CS26020: Experimenting with Sensors

Nathan Williams - naw21

March 5, 2018

Contents

List of Figures

1 INTRODUCTION

1.1 Purpose of this Document

This document shows and discusses the result of experimenting with the IRSensors and Encoders on the Formula AllCode robots[1] and the in-robot API[2]

1.2 Objectives

- Collect Infrared Sensor data.
- Collect Encoders data
- Visualize the data.
- Analyze and discuss the data.

2 Infrared Sensors

2.1 Programming

To get the Sensor reading from the robot I wrote a function to display the live data on screen with some basic averaging to get rid of noise.

```
void IRSensors(){
 double IRDataSum[8] = {0,0,0,0,0,0,0,0};
 int j;
 for(j =0;j<10;j++){
  int i;
  for(i=0;i<8;i++){
        IRDataSum[i]=IRDataSum[i]+FA_ReadIR(i);
  }
 }
 double IRDataAverage[8];
 int i;
 for(i=0;i<8;i++){
  IRDataAverage[i]=IRDataSum[i]/10.0;
  if (IRDataAverage[i]>600) {
   FA_LEDOn(i);
  } else {
   FA_LEDOff(i);
  }
 }
 FA_LCDClear();
 FA_LCDNumber(IRDataAverage[IR_RIGHT],0,12,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_REAR_RIGHT],0,1,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_REAR], 40,1,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_REAR_LEFT],80,1 ,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_LEFT],80,12,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_FRONT_LEFT],80,20,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_FRONT],40,20,FONT_NORMAL,LCD_OPAQUE);
 FA_LCDNumber(IRDataAverage[IR_FRONT_RIGHT],0,20,FONT_NORMAL,LCD_OPAQUE);
 FA_DelayMillis(100);
```

```
}
```

2.2 Data Acquistion

To acquire the data for the IR sensors I chose a range of suitable scenarios to cover each of the categories:

2.2.1 range of distances

For this I collect a set of data across a wide range of values for 5mm to 125mm for one sensor to give me an idea of the range of useful distances that can be used.

2.2.2 Repeatability & Reliability

For this I collected a set of 20 readings at 20mm, 50mm and used light and dark backgrounds for one sensor to see how repeatable and reliable the data values are.

2.2.3 Similarity

For this I chose to take 20 readings from 4 different sensors at a distance of 20mm to see how they would compare to each other.

2.3 Data Captured

2.3.1 range of distances

	A	B	C	E	G	I	J
1	MM		Readings 1	Readings 2	Readings 3		Average
2	5		3835	3835	3845		3838.33
3	10		3766	3797	3820		3794.33
4	15		2550	2794	2947		2763.67
5	20		1750	1824	1980		1851.33
6	25		1200	1249	1444		1297.67
7	30		845	917	1053		938.333
8	35		656	677	779		704
9	40		488	530	580		532.667
10	45		394	414	471		426.333
11	50		314	327	367		336
12	55		261	269	315		281.667
13	60		214	221	260		231.667
14	65		180	187	217		194.667
15	70		150	155	174		159.667
16	75		126	129	156		137
17	80		103	108	126		112.333
18	85		89	91	105		95
19	90		72	76	90		79.3333
20	95		60	63	72		65
21	100		51	52	56		53
22	105		42	43	53		46
23	110		34	34	45		37.6667
24	115		30	31	39		33.3333
25	120		18	24	25		22.3333
26	125		16	20	22		19.3333

