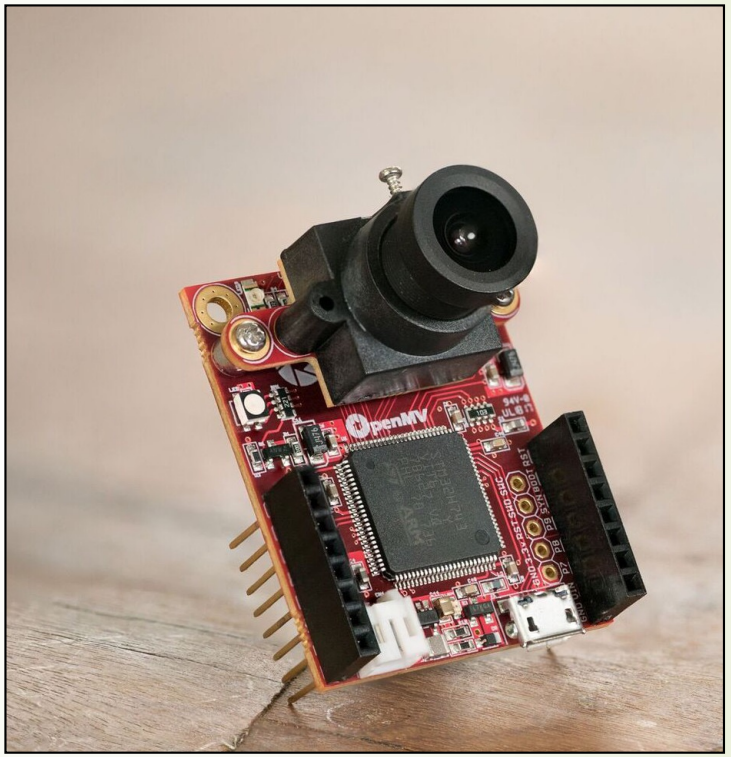
Introduction to the OpenMV H7 Machine Vision Unit

