

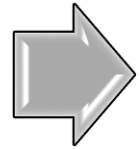
Color Detection Using Raspberry Pi and Python Animation Tools



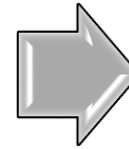
إعداد: م. علا جزماتي

System Diagram

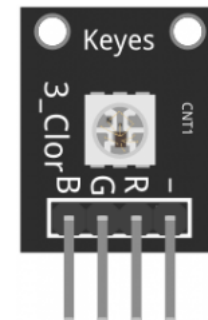
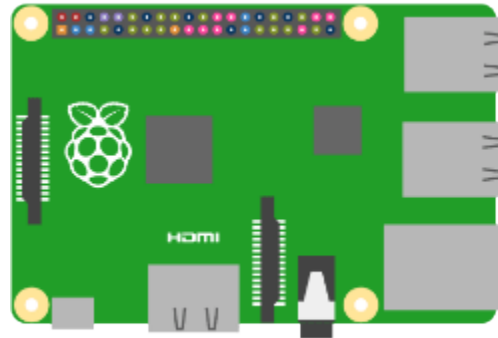
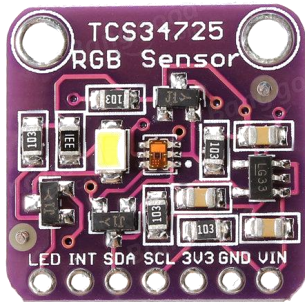
TCS34725 **RGB**
Color Sensor
with IR filter



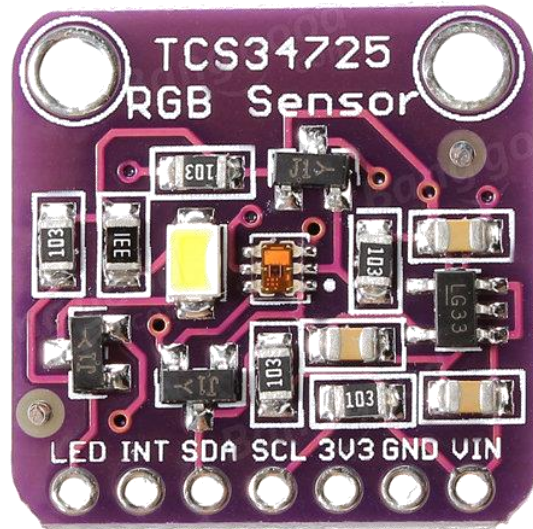
Raspberry Pi3
Model B



RGB LED SMD
Module



A Brief Note on Color Sensor



TCS34725 **RGB** Color Sensor with IR filter

The TCS34725 is a highly accurate RGB color and ambient light sensor that includes interrupts based on color thresholds so it can ping you when a green one goes by.

The module combines the sensor with a logic controllable white LED for illuminating the object to be measured and it comes with the easy to use I2C interface. The module includes logic level shifting on the I2C bus for easy interfacing to both 3.3V and 5V microcontrollers.

TCS34725 **RGB** Color Sensor with IR filter

Features

- Integrated IR blocking filter
- 3.8M:1 dynamic range
- Four independent analog-to-digital converters
- A reference channel for color analysis (clear channel photodiode)

Benefits

- Minimizes IR and UV spectral component effects to produce accurate color measurement
- Enables accurate color and ambient light sensing under varying lighting conditions
- Minimizes motion/transient errors
- Clear channel provides a reference which allows for isolation of color content

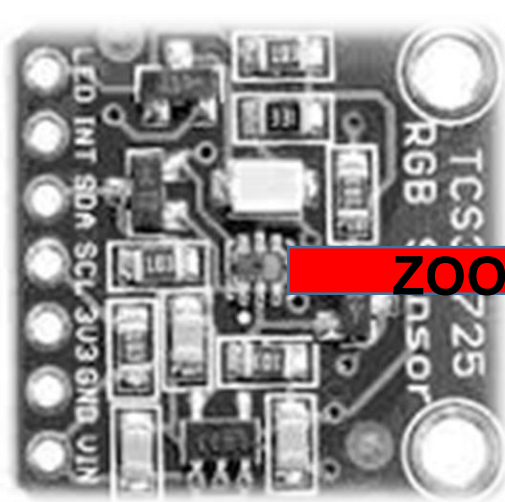
TCS34725 Sensor Data Sheet

Terminal Functions

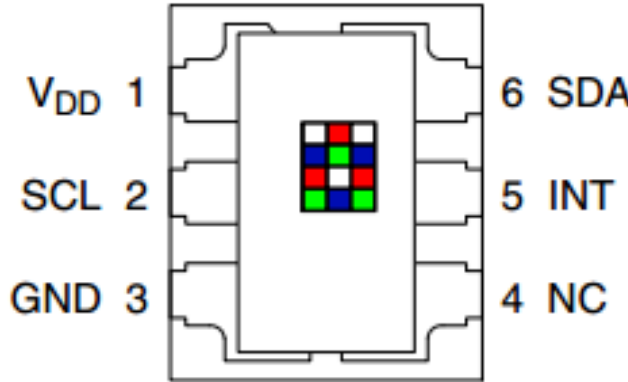
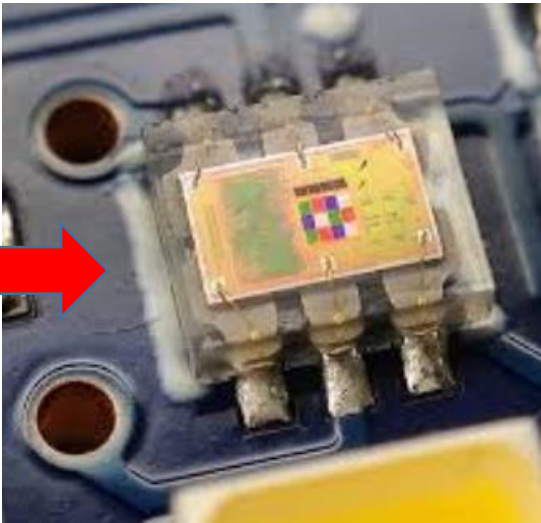
TERMINAL NAME	NO.	TYPE	DESCRIPTION
GND	3		Power supply ground. All voltages are referenced to GND.
INT	5	O	Interrupt — open drain (active low).
NC	4	O	No connect — do not connect.
SCL	2	I	I ² C serial clock input terminal — clock signal for I ² C serial data.
SDA	6	I/O	I ² C serial data I/O terminal — serial data I/O for I ² C .
V _{DD}	1		Supply voltage.

