ISTA 303: Assignment 3
MeArm Robotic Arm and Application

Learning Goal: The third assignment is intended for you to become familiar with projects that move in more than one dimension through a popular educational open source project, the MeArm robotic arm. In the base project you will gain experience with using hand tools and instructions to assemble a medium-sized device, and use trigonometry applied to servo motor control to write the Arduino kinematics code. In the recommended extensions, you will have the opportunity to further hone your skills in several areas, e.g. introductory sensors (e.g. binary IR distance sensor/joystick), rapid prototyping (e.g. designing several platforms at different heights for the arm to pick things up from/place on top of), and the associated Arduino software.

Hint: No single aspect of the assignment is a lot of work, but each aspect takes time, and you should make regular progress between now and the due date to be successful. You should have your arm built and functioning before class day 2, and Kinematics code working by the end of class day 3, to have enough time to work on extensions.

Fall 2022 Special: Historically the base assignment requires assembling the arm and writing working kinematics code to reach a C, then doing extensions from there. For Fall 2022 we'll try an a-la-carte model:

- **Module 1: Kinematics code:** Write working kinematics code and demo this (as described in this assignment and the attached example sheet. Evaluation for this will be based on (a) the arm successfully moving to a number of (X, Y, Theta) coordinates that I give you during demo time.
- **Module 2: Joystick controller:** Create a handheld controller that controls all aspects of the arm. The controller should use a minimum of 2 joysticks, and be well-designed with a 3D printed or laser-cut enclosure. Evaluation for this will be based on the (a) functionality of the controller, and (b) the enclosure embodying strong 2.5D/3D design techniques.
- **Module 3: Make the arm do something cool (automatically):** Past examples of this include getting your arm to play chess, play towers of hanoi, fire a 3D printed catpult (of your own design), assemble lego, assemble the parts to a paper (or, laser cut) airplane, juggle between two arms, sort items coming down a conveyer belt by colour, etc. Evaluation for this will be based on the (a) task having sufficient complexity/automaticity, (b) the extra bits you design (either items that the arm uses, or modifications to the arm itself) are sufficiently complex and emboding strong 2.5D/3D design techniques, (c) it working successfully.

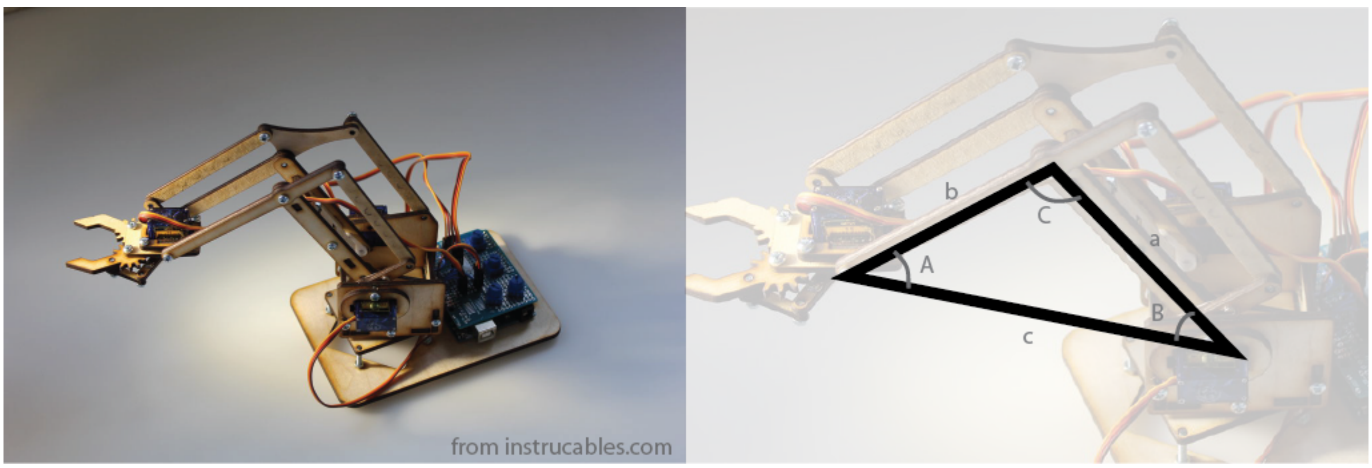
Grading:

- Assembled and working arm + one module from above = up to a C
- Assembled and working arm + any two modules from above = up to an A

Simplifications: I have included an example Arduino sketch that you can use as a scaffold for developing your kinematics code. The kinematics code in the Arduino sketch uses cylindrical coordinates (theta, r, z) instead of cartesian coordinates (x, y, z), further simplifying the kinematics problem.

