**Tracking Sensor Module**

Overview

The purpose of this course is to use the the Raspberry Pi to capture the signal of the trackingsensor and controls the on and off of the LED lamp according to the captured signal.

**Experimental Materials**

Raspberry Pi \*1

Breadboard \*1

Tracking Sensor \*1

Led \*1

Dupont Line

**The Preparatory Work**

1. Install python interpreter in your Raspberry Pi system

2. Install the RPi.GPIO library in your Raspberry Pi system

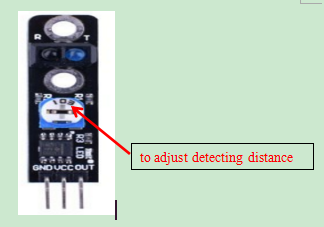
3. Install the wiringPi library in your Raspberry Pi system

For more details, please refer to the attachment for installing the python interpreter and corresponding libraries in the Raspberry Pi system.

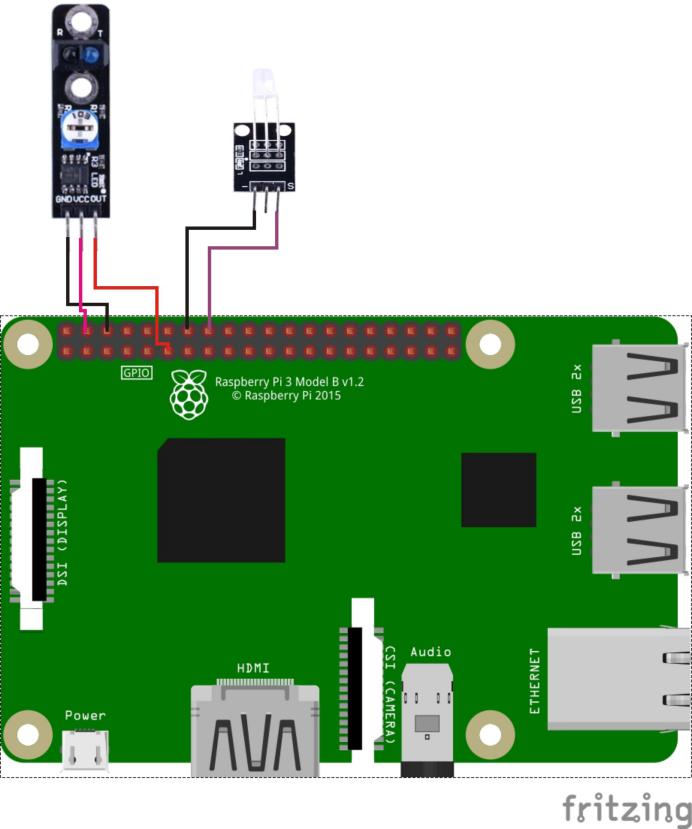
Product Description

Brief Introduction

The infrared tracking sensor is characterized by different reflection intensity on the surface of different colored objects and Infrared light is continuously emitted to the ground through the infrared emitting tube on the tracking module during the running of the car. Diffuse reflection occurs when infrared light encounters the white floor and reflected light is received by the infrared receiver on the patrol module. If the black line is encountered, infrared light is absorbed. So the infrared receiver on the patrol module receives no infrared signal. When the black line is not detected, the level of the OUT pin of the patrol module is low level. When the black line is detected, the level of the OUT pin of the tracking module is high level. So we can use the Raspberry Pi to detect the change in the level of this pin to make the corresponding action.



Wiring diagram



**The Sample Code**

1. **Python Code**

#!/usr/bin/env python

import RPi.GPIO as GPIO

TrackPin = 11

LedPin = 16

def setup():

GPIO.setmode(GPIO.BOARD) # Numbers GPIOs by physical location

GPIO.setup(LedPin, GPIO.OUT) # Set LedPin's mode is output

GPIO.setup(TrackPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

def loop():

while True:

if GPIO.input(TrackPin) == GPIO.LOW:

GPIO.output(LedPin, GPIO.LOW) # led on

else:

GPIO.output(LedPin, GPIO.HIGH) # led off

def destroy():

GPIO.output(LedPin, GPIO.HIGH) # led off

GPIO.cleanup() # Release resource

if \_\_name\_\_ == '\_\_main\_\_': # Program start from here

setup()

try:

loop()

except KeyboardInterrupt:

destroy()

1. **C Code**

#include <wiringPi.h>

#include <stdio.h>

#define TrackSensorPin 0

#define LedPin 4

int main(void)

{

if(wiringPiSetup() == -1)

{

printf("setup wiringPi failed !");

return 1;

}

pinMode(TrackSensorPin, INPUT);

pinMode(LedPin, OUTPUT);

while(1)

{

if(digitalRead(TrackSensorPin) == LOW)

{

digitalWrite(LedPin, LOW);

delay(100);

}

else

{

digitalWrite(LedPin, HIGH);

delay(100);

}

}

return 0;

}

**Experimental Phenomena**

When the line tracking module detects a black line, the LED lamp will light up. When the line tracking module does not detect a black line, the LED lamp will go out.