Building Machine Vision Applications using OpenMV

Class 2: Writing our First OpenMV Application

June 9, 2020

Jacob Beningo

DesignNews

CEC CONTINUING EDUCATION CENTER

Presented by: Digi-Key ELECTRONICS

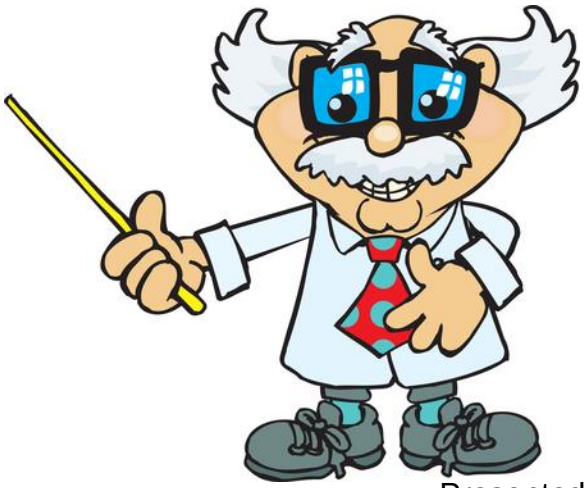
Course Overview

Topics:

- Introduction to Machine Vision and OpenMV
- **Writing our First OpenMV Application**
- Working with the OpenMV I/O
- Utilizing Machine Learning to Detect Objects
- Designing a Machine Vision Application

