COMPONENTS AND DEVICES

Laboratory

OPTICAL DISTANCE SENSOR

Group No: 2.4

By,

Mithiliesh Muley (258575)

Sagar Agarwal (258472)

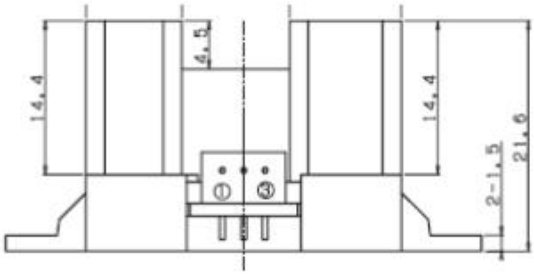
Aim:

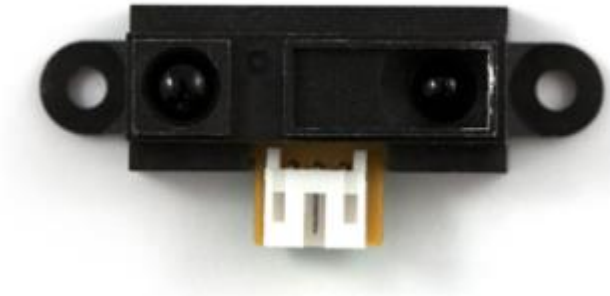
To calibrate and analyse the optical displacement sensor and to measure the characteristics of distance sensor

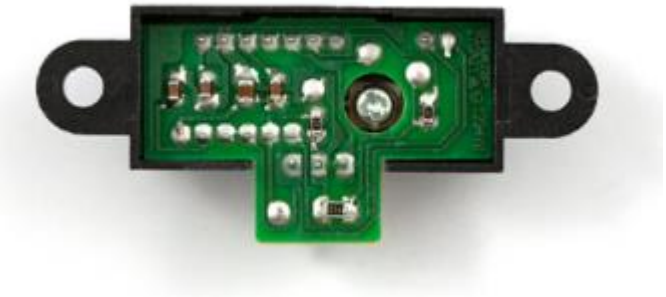
Theoretical background:

The Optical Distance Sensor uses a pulsed light signal to return a value representing the distance from a target surface. As the distance from the target changes the returned value changes indicating whether the movement is towards the target or away from the target.

The Optical Distance Sensor (ODS) is an analog sensor that uses electro optical proximity detection to calculate distance from an object based on the intensity of the light. A optical distance meter works by use measuring the time it takes a pulse of laser light to be reflected off a target and returned to the sender. This is known as the "time of flight" principle, and the method is known either as “time of flight” or “pulse” measurement.

