Introduction

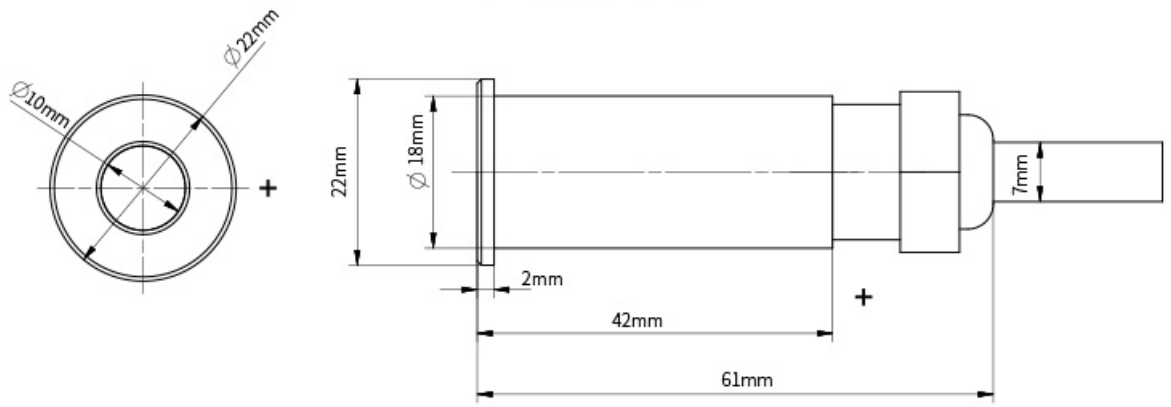
This laser range sensor can be used to detect objects in 4~400cm with ±2cm accuracy. With 3 measurement modes supported, it is well applicable to various detection scenarios like security gate detection, access control systems, security alarm devices, smart trash cans, smart cars or robots for obstacle avoidance.
The sensor metal black shell adopts a threaded barrel design and its probe comes with an optical cover. The integrated design makes the product able to effectively filter optical interference, waterproof and shockproof. The sensor uses invisible laser and there is a voltage regulator circuit inside the module. Featuring stable performance and UART output, the sensor has an alarm output line that will be triggered to output Low steadily when the measurement distance is smaller than the user-set threshold.

```
Note: The data will be unstable when measuring black objects.
```

Specification

- Power Supply Voltage: 6-36V
- Measuring Distance: 4-400cm
- Measurement Accuracy: ±2cm
- Launch Angle: 39.6°
- Receiving Angle: 36.5°
- Working Current: <38mA
- Communication Interface: UART
- Waterproof Rating: IP67
- Baud Rate: 2400-921600 optional 115200 (default)
- Frequency: 20Hz (default)
- Working Temperature: -20~70°C
- Size: 21.5x21x8 mm / 0.85×0.83×0.31 inch

Dimension Figure



Board Overview

Label	Name	Description
Red	VCC	6-36V power supply
Green	RX	UART receiving data line
Yellow	TX	UART transmitting data line line
Black	GND	Power ground
White	ALARM	Police connection line

Tutorial

Requirements

- **Hardware**
  - Arduino UNO x 1
  - 6-in-1 multi-function to serial port module (https://www.dfrobot.com/product-2291.html) or FT232 USB to TTL Serial Cable (https://www.dfrobot.com/product-1277.html)

