MEGN 441
Christian Prather

Lab 2 Open Loop Control

Problem Statement:

A robotic platform that is built to operate in the real world is only as good as its ability to sense the real word. Variations in terrain and conditions often lead to discrepancies in what a bot is programmed to do and what it actually does. This lab aims to discover how optimal an open loop robot can perform in real world conditions.

Methods:

a)

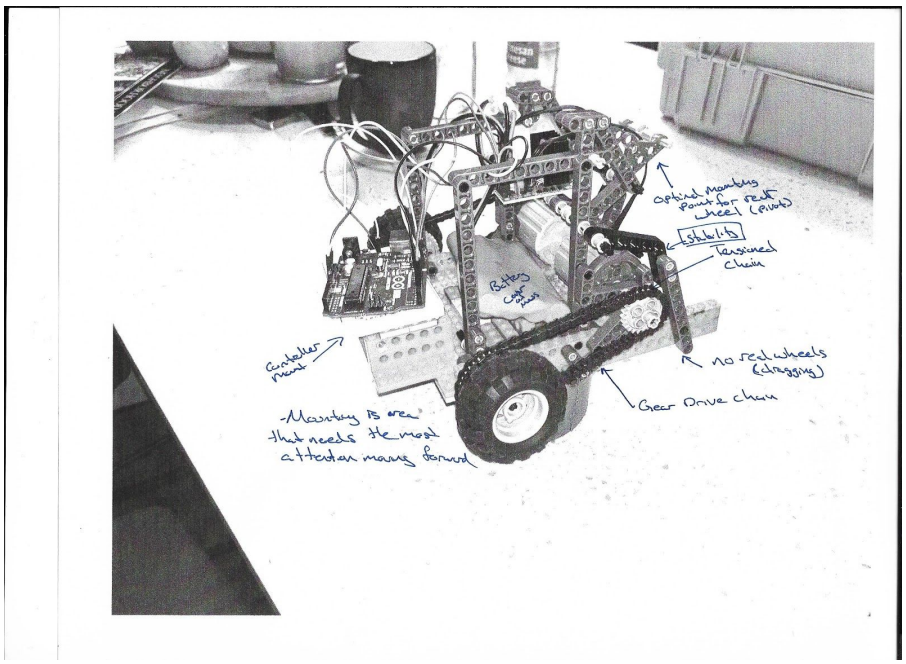


Fig 1 (Photo of Robot)

b)

There were many considerations that had to be taken into account when choosing a mechanical configuration. I focused my attention on exploring two primary key sections of my robot to explore different options: drive method and rear wheel layout. I knew from the results of lab 1 that the Lego gears do not offer great resilience from slipping unless well built, as I was not confident in my ability to design a well constructed Lego gearbox I choose to use a chain as contrary to common belief when done in a chain sprocket configuration and properly tensioned are far less susceptible to slipping than direct drive connections. This did present a question as to if I would need a method of dynamically tensioning the chain. I explored the ability to simply use an idler sprocket to wrap around but as it was not spring tensioned this had no more dynamic tensioning effect than the direct two gear configurations have. I then looked at, and constructed a spring tension system utilizing a shock found in the lego sets. I was able to construct a test that worked in applying the proper force for tension but I could not find an

appropriate mounting method for the shock and soon scrapped the idea, opting for a careful static tensioning. This is likely a great cause in my variance. I spent a significant amount of time looking at my rear wheel configurations. I knew that I wanted to do a dragged third wheel that was free to pivot and was not driven. I constructed such a mechanism however once again mounting to my base became problematic and after about an hour I chose not to spend more time on it. I then explored a simple 2 free wheel configuration that would behave well when driving linearly but as the friction was too great when turning I realized that I would be better off minimizing friction and doing a simple drag system, which is what I used for the remainder of this lab. In summary I had the majority of my issues with using Legos as the platform, I was not able to construct a sufficient base to give me the freedom I needed in the mechanisms I wanted to attach and in future labs this may require adjusting.

c)

For testing I first had to go through some experimentation to understand the dynamics of my platform. I began by utilizing the inintal_testing code provided with some minor adjustments to get my motor calibrated for driving in a linear path on my given floor. I knew from Lab 1 my motors have different characteristics and that a single PWM value for both would not be sufficient. I quickly found that motor A needed to be held back from motor B by about 10%. I then used a simple trial method of finding how long it took my robot to travel 50 cm by adjusting a delay driven program. Changing the delay accordingly to over and under shoot. Once these values were set I was able to use them for the primary function code. Testing then became straight forward, I began by selecting a single measurement point on my robot. This was the center front and proved to be an easy repeatable measurement point. I then selected a starting point to base all future measurements on the track with. I chose the back line of the tape used to account for the thickness of the tape. I then let the robot run one section at a time including its respective turn and measured the resulting point from the new relative starting point on the track section. I then measured drift relative to x, y coordinates that are relative to the bot as in positive y is forward and positive x is right.

d)

Most of the code was provided and used as is, however I did do some simple adjustments for convenience and to work with my motor configurations. I set up two turning functions to allow for simple calls when running. The run time for both linear and turning was calculated from a given distance and travel time found in the calibration step. The primary flow of the code is that it waits for button input to begin running the next step in an array representation of the path. If it's linear then delay time is calculated and the motor run code is executed. If a turn it is determined whether it's a left or right turn. This primary function then calls its respective turn function. This is then repeated for each step on the path delaying in execution until user interaction through the push button.