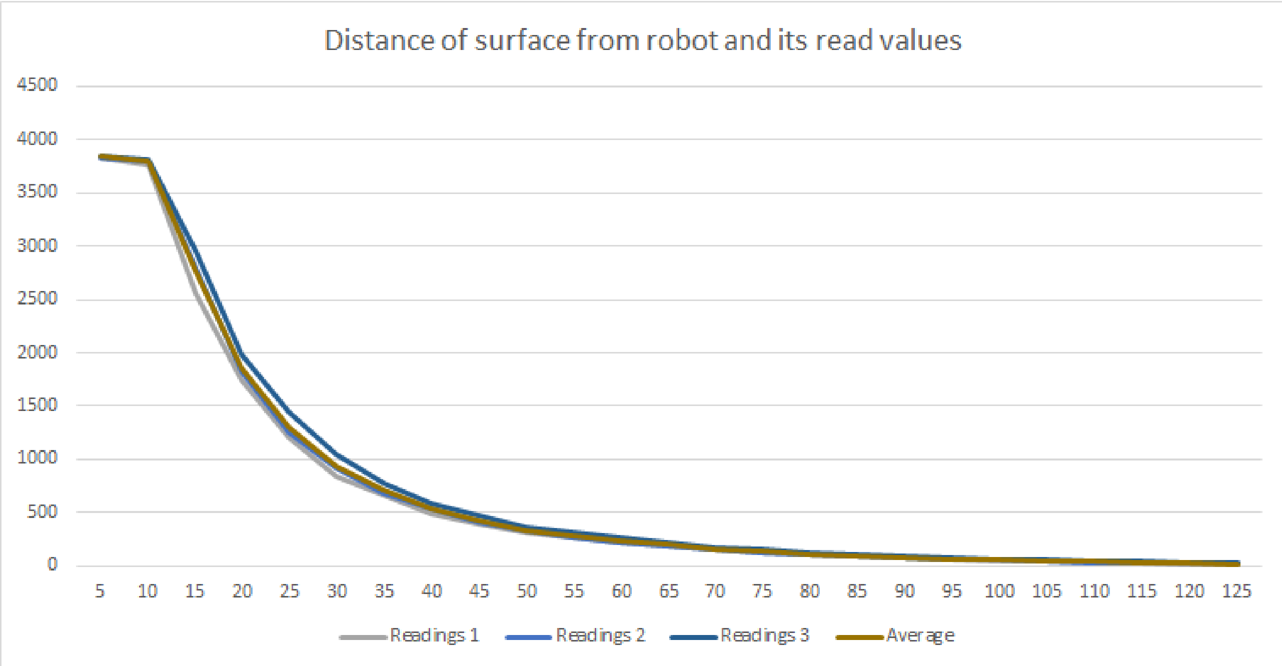
2.3.2 Repeatability & Reliability

	A	B	C	D	E
1		50mm	20mm	50mm	20mm
2		white	white	black	black
3	1	277	1394	30	340
4	2	276	1391	29	341
5	3	278	1393	27	339
6	4	276	1394	31	342
7	5	277	1392	27	345
8	6	279	1391	29	341
9	7	278	1390	28	345
10	8	275	1392	28	343
11	9	277	1390	29	344
12	10	278	1389	30	342
13	11	277	1390	30	346
14	12	278	1388	31	345
15	13	275	1389	32	345
16	14	276	1387	28	345
17	15	276	1390	28	346
18	16	276	1387	30	342
19	17	277	1392	32	340
20	18	278	1394	31	344
21	19	276	1389	30	346
22	20	275	1392	29	345

