物聯網與微處理機系統設計
Internet of Things and Microprocessor System Design

Lecture 07 – Camera

Lecturer: 陳彥安 Chen, Yan-Ann

YZU CSE

Partial from: NCTU IoT Course
https://github.com/coldwufish/RaspPI

Outline

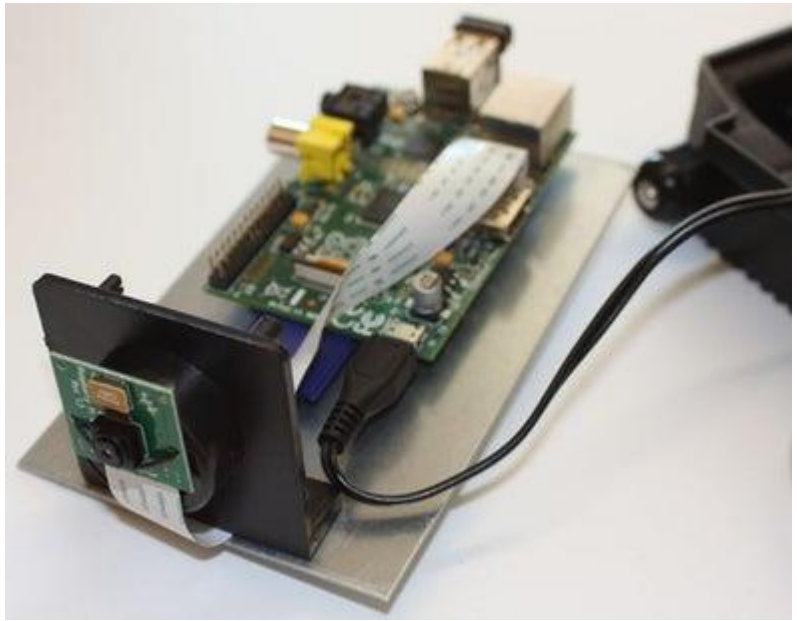
- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

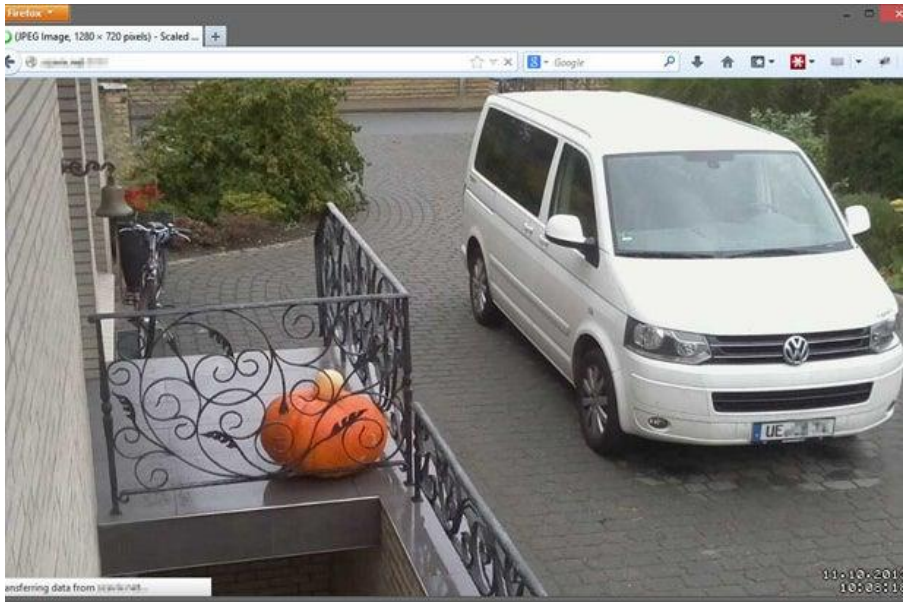
Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Camera Application

- IP Security Camera

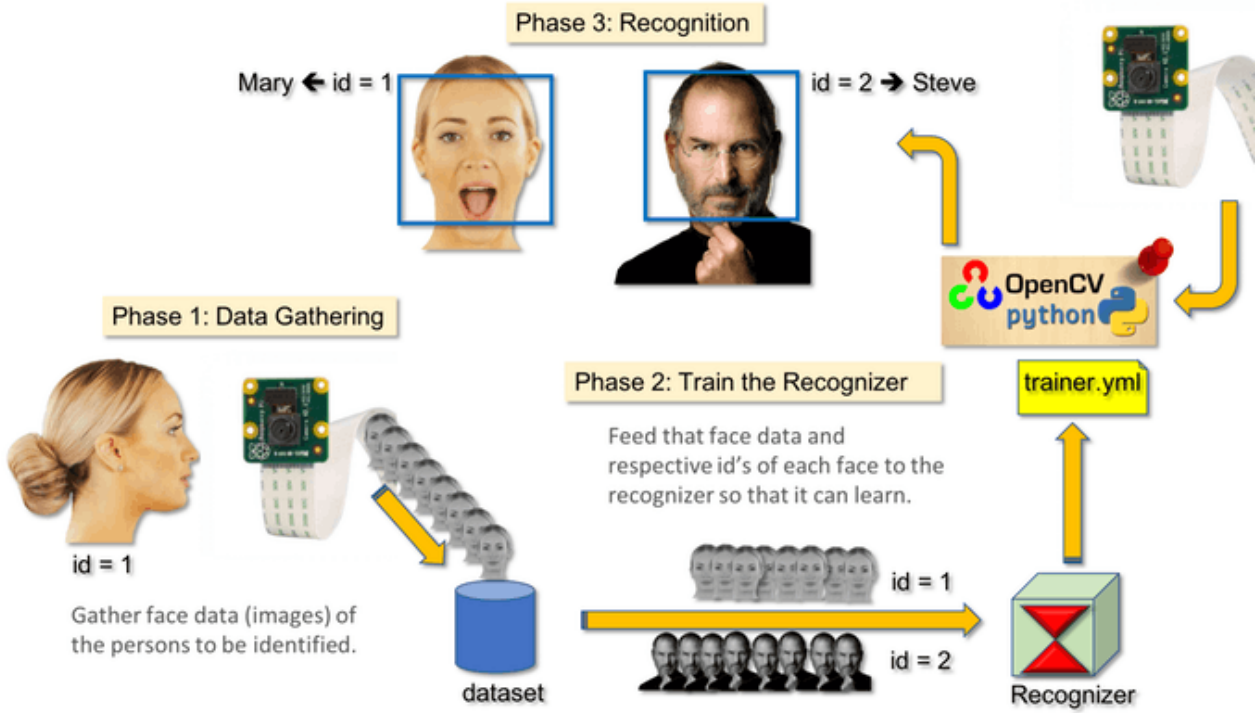


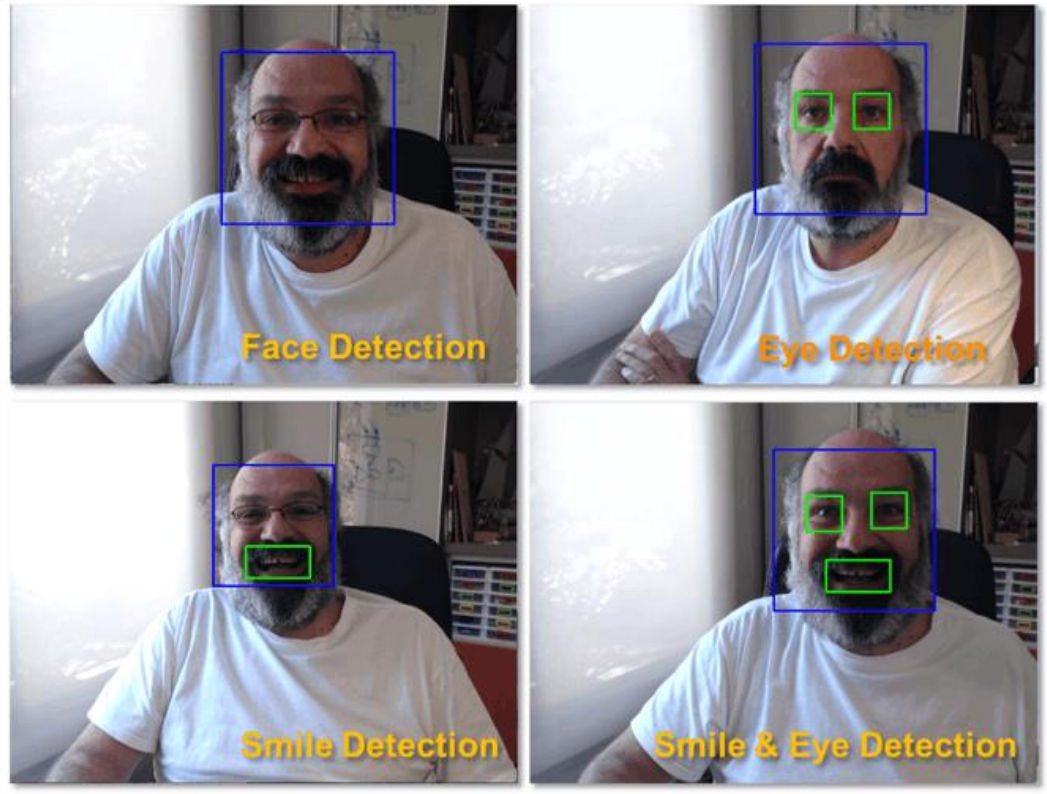


Ref: https://www.instructables.com/Raspberry-Pi-as-low-cost-HD-surveillance-camera/

