



# *Self-Adjusting Antenna Array*

*Tagwa Abdelrahman, Andrew Boggio-Dandry*

# Member Contributions

## Tagwa

Modeling: motor and driver circuit.

Implementation: hardware setup, control code.

Analysis: performing tests, calculating results.

## Andrew

Modeling: physical system.

Implementation: hardware setup, control code.

Analysis: performing tests, calculating results.



# Original Scope

- Develop a cyber physical system to manage the spacing between antennae in an array
- 8 antennae were planned to be arranged on a plexiglass or similar surface and moved independently using 8 stepper motors
- A C-program running on a Raspberry Pi was to control the motor movement to create the desired antenna spacing
- The distance between antennae was to be verified using ultrasonic sensors



# Physical System Modeling

$$\lambda = \frac{c}{f}$$

$$\frac{\lambda}{2}$$

Where  $c$  denotes a constant, the speed of light,  $f$  represents the frequency, and  $\lambda$  denotes the wavelength.

To ensure optimal performance, the antennae will have to be separated by a distance of  $\lambda / 2$ . This will avoid the “standing wave” phenomenon. Voltage nulls appear at half-wavelength points.

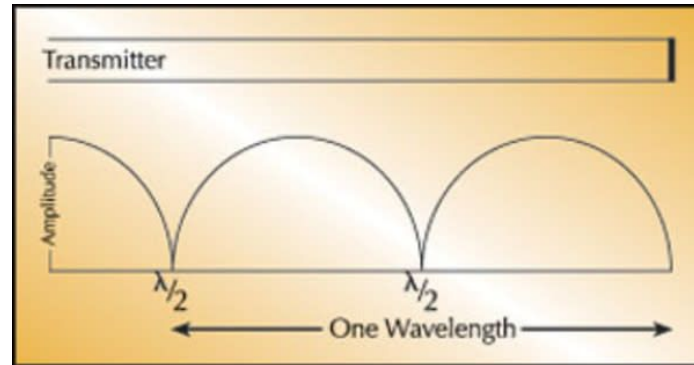
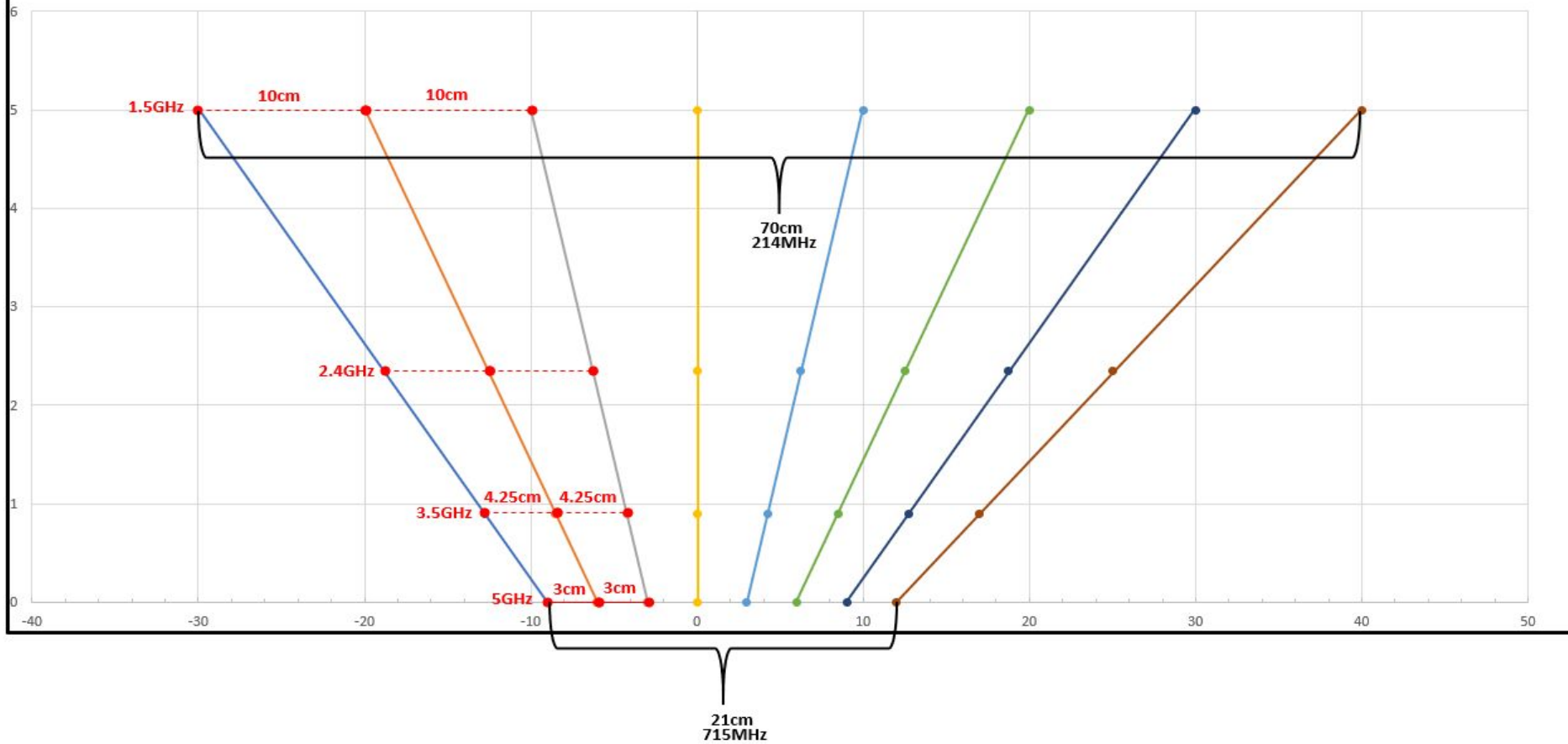