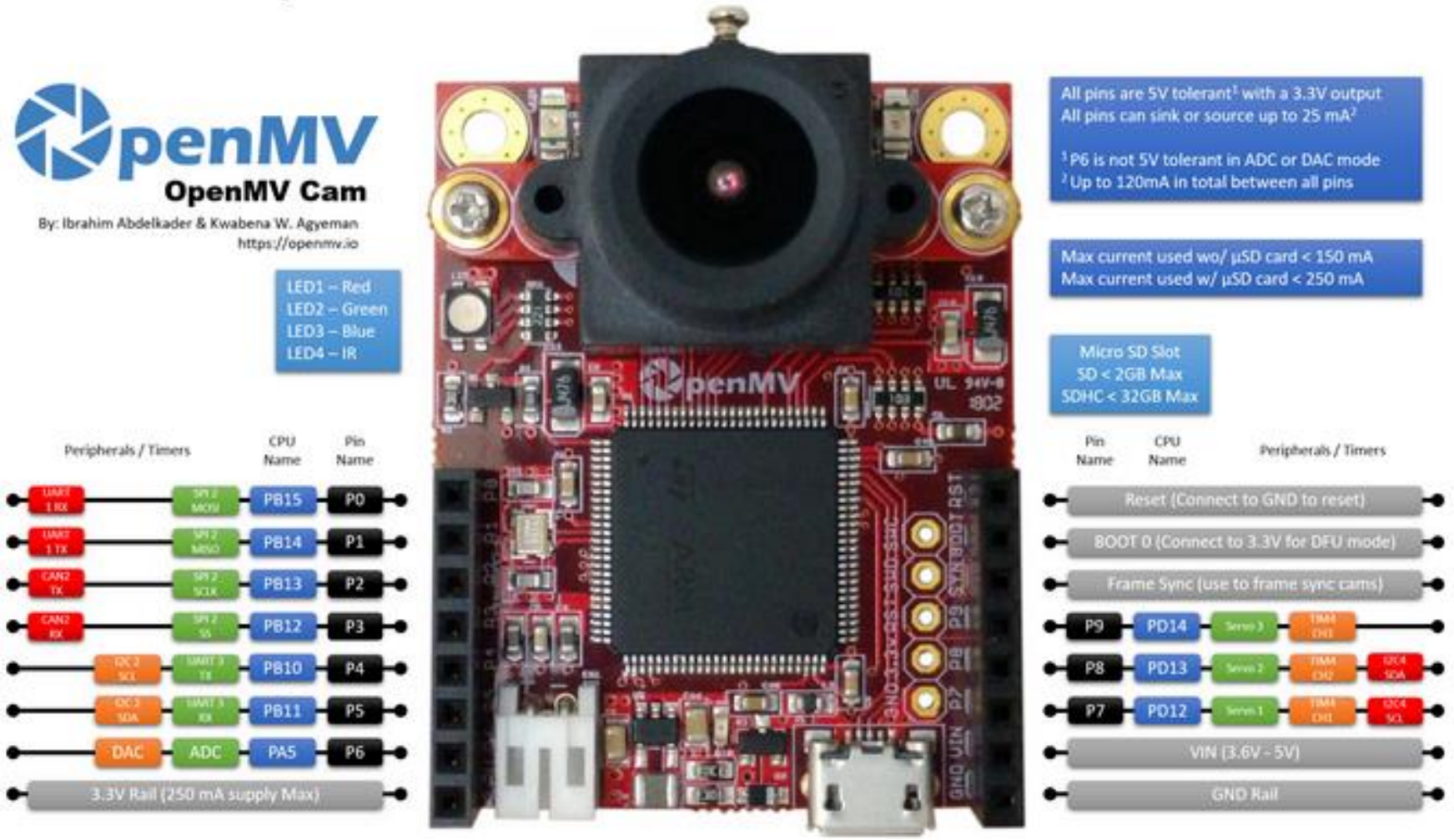
Session Overview

- OpenMV Cam H7 Pin-Outs
- OpenMV Cam H7 LEDs
- Hello World Script
- Adjusting Image Resolution
- Adjusting pixel mode
- Detecting a Circle

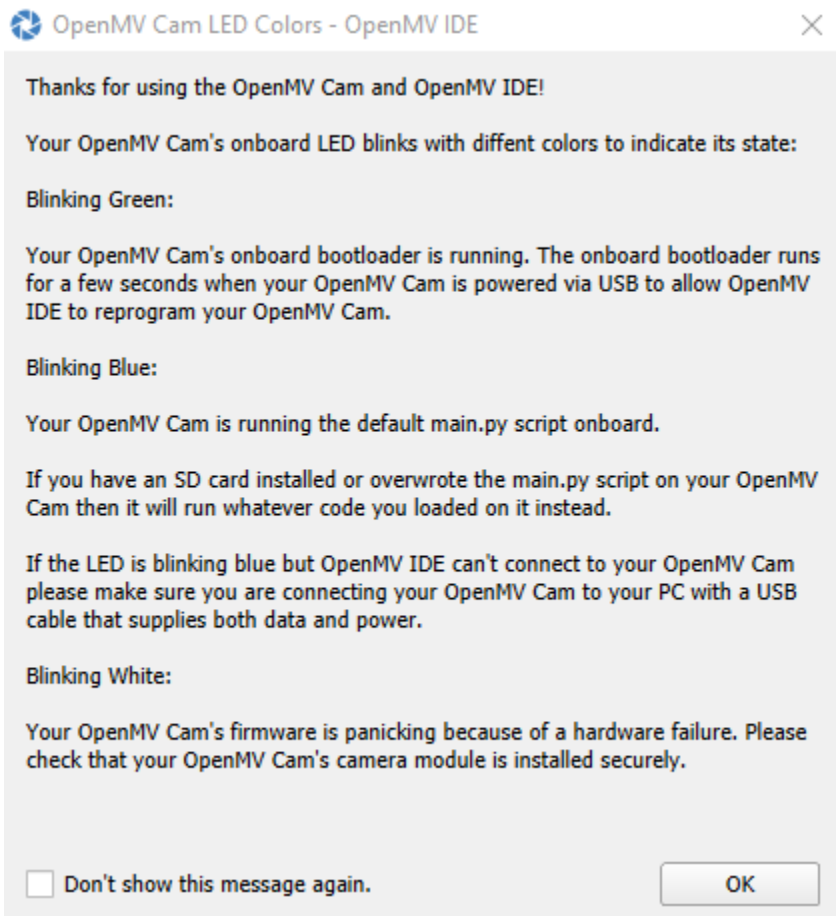


Presented by:

