Robot Arm Project

By Greg Flynn, Craig Lombardo and Austin Wiles

# Introduction

> You should have a short segment that introduces microcontrollers and robots and why they are a useful thing to

> study.

Microcontrollers are small computers capable of interacting with electric circuits. The arduino is a revolutionary device

# Purpose of lab

The lab taught us how to program arduino based software using C, a language none of us were very familiar with. Also we learnt the concepts of PWM and how to use it to control a servo with precision. Another aspect of the lab that taught us a lot was the trigonometry for calculating all of the angles. The purpose, or takeaway from this project, was to learn how to actively control multiple changing components at once to complete a task.

# Requirements

The requirements of this lab was to control 5 servos to work together in order to navigate around and move blocks. The goal of this lab was to pick up 4 blocks from pre-specified positions and to stack them all at a single location. This robot is a small scale version of robots that work on assembly plants helping to manufacture new products. In this document we will discuss how we implemented the autonomous movement of the arm and tested it safely.

# High-level Design

The first thing we did was establish our final objective; we wanted to be able to give the robot arm any x, y, z and claw state values and have the robot do that. The robot arm was programmed so we would provide four values, the x,y and z positions as well as the state of the grip. The arm would then convert these values into certain angles for the servos. We used trigonometry to calculate all of the required servo angles. We chose to fix the wrist angle relative to the horizon since we had to fix an angle to get a unique solution. We then stepped the servos until they were in the correct position.

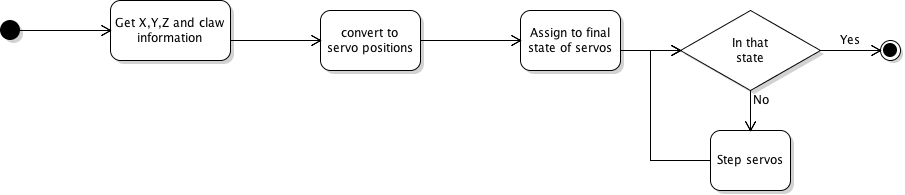


Figure - High level flow

# Detailed Design

## Auto level function

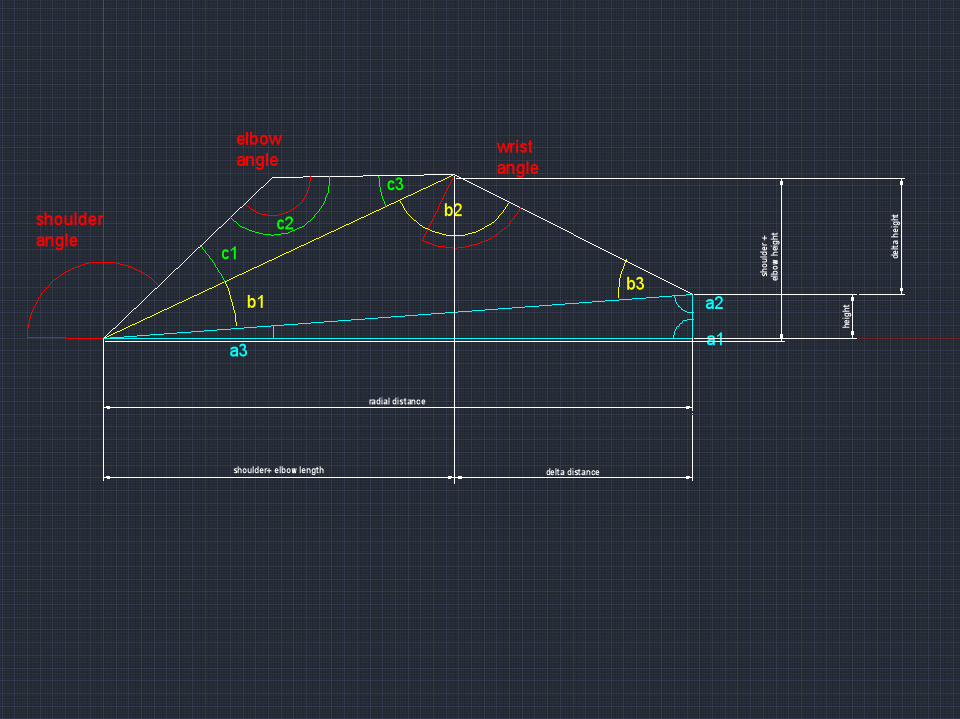
To provide one unique solution we had to fix an angle on the robot arm. The most logical angle to fix was the robot wrist angle. Instead of keeping the angle fixed relative to the previous arm section we kept it flat to the horizon.

## Smoothing function

## Algorithm for motion

We used trigonometry to find the shoulder and elbow angles. (0,0,0) is located at the base of the robot. First we calculated the radial distance with the Pythagorean theorem on the x and y components.

Then we did found the arctan of the x and y to get the base angle.

