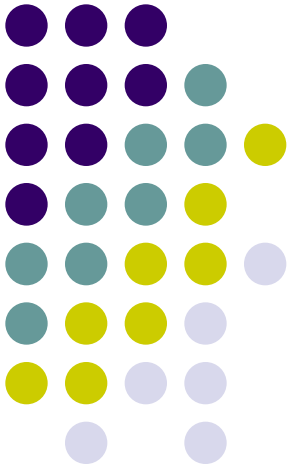
Chapter 14: OpenMV

EE2405

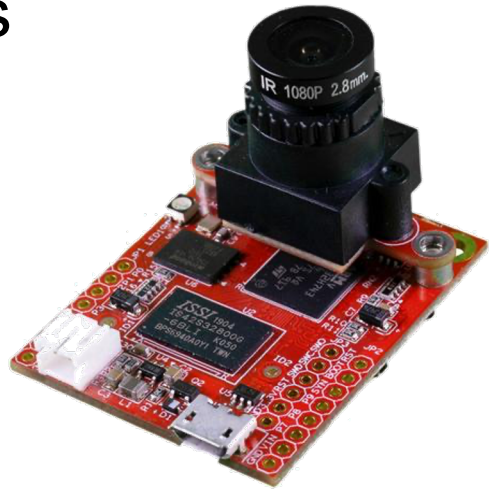
嵌入式系統與實驗

Embedded System Lab

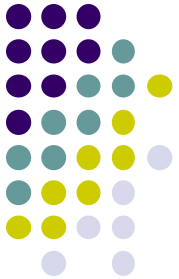


OpenMV Cam H7 Plus

- Based on ARM Cortex M7
- Program with MicroPython
  - A subset of Python
  - With libraries for peripheral drivers
- OV5640
  - 2592x1944 (5MP)
  - 25-50 FPS for simple algorithms

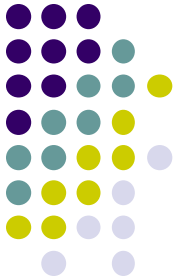


pyb module



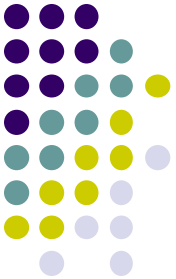
- Specific functions related to the board.
  - pyb.LED(1).off(), pyb.LED(2).on(), and pyb.LED(3).toggle()
  - uart = pyb.UART(3, 115200, timeout_char = 1000)
    uart.write("Hello World\n")
  - p = pyb.Pin("P0", pyb.Pin.IN)
    p.value()
    p = pyb.Pin("P0", pyb.Pin.OUT_PP)
    p.high()
    p.low()
  - adc = pyb.ADC(pyb.Pin('P6'))
    print("%f volts" % (((adc.read() * 3.3) + 2047.5) / 4095))

Take a Picture (MicroPython)



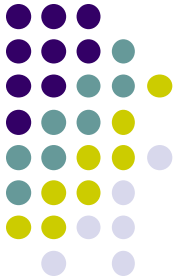
```
import sensor, image, pyb

RED_LED_PIN = 1
BLUE_LED_PIN = 3

sensor.reset() # Initialize the camera sensor.
sensor.set_pixformat(sensor.RGB565) # or sensor.GRAYSCALE
sensor.set_framesize(sensor.QVGA) # or sensor.QQVGA (or others)
sensor.skip_frames(time = 2000) # Let new settings take affect.

pyb.LED(RED_LED_PIN).on()
sensor.skip_frames(time = 2000) # Give the user time to get ready.

pyb.LED(RED_LED_PIN).off()
pyb.LED(BLUE_LED_PIN).on()

print("You're on camera!")
sensor.snapshot().save("example.jpg") # or "example.bmp" (or others)

pyb.LED(BLUE_LED_PIN).off()
print("Done! Reset the camera to see the saved image.")
```

sensor module



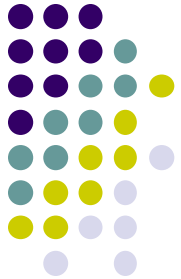
- Functions for taking pictures.
- sensor.set_pixformat(sensor.RGB565)
  - RGB565: 16-bits per pixel.
- sensor.set_framesize(sensor.QVGA)
  - QVGA: 320x240
  - Higher resolution → larger memory and longer processing time
  - Tune resolution to specific application
- sensor.skip_frames(time = 2000)
  - Skip frames for 2000 milliseconds to wait for the format change of the module.