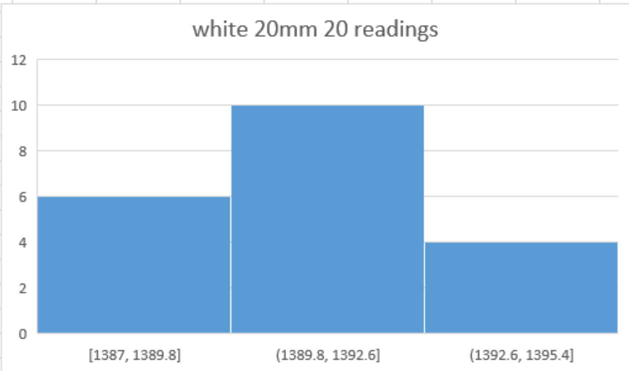
2.3.3 Similarity

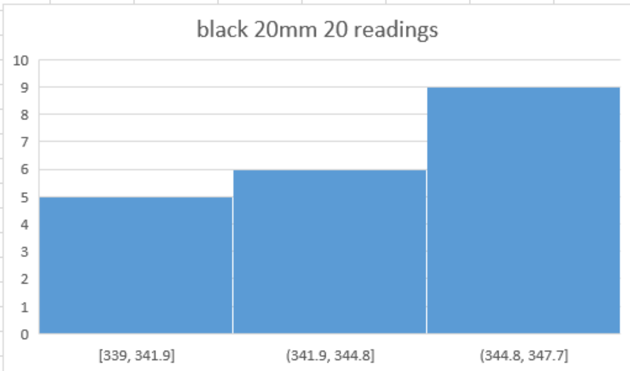
2.4 Data Visualised

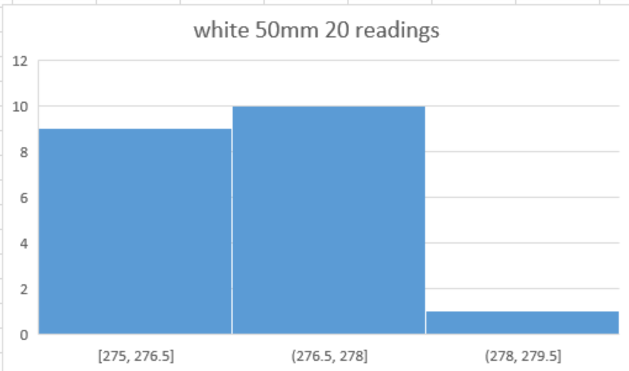
2.4.1 range of distances

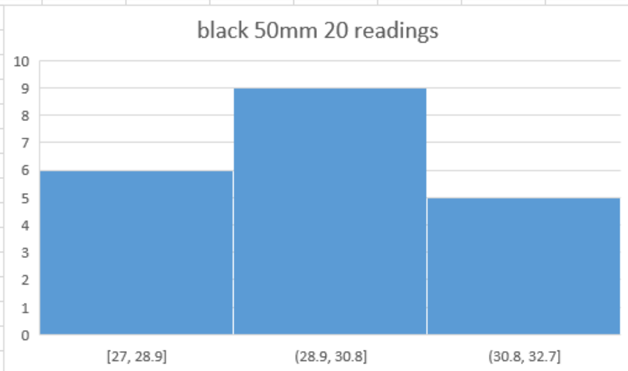


2.4.2 Repeatability & Reliability









2.4.3 Similarity

2.5 Discussion

2.5.1 range of distances

From the data we can determine the range of useful distances are from 10mm to around 120mm where the difference between the sensor values for different distances is not significant.

We can also determine a function for the relationship between the sensor reading and actual distance in mm. By logging the sensor reading we end up with a reasonably straight line to fit the y = mx + c model which allows us to calculate a gradient and a y intercept.

	A	I	J	K	L	N	O	R	T	U	V
1	MM		Average	LOG of Average	LOG of LOG		log(Average)*m +c	difference from actual value		log(log(Average))*m+c	difference from actual value
2	5		3838.333	3.584142688	0.554385291		0.710669271	4.289330729		9.12043249	-4.12043249
3	10		3794.333	3.579135481	0.553778138		0.973338604	9.026661396		9.289924745	0.710075255
4	15		2763.667	3.44148566	0.536745964		8.19420855	6.80579145		14.04461115	0.955388855
5	20		1851.333	3.267484621	0.514213552		17.32200016	2.67799984		20.3347392	-0.334739198
6	25		1297.667	3.113163149	0.493201881		25.41743589	-0.417435891		26.20033854	-1.200338535
7	30		938.3333	2.972357144	0.473100991		32.80387375	-2.803873746		31.81168502	-1.811685021
8	35		704	2.847572659	0.454474814		39.3498506	-4.349850597		37.01135183	-2.011351831
9	40		532.6667	2.72645552	0.435598417		45.70344484	-5.703444838		42.28087008	-2.280870083
10	45		426.3333	2.62974929	0.419914346		50.77648533	-5.776485325		46.65922118	-1.659221176
11	50		336	2.526339277	0.402491674		56.20119454	-6.201194538		51.52291881	-1.52291881
12	55		281.6667	2.449735454	0.389119188		60.21969773	-5.219697734		55.2559702	-0.255970199
13	60		231.6667	2.36486355	0.373806087		64.67193007	-4.671930075		59.53076156	0.469238439
14	65		194.6667	2.289291592	0.359701113		68.63630339	-3.636303393		63.46829354	1.531706455
15	70		159.6667	2.203214259	0.343056734		73.15177046	-3.151770459		68.11472371	1.885276287
16	75		137	2.136720567	0.32974773		76.63991375	-1.639913752		71.83005316	3.169946844
17	80		112.3333	2.050508646	0.311861605		81.16244103	-1.162441033		76.82312797	3.176872034
18	85		95	1.977723605	0.296165597		84.98061756	0.019382442		81.20481146	3.79518854
19	90		79.33333	1.899455702	0.27862917		89.08641548	0.913584517		86.10026486	3.899735137
20	95		65	1.812913357	0.258377049		93.62627627	1.373723732		91.75382889	3.246171115
21	100		53	1.72427587	0.236606751		98.27604452	1.72395548		97.83120586	2.16879414
22	105		46	1.662757832	0.220829002		101.5031737	3.496826313		102.235708	2.764291985
23	110		37.66667	1.575957189	0.197544416		106.0565843	3.943415705		108.7358124	1.264187638
24	115		33.33333	1.522878745	0.182665325		108.840987	6.159012969		112.8894459	2.110554084
25	120		22.33333	1.348953548	0.129996995		117.9648001	2.035199922		127.59229	-7.592290008
26	125		19.33333	1.286306739	0.109344545		121.2511426	3.748857383		133.3576095	-8.357609457