Detecting Color with MicroPython



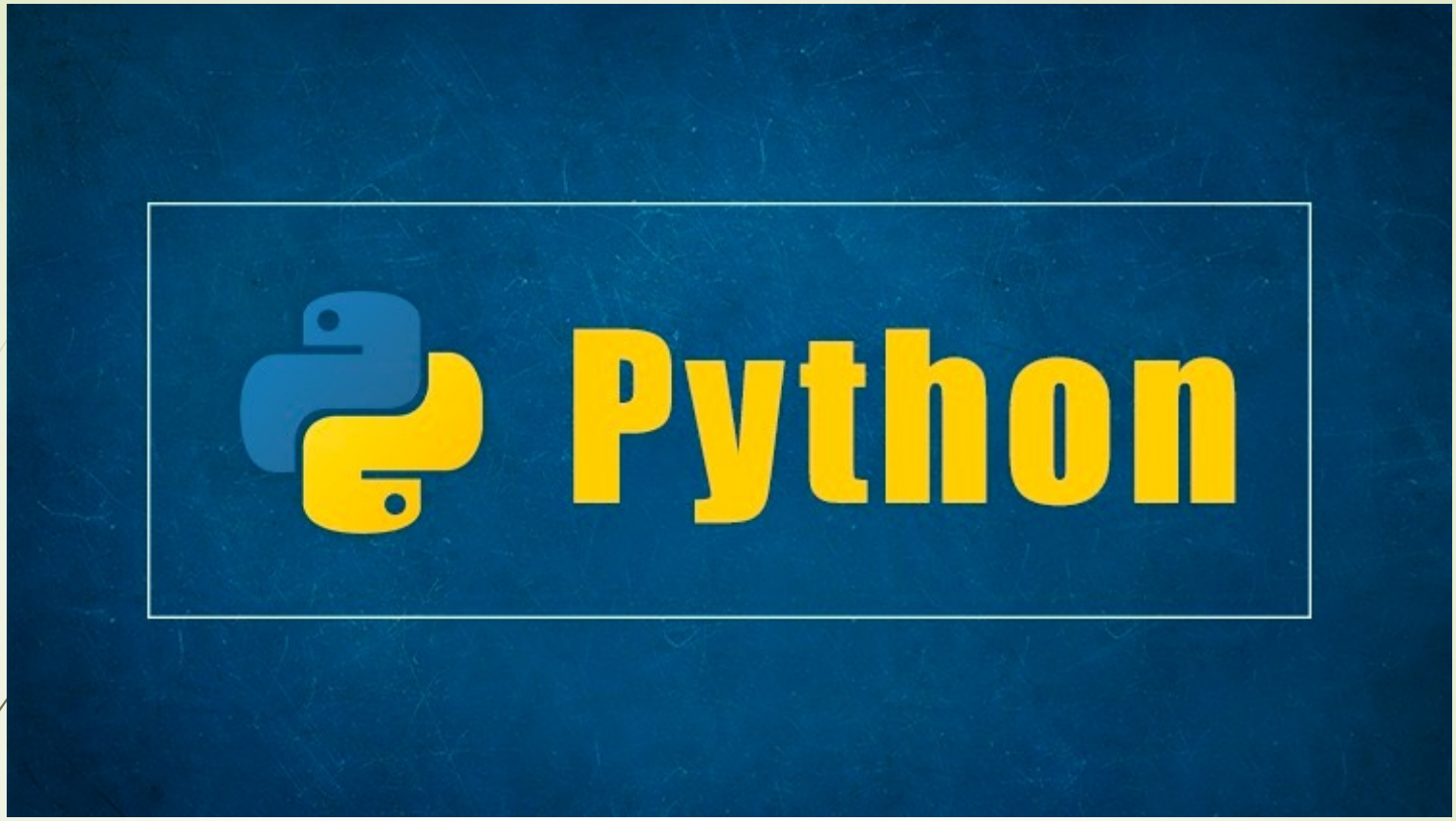
[Untitled photo of OpenMV camera] (2019) Retrieved from https://docs.openmv.io/openmvcam/tutorial/overview.html

Ethan Bressler, CTM

John R. Wright, Jr., PhD, CSTM, CLSSGB, CSCE, F.ATMAE

AENG 467 Mobile Robotics

Millersville University


Python


MicroPython
Python for microcontrollers
open source

Most Popular Programming Languages 1965 - 2019

- Https://www.youtube.com/watch?v=Og847HVwRSI

What is Python & Why Learn it?

