

Building Machine Vision Applications using OpenMV

Class 1: Introduction to Machine Vision and OpenMV

June 8, 2020
Jacob Beningo

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

The Lecturer – Jacob Beningo



Jacob Beningo

President



Social Media / Contact

E : jacob@beningo.com

T : 810-844-1522

Twitter : Jacob_Beningo

f : Beningo Engineering

in : JacobBeningo

EDN : Embedded Basics

ARM Connected Community

Consulting

- Advising
- Coaching
- Content
- Consulting
- Training

www.beningo.com

Jacobs CEC Courses

CEC 2013 – 2016

Fundamentals of Embedded Software (2013)

Mastering the Software Design Cycle (2014)

Python for Embedded Systems(2014)

Software Architecture Design (2014)

Baremetal C (2015)

Mastering the ARM Cortex-M Processor (2015)

Writing Portable and Robust Firmware in C (2015)

Design Patterns and the Internet (2015)

Bootloader Design for MCUs (2016)

Rapid Prototyping w/ Micro Python (2016)

Debugging (2016)

Professional Firmware (2016)

CEC 2017 - 2018

API's and HAL's
February 2017

Baremetal to RTOS
April 2017

Designing IoT Sensor Nodes
July 2017

From C to C++
October 2017

Connecting Edge Devices
(March 2018)

Building an IoT Connected PLC (April 2018)

Securing IoT Devices using Arm TrustZone (Nov 2018)

Minimizing Defects
(Dec 2018)

CEC 2019

Machine Learning for Embedded (April 2019)

Designing Embedded Systems using MicroPython

Launching a Product
(Nov 2019)

CEC 2020

Getting Started with Secure Software

Building Machine Vision Applications using OpenMV

Techniques for Interfacing with Modern Sensors (October)

Designing Embedded Systems using the ESP32 (December)

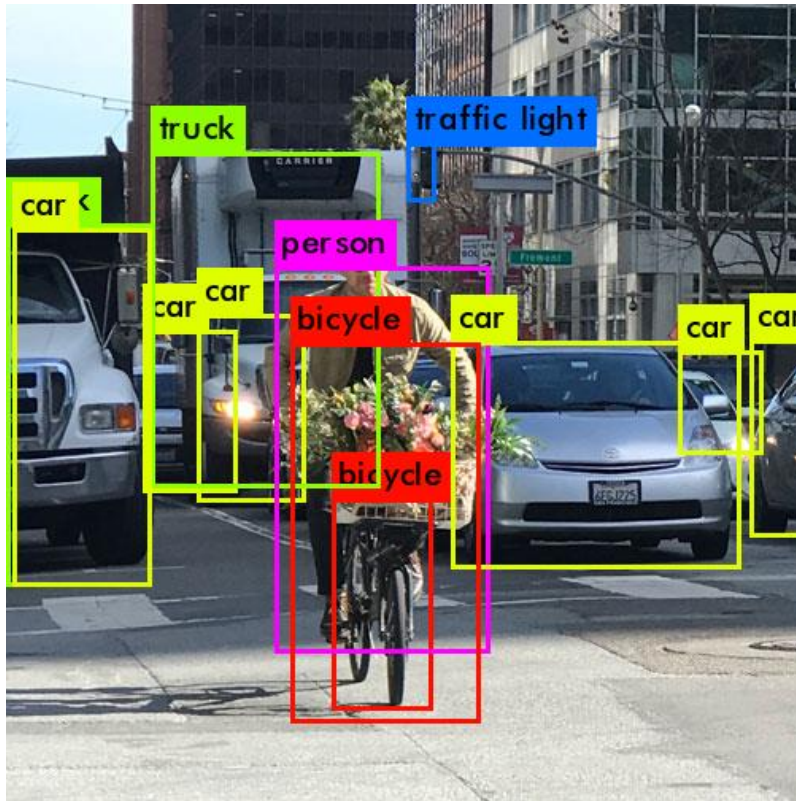
Session Overview

- Machine Vision in Embedded
- Machine Vision Architectures
- OpenMV Cam H7
- OpenMV IDE



Presented by:

Machine Vision in Embedded



Source: Papers with Code



Source: Target.com

Machine Vision



Image Source:
Kwiksource.com



Image Source:
Safewise

Machine Vision



Image Source:
Kwiksource.com



Image Source:
Safewise

Machine Vision



Source: aimee.io

Machine Vision Architectures



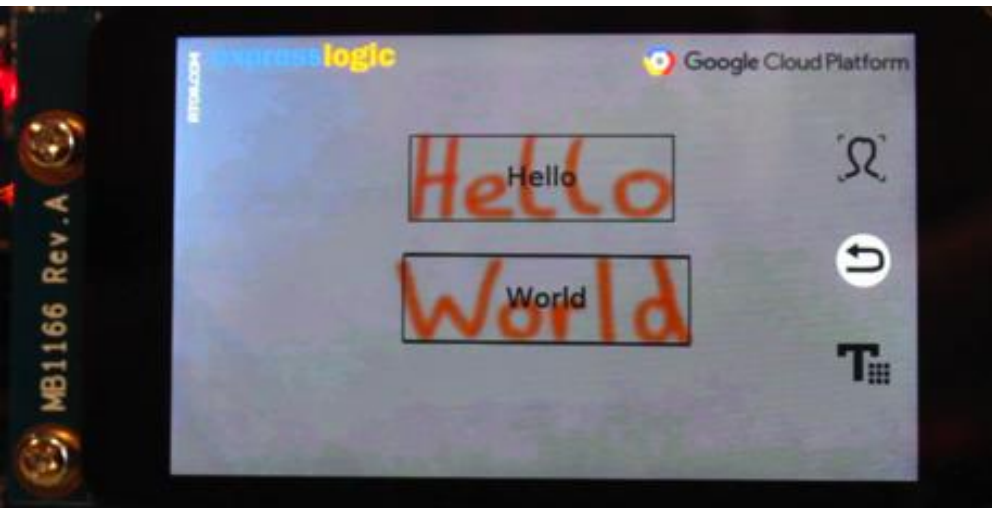
Machine Vision Architectures

What can an online API do?

- Insights from images
 - 2 Cats
 - 1 Dog
 - 1 Squirrel
- Extract text
 - Optical Character Recognition (OCR)
- Face detection
- Content moderation
 - SafeSearch



Machine Vision Architectures

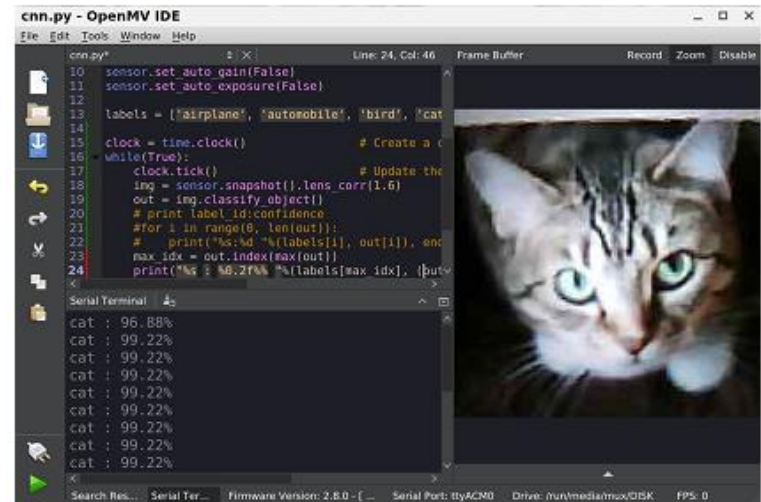


```
Output: Log file: Out
DHCP client running...
IP address: 10.0.0.221
Network mask: 255.255.255.0
Google Vision API is ready
DNS Lookup for GOOGLE APIs Domain:vision.googleapis.com
GOOGLE APIs Domain IPv4 address: 172.217.0.10
TCP session established.
TLS session established.
Text[(258, 111), (541, 111), (541, 203), (258, 203)]: Hello
Text[(252, 240), (558, 242), (558, 336), (252, 334)]: World
```


Machine Vision Architectures



[OpenMV Cam with a Cortex-M7](#)



[Video to Watch](#)

OpenMV Cam H7

OpenMV is a project that brings machine vision into a simple Arduino like environment for “simple” applications.

