**Bipolar Stepper Motor Implementation On Arm Manipulator 6 DoF With Master Controller**

Riska Wulandari1[[1]](#footnote-2), Setyawan Ajie Sukarno 1[[2]](#footnote-3) and Susetyo Bagas Bhaskoro 2[[3]](#footnote-4)

1*1Department of Automation Engineering, Bandung Polytechnic of Manufacturing, Bandung, Indonesia*

[riskaw0930@gmail.com](mailto:riskaw0930@gmail.com), setyawanajies@gmail.com, [bagas@ae.polman-bandung.ac.id](mailto:bagas@ae.polman-bandung.ac.id)

Keywords: Arm Manipulator, Stepper Motor, Master Controller

Abstract: Arm Manipulator is a robot that uses a motor with a shape like a hand to facilitate human work in the industrial sector. In this final project, a 6 DoF Arm Manipulator or in other words is a robotic arm that has 6 degrees of freedom. The arm manipulator is made using a stepper motor driven by a joystick called the master controller. Control is done remotely using serial communication. In this case, testing aims to determine how the implementation of the stepper motor on the 6 DoF arm manipulator and to determine the accuracy of the stepper motor based on the pulse given. The results obtained from the test are that the error at joint 1 is 0.27%, joint 2 is 0.4%, joint 3 is 0.13%, joint 4 is 0.09%, joint 5 is 0.15%, and joint 6 by 0.03%. From the testing of the joint, an average error of 0.14% was obtained.

# 1 Introduction

Robot technology is experiencing a very rapid progress today. Robots have replaced manual tools that require a lot of manpower. In the industrial world, robots are one of the tools that are needed under certain conditions. There are certain conditions in the industrial sector that are impossible to handle by humans. Robots have many advantages that humans do not have, namely producing the same quality when doing a job repeatedly, not getting tired easily, and can be reprogrammed so that it can be used for several different tasks. And among the robots that are often used in industry is the Arm Manipulator. The application of Arm Manipulators is found in many industries, such as welding, painting in the automotive industry and packaging and filling in the chemical industry.

In this final project, an Arm Manipulator is made that allows it to help human work. The Arm Manipulator has 6 degrees of freedom (DOF), each DOF is driven by a stepper motor controlled by the Arduino Mega microcontroller. The arm manipulator is controlled by a pendant called the master controller. Control is done remotely / remotely using serial communication. The master controller has approximately 7 sensors attached to it, each of which controls an actuator located on the joint arm manipulator. On the master controller there are also many buttons that are intended for various features such as scaling movement, start, stop, translational movement, and others.

In order to make a small contribution to the advancement of technology in the industry in Indonesia and with the aim of learning for writers and the next generation. This final project is designed which is a 6 DOF robotic arm controlled by a master controller. Controlled using 2 Arduino Mega which are communicated serially with the interface display in the form of an LCD display to display the position of the robot arm.

# 2 research methodology

First, the author identifies the problem, then makes demands for the design results to be achieved, after that makes the concept of the design system. After being conceptualized as a whole, if the system made is not suitable, then a re-conceptualization is carried out, if the system is appropriate, then the system installation and testing is carried out. If the system that has been tested is not suitable, then retest is carried out, if the system that has been tested is appropriate, then the system is good to implement.

**2.1 Tools and Materials**

The technologies used in this final project are:

1. Interfacing: LCD Display 20x4

2. Process: Microcontroller - Arduino Mega 2560

Hardware:

1. Stepper Motor : a. Nema 8 (8HS11-0204S)

b. Nema 11 (28HD1411-02)

c. Nema 17 (17HS4401)

d. Nema 23 (5718HB2401)

2. Stepper Motor Driver :

a. A4988 Driver Stepper Module

b. TB6600 Driver Stepper Module

3. Digital Input : a. Push Button

b. Toggle Switch

4. Digital Output: LED

5. Sensor: 100K Ohm Potentiometer

Software:

1. Arduino IDE

2. Solidworks 2020

3. Eagle PCB Design

**2.2 Work Principle**

Hardware-wise, the final project system consists of a 6 DOF arm manipulator and a master controller. Arm manipulator 6 DOF is an arm-shaped robot that has 6 joints of movement (joint). While the master controller is a pendant that controls the movement of the arm manipulator. The joint movement on the manipulator is generated from the joystick sensor readings attached to the master controller. One joint on the joystick sensor will control one joint on the arm manipulator. The master controller will generate degree data from sensor readings. The sensor data will be sent to the Arduino slave to be converted into pulse data to drive the stepper motor.