- https://www.youtube.com/watch?v=Y8Tko2YC5hA

The BoD

- https://www.youtube.com/watch?v=Wpx6XnankZ8

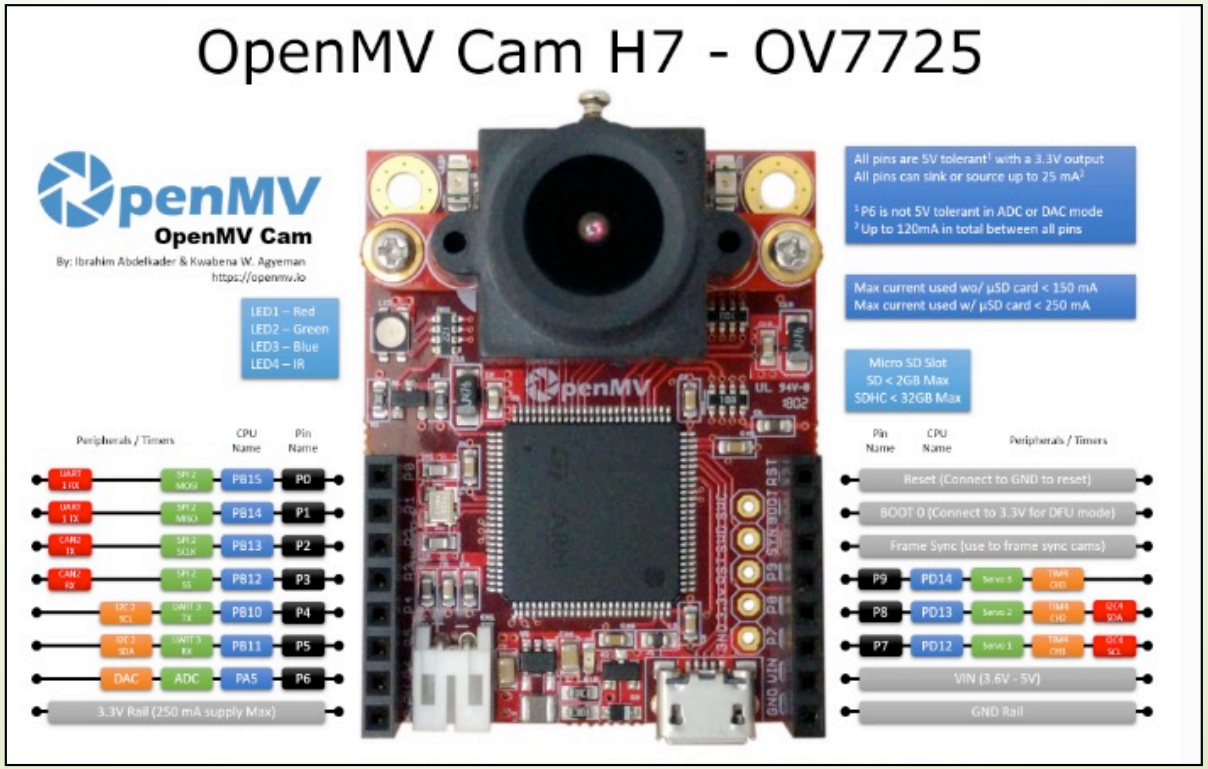
Overview of the OpenMV H7

- OpenMV is a vision system that can be used in conjunction with microcontrollers.
- Capable of color tracking, face detection, QR code detection, shape recognition and more.
- Uses MicroPython and the OpenMV IDE.

OpenMV Cam H7

- Runs on 5 VDC, can be powered by USB, JST, or GND and VIN pins.

- 10 Output pins that run 3.3-5VDC



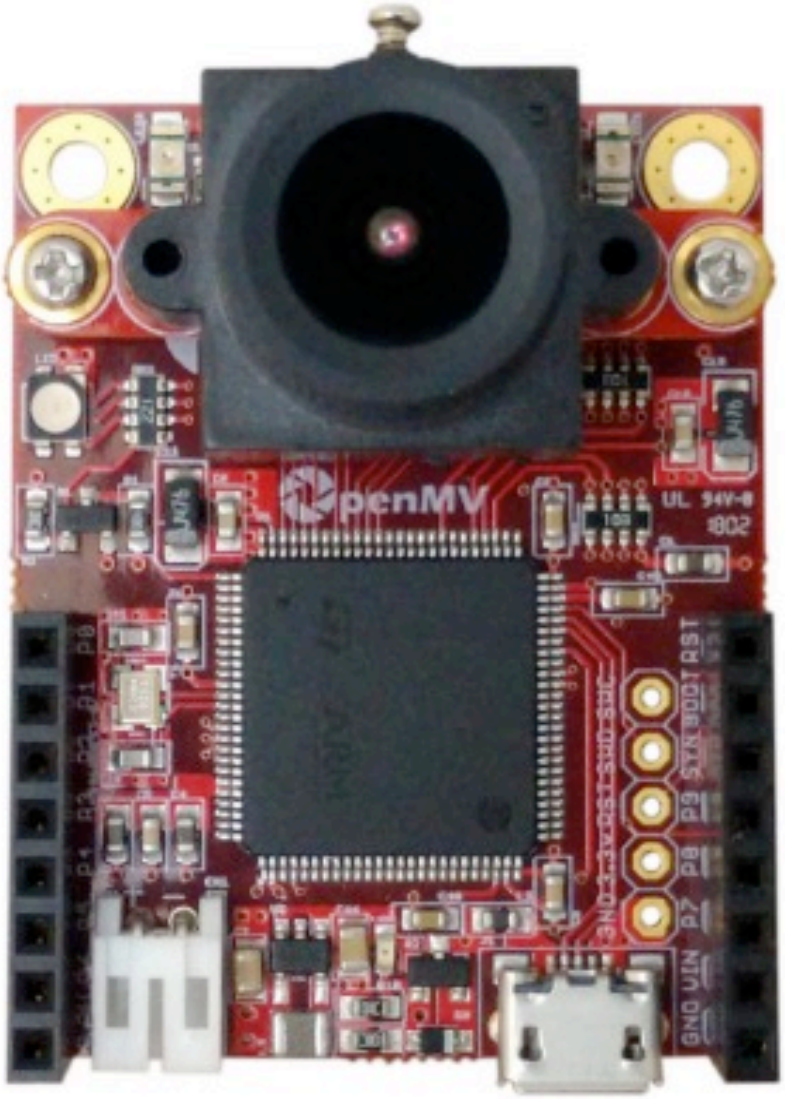
[Untitled photo of OpenMV Cam H7] (2019) Retrieved from https://docs.openmv.io/openmvcam/tutorial/io_tutorial.html

