2D Range sensor for robotic object detection (Robotics and Controls)

Kalana Jayasooriya
Electrical and Electronics Department
University of Peradeniya
Sri Lanka
e18153@eng.pdn.ac.lk

Pandula Thennakoon
Electrical and Electronics Department
University of Peradeniya
Sri Lanka
e18359@eng.pdn.ac.lk

Shiham Firdous
Electrical and Electronics Department
University of Peradeniya
Sri Lanka
e18102@eng.pdn.ac.lk

I. ABSTRACT

The paper describes the working method of a simple 2D range sensor. The calibration, validation and verification of the sensor are also described.
This sensor is built using two IR range sensors attached to either side of a wooden board which is rotated by 160° using a servo motor. The servo motor is controlled by a microcontroller (Arduino mini), which rotates at specified angles at a specific time duration.

II. INTRODUCTION

The 2D range sensor, as the name suggests, is designed to measure the 2D distance of objects around it from a specific point. A linear distance measuring sensor (laser, IR, LIDAR, Ultrasonic, etc.) is rotated to the angle of need to map the 2D plane. The voltages obtained by these sensors are calibrated, and these readings are used to determine the actual distance of the object from the sensor. As the sensors are rotated, this gives the distance in a 2D plane [1]. These sensors are used in various applications such as robot perception, detecting components and parts, and handling and packaging.[2]

III. INITIAL SPECIFICATIONS

Initial Specifications of the Sensor

- 2.6V to 5.5V operating voltage
- Average active ranging sensor current: 19mA
- Measures absolute range up to 2m
- Operates in high infrared ambient light levels
- Linear resolution - 1mm
- Angular resolution - less than 5 deg
- Field of view - 360 deg
- Operating speed < 2 min
- The minimum distance measured: 5cm

IV. METHODS

A. *Principle of Operation and design of the sensor.*

IR sensors were used to measure the distance. An IR wave is sent out from the transmitter (Tx). Eventually, this wave of radiation will hit some object and will bounce back to the receiver (Rx). A voltage will be generated at the data pin corresponding to the intensity of this receiving wave (Fig1). Depending on the distance from the object to the sensor, the intensity of the receiving wave will change. Therefore, the output voltage at the data pin can be represented as a function of distance. Our interest is to find this relationship and analyze it statistically to calibrate the sensor.[1], [3]

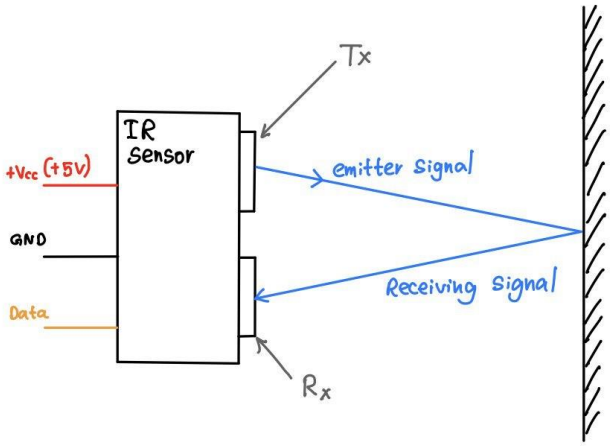


Fig. 1. Operation of the IR sensor

An Arduino board was used to make the rest of the sensor. To obtain the angular position, we have calibrated the Arduino. (Using the Arduino to sense the angle is not a valid method, some better methods are discussed in the discussion) We have attached the IR sensors to the circular platform that can be rotated around a vertical axis (Fig.2).

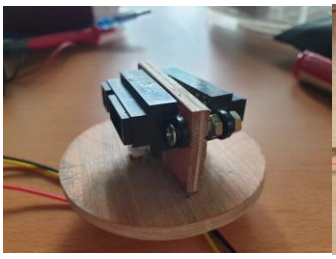


Fig. 2.a. Side view

Fig. 2.b. Top view

In order to rotate the platform, it was decided to use a stepper motor at first. But it came with some problems of its own, which are discussed in detail in the discussion. To avoid such issues, a servo motor that can rotate 180 degrees was used. But it was not possible to find a servo that could rotate 180 degrees. Due to this, a servo with a 160-degree rotation angle was used (Fig.3). In order to increase the range, two IR sensors were used to give out readings simultaneously. This improves the range as well as the operational time of the sensor.