Camera Application

- Face Recognition

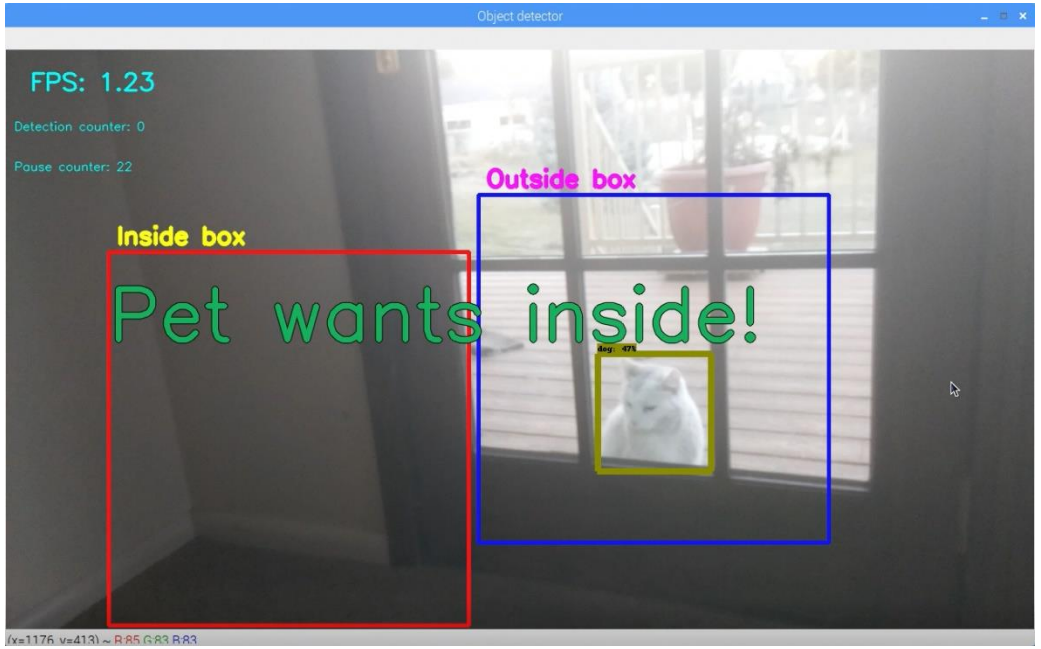




Ref: https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348

Camera Application

- Object detection

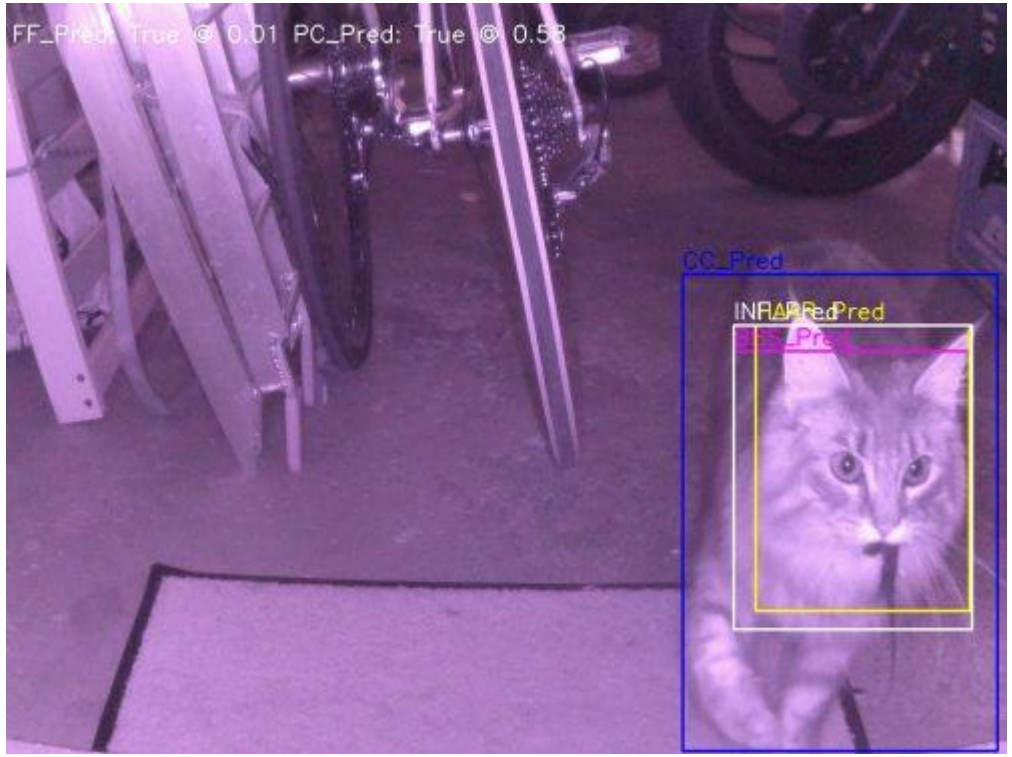


Ref: https://github.com/EdjeElectronics/TensorFlow-Object-Detection-on-the-Raspberry-Pi

Camera Application

- Pet detection

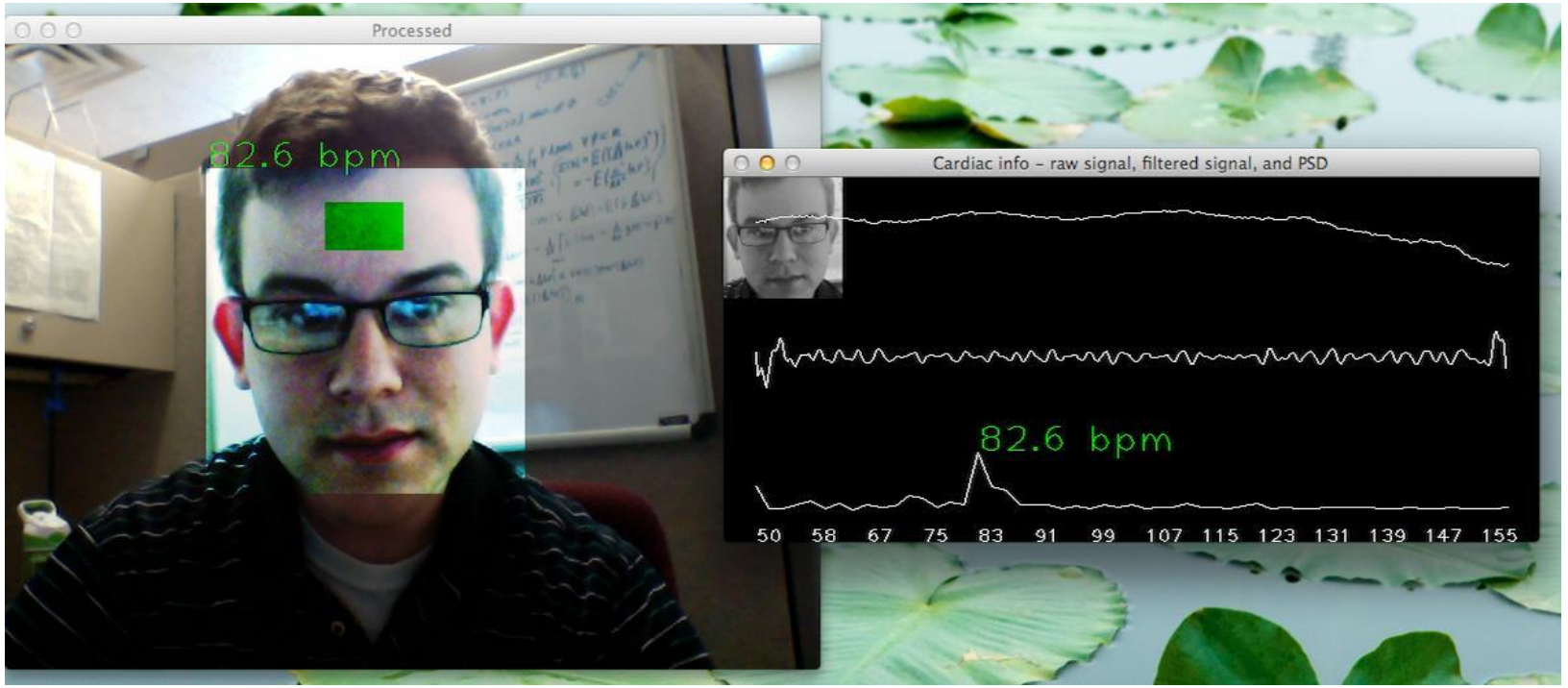
Ref: https://www.saydigi.com/2017/06/catfi-computex-2017.html



Ref: https://www.raspberrypi.org/blog/deep-learning-cat-prey-detector/

Camera Application

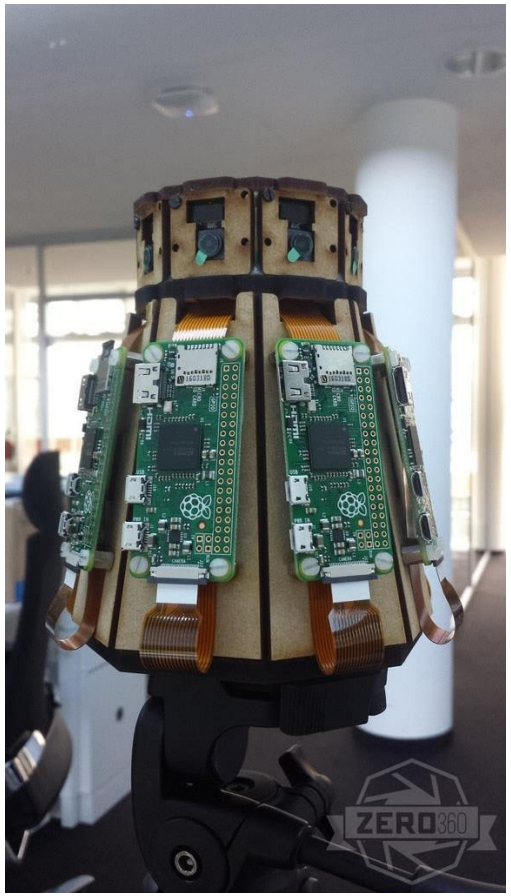
- Pulse detector



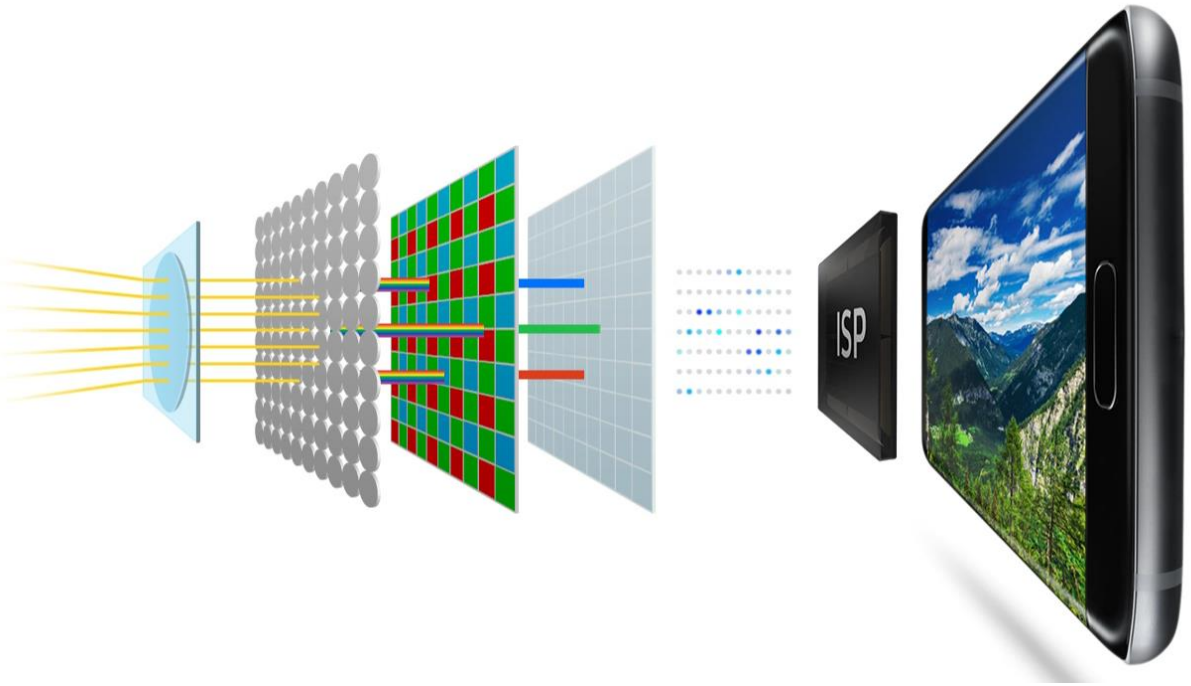
Ref: https://github.com/thearn/webcam-pulse-detector

Camera Application

- 360 Camera



Digital Camera



Ref: https://www.samsung.com/semiconductor/minisite/exynos/technology/advanced-isp/

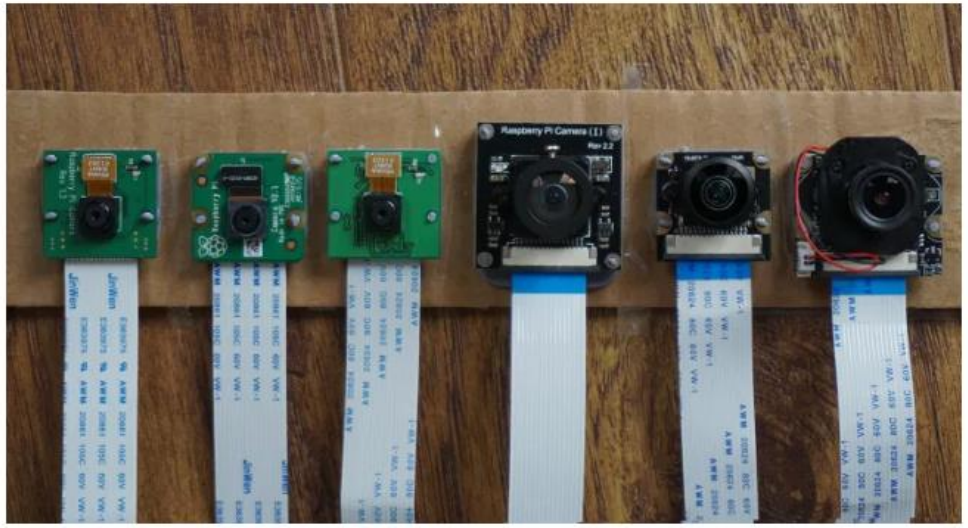
Image Signal processor (ISP)