LED
INT
SDA
SCL

GND
V_{in}



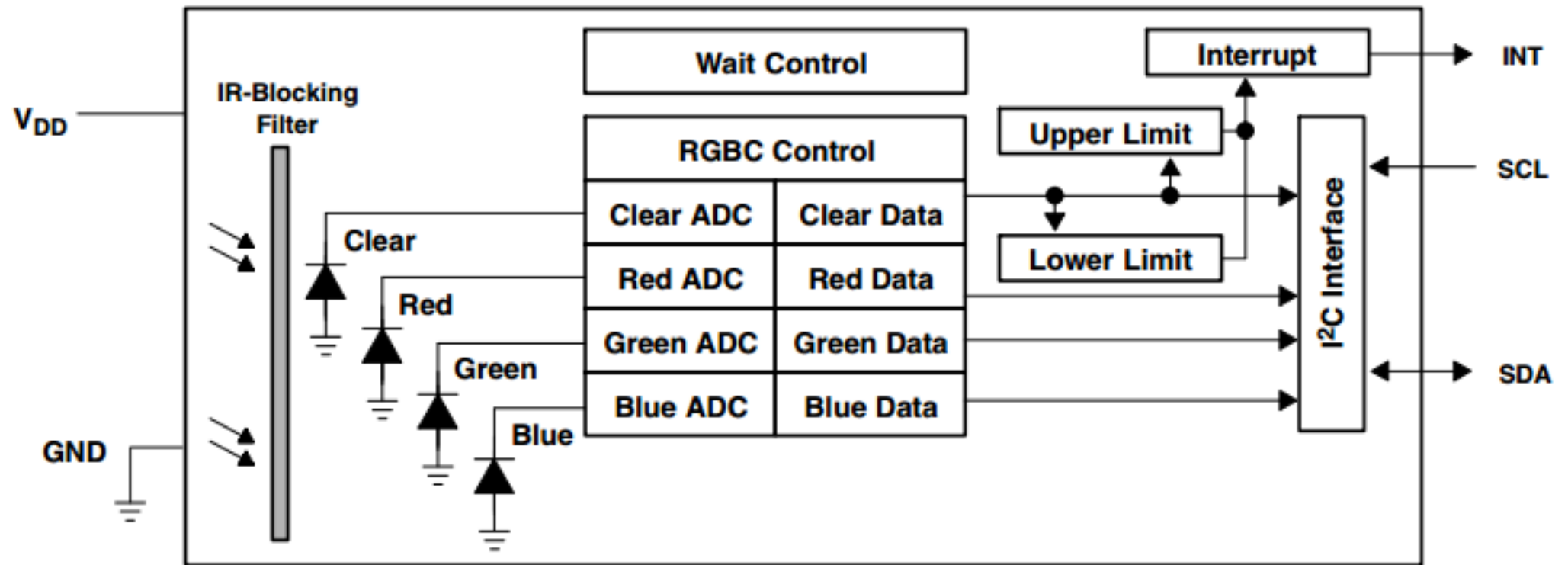
ZOOM →



Package Drawing Not to Scale

TCS34725 Sensor Data Sheet

Functional Block Diagram



TCS34725 Sensor Data Sheet

Detailed Description

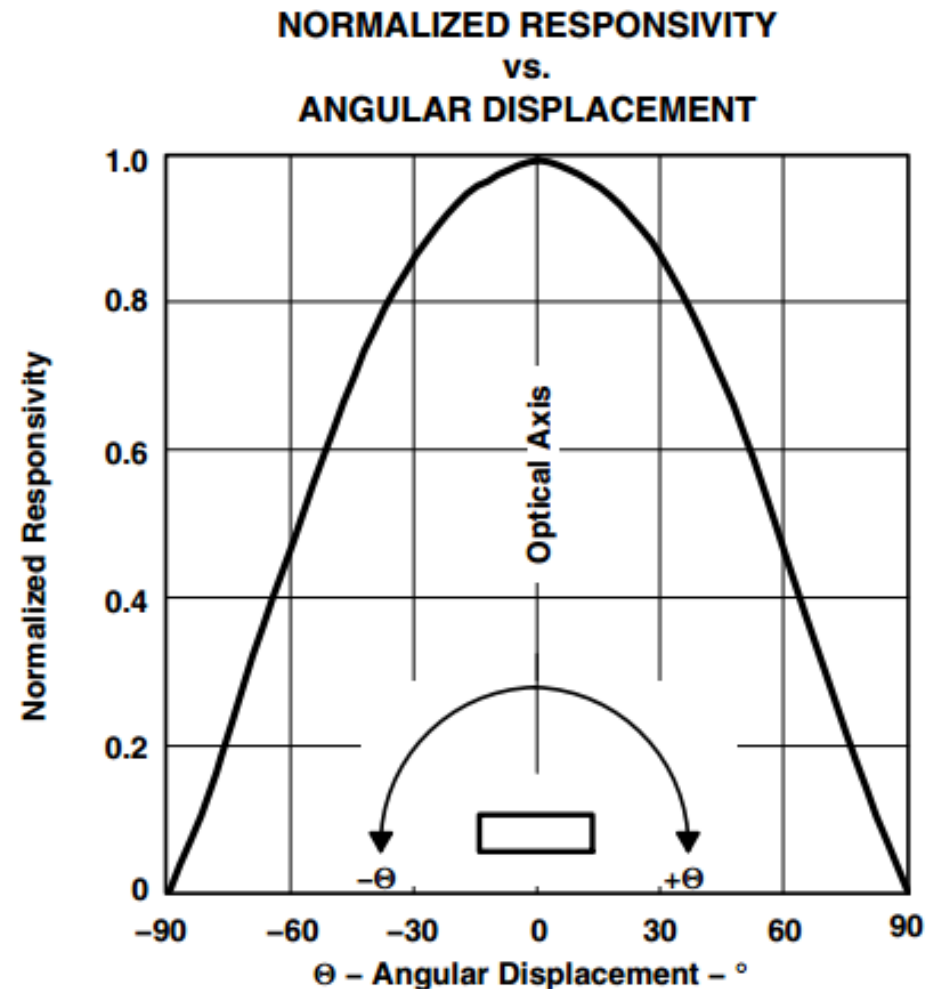
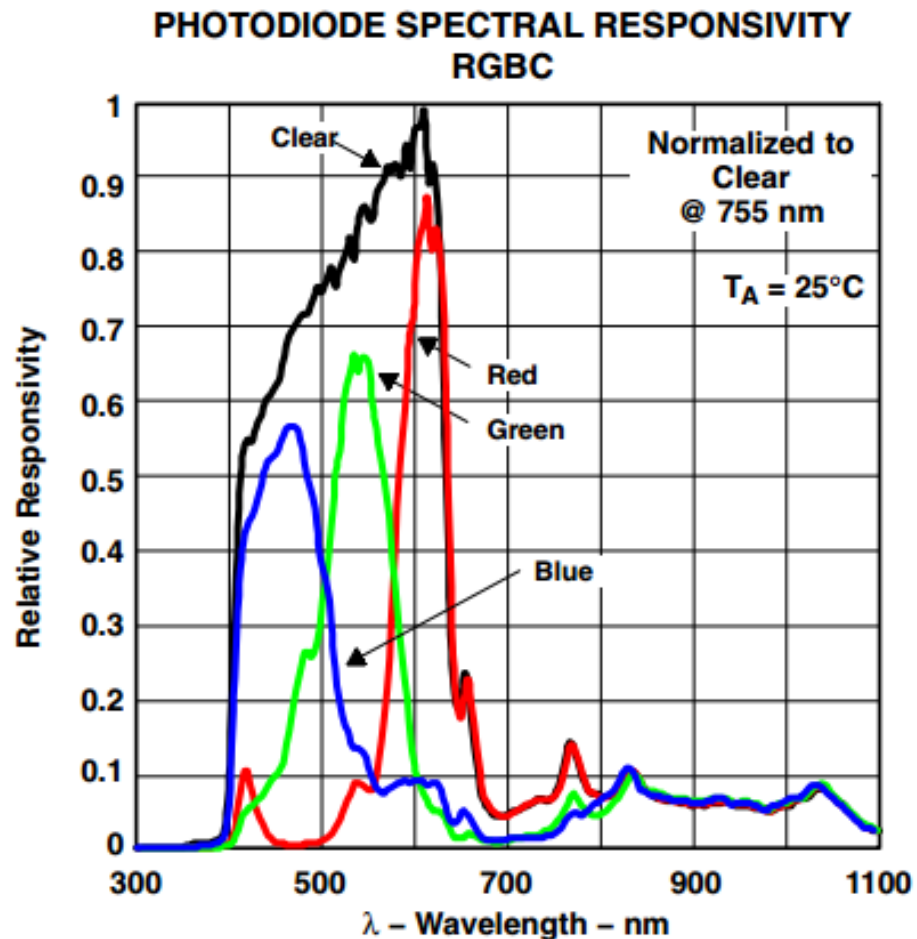
The TCS3472 light-to-digital converter contains a 3×4 photodiode array, four analog-to-digital converters (ADC) that integrate the photodiode current, data registers, a state machine, and an I²C interface. The 3×4 photodiode array is composed of red-filtered, green-filtered, blue-filtered, and clear (unfiltered) photodiodes. In addition, the photodiodes are coated with an IR-blocking filter. The four integrating ADCs simultaneously convert the amplified photodiode currents to a 16-bit digital value. Upon completion of a conversion cycle, the results are transferred to the data registers, which are double-buffered to ensure the integrity of the data. All of the internal timing, as well as the low-power wait state, is controlled by the state machine.

Communication of the TCS3472 data is accomplished over a fast, up to 400 kHz, two-wire I²C serial bus. The industry standard I²C bus facilitates easy, direct connection to microcontrollers and embedded processors.

In addition to the I²C bus, the TCS3472 provides a separate interrupt signal output. When interrupts are enabled, and user-defined thresholds are exceeded, the active-low interrupt is asserted and remains asserted until it is cleared by the controller. This interrupt feature simplifies and improves the efficiency of the system software by eliminating the need to poll the TCS3472. The user can define the upper and lower interrupt thresholds and apply an interrupt persistence filter. The interrupt persistence filter allows the user to define the number of consecutive out-of-threshold events necessary before generating an interrupt. The interrupt output is open-drain, so it can be wire-ORed with other devices.

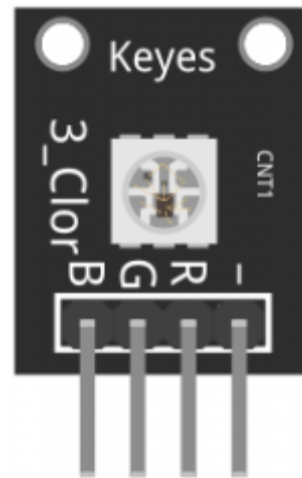
TCS34725 **RGB** Color Sensor with IR filter

It is important to note that the color detection of the photodiodes themselves is affected by infrared and to get a better result that is not affected by infrared you have to use an infrared filter