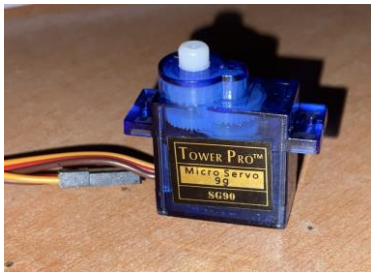


Fig. 3. The servo used.

But because the rotation angle of the sensor was 160 degrees, there is a blind range from 330 deg to 360 deg. The initial position of sensor 1 was 0 deg, while the initial position of sensor 2 was 170 deg. (10 deg angular resolution (Fig. 4)).

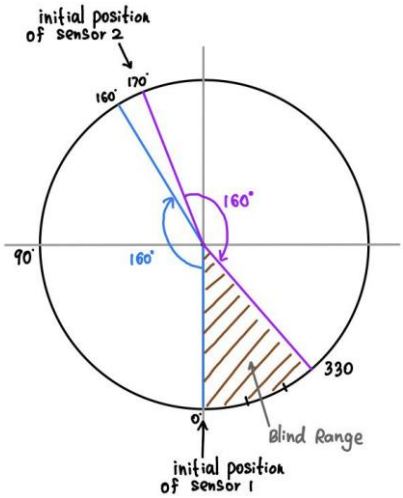


Fig. 4. Initial positions of two sensors

B. Calibration Setup

The servo motor was calibrated using the Arduino board to turn in 10-degree intervals. It will turn by 10 degrees and stop there for 1000 ms to give out the two outputs for angles. The calibration setup for the servo is shown below in fig 5. At each 10 degrene angle rotated, the output voltage was taken from the Arduino.
(Again, this is not a valid method to get the angular position). The code in figure 8 describes this calibration method.

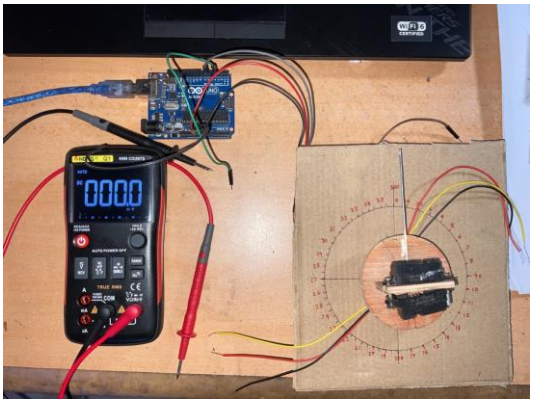


Fig. 5. The calibration setup for angular measurements.

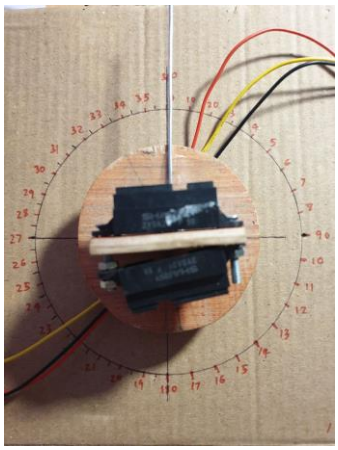


Fig. 5.b. Close-up image of the platform while calibrating.

Next up, the IR sensors were calibrated. The calibration setup is shown in Fig. 6. An object was placed in front of the sensor at certain intervals, and the output voltage was measured using a multimeter. The Zero reference line for distance measurement is shown in red color in Fig. 7. The distance was measured from this line to the obstacle surface.

Fig. 6. Calibration setup for the IR sensor.