Task: You will laser cut and assemble the MeArm robotic arm, and write the kinematics code to move it to a specified location in cylindrical (theta, r, z) space. For extensions (i.e. beyond a C), you are asked to make the arm do something interesting/exciting that appeals to your interests (we've discussed several possibilities in class, e.g. picking up and moving a chess piece between several small laser cut pillars).

Kinematics Hints: The MeArm has been designed so that it shouldn't take more than basic trigonometry (triangle/sin/cos) to figure out the kinematics. Try commanding the two relevant servos individually to a few known angles, then together (to known angles), to visualize the triangle and how it's shape is controlled using the servos – then draw a diagram.



Minimum to submit:

1. Picture of assembled MeArm
2. Schematic
3. Code (the write-up should describe the reasoning behind it, ideally including a diagram, and supporting data describing your calibration if you do the kinematics aspect).
4. Kinematics: If you do this aspect, you will have to demo the arm moving to a variety of specific theta, r, z coordinates given at the time of testing (I will choose the coordinates to test at).
5. Include diagrams, schematics, pictures, etc., of any extensions (and demo these as well).
6. **If you worked in a team**, to ensure that the workload was appropriately shared, you must submit a statement of what each team member contributed, and a statement that the workload hours were shared approximately 50%/50%, or reasons for any significant deviations from this.

Grading Team Work: You need only submit a single write-up for team work. **If a team member contributes less than one third of the total effort, their final grade will be substantially penalized**. This penalty will be commensurate with the total contributions of that team member, *but will be no less than 2 letter grades (-20%).*

Where to submit: Submit two files to D2L: A PDF with your writeup, and an Arduino code file with your program. Please do not submit word files, or any file other than a PDF for the writeup.

Code style: The programs that you write should reflect the knowledge and best practices of an undergraduate student who has studied computer science/information science, and include appropriate documentation, structure, functional decomposition, naming conventions, readability, etc.

Due Date: Approximately 2.5 weeks of in-class time (up to Monday October 31st). Project must be demoed no later than after class Monday November 7th (i.e. by 4:45pm), and submitted to D2L that day.

Note on equipment availability: Given that the rapid prototyping equipment is shared, it is expected that you will make good use of the open lab times to print/laser cut your designs.

Grading Reminder (PLEASE READ THIS!): The best way to learn rapid prototyping is to experiment and make a project your own. As such, the structured content of the assignments has been designed to be minimal in order to encourage you to explore your own interests, and give wide latitude in the additional content that you submit.

What the grades mean:
A – Exemplary, far beyond requirements/expectations
B – Exceeds requirements/expectations
C – Meets requirements/expectations
D – Falls short of requirements/expectations
F – Substantially does not meet requirements/expectations

Simply completing the laboratories/assignments will result in a grade of "C". Students must extend the work in one or more directions (additional hardware, software, designs, and/or analyses, etc) in substantial ways, and/or submit exceedingly good documentation/writeups with exceptional presentation to receive higher grades.

A helpful rule of thumb is that the additional hardware/software/designs/analyses should be approximately half again as much intellectual and implementation effort as the main assignment. Please discuss your plans for additions with the instructor if you have any questions.

Safety reminder for building real circuits:

Before energizing (powering on) your circuit, ask the instructor to review your schematic and your physical breadboarded circuit. This review should happen before you energize your circuit each time there is a change in the circuit. This review can happen after class, or during class as appropriate.

Please review the additional safety notes for the Arduino.