Color	Wavelength (nm)
Red	700 ~ 635
Green	560 ~ 520
Blue	490 ~ 450

A Brief Note on RGB Full color LED SMD Module



KY-009 RGB Full color LED SMD Module

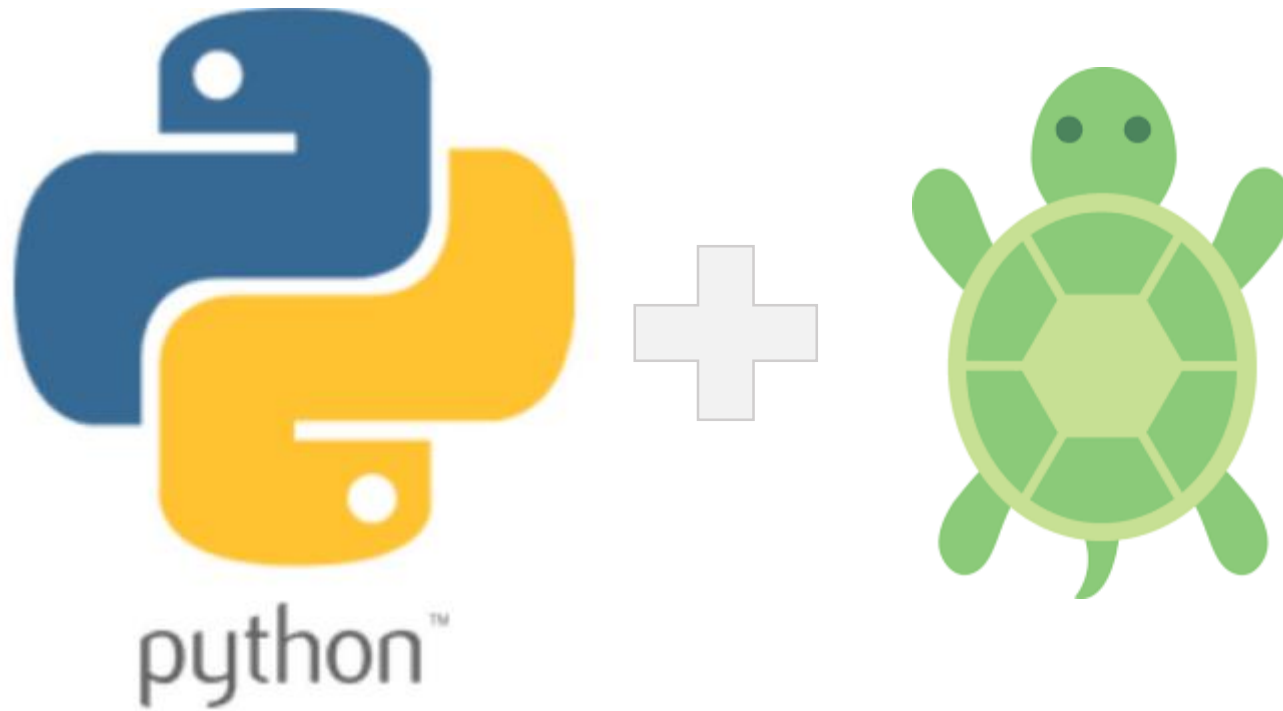
RGB full color LED Module KY-009, emits a range of colors by mixing red, green and blue. The amount of each primary color is adjusted using PWM.

KY-009 Specifications

The KY-009 RGB Full Color LED SMD Module consists of a 5050 SMD LED, use with limiting resistors to prevent burnout. Compatible with popular electronics platforms like Arduino, Raspberry Pi and ESP8266.

Operating Voltage	5V max Red 1.8V ~2.4V Green 2.8V ~ 3.6V Blue 2.8V ~ 3.6V
Forward Current	20mA ~ 30mA
Operating Temperature	-25°C to 85°C [-13°F ~ 185°F]
Dimensions	18.5mm x 15mm [0.728in x 0.591in]

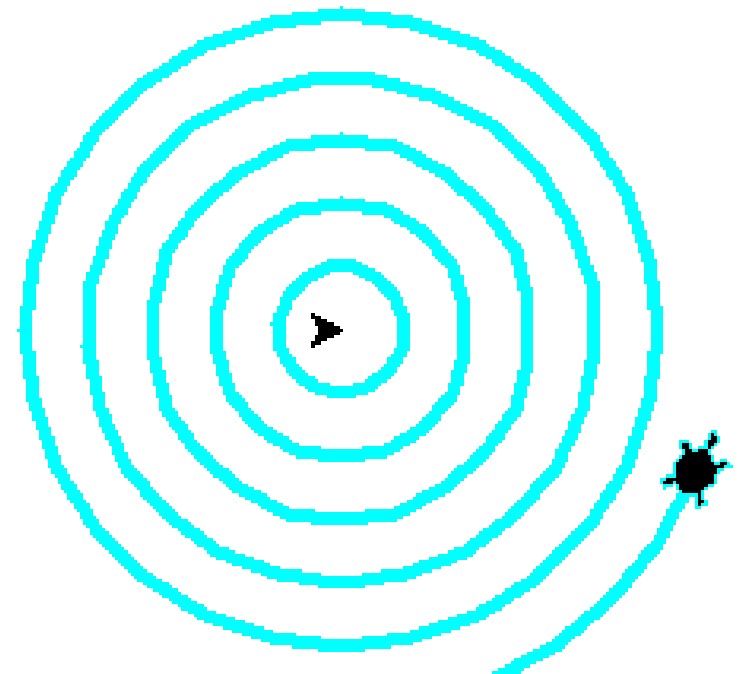
A Brief Note on Python Turtle Module



Python Turtle Tutorial and Animations

Python Turtle is something that evolved from Logo programming language, invented in 1966 by Wally Feurzig. With the aid of **Object Oriented Programming** approach, we can create an impressive set of animations easily. The following animation was created by Python Turtle.

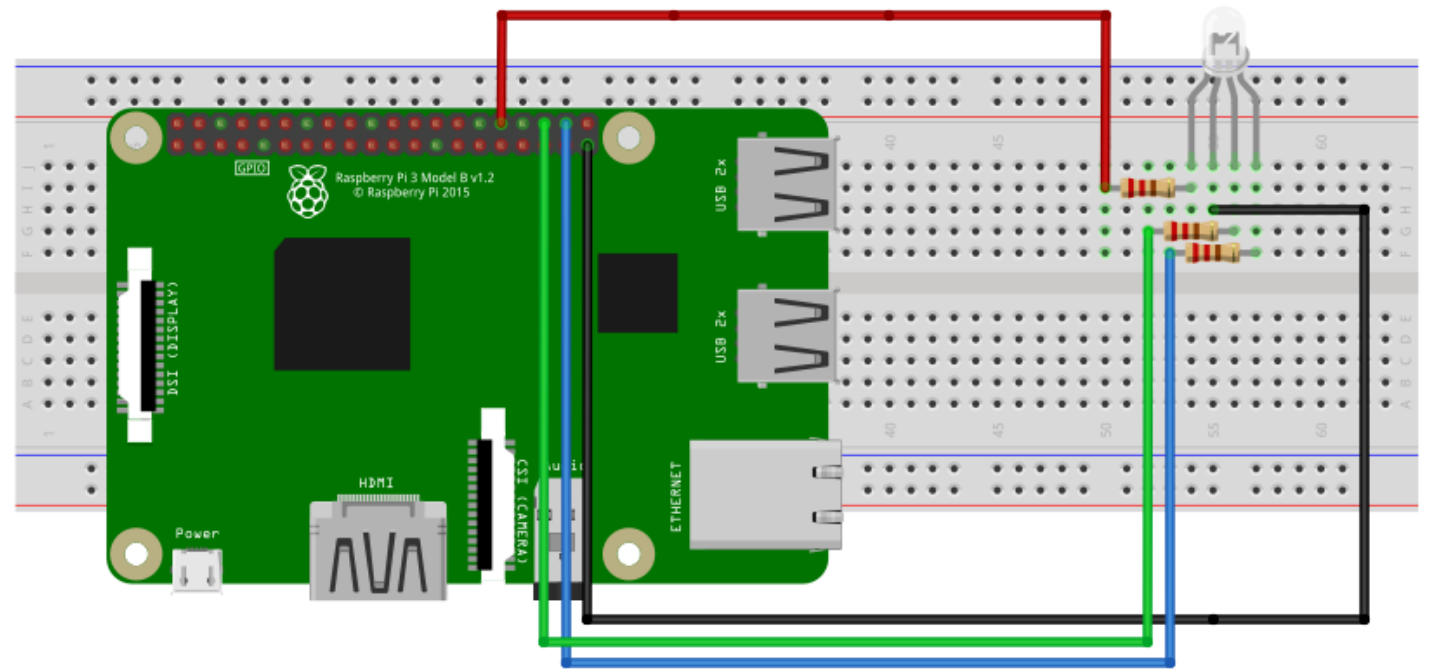
```
1 import turtle      # importing the module
2 trtl = turtle.Turtle()    #making a turtle object of Turtle
3 screen=turtle.Screen()    #making a canvas for drawing
4 screen.setup(420,320)     #choosing the screen size
5 trtl.pencolor('cyan')     #making colour of the pen red
6 trtl.pensize(4)          #choosing the size of pen nib
7 trtl.shape('turtle')     #choosing the shape of pen nib
8 n=0
9 while n<7:    #loop for 7 circles
10     n=n+1
11     trtl.penup()
12     trtl.setpos(0,-n*20)
13     trtl.pendown()
14     trtl.circle(20*n)
```



Hardware Design and Implementation

Connect the RGB LED as bellow:

- **Pin GND** to sensor GND
- **Pin R** to sensor 32
- **Pin G** to sensor 36
- **Pin B** to sensor 38