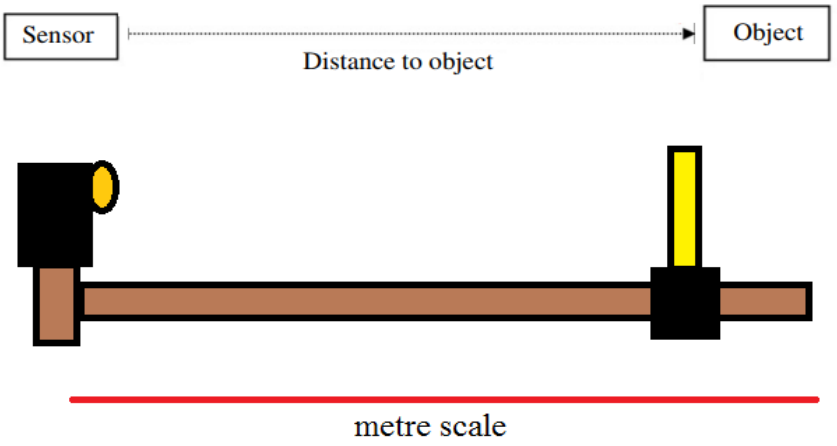




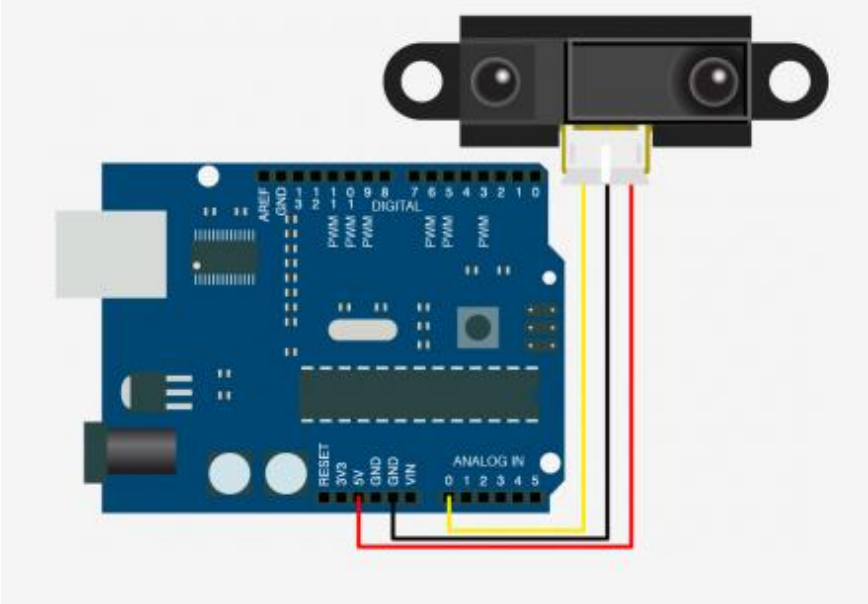


Experimental setup:

In this experiment we have used Optical distance sensor (Gp2YoA21Yk) , Arduino Microcontroller board, LCD display and different objects like Brown, yellow and aluminium plate and a measuring tape. The experimental setup figure is shown below.



Sensor connected with Arduino board:



LCD Display:

Procedure:

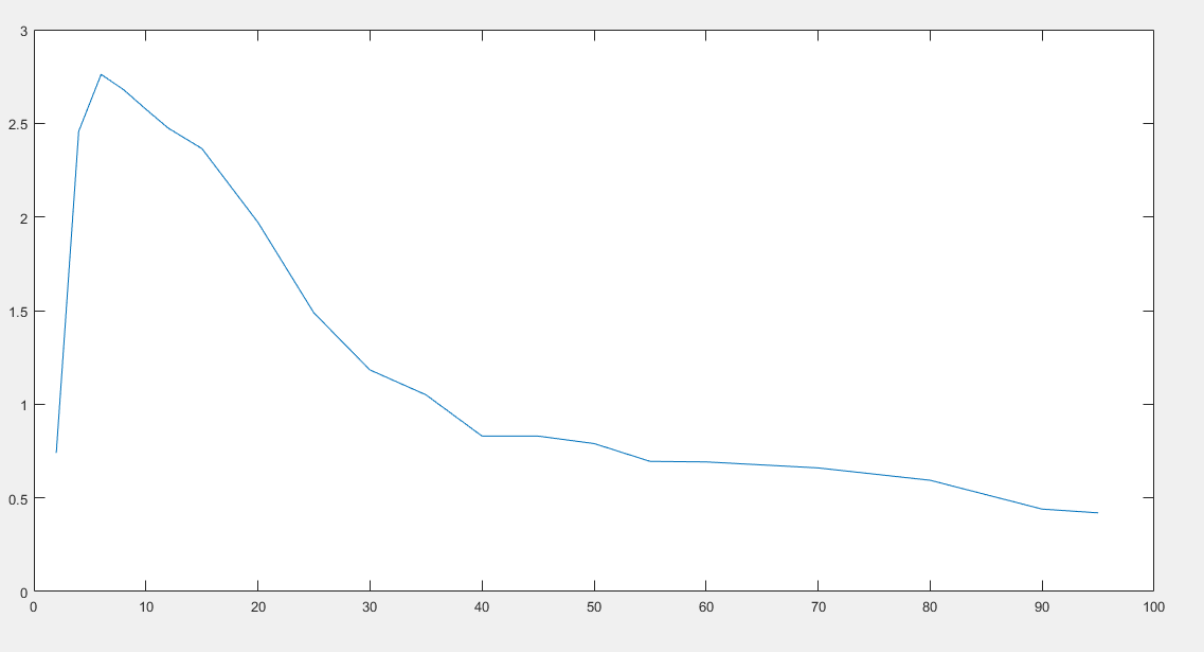
Initially we placed the aluminium plate near the sensor starting from 2cm and the voltage is measured using the multi-meter. We used different plates like brown and yellow and we calculated the voltage value respectively. The corresponding values are noted down and are shown in the tables and we plotted the graph in order to understand the characteristics of sensor output.