Technically, the robot arm is controlled by the Master controller and the reading data in the form of an angle will be displayed on the LCD display while sending data serially to the Arduino Mega Slave Controller for further data processing.

The user moves the robot arm manually by moving the joystick on the master controller. Then the movement that is read by the potentiometer in the form of data is processed by the program and displayed in the form of an angle.

The master controller is built on various input components and sensors, while the input components consist of a potentiometer, joystick module, push button, and toggle switch. The resulting data will then be processed by Arduino (master) as a controller. Then the data will be displayed through the LCD as well as sent via UART serial communication.

The data sent in the form of an angle will be received by Arduino (slave) and will be converted into digital pulses for which the motor driver will drive each stepper motor actuator located at each joint arm of the manipulator/robot arm.

**2.3 Convert Angle Data To Step Data**

Several stepper motors are used at each joint arm manipulator to generate movement and a motor driver is needed which is useful for generating periodic pulses in the stepper motor. All microstep settings on the TB6600 driver use 1600 Pulse/rev. The reason for choosing 1600 Pulse/rev is because based on calculations,

(1)

Where 360 ​​is the angle of one complete turn and 1600 is the number of pulses/rev used. Thus, it is known that the motion per pulse is 0.225°. When given 1 pulse, the motor will move by 0.225 ° which makes the movement of the stepper motor smoother. Current (current) that is set on the TB6600 is different at each joint, the current setting on the TB6600 is based on the specifications of each stepper motor used.

Meanwhile, at joint 6, the Nema 8 8HS11-0204S stepper motor uses the A4988 motor driver. To use this driver, it is necessary to adjust the current to limit the current beyond the rated current of the motor. Performed calculations,