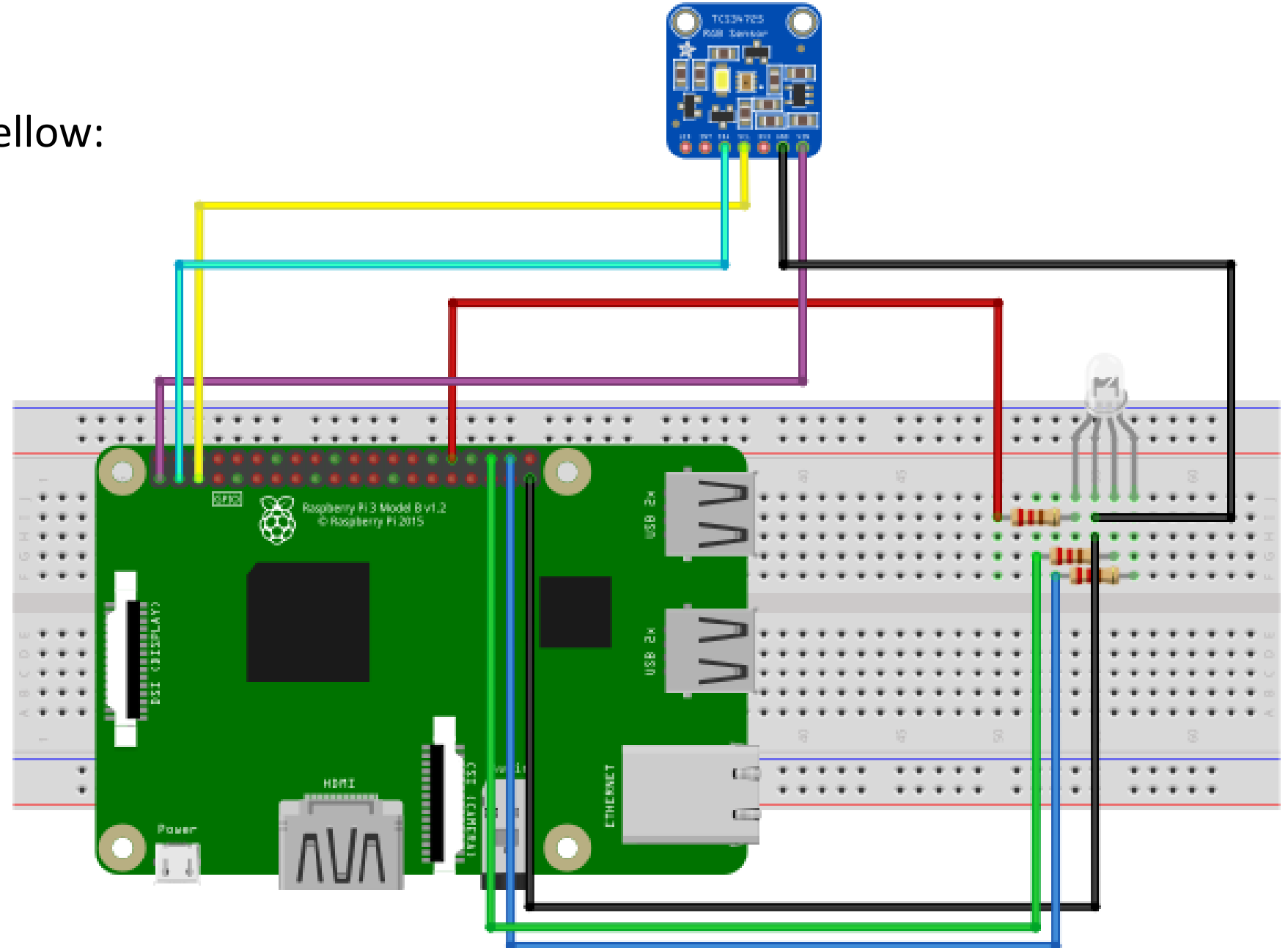


fritzing

Hardware Design and Implementation

Connect the Sensor as bellow:

- Pin 3V3 to sensor VIN
- Pin GND to sensor GND
- Pin SCL to sensor SCL
- Pin SDA to sensor SDA



TCS34725 color sensor Library

Installation

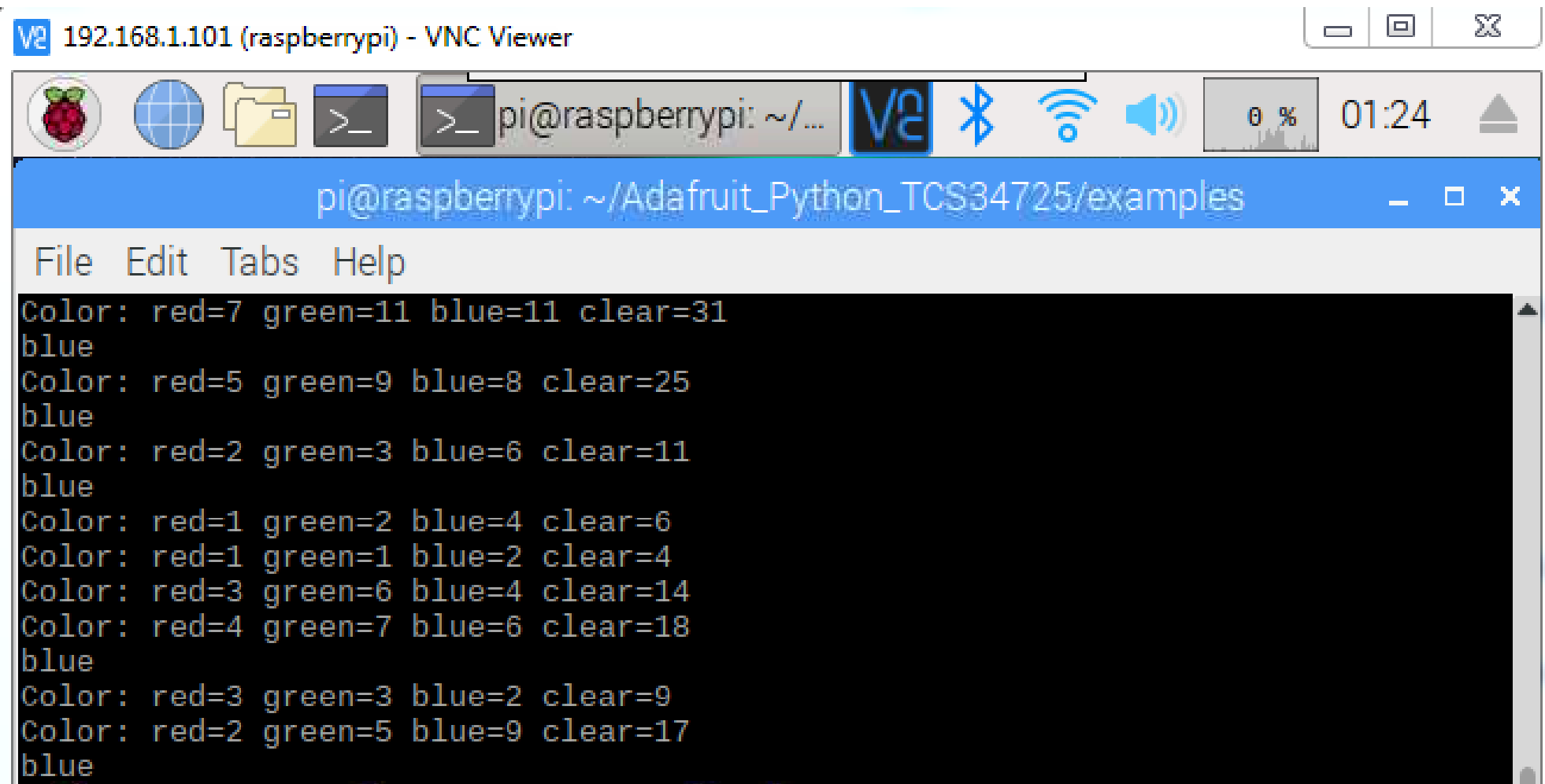
To install the library from source (recommended) run the following commands on a Raspberry Pi or other Debian-based OS system:

```
sudo apt-get install git build-essential python-dev  
cd ~  
git clone https://github.com/adafruit/Adafruit_Python_TCS34725.git  
cd Adafruit_Python_TCS34725  
sudo python setup.py install
```


Experiment

When we place the blue object in front of the sensor (1-3 cm), the blue frequency (B) values oscillate between **5** and **40**.

Check the values displayed on Terminal. R & G & B readings – see figure below.