Then we split the shoulder angle into two parts, one part was from the arctan of the radial distance and the height above or beneath the xy plane. After that we calculated the change in radial distance and height due to the wrist angle. We subtracted this off of the total radial distance and height. This gave us the position that the wrist needed to be placed at. After that we calculated the hypotoneouse of the triangle formed by the corrected radial distance and the corrected height. By using laws of cosines on the triangle the elbow angle could be calculated easily. After that the laws of sines could be applied to find the second half of the shoulder angle. All angles have been solved for.

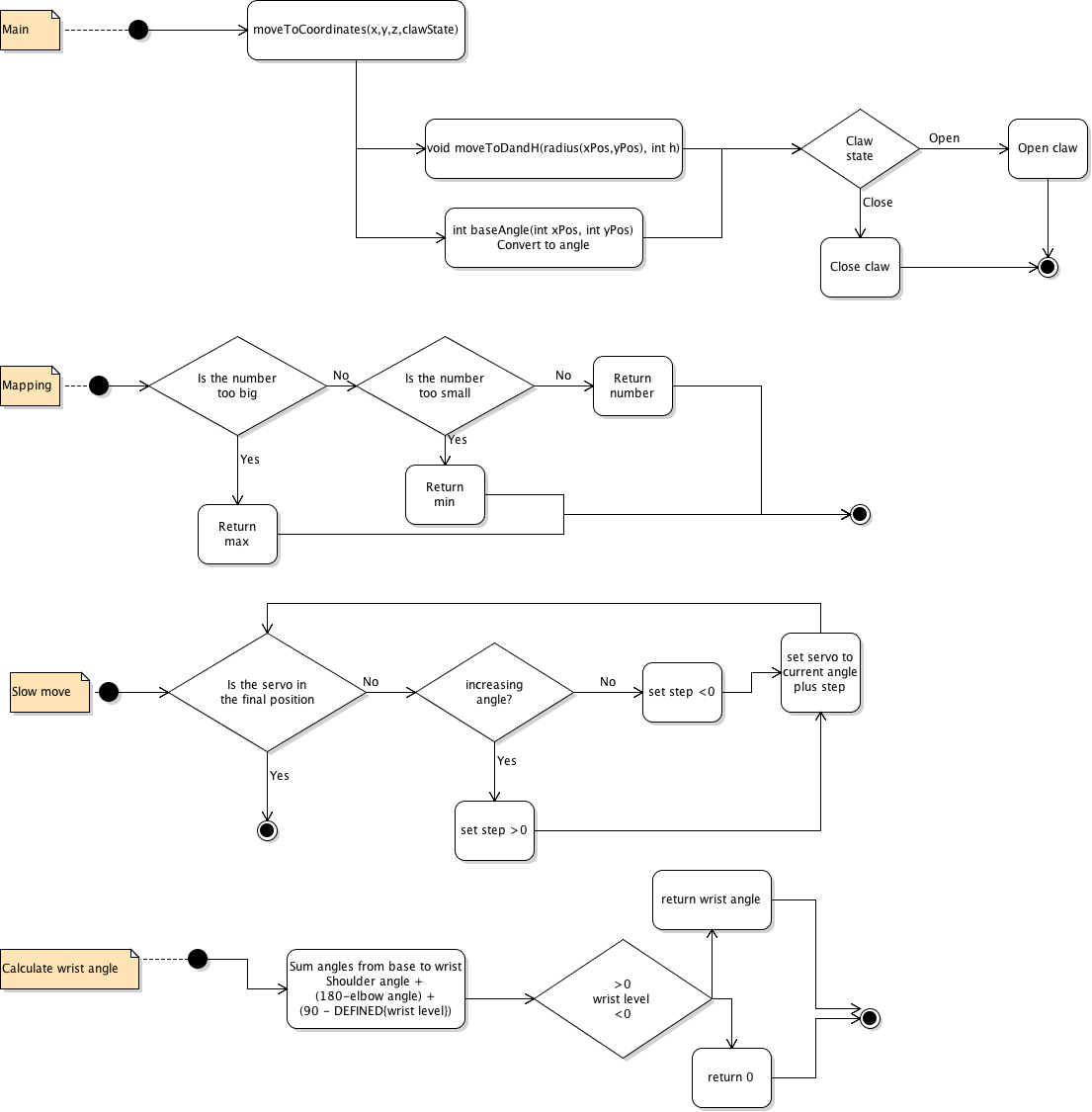
> I would recommend putting the math first in your description. Then you can talk about how having four independent angles

> makes the solution difficult, which points to fixing the wrist angle relative to the ground.

> I would also recommend showing the equations you used, rather than just describing the steps.

# Software Design

Our final objective was to have a function that accepts 4 parameters, x,y and z co-ordinates and a Boolean specifying if the claw is closed. This enabled us to move the arm from any position on the grid to any other. While moving the arm would first move up to ensure that no collision with objects on the grid would happen.



# System Testing

## Testing the leveling function

To test the robot wrist angle we reverted back to the code that controlled the robot arms with servos but removed the wrist from the servo control. This let us quickly test a variety of different angles and visually inspect what was happening.