OpenMV Cam H7 - OV7725

OpenMV Cam

By: Ibrahim Abdelkader & Kwabena W. Agyeman
https://openmv.io

LED1 – Red
LED2 – Green
LED3 – Blue
LED4 – IR



All pins are 5V tolerant[1] with a 3.3V output
All pins can sink or source up to 25 mA[2]

[1] P6 is not 5V tolerant in ADC or DAC mode
[2] Up to 120mA in total between all pins

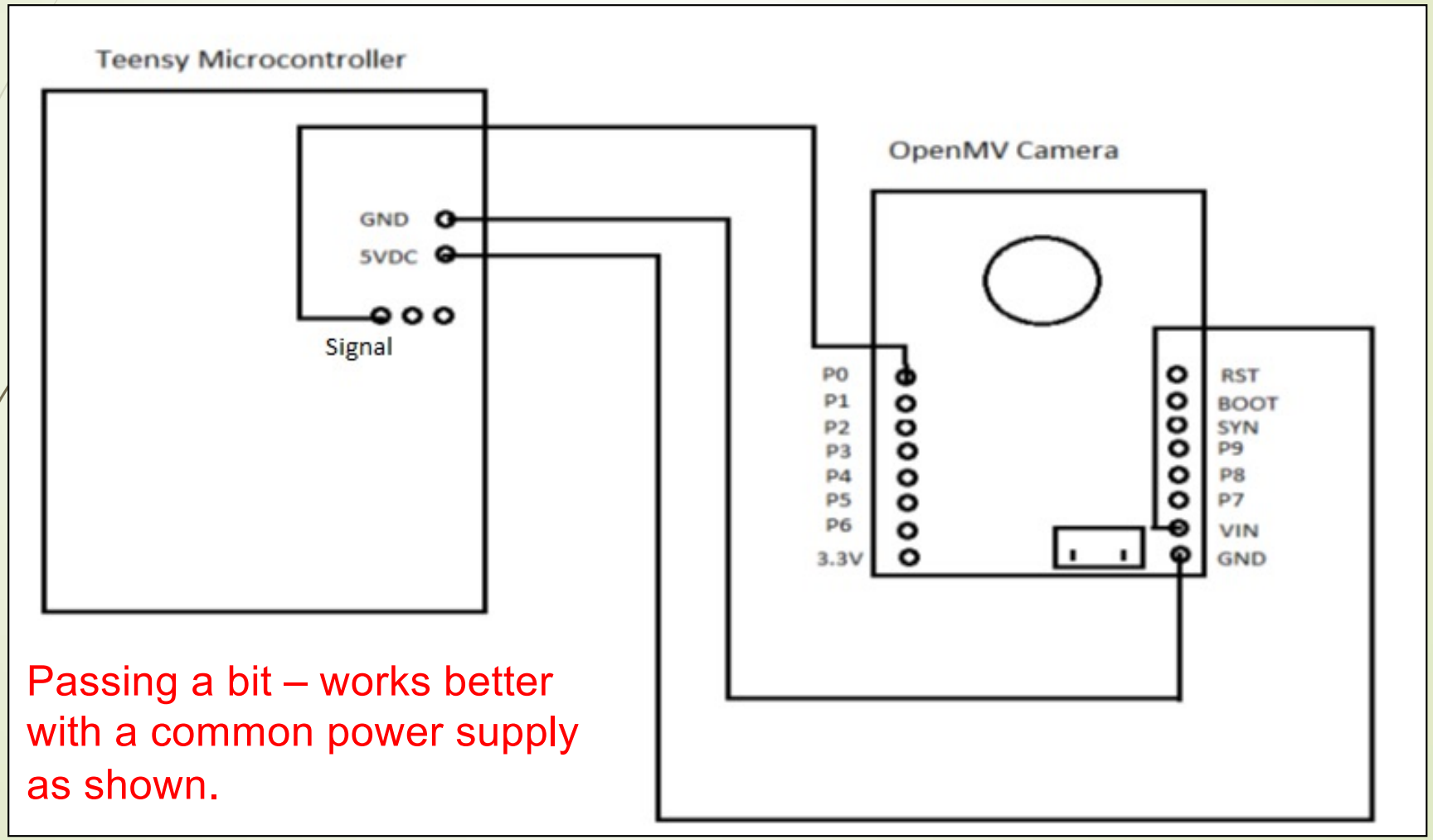
Max current used wo/ µSD card < 150 mA
Max current used w/ µSD card < 250 mA