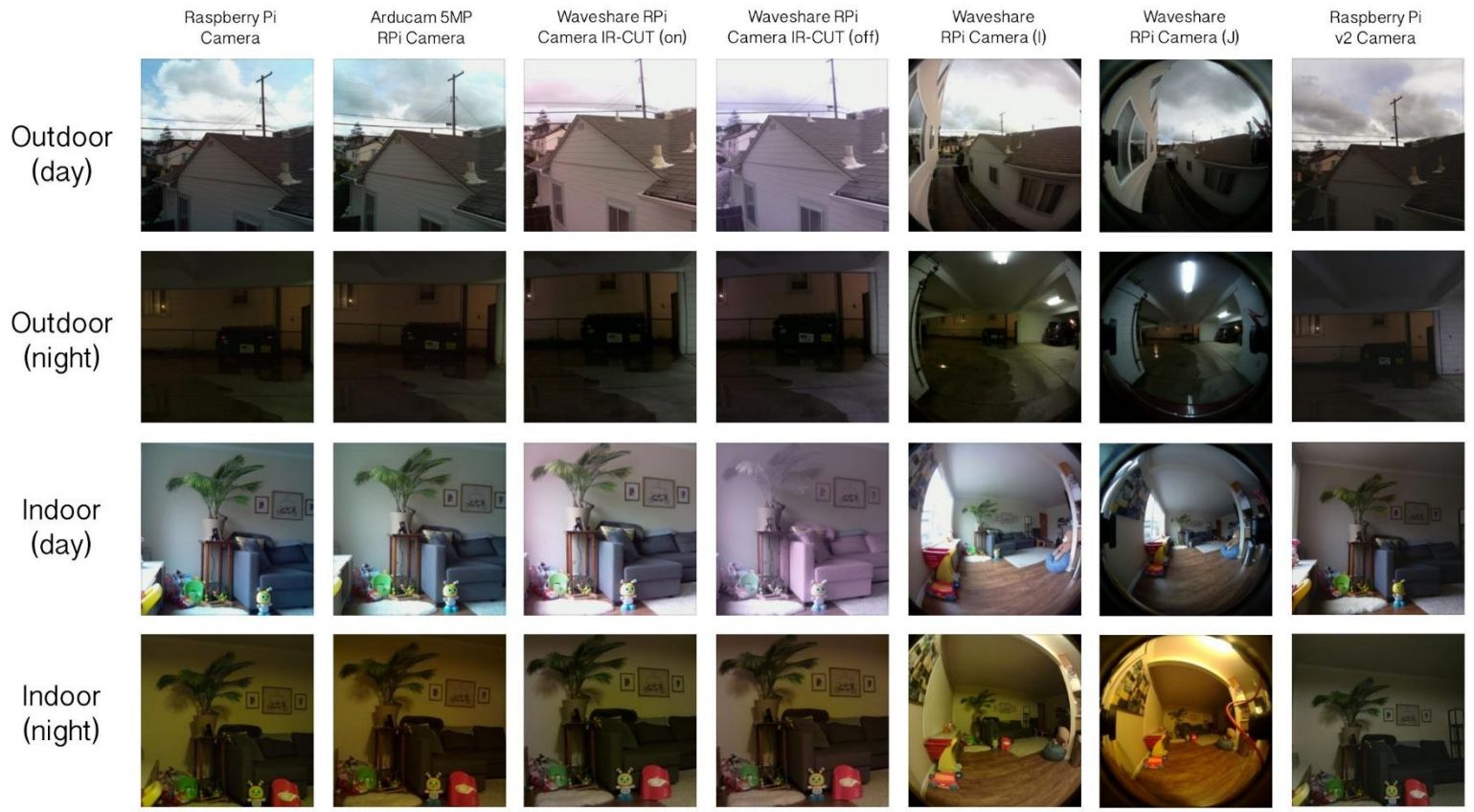
Noise Reduction	Digital images are prone to various types of noises during the image acquisition process that results in an abrupt change in pixel values that do not reflect the true intensities of the real scene. Denoising techniques are applied to image data that erase noise created depending on behavior & type of data and provides clear images
Auto white-balance & Color correction	Processing operations performed to ensure proper color fidelity in a captured digital camera image which applies color correction matrix (CCM) that transforms to adjust the colors to fit a particular output color space
Colour interpolation	Receiving Bayer inputs from the image sensor converts raw image, typically captured using a Bayer color filter array (CFA) into a color RGB image. This process is also known as demosaicing.
Lens shading correction	Is applied to improve brightness and color non-uniformity towards the image periphery
Defect pixel correction	Corrects defective pixels on the image sensor
Gamma correction	Compensates for the nonlinearity of relative intensity as the frame buffer value changes in output displays
Local tone mapping	Combines different exposures together in order to increase the local contrast within disparate regions of an HDR scene
Auto Exposure	Performs automatic fine tuning of the image brightness according to the amount of light that reaches the camera sensor
Auto Focus	Auto focus automatically adjusts the sharpness of the image, which improves the image definition. All types of actuator, lens position tuning, AF stats engine tuning etc

Ref: https://www.pathpartnertech.com/camera-tuning-understanding-the-image-signal-processor-and-isp-tuning/

Camera Models

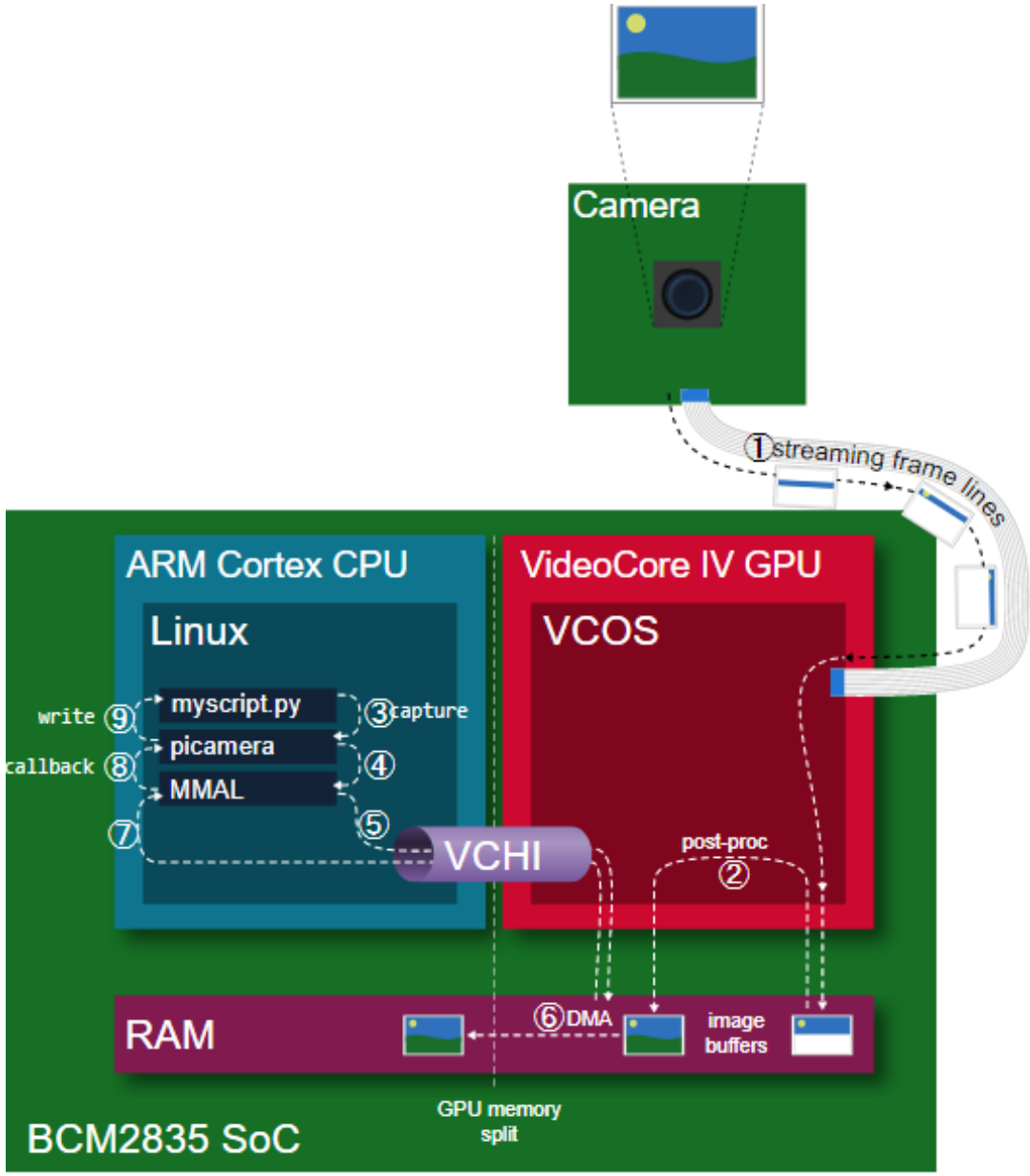
Model	Chipset	Megapixels	Advertised FOV
Raspberry Pi Camera	OV5647	5MP	54° (h) x 41° (v)
Raspberry Pi v2 Camera	IMX219	8MP	62.2° (h) x 48.8° (v)
Arducam 5MP RPi Camera	OV5647	5MP	54° (h) x 41° (v)
Waveshare RPi Camera (I)	OV5647	5MP	170°
Waveshare RPi Camera (J)	OV5647	5MP	222°
Waveshare RPi Camera IR-CUT	OV5647	5MP	75.7°



