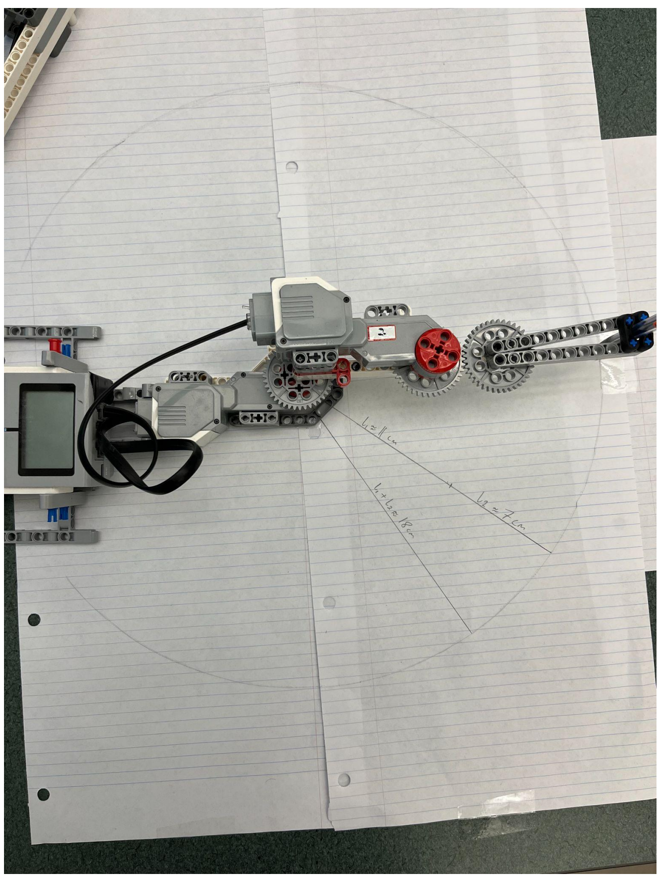
Lab 2 Report

Jasper Eng, Firas Al Chalabi

Introduction

In this lab we created a robot arm that was able to move in 2D space. Its movement was limited to the x, y plane horizontal to the ground. For the design of the arm we tried to keep both of the motors as close to the base as possible, to reduce the load on the first motor. To achieve this we used a gear linkage to connect the second motor to the arm joint. This allowed us to mount both motors close to the base of the arm. We also opted for a shorter arm length so as to further reduce the load placed on each motor. There are also other motor / linkage configurations that could achieve the same movements, an example would be the cable method discussed in class where all the motors sit at the base of the robot, and then you would use cables to move the linkages to certain angles.

Below is a picture of our workspace:



The initial position of the end effector is at the point (18, 0), where the base of the robot is at the point (0, 0). The workspace of our arm is any point within a unit circle that is 36 cm in diameter. It should also be noted that certain regions of the circle do not fall within our workspace. Specifically those that lie too close to the base of the arm where the linkages would either run into each other, or the base itself.

Forward Kinematics

We then implemented forward kinematics into our robot. This allowed us to perform various tasks. The first, and most simple, task allowed us to input 2 angles to our robot, and then have the arm move to the corresponding joint angles, and return the (x,y) position of the end effector. We were able to be accurate to the expected (x, y) coordinates to ~90%, being off by about 1-2 centimeters. The second operation that our robot was able to perform was for us to move the end effector to two different points in the workspace, and then for the robot to tell us the euclidean distance between those two points. The error incurred during this task was, in large part, due to gear backlashing in both the motors, as well as the gear linkages previously mentioned. This caused a slight difference between the joint angles that the robot thought it was at, versus the ground truth. The last operation we were able to achieve using forward kinematics was for the user to move the end effector to three different points in the workspace, and then for our robot to return the angle that was made by the intersection of the two lines that were created. The error in this operation is the same as for the distance task.

Accuracy:

	(theta1°, theta2°)		
	(90°, 0°)	(-45°, -45°)	(115°, -25°)
Expected (x, y)	(0, 18)	(7.8, -14.8)	(-4.6, 17.0)
Measurement 1	(2.5, 17.8)	(8.7, -15.7)	(-3.1, 17.3)
Measurement 2	(2.6, 17.7)	(9.4, -15.2)	(-3.5, 17.1)
Measurement 3	(2.5, 17.8)	(10, -15.0)	(-4.5, 16.9)
Root Mean Squared Error (1 d.p.)	2.54 ~ 2.5 cm	1.75 ~ 1.8 cm	1.09 ~ 1.1 cm
Accuracy	1 - (2.5/|(0, 18)|) = 85.9%	1 - (1.8/|(7.8, -14.8)|) = 89.2%	1 - (1.1/|(-4.6, 17.0)|) = 93.8%

Inverse Kinematics

We then went on to utilize the inverse kinematics of our robot to have it be able to move the end effector to a given point in space. We first implemented two different methods of inverse kinematics; numerically using Newton's method to find the joint angles that achieve the goal pose, and analytically finding the joint angles using the equations from the lecture notes. Once we had a way to get the end effector to a point in the workspace through the movement of our joints we could perform two different operations. The first task was simply to input an (x, y) coordinate to our robot and have it move to that position. This was simple once we had implemented the inverse kinematics methods outlined above, we just moved the motors to the angles returned by our inverse kinematics calculations. Though it should be noted that while for the analytical method this is trivial, for the numerical solution if the destination position is too far from the starting position Newton's method may fail to converge. To solve this issue, we broke up the vector that represents the linear motion from the initial position to the goal into smaller intervals, and then iteratively calculated the joint angles to get from one interval pose to the next. This allowed us to make the calculations over a much smaller distance (in our case one tenth of the total distance), allowing Newton's method to converge. The final task that we implemented on our robot was to have the user move the end effector to two points in the workspace, and then have the robot move the end effector to the position between these two points. This was simply a combination of the distance calculation task outlined in the forward kinematics section, and the movement task outlined above. Once we had calculated the distance between the two points we vectorized the line between them, and then using the midpoint method we calculated the (x, y) coordinates of the midpoint and fed this position into the movement task.

Analytical vs. Numerical Inverse Kinematics

There are advantages and disadvantages to both methods of calculating the inverse kinematics of an armed robot. With the numerical solution you are able to overcome errors in the length of your linkage calculations in the Jacobian, whereas with the analytical solution, a relatively small change in linkage lengths can cause high error. In general, it is very hard to find analytical solutions for complex manipulators, so it is far simpler and more preferable to use the numerical solution. However, in this case where we only have 2 linkages, it is much easier to use the analytical solution as the formulas are relatively straightforward. Also, the analytical solution is more accurate than the numerical solution, however we can set our convergence threshold to be very low so the difference in accuracy can be overcome practically. The numerical solution can diverge too, so we have to take that into account and specially deal with singularity points in the Jacobian.