OpenMV Cam H7 Pin-Out



OpenMV Cam H7 LEDs



“Hello World” Script

Script format follows a typical Python format:

import …

…

One-time initialization

…

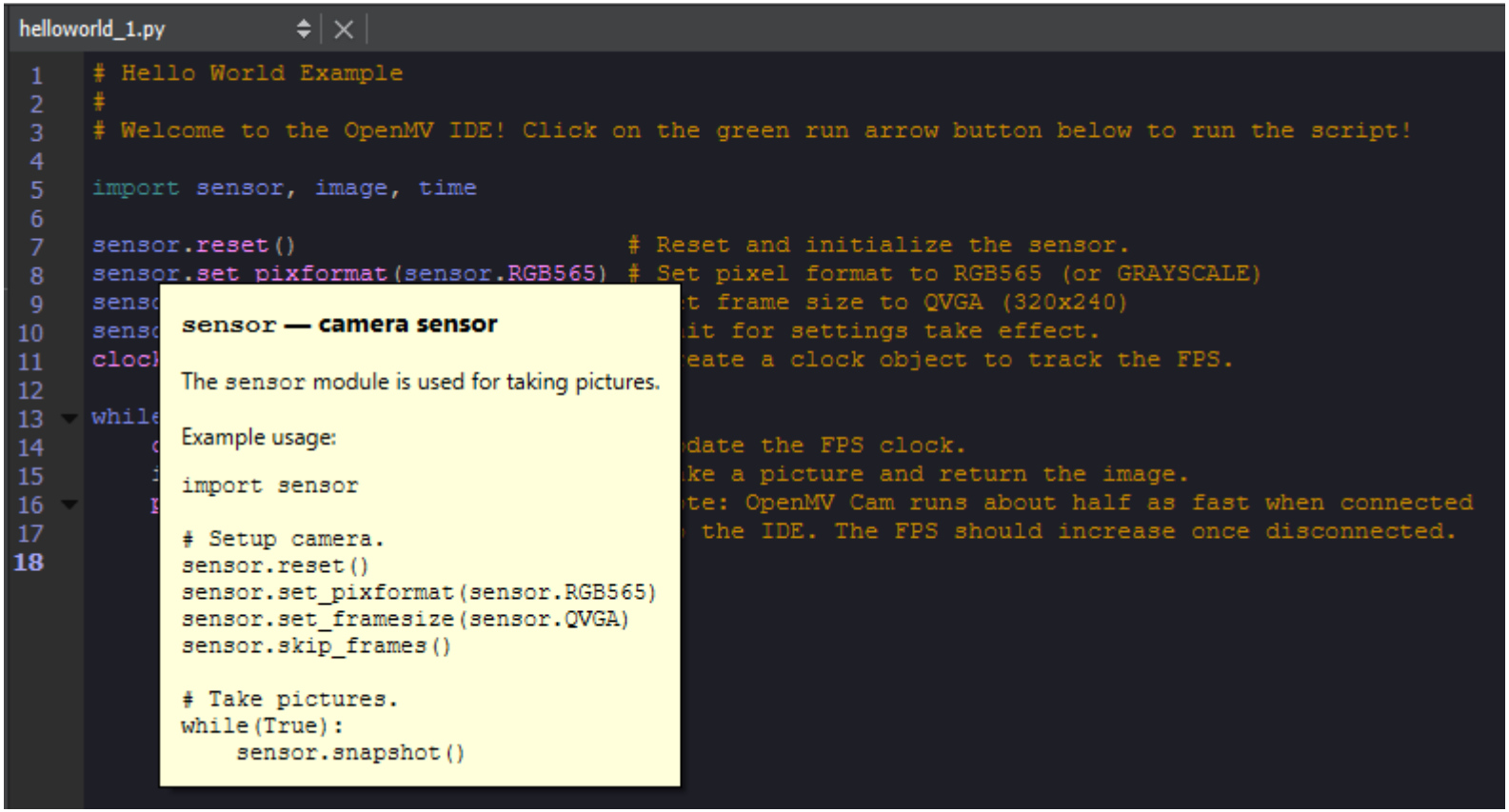
while(True):

