Lesson 3 Obstacle Avoidance

CONTENT

Ⅰ. Brief Introduction

In this chapter you will learn the principle of ultrasonic ranging, how to use infrared photoelectric sensor and how to program Penguin Bot to realize obstacle avoidance by using ***ultrasonic ranging and optoelectronic switch together***.

Ⅱ. Principle of Obstacle Avoidance

Principle of Ultrasonic Ranging

HC-SR04 Ultrasonic Sensor Module

Feature of the module: testing distance, high precision module.
Application of the products: robot obstacle avoidance、object testing distance、liquid testing、public security、parking lot testing.
Main technical parameters:

- voltage used: DC---5V
- static current: less than 2mA
- level output: higher than 5V

Tips: If you have any questions or run into any problems during assembling and testing Penguin Bot please feel free to contact us at service@elegoo.com or euservice@elegoo.com (Europe customers).

- level output: lower than 0
- detection angle: not bigger than 15 degree
- detecting distance: 2cm-450cm
- high precision: up to 0.2cm

Method of connecting lines: VCC, trig (the end of controlling), echo (the end of receiving), GND

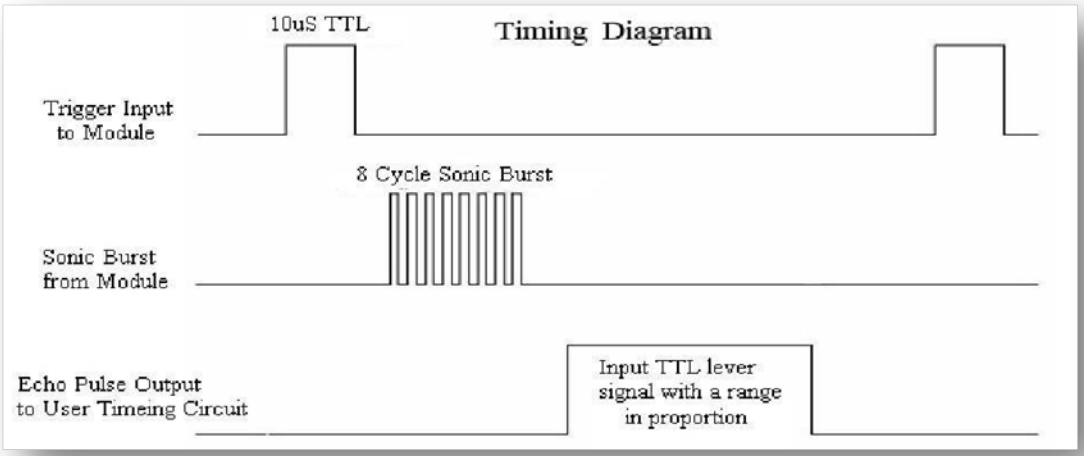
How does the module work:

- Apply IO port of TRIG to trigger ranging, give high level signal, at least 10us one time;
- The module sends 8 square waves of 40kz automatically, tests if there are signals returned automatically;
- If there are signals received, the module will output a high level pulse through IO port of ECHO, the duration time of high level pulse is the time between the wave sending and receiving. So the module can know the distance according to the time.

Testing distance= (high level time* velocity of sound (340M/S))/2);

Actual operation:

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



How to Use

Connect the ultrasonic sensor with NANO R3 Board from pin ECHO to D4, pin TRIG to D5 and pin VCC to GND. Then use get Distance Function in the program to obtain distance data. The unit of ranging data is cm.

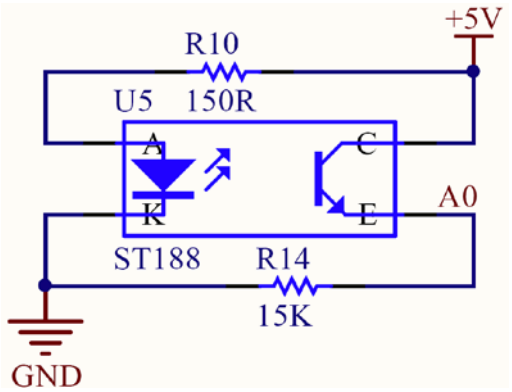
```
#define ECHO_PIN 4
#define TRIG_PIN 5
int getDistance() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  return (int)pulseIn(ECHO_PIN, HIGH) / 58;
}
```

Principle of Infrared Sensor ST188

Features of ST188

- Consists of high-transmitting-power infrared photodiode and high-sensitivity phototransistor
- Detecting distance: 4-13mm

Circuit Diagram



A, K are the positive and negative anodes of infrared emitting diode, C, E are the positive and negative anodes of infrared receiving diode. Polarity A connects to high level and Polarity K connects to low level then infrared emitting diode can transmit infrared.

When there is no reflect of infrared, Polarity C and E will be cut off and no current flows, then Polarity E is low level. When there is reflect of infrared, Polarity C and E will be connected and current flows, then Polarity E is low level.