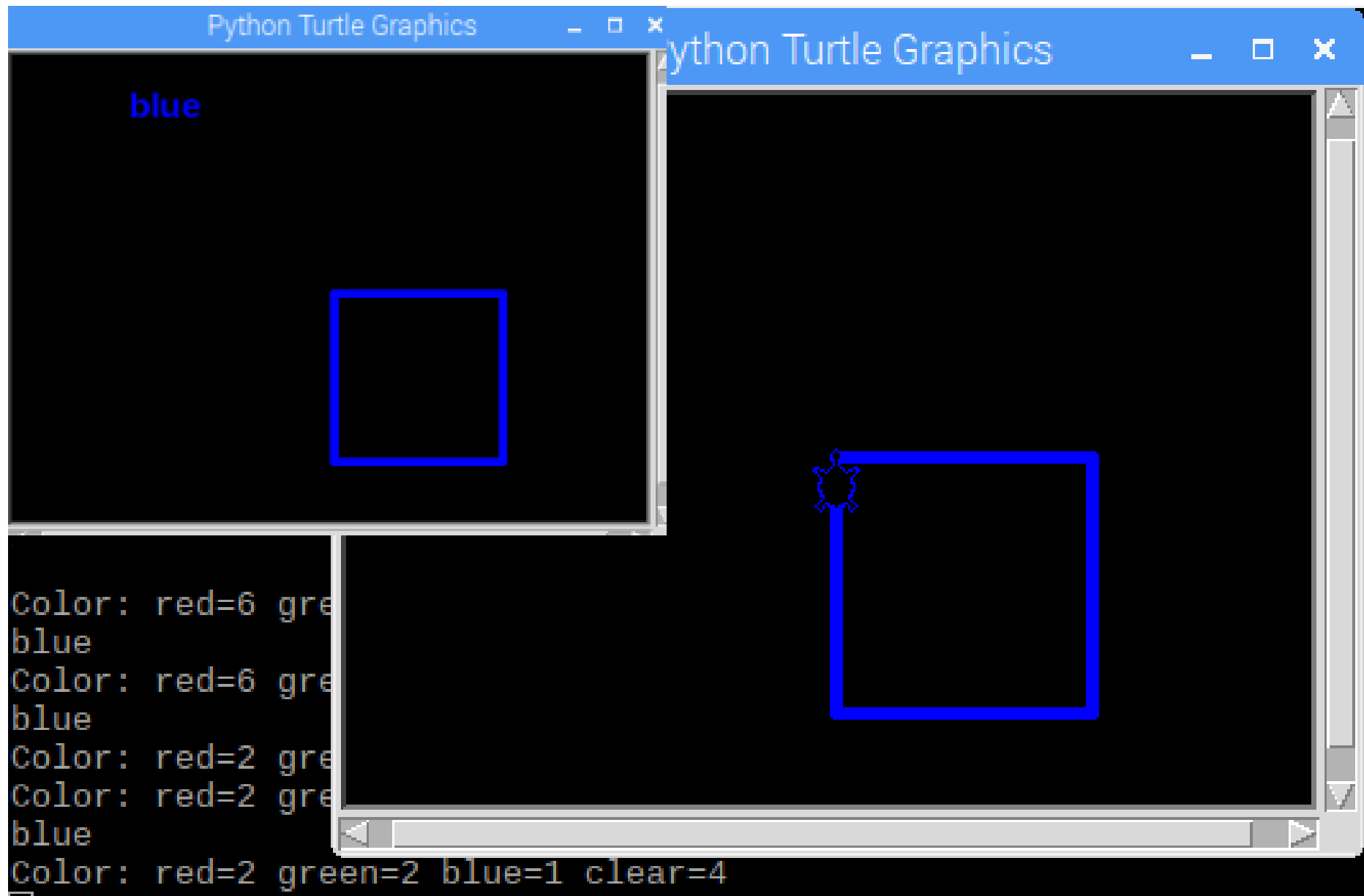
VNC 192.168.1.101 (raspberrypi) - VNC Viewer

pi@raspberrypi: ~/...

pi@raspberrypi: ~/Adafruit_Python_TCS34725/examples

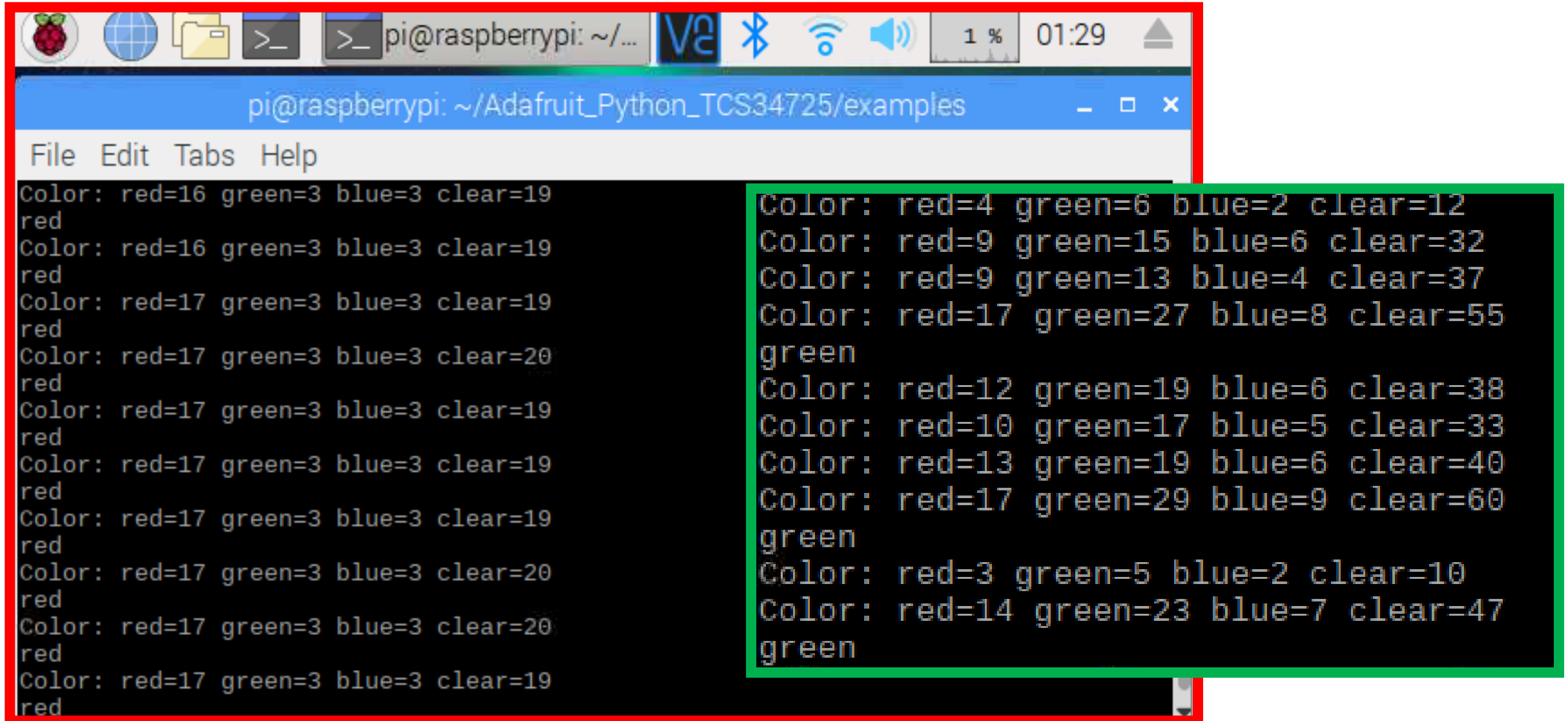
```
File Edit Tabs Help
Color: red=7 green=11 blue=11 clear=31
blue
Color: red=5 green=9 blue=8 clear=25
blue
Color: red=2 green=3 blue=6 clear=11
blue
Color: red=1 green=2 blue=4 clear=6
Color: red=1 green=1 blue=2 clear=4
Color: red=3 green=6 blue=4 clear=14
Color: red=4 green=7 blue=6 clear=18
blue
Color: red=3 green=3 blue=2 clear=9
Color: red=2 green=5 blue=9 clear=17
blue
```

Experiment Result



Experiment Result

I Repeat this process with a green and red objects and write down the upper and bottom frequency limits for each color.



The image shows a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~/Adafruit_Python_TCS34725/examples'. The terminal output consists of two columns of data. The left column, which is highlighted with a red border, contains 12 lines of data for a red object. The right column, which is highlighted with a green border, contains 10 lines of data for a green object. Each line starts with 'Color:' followed by four values: red, green, blue, and clear, separated by spaces. The values are integers representing frequency limits.

```
pi@raspberrypi: ~/Adafruit_Python_TCS34725/examples
File Edit Tabs Help
Color: red=16 green=3 blue=3 clear=19
red
Color: red=16 green=3 blue=3 clear=19
red
Color: red=17 green=3 blue=3 clear=19
red
Color: red=17 green=3 blue=3 clear=20
red
Color: red=17 green=3 blue=3 clear=19
red
Color: red=17 green=3 blue=3 clear=19
red
Color: red=17 green=3 blue=3 clear=19
red
Color: red=17 green=3 blue=3 clear=20
red
Color: red=17 green=3 blue=3 clear=20
red
Color: red=17 green=3 blue=3 clear=19
red
Color: red=4 green=6 blue=2 clear=12
Color: red=9 green=15 blue=6 clear=32
Color: red=9 green=13 blue=4 clear=37
Color: red=17 green=27 blue=8 clear=55
green
Color: red=12 green=19 blue=6 clear=38
Color: red=10 green=17 blue=5 clear=33
Color: red=13 green=19 blue=6 clear=40
Color: red=17 green=29 blue=9 clear=60
green
Color: red=3 green=5 blue=2 clear=10
Color: red=14 green=23 blue=7 clear=47
green
```

Note :