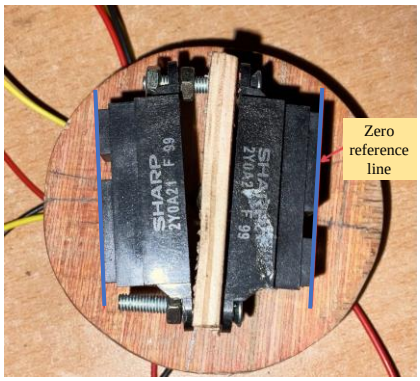


Fig. 7. The reference lines for distance measurements

The Arduino code for the servo calibration is shown below.

```
# include<Servo.h>

Servo serv;
int t = 1000;

void setup() {
  pinMode(3,OUTPUT);
  serv.attach(6);
}
void loop() {
serv.write(0);   //0
analogWrite(3, 0);
delay(t);

serv.write(10);   //10
analogWrite(3, 16);
delay(t);

serv.write(20);   //20
analogWrite(3, 32);
delay(t);

serv.write(31);   //30
analogWrite(3, 48);
delay(t);

serv.write(42);   //40
analogWrite(3, 64);
delay(t);

serv.write(53);   //50
analogWrite(3, 80);
delay(t);

serv.write(65);   //60
analogWrite(3, 96);
delay(t);

serv.write(76);   //70
analogWrite(3, 111.5);
delay(t);

serv.write(87);   //80
analogWrite(3, 127.5);
delay(t);

serv.write(100);   //90
analogWrite(3, 143);
delay(t);

serv.write(112);   //100
analogWrite(3, 159);
delay(t);

serv.write(124);   //110
analogWrite(3, 175);
delay(t);

serv.write(135);   //120
analogWrite(3, 191);
delay(t);

serv.write(146);   //130
analogWrite(3, 207);
delay(t);

serv.write(157);   //140
analogWrite(3, 223);
delay(t);

serv.write(168);   //150
analogWrite(3, 239);
delay(t);

serv.write(180);   //160
analogWrite(3, 255);
delay(t);

serv.write(0);   // back to 0 deg
analogWrite(3, 0);
delay(t);
}
```

Fig. 8. The Arduino code

C. *Statistical Analysis*

a. *IR Sensor*

First of all, for the IR sensor, a graph was drawn with one reading at one point from 1cm to 80cm for steps of 1 cm. This was done to get a rough idea of the shape of the graph. This came in handy in later calculations, which are described in observations. After that, ten readings were taken from each point in steps of 5cm. MS EXCEL was used to plot the graphs and obtain the necessary parameters and errors. [4, Ch. 2,7]

b. *Angle sensor*

The method used for angle sensing was not acceptable. The Arduino was calibrated to give out a voltage corresponding to the position of the servo. This is further discussed under discussion. Because this value does not change regardless of the number of samples per point, statistical analysis for this method is useless. However, in future designs, this will be improved further.

D. *Final Design*

After assembling the sensor, the final design is shown in fig 9.

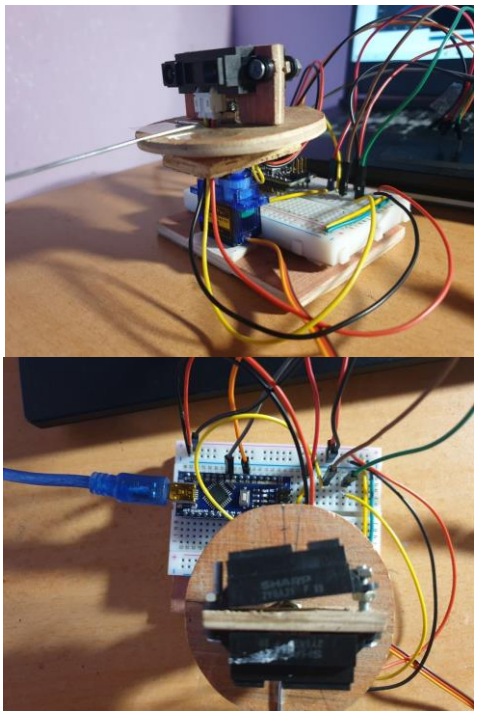
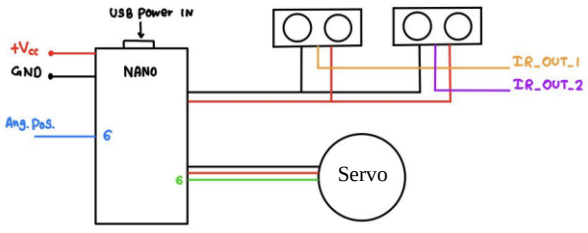


Fig. 9. Top and side view of the sensor.

The circuit diagram is shown below in Fig. 10.



Fig, 10. The circuit diagram

E. *Testing Setup*

The testing was done separately to the angle sensor and the IR sensor. There is a problem with the method used to obtain the angular measurement. This is addressed further in the discussion and possible techniques to eliminate such issues.

The same setup as the calibration setup for the IR sensor was used (Fig. 6). The obstacle was placed at a known distance, and the voltage was measured. Then the distance value was obtained using the calculated parameter values.