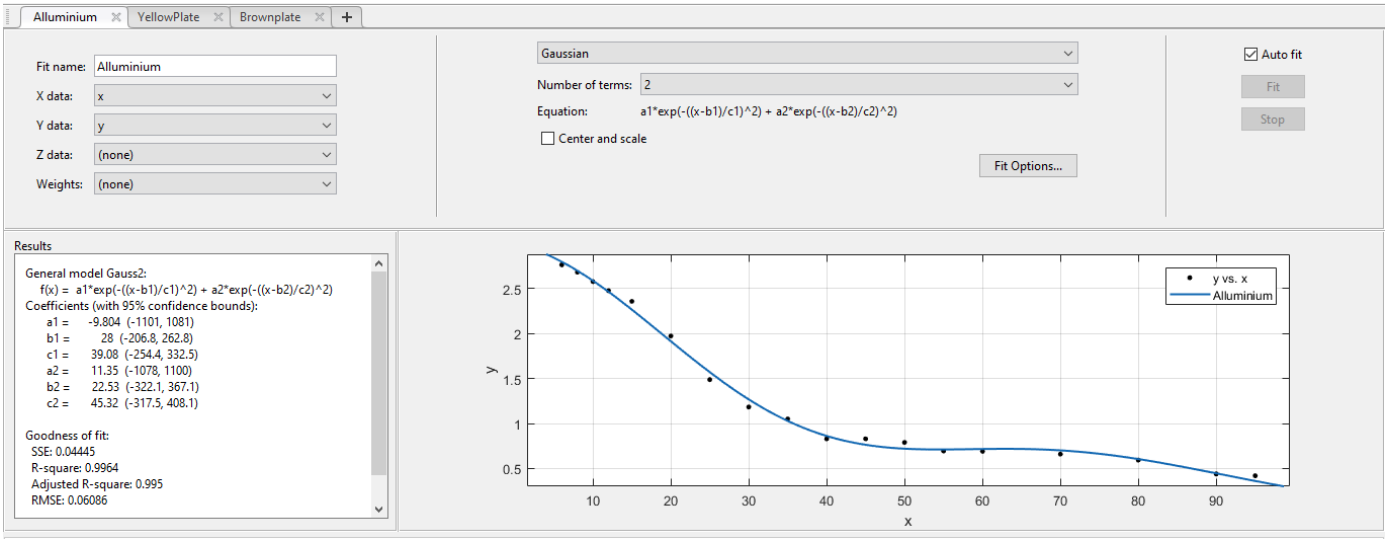
Aluminium plate:

Distance (cm)	Outputvoltage (V)
2	0.74
4	2.457
6	2.761
8	2.68
10	2.575
12	2.474
15	2.365
20	1.971
25	1.487
30	1.183
35	1.050
40	0.83
45	0.83
50	0.79
60	0.692
70	0.66
80	0.594
90	0.440
95	0.420

Max value V at distance = 6 cm, region before this is neglected

The resulting graph is shown below.