From the data we can see that the distance from the actual value is less for the log of log and from this we can determine that the relationship.

Obstacle distance in mm = log(log(sensor value))*-M+C

For this set of data the gradient is -279 and the intercept is 163 which used a white obstacle. The experiment was re done with a black obstacle and gradient came out at -198 and intercept at 110. As these are both the extreme cases of objects reflection and light absorption, an average was take to determine the distance more reliably at a gradient of -238 and intercept of 136.

2.5.2 Repeatability & Reliability

	50mm white	20mm white	50mm black	20mm black
max	279	1394	32	346
min	275	1387	27	339
range	4	7	5	7
Standard Deviation	1.16416	2.17885	1.5035	2.25015
median	277	1390.5	29.5	344

2.5.3 Similarity

3 Encoders

3.1 Programming

```
void encodersDataGathering(int power){
 FA_SetMotors(power,power);
 FA_DelayMillis(500 + power*10);
 int i;
 for(i = 0;i<5;i++){
  FA_ResetEncoders();
  FA_DelayMillis(1000);
  FA_LCDNumber(FA_ReadEncoder(1), 25*i,
    1, FONT_NORMAL, LCD_OPAQUE);
  FA_LCDNumber(FA_ReadEncoder(0), 25*i,
    12, FONT_NORMAL, LCD_OPAQUE);
 }
 FA_SetMotors(0,0);
}

int main(){
 FA_RobotInit();
 FA_LCDBacklight(50);
 int power = 0;

 while(1){
  if(FA_ReadSwitch(0)){
   FA_LCDClear();
   encodersDataGathering(power);
  }
  if(FA_ReadSwitch(1)){
   FA_LCDNumber(power, 0,
    24, FONT_NORMAL, LCD_OPAQUE);
   power = power + 10;
```

```
    FA_DelayMillis(200);
  }
 }
}
```

3.2 Data Acquistion

3.2.1 Range of power

To see the range of useful encoder powers I took a range of measurements from 0 to 150 to see where the graph started and levelled out.

3.2.2 Repeatability & Similarity

I ran the experiment at powers of 20 and 40 multiple times to see how consistent and repeatable running the robot at certain powers were.