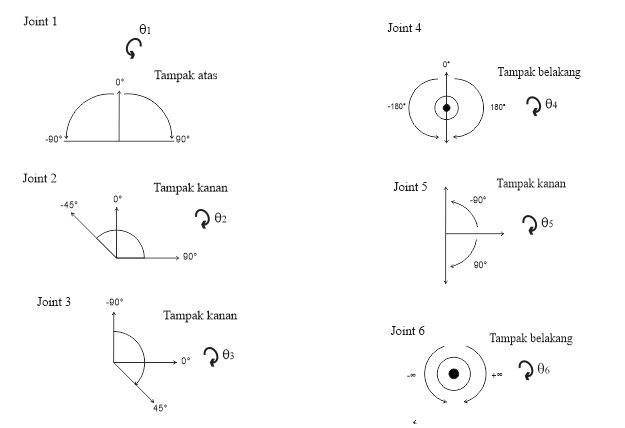
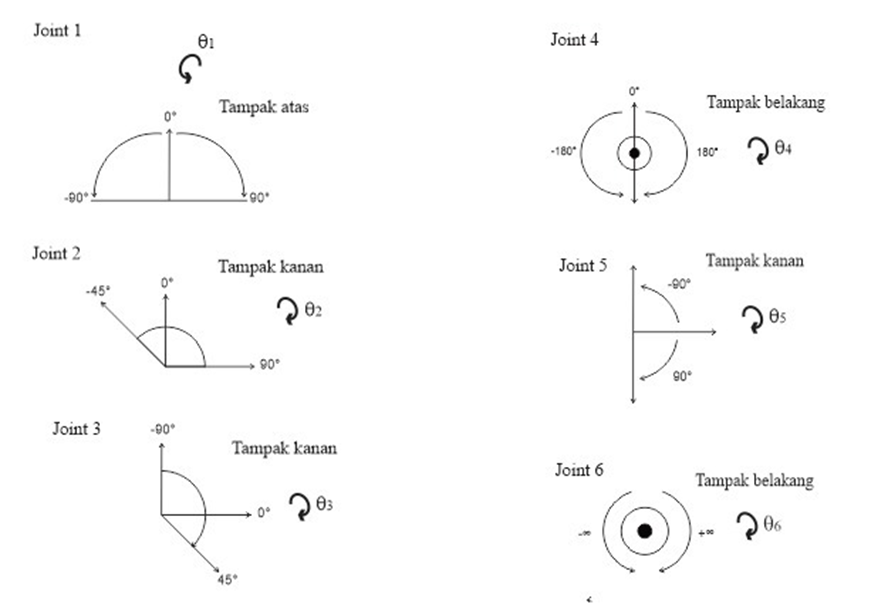
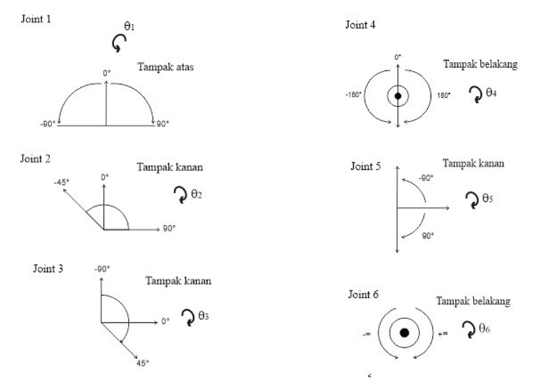
