

# **DISTANCE MEASUREMENT OVER NON LINEAR** **SURFACES**

**Internet of Things ( IoT )**

CS - 744

*A project submitted in partial fulfilment of the requirements*

*for the award of the degree of*

**Bachelor of Technology**

*in*

**Computer Science and Engineering (Dual Degree)**

*by*

**Avin Rajan (20DCS020)**



**Submitted to:**

**Dr. Robin Singh Bhadoria**

*Department of Computer Science and Engineering*  
*National Institute of Technology Hamirpur, (HP)*

# ACKNOWLEDGEMENT

The successful completion of this project owes much to Dr. Robin Singh Bhadoria's expertise in Internet of Things technology. His guidance and teachings were crucial in shaping the project's direction and ensuring its success. I'm grateful for the opportunity to learn from him and for his significant contribution to this project. I am grateful for the chance to benefit from his expertise and thankful for his substantial contributions to this endeavor.

# INTRODUCTION

This IoT Project addresses a critical need in the automotive industry by leveraging IoT capabilities to revolutionize odometer tracking and management. Traditional odometer systems often suffer from inaccuracies, limited functionalities, and lack of real-time monitoring capabilities. Our project seeks to overcome these challenges by harnessing the power of IoT to create a robust and efficient odometer tracking solution.

# OBJECTIVE

The objective of this Project is to revolutionize the way vehicle mileage is tracked and managed by harnessing the power of IoT technology. Through the development of a comprehensive solution, the aim is to provide accurate and real-time monitoring of odometer readings, ensuring precision and reliability in mileage tracking. By leveraging advanced sensors and data analytics, we seek to minimize errors and inaccuracies inherent in traditional odometer systems, thereby improving overall efficiency and effectiveness. Additionally, our objective includes the optimization of maintenance scheduling through the utilization of odometer data insights, enabling proactive maintenance practices based on actual vehicle usage. Furthermore, data security is a paramount concern, and our objective encompasses the implementation of robust security measures to safeguard odometer data against unauthorized access and ensure compliance with privacy regulations.

# SIMULATOR USED

## SimulIDE

SimulIDE is a real-time electronic circuit simulation environment with PIC, AVR, and Arduino simulation. It has a spartan interface, aiming to be fast, simple, and easy to use .

You can design your own circuits and program them with a code editor and debugger for GcBasic, Arduino, PIC, and AVR.

Simulation speed is one of the most relevant characteristics of this simulator. It has been well optimized to achieve excellent speeds and low cpu usage.

# COMPONENTS USED

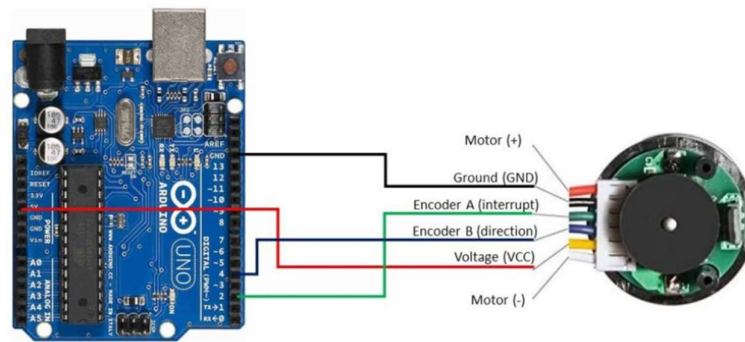
1. Arduino UNO R3



2. LCD Display



### 3. DC Encoder motor



### 4. Encoder Monitor along with positional leds



### 5. Wheel Radius and Torque adjustment devices



# CONTROLS

- **START/STOP]:**

To start or stop the measurement of distance.

The wheel will rotate if the <PAUSE> switch and the power supply have a voltage between 0 and 5 volts to simulate the measurement speed.

The degree LEDs will begin to move at the rate of rotation of the measuring wheel marking each of the quarters of the wheel where the measurement is taken, these ranges are: 0 to 89 degrees, 90 to 179, 180 to 269 and 270 to 359.

- **[RESET]** In STOP mode the odometer counter can be reset.
- **ADJUSTMENT POTENTIOMETER.** In STOP mode, the wheel radius can be adjusted from 0 to 0.99 meters. The data is displayed on the OLED.
- **MEASURING WHEEL:** It is associated with the motor shaft that moves the potentiometer of the internal encoder of the DC motor.



The screenshot displays the Proteus software interface for a circuit simulation. The main workspace shows a schematic diagram of a distance measurement system. The components include an Arduino Uno microcontroller, an SSD1306 display module, an encoder motor, and a rotation speed display. The display shows 'WHEEL DISTANCE: 55.1 m'. The rotation speed display shows '1.82 V'. The simulation is running, and the rotation speed is 1.82 V. The interface includes a top toolbar, a left sidebar with component libraries, and a bottom status bar.