Results:

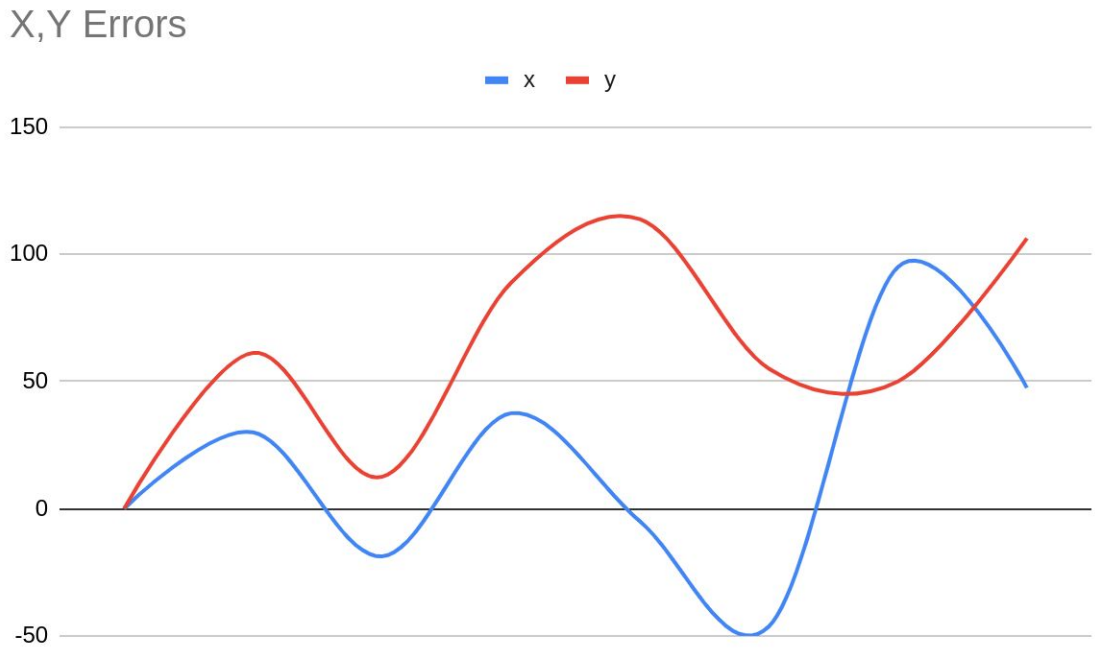
a)

Average Error

Average Error			
X mm	Y mm	% X	% Y
0	0	0	0
30	61.25	3.333333333	6.805555556
-18.75	12.5	-3.125	2.083333333
37.5	88.75	2.083333333	4.930555556
-5	113.75	-0.5	11.375
-46.25	55	-7.708333333	9.166666667
95	50	9.5	5
47.5	106.25	3.166666667	7.083333333

Standard Deviation

SD:	
x	y
18.70828693	7.071067812
18.70828693	25.58686186
24.07670036	25.86020108
43.80353867	73.34635301
43.87482194	59.51627929
66.08469944	66.8954408
18.02775638	47.4341649
29.47456531	103.6445247



Averages average			Averaged SD	
x	y		x	y
17.5	60.9375		44.16385885	41.11042403

Conclusion:

a)

The results indicate that there is a problem with consistency across time of the performance of my robot. I can see from the average errors that while my platform is capable of doing fairly consistent in run per run basis however it is not an accurate robot. Elements that contributed to the error are an inconsistent floor surface, my living room while wood is not even and this definitely resulted in enough change in terrain to affect precision. Other sources of error include the tilt of my drive wheels as their axis proved to not be supported well enough to handle the chain tension. Locations that produced the largest error in terms of percentage of distance traveled was from point 5-6, this area seems to have the worst ability to stay straight and travel the appropriate distance out of all sections.

b)

The positives of the robot design are in its chain drive as this proved to be very low slip, so much so that I need to now be careful about stalling the motor under load as there is no way to relieve the force. There are also benefits in its openness as a platform with minimalist components I have a great deal of space for mounting. The code is also very strong having been mostly provided for us. Problems with the bot however include the rigidity of the drive axis, and the rear wheel drives, both vary too much during drive and offer harder conditions for repeatability in testing. I had a significantly harder time in construction of this robot using Legos than I would have simply modeling and printing it. I am not a fan of the limitations it has imposed that are not indicative of my robotics ability but rather of my ability to use a toy.

c)

The advantages of open loop control are that it is simple, there is no need for advanced sensors or complexities in code such as real time calculations or sensor data fusing.

Reference:

N/A

Appendix:

See attached pages