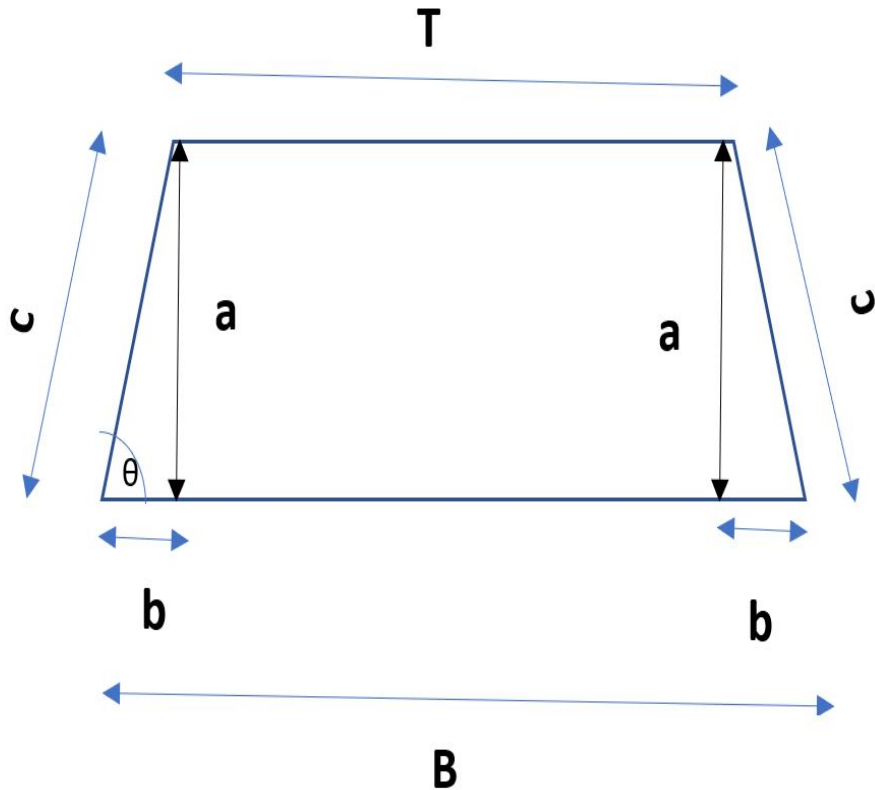
Image credit: <https://www.radioworld.com/misc-1/standing-waves-and-antennas>

# Planned Array Design

Array of 8 Antennae: Range: Minimum 214MHz, Maximum 5GHz



# Calculating Distance Between Linear Tracks



$$\sin \theta = \frac{\textit{opposite}}{\textit{hypotenuse}}$$

$$\sin \theta = \frac{a}{c}$$

From the triangle:

$$c^2 = a^2 + b^2$$

$$b = \sqrt{c^2 - (c * \sin \theta)^2}$$

From the Trapezoid we have:

$$T = B - 2b$$

$$T = B - 2(\sqrt{c^2 - (c * \sin \theta)^2})$$

$$B = T + 2(\sqrt{c^2 - (c * \sin \theta)^2})$$

## Motor Modeling

$$v(t) = Ri(t) + \frac{Ldi(t)}{dt} + k_b w(t)$$

$$T(t) = k_T i(t) - \eta \omega(t) - \tau(t)$$

$$T(t) = I d\omega(t)/dt$$

# Motor Modeling Continued...

- From the motor data sheet 42BYGH48 stepper motor:

$$i = 1.2 \text{ amp}$$

$$v = 24v$$

$$R = 2 \text{ ohms @ } 25c$$

$$L = 2.8 \text{ mH}$$

$$T = 0.45 \text{ Nm (3.98 lb.in)}$$
$$(0.045 \text{ Kg.m})$$

$$I = 68 \text{ g.cm}^2 \text{ (680Kg.m}^2)$$

$$W = 360/1.8 = 200 \text{ RPM}$$

- From equation 3:

$$T(t) = I \, d\omega(t)/dt$$

$$d\omega(t)/dt = 0.045/680 = 6.6 * 10^{-5}$$

- To calculate the torque of the load, assume we have:

$$\text{Maximum weight} = 4\text{oz}$$
$$= 0.1133 \text{ kg}$$

$$\text{Distance} = 500\text{mm} = 0.5\text{m}$$

$$a = d\omega(t)/dt = 6.6 * 10^{-5}$$

$$\text{Torque} = \text{force} * \text{distance}$$

$$\text{Force} = m * a$$

$$\tau = 0.1133 * 0.5 * 6.6 * 10^{-5} = 0.0000353 \text{ N.m}$$

- From equation 2:

$$\text{Assume } \eta = 0$$


$$T(t) = kT i(t) - \tau(t)$$

$$0.45 = kT * 1.68 - 0.0000353$$

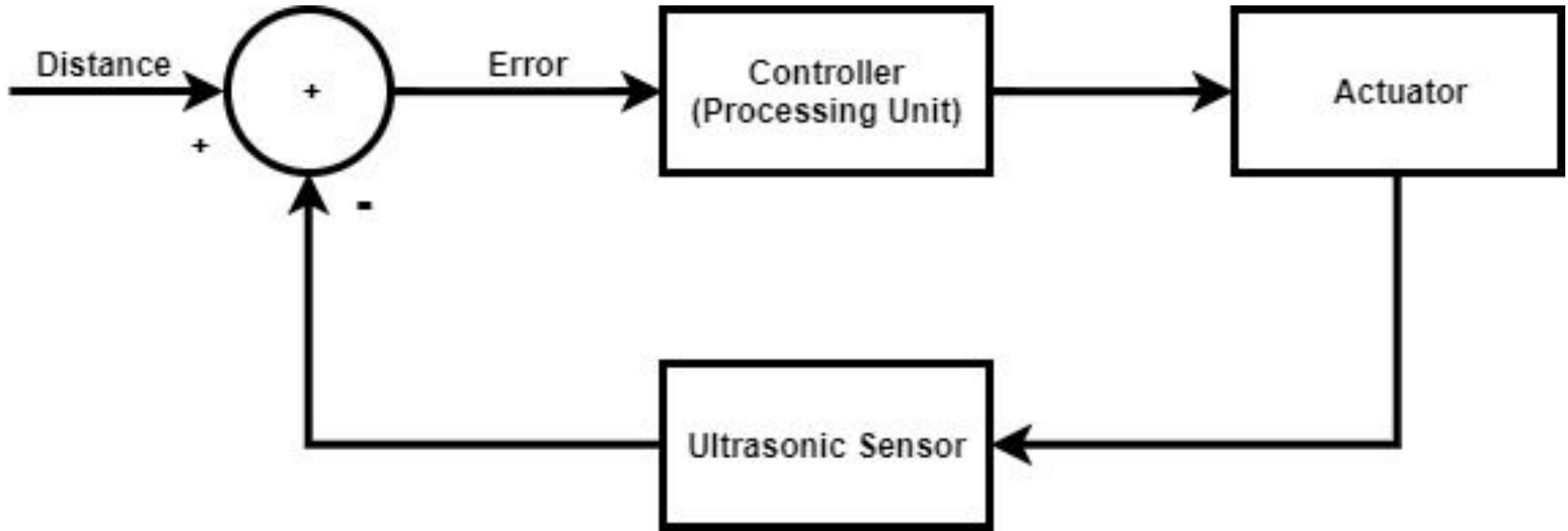
$$kT = 0.2678 \text{ N.m/amp}$$



# Project Scope

- We developed a cyber physical system to manage the spacing between 2 antennae in an array
  - A linear actuator was used to move an antenna using a stepper motor and leadscrew
  - The second (virtual) antenna would be on a linear actuator, placed back-to-back
  - A C-program running on a Raspberry Pi controls the motor movement to create the desired antenna spacing
  - The distance moved is verified using an ultrasonic sensor (not accurate)
  - The antenna returns to the starting position at the end of the program based on user input.
- 

# System Block Diagram

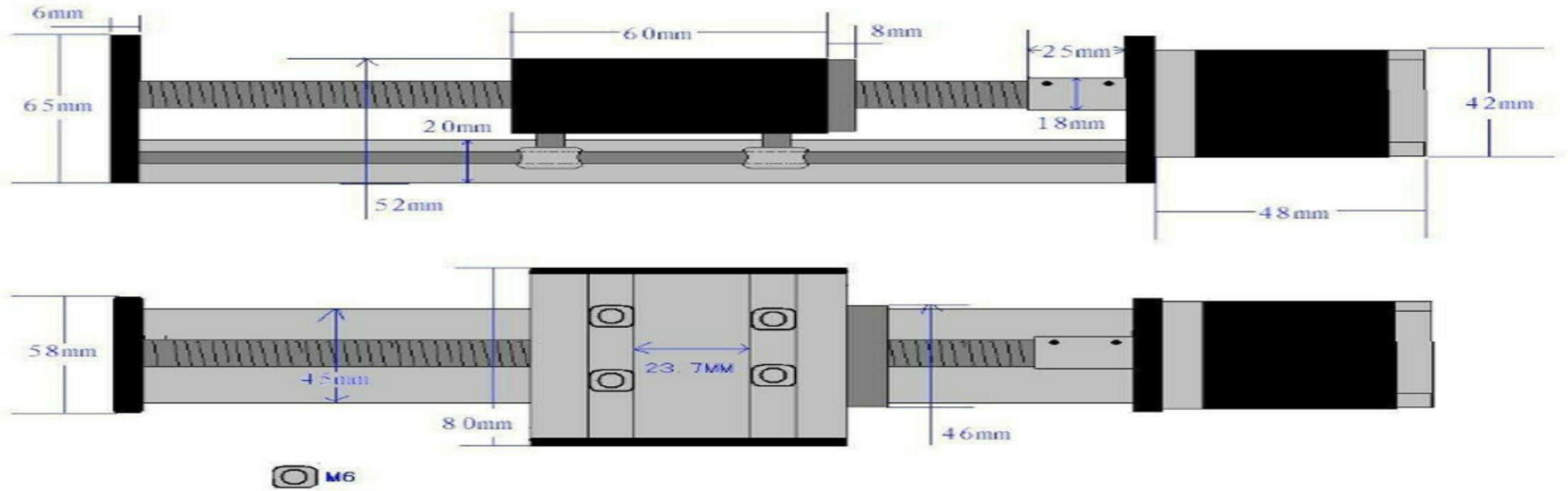


# Design Components

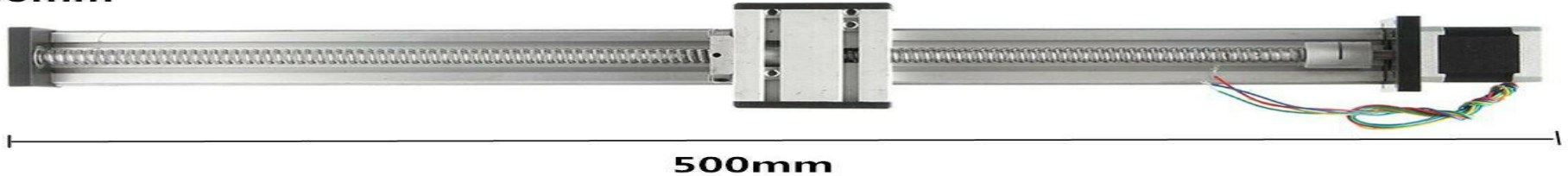
- Ball Screw Linear Slide Stroke Stage Actuator (Model 581770)
- 42BYGH48 Stepper Motor
- DRV8825 Driver Circuit
- HCSR04 Ultrasonic Distance Sensor
- Cardboard Rectangle (stand-in for antenna)



