Group 19 - Lab 1 Report

Anthony Johansen (260712168) & Bilal Yousuf (260680182)

ECSE 211: Design Principles & Methods
Winter 2018

Section 1: Design Evaluation

For Lab1, we began by constructing the Bang-Bang controller first. Before constructing the controller, we brainstormed what possible commands the robot would have to execute and came up with 3 primary commands:

1. A command instructing the robot to continue moving in a straight line when it is within the bandwidth
2. A command instructing the robot to begin turning to the right when the measured distance falls below the bandcenter
3. A command instructing the robot to turn left when the measure distance falls above the bandcenter. Following some testing we decided to implement a safety feature allowing the robot to reverse itself and turn on its axis when the measured distance falls below 10. We also included the filter from the P-type Controller class which allowed the robot to ignore gaps in the wall and clear some distance from the end of the wall before performing 180 degree turns.

The P-type Controller had a very similar thought process however we also introduced a new variable to keep count of the error of the robots path. The error was calculated by taking the recorded distance and subtracting it by our bandcenter. This value was then multiplied by our P-constant and either added or subtracted to the default motor speed of 200 (for inside turns we chose to lower the speed of the outer wheel, for outside turns we increased the speed of the outer wheel). Initially, this produced very jerky movements however after testing with different filter values and P-constants the robot was stable. We also introduced an upper and lower bound for the error to prevent extremely jerky movements with very high distances, and also to prevent the motor from crashing. When the motorspeed was too high, the motor would not engage and would remain frozen for the duration of the test, and when the motor speed went below 0 the motor would also remain frozen. It is possible to include code to restart the motors however we found it more simple to simply include bounds. We also included a similar safety feature which allowed the robot to back up during sharp interior turns.

Before beginning our hardware design we decided to observe other groups designs to see if there were any major problems with the classic 2 wheeled version. One of the main issues we observed was that at times the 2 wheeled version did not provide much traction, especially with low clearance bots whose parts could sometimes touch the grounds. 2 wheeled models also had difficult reversing straight backwards (for models with a rear swivel wheel).

As a result, we decided to implement a tread style propulsion system for our bot. This simple system allowed to use a small number of parts to move the bot and also gave us a greater range of maneuverability on the track. Our robot is able to turn 360 degrees in place and move straight forwards and backwards with ease. Our initial frame was not very strong however we were able to install improve the strength of the frame using crossbars which reduced sagging and gave the motors a higher clearance off the ground. One of the main issues we faced was that our wheelers had to extend outwards a bit to clear the motors, initially we had problems with the wheels falling off however we were able to secure them using caps at the end of the X bars.

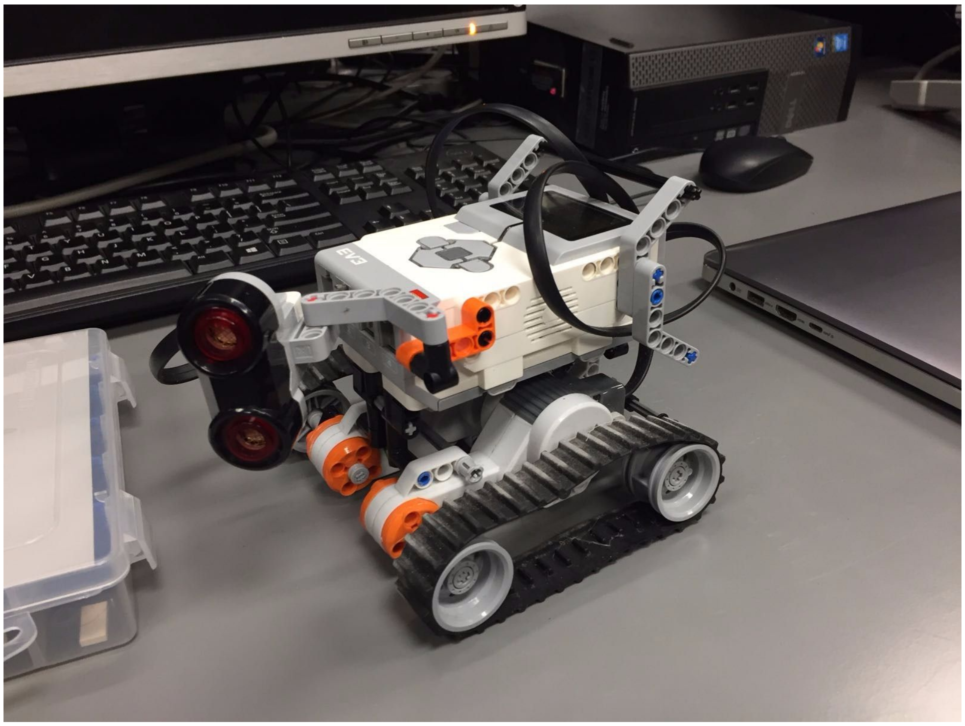

Fig. 1.1: Side profile of our robot “D-Tank”

Section 2: Test Data

For each controller, we performed trials to collect test data. This allowed us to analyze our data and make our robot as efficient as possible. The test data is presented below in Tables 2.1 & 2.2.