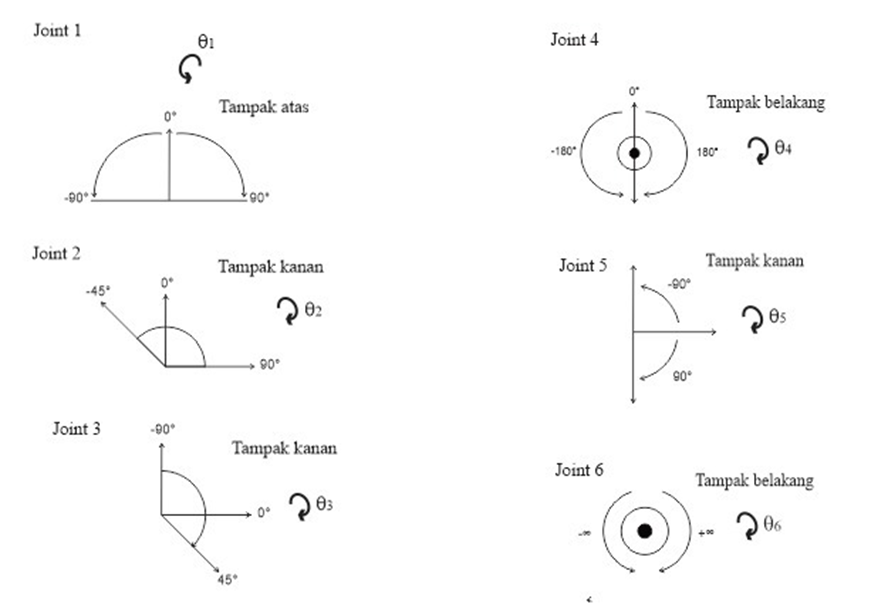
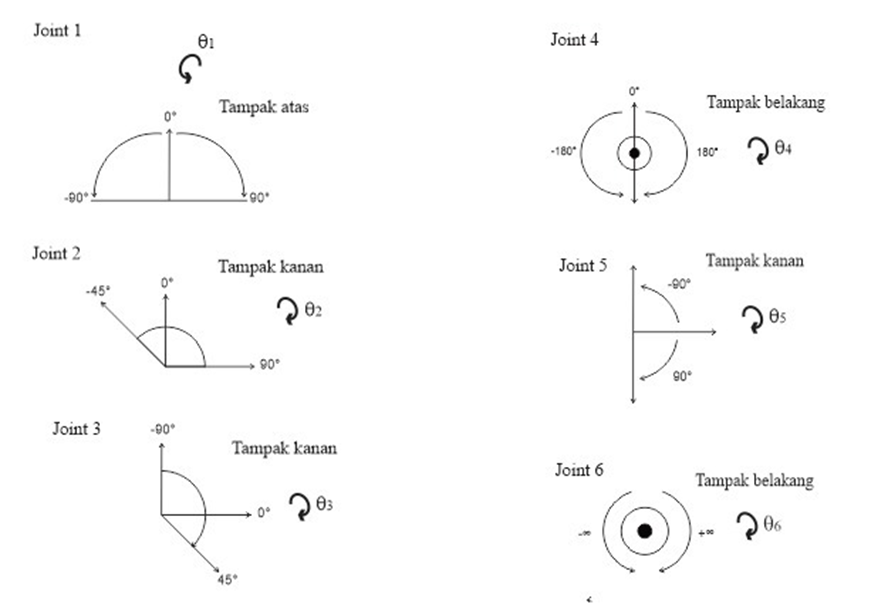
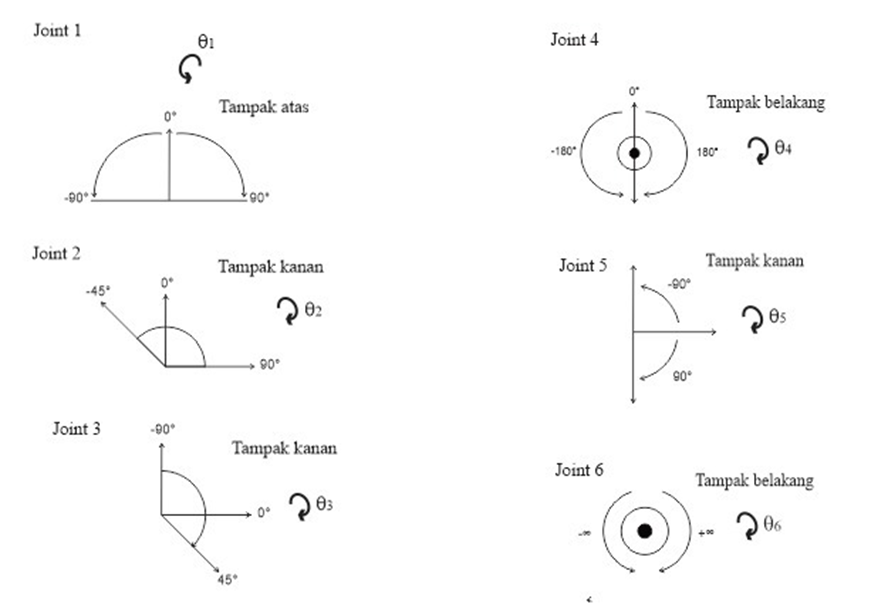