# Linear slide actuator




80mm



500mm

# Specifications of the linear actuator

- 42BYGH48 stepper motor attached at end
  - Rated Dynamic Load: 40KG
  - Rated Static Load: 60KG
  - **Voltage: 24V**
  - **Current: 1.2A**
  - Step Angle:  $1.8^\circ$
  - Torque: 45N.cm
  - **Precision: 0.02mm**
  - Round Trip Accuracy: 0.05-0.08mm
  - Precision: 0.001mm
- 

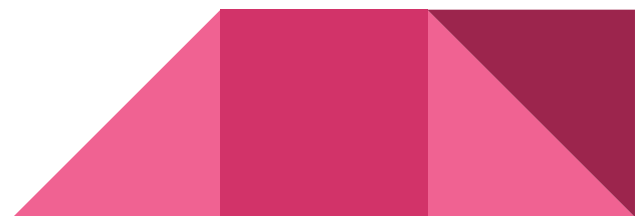
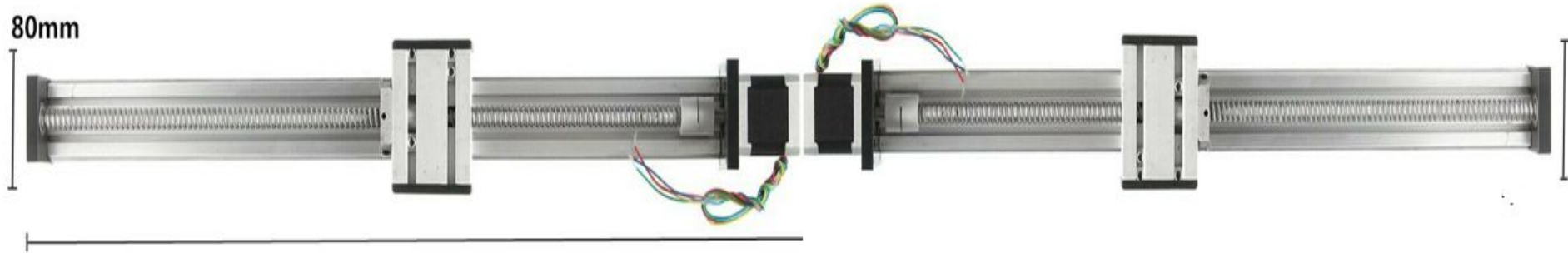
# The Achieved Antenna Array

maximum Distance= 852

Minimum Distance = 200mm

80mm

500mm



# Converting from Circular to Linear Motion

To calculate the linear motion, the following equation was used:

$$\text{Leadscrew} \cdot \left( \frac{\text{Revolution}}{\text{mm}} \right) \left( \frac{1}{\text{Microstep}} \right) \times \text{Motor} \cdot \left( \frac{\text{Step}}{\text{Revolution}} \right) = \frac{\text{Step}}{\text{mm}}$$

In order to calculate the lead screw's revolution per  $mm$ , the following equation was used:

Pitch is distance between screw grooves

$$\frac{\text{Revolution}}{\text{mm}} = \frac{1}{\text{pitch}}$$



# Precision Calculation

The step per revolution of our motor is 200 steps. Inserting that into the equation:

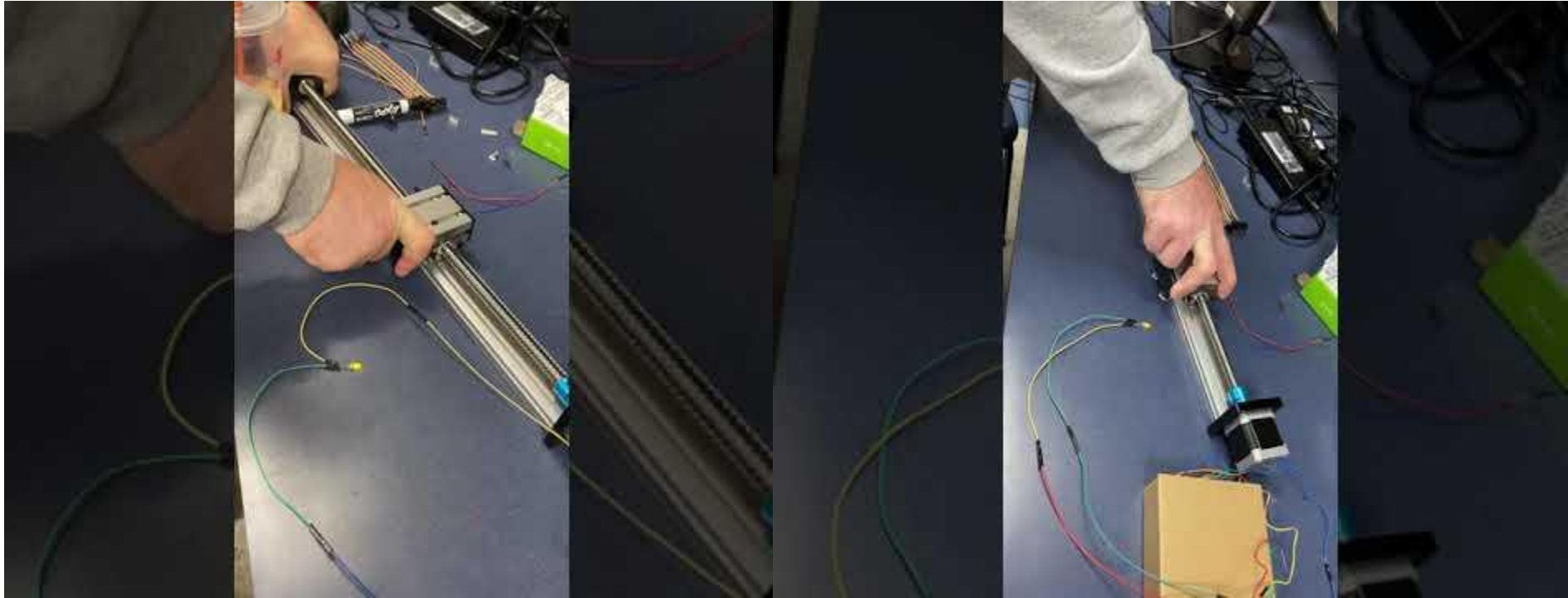
$$\frac{1}{4mm} \cdot 200step = 50 \frac{step}{mm}$$

Performing the division gives a result of  $0.02mm$ , meaning our precision is  $0.02mm$ .