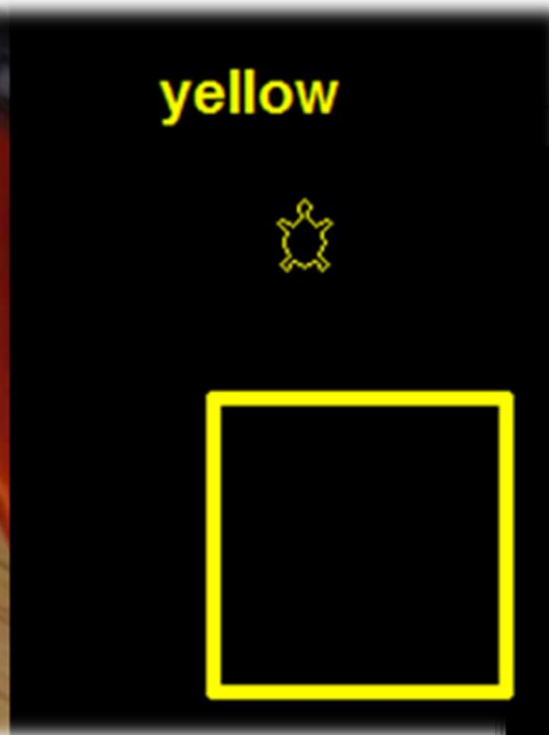
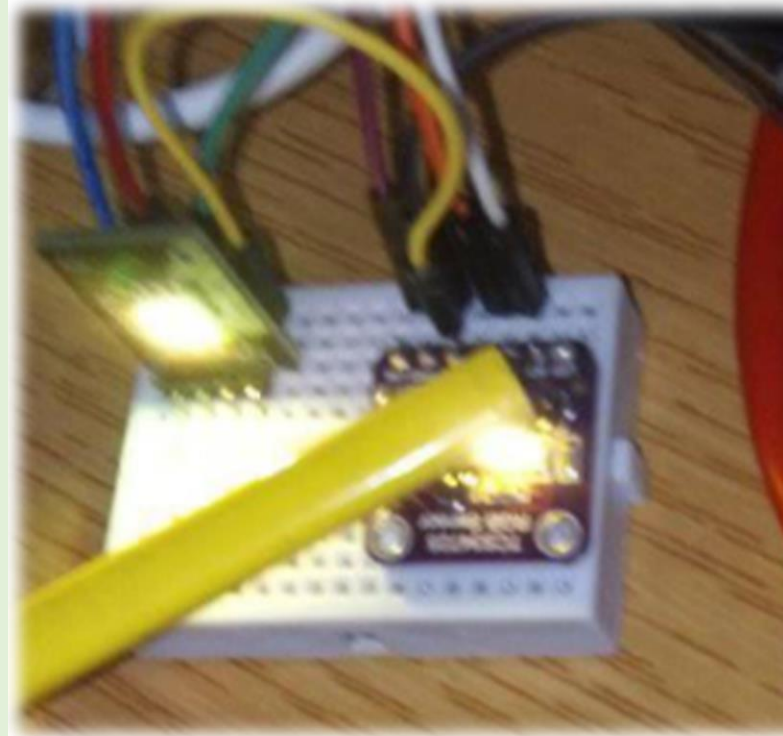
To distinguish between different colors we have three conditions:

- When the R is the maximum value (in RGB parameters) we know we have a red object
- When G is the maximum value, we know we have a green object
- When B is the maximum value, we know we have a blue object

Now, place something in front of the sensor. It should print the color detected: red, green or blue.

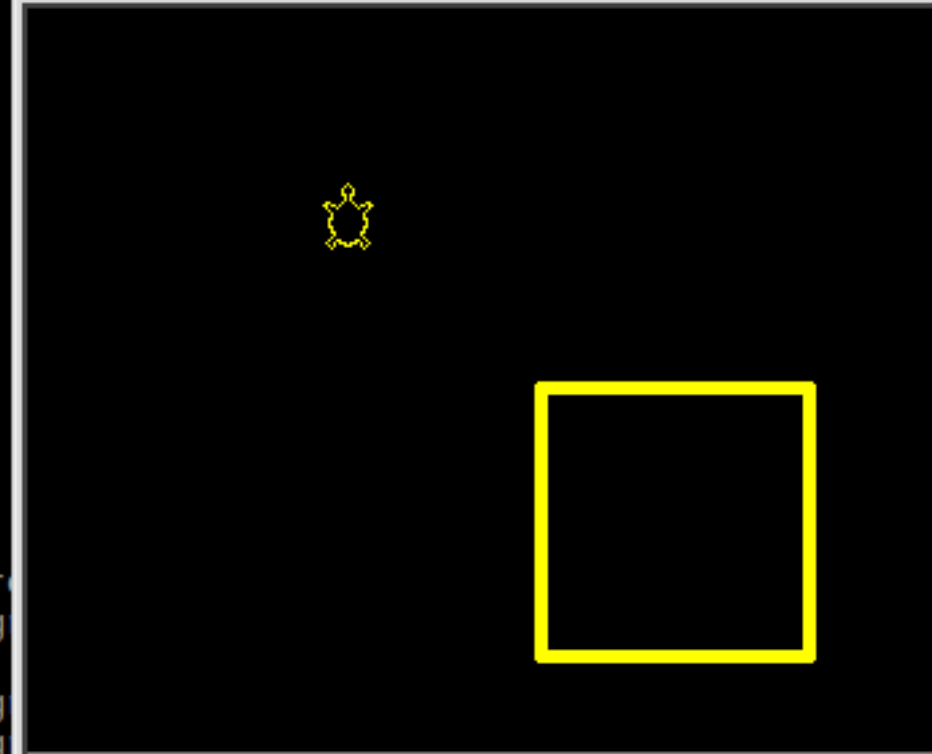
The sensor can also detect other colors with more conditions.

Experiment Result



File Edit Tabs Help

Python Turtle Graphics

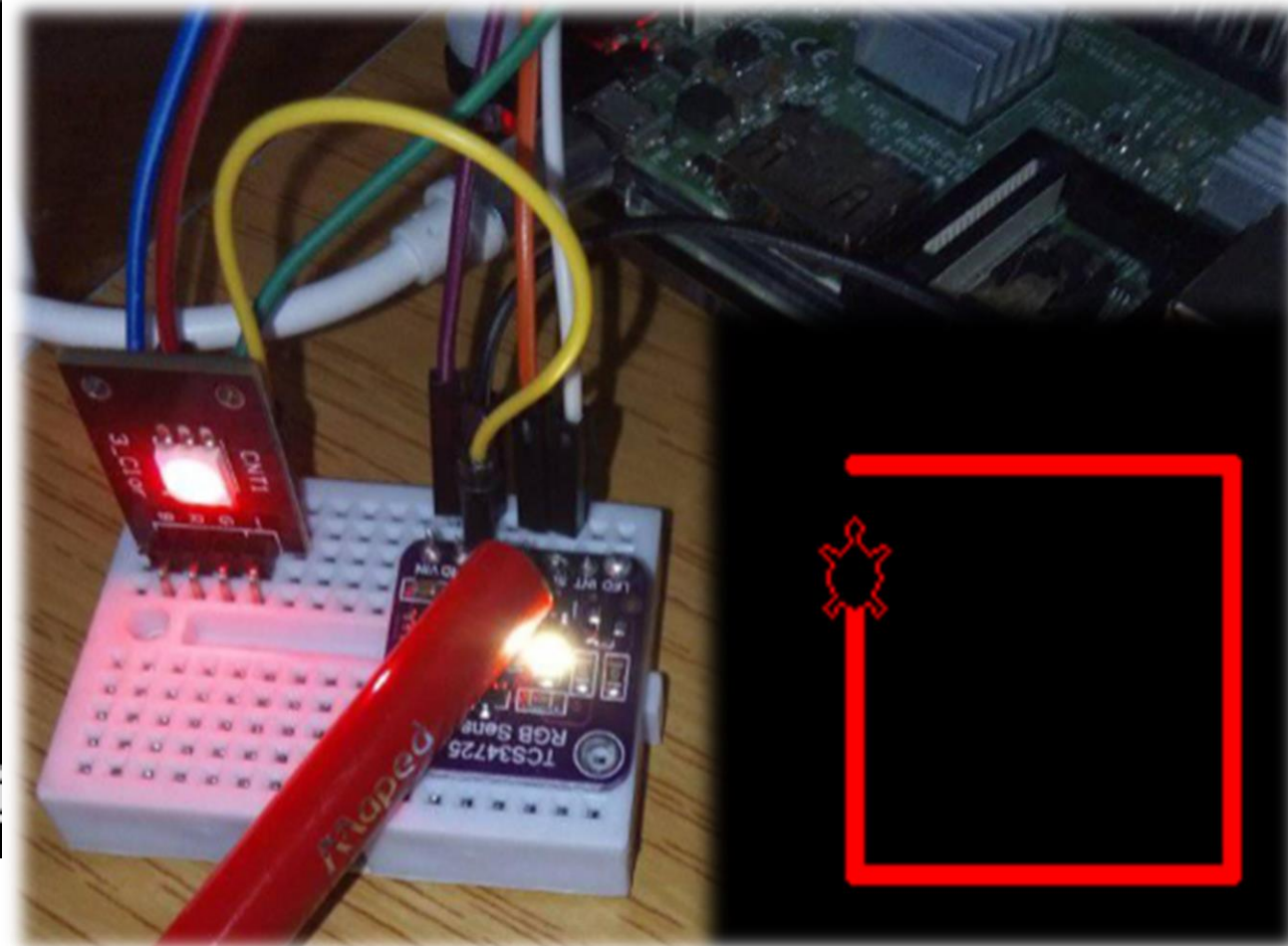
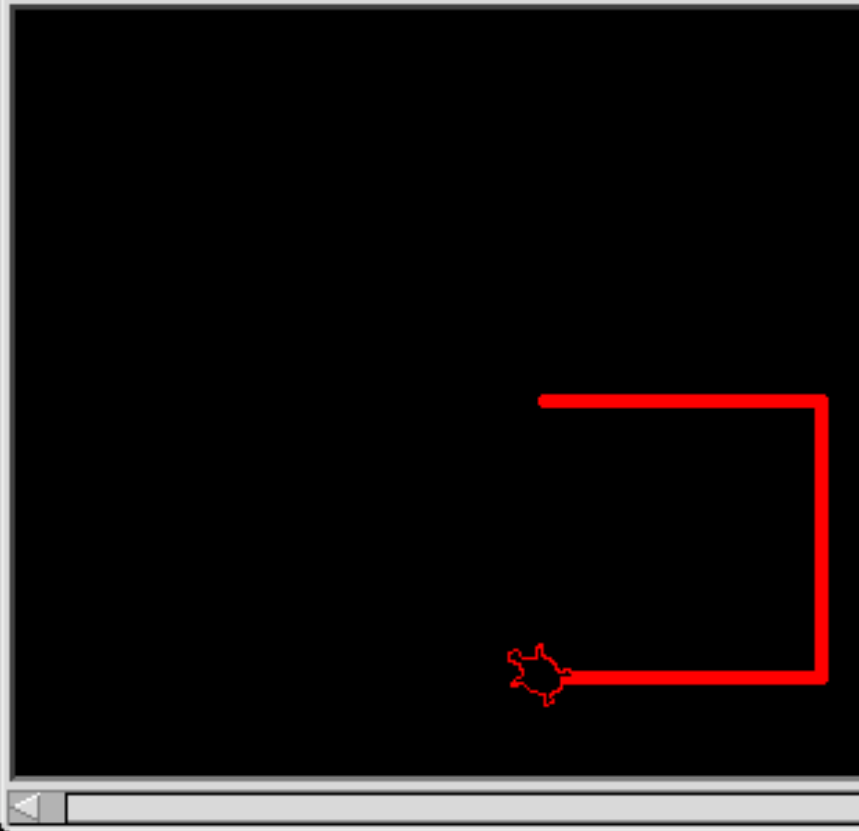


```
Color: red=2 gr  
Color: red=22 g  
yellow  
Color: red=33 g  
Color: red=50 g  
Color: red=24 g  
yellow
```