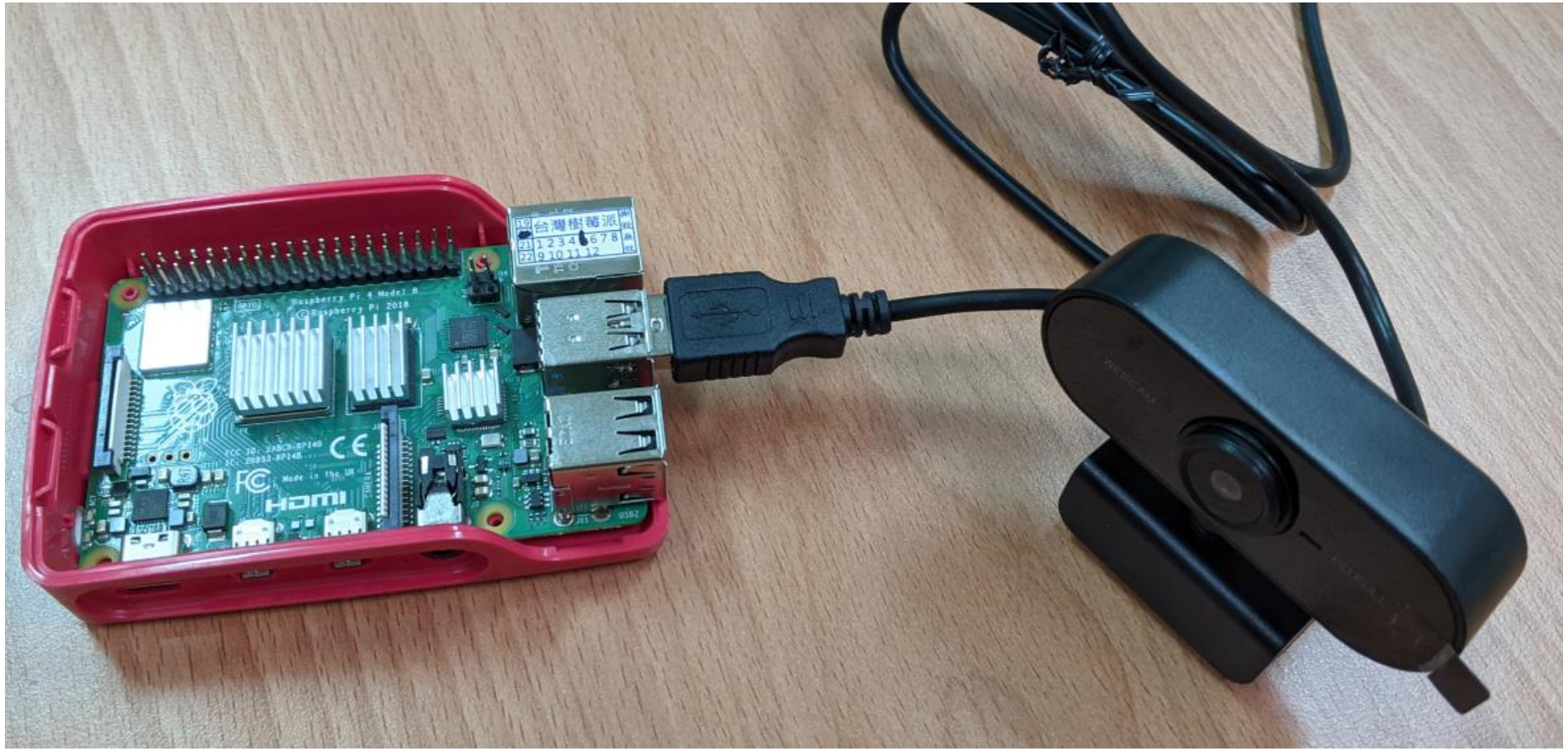


Pi Camera

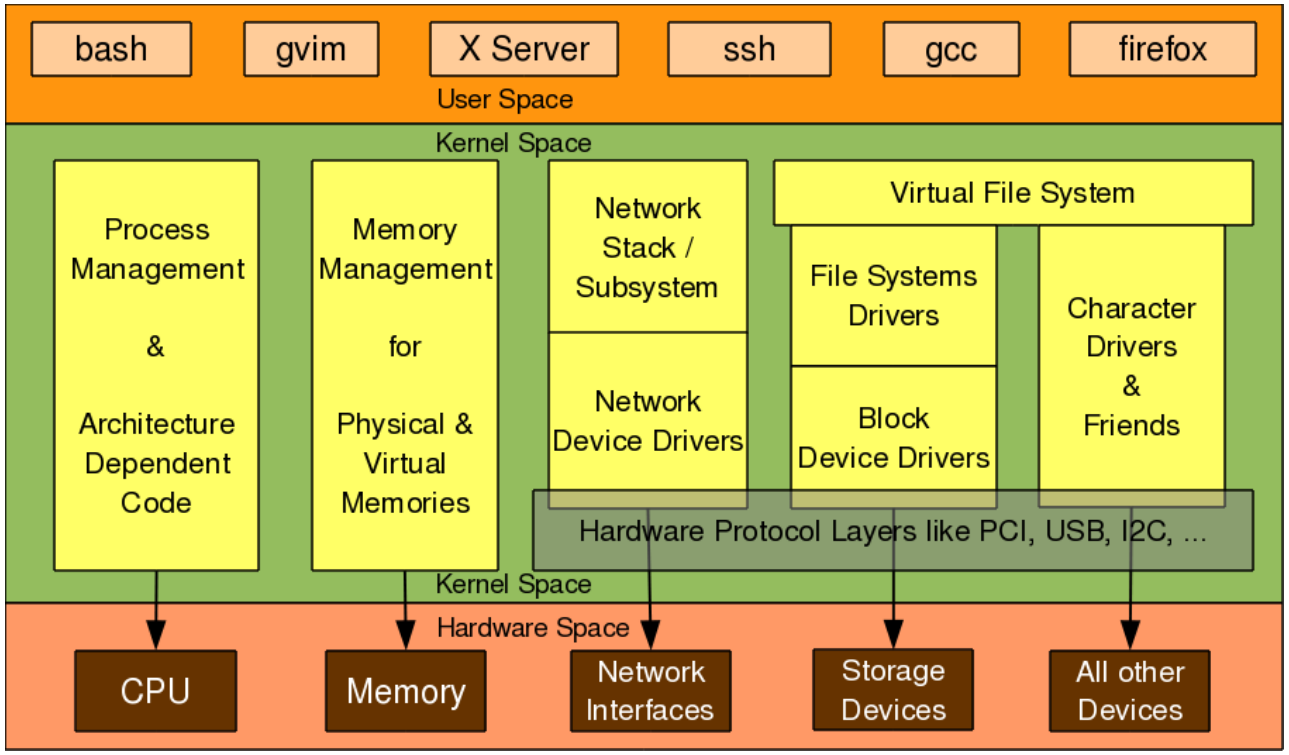
- The dataflow of capturing image when using Pi camera