…

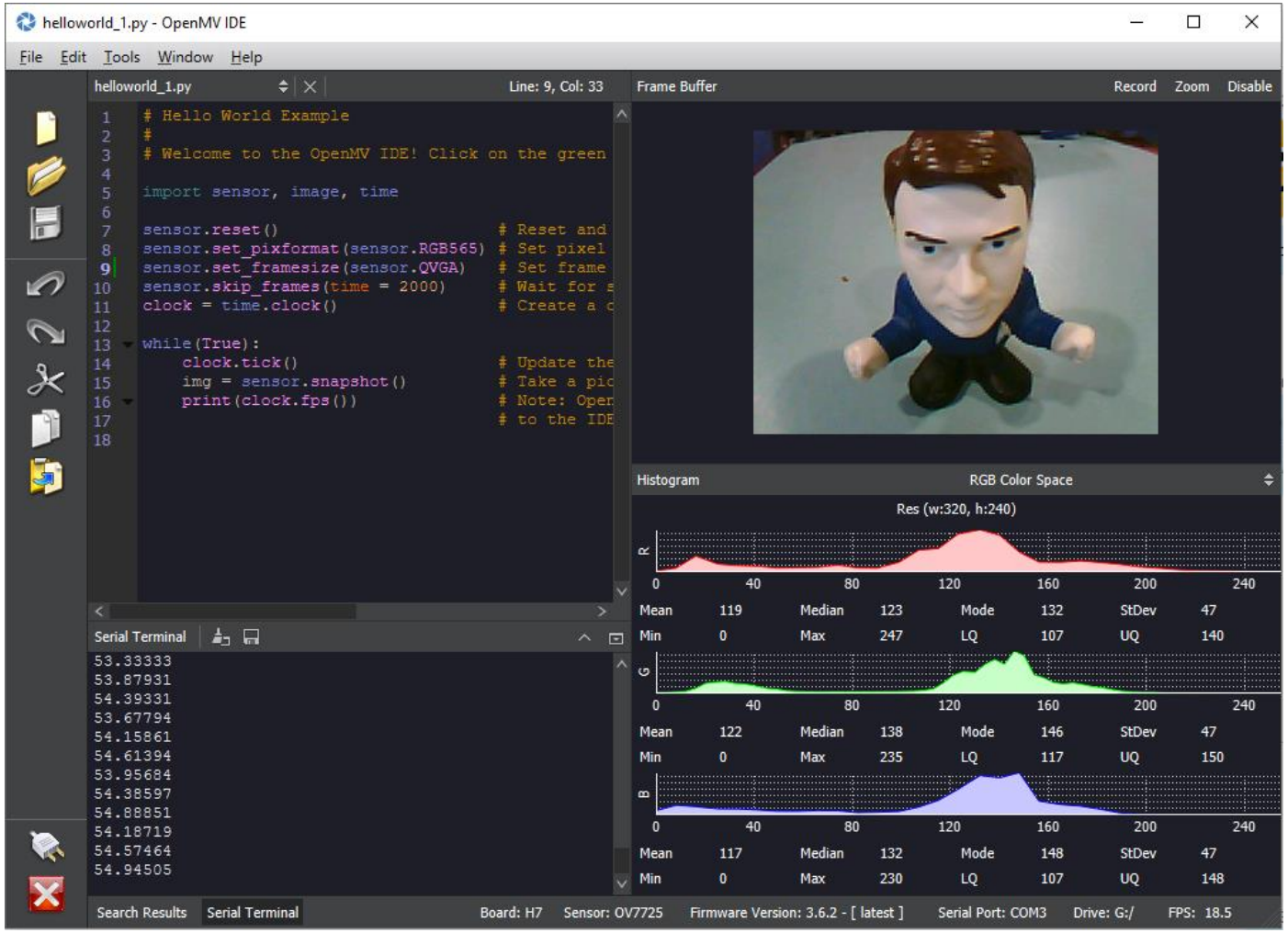
“Hello World” Script

```
# Hello World Example
#
# Welcome to the OpenMV IDE! Click on the green run arrow button below to run the script!

import sensor, image, time

sensor.reset()                      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_framesize(sensor.QVGA)   # Set frame size to QVGA (320x240)
sensor.skip_frames(time = 2000)     # Wait for settings take effect.
clock = time.clock()                # Create a clock object to track the FPS.

while(True):
    clock.tick()                    # Update the FPS clock.
    img = sensor.snapshot()         # Take a picture and return the image.
    print(clock.fps())              # Note: OpenMV Cam runs about half as fast when connected
                                    # to the IDE. The FPS should increase once disconnected.
```