## Radial angle

To test the base angle function was working we did something similar to the wrist leveling program; we removed control of the base from the potentiometers to the microcontroller. Once that was done we programmed the robot to pick up the blocks. Since the servos that control height were not functional the robot couldn't actually pick up any blocks but it could line up above the blocks. We visually inspected the robot to ensure that the base angle worked. We held a ruler in line with the robot arm to ensure that the line between the claw position and the base went through the coordinates we specified.

## Simulation

We implemented a java based virtual robot arm which accepted three parameters for the angles of the servo. The simulator only showed two dimensions, the radial distance and the height. We felt like that was adequate since we already had solved the base angle. By having a simulator we were able to check our math without causing unnecessary damage to the robot due to it slamming into the ground.

## Final test

Once we proved that each individual angle worked we set up the robot to stack and un-stack the blocks. This let us check not only that the robot worked but that it was consistent. Also we didn't need to supervise it as intensely since it was repeating the same task over and over again. This proved to be a very effective way of fine tuning the robot arm to ensure that it could keep stacking the blocks repeatedly.

# Conclusion

We completed all primary objectives of the assignment; our robot arm was capable of picking up the blocks and stacking them. We also took the project one step further since we have our arm capable of going to any position on the grid with the change of a single parameter. Also we have a simulator of the robot arm which reduces the damage cause on the arm due to poor logic or coding technique.

# Appendix

Introduction: A paragraph or two that explains the purpose of the project, which was to programming a robot arm to stack blocks. State your intended audience for the document, which for this lab would be another student at your level who needs to implement the system. Feel free to explain why microcontrollers and robots are important and get the reader interested. Also, briefly explain what to expect in the remainder of the document.

1. High-Level Design: Describe your design from a high level. Draw a simplified block diagram of the system and explain what you are trying to accomplish. State any requirements or assumptions you had for your design. If there were multiple ways to do the design, explain why you chose your particular approach.

2. Detailed Design: Include one or more sections that describe the internal details of the blocks in the high-level design. For example, if you used trigonometry to position the arm, you could have a section called "Algorithms." Another section could be labeled "Servo Control," etc. The description given in these sections should be sufficiently detailed that someone could understand what you did and implement it themselves.

3. Software Design: Explain how your code works. For example, you could show the basic series of operations with a flow chart. You can also refer to different functions in your code, but I recommend you do not bore the reader with your full code, but rather put it in an appendix.

4. System Testing: Describe the test cases you used, which should be sufficient to convince your reader that the system is working.

5. Conclusion: Give some parting words to reader summarizing what was done in the project and whether it was successful or not. Briefly explain outstanding things that would have to be looked at in the future if you were not completely successful or discovered unexpected things.

6. Appendix: Here you could put a listing of your code or any information you feel is too detailed (or boring) for the main project description.

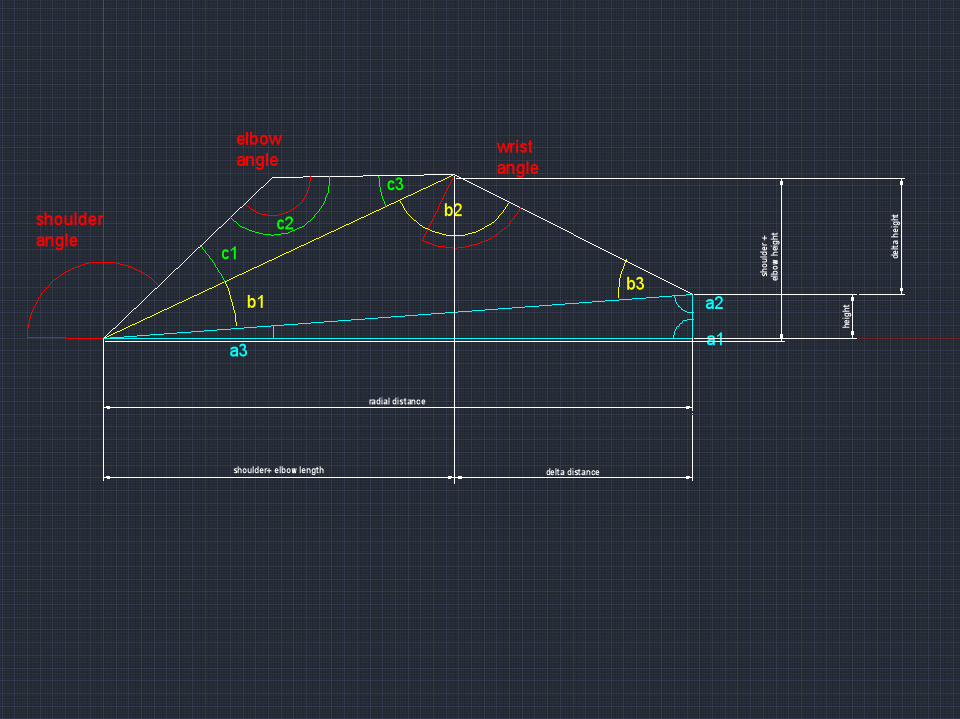


Figure - angle diagram