# Sequential motion - Upper Mcecsbot

**Project :ECE 406**

**Prof. Marek Perkowski**

**Krish Ramkumar**

**Index**

Introduction ----------------------------------------------------------------------------3

Setup--------------------------------------------------------------------------------------4

Connections------------------------------------------------------------------------------5

Problems resolved----------------------------------------------------------------------6

Functionality------------------------------------------------------------------------------7

References----------------------------------------------------------------------------------8

**Introduction:**

Mcecs bot is a humanoid guide robot with a base with rotatory wheels and an upper body.

The purpose of this project was to control the upper body/waist of the Mcecs bot to perform a sequence of motions. The robot moves in a sequence initiated by the user. The main objective of the project was met in most ways. These motion allow the waist of the Mcecs bot to move sideways forward backward upward in a sequential manner. The robot is operated primarily by motor controllers and an open source single board microcontroller (Arduino). The upper part of the robot consists of four linear actuators that are controlled by the two motor controllers.

**Setup:**

The robot uses

1. 2 sabertooth 2x5 motor controllers
2. 4 Firgelli actuators
3. 1 Arduino Mega

Sabertooth 2x5 :

The connections of the sabertooth to the Arduino are given in the figure below.



Fig1



When you are connecting the driving outputs, we must make sure to connect systematically. For instance one actuator should be connected to M1A and M1B. Since M1A and M2B operate a single actuator, connecting one wire of the actuator to M1A and one to M2A/M2B will give an error. The Baud rate selected for this specific project was 9600. The Baud rate tells us how data is transmitted between the arduino and the sabertooth.



Fig2

 **Connection setup:**



Fig3



|  |  |
| --- | --- |
| Fergelli actuator wires | Where they are connected |
| Red wire | M1A/M2B |
| Black wire | M1B/M2A |
| Yellow wire (potentiometer wiper) | Analog pin A0-A4 on the Arduino mega  |
| White/grey wire | 5v pin on Arduino mega |
| Blue wire | Gnd pin on Arduino mega |

The voltage that was used to power the motors were around 8v from an external power supply. Supplying higher voltage made the motors unstable.

**Problems and points resolved:**

The potentiometer in the fergelli actuators determines the position of the actuators.

The potentiometer had values ranging from 0- 1023 with 0 being maximum (actuators fully extended) 1023 being minimum (Fully retracted).This range was found by using a manual potentiometer, when the potentiometer knob was turned completely to both sides. It was first test without feedback. But since these were the absolute maximum and minimum cutoff values, the maximum and minimum values for this particular project were reduced to 150 and 875 respectively to avoid damage to the actuators from extensive pressure.

The baud rate which controls the transmission of information between the sabertooth and Arduino gave minor problems but was fixed with a specific baud rate of 9600. Any other baud rate other than 9600 did not seem to transmit data.

**Functionality:**

**Initializing the forward and reverse motion was applied from a range of values of the motor controllers that determine the forward or reverse motion of the actuators. The table for the range is given below. These values determine how much the actuators move forward or in reverse. One must not confuse these values with the potentiometer values of variables max and min (150 and 875 respectively). The potentiometer value determines only the present position of linear actuator.**



Apart from the initializing functions, the program that runs this robot has five main functions namely maximum(), minimum (), middle(), Allup() and Alldown() respectively.

As states earlier, the potentiometer value determines the position of the actuator. The max value is 150(fully extended), min value is 875(fully retracted) and middle value is 500 (mid position). These max and min values can be changed at any point. We just have to make sure to change the values for variables max ,min and mid in the program.

* Maximum() function validates the present actuator value with the final and enables the actuator to move forward (upward) to its maximum position value which is 150. If the actuator value is greater than 150, it will force the actuator to move forward till it reaches the value of 150. This is done for each actuator separately with the corresponding sabertooth motor controller associated with it.
* Minimum() function is similar to maximum in working except it validates for the minimum position value which is 875. Any value less than 875 will force the actuators to retract until it reaches the minimum value of 875. This function is also executed for each actuator separately with the associated sabertooth.
* Middle() function validates for the mid value of 500. The motive of this function is to bring the actuator to the middle position (value 500) and stop. This is slightly more misleading than the maximum and minimum functions because actuators do not always stop at the exact position every time. Sometimes they might stop at 505 or 497. For this reason there is a leverage margin added to the validation. 500 +/- 10 is considered as middle. If the actuator is above the middle value, it will move forward (upward) until it reaches 500 +/- 10 and if the value is below the mid value, the actuator will move down (reverse) until it reaches 500 +/-10. This function is also done for each actuator separately so that it will allow us to move one actuator alone to the middle if necessary.
* Allup() function validates all actuators simultaneously and enables them to move upward (forward) till all actuators reach maximum position value (150). The four actuators will move upward (forward) until the validation is satisfied. This is similar to the maximum function except it takes care of the case where of all four actuators need to move upward together.
* Alldown() function does a similar validation to check for the minimum position for all four actuators.

These functions are finally executed through the motion() function. We can manipulate the robot in anyway by calling these functions in a sequence we please.

**Future additions:**

We could make the upper body of the Mcecs bot stronger by reinforcing it with another plate and then try to increase the speed of the movement since it is unstable at higher speeds. Speeding up the motions will certainly make the motions look more elegant. The motions can also have voice activated input to make it more user friendly.

References and Acknowledgements:

Dr. Marek Perkowski

Actuators and You, An In Depth Guide, David Gaskin

The data sheets for the sabertooth2x5 and arduino and fergelli actuators were also provided in the Actuators and you manual.