“Hello World” Script



“Hello World” Script



Adjusting Image Resolution

Image Resolution with the OV7725

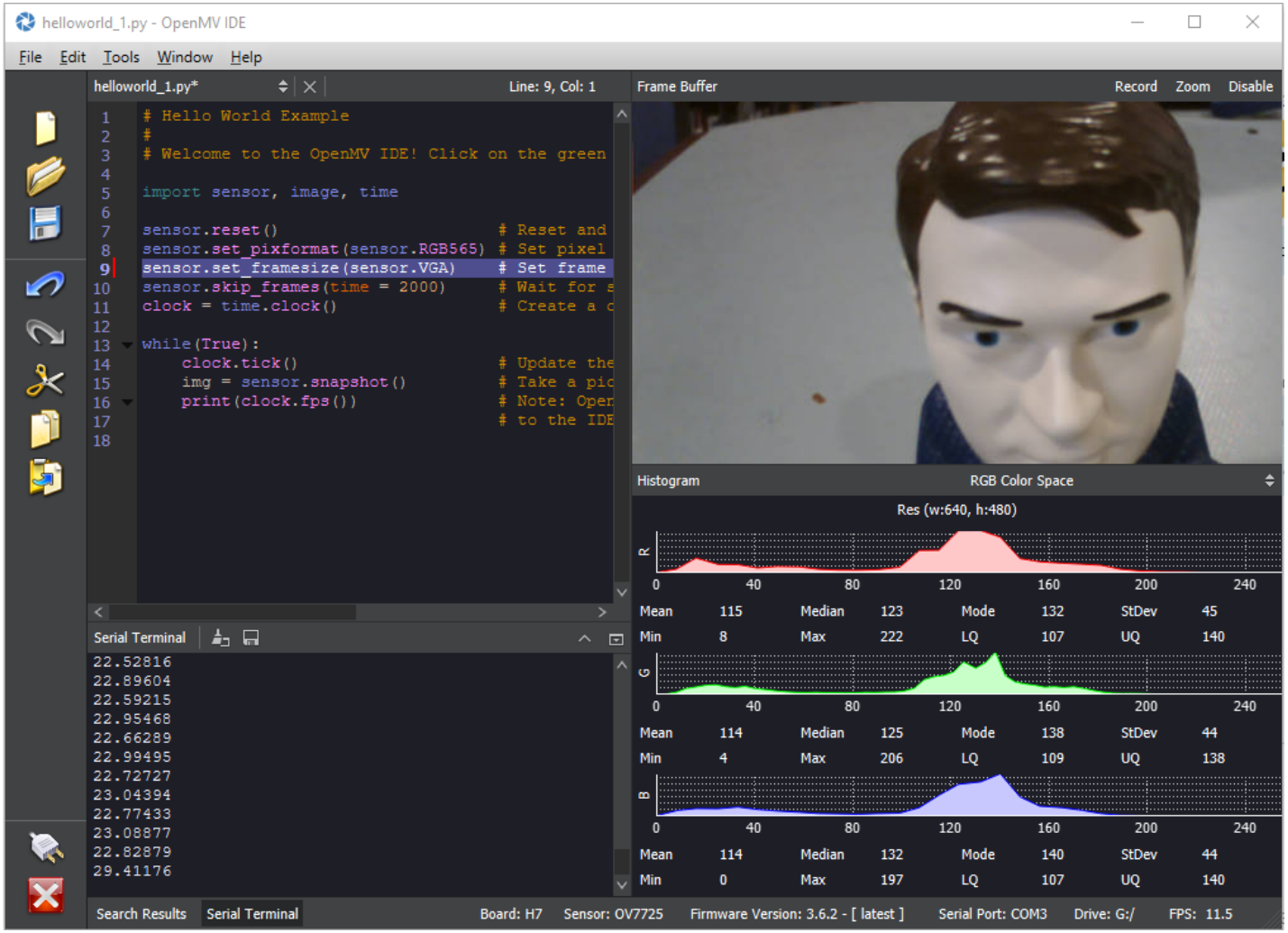
- Default image size is 320x240 (QVGA)
- Adjust image size to 640x480