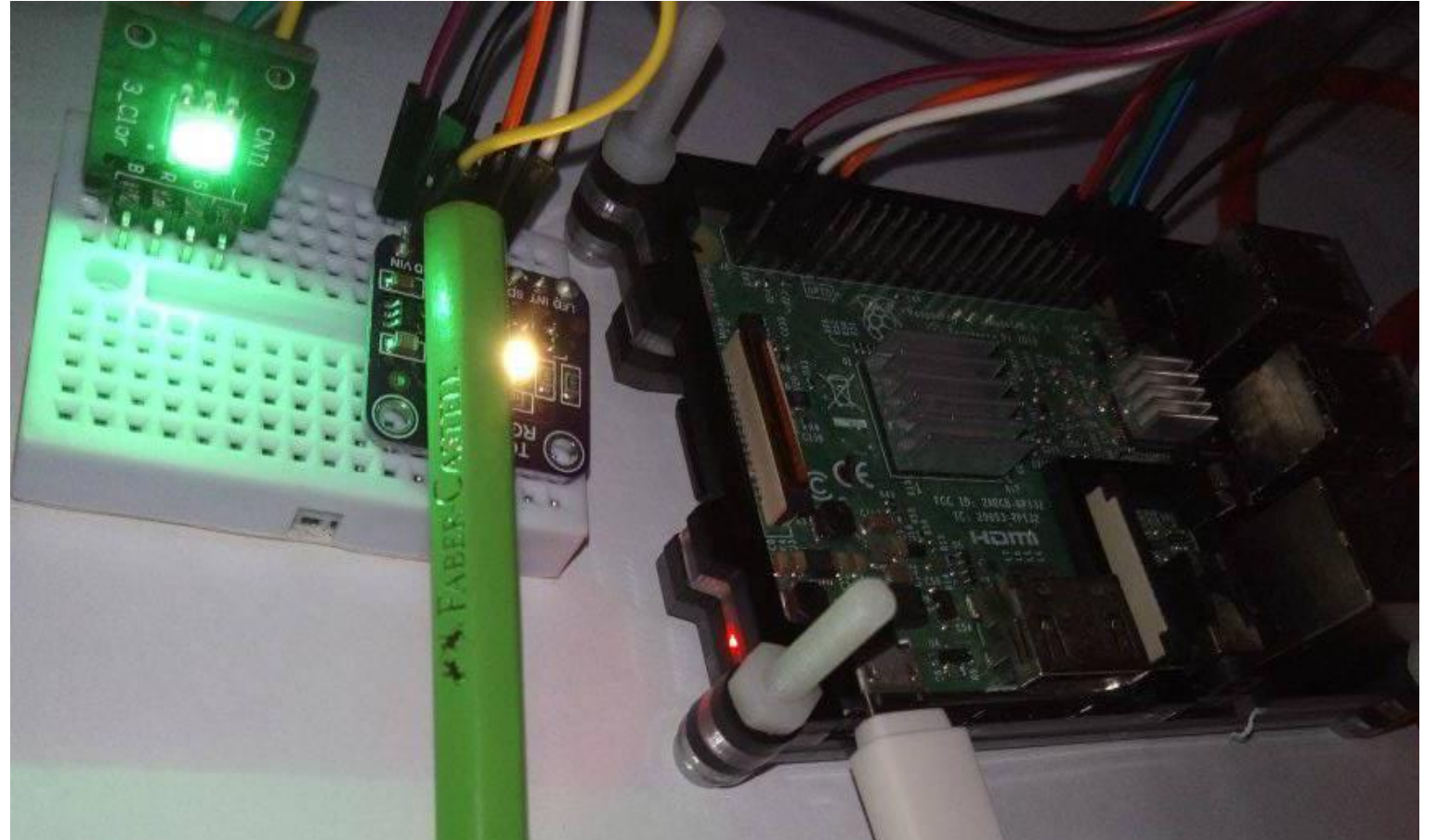
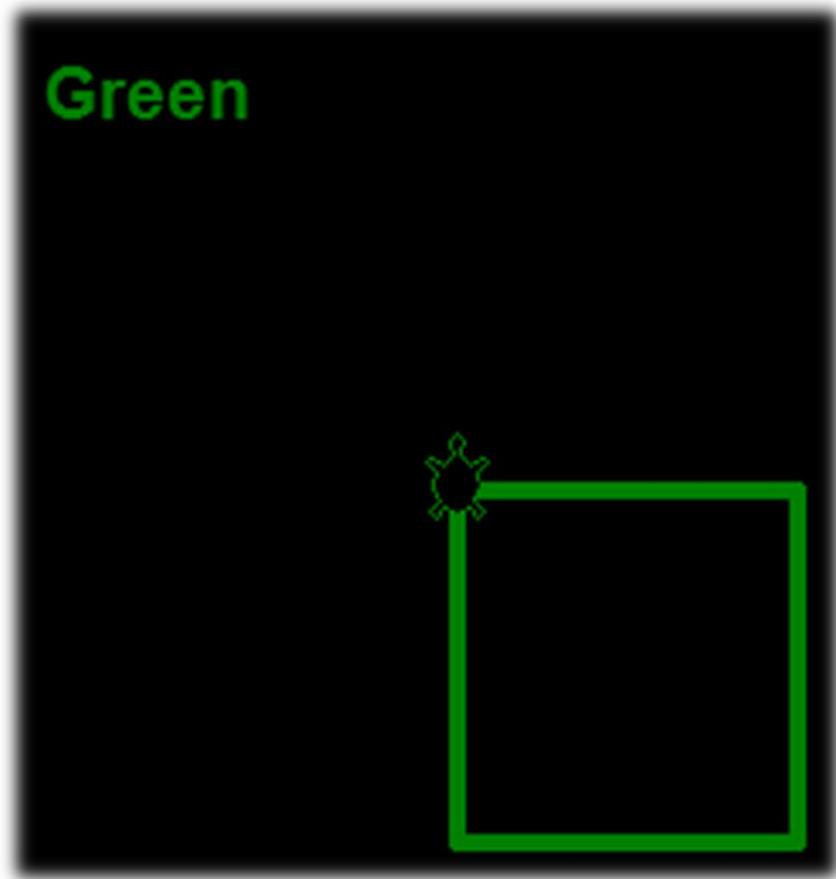
Experiment Result

```
File Edit Tabs Help
yellow
Color: red=27 g
Color: red=24 g
yellow
pi@raspberrypi:~$
Color: red=25 g
yellow
Color: red=12 g
red
Color: red=13 g
red
Color: red=79 g
Color: red=11 g
red
```