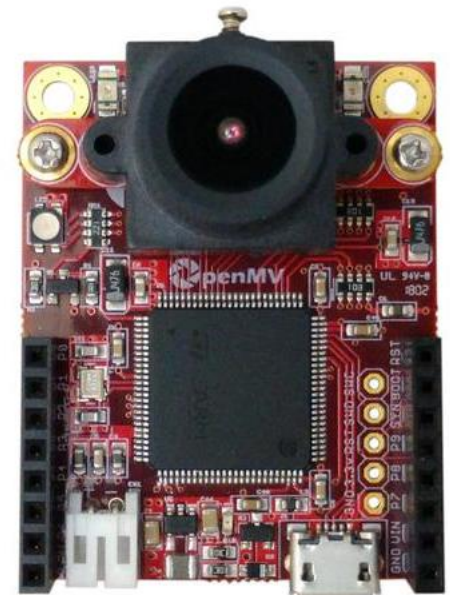
- Can write scripts in Python
- Expandable I/O for interfacing
- Built-in camera module
- Expandable camera options
- Custom IDE for developing and deploying scripts



Source: openmv.io

OpenMV Cam H7

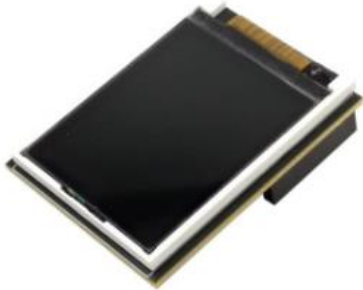
- Arm Cortex-M7 Processor
 - STM32H743VI
 - 400 MHz
 - 1MB RAM
 - 2 MB Flash
- OV7725 image sensor
 - 640x480 16-bit RGB565 @ 60 FPS
 - 640x480 8-bit Grayscale images
- 2.8 mm lens



Source: openmv.io

OpenMV Cam H7 Expansions

LCD



Wi-Fi



Servo



Pan and Tilt



Motor



Flit Camera Adapter



OpenMV Cam H7 Availability



OpenMV Cam H7 – Getting Started

- Use a screwdriver to remove the two-lens mount screw from the lens mount on your OpenMV Cam.
- Apply some isopropyl alcohol to a small part of the cloth.
- Rub the wet part of the cloth on the camera IC gently. Any dirt spots on the camera IC will be microscopic to the human eye so just try to generally rub down the top of the camera IC. Note that the top of the camera IC is glass.
- After cleaning the camera IC make sure the alcohol has evaporated completely and that no cloth strands were left behind. Note that we're using isopropyl alcohol versus water since isopropyl alcohol evaporates quickly and doesn't leave anything behind.
- Use the screwdriver to re-attach the lens mount. Make sure that the set screw on the lens mount points off the top/back of the OpenMV Cam.

Source: openmv.io

The OpenMV IDE

The screenshot displays the OpenMV IDE interface. The main window shows a Python script named 'helloworld 1.py' with the following code:

```
1 import sensor, image, time
2
3 # Setup Camera
4 sensor.reset()
5 sensor.set_pixformat(sensor.RGB565)
6 sensor.set_framesize(sensor.QQVGA)
7 sensor.skip_frames(10)
8 threshold = (10, 90, -80, -30, 20, 50)
9 clock = time.clock()
10
11 # Find blobs
12 while(True):
13     clock.tick()
14     img = sensor.snapshot()
15     for b in img.find_blobs([threshold]):
16         img.draw_rectangle(b[0:4])
17         print("====\nBlob %s" % str(b))
18     print("FPS %d" % clock.fps())
19
```

The right side of the interface shows a live camera feed with several colored blobs (red, blue, green) detected and outlined in white. Below the camera feed is a histogram of the LAB color space, showing the distribution of colors in the image. The histogram is divided into three sections: L (Lightness), A (Red-Green), and B (Blue-Yellow). The L section shows a peak around 80, the A section shows a peak around -17, and the B section shows a peak around -7. The histogram also displays statistical data for each section:

Section	Mean	Median	Mode	StDev	Min	Max	LQ	UQ
L	76	86	88	20	11	100	56	92
A	2	-17	-18	40	-74	81	-21	32
B	7	-5	-7	31	-99	68	-9	29

The bottom of the interface shows the Serial Terminal output, which matches the print statements in the script. The terminal output shows the following:

```
====
Blob (88, 33, 46, 47, 833, 115, 50, 0.898667, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 413, 37, 37, 2.638215, 1, 1)
====
Blob (88, 33, 47, 48, 842, 116, 50, 0.9409514, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 408, 37, 37, 2.625929, 1, 1)
====
Blob (88, 33, 47, 48, 839, 115, 50, 0.9321314, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 410, 37, 37, 2.628374, 1, 1)
====
Blob (88, 33, 46, 47, 854, 116, 50, 0.9656991, 1, 1)
FPS 15
```

The bottom status bar indicates the Firmware Version: 2.0.0 - [latest], Serial Port: COM5, Drive: I:/, and FPS: 15.2.

Source: openmv.io

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