3.3 Data Captured

3.3.1 Range of power

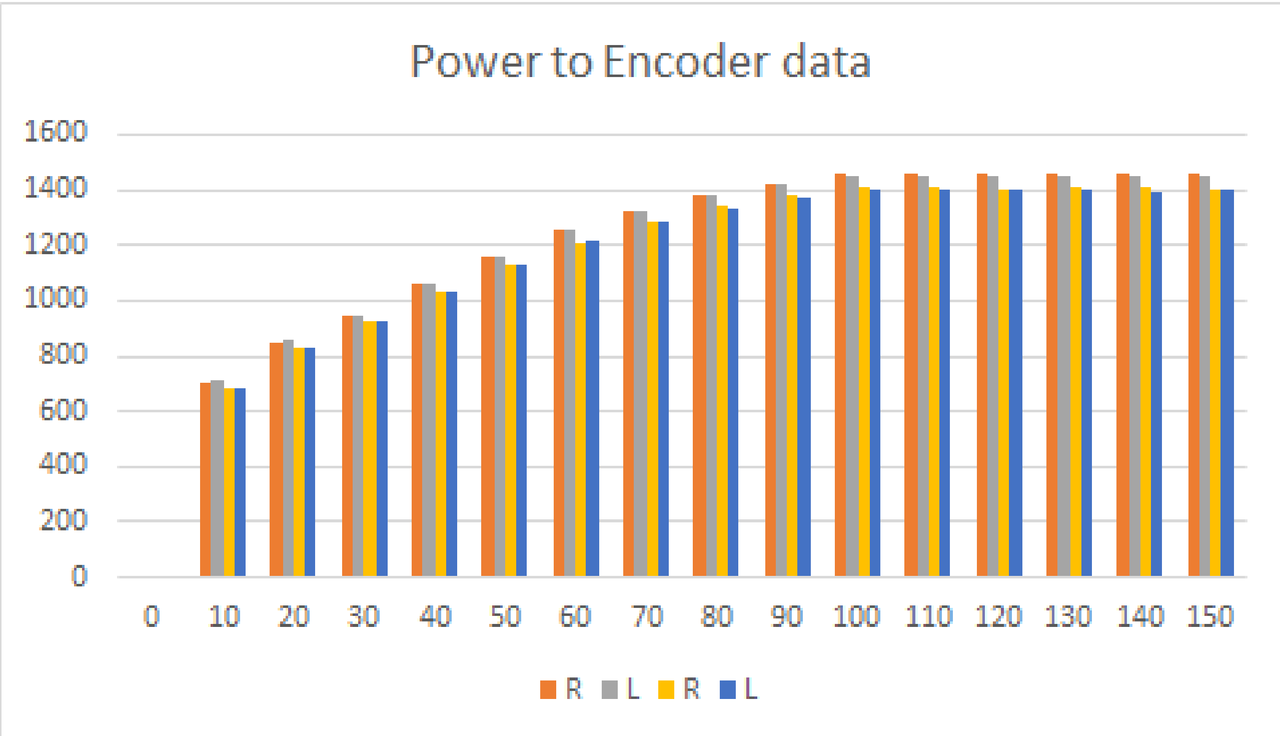
	A	B	C	D	E
1	Power	R	L	R	L
2	0	0	0	0	0
3	10	699	709	682	684
4	20	851	854	825	826
5	30	950	950	922	924
6	40	1061	1066	1029	1034
7	50	1158	1163	1126	1130
8	60	1254	1254	1211	1212
9	70	1320	1326	1288	1280
10	80	1386	1378	1346	1332
11	90	1425	1418	1380	1371
12	100	1457	1448	1406	1402
13	110	1460	1446	1407	1401
14	120	1460	1451	1405	1398
15	130	1462	1452	1408	1400
16	140	1459	1452	1406	1394
17	150	1461	1452	1404	1398