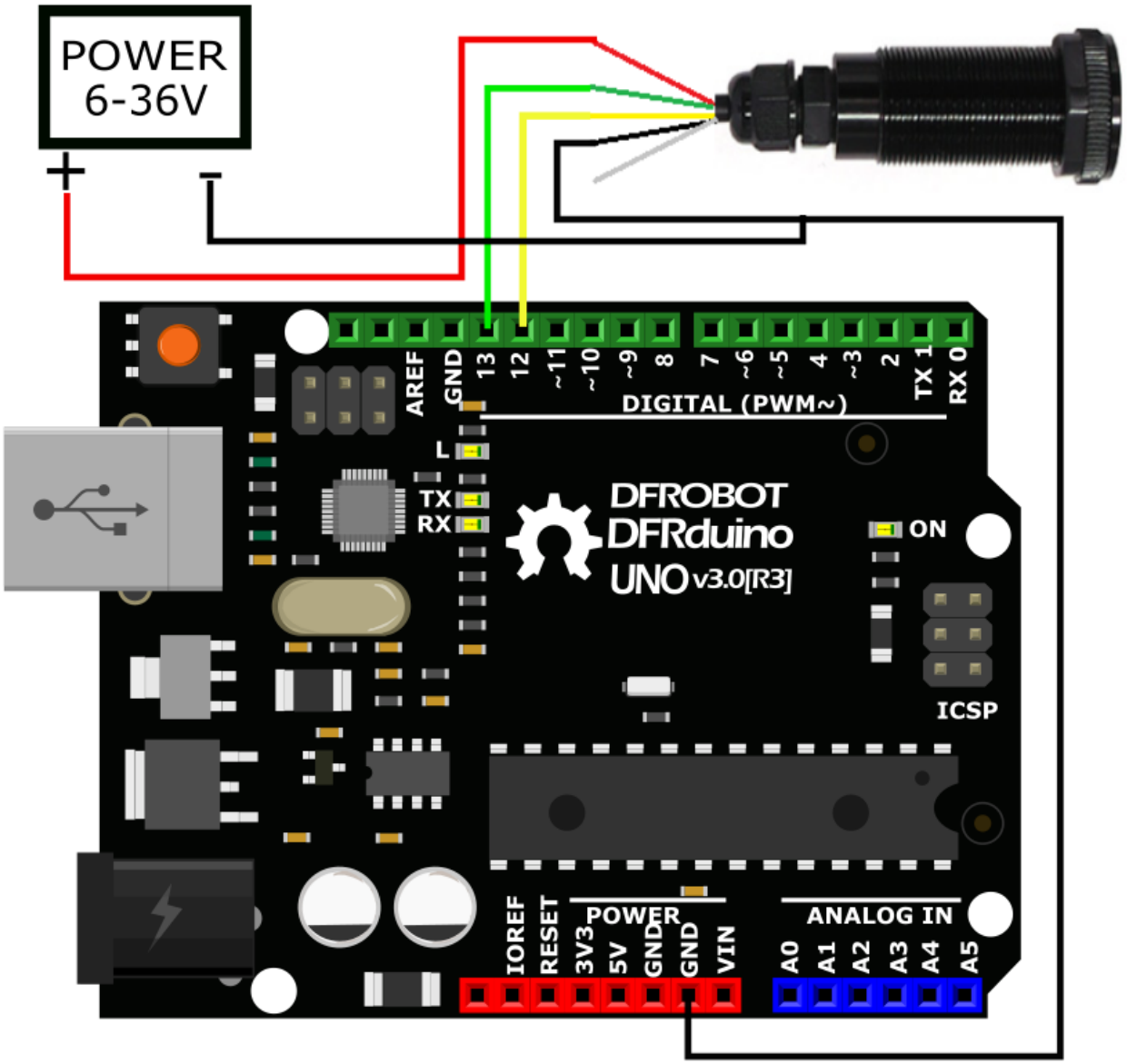
1277.html)

- Laser Ranging Sensor UART 4m × 1

- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)

Usage for Arduino

Connection Diagram



Sample Code

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(12, 13); //Define soft serial port, port 13 is TX, port 12 
uint8_t Data[34] = {0};

void setup()
{
  Serial.begin(115200);
  mySerial.begin(115200);
}


void loop()
{

  readData(Data);
  int Distance = readD(Data);
  if (Distance >= 0) {
    Serial.print("Distance: "); Serial.print(Distance); Serial.println(" mm");
    Serial.print("\n");
  } else {
    Serial.println("Invalid data");
    Serial.print("\n");
  }
  delay(500);
}

int readD(uint8_t *buf)
{
  int d;
  char *p = strstr(buf, "Range Valid");
  if (p != 0) {
    d = atoi(&Data[25]);
    return d;
  } else {
    return -1;
  }
}

void readData(uint8_t *buf)
{
  bool flag = 0;
  uint8_t ch;
  while (!flag) {
    if (readN(&ch, 1) == 1) {
      if (ch == 'S') {
        Data[0] = ch;
```

```
        if (readN(&ch, 1) == 1) {
          if (ch == 't') {
            Data[1] = ch;
            if (readN(&ch, 1) == 1) {
              if (ch == 'a') {
                Data[2] = ch;
                if (readN(&Data[3], 30) == 30 && Data[31] == 'm' && Data[32] == 'm'
                  flag = 1;
                }
              }
            }
          }
        }
      }
    }
  }
}
int readN(uint8_t *buf, size_t len)
{
  size_t offset = 0, left = len;
  long curr = millis();
  while (left) {
    if (mySerial.available()) {
      buf[offset++] = mySerial.read();
      left--;
    }
    if (millis() - curr > 500) {
      break;
    }
  }
  return offset;
}
```

Usage for Serial Port Assistant

Connection Diagram

Turn the mode of 6-in-1 multi-function to serial port module to USB to TTL, as shown below:

Mode	Dial code 1 (USB)	Dial code 2 (485)	Switch S1	Diagram
USB to TTL	ON	OFF	Down (232-485)	

Connect the sensor to 6-in-1 multi-function module, then plug the module into the usb

port of computer, open the serial assistant, select Baud rate 1152000, untick HEX display and open the serial port, it will directly output distance value as shown below.