sensor.set_framesize(sensor.QVGA) *# Set frame size to QVGA (320x240)*

To

sensor.set_framesize(sensor.VGA) *# Set frame size to QVGA (640x480)*

Adjusting Image Resolution



Adjusting the Grayscale

Image pixel format set to RGB565

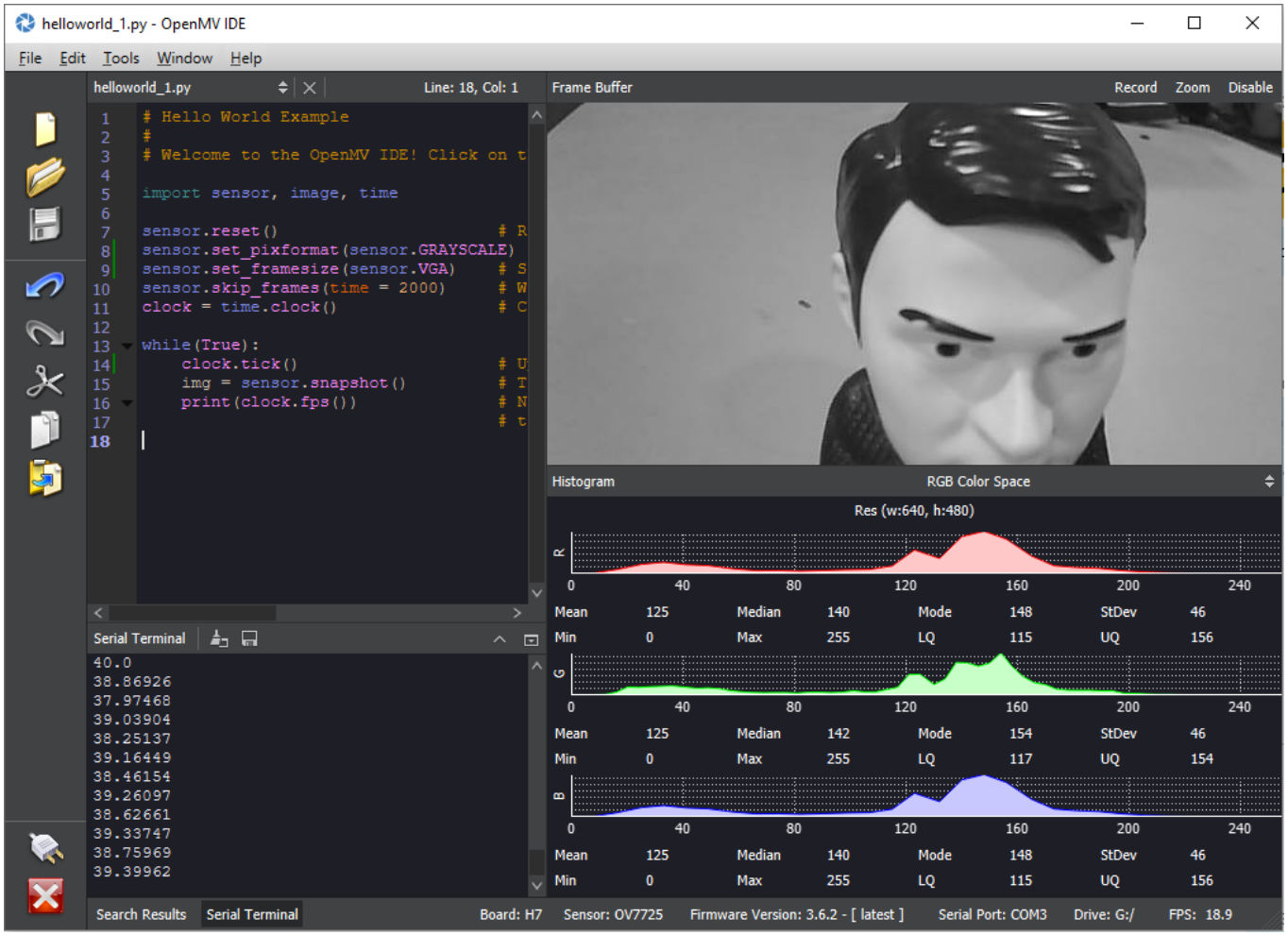
- Default pixel format is 16-bit (RGB565)
- Adjust pixel format to 8-bit (Grayscale)

```
sensor.set_pixformat(sensor.RGB565)       # Set pixel format to RGB565
```