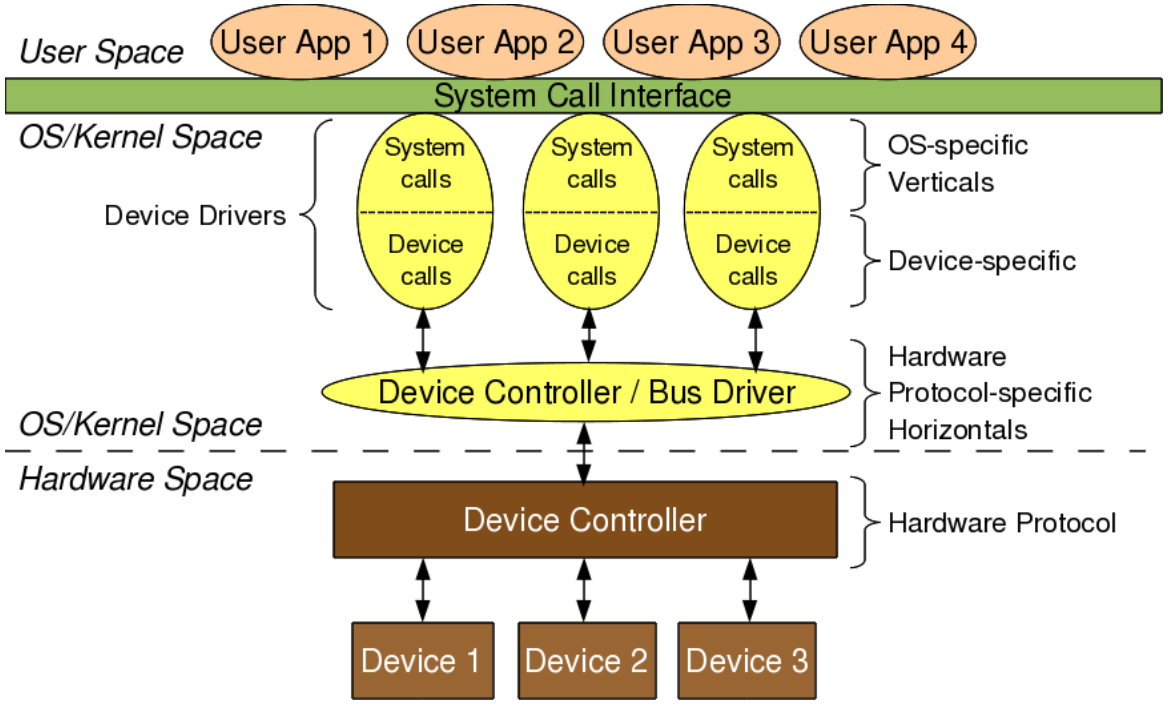


USB Camera



Linux Driver

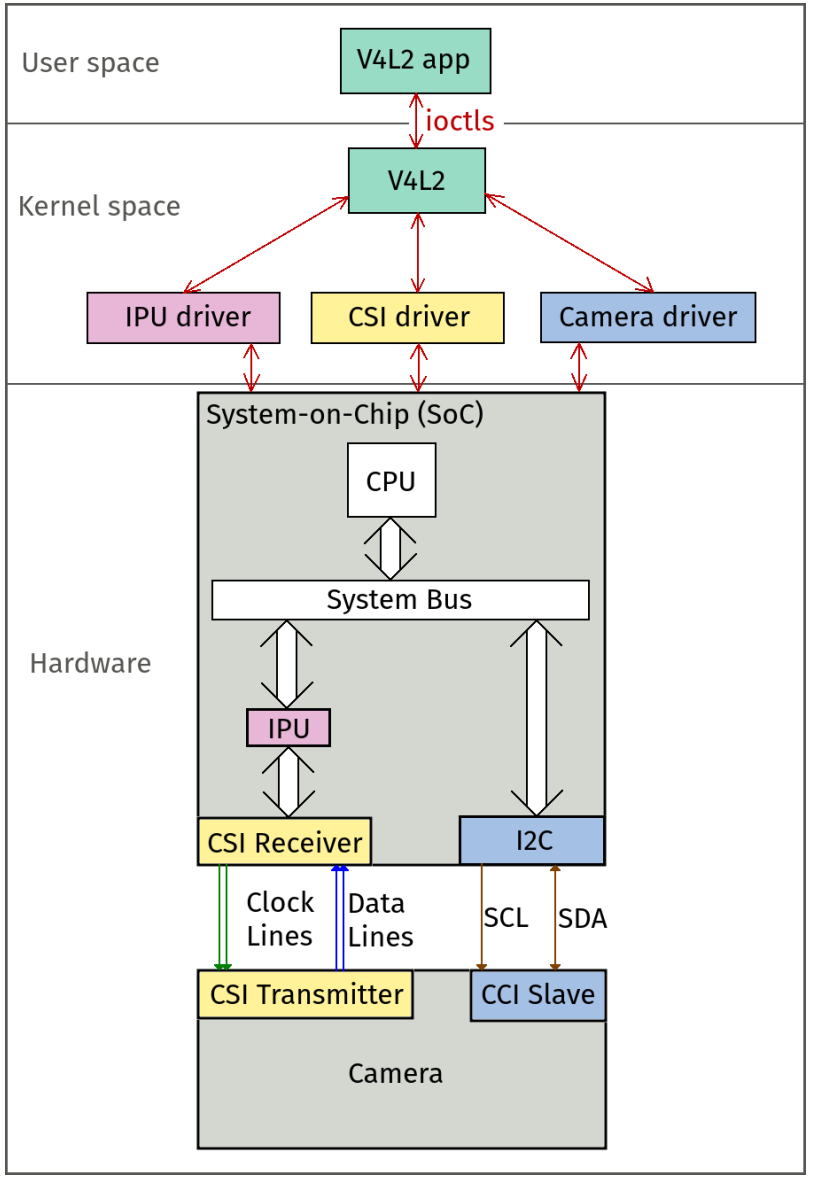




Ref : https://sysplay.github.io/books/LinuxDrivers/book/Content/Part01.html

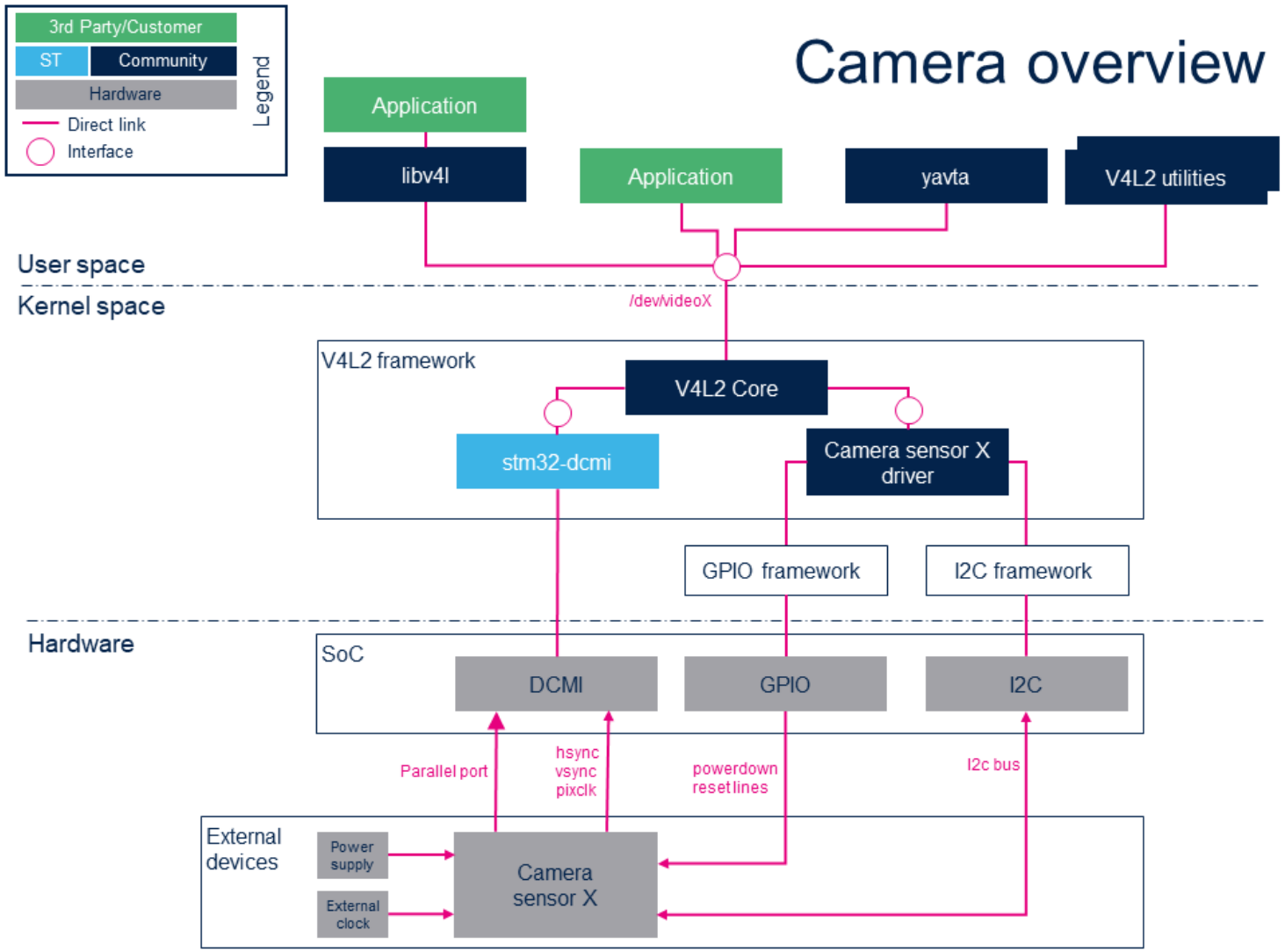
V4L2

- Video4Linux (V4L) is a collection of device drivers and an API for supporting real-time video capture on Linux systems.
- V4L2 is the second version of V4L.



Ref: https://stackoverflow.com/questions/56731704/where-does-linux-call-a-camera-driver-for-image-delivery

V4L2



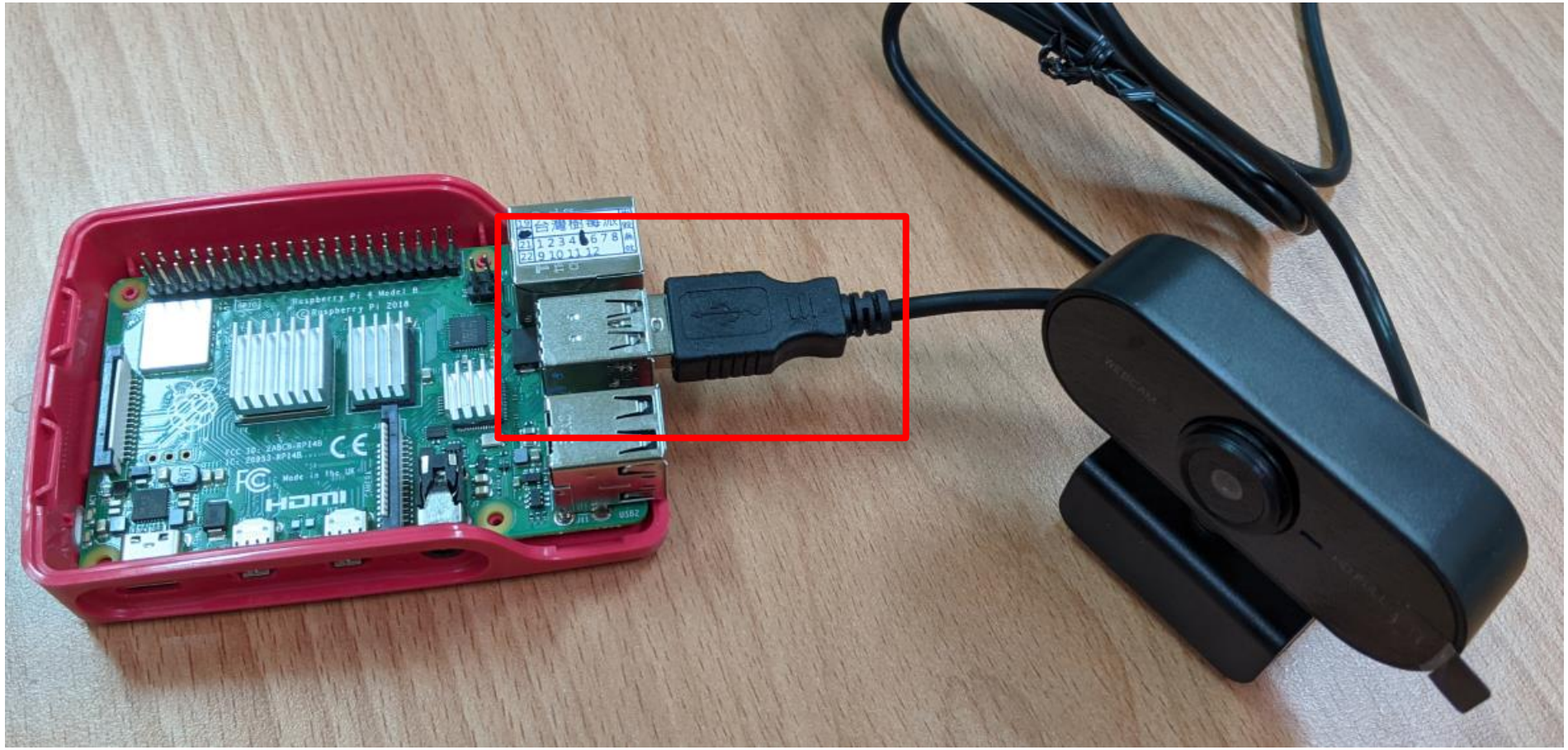
Ref: https://wiki.st.com/stm32mpu/wiki/V4L2_camera_overview

Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Camera Installation

- Connect the camera to a USB port.



USB Camera

$ lsusb

```
pi@rpi4-A00:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 006: ID 1b3f:2247 Generalplus Technology Inc.
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

$ ls /dev/video?

```
pi@rpi4-A00:~ $ ls /dev/video?
/dev/video0  /dev/video1
```

$ v4l2-ctl --list-devices

```
pi@rpi4-A00:~ $ v4l2-ctl --list-devices
bcm2835-codec-decode (platform:bcm2835-codec):
        /dev/video10
        /dev/video11
        /dev/video12
        /dev/video18

bcm2835-isp (platform:bcm2835-isp):
        /dev/video13
        /dev/video14
        /dev/video15
        /dev/video16

GENERAL WEBCAM: GENERAL WEBCAM (usb-0000:01:00.0-1.3):
        /dev/video0
        /dev/video1
```

USB Camera

$ v4l2-ctl --info --device /dev/video0

```
pi@rpi4-A00:~ $ v4l2-ctl --info --device /dev/video0
Driver Info:
        Driver name      : uvcvideo
        Card type        : GENERAL WEBCAM: GENERAL WEBCAM
        Bus info         : usb-0000:01:00.0-1.3
        Driver version   : 5.10.63
        Capabilities     : 0x84a00001
                Video Capture
                Metadata Capture
                Streaming
                Extended Pix Format
                Device Capabilities
        Device Caps      : 0x04200001
                Video Capture
                Streaming
                Extended Pix Format
```

$ v4l2-ctl --info --device /dev/video1

```
pi@rpi4-A00:~ $ v4l2-ctl --info --device /dev/video1
Driver Info:
        Driver name      : uvcvideo
        Card type        : GENERAL WEBCAM: GENERAL WEBCAM
        Bus info         : usb-0000:01:00.0-1.3
        Driver version   : 5.10.63
        Capabilities     : 0x84a00001
                Video Capture
                Metadata Capture
                Streaming
                Extended Pix Format
                Device Capabilities
        Device Caps      : 0x04a00000
                Metadata Capture
                Streaming
                Extended Pix Format
```

USB Camera

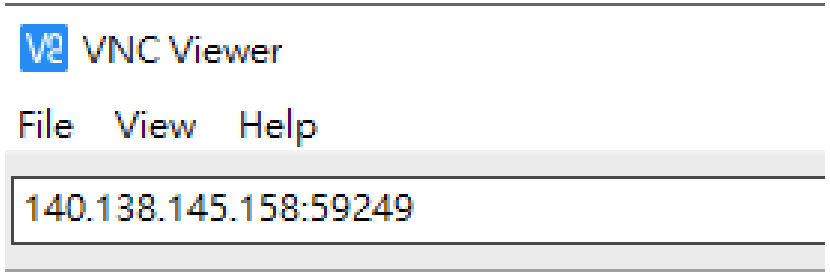
$ v4l2-ctl -d /dev/video0 --list-formats-ext

```
pi@rpi4-A00:~/iot $ v4l2-ctl -d /dev/video0 --list-formats-ext
ioctl: VIDIOC_ENUM_FMT
        Type: Video Capture