```
[17:22:16.494]收←◆State:0 , Range Valid
d:  255 mm

[17:22:16.585]收←◆State:0 , Range Valid
d:  255 mm

[17:22:16.694]收←◆State:0 , Range Valid
d:  250 mm

[17:22:16.785]收←◆State:0 , Range Valid
d:  250 mm

[17:22:16.893]收←◆State:0 , Range Valid
d:  249 mm

[17:22:16.984]收←◆State:0 , Range Valid
d:  249 mm

[17:22:17.094]收←◆State:0 , Range Valid
d:  251 mm

[17:22:17.185]收←◆State:0 , Range Valid
d:  251 mm

[17:22:17.293]收←◆State:0 , Range Valid
d:  252 mm
```

Register Table

Register name	Register address	Command	Description
System recovery	0x00	MODADDR 06 00 00 00 01 CRCH CRCL	Write 0x01, the sensor restores the default setting
Alarm threshold	0x02	MODADDR 06 00 03MH ML CRCH CRCL	MH alarm threshold high bit, ML alarm threshold low bit, threshold setting range 40~4000mm
Baud rate setting	0x04	MODADDR 06 00 04 00 00 CRCH CRCL	Write 0x00, baud rate 2400
		MODADDR 06 00 04 00 01 CRCH CRCL	Write 0x01, baud rate 4800
		MODADDR 06 00 04 00 02 CRCH CRCL	Write 0x02, the baud rate is 9600
		MODADDR 06 00	

		MODADDR 06 00 04 00 03 CRCH CRCL	Write 0x03, the baud rate is 19200

Register name	Register address	Command	Description
		MODADDR 06 00 04 00 04 CRCH CRCL	Write 0x04, the baud rate is 38400
		MODADDR 06 00 04 00 05 CRCH CRCL	Write 0x05, baud rate 57600
		MODADDR 06 00 04 00 06 CRCH CRCL	Write 0x06, baud rate 115200
		MODADDR 06 00 04 00 07 CRCH CRCL	Write 0x07, baud rate 230400
		MODADDR 06 00 04 00 08 CRCH CRCL	Write 0x08, baud rate 460800
		MODADDR 06 00 04 00 09 CRCH CRCL	Write 0x09, the baud rate is 921600
Timing preset time (not recommended to modify, default 200MS)	0x07	MODADDR 06 00 07 TIMEBUDGETH	TIMEBUDGET: 20-1000 ms, can be changed to 0x0014-0x03e8
Measurement interval (not recommended to modify, default 50MS)	0x08	MODADDR 06 00 08 PERIODH PERIODL CRCH CRCL	PERIOD： 1-1000 ms, can be changed to 0x0001-0x03e8
ID setting	0x1A	MODADDR 06 00 1a 00 MODADDRL CRCH CRCL	Can write 0x00~0xFE
		MODADDR 03 00	Read, the upper 8 bits of

Measurement data	0x34	34 00 01 CRCH CRCL	the distance to the lower 8 bits of the distance

Register name	**Register address**	**Command**	**Description**
Output state	0x35	MODADDR 03 00 35 00 01 CRCH CRCL	Read: 0x07, sensor No Update
			Read: 0x00, Sensor Range Valid
			Read: 0x01， Sensor Sigma Fail
			Read: 0x02, Sensor Signal Fail
			Read: 0x03, Sensor Min Range Fail
			Read: 0x04, Sensor Phase Fail
			Read: 0x05, Sensor Hardware Fail
Measurement mode	0x36	MODADDR 06 00 36 00 01 CRCH CRCL	Write 0x01, short distance (up to 1.3m, better environmental immunity)
		MODADDR 06 00 36 00 02 CRCH CRCL	Write 0x02, middle distance (up to 3 meters)
		MODADDR 06 00 36 00 03 CRCH CRCL	Write 0x03, long distance mode (up to 4 meters)
Calibration mode	0x37	MODADDR 06 00 37 00 04 CRCH CRCL	Write 0x04 to enter the calibration state
		MODADDR 03 00 37 00 01 CRCH	Read: 0x01, start

		CRCL	calibration

Register name	Register address	Command	Description
		MODADDR 03 00 37 00 02 CRCH CRCL	Read: 0x02, calibration failed
		MODADDR 03 00 37 00 03 CRCH CRCL	Read: 0x03, calibration is complete