# Find the lines for each coil of the motor

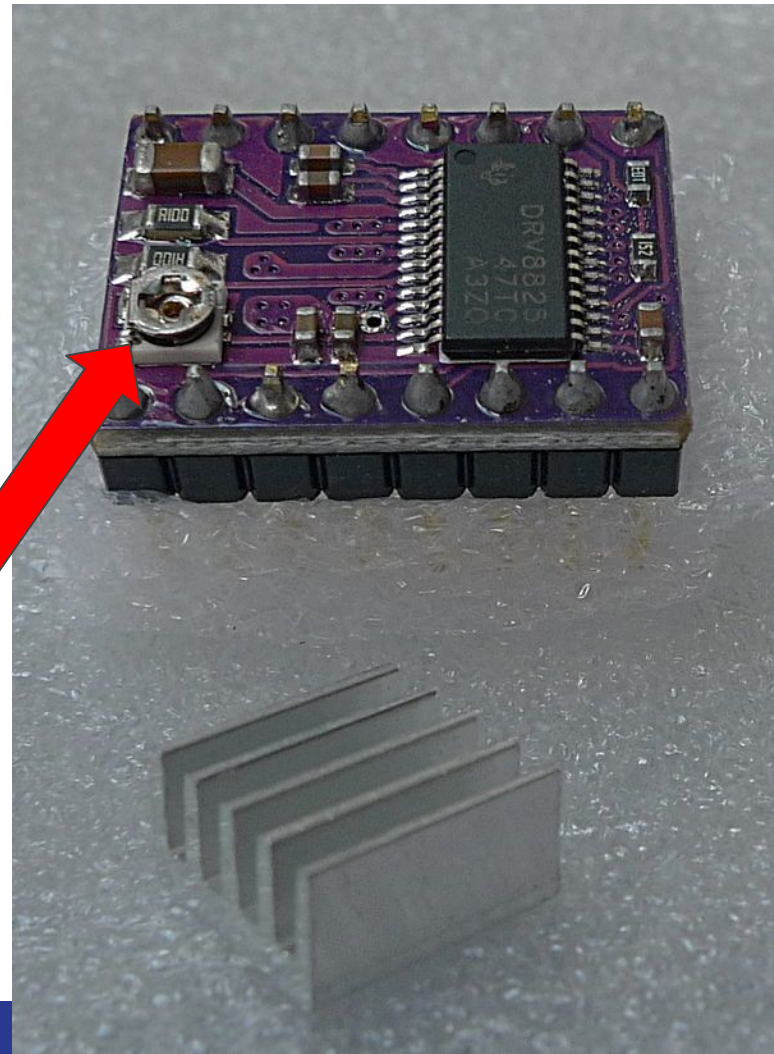


# DRV8825 Driver Circuit

The DRV-8825 maximum current limit ( $I$ ) is equal to twice the  $V_{ref}$  or voltage reference:

$$I_{limit} = V_{ref} \times 2$$

The driver circuit includes a potentiometer to adjust  $V_{ref}$  to limit the output current to the motor



# Drv8825 Continued

According to the motor's datasheet, only 71% of the maximum 1.2A should be supplied. Therefore, using the equation from the previous slide:

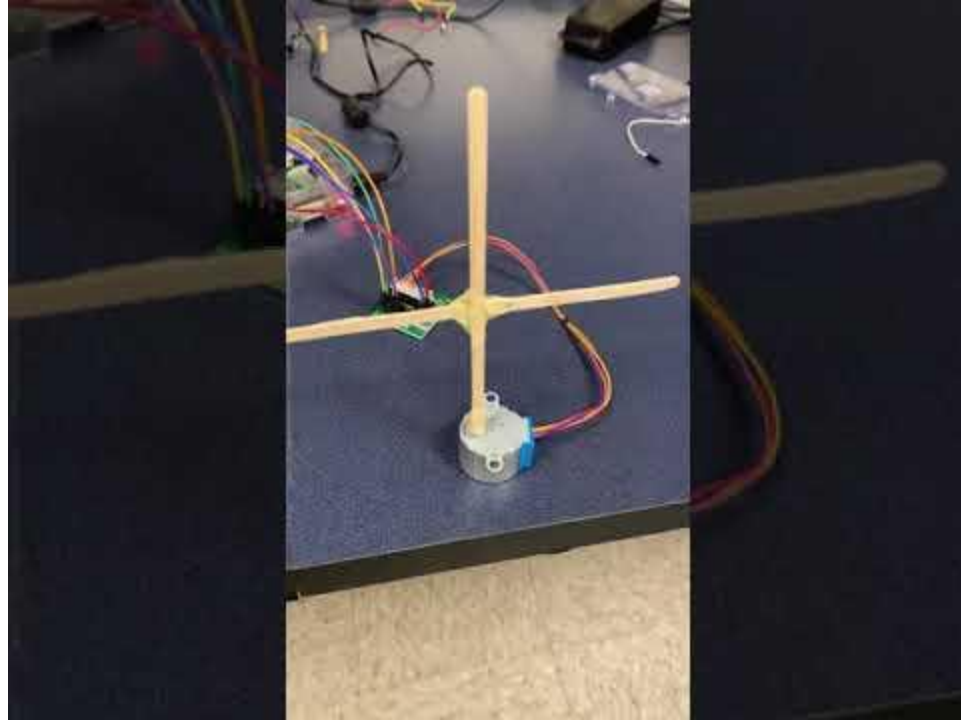
$$V_{ref} = \frac{0.71 * 1.2}{2} = 0.426V$$

As such, our DRV8825's potentiometer was tuned to 0.426V to supply 0.852A to the motor.

