arm:
We will have printed all of the arm parts and have assembled one of the fingers. To force the motion of the finger, we will manually move the finger to show the range of motion of the hinge joint. Additionally, we will be demonstrating the mechanical soundness of the printed parts. The parts are expected to show no distress such as cracks or bends.

III. Pre-testing Setup Procedure

Motor controller:

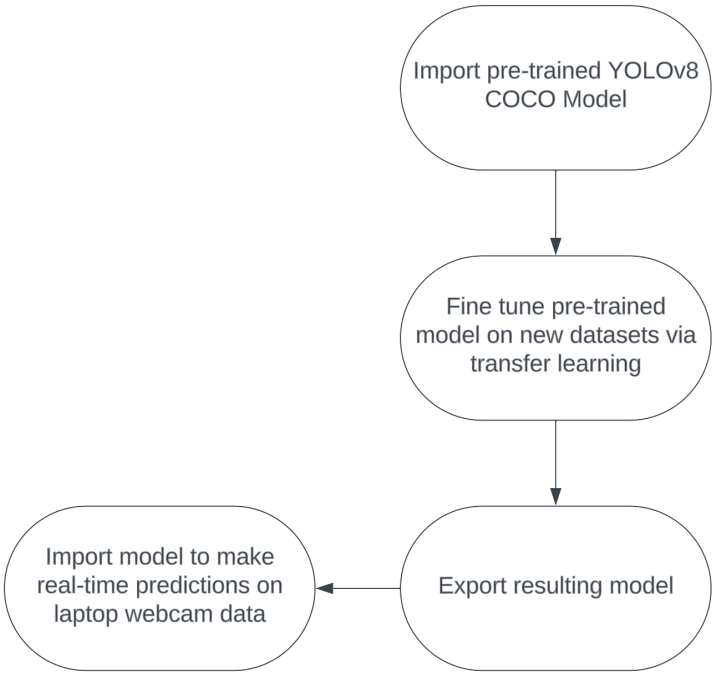
1. Connect Arduino to computer via USB for power and to make changes to Direction/Max RPM.
2. Connect the electronic Speed Controller to the Computer via USB to read EMF feedback data for RPM
3. Plug in the AC-DC power supply to turn on the speed controller and for the speed controller to apply power to the motor.
4. Press the button on the breadboard to start the Arduino's acceleration mode.



Object Recognition:

1. Make sure to place your *detect_objects.py* and your fine-tuned YOLOv8 model (*yolov8_tuned.pt*) in the same directory
2. Make sure you have a >=3.8 version of Python installed
3. Install the ultralytics library with the command: *pip install ultralytics* or *pip3 install ultralytics*
4. Activate your webcam
5. Run the script with the command: *python3 detect_objects.py*

Here is how the overall object detection works:



Robotic Arm:

1. Inspect the finger to make sure it is holding together
2. Ensure the parts are well attached
3. Use finger for tests

IV. Testing Procedure

Motor Controller:

1. Plug in Arduino via USB to computer for power and changing direction
2. Plug in the AC-DC adapter for powering the motor
3. Plug in the speed Controller via USB to the computer to read EMF feedback

4. Press the start button on the breadboard
5. Record EMF feedback via a camera during acceleration phase
6. Record Motor rotation during constant speed phase via camera to check EMF's validity
7. Record EMF feedback via camera during deceleration phase
8. Change direction in Arduino IDE and repeat steps 4-7

Object Recognition:

1. Activate your webcam
2. Navigate to the directory that contains your *object_detection.py* Python script and your fine-tuned pre-trained model *yolov8_tuned.pt*
3. Run the script: *python3 object_detection.py*
4. Wait for the screen to show your real-time webcam recordings
5. Place objects to detect in front of the webcam
6. Check whether there is a bounding box around the object to determine whether the model recognizes an object is present in the frame
7. If a bounding box exists, check the label around the bounding box to determine whether the object has been classified correctly

Object	Detected Object in Frame?	Classified Object Correctly?

Robotic Arm:

1. Pick up finger
2. Move at the hinge joints using attached strings or manually with your hand to test angle that can be achieved with the printed parts

V. Measurable Criteria

Motor Controller:

1. Linear relationship between rpm and time during acceleration and deceleration phases
2. Comparable EMF feedback to recorded rpm during constant speed phase

3. Motor is capable of being controlled both clockwise and counterclockwise

Object Detection:

1. The model should detect that there is an object in the frame at least 80% of the time
2. The model should correctly classify the object at least 60% of the time
3. The model should detect the object's presence within a second

Robotic Arm:

1. Fingers can close up to 290 degrees
2. Parts are still mechanically sound: no cracks, breaks, or bends

Hardware PinOut(Motor Controller):

Pin	Usage/Description
6	PWM analog 0-5V->Motor Controller(Speed)
7	Digital Input_Pullup<-Button(on/off)
11	Analog High/Low(0/5V)->Motor Controller(Direction)