

SENSOR LIDAR

El nombre de LIDAR en realidad es un acrónimo que podría traducirse como sistema de medición y detección de objetos mediante láser. Los LIDAR son sensores que emiten rayos de luz. Cuando ese rayo de luz toca un objeto rebota y vuelve a la posición en la que se encuentra el sensor. Calculando el tiempo que tarda el rayo láser en ir y volver se puede descifrar la distancia que hay entre el sensor y cualquier objeto. Para eso es necesario contar con el emisor de rayos láser infrarrojos, una lente que recoja los haces de luz cuando reboten y un chip o sistema que procese todos esos datos hasta construir un mapa 3D de la escena que hay delante del sensor. El sensor mide el cambio de fase entre las señales transmitidas y reflejadas.

La longitud de onda de la señal de modulación obedece a la ecuación:

$$C = f \cdot \tau$$

Donde c es la velocidad de la luz, f la frecuencia de modulación y τ la longitud de onda de modulación conocida.

La distancia total D' cubierta por la luz emitida es:

$$D' = B + 2A = B + (\theta * \tau) / 2\pi$$

Donde A es la distancia medida y B es la distancia desde la unidad de medida de fase. Por lo tanto, la distancia requerida D , entre el divisor de haz y el objetivo, viene dada por:

$$D = \tau * \theta / 4\pi$$

Donde θ es la diferencia de fase medida electrónicamente entre los haces de luz transmitidos y reflejados.

El LIDAR que utilizaremos será el YDLIDAR X4. Es un escáner de rango láser bidimensional de 360 grados (LIDAR). Este dispositivo utiliza el principio de triangulación para medir la distancia, junto con el diseño de algoritmo óptico, eléctrico y apropiado, para lograr una medición de distancia de alta precisión.

Su rango va desde los 0.12 metros hasta los 3. Tiene una resolución menor que 0.5 mm para un rango menor a 2 m, y de un 1% mejor que la distancia medida para cifras superiores a 2 m.


```

10 #include <Wire.h>
11 #include <VL53L0X.h> //Download it here: https://www.electrooobs.com/eng_arduino_Adafruit_VL53L0X.php
12
13
14 VL53L0X sensor; //Define our sensor
15
16 #define LONG_RANGE
17 #define HIGH_SPEED
18 // #define HIGH_ACCURACY
19
20
21 //Outputs/inputs
22 #define dirPin 3 //Pin para la dirección del conductor paso a paso.
23 #define stepPin 4 //Pin para pasos del conductor paso a paso.
24 #define Enable 5 //Pin para habilitar el controlador paso a paso.
25
26
27 //Variables
28 int Value = 1200; //Valor de retraso entre pasos.
29 float angle = 0; //Empezar ángulo.
30
31 /* -----Cálculo de ángulo de paso-----
32 * Necesitamos 1.5 rotaciones para 360°. (relación de pulso 1.5: 1)
33 * Cada 200 pasos, el motor hará una rotación.
34 * Nos movemos 2 pasos y luego hacemos una medición.
35 * Esto equivale a 360° / (200 pasos * 1.5) * 2 = 2.4 ángulo / bucle ->
36 -----Cálculo de ángulo de paso-----*/
37 float angle_step = 2.4; //Así que pon ese valor aquí
38
39 float maxdist = 400; //He establecido la distancia máxima alrededor del sensor a solo 400 mm. Cambiar a cualquier otro valor.
40 bool loop_starts = false;
41 byte last_PIN_state;
42
43
44 void setup() {
45 // Declarar pines como salida:
46 pinMode(stepPin, OUTPUT);
47 pinMode(dirPin, OUTPUT);
48 pinMode(Enable, OUTPUT);
49 digitalWrite(Enable, LOW); //Habilite a bajo para que el controlador esté habilitado
50 digitalWrite(dirPin, HIGH); //Coloque dirPin en ALTO para que giremos en sentido horario
51 Serial.begin(9600); //Iniciar puerto serie
52
53 Wire.begin();
54 sensor.init();
55 sensor.setTimeout(500);
56 PCICR |= (1 << PCIE0); //habilite el escaneo PCMSK0 para que podamos usar interrupciones
57 PCMSK0 |= (1 << PCINT0); //Establecer pin "D8" desencadenar una interrupción en "cualquier" cambio de estado.
58 //Ver el vector de interrupción debajo del bucle vacío
59
60
61 #if defined LONG_RANGE
62 // baja el límite de velocidad de la señal de retorno (el valor predeterminado es 0.25 MCPS)
63 sensor.setSignalRateLimit(0.1);
64 // aumentar los periodos de pulso láser (los valores predeterminados son 14 y 10 PCLK
65 sensor.setVcselPulsePeriod(VL53L0X::VcselPeriodPreRange, 18);
66 sensor.setVcselPulsePeriod(VL53L0X::VcselPeriodFinalRange, 14);
67 #endif
68 #if defined HIGH_SPEED
69 // reduce el presupuesto de tiempo a 20 ms (el valor predeterminado es aproximadamente 33 ms)
70 sensor.setMeasurementTimingBudget(20000);
71 #elif defined HIGH_ACCURACY
72 // // aumentar el presupuesto de tiempo a 200 ms
73
74 // sensor.setMeasurementTimingBudget(200000);
75 #endif
76 } //Fin del bucle de configuración
77
78
79
80
81
82 void loop() {
83 if (loop_starts) //Restablecemos el ángulo cuando se detecta el imán en D8
84 {
85 angle = 0;
86 loop_starts = false;
87 }
88
89 digitalWrite(stepPin, HIGH); //Haz un paso
90 delayMicroseconds(Value); //Un pequeño retraso
91 digitalWrite(stepPin, LOW); //Haz otro paso
92 delayMicroseconds(Value); //Made otro retraso
93 int r = sensor.readRangeSingleMillimeters(); //Coger la distancia del sensor
94 if (r > maxdist) //Limit the distance to maximum set distance above
95 {
96 r = maxdist;
97 }
98 Serial.print(angle); //Imprima los valores en el puerto serie
99 Serial.print(",");
100 Serial.print(r);
101 Serial.println(",");
102
103 angle = angle + angle_step; //Aumente el valor del ángulo por el valor del ángulo / bucle establecido anteriormente (en este caso 2.4° por bucle)
104 } //end of void loop
105

```