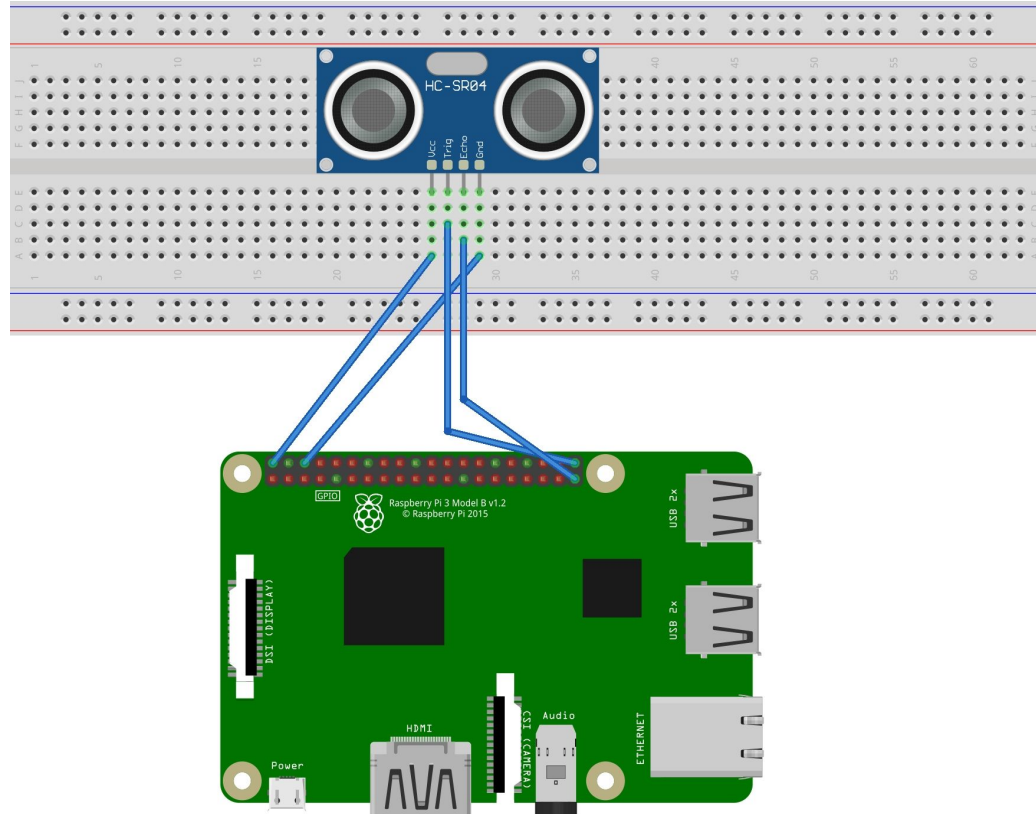




# Demonstration of Working Stepper Motor



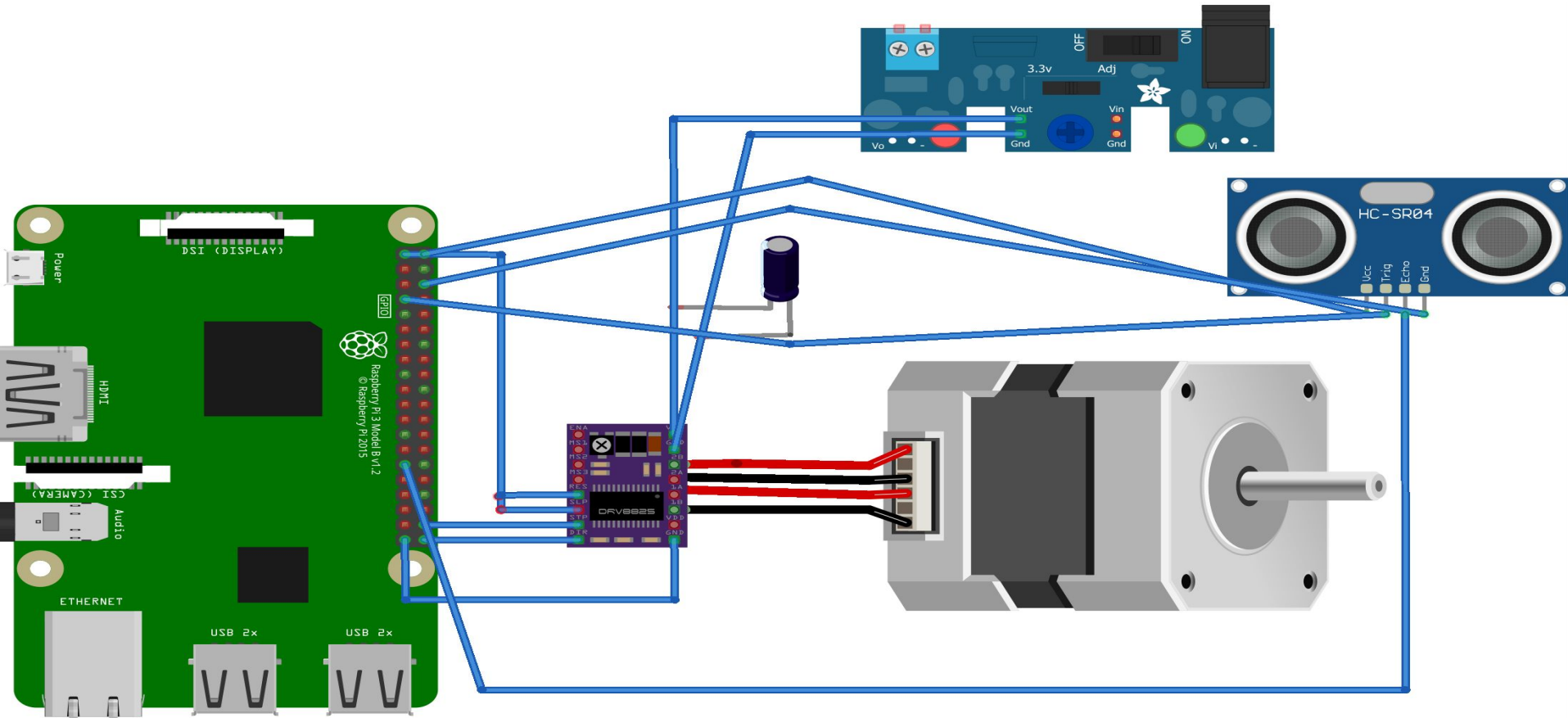
# Ultrasonic Sensor Connections to Raspberry Pi