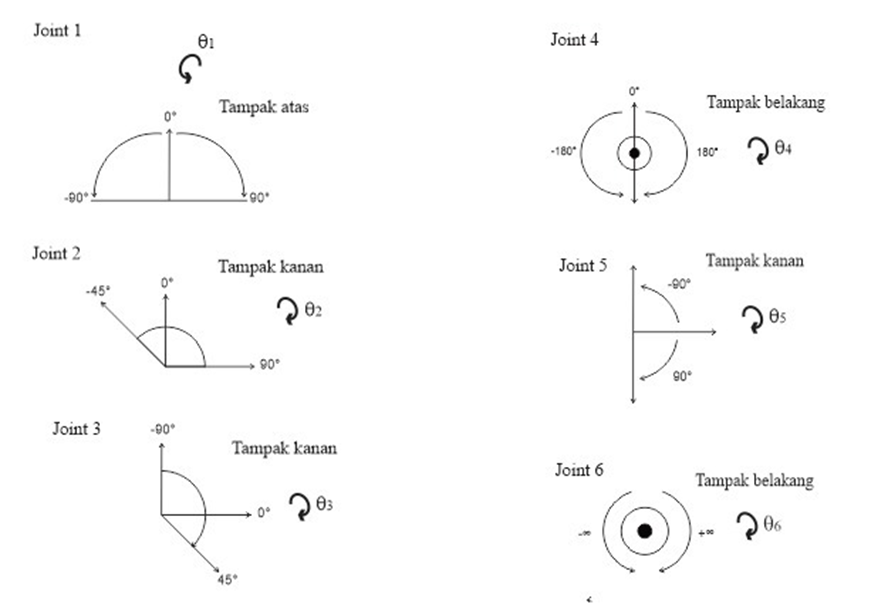
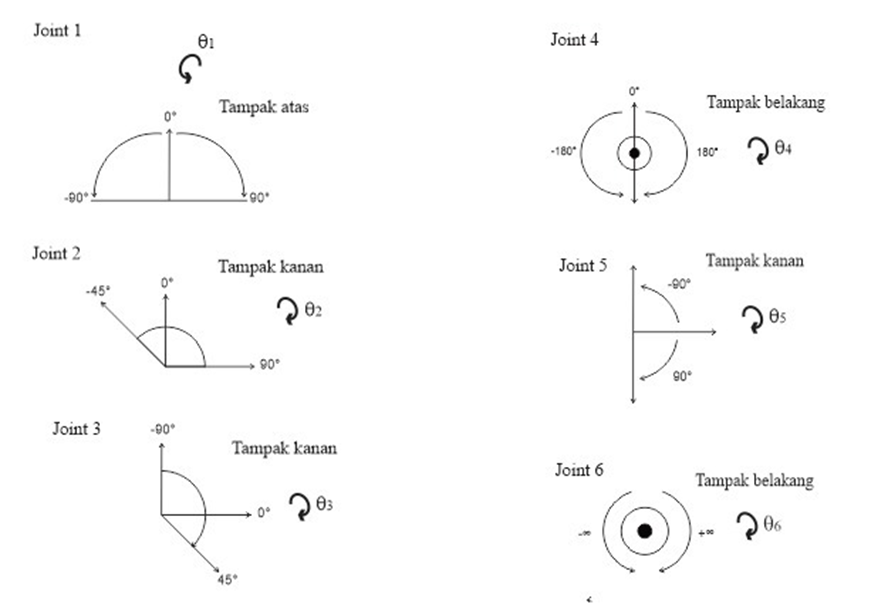
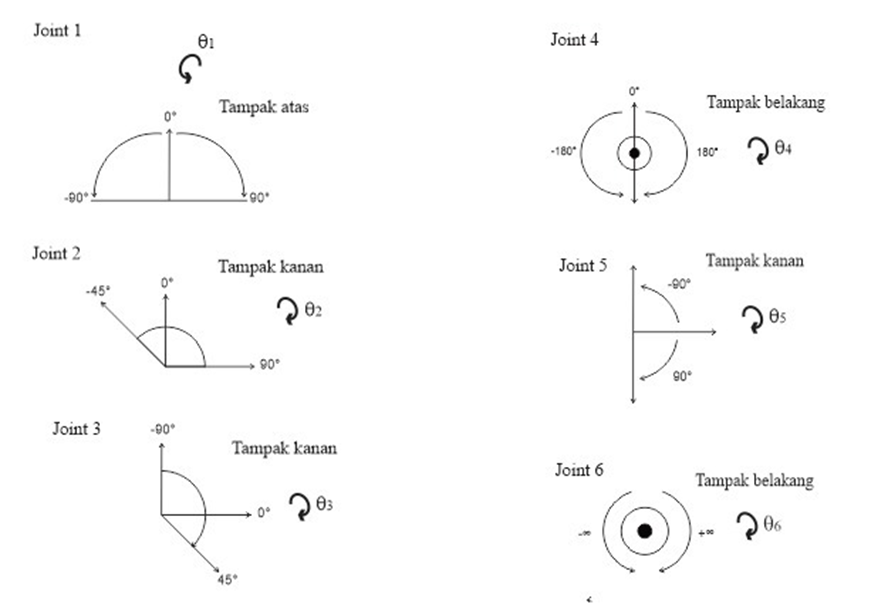