Micro SD Slot
SD < 2GB Max
SDHC < 32GB Max

Peripherals / Timers		CPU Name	Pin Name
UART 1 RX	SPI 2 MOSI	PB15	P0
UART 1 TX	SPI 2 MISO	PB14	P1
CAN2 TX	SPI 2 SCLK	PB13	P2
CAN2 RX	SPI 2 SS	PB12	P3
I2C 2 SCL	UART 3 TX	PB10	P4
I2C 2 SDA	UART 3 RX	PB11	P5
DAC	ADC	PA5	P6
3.3V Rail (250 mA supply Max)			

Pin Name	CPU Name	Peripherals / Timers		
Reset (Connect to GND to reset)				
BOOT 0 (Connect to 3.3V for DFU mode)				
Frame Sync (use to frame sync cams)				
P9	PD14	Servo 3	TIM4 CH3	
P8	PD13	Servo 2	TIM4 CH2	I2C4 SDA
P7	PD12	Servo 1	TIM4 CH1	I2C4 SCL
VIN (3.6V - 5V)				
GND Rail				

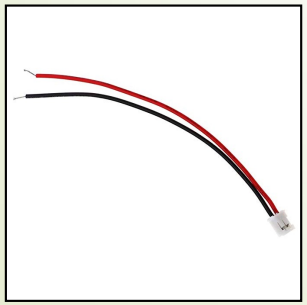
Wiring Diagram



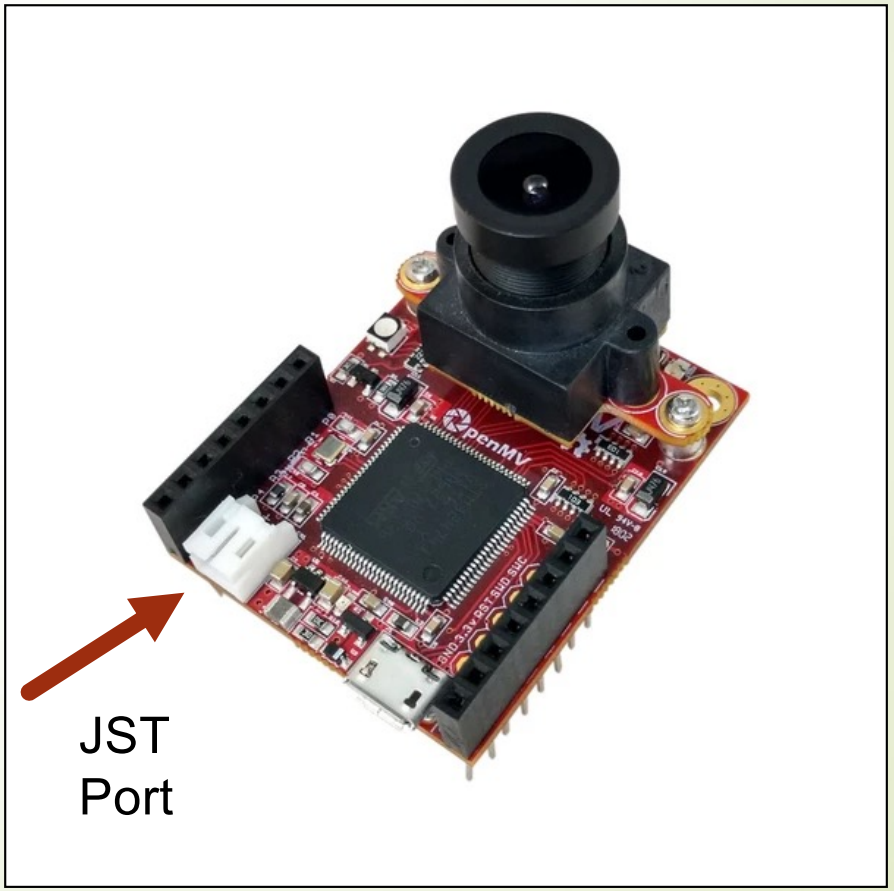
Passing a bit – works better with a common power supply as shown.