V. RESULTS

A. *IR sensor 1*

a. *Observations*

At first, a graph with distance vs. voltage was drawn. The following shape in Figure 11 was obtained.
This is similar to the graph shown in the datasheet for the sensor.[5]

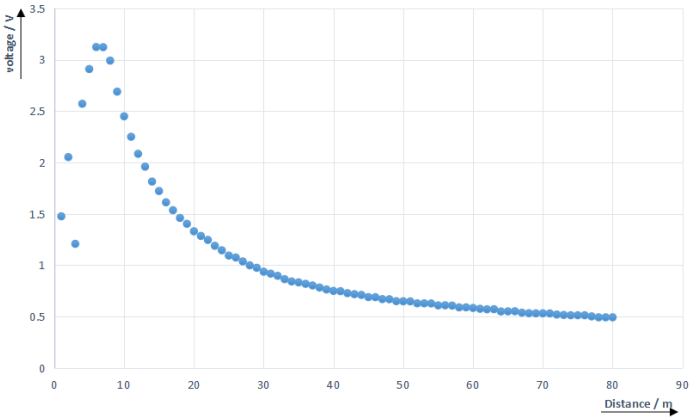


Fig. 11. Graph of Distance vs. Voltage

As we can see, from 0 cm to 7 cm, voltage increases rapidly. Then from 7 cm to 80 cm, voltage again decreases. Therefore, we selected the region 7 cm to 80 cm as our range[6]. And also, from observing the graph from 7 cm to 80 cm, we assumed a log relationship. Extracting information from this graph was very helpful later.
Next up, ten measurements per point were taken. Distance points were chosen as, 7cm, 10cm, 15cm,…, 75cm,80cm (5 cm steps).
The results are shown below.

TABLE I

Distance(D)	Voltage reading(V)									
	1	2	3	4	5	6	7	8	9	10
7	3.090	3.090	3.092	3.099	3.112	3.08	3.108	3.091	3.106	3.107
10	2.374	2.374	2.467	2.391	2.3339	2.374	2.392	2.4	2.4052	2.427
15	1.688	1.688	1.689	1.668	1.65	1.687	1.7	1.705	1.687	1.686
20	1.313	1.313	1.314	1.35	1.332	1.331	1.33	1.313	1.316	1.313
25	1.096	1.096	1.097	1.095	1.099	1.096	1.095	1.094	1.114	1.096
30	0.951	0.951	0.952	0.954	0.957	0.95	0.959	0.94	0.938	0.94
35	0.842	0.842	0.86	0.862	0.861	0.842	0.843	0.843	0.843	0.842
40	0.766	0.766	0.765	0.767	0.769	0.767	0.766	0.768	0.766	0.765
45	0.708	0.708	0.708	0.706	0.705	0.704	0.708	0.708	0.707	0.708
50	0.67	0.67	0.661	0.657	0.665	0.659	0.663	0.65	0.653	0.655
55	0.622	0.622	0.611	0.611	0.615	0.619	0.611	0.612	0.625	0.611
60	0.588	0.588	0.587	0.584	0.581	0.579	0.579	0.577	0.58	0.58
65	0.552	0.552	0.564	0.56	0.558	0.554	0.553	0.553	0.557	0.554
70	0.534	0.534	0.533	0.531	0.532	0.533	0.533	0.534	0.535	0.533
75	0.514	0.514	0.514	0.514	0.51	0.503	0.512	0.511	0.515	0.51
80	0.495	0.495	0.492	0.49	0.489	0.494	0.494	0.493	0.494	0.49

b. *Input-Output Characteristic Graphs*

Using the trendline function on excel, the following graph in Fig. 12.b. was plotted and the curve fitted in log scale.