ST188 Infrared Sensor Application

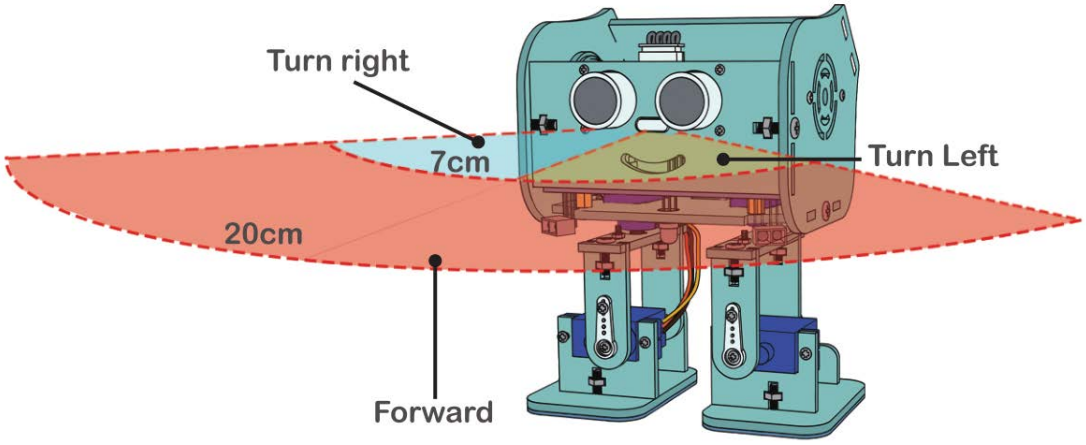
On the main board of Penguin Bot, connect Polarity E of the right ST188 to Pin A1 of UNO board and connect Polarity E of the left ST188 to Pin A0 of UNO board, by measuring voltage from Pin ADC of UNO board to Polarity E Penguin Bot can detect whether there are obstacles in front.

Using analogRead function in the program can read the data of ST188 which is from 0 to 1023.

```
#define ST188_R_PIN A1
#define ST188_L_PIN A0
int st188Val_L;
int st188Val_R;
st188Val_L = analogRead(ST188_L_PIN);
st188Val_R = analogRead(ST188_R_PIN);
```

Principle of Obstacle Avoidance

Ultrasonic sensor detects the distance of the obstacle in front and ST188 detects where there obstacles on the left and right side. If the obstacle is very near then Penguin Bot will make a turn or walk backwards. Otherwise it will continue walking forwards.



Ⅲ. Write Program of Obstacle Avoidance

The sketch used in this chapter is saved in below path and please refer to *Upload Penguin-Bot program* and upload the codes.

\ELEGOO Penguin Bot V2.0\Penguin Bot Function Introduction\ Lesson 3 Obstacle Avoidance\Obstacle\ Obstacle.ino

Code reviews

```
#include "Oscillator.h"
#include <Servo.h>

#define YL_PIN 10 // 3
#define YR_PIN 9  // 2
#define RL_PIN 12 // 1
#define RR_PIN 6  // 0
#define ECHO_PIN 4
#define TRIG_PIN 5
#define ST188_R_PIN A1
#define ST188_L_PIN A0
#define N_SERVOS 4

int distance;
int st188Val_L;
int st188Val_R;
int t = 495;
Oscillator servo[N_SERVOS];

void oscillate(int A[N_SERVOS], int O[N_SERVOS], int T, double
phase_diff[N_SERVOS]) {
  for (int i = 0; i < 4; i++) {
    servo[i].SetO(O[i]);
    servo[i].SetA(A[i]);
    servo[i].SetT(T);
    servo[i].SetPh(phase_diff[i]);
  }
  double ref = millis();
  for (double x = ref; x < T + ref; x = millis()) {
    for (int i = 0; i < 4; i++) {
      servo[i].refresh();
    }
  }
```

Define pins of

Define pins of ST188

Save the data of ultrasonic ranging

Save the data of ST188 sensor

```
}

void walk(int steps, int T, int dir) {
  int A[4] = {30, 30, 30, 30};
  int O[4] = {0, 0, 0, 0};
  double phase_diff[4] = {DEG2RAD(0), DEG2RAD(0), DEG2RAD(dir * 90),
                      DEG2RAD(dir * 90)};
  for (int i = 0; i < steps; i++)
    oscillate(A, O, T, phase_diff);
}

void turn(int steps, int T, int dir) {
  int A[4] = {30, 30, 0, 0};
  if (dir == 1) {
    A[2] = 30;
    A[3] = 10;
  } else {
    A[2] = 10;
    A[3] = 30;
  }
  int O[4] = {0, 0, 0, 0};
  double phase_diff[4] = {DEG2RAD(0), DEG2RAD(0), DEG2RAD(90),
DEG2RAD(90)};
  for (int i = 0; i < steps; i++)
    oscillate(A, O, T, phase_diff);
}

int getDistance() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  return (int)pulseIn(ECHO_PIN, HIGH) / 58;
}

void obstacleMode() {
  bool turnFlag = true;
  distance = getDistance();
  if(distance >= 1 && distance <= 300) {
    st188Val_L = analogRead(ST188_L_PIN);
    st188Val_R = analogRead(ST188_R_PIN);
    if(st188Val_L >= 1000 && st188Val_R >= 1000) {
      walk(3, t * 4, -1);
```