Powering the OpenMV H7 Camera from the JST Connector

https://www.amazon.com/gp/product/B01CVJC8I4/ref=ppx_yo_dt_b_search_asin_image?ie=UTF8&psc=1



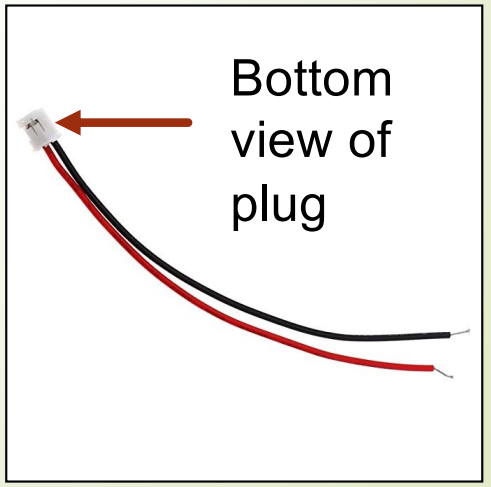
https://www.amazon.com/gp/product/B07RJG81HX/ref=ppx_yo_dt_b_search_asin_image?ie=UTF8&psc=1



https://openmv.io/collections/products

Powering the OpenMV H7 Camera from the JST Connector

- Be careful when using the JST connector as the positive is on the left (OpenMV Cam H7)
- Male Connector that we have (MU) will connect the black wire to + instead of red.



https://www.amazon.com/gp/product/B07RJG81HX/ref=ppx_yo_dt_b_search_asin_image?ie=UTF8&psc=1

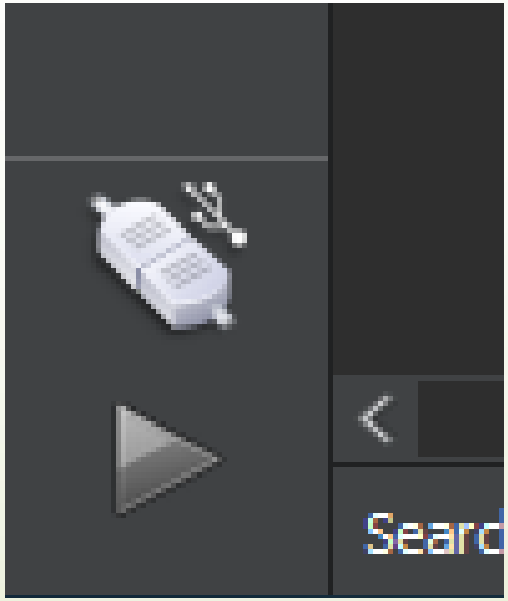
OpenMV IDE

- To install the OpenMV IDE use this link and download the correct IDE for your computer.