**Left Sidebar (Component Libraries):**

- Components:**
  - Resistor
  - Capacitor
  - Inductor
  - Diode
  - Transistor
  - IC
  - LED
  - Display
  - Motor
  - Other Outputs
  - Micro
- Switches:**
  - Push
  - Switch (all)
  - Switch Dip
  - Relay (all)
  - Keypad
- Passive:**
  - Resistors
  - Resistive Sensors
  - Reactive
- Active:**
  - Resistors
  - Resistive Sensors
  - Reactive
- Other Active:**
  - Resistors
  - Resistive Sensors
  - Reactive

**Simulation Results:**

Simulation Time: 00:01:41 x 099 ms 164 ps 625 ns 0

Target Speed: 100.00 %

Real Speed: 099.83 %

Engine Load: 033.33 %

Update Load: 016.45 %

Starting Circuit Simulation...

Simulation Running...

**Code Snippets:**

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH,
SCREEN_HEIGHT, &Wire, 1);
int pinD0 = 2;
int pinD1 = 3;
int pinD2 = 4;
int pinD3 = 5;
int pinD4 = 6;
int pinD5 = 7;
int pinD6 = 8;
int pinD7 = 9;
int pinD8 = 10;
int pinD9 = 11;
int pinD10 = 12;
int pinD11 = 13;
int pinD12 = 14;
int pinD13 = 15;
int pinD14 = 16;
int pinD15 = 17;
int pinD16 = 18;
int pinD17 = 19;
int pinD18 = 20;
int pinD19 = 21;
int pinD20 = 22;
int pinD21 = 23;
int pinD22 = 24;
int pinD23 = 25;
int pinD24 = 26;
int pinD25 = 27;
int pinD26 = 28;
int pinD27 = 29;
int pinD28 = 30;
int pinD29 = 31;
int pinD30 = 32;
int pinD31 = 33;
int pinD32 = 34;
int pinD33 = 35;
int pinD34 = 36;
int pinD35 = 37;
int pinD36 = 38;
int pinD37 = 39;
int pinD38 = 40;
int pinD39 = 41;
int pinD40 = 42;
int pinD41 = 43;
int pinD42 = 44;
int pinD43 = 45;
int pinD44 = 46;
int pinD45 = 47;
int pinD46 = 48;
int pinD47 = 49;
int pinD48 = 50;
int pinD49 = 51;
int pinD50 = 52;
int pinD51 = 53;
int pinD52 = 54;
int pinD53 = 55;
int pinD54 = 56;
int pinD55 = 57;
int pinD56 = 58;
int pinD57 = 59;
int pinD58 = 60;
int pinD59 = 61;
int pinD60 = 62;
int pinD61 = 63;
int pinD62 = 64;
int pinD63 = 65;
int pinD64 = 66;
int pinD65 = 67;
int pinD66 = 68;
int pinD67 = 69;
int pinD68 = 70;
int pinD69 = 71;
int pinD70 = 72;
int pinD71 = 73;
int pinD72 = 74;
int pinD73 = 75;
int pinD74 = 76;
int pinD75 = 77;
int pinD76 = 78;
int pinD77 = 79;
int pinD78 = 80;
int pinD79 = 81;
int pinD80 = 82;
int pinD81 = 83;
int pinD82 = 84;
int pinD83 = 85;
int pinD84 = 86;
int pinD85 = 87;
int pinD86 = 88;
int pinD87 = 89;
int pinD88 = 90;
int pinD89 = 91;
int pinD90 = 92;
int pinD91 = 93;
int pinD92 = 94;
int pinD93 = 95;
int pinD94 = 96;
int pinD95 = 97;
int pinD96 = 98;
int pinD97 = 99;
int pinD98 = 100;
int pinD99 = 101;
int pinD100 = 102;
int pinD101 = 103;
int pinD102 = 104;
int pinD103 = 105;
int pinD104 = 106;
int pinD105 = 107;
int pinD106 = 108;
int pinD107 = 109;
int pinD108 = 110;
int pinD109 = 111;
int pinD110 = 112;
int pinD111 = 113;
int pinD112 = 114;
int pinD113 = 115;
int pinD114 = 116;
int pinD115 = 117;
int pinD116 = 118;
int pinD117 = 119;
int pinD118 = 120;
int pinD119 = 121;
int pinD120 = 122;
int pinD121 = 123;
int pinD122 = 124;
int pinD123 = 125;
int pinD124 = 126;
int pinD125 = 127;
int pinD126 = 128;
int pinD127 = 129;
int pinD128 = 130;
int pinD129 = 131;
int pinD130 = 132;
int pinD131 = 133;
int pinD132 = 134;
int pinD133 = 135;
int pinD134 = 136;
int pinD135 = 137;
int pinD136 = 138;
int pinD137 = 139;
int pinD138 = 140;
int pinD139 = 141;
int pinD140 = 142;
int pinD141 = 143;
int pinD142 = 144;
int pinD143 = 145;
int pinD144 = 146;
int pinD145 = 147;
int pinD146 = 148;
int pinD147 = 149;
int pinD148 = 150;
int pinD149 = 151;
int pinD150 = 152;
int pinD151 = 153;
int pinD152 = 154;
int pinD153 = 155;
int pinD154 = 156;