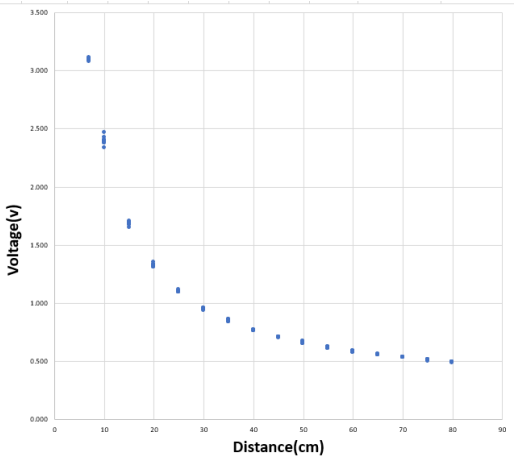


Fig. 12.a. Distance vs. Voltage Characteristic graph

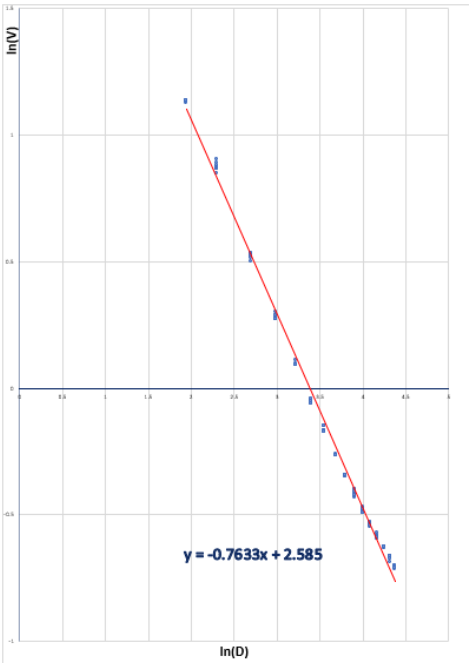


Fig. 12.b. Characteristic graph in log scale

c. *Parameter Values*

$$ln(v) = -0.7633ln(D) + 2.585$$

$$ln(v) + ln(D^{0.7633}) = 2.585$$

$$VD^{0.7633} = e^{2.585}$$

$$D = \left(\frac{A}{V}\right)^{B} \qquad (1)$$

Where $A = e^{2.585}$ and $B = \frac{1}{0.7633}$ are the parameter values. We can calculate the systematic error by calculating the root mean square errors for the values obtained from the above (1).

d. *Calibration Results*

The next step was to calculate the errors.
The standard error at each point was calculated to obtain the random error using (2).

$$\text{standard error} = \frac{\sigma}{\sqrt{n}} \qquad (2)$$

The following Table II shows the calibration results with errors.

TABLE II

.

avg	std deviation	std error (RANDOM	Distance from eqn oor mean volta	deviation
3.098	0.010	0.0033	6.7222	-0.2778
2.394	0.036	0.0113	9.4219	-0.5781
1.685	0.016	0.0049	14.9274	-0.0726
1.323	0.013	0.004	20.4996	0.4996
1.098	0.006	0.0018	26.1636	1.1636
0.949	0.007	0.0023	31.6556	1.6556
0.848	0.009	0.0028	36.694	1.694
0.767	0.001	0.0004	41.8878	1.8878
0.707	0.001	0.0005	46.5653	1.5653
0.660	0.007	0.0022	50.9265	0.9265
0.616	0.006	0.0018	55.7891	0.7891
0.582	0.004	0.0013	60.0437	0.0437
0.556	0.004	0.0013	63.8368	-1.1632
0.533	0.001	0.0004	67.3888	-2.6112
0.512	0.004	0.0011	71.1223	-3.8777
0.493	0.002	0.0007	74.7566	-5.2434
Avg. random error=		0.0025	root mean square error=	2.5657

$$root\ mean\ square\ error = \sqrt{\frac{deviation^2}{10}}$$

$$= 2.5657\ cm$$

The above value is somewhat a significant error and not acceptable. By further inspecting the data, we can see that the deviation for the last 3 points significantly affects the root mean square error. Therefore, we can be confident that our characteristic equation gives better results by removing these three points. Note that this process will shorten the range.

Table III shows the newly calculated errors.

TABLE III

avg	std deviation	std error (RANDOM)	Distance from eqn for mean voltage	deviation
3.098	0.010	0.0033	6.7222	-0.2778
2.394	0.036	0.0113	9.4219	-0.5781
1.685	0.016	0.0049	14.9274	-0.0726
1.323	0.013	0.004	20.4996	0.4996
1.098	0.006	0.0018	26.1636	1.1636
0.949	0.007	0.0023	31.6556	1.6556
0.848	0.009	0.0028	36.694	1.694
0.767	0.001	0.0004	41.8878	1.8878
0.707	0.001	0.0005	46.5653	1.5653
0.660	0.007	0.0022	50.9265	0.9265
0.616	0.006	0.0018	55.7891	0.7891
0.582	0.004	0.0013	60.0437	0.0437
0.556	0.004	0.0013	63.8368	-1.1632
Avg. random error=		0.0029	root mean square error=	1.2838

$$new\ root\ mean\ square\ error = 1.2838\ cm$$

Hence, we can get a less room mean square error. It is clear that at 70 cm, 75 cm, 80 cm, the characteristic equation yields a higher error. That is why the range of the sensor was reduced to 65 cm.

B. *IR sensor 2*

a. *Observations*

Doing the calibration in the same way, we obtain the following.