- https://openmv.io/pages/download/

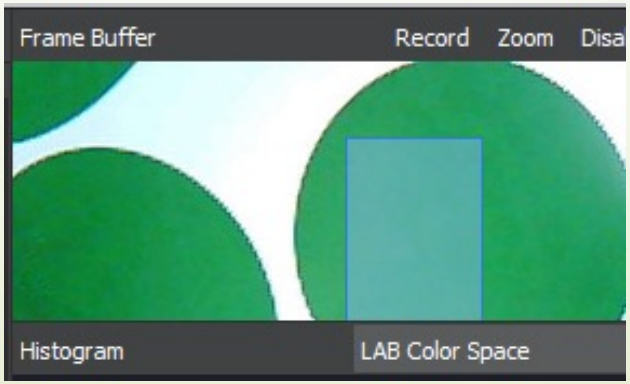
OpenMV IDE

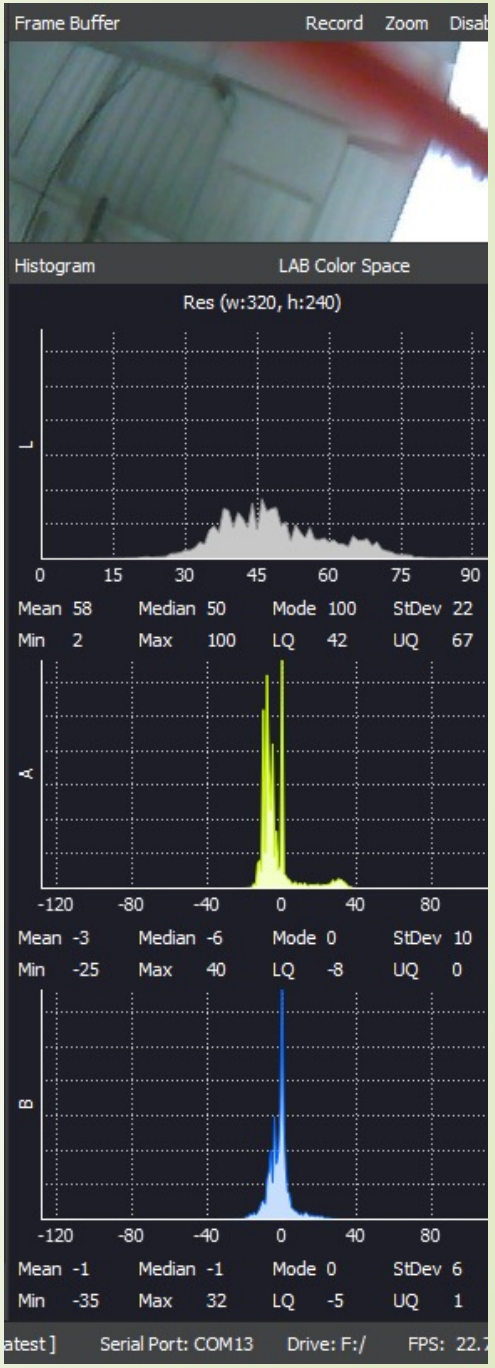
- Connect and disconnect to the camera using the Connect button.
- Begin running the camera with the play button.



Colors

- Monitor screen
- To select a color, make a rectangle around the desired color.
- Use the LAB graphs to get values for the color thresholds.