# Demonstration of Operational Ultrasonic Sensor



# System connections





# Software Implementation

```
distance = (C / frequency) / 2;
distance -= 63; // this is the shaft and metal plate added together to the middle of the carriage
distance /= 2; // divide the distance in half, because each antenna only has to move half the distance
printf("Distance calculated to be: %.2lfmm\n", distance);

if(distance > MAXDISTANCE)
    error("This distance exceeds the maximum allowed distance of 852mm.");

if(distance < MINDISTANCE)
    error("This distance is less than the minimum allowed distance of 234mm.");

steps = distance / ACCURACY;
printf("Number of steps needed (with accuracy of 0.02mm): %d\n", steps);
printf("Homing the antenna carriage. Please wait . . .\n");
```

Calculating the distance from user-input frequency  
and the necessary number of steps

# Software Implementation

```
moveCounterClockwise(steps);

printf("Verifying the distance using ultrasonic sensor . . .\n");

if(getDistance() - SENSORSTART == distance)
    printf("No corrections needed.\n");
else if(getDistance() - SENSORSTART < distance)
{
    printf("Correcting distance . . .\n");
    moveCounterClockwise(distance - getDistance() - SENSORSTART);
}
else if(getDistance() - SENSORSTART > distance)
{
    printf("Correcting distance . . .\n");
    moveClockwise(getDistance() - SENSORSTART - distance);
}
```

Moving the carriage the ultrasonic sensor to verify correct position and make any necessary corrections

# Software Implementation

```
void moveClockwise(int steps)
{
    digitalWrite(DIRECTION, CLOCKWISE);

    for(i = 0; i < steps; i++)
    {
        digitalWrite(STEP, 1);
        delay(2);
        digitalWrite(STEP, 0);
        delay(2);
    }

    delay(1000);
}
```

```
void moveCounterClockwise(int steps)
{
    digitalWrite(DIRECTION, C_CLOCKWISE);

    for(i = 0; i < steps; i++)
    {
        digitalWrite(STEP, 1);
        delay(2);
        digitalWrite(STEP, 0);
        delay(2);
    }

    delay(1000);
}
```

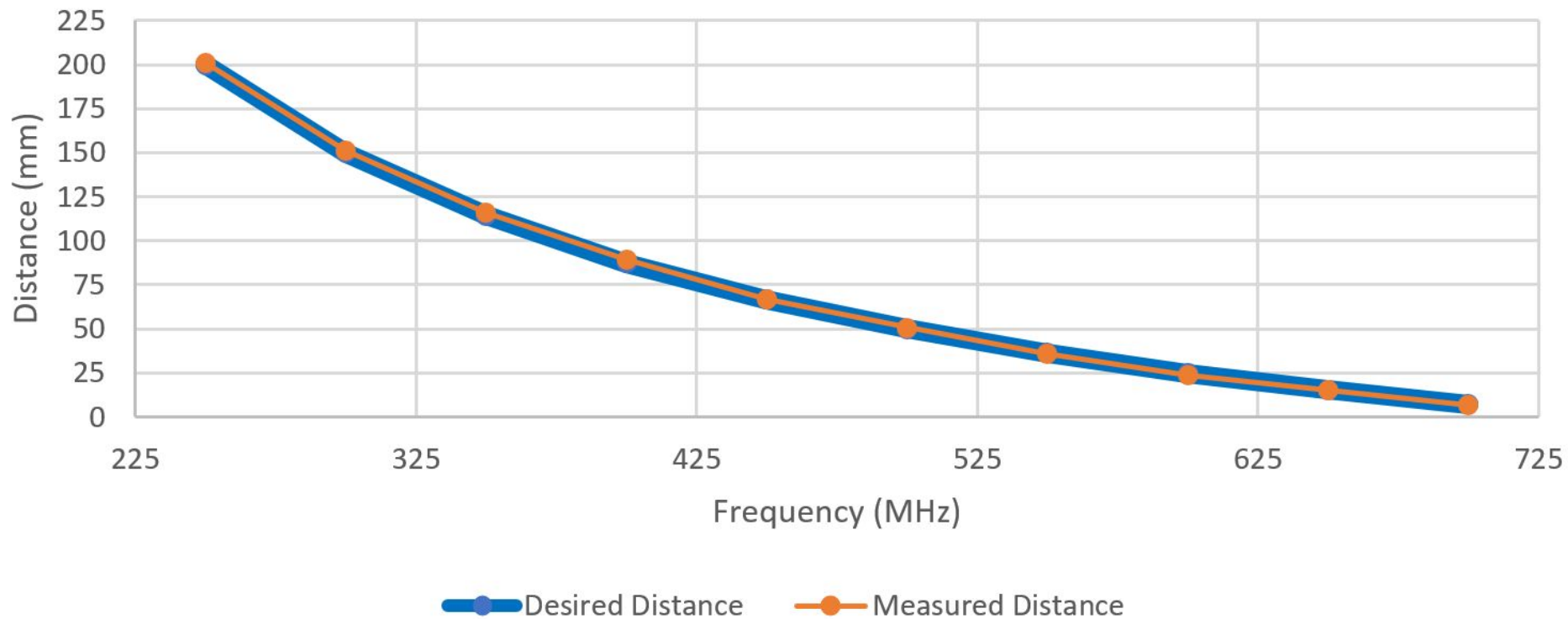
Code to control the step motor through the DRV8255 to move the linear actuator in the desired direction, for the desired number of steps.