TABLE IV

Distance(D)	Voltage reading(V)									
	1	2	3	4	5	6	7	8	9	10
7	3.104	3.105	3.1	3.099	3.076	3.077	3.096	3.011	3.004	3.103
10	2.465	2.464	2.465	2.425	2.426	2.43	2.42	2.425	2.444	2.443
15	1.695	1.693	1.709	1.677	1.68	1.66	1.677	1.676	1.683	1.68
20	1.319	1.32	1.318	1.3	1.301	1.299	1.31	1.3	1.31	1.318
25	1.09	1.096	1.095	1.082	1.087	1.1	1.083	1.102	1.083	1.087
30	0.93	0.94	0.939	0.94	0.926	0.927	0.926	0.94	0.944	0.94
35	0.809	0.81	0.81	0.811	0.816	0.89	0.81	0.809	0.81	0.81
40	0.715	0.712	0.713	0.72	0.718	0.73	0.721	0.712	0.715	0.713
45	0.648	0.668	0.648	0.649	0.648	0.649	0.641	0.647	0.647	0.6487
50	0.608	0.609	0.611	0.609	0.608	0.608	0.59	0.6	0.605	0.604
55	0.569	0.568	0.569	0.567	0.569	0.569	0.568	0.57	0.568	0.569
60	0.53	0.531	0.53	0.53	0.532	0.529	0.537	0.541	0.54	0.53
65	0.509	0.514	0.515	0.515	0.511	0.51	0.509	0.5	0.51	0.51

b. *Input-Output Characteristic Graphs*

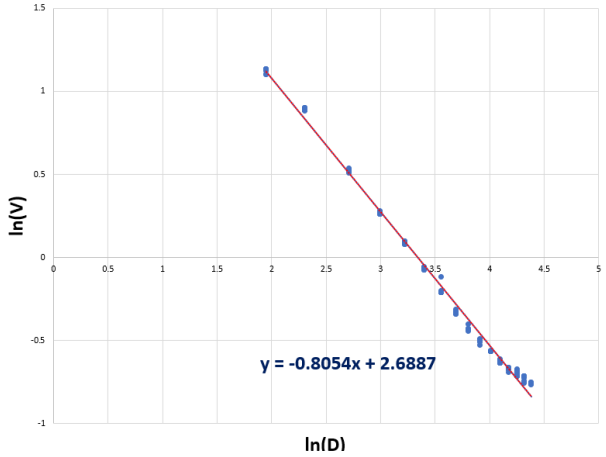


Fig. 13. Characteristic graph for IR sensor 2

c. Parameter Values

$$ln(v) = -0.8054ln(D) + 2.6887$$

$$ln(v) + ln(D^{0.8054}) = 2.6887$$

$$VD^{0.8054} = e^{2.6887}$$

$$D = \left(\frac{A}{V}\right)^{B} \qquad (3)$$

Where $A = e^{2.6887}$ and $B = \frac{1}{0.8054}$ are the parameter values.

d. Calibration Results

TABLE V

avg	std deviation	std error (RANDOI	Distance from eqn for mean volta	deviation
3.078	0.038	0.0121	6.977	-0.023
2.441	0.018	0.0058	9.3042	-0.6958
1.683	0.013	0.0042	14.7609	-0.2391
1.310	0.009	0.0028	20.1569	0.1569
1.091	0.007	0.0023	25.2992	0.2992
0.935	0.007	0.0022	30.616	0.616
0.819	0.025	0.008	36.1261	1.1261
0.717	0.006	0.0018	42.5881	2.5881
0.649	0.007	0.0022	48.1544	3.1544
0.605	0.006	0.002	52.5559	2.5559
0.569	0.001	0.0003	56.7884	1.7884
0.533	0.005	0.0014	61.5352	1.5352
0.510	0.004	0.0014	64.952	-0.048
total random error=		0.0036	root mean square error=	1.7619

$$root\ mean\ square\ error = 1.7619\ cm$$

C. Angle sensor

The method used to sense the angle is not a valid one. A voltage value corresponding to the position of the servo was obtained. The results are shown below (Table VI).