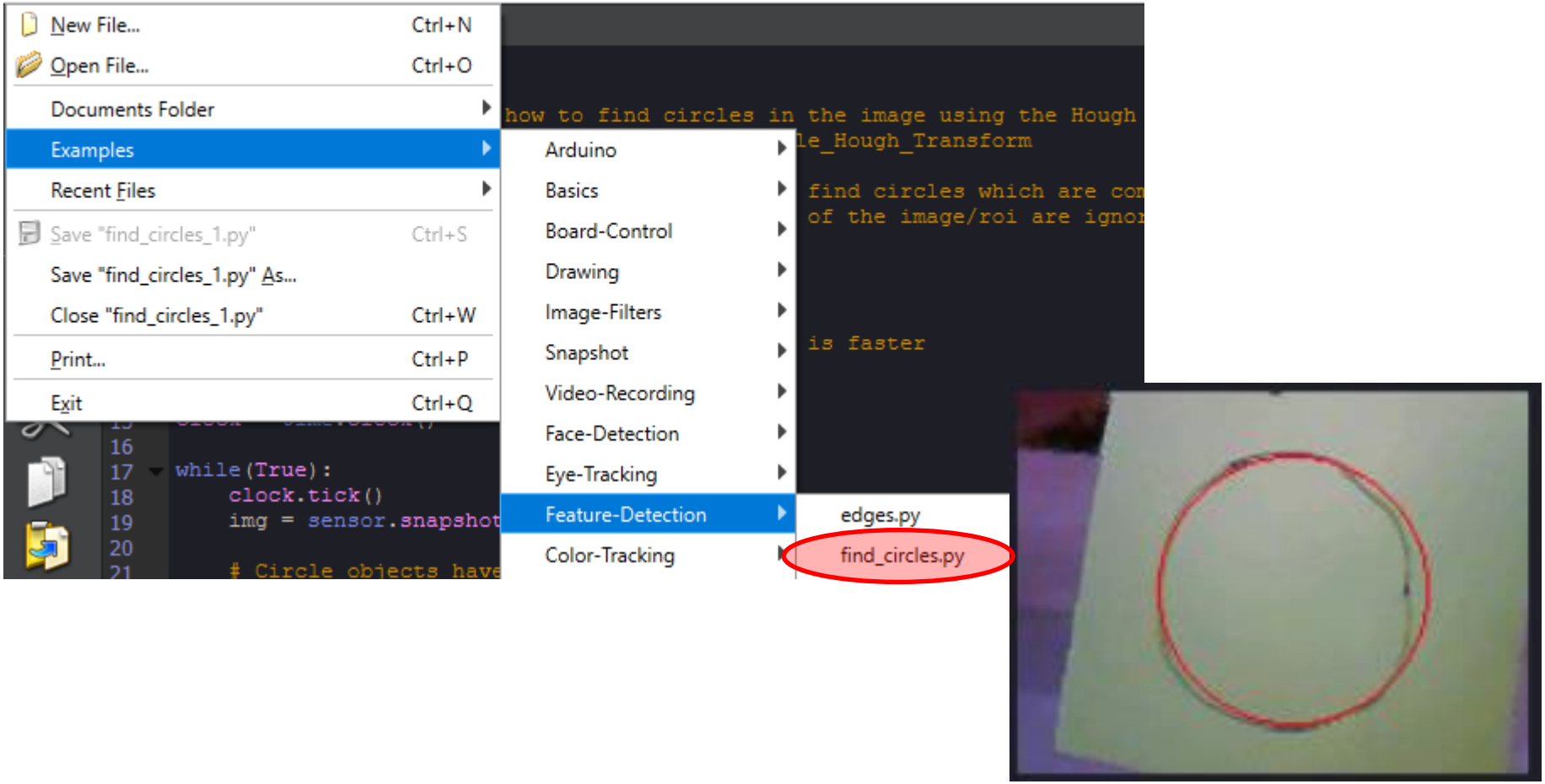
To

```
sensor.set_pixformat(sensor.GRAYSCALE)  # Set pixel format to (GRAYSCALE)
```