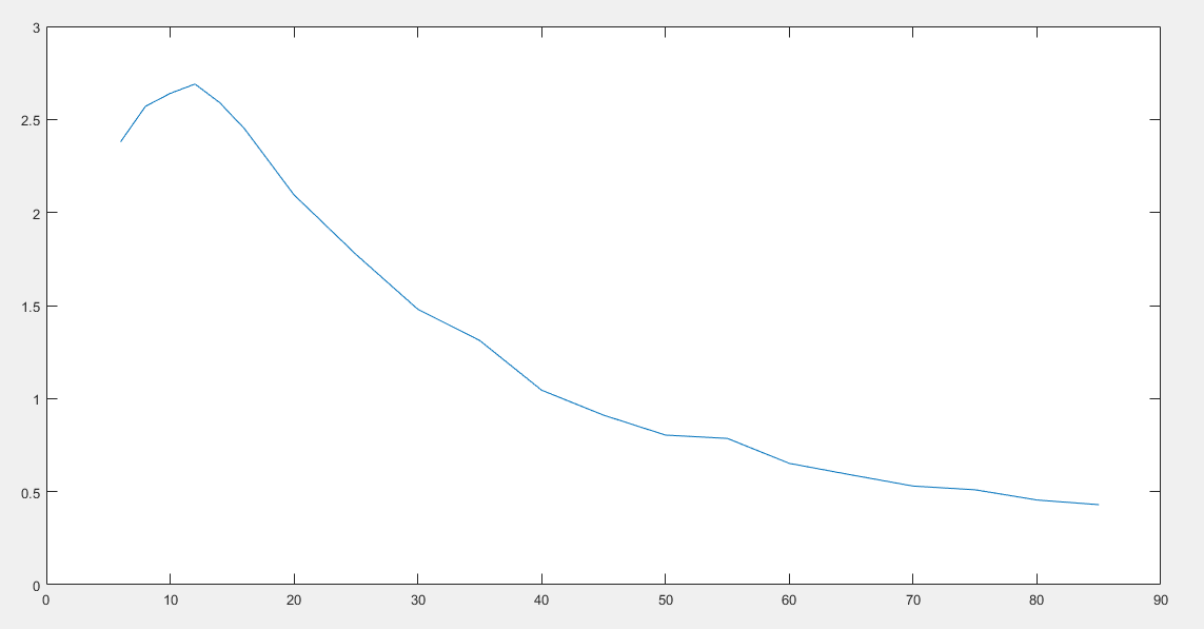


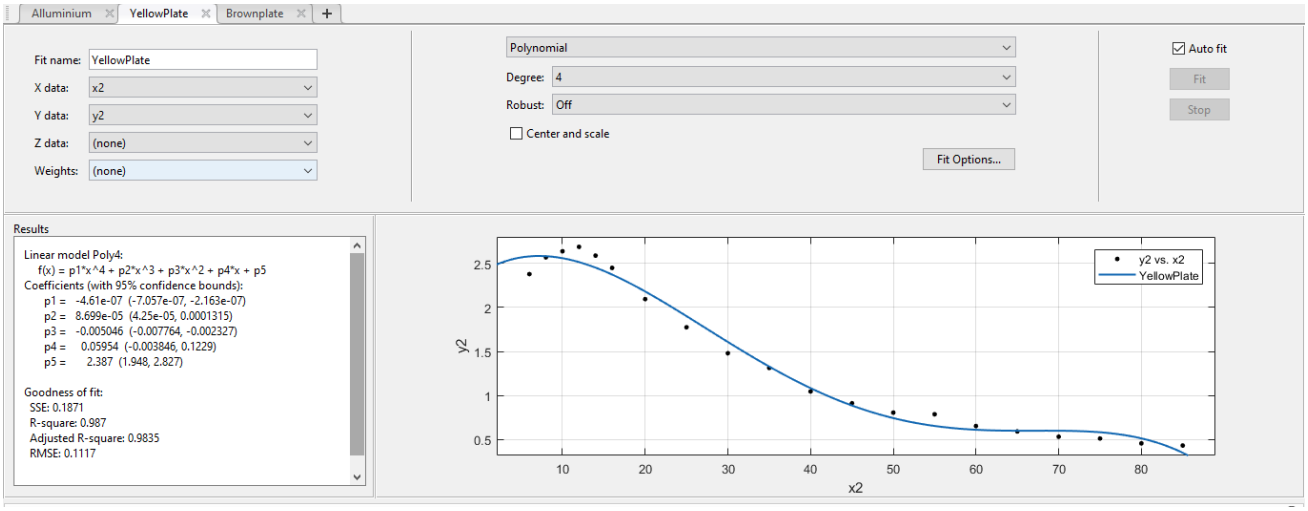
The graph above is a plot of the different distance and voltage. Distance along the X-axis and corresponding voltage measured along the Y-axis. The graph shows an exponentially decreasing curve with an equation **p1*pow(x,4) + p2*pow(x,3) + p3*pow(x,2) + p4*x + p5**