Code Snippet for MicroPython (Color Detection)

Vision Code — Line: 23,Col: 45

```
# Vision Code - By: iansmith (Actually Dr. Wright but you know) - Tue Sep 28 2021
#This code was adapted from examples given in the OpenMV IDE on color tracking and outputs.
# This example shows off single color code tracking using the OpenMV Cam.
#A color code is a blob composed of two or more colors.
#The example below will only track colored objects which have both the colors below in them.

import sensor, image, time, math, pyb
from pyb import Pin

from pyb import Pin

thresholds = [(15, 35, 40, 80, 20, 40),
              (30, 50, -64, -8, -32, 32)]

sensor.reset()
sensor.set_pixformat(sensor.RGB565)
sensor.set_framesize(sensor.QVGA)
sensor.skip_frames(time = 2000)
sensor.set_auto_gain(False) # must be turned off for color tracking
sensor.set_auto_whitebal(False) # must be turned off for color tracking
clock = time.clock()

# Only blobs that with more pixels than "pixel_threshold" and more area than "area_threshold" are
# returned by "find_blobs" below. Change "pixels_threshold" and "area_threshold"
#if you change the camera resolution. "merge=True" must be set to merge overlapping color blobs for color codes.

while(True):
    clock.tick()
    img = sensor.snapshot()
    for blob in img.find_blobs(thresholds,
pixels_threshold=100, area_threshold=100, merge=True):

        if blob.code() == 1: # r/g code == (1 << 1) | (1 << 0)
            #1==RED 2==GREEN
            p = pyb.Pin("P0", pyb.Pin.OUT_PP)
            p.high() # or p.value(1) to make the pin high (3.3V)
        if blob.code() == 2: # r/g code == (1 << 1) | (1 << 0)
            p = pyb.Pin("P0", pyb.Pin.OUT_PP)
            p.low() # or p.value(0) to make the pin low (0V)
```

Code Snippet for MicroPython (Color Detection)

```
# Color Tracking Thresholds (L Min, L Max, A Min, A Max, B Min, B  Max) The below thresholds track in general red/green things. You may wish to tune them... First is generic_red_thresholds. Second is generic green.


thresholds = [(15, 35, 40, 80, 20, 40), (30, 50, -64, -8, -32, 32)]

sensor.reset()

sensor.set_pixformat(sensor.RGB565)

sensor.set_framesize(sensor.QVGA)

sensor.skip_frames(time = 2000)

sensor.set_auto_gain(False) # Must be turned off for color tracking

sensor.set_auto_whitebal(False) # Must be turned off for color tracking

clock = time.clock()
```

Code Snippet for MicroPython (Color Detection)

```
# Only blobs that with more pixels than "pixel_threshold" and more area than
"area_threshold" are returned by "find_blobs" below. Change "pixels_threshold" and
"area_threshold" if you change the camera resolution. "merge=True" must be set to
merge overlapping color blobs for color codes.

while(True):

    clock.tick()

    img = sensor.snapshot()

    for blob in img. find_blobs(thresholds, pixels_threshold=100,
    area_threshold=100, merge=True):

        if blob.code() == 1: #1==RED

            p = pyb.Pin("P0", pyb.Pin.OUT_PP)

            p.high() # or p.value(1) to make the pin high (3.3V)

        if blob.code() == 2: #2==GREEN

            p = pyb.Pin("P0", pyb.Pin.OUT_PP)

            p.low() # or p.value(0) to make the pin low (0V)
```

Teensy Code (C++) Code Snippet for Reading a Bit passed by the H7 Cam

```
const int Cam = 3;              //Where sensor is connected on board
int val = 0;                    //Set val to zero (initialized value)


void setup() {
  pinMode(Cam, INPUT);          //Set the pin direction to input
  Serial.begin(9600);           //Establish serial buad rate
}
void loop() {
  val=digitalRead(Cam);         //Read value from sensor
  Serial.println(val);          //Print to monitor
  delay(1000);                  //Wait 200ms
}
```

Tips

- MicroPython uses indentations as part of the syntax, unlike C++.
- The threshold values in the MicroPython code can be changed to match different colors.
- To save code to the camera click Tools, Save open script to OpenMV H7 Cam.

References

- https://openmv.io/collections/products/products/openmv-cam-h7
- https://docs.openmv.io/openmvcam/tutorial/overview.html
- https://openmv.io/products/openmv-cam-h7