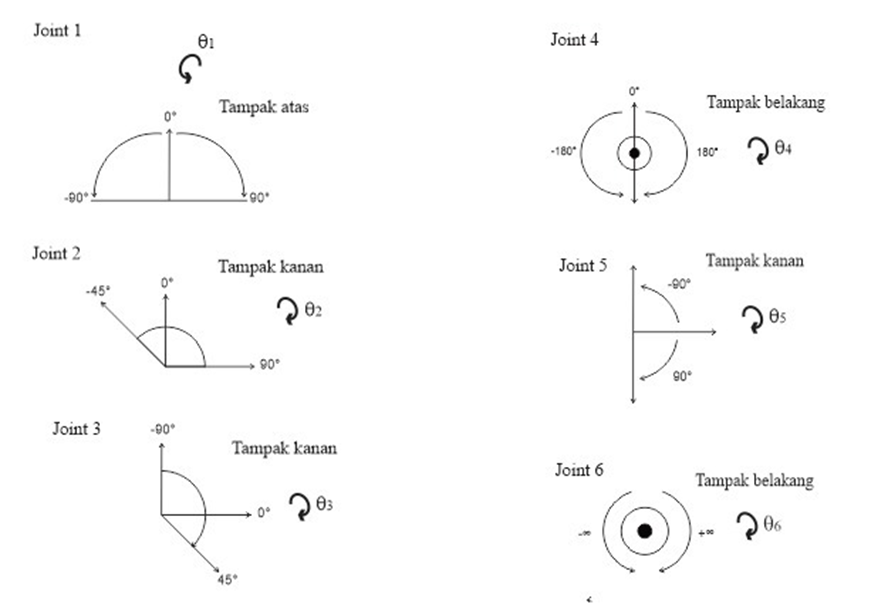
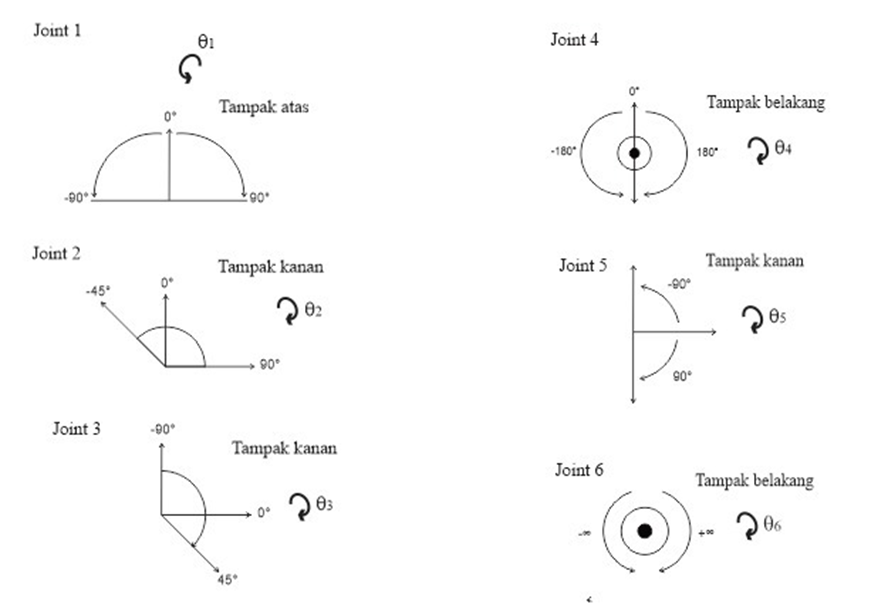
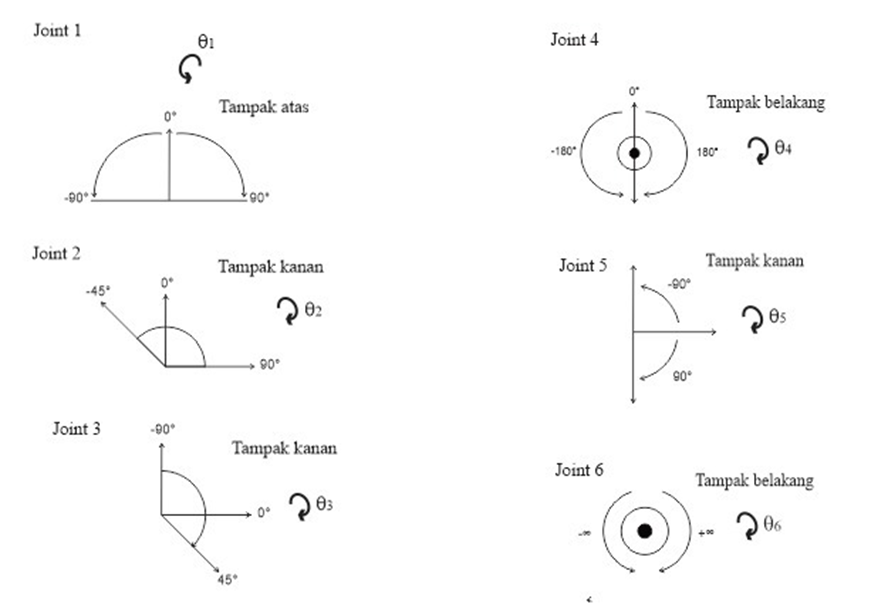
Is known:

The rated current of the Nema 8 8HS11-0204S stepper motor at joint 6 is 0.2 A.

Then we get Vref = 0.08 V.

The microstepping used is eighth step which means 1600 microstep/revolution.

In order for the stepper motor to move, pulses are needed to be given to the motor. This 6 DOF Arm Manipulator moves to follow the movement of the joystick on the master controller using Arduino serial communication which only uses 2 pins, namely RX and TX. The master controller receives data from the sensor in the form of degree data, the data will be sent to the Arduino slave to be converted into pulse data and executed to be able to drive the stepper motors in each joint. The first thing to do to change the degree data to step data is to do a calculation to get the step parameter at a certain angle. The following is a picture of the range of movement angles of each joint



The picture above is the angles of the desired range of movement. The movement generated from the stepper motor will be transmitted to the drive pulley. Calculations performed to determine the step parameters at a certain angle are based on the pulley used. This calculation focuses on the number of teeth on the pulley. The following is an analogy of the calculations performed,

Information:

A = number of teeth on drive pulley

B = number of teeth on the driven pulley

C = pulse/rev used

X = pulse/rev pulley drive

The following is pulley data based on the number of teeth used in each joint. The analogy of the calculation above is an analogy taken from the calculation of the speed of the gears which is felt to be almost the same as using the pulley.

Table 1 : Data pulley used

|  |  |  |
| --- | --- | --- |
| Joint | Drive Pulley (teeth) | Driven Pulley (teeth) |
| 1 | 20 | 96 |
| 2 | 20 | 80 |
| 3 | 20 | 100 |
| 4 | 20 | 56 |
| 5 | 20 | 43 |

Note: for joint 6 do not use a pulley because it is an end effector.

In table 2.11 it is known that each joint uses a pulley with a total of 20 joint teeth and the number of teeth on the driven pulley is different in each joint.

The result of the calculation is the step to move the pulley for one rotation or 360°. All these results are used as pulse parameters to move a certain angle by entering it in the program.

The results of these calculations are entered into the main program to be used as a certain angle step parameter. After obtaining the angle parameters and step parameters, the conversion can be done using the mapping function. This map function is used to convert a number in a range to another number range, the following is the program used:

// ==== convert data degree ke step

currentStateJ1 = thetaConversions(T1, minAngleJ1, maxAngleJ1, batasJ1Min, batasJ1Plus);

currentStateJ2 = thetaConversions(T2, minAngleJ2, maxAngleJ2, batasJ2Min, batasJ2Plus);

currentStateJ3 = thetaConversions(T3, minAngleJ3, maxAngleJ3, batasJ3Min, batasJ3Plus);

currentStateJ4 = thetaConversions(T4, minAngleJ4, maxAngleJ4, batasJ4Min, batasJ4Plus);

currentStateJ5 = thetaConversions(T5, minAngleJ5, maxAngleJ5, batasJ5Min, batasJ5Plus);

currentStateJ6 = thetaConversions(T6, minAngleJ6, maxAngleJ6, batasJ6Min, batasJ6Plus);

// ==== Function for converting degree to step ====

int thetaConversions (int angle, int minAngle, int maxAngle, int minStep, int maxStep) {

int compare = 0;

compare = map(angle, minAngle, maxAngle, minStep, maxStep);

if (compare > maxStep) {

compare = 0;

}

else if ( compare < minStep) {

compare = 0;

}

return compare;

}

In the above program, thetaConversions is a function to map the values to be converted. Inside the brackets there are T1-T6 which are the conversion values from angle data to step data. MinAngle and maxAngle are the range of angle values that will be converted according to the parameters declared above, while the Min and Plus limits are the desired step value ranges.