TABLE VI

Position/deg	Voltage/mV
0	0
10	298
20	595
30	893
40	1190
50	1488
60	1782
70	2065
80	2364
90	2662
100	2959
110	3257
120	3554
130	3840
140	4150
150	4439
160	4745

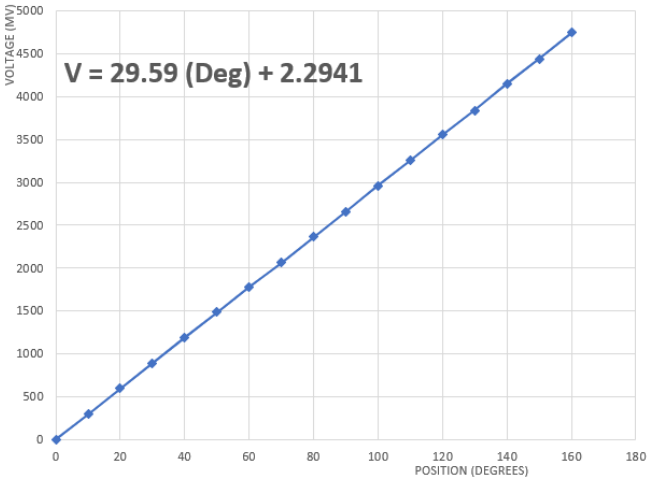


Fig. 14. Characteristic graph for angle sensing method

As we can see from Fig 14, a perfectly linear graph was obtained.

$$voltage = 29.59(angle) + 2.2941$$
$$angle = 0.0338(voltage) - 0.7753 \quad (4)$$

Above (4) describes the angle with voltage value.
Only one set of voltage values were taken because repeating the same angle makes no difference on the output voltage. Therefore, in this case, doing statistics is useless because there is no error in angle measurement. But above method is not a valid one. Alternate methods are discussed in the discussion.

Verification Results

a. For sensor 1

The same calibration setup was used as the verification setup. A known distance was given, and the voltage values were noted down. Then using the characteristic equation for sensor 1, the predicted distance was calculated. Following results were obtained (Table VII).

TABLE VII

Known Distance (cm)	Voltage reading(V)	Distance obtained from equation(cm)	Variation	Percentage Variation
7	3.081	6.769	-0.231	-3.294
10	2.427	9.254	-0.746	-7.465
15	1.688	14.890	-0.110	-0.731
20	1.332	20.308	0.308	1.541
25	1.115	25.636	0.636	2.544
30	0.939	32.107	2.107	7.023
35	0.848	36.694	1.694	4.840
40	0.75	43.099	3.099	7.748
45	0.69	48.074	3.074	6.831
50	0.65	51.986	1.986	3.973
55	0.611	56.376	1.376	2.502
60	0.59	59.019	-0.981	-1.635
65	0.55	64.705	-0.295	-0.454
		mean variation =	0.917	

As we can see, the deviation from the actual distance value is minimal (< ± 8 %).

In the same way, sensor 2 and the angle sensor can be verified.

Validation Results

Initially, the plan was to use only one distance sensor, but two distance sensors were used as the project proceeded.

Initial operating time = 2 min per cycle
Obtained operating time = 16 seconds per cycle

i. Distance sensor

- Initial range = 5 cm to 2 m
- Obtained range = 7 cm to 65 cm

ii. Angular sensor

- Initial range = 360 deg
- Obtained range = 330 deg
- Initial resolution = less than 5 deg
- Obtained resolution = 10 deg
-

Final Specifications obtained.

- Operating time = 16 seconds per cycle
- Linear range = 7 cm to 65 cm
- Angular range = 0 deg to 330 deg
- Angular resolution = 10 deg
- Power input = 6 V to 10 V

VI. DISCUSSION

A. *Strengths and weaknesses*

The main weakness of the current sensor version is that the method used for angular sensing is not a valid one. The main problem with this is, Arduino controls the output voltage. For example, let's say the sensor is going from 30 deg to 40 deg. The Arduino will give a voltage relevant to 40 deg to the output terminal. But if the sensor got blocked at 35 deg for some reason (getting stuck by an obstacle, for instance), the output voltage should correspond to 35 deg, not 40 deg. This is a huge problem, and we are aware of it. However, we were forced to use this method due to time restrictions. Also, we could not find any encoders within time to finish designing a new setup arrangement. Implementing an encoder to the current system would cause the whole design to change, taking more time. In future versions of this sensor, our goal is to implement a better method for angle sensing.

Some ideas for future development plan

i. One of the methods that we thought to implement was a rotary potentiometer.

The following Figure 15 shows the working of this method.