        [0]: 'MJPG' (Motion-JPEG, compressed)
                Size: Discrete 1920x1080
                        Interval: Discrete 0.033s (30.000 fps)
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1280x720
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 800x480
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 640x480
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 640x360
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 320x240
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 176x144
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 800x600
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1920x1080
                        Interval: Discrete 0.033s (30.000 fps)
                        Interval: Discrete 0.033s (30.000 fps)
        [1]: 'YUYV' (YUYV 4:2:2)
                Size: Discrete 640x480
                        Interval: Discrete 0.033s (30.000 fps)
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 640x360
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 320x240
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 176x144
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 640x480
                        Interval: Discrete 0.033s (30.000 fps)
                        Interval: Discrete 0.033s (30.000 fps)
```

VNC

- Launch “VNC Viewer”

- The default port of VNC is 5900.
- But, your RPi is under NAT of a router.
- Set your VNC port to a number 59xxx where xxx is the last byte of your IP.
- Ex: If your IP is 192.168.88.249, then your VNC port is 59249.



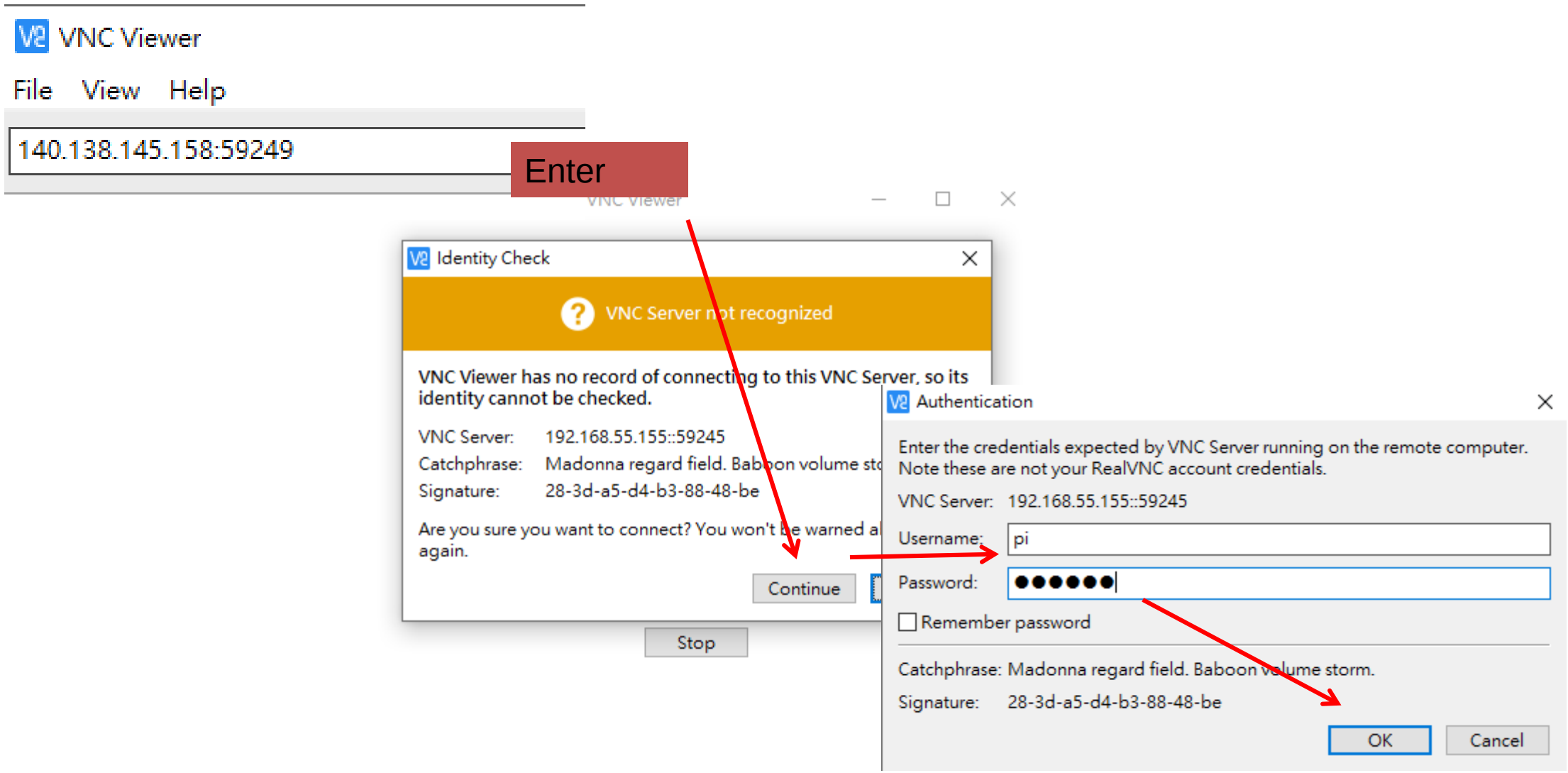

VNC Viewer
File View Help
140.138.145.158:59249
Enter
Identity Check
VNC Server not recognized
VNC Viewer has no record of connecting to this VNC Server, so its identity cannot be checked.
VNC Server: 192.168.55.155::59245
Catchphrase: Madonna regard field. Baboon volume sto
Signature: 28-3d-a5-d4-b3-88-48-be
Are you sure you want to connect? You won't be warned a
again.
Continue
Stop
Authentication
Enter the credentials expected by VNC Server running on the remote computer.
Note these are not your RealVNC account credentials.
VNC Server: 192.168.55.155::59245
Username: pi
Password:
Remember password
Catchphrase: Madonna regard field. Baboon volume storm.
Signature: 28-3d-a5-d4-b3-88-48-be
OK
Cancel

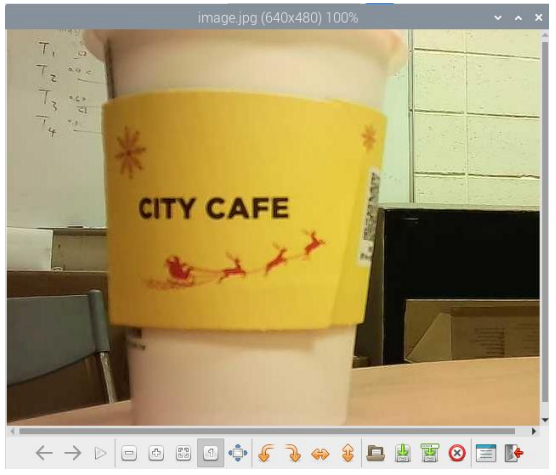
Image Capture

$ sudo apt-get install fswebcam

$ fswebcam -r 640x480 --no-banner image.jpg

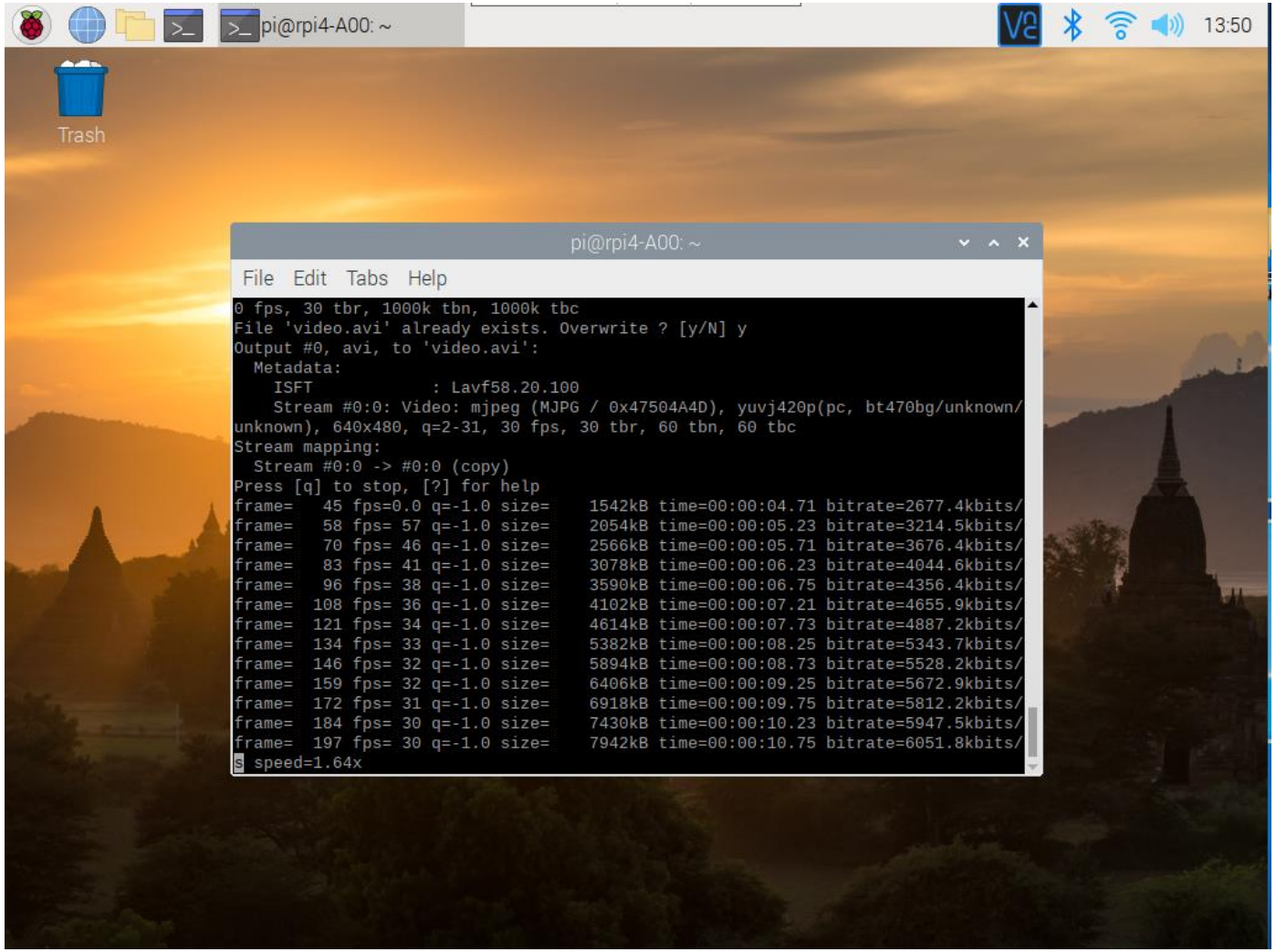
```
pi@rpi4-A00:~ $ fswebcam -r 640x480 --no-banner image.jpg
--- Opening /dev/video0...
Trying source module v4l2...
/dev/video0 opened.
No input was specified, using the first.
--- Capturing frame...
Captured frame in 0.00 seconds.
--- Processing captured image...
Disabling banner.
Writing JPEG image to 'image.jpg'.
```

- Browse the captured image in VNC.



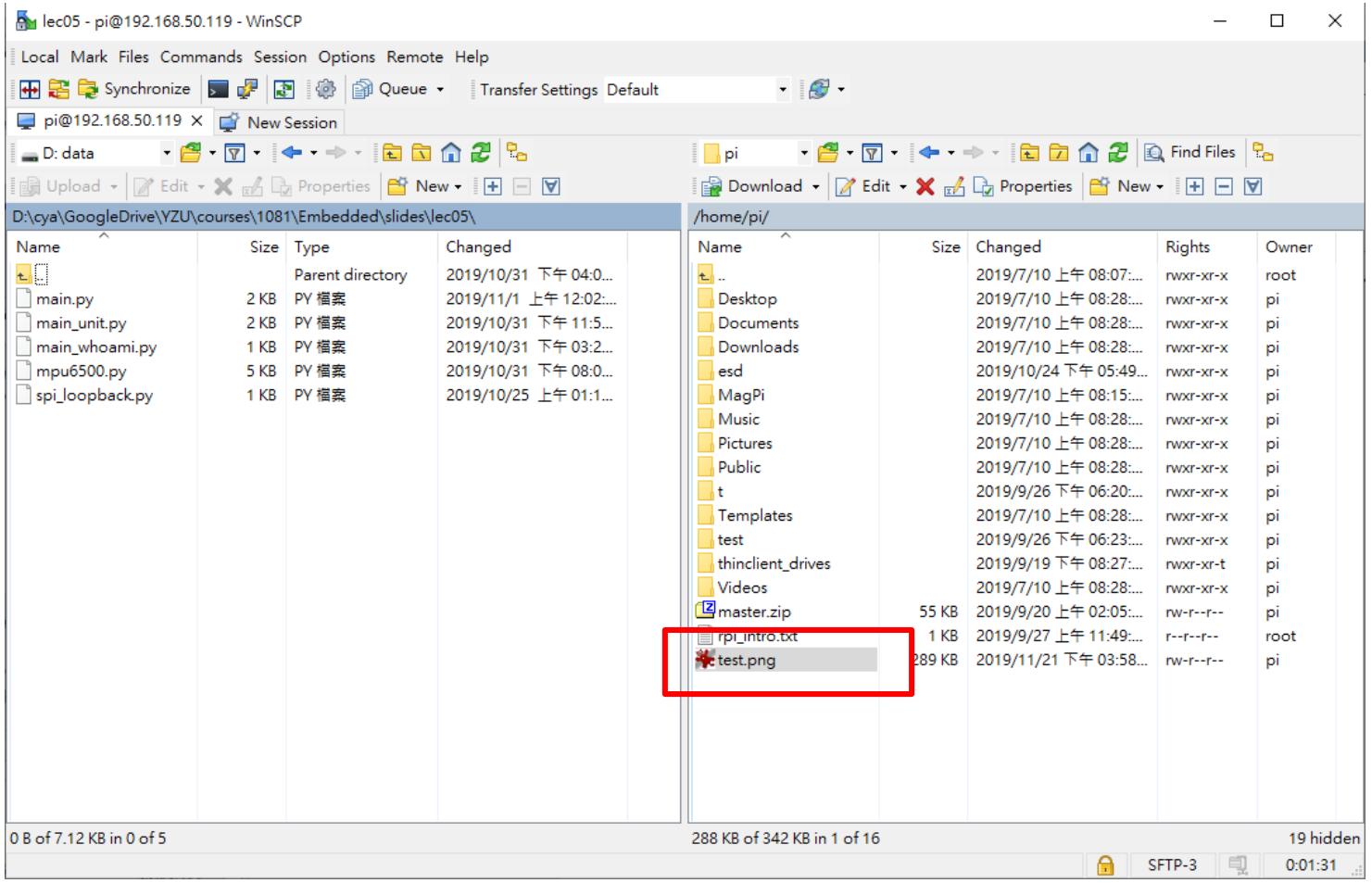
Video Capturing

$ ffmpeg -f video4linux2 -input_format mjpeg -video_size 640x480 -framerate 30 -i /dev/video0 -vcodec copy -an video.avi



File Transmission

- You can also use WinSCP to download the image from RPi.



Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Image Capture

- Miscellaneous Operating System Interfaces