Trial No.	Touches Wall	Band Center (cm)	Minimum Distance from Band Center (cm)	Maximum Distance from Band Center (cm)
1	No	28	0	9
2	No	28	2	20
3	No	28	0	34

Table 2.1: Bang Bang Controller

Trial No.	Touches Wall	Band Center (cm)	Minimum Distance from Band Center (cm)	Maximum Distance from Band Center (cm)
1	No	28	12	32
2	Yes	28	11	33
3	No	28	6	37

Table 2.2: P-Type Controller

An important consideration for the design of the P-type controller was selecting an appropriate constant of proportionality. The following table depicts test values for our selected constant, and for constants above and below this number.

	Constant Value	Touches Wall	Band Center (cm)	Minimum Distance from Band Center (cm)	Maximum Distance from Band Center (cm)
Below	10	Yes	28	0	40
Demo Constant	13	No	28	11	33
Above	16	No	28	0	70

Table 2.3: P-Type Controller Constant

Section 3: Test Analysis

The following questions have been answered based off the above test data and other observations made during the lab.

What happens when your P-type constant is different from the one used in the demo?
For a p-type constant of value 10 (below our demo constant) our robot touched the wall while performing the 180 degree turn. The low constant value resulted in the robot being unable to perform a wide enough turn.
When the constant was of value 16 (higher than our demo constant), the robot was able to complete a lap successfully. The movement of the robot, however, was haphazard. It adjusted itself with sudden jerks every time it tried to straighten its path of movement.

How much does your robot oscillate around the band center?
The minimum distance from the band center for the Bang-Bang Controller is 0 cm. This indicates that during these moments our robot was perfectly on track. The maximum distance was 34 cm.
For the P-type Controller, the minimum and maximum distances were 6 cm and 37 cm respectively. These values are on average higher than those for the Bang-Bang Controller.

Did it ever exceed the bandwidth? If so, by how much?
At times during a lap, our robot exceeded the bandwidth. This usually occurred when performing the outside U-turns. It would exceed the bandwidth by more than 20 for these turns. While following the wall on the “insides” of the track the maximum it exceeded the bandwidth was by 10.

Describe how this occurs qualitatively for each controller?

For the Bang-Bang Controller this occurs due to the two set speeds at which each wheel is allowed to move. This can be better understood as an ON/OFF mechanism. The result is very wide turns (much wider than p-type).

For the P-Type Controller this takes place because the speed of the wheels change dynamically with respect to the sensor readings. This implies that the further the robot is from the wall the faster its wheels spin to bring it back on course. Similarly, smaller deviations from the band center are adjusted with lesser wheel power.

Section 4: Observations and Conclusion

Based on your analysis, which controller would you use and why?

Our preferred controller is the P-type controller. This is because it corrects its path more frequently and accurately. Overall, it was also the faster of the two controllers, particularly during the 180 degree turns.

Does the ultrasonic sensor produce false positives and/or false negatives? How frequent were they? Were they filtered?

For our earlier test drives, we encountered false negatives. While travelling straight, the sensor was unable to identify a brick placed by its side and, consequently, would turn into it. No false positives were detected. This occurred very frequently during the early stages of development, while we were still fine-tuning our code. With the help of the filter algorithm we were able to overcome this issue.

Section 5: Further Improvements

What software improvements could you make to address the ultrasonic sensor errors? Give 3 examples.

- *Improving the filter algorithm to sift through "bad values" in a more precise fashion. This could also reduce false positive and false negative readings.*
- *Implementing machine learning, thereby allowing the robot to adapt to collisions during a lap. This prevents the same mistakes from being made twice.*
- *By increasing the sensor poll rate the robot can take in more information and make a more accurate estimate in regards to the distance polled*

What hardware improvements could you make to improve the controller performance? Give 3 examples.

· *Multiple sensors*

· *Rotating sensor*

· *Sturdier support for the wheelbase. This would prevent the robot from falling apart if it fell and ensure perfectly even wheels. We ran out of the Lego parts required to improve the base.*

· *Smaller motors that can be repositioned and replaced to allow for an alternate, and more efficient, wheel placement.*

What other controller types could be used in place of the Bang-Bang or P-type?

Alternatively, a PID controller could have been used. PID controllers are an amalgamation of proportional, derivative, and integral controllers and thus afford the user incredible flexibility and precision.