AprilTag Scan



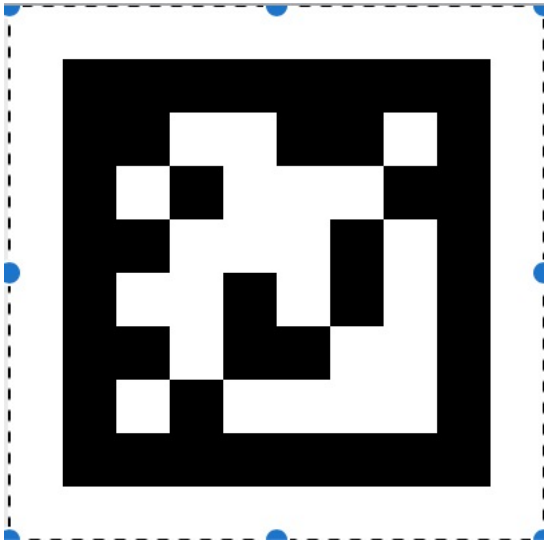
```
import sensor
import image
import time
import math

… # sensor initialization
sensor.set_auto_gain(False)  # must turn this off to prevent image
washout...
sensor.set_auto_whitebal(False)
clock = time.clock()
tag_families = 0
tag_families |= image.TAG36H11  # Default

while(True):
    clock.tick()
    img = sensor.snapshot()
    for tag in img.find_apriltags(families=tag_families):
        img.draw_rectangle(tag.rect(), color=(255, 0, 0))
        img.draw_cross(tag.cx(), tag.cy(), color=(0, 255, 0))
        print_args = ("TAG36H11", tag.id(), (180 * tag.rotation()) /
                      math.pi), tag.x_rotation(), tag.y_rotation(),
tag.z_rotation()
        print("Tag Family %s, Tag ID %d, rotation %f (degrees), at (%f,
%f, %f) angle" % print_args)
    print(clock.fps())
```

image module



- Many functions and objects for machine vision.
- image.lens_corr()
  - Performs lens correction to un-fisheye the image due to the lens distortion.
- image.find_datamatrices()
  - Finds all datamatrices and return objects.
- image.find_apriltags()
  - Finds all apriltags in a region of interests
- image.draw_rectangle(), image.draw_cross()
  - Draw a rectangle and a cross sign

AprilTag class

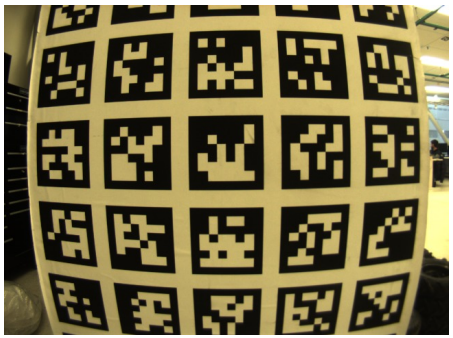


tag36_11_00101

- Distance from camera to AprilTag
  - apriltag.x_translation()
  - apriltag.y_translation()
  - apriltag.z_translation()
  - Need to make a conversion table
  - Measure and make the table to convert AprilTag x, y, z to actual distance.
- Angle from camera to AprilTag
  - apriltag.x_rotation()
  - apriltag.y_rotation()
  - apriltag.z_rotation()
- An AprilTag demo video: https://www.youtube.com/watch?v=keb0B11zj5g

AprilTag Application

- Estimate distance and angles of objects in an image
- Multiple tags can be used jointly to improve the estimation.



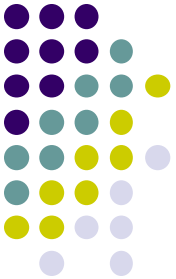
(a) 0.6m (closest)

(b) 7.0m (farthest)

Ref: AprilTag 2: Efficient and robust fiducial detection

Download AprilTag: https://github.com/AprilRobotics/apriltag-imgs/

Image Line Detection



- Find line segments in an image:

```
for l in img.find_line_segments(merge_distance = 0,
max_theta_diff = 5):
        img.draw_line(l.line(), color = (255, 0, 0))
```

- Line detection demo video on a robot car: https://www.youtube.com/watch?v=Pm88BEz3upM

- Learn more about OpenMV libraries https://docs.openmv.io/library/index.html#libraries-specific-to-the-openmv-cam