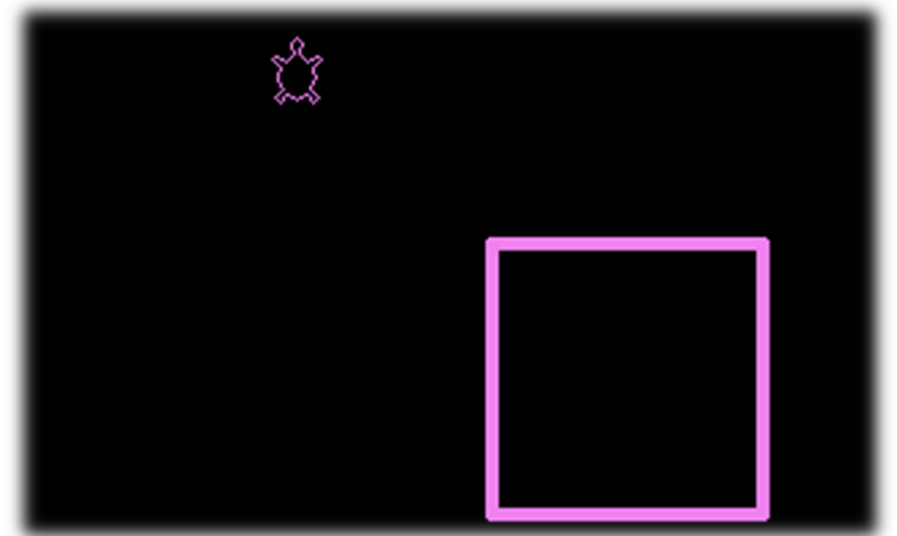
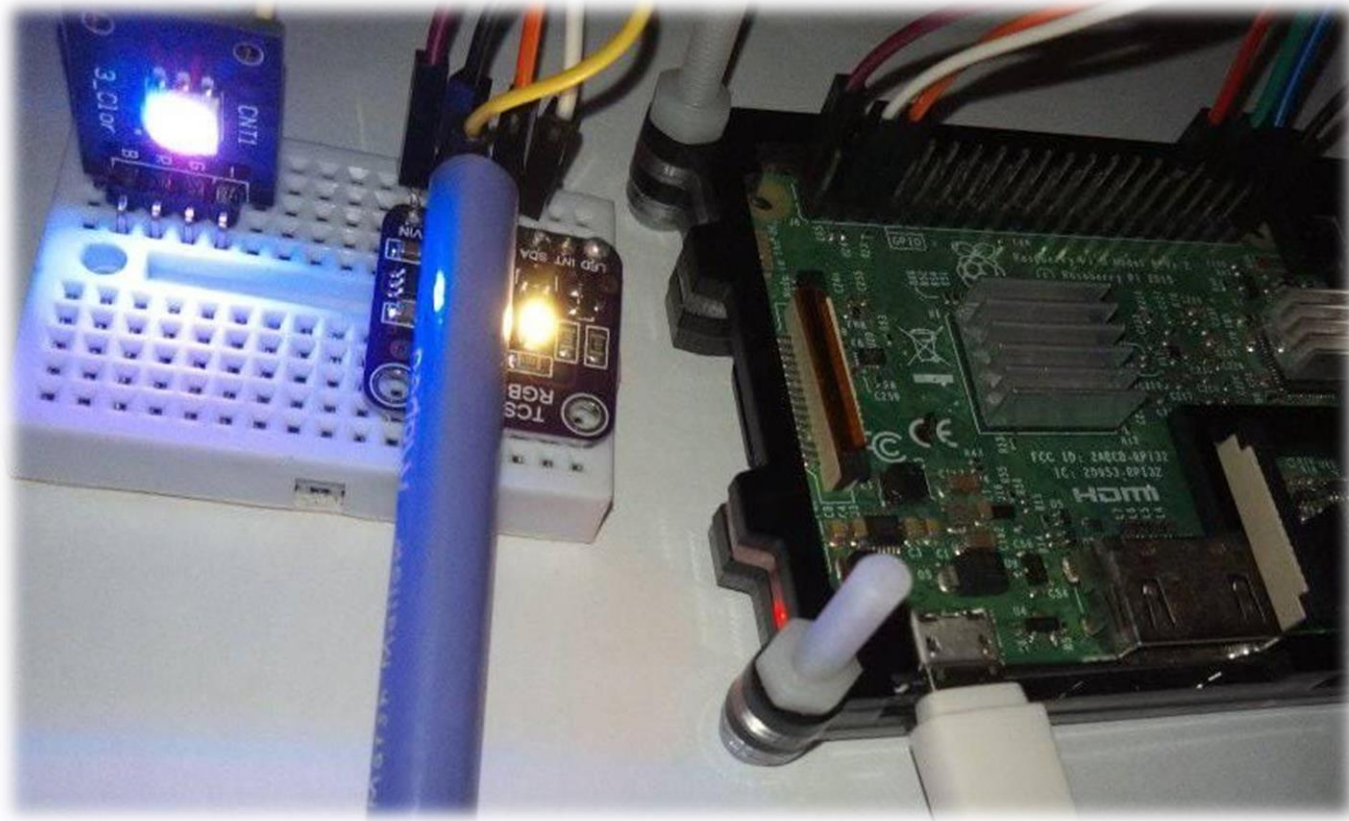
Python Turtle Graphics



Experiment Result



Experiment Result



Applications

Color Sensors can be implemented in a variety of projects and applications like:

RGB LED Backlight Control

Light Color Temperature Measurement

Ambient Light Sensing for Display

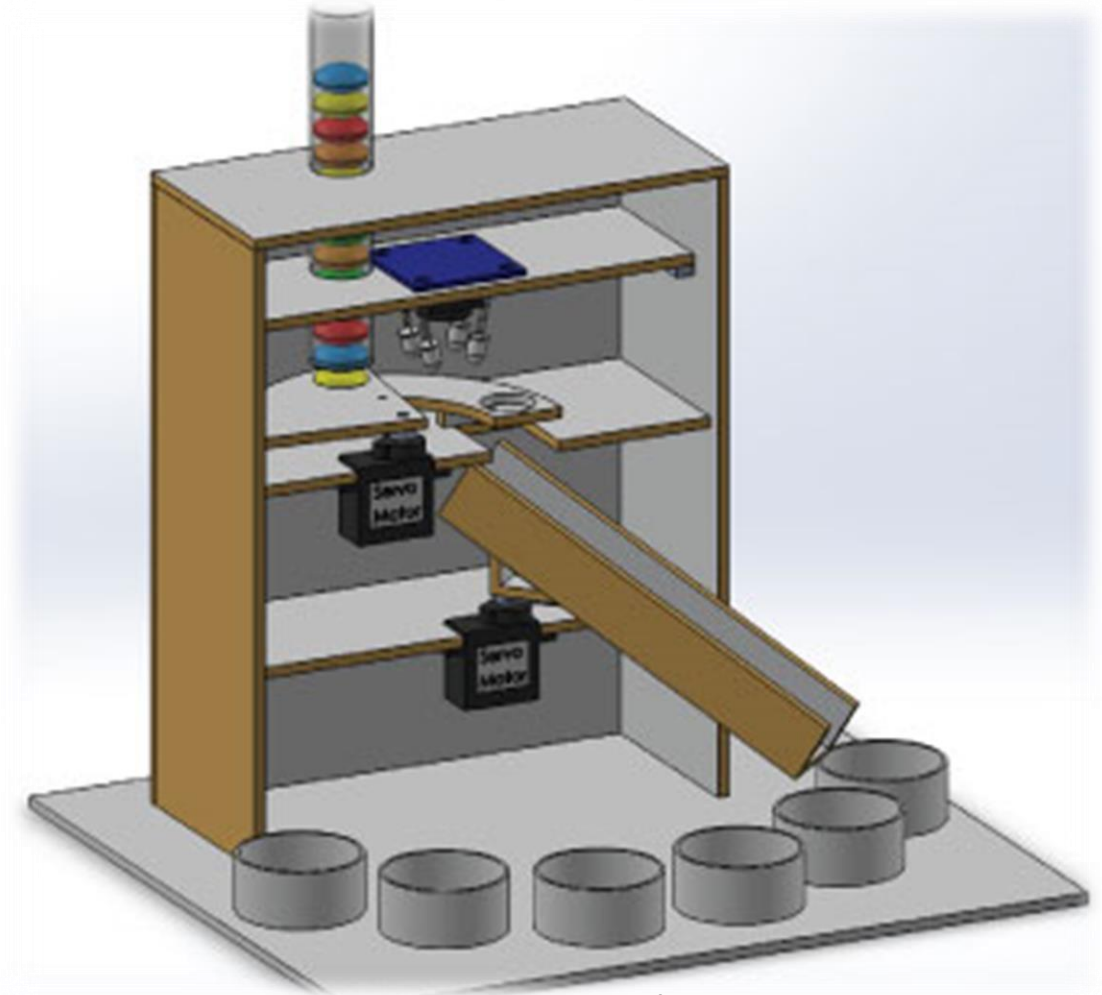
Fluid and Gas Analysis

Product Color Verification and Sorting

Consumer and Commercial Printing

Medical and Health Fitness

Industrial Automation

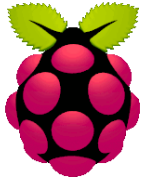


Product Sorting

References:

1. <https://howtomechatronics.com/projects/arduino-color-sorter-project>
2. <https://www.adafruit.com/product/1334>
3. https://github.com/adafruit/Adafruit_Python_TCS34725
4. <https://www.vivaxsolutions.com/web/python-turtle.aspx>
5. <https://www.electronicshub.org/raspberry-pi-color-sensor-tutorial/>
6. <https://www.tutorialspoint.com/turtle-programming-in-python>





My code :

<https://gist.github.com/aula9/69473ef635e3f8610dcc36815116b4f4>

Note : The code in progress to test new colors



Thank You