Yellow plate:

Distance (cm)	Output voltage (V)
6	2.380
8	2.570
10	2.640
12	2.690
14	2.59
16	2.45
20	2.095
25	1.775
30	1.480
35	1.312
40	1.045
45	0.911
50	0.804
55	0.786
60	0.652
65	0.590
70	0.530
75	0.510
80	0.455
85	0.430

The resulting graph for the yellow plastic is shown below.





The graph shows an exponentially decreasing curve with an equation

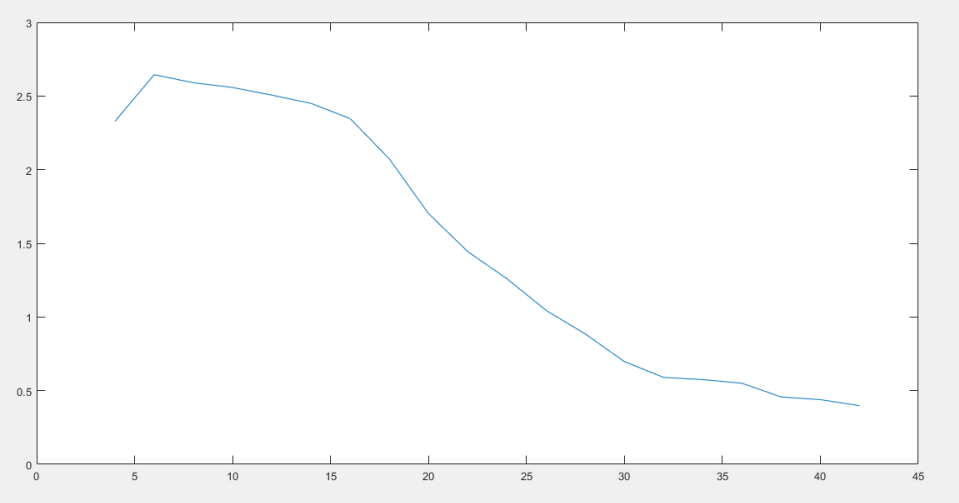
p1*pow(x,3) + p2*pow(x,2) + p3*pow(x,1) + p4

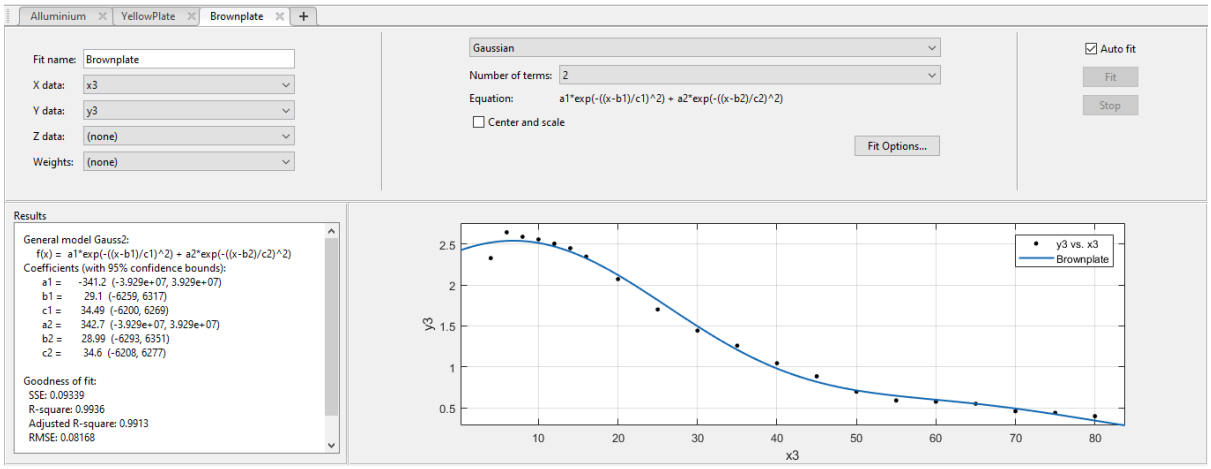
Brown Plate:

The calculated values are shown in the table below.

Distance (cm)	**Output voltage (V)**
4	**2.329**

6	**2.645**
8	**2.590**
10	**2.558**
12	**2.506**
14	**2.45**
16	**2.347**
20	**2.073**
25	**1.702**
30	**1.444**
35	**1.260**
40	**1.045**
45	**0.885**
50	**0.697**
55	**0.590**
60	**0.575**
65	**0.550**
70	**0.457**
75	**0.439**
80	**0.398**





The graph shows an exponentially decreasing curve with an equation

p1*pow(x,3) + p2*pow(x,2) + p3*pow(x,1) + p4

Connections for Sensor to Arduino microcontroller board:

The connection details made above are as follows:

Arduino A0 – One end of Sensor.

Other end of the resistor is grounded. Arduino A4- SDA of LCD

Arduino A5- SCL of LCD

LCD ground and +5V are given as respected.

The above connections are made, the software code as given below is uploaded and the program is run. The LCD thus displays the relevant distance in cm on the LCD, when the object is moved from the sensor.

Software Code for Aluminium:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>#define BACKLIGHT_PIN
13int sensorValue = 0;LiquidCrystal_I2C lcd(0x20);
// set the LCD address to 0x20// the setup routine runs once when you press reset:void setup() {
// initialize serial communication at 9600 bits per second:  Serial.begin(9600);
// Switch on the backlight  pinMode ( BACKLIGHT_PIN, OUTPUT );
digitalWrite ( BACKLIGHT_PIN, HIGH );
lcd.begin(16,2);
// initialize the lcd   lcd.home ();
// go home  lcd.print("Inialized");
delay(500);
lcd.clear();  lcd.setCursor ( 0, 0 );
lcd.print("Dist");  }// the loop routine runs over and over again forever:void loop() {
// read the input on analog pin 0:
sensorValue = analogRead(A1);
// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
float voltage = sensorValue * (5.0 / 1023);
// print out the value you read:  Serial.println(voltage);
float x = voltage;
float  p1 =   -11.89;
float  p2 =    65.15;
float  p3 =    -131.8;
float  p4 =   118.5 ;
float distance= p1*pow(x,3) + p2*pow(x,2) + p3*pow(x,1) + p4;  delay(1000);
// delay in between reads for stability  lcd.setCursor ( 1, 1 );
lcd.print(distance); }
```

Actual distance (cm)	LCD distance (cm)
17	17.83
12	15.63
20	19.903
25	21.923
30	24.628
35	29.023
40	38.073
45	44.04
50	50.74

Yellow plate:

```
/*  AnalogReadSerial  Reads an analog input on pin 0, prints the result to the serial monitor.
Graphical representation is available using serial plotter (Tools > Serial Plotter menu)
Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.
This example code is in the public domain.
*/#include <Wire.h>
 #include <LiquidCrystal_I2C.h>#define BACKLIGHT_PIN     13int sensorValue = 0;
LiquidCrystal_I2C lcd(0x20);  // set the LCD address to 0x20// the setup routine runs once when you press reset:
void setup() {  // initialize serial communication at 9600 bits per second:  Serial.begin(9600);
  // Switch on the backlight  pinMode ( BACKLIGHT_PIN, OUTPUT );
 digitalWrite ( BACKLIGHT_PIN, HIGH );
  lcd.begin(16,2);
       // initialize the lcd   lcd.home ();
                  // go home  lcd.print("Inialized");
   delay(500);  lcd.clear();
  lcd.setCursor ( 0, 0 );
 lcd.print("Dist");
 }// the loop routine runs over and over again forever:void loop() {  // read the input on analog pin 0:
 sensorValue = analogRead(A1);
 // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
 float voltage = sensorValue * (5.0 / 1023);
 // print out the value you read:  Serial.println(voltage);
 float x = voltage;
  float  p1 =   7.578;
float p2 =     -70.63;
float  p3 =      232.5;
 float  p4 =         -333.6 ;
  float   p5 =       200.8 ;
 float distance= p1*pow(x,4) + p2*pow(x,3) + p3*pow(x,2) + p4*x + p5;  delay(1000);
      // delay in between reads for stability
  lcd.setCursor ( 1, 1 );  lcd.print(distance); }
```

Actual distance (cm)	LCD distance (cm)
20	19.97
25	24.56
30	29.47
35	35.48
40	39.89
45	44.23
50	47.62
55	50.87

Brown plate:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>#define BACKLIGHT_PIN
13int sensorValue = 0;LiquidCrystal_I2C lcd(0x20);
// set the LCD address to 0x20// the setup routine runs once when you press reset:void setup() {
// initialize serial communication at 9600 bits per second:  Serial.begin(9600);
// Switch on the backlight  pinMode ( BACKLIGHT_PIN, OUTPUT );
digitalWrite ( BACKLIGHT_PIN, HIGH );
lcd.begin(16,2);
// initialize the lcd
lcd.home ();
// go home  lcd.print("Inialized");
delay(500);  lcd.clear();
lcd.setCursor ( 0, 0 );
lcd.print("Dist");
}// the loop routine runs over and over again forever:void loop() {
// read the input on analog pin 0:
sensorValue = analogRead(A1);
// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
float voltage = sensorValue * (5.0 / 1023);
// print out the value you read:
Serial.println(voltage);
float x = voltage;
float  p1 =   -11.29;
float  p2 =     67.36;
float  p3 =     -145.4;
float  p4 =    131.9 ;
float distance= p1*pow(x,3) + p2*pow(x,2) + p3*pow(x,1) + p4;  delay(1000);
// delay in between reads for stability  lcd.setCursor ( 1, 1 );
lcd.print(distance); }
```

Actual distance (cm)	LCD distance (cm)
15	**14.61**
20	**18.69**
25	**22.00**
30	**28.02**
35	**33.17**
40	**39.79**
45	**47.03**
50	**33.99**
55	**60.67**

Conclusion:

The optical displacement sensor is found to be an adequate source for measuring distance between two points.

The output voltage changed for the different material being used for reflecting the optical beam. This shows that the sensor depends on the material used as the reflecting plate and needs to be calibrated accordingly.

The sensor showed an approximate linear increase within distances very close between sensor and reflecting plate. After a certain point reaching the maximum value it starts to decrease for a larger distance. As the distance for the increase is very small, 12 cm max, the sensor is considered inoperative in this region and the minimum distance is calculated for measurement.

The sensor is highly sensitive to reflection angle. Therefore care must be taken to keep the reflecting plate at the same angle while traversing along the distance. A very high change in voltage values and this the calibrated distance was observed and recorded because the reflecting plate was tilting during translation.

The curve fitting for the values of voltage against distance used for the calibration of the sensor show a quadratic behaviour and the graph is 99% accurate at the recorded points. For unrecorded points, there degree of error is higher which is reflected in the results.

The actual distance agree with the calibration.

References:

1. http://modernroboticsinc.com/Content/Images/uploaded/Sensors/Sensor%20Documentation.pdf

2. https://www.sparkfun.com/products/242