```
$ python3
>>> import os
>>> os.system("fswebcam -r 640x480 --no-banner image.jpg")
>>> exit()
```

Schedule

$ pip3 install schedule

- Sample code

```
import schedule
import time

def job():
    print("I'm working...")

schedule.every(10).minutes.do(job)
schedule.every().hour.do(job)
schedule.every().day.at("10:30").do(job)
schedule.every().monday.do(job)
schedule.every().wednesday.at("13:15").do(job)
schedule.every().minute.at(":17").do(job)

while True:
    schedule.run_pending()
    time.sleep(1)
```

https://schedule.readthedocs.io/en/stable/api.html#schedule.Job.at

at(*time_str*) [source]

Specify a particular time that the job should be run at.

Parameters: **time_str** – A string in one of the following formats: *HH:MM:SS*, *HH:MM*,`:MM`, *:SS*. The format must make sense given how often the job is repeating; for example, a job that repeats every minute should not be given a string in the form *HH:MM:SS*. The difference between *:MM* and *:SS* is inferred from the selected time-unit (e.g. *every().hour.at(':30')* vs. *every().minute.at(':30')*).

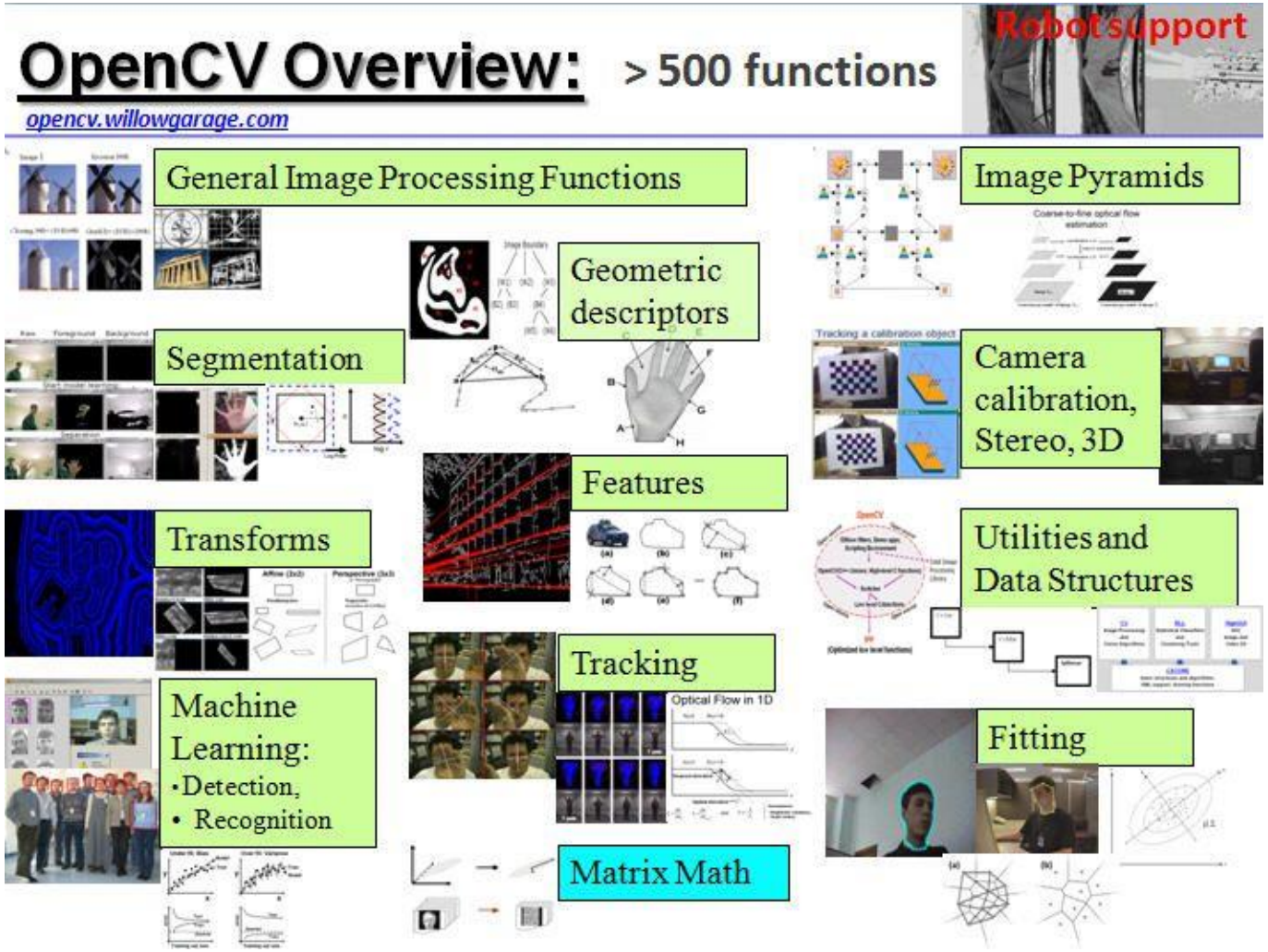
Returns: The invoked job instance

Outline

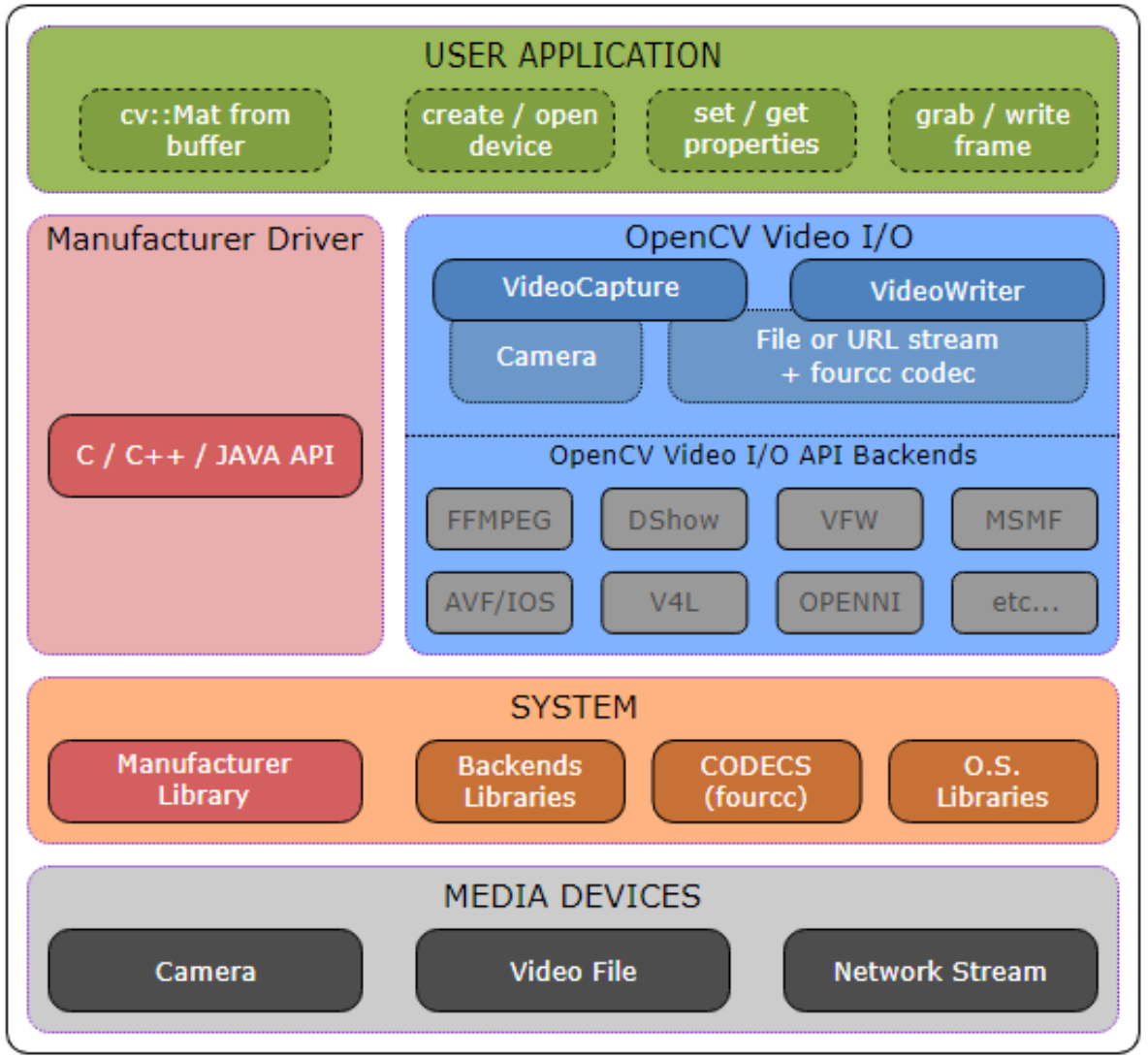
- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

OpenCV

- Open source Computer Vision library



OpenCV


USER APPLICATION
cv::Mat from buffer
create / open device
set / get properties
grab / write frame
Manufacturer Driver
C / C++ / JAVA API
OpenCV Video I/O
VideoCapture
VideoWriter
Camera
File or URL stream + fourcc codec
OpenCV Video I/O API Backends
FFMPEG
DShow
VFW
MSMF
AVF/IOS
V4L
OPENNI
etc...
SYSTEM
Manufacturer Library
Backends Libraries
CODECS (fourcc)
O.S. Libraries
MEDIA DEVICES
Camera
Video File
Network Stream

OpenCV Installation

- https://github.com/cyysky/OpenCV-Raspberry-Pi-4-Package-for-Python

$ wget https://github.com/cyysky/OpenCV-Raspberry-Pi-4-Package-for-Python/raw/master/opencv_4.5.0-1_armhf.deb

$ sudo dpkg -i opencv_4.5.0-1_armhf.deb

```
dpkg: error processing package opencv (--install):
 dependency problems - leaving unconfigured
Errors were encountered while processing:
 opencv
```

$ sudo apt-get -f install

```
0 upgraded, 482 newly installed, 0 to remove and 0 not upgraded.
1 not fully installed or removed.
Need to get 197 MB/232 MB of archives.
After this operation, 819 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

$ sudo dpkg -i opencv_4.5.0-1_armhf.deb

OpenCV

- After the installation, try to import OpenCV module.