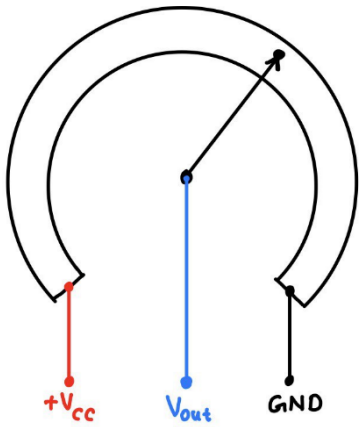


Fig. 15. Potentiometer idea

The V_{out} pin gives a voltage corresponding to the angular position of the sensor.
A rotary potentiometer was tested to find more details, and it was found that the rotary potentiometer was not linear. Also, it had a dead zone from 0 deg to nearly 20 deg. This was not acceptable because the range would be reduced, and the error will be significant.

ii. Another method was to use an absolute encoder.

An advantage of our sensor is its shorter operating time. Because two IR sensors are working simultaneously to give out the distance, the operating time was reduced by half. We were able to reduce the operating time to 16 seconds.
Other disadvantages are reduced angular range, reduced-resolution (angular and linear), reduced linear range.

B. *Difficulties faced*

There were so many obstacles.

i. One of the main problems faced was access to equipment and tools. The initial specifications indicated that a laser rangefinder would be used. But we were unable to find a laser rangefinder module due to stocks not being available. Therefore, we were forced to use IR sensors which were much cheaper and easy to find. Also, we needed power drills and saws. We did not use any glue because we wanted

everything to be assembled with nuts and bolts so that, in case we decided to change something later, it would be much easier to recover parts and do so. Also, we could not visit any stores due to the current situation in the country. We had to buy all the sensors and equipment online. It took three days for the courier to deliver an order. When the design ideas were changed, we had to reorder and wait three days to restart the project.

ii. Another problem was with the motor. We initially used a stepper motor and ran into some issues. Because we were not using an absolute encoder, if the power went off while the sensor was operating, the position must be recalibrated for the sensor to work correctly. Therefore, we decided to use servos to return to the initial position if the power went off in the middle of the operation. However, this led to another problem. The servos we could find did not rotate 360 degrees. Therefore, we decided to go with a 180-degree servo. But when we received the servo, we found out that it can only rotate 160 degrees. This was due to the servo being a copy of an original one. Since it was hard and expensive to find an original servo, we had to use the 160-degree servo. Since this would affect the angular range, we decided to use two IR sensors simultaneously. By doing this, we were able to recover up to 330 degrees of range. However, this idea cut down the operating time by half.

iii. Another problem was with the angular position sensor. We were unable to find an absolute encoder with the time limit. Therefore, even it is not a valid method; we had to take the angular output from the Arduino.

iv. The last problem was with the team members being far away. It was not easy to meet and work on the project. Often time we used online platforms to plan the project.

B. Comparison with other similar sensors

Compared to commercial-grade sensors, the current version of the sensor does not hold well. The angular resolution of 10 deg is inferior compared to the resolution of these sensors (Hokuyo UTM-30LX-EW sensor has an angular resolution of 0.25 deg) [7]

Also, the range is poor. (RPLidar A2 sensor has a full 360-degree range) [8]

Linear resolution is also poor because there was a somewhat significant error.

This version of the sensor was done under Rs. 5000 (Excluding the budget for all the tools)

VII. CONCLUSION

The main aim of the project was to develop a 2D range sensor. The final design of the sensor is capable of measuring distances in a range of 7 to 65 cm with an angular range of 0º to 330º. However, the method used to sense the angle is not acceptable, and it will be subjected to changes in future designs. This sensor is a cheap solution for sensors in the market which have a short range of operation.

VIII. REFERENCES

[1]“ir_distance_sensor_arduino_tutorial_sharp_gp2y0a21yk0f.pdf.”

[2] D. Ibrahim, “Mobile robot project: the Buggy,” 2021, [Online]. Available: https://www.sciencedirect.com/topics/engineering/distance-sensor

[3] R. Burnett, “Ultrasonic vs Infrared (IR) Sensors – Which is better?,” Nov. 2017, [Online]. Available: https://www.maxbotix.com/articles/ultrasonic-or-infrared-sensors.htm

[4] A. J. Wheeler and A. R. Ganji, *Introduction to engineering experimentation*, 3rd ed. Upper Saddle River, N.J: Pearson Higher Education, 2010.

[5] “28995-Sharp-GP2Y0A21YK0F-IR-Datasheet.pdf.”

[6]“Adarsh_2016_IOP_Conf._Ser.__Mater._Sci._Eng._149_012141.pdf.”

[7] “UTM-30LX-EW_Specification.pdf.”

[8] “robopeak_2d_lidar_brief_en_A2M4.pdf.”