3.3.2 Repeatability & Similarity

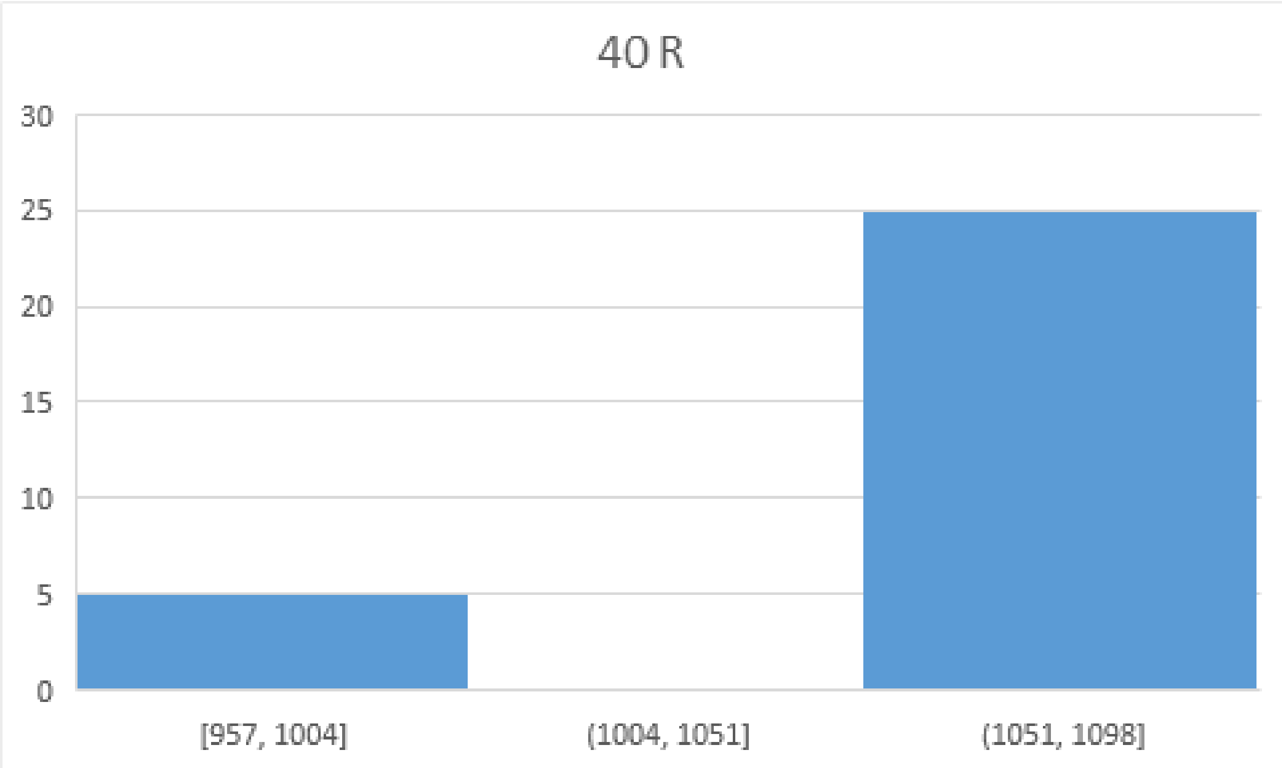
40R	20L
957	792
958	793
958	794
959	796
960	797
1063	796
1067	798
1067	800
1068	802
1070	803
1065	798
1068	802
1068	802
1069	801
1071	802
1069	798
1068	800
1069	800
1070	800
1072	801
1066	799
1066	800
1068	800
1069	800
1071	800
1066	800
1067	801
1069	802
1071	802
1072	802
	793
	799