# 3 result

**3.1 Test**

Tests are carried out to see how the pulse data is calculated from the pulley and the accuracy at the specified angle. Based on the calculation of the gears using the number of existing teeth, the results obtained are in the form of steps to achieve 1 rotation of the gear or 360° by analogy to the following formula,

Information:

A = number of teeth on drive pulley

B = number of teeth on the driven pulley

C = pulse/rev used

X = pulse/rev pulley drive

To get the step results at a certain angle, a little additional calculation is needed. The following is an additional calculation to get the step results at a certain angle.

**3.2 Test Result**

The parameters tested are attached in table 2 to table 7 where the explanation is as follows,

- The pulse given is an input to the arm manipulator in the form of a pulse obtained from the calculation results.

- The resulting angle is the arm manipulator angle that should be obtained from the pulse input.

- The actual angle is the angle generated on the arm manipulator from the test results

- Error is the ratio of the error between the resulting angle and the actual angle.

Measurements in the test are based on a digital arc meter. The digital arc used in this test has a resolution of 0.05º and an accuracy of ±0.03º.

Table 2: Test results at joint 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 1920 | 90 | 89.95 | 0.06 |
| 2 | 960 | 45 | 45 | 0.00 |
| 3 | 640 | -30 | -30.05 | 0.17 |
| 4 | 1280 | -60 | -59.5 | 0.83 |
| Average Error | | | | 0.27 |

Table 3 : Test results at joint 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 1600 | 90 | 90 | 0.00 |
| 2 | 800 | 45 | 45.05 | 0.11 |
| 3 | 533 | -30 | -30.1 | 0.33 |
| 4 | 800 | -45 | -45.05 | 0.11 |
| Average Error | | | | 0.14 |

Table 4 : Test results at joint 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 1000 | 45 | 45 | 0.00 |
| 2 | 555 | 25 | 24.95 | -0.20 |
| 3 | 1333 | -60 | -60.15 | 0.25 |
| 4 | 2000 | -90 | -90.05 | 0.06 |
| Average Error | | | | 0.13 |

Table 5 : Test results at joint 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 1867 | 150 | 150.1 | 0.07 |
| 2 | 1120 | 90 | 90 | 0.00 |
| 3 | 560 | -45 | 45.05 | 0.11 |
| 4 | 747 | -60 | 60.1 | 0.17 |
| Average Error | | | | 0.09 |

Table 6 : Test results at joint 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 860 | 90 | 90 | 0.00 |
| 2 | 430 | 45 | 45.15 | 0.33 |
| 3 | 287 | -30 | 29.95 | -0.17 |
| 4 | 573 | -60 | 60.05 | 0.08 |
| Average Error | | | | 0.15 |

Table 7 : Test results at joint 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Given Pulse | Generated Angle (°) | Actual Angle (°) | *Error*  (%) |
| 1 | 133 | 30 | 30.00 | 0.00 |
| 2 | 266 | 60 | 60.05 | 0.08 |
| 3 | 800 | 180 | 180.00 | 0.00 |
| 4 | 933 | 210 | 210.05 | 0.02 |
| Average Error | | | | 0.03 |

Table 8 : Average Percent Error

|  |  |
| --- | --- |
| Joint | Average Error (%) |
| 1 | 0.27 |
| 2 | 0.14 |
| 3 | 0.13 |
| 4 | 0.09 |
| 5 | 0.15 |
| 6 | 0.16 |
| Average Overall Error | 0.27 |

# 4 conclusion

Arm Manipulator 6 DoF is a robotic arm that has 6 joints and is controlled using a master controller. This Arm Manipulator is moved using Joystick mode. A stepper motor is used as an actuator in this project. The movement of the stepper motor at each joint can be driven simultaneously in one motion. The microstep setting of 1600 pulse/rev is used for each motor driver because it is optimally applied to the 6 DoF arm manipulator. The conversion of degree data into step data utilizes the calculation of the ratio of the number of teeth on the pulley which is used as a parameter. The accuracy of the stepper motor applied to the 6 DoF arm manipulator for the given pulse shows that the stepper motor has an accuracy of 99.86% with an average error of 0.14% for all joints.

**References**

Moore, R., Lopes, J. (1999). Paper templates. In *TEMPLATE’06, 1st International Conference on Template Production*. SCITEPRESS.

Smith, J. (1998). *The book*, The publishing company. London, 2nd edition.

1.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-2)
2.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-3)
3.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-4)