int pinD155 = 157;
int pinD156 = 158;
int pinD157 = 159;
int pinD158 = 160;
int pinD159 = 161;
int pinD160 = 162;
int pinD161 = 163;
int pinD162 = 164;
int pinD163 = 165;
int pinD164 = 166;
int pinD165 = 167;
int pinD166 = 168;
int pinD167 = 169;
int pinD168 = 170;
int pinD169 = 171;
int pinD170 = 172;
int pinD171 = 173;
int pinD172 = 174;
int pinD173 = 175;
int pinD174 = 176;
int pinD175 = 177;
int pinD176 = 178;
int pinD177 = 179;
int pinD178 = 180;
int pinD179 = 181;
int pinD180 = 182;
int pinD181 = 183;
int pinD182 = 184;
int pinD183 = 185;
int pinD184 = 186;
int pinD185 = 187;
int pinD186 = 188;
int pinD187 = 189;
int pinD188 = 190;
int pinD189 = 191;
int pinD190 = 192;
int pinD191 = 193;
int pinD192 = 194;
int pinD193 = 195;
int pinD194 = 196;
int pinD195 = 197;
int pinD196 = 198;
int pinD197 = 199;
int pinD198 = 200;
int pinD199 = 201;
int pinD200 = 202;
int pinD201 = 203;
int pinD202 = 204;
int pinD203 = 205;
int pinD204 = 206;
int pinD205 = 207;
int pinD206 = 208;
int pinD207 = 209;
int pinD208 = 210;
int pinD209 = 211;
int pinD210 = 212;
int pinD211 = 213;
int pinD212 = 214;
int pinD213 = 215;
int pinD214 = 216;
int pinD215 = 217;
int pinD216 = 218;
int pinD217 = 219;
int pinD218 = 220;
int pinD219 = 221;
int pinD220 = 222;
int pinD221 = 223;
int pinD222 = 224;
int pinD223 = 225;
int pinD224 = 226;
int pinD225 = 227;
int pinD226 = 228;
int pinD227 = 229;
int pinD228 = 230;
int pinD229 = 231;
int pinD230 = 232;
int pinD231 = 233;
int pinD232 = 234;
int pinD233 = 235;
int pinD234 = 236;
int pinD235 = 237;
int pinD236 = 238;
int pinD237 = 239;
int pinD238 = 240;
int pinD239 = 241;
int pinD240 = 242;
int pinD241 = 243;
int pinD242 = 244;
int pinD243 = 245;
int pinD244 = 246;
int pinD245 = 247;
int pinD246 = 248;
int pinD247 = 249;
int pinD248 = 250;
int pinD249 = 251;
int pinD250 = 252;
int pinD251 = 253;
int pinD252 = 254;
int pinD253 = 255;
int pinD254 = 256;
int pinD255 = 257;
int pinD256 = 258;
int pinD257 = 259;
int pinD258 = 260;
int pinD259 = 261;
int pinD260 = 262;
int pinD261 = 263;
int pinD262 = 264;
int pinD263 = 265;
int pinD264 = 266;
int pinD265 = 267;
int pinD266 = 268;
int pinD267 = 269;
int pinD268 = 270;
int pinD269 = 271;
int pinD270 = 272;
int pinD271 =
```

# CONCLUSION

We have demonstrated the integration of a distance measuring system using ArduinoUno and DC encoder motor within the SIMULIDE simulation environment. By using encoder monitors and adjustment devices we have showcased potential for measuring physical characteristics accurately and in remote areas without human intervention.

Moving forward, further enhancements and real world implementations can be explored to refine the system's functionality and accessibility.

## FUTURE WORK

For future work, the IoT Project could focus on integrating machine learning algorithms for improved accuracy and predictive maintenance insights. Expanding compatibility, exploring blockchain for enhanced security, and implementing anomaly detection mechanisms are also key areas. Additionally, integrating environmental data and optimizing scalability and efficiency would further enhance the platform's capabilities and competitiveness in the automotive industry.

## REFERENCES

1. [https://simulide.com/p/long-distance-measurement\\_df353/https://www.youtube.com/watch?v=ZxQf2L-UWEA](https://simulide.com/p/long-distance-measurement_df353/https://www.youtube.com/watch?v=ZxQf2L-UWEA)
2. <https://simulide.com/p/>
3. <https://forum.arduino.cc/t/programing-motor-with-encoder/962793>