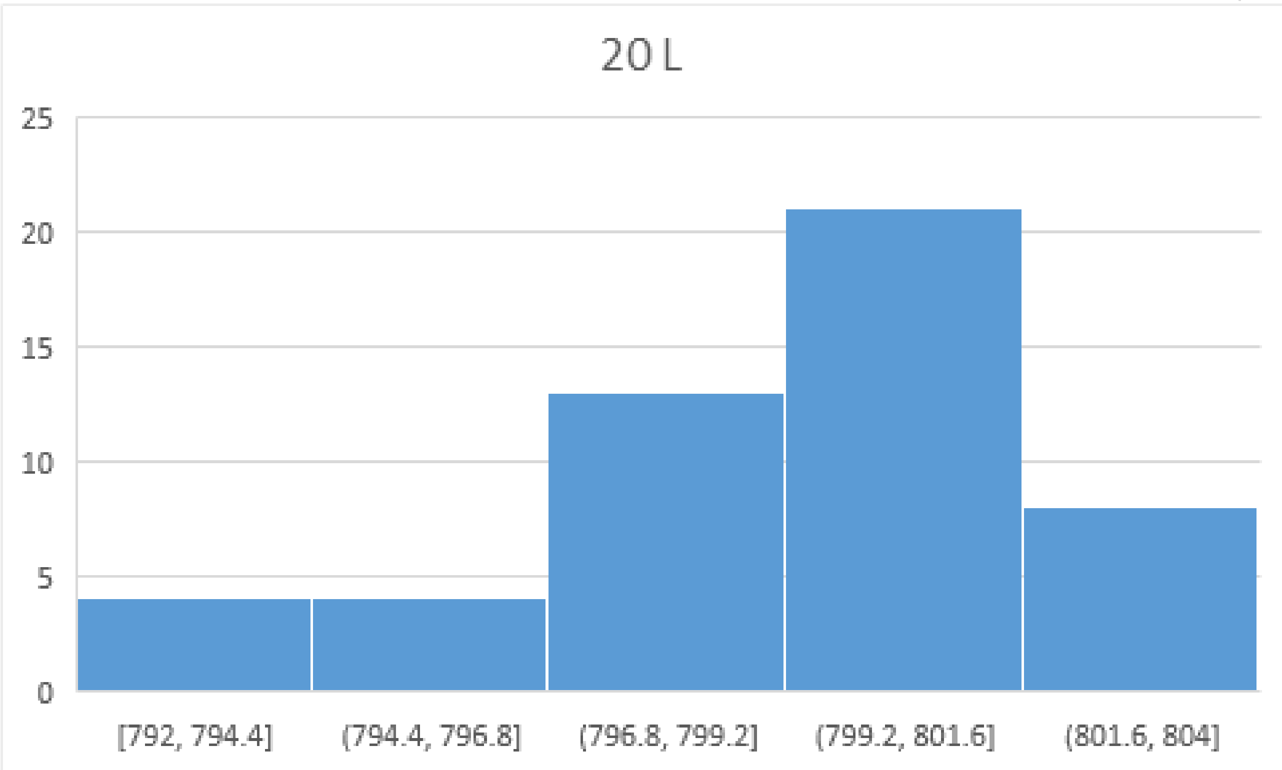
3.4 Data Visualised

3.4.1 Range of power

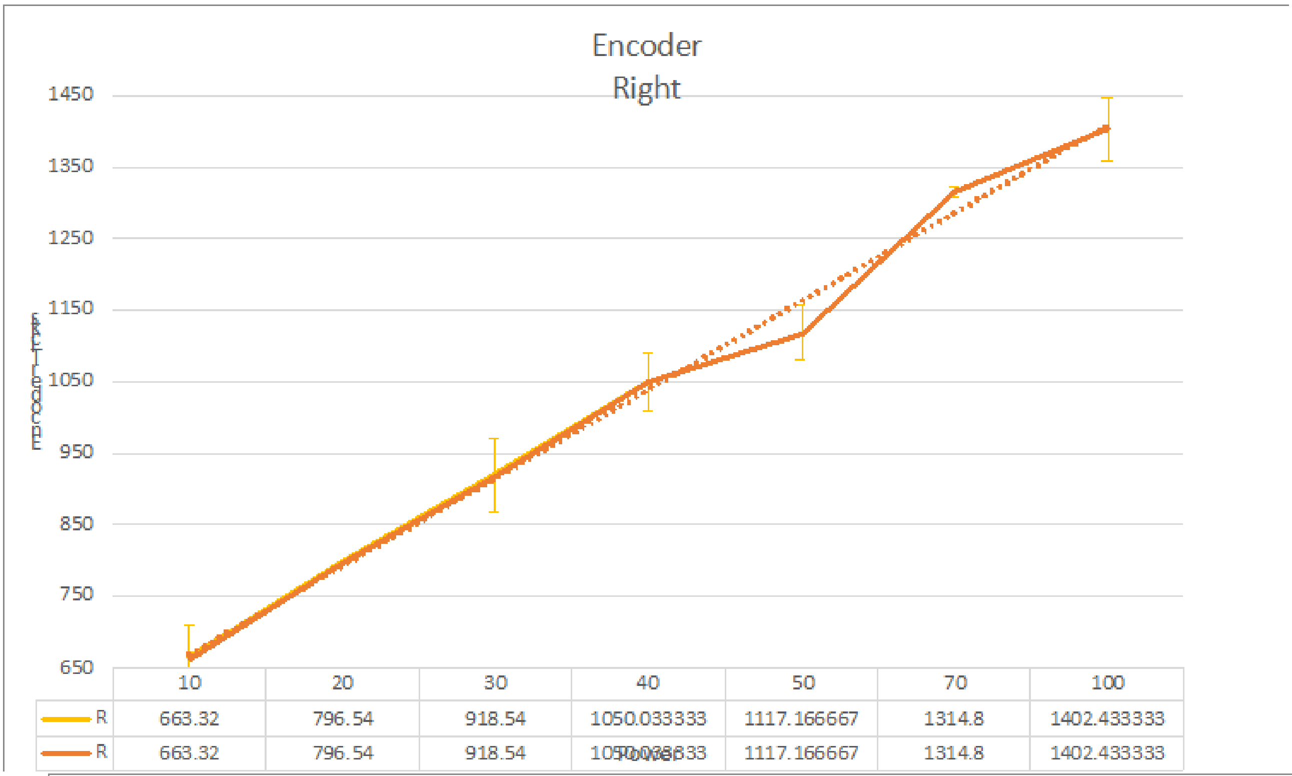


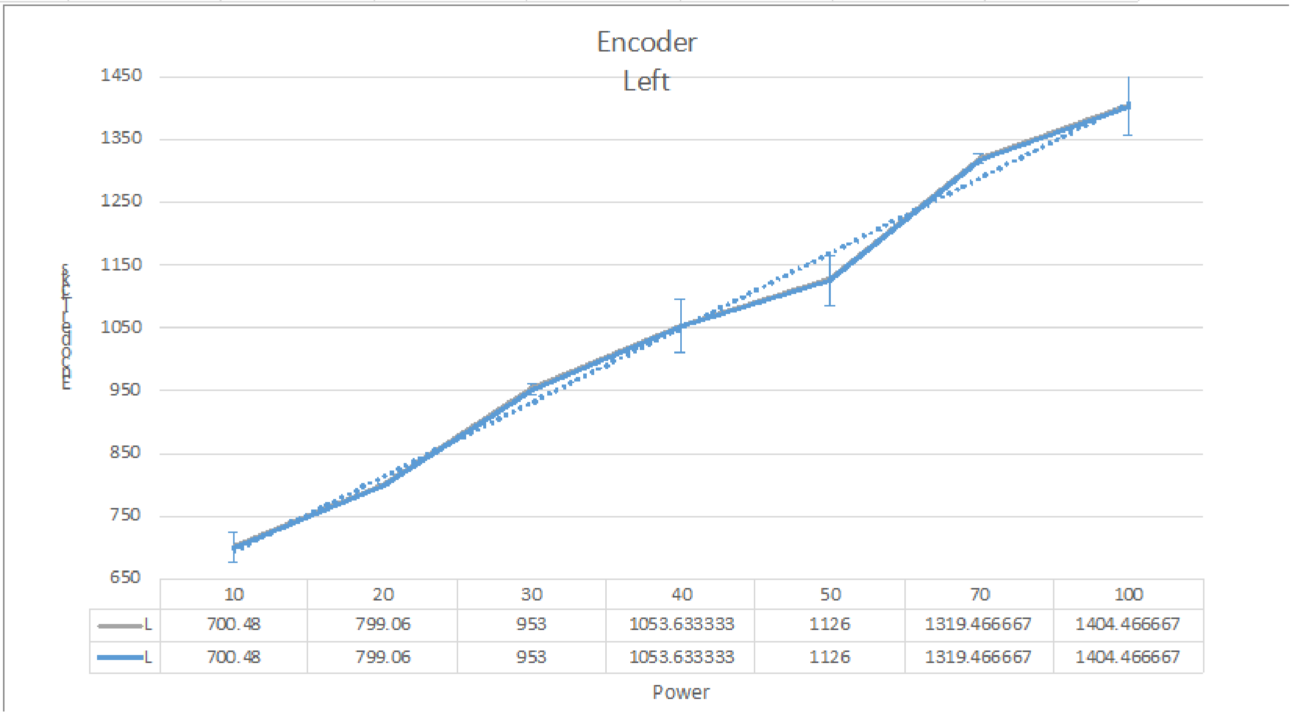
3.4.2 Repeatability & Reliability





3.4.3 Similarity





3.5 Discussion

3.5.1 Range of power

Without resistance on the motors, below a power of 10 the motors struggle to kick in and above 100 the encoder count levels out showing it has reached max speed. With resistance on the motors, below a power of 20 the motors struggle to kick in and above 120 the encoder count levels out showing it has reached max speed.

3.5.2 Repeatability

From running the robot at the same power over and over we can see that there is a range of encoder values showing that the encoder values fluctuate showing that the robot will not always travel the same distance in a set amount of time at a certain power.

3.5.3 Similarity

From running the encoders multiple times at different powers we can see that the error bars that there is a variation of how fast the motors go even at the same speeds.

References

[1] *Formula AllCode Robot* `https://www.matrixtsl.com/allcode/formula/`

[2] *in-robot API* allcode_api.h & allcode_api.o Pete Todd & Laurence Tyler 1.1