**ED2 Hardware Mounted Pointing System - Software Requirements Specification**

| **Version** | **Date** | **Description** | **Author(s)** |
| --- | --- | --- | --- |
| 1.0 | 2023-04-17 | Initial draft of requirements document | Kevin Ehrichs, Alexis Bernal |

1. **Introduction:**

1.1 **Purpose:** The purpose of this document is to define the software requirements for a system controlled by a microcontroller to instruct a stepper motor to rotate 0 to 360 degrees. This system will require the selection of a microcontroller platform, a stepper motor, and a driver. The initial phase of the project involves getting one stepper motor up and running, but the system must be scalable to include multiple stepper motors. The commands sent to the microcontroller must have a clear pattern that can be easily inferred.

1. **Requirements**

2.1 **Platform Selection:** The system must be based on a microcontroller platform. The platform must be selected based on the following requirements:

// **Platform that we will be using is the Arduino Uno.**

* Ability to control stepper motor(s)
  + // **We have full control of the stepper motor, can rotate it at any degree and control its speed (Pulses/Rev).**
* Availability of libraries and APIs to support stepper motor control.
  + // **Currently not in need of any libraries, but if we do, there are libraries and APIs and that will help.**
* Sufficient number of input/output pins to control the required number of stepper motors
  + //**The Arduino Uno has 14 digital pins, and we only require 3, so we have 11 left that can be used to function any additional system.**
* Availability of programming tools and support for the selected platform
  + **// Enough programming tools and support to fully develop this project.**

2.2 **Stepper Motor Selection:** The system must include a stepper motor. The motor must be selected based on the following requirements:

// **The motor we have decided to use is the NEMA 23Y.**

* Appropriate torque and speed for the intended application
  + // **Enough torque to move the system mounted onto it and the perfect speed for quick and precise adjustments.**
* Availability of appropriate driver for the selected motor
  + // O**ur motor will work best with what we have selected which is the TB6600 Micro-Step driver.**
* Compatibility with the selected microcontroller platform
  + // **Our microcontroller platform works perfect with our motor, allowing us to take full advantage of our motor without any hiccups.**

2.3 **Stepper Motor Driver Selection:** The system must include a stepper motor driver. The driver must be selected based on the following requirements:

**//The driver we will be using is the TB6600 Micro-Step driver.**

* Compatibility with the selected stepper motor
  + //**The driver chosen works best with the NEWA series motors, it is easy to setup and wire together due to the motor not having too many wires and clear labeling on the cover of the driver.**
* Compatibility with the selected microcontroller platform
  + //**Works great with our Arduino Uno board, the simple wiring works great, and we can control the driver using our platform.**
* Adequate current handling capability for the selected motor
  + //**Yes, 2.8 A, and this current can be handled by our Arduino Uno board as well.**

2.4 **Rotational Control:** The system must be capable of instructing the stepper motor to rotate from 0 to 360 degrees. The control must be precise and accurate, with a resolution of at least 1 degree. Our shaft should not be able to rotate MORE than 360 degrees, due to the wires of our system mounted, avoiding the twisting of cables and malfunction of our project.

//**Check, the motor rotates in a CCW direction with a 0.9 angle step, but will work on the option of being able to rotate it also in CW.**

//**Check, the final and full functional code of the motor works and passes all the tests that check if our code is fully functional, and the programming of the motor sets it to have a radius of 360 degrees to avoid the breaking/tearing of cables.**

2.5 **Scalability:** The system must be scalable to include multiple stepper motors. The number of stepper motors that can be controlled by the system should be configurable and must not be limited by the selected microcontroller platform.

//**N/A, have not tried adding in another motor to our system.**

2.6 **Command Pattern:** The commands sent to the microcontroller must have a clear pattern that can be easily inferred. The pattern should be consistent across all commands and must be documented in the system manual.

**//The controlling and commanding of the motor are quite simple and easy to follow and use. As it guides you while choosing rotational direction and degree adjustments. Can also set a clear degree change by using a “clock” with certain angle adjustments or use a text box where a precise angle adjustment can be inputted and set that degree.**

1. **Non-Functional Requirements**

3.1 **Reliability:** The system must be reliable and must not fail during operation. The system should be able to recover from any errors or faults without losing data or causing any damage to the system or components.

3.2 **Performance:** The system must be capable of controlling the stepper motor with high accuracy and precision. The system should be able to respond to commands promptly and provide real-time feedback on the stepper motor's status.