Write getDistance function to obtain the ultrasonic ranging data

Write obstacle avoidance function and write obstacle avoidance program in this

Obtain ultrasonic ranging data

Validate the ultrasonic ranging

Obtain ST188 sensor data

If obstacle is on the left and right side

```
      if(turnFlag) {
        turn(3, t * 4, 1);
      } else {
        turn(3, t * 4, -1);
      }
    } else if(st188Val_L >= 1000 && st188Val_R < 1000) {
      turnFlag = true;
      turn(3, t * 4, 1);
    } else if(st188Val_L < 1000 && st188Val_R >= 1000) {
      turnFlag = false;
      turn(3, t * 4, -1);
    } else if(st188Val_L < 1000 && st188Val_R < 1000) {
      if (distance < 5) {
        walk(3, t * 4, -1);
        if(turnFlag) {
          turn(3, t * 4, 1);
        } else {
          turn(3, t * 4, -1);
        }
      } else if (distance >= 5 && distance <= 20) {
        if(turnFlag) {
          turn(1, t * 4, 1);
        } else {
          turn(1, t * 4, -1);
        }
      } else {
        walk(1, t * 4, 1);
      }
    }
  }
}

void setup() {
  pinMode(ECHO_PIN, INPUT);
  pinMode(TRIG_PIN, OUTPUT);
  servo[0].attach(RR_PIN);
  servo[1].attach(RL_PIN);
  servo[2].attach(YR_PIN);
  servo[3].attach(YL_PIN);
}

void loop() {
  obstacleMode();
}
```

If obstacle is on the left side

If obstacle is on the right side

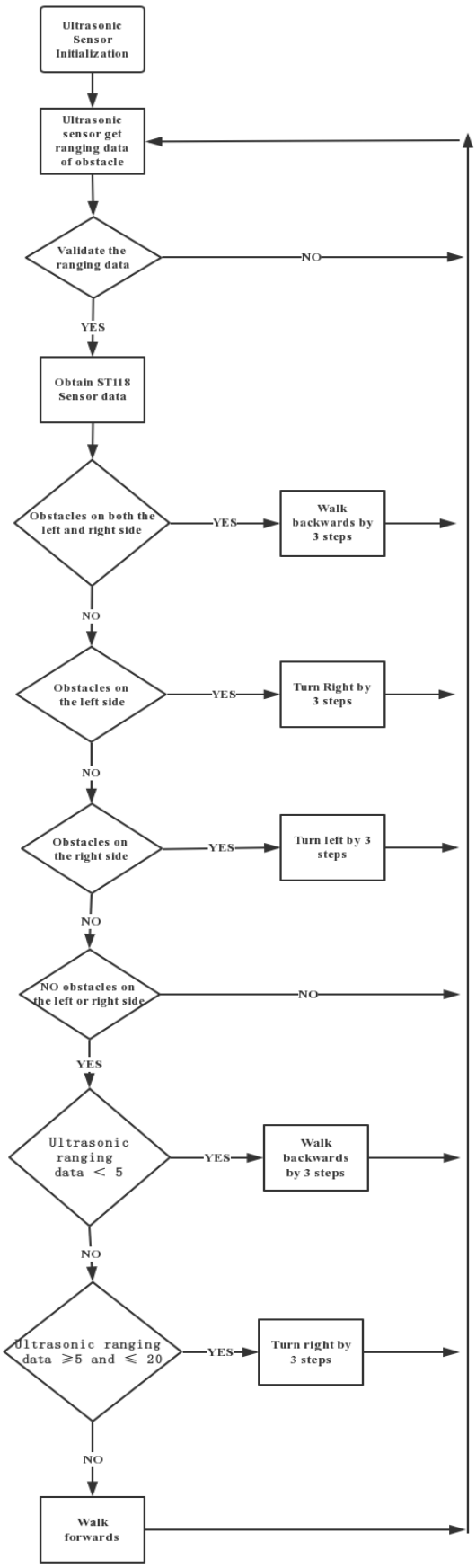
If no obstacle is detected

If detected obstacle is very near

Ultrasonic sensor initialization

Obstacle avoidance function

Workflow of Obstacle Avoidance



Tips: If you have any questions or run into any problems during assembling and testing Penguin Bot please feel free to contact us at service@elegoo.com or euservice@elegoo.com (Europe customers).

Ⅳ. Analysis of Obstacle Avoidance Function

From the workflow of obstacle avoidance we can see that after uploading the codes and turn on Penguin Bot, it will ***walk forwards until it come across obstacles*** (20cm in front) and then make a turn and if Penguin Bot detects no obstacles ahead it will ***keep walking forwards again***. If ST188 Sensor detects obstacles then Penguin Bot will make a turn or walk backwards.