Adjusting the Grayscale



Detecting a Circle



Detecting a Circle

```
import sensor, image, time

sensor.reset()
sensor.set_pixformat(sensor.RGB565) # grayscale is faster
sensor.set_framesize(sensor.QQVGA)
sensor.skip_frames(time = 2000)
clock = time.clock()

while(True):
    clock.tick()
    img = sensor.snapshot().lens_corr(1.8)

    for c in img.find_circles(threshold = 2000, x_margin = 10, y_margin = 10, r_margin = 10,
            r_min = 2, r_max = 100, r_step = 2):
        img.draw_circle(c.x(), c.y(), c.r(), color = (255, 0, 0))
        print(c)

    print("FPS %f" % clock.fps())
```

Detecting a Circle

```
import sensor, image, time, pyb

sensor.reset()
sensor.set_pixformat(sensor.RGB565) # grayscale is faster
sensor.set_framesize(sensor.QQVGA)
sensor.skip_frames(time = 2000)
clock = time.clock()
ledRed = pyb.LED(1)

while(True):
   clock.tick()
   img = sensor.snapshot().lens_corr(1.8)

   for c in img.find_circles(threshold = 2000, x_margin = 10, y_margin = 10, r_margin = 10,
         r_min = 2, r_max = 100, r_step = 2):
      img.draw_circle(c.x(), c.y(), c.r(), color = (255, 0, 0))
      ledRed.on()
      print("Circle Detected!")
      pyb.delay(500)
      ledRed.off()
```

Additional Resources

- Beningo.com
  - Blog, White Papers, Courses
  - Embedded Bytes Newsletter
    - http://bit.ly/1BAHYXm
- OpenMV.io

From www.beningo.com under

- Blog > CEC – Building Machine Vision Applications using OpenMV