# Analysis

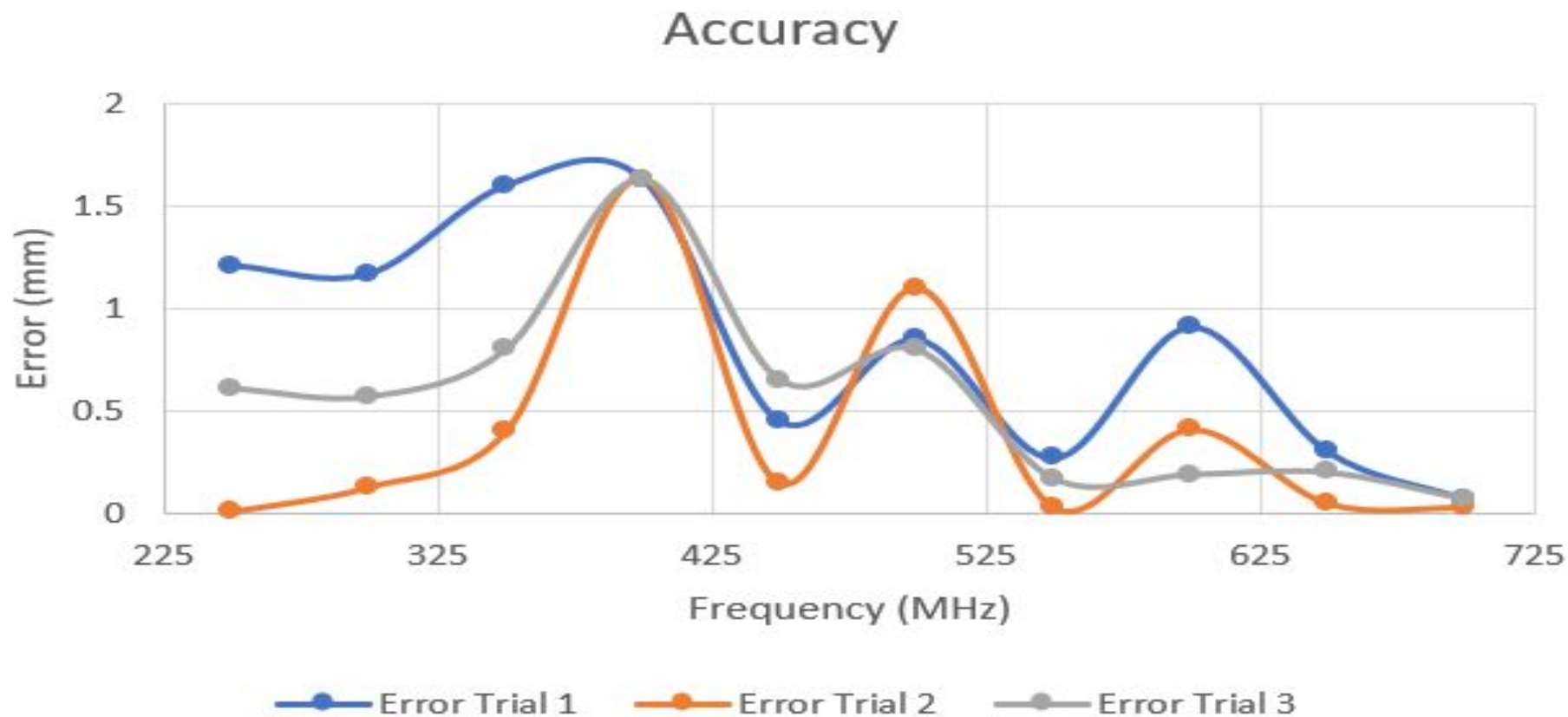
Frequency (MHz)	Total Desired Distance (mm)	Desired Distance Moved (mm)	Measured Movement (mm)			Error 1	Error 2	Error 3
			Trial 1	Trial 2	Trial 3			
700	214.14	7.07	7	7.1	7	0.07	0.03	0.07
650	230.6	15.3	15	15.25	15.1	0.3	0.05	0.2
600	249.82	24.91	24	24.5	25.1	0.91	0.41	0.19
550	272.54	36.27	36	36.3	36.1	0.27	0.03	0.17
500	299.8	49.9	50.75	51	50.7	0.85	1.1	0.8
450	333.1	66.55	67	66.4	67.2	0.45	0.15	0.65
400	374.74	87.37	89	89	89	1.63	1.63	1.63
350	428.8	114.4	116	114.8	115.2	1.6	0.4	0.8
300	499.66	149.83	151	149.7	150.4	1.17	0.13	0.57
250	599.58	199.79	201	199.8	200.4	1.21	0.01	0.61

# Analysis

## Accuracy



# Analysis (error)



# Demo











# Conclusions

- We were able to control the movement of the linear slide actuator in millimeter accuracy to a desired distance based on a user-input frequency
  - The ultrasonic sensor being used is not accurate enough to provide meaningful feedback
  - The dimensions of the linear actuator used prohibit distances in the higher frequency range
  - We were able to provide frequency range of 145.8 - 749.5MHz
- 

# Future work

- Implement the whole system that includes all 8 antennae
  - Update the software in order to control 8 linear actuators and ultrasonic sensors
  - Research more accurate distance measurement sensor
  - Find a better linear actuator for this project with a smaller plate, or build it, to be able to provide a higher frequency range
- 

Thank you!

The image features the words "Thank you!" rendered in a highly decorative, hand-drawn style. The letters are thick and filled with various colors and patterns. The word "Thank" is on the top line, and "you!" is on the bottom line. The letters are decorated with patterns like zig-zags, stripes, and floral motifs. There are several stylized flowers in blue, pink, and purple scattered around the text. The exclamation point is also decorated with a blue and green pattern.