$ python3

>>> import cv2

>>> cv2.__version__

>>> exit()

```
pi@rpi4-A00:~/iot/lec07 $ python3
Python 3.7.3 (default, Jul 25 2020, 13:03:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'4.5.0'
```

Image Capture

- cvimage.py

```
import cv2

cam = cv2.VideoCapture(0)

ret, image = cam.read()
cv2.imshow('preview',image)
cv2.waitKey(0)
cv2.imwrite('/home/pi/cvimage.jpg', image)
cam.release()
cv2.destroyAllWindows()
```

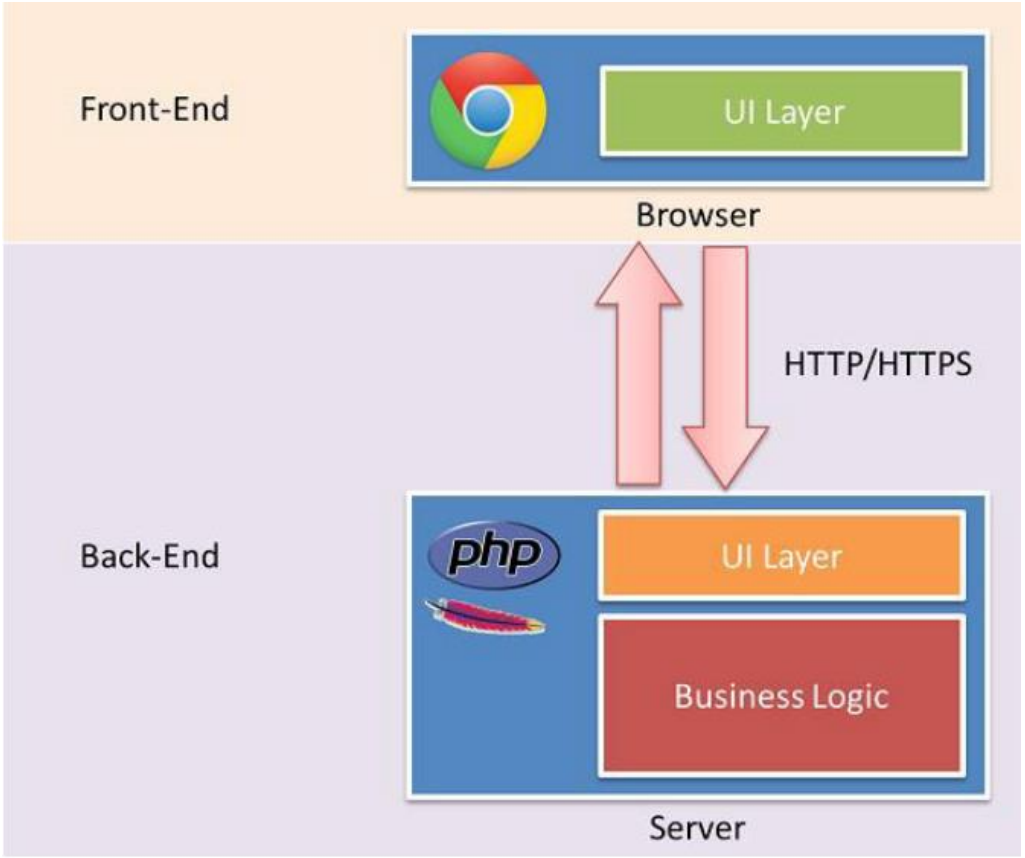
$ python3 cvimage.py

Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

HTTP + MJPG

- MJPEG = Motion JPEG
  - A video encoding
  - Every frame is encoded by JPEG.
  - Low computation and memory requirements
  - Browser supported
- Flask
  - A lightweight python web framework.



MJPG on RPi

- Load Linux V4L2 camera driver

  $ sudo modprobe bcm2835-v4l2

- Download sample code from

  $ wget https://github.com/yachentw/yzucseiot/raw/main/lec07/mjpg.tar.gz

- Unzip the downloaded file

  $ tar -zxvf mjpg.tar.gz

```
pi@rpi4-A00:~ $ tar -zxvf mjpg.tar.gz
mjpg/
mjpg/camera_pi.py
mjpg/app-camera.py
mjpg/templates/
mjpg/templates/stream.html
```

app-camera.py

```python
from flask import Flask, render_template, Response
from camera_pi import Camera

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('stream.html')

def gen(camera):
    while True:
        frame = camera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    return Response(gen(Camera()), mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=8000, debug=True)
```

stream.html

```html
<h1>Hello Stream</h1>
<img id="bg" src="{{ url_for('video_feed') }}">
```


Ref: https://flask.palletsprojects.com/en/1.1.x/patterns/streaming/

camera_pi.py

```python
import cv2

class Camera(object):
    def __init__(self):
        if cv2.__version__.startswith('2'):
            PROP_FRAME_WIDTH = cv2.cv.CV_CAP_PROP_FRAME_WIDTH
            PROP_FRAME_HEIGHT = cv2.cv.CV_CAP_PROP_FRAME_HEIGHT
        elif cv2.__version__.startswith('3') or cv2.__version__.startswith('4'):
            PROP_FRAME_WIDTH = cv2.CAP_PROP_FRAME_WIDTH
            PROP_FRAME_HEIGHT = cv2.CAP_PROP_FRAME_HEIGHT

        self.video = cv2.VideoCapture(0 , cv2.CAP_V4L)
        #self.video = cv2.VideoCapture(1)
        #self.video.set(PROP_FRAME_WIDTH, 640)
        #self.video.set(PROP_FRAME_HEIGHT, 480)
        self.video.set(PROP_FRAME_WIDTH, 320)
        self.video.set(PROP_FRAME_HEIGHT, 240)

    def __del__(self):
        self.video.release()

    def get_frame(self):
        success, image = self.video.read()
        ret, jpeg = cv2.imencode('.jpg', image)
        return jpeg.tostring()
```

MJPG on RPi

$ cd mjpg

$ python3 app-camera.py

```
pi@rpi4-A00:~/iot/lec07/mjpg $ python3 app-camera.py
 * Serving Flask app "app-camera" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 308-914-434
```

- http://140.138.145.158:8xxx
  - Replace “xxx” with your IP’s last 3 digits.

Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Post Image

- image_upload.py

```
import requests
import random
import time
import base64
from camera_pi import Camera

'''
global variables
'''

ENDPOINT = "things.ubidots.com"
DEVICE_LABEL = "camera_rpi"
VARIABLE_LABEL = "image"
TOKEN = "..."
DELAY = 15  # Delay in seconds
```

Replace with your token.

```
def post_var(payload, url=ENDPOINT, device=DEVICE_LABEL, token=TOKEN):
    try:
        url = "http://{}/api/v1.6/devices/{}".format(url, device)
        headers = {"X-Auth-Token": token, "Content-Type": "application/json"}

        attempts = 0
        status_code = 400

        while status_code >= 400 and attempts < 5:
            print("[INFO] Sending data, attempt number: {}".format(attempts))
            req = requests.post(url=url, headers=headers,
                                json=payload)
            status_code = req.status_code
            attempts += 1
            time.sleep(1)

        print("[INFO] Results:")
        print(req.text)
    except Exception as e:
        print("[ERROR] Error posting, details: {}".format(e))

def capture(camera):
    img = camera.get_frame_b64()
    payload = {VARIABLE_LABEL: {"value" : 1, "context" : {"img" : img}}}
    # print(payload)
    # Sends data
    post_var(payload)


if __name__ == "__main__":
    camera = Camera()
    while True:
        capture(camera)
        time.sleep(DELAY)
```

Post Image

- Download the code into “mjpg” folder

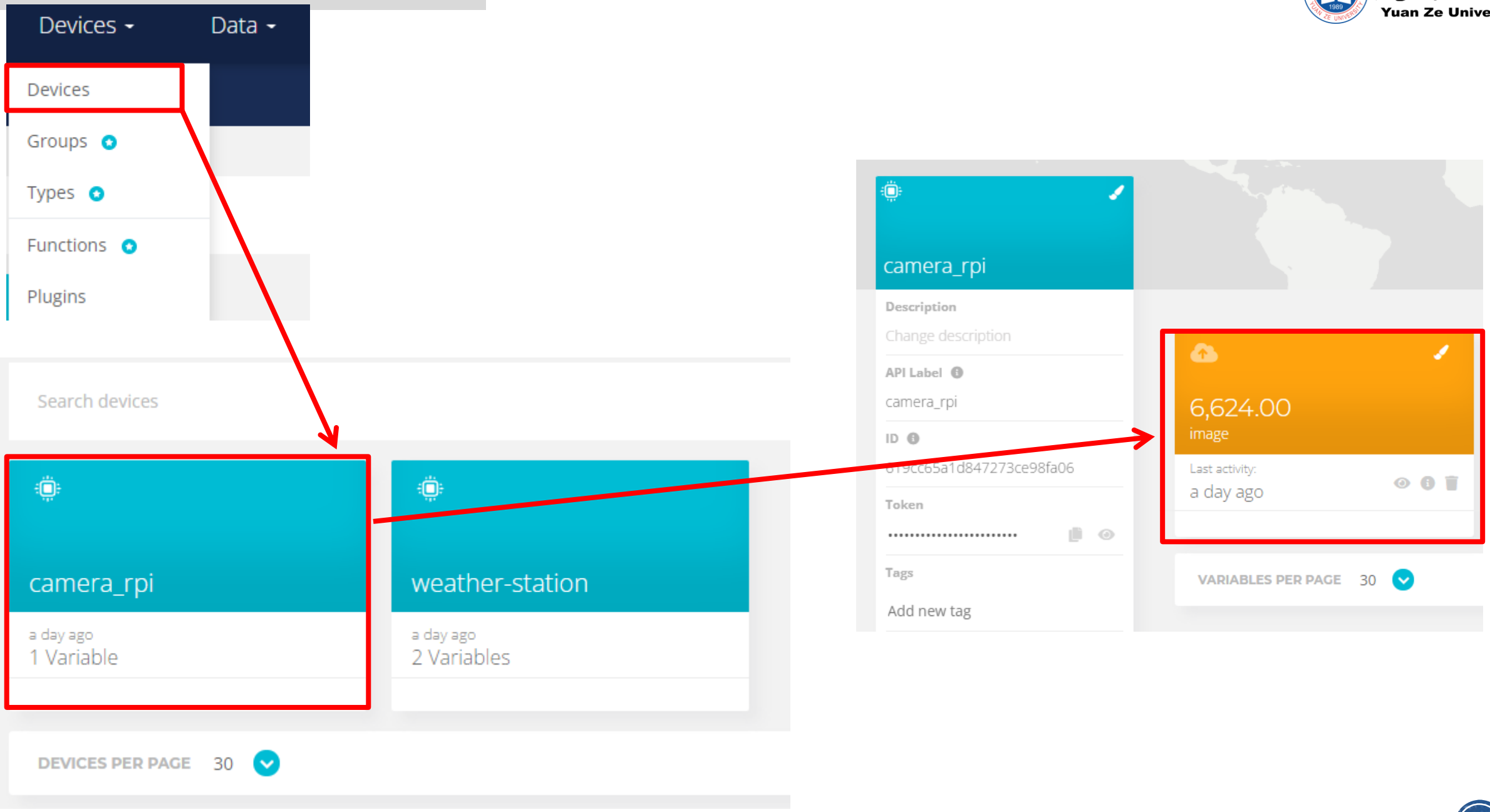
$ wget https://raw.githubusercontent.com/yachentw/yzucseiot/main/lec07/image_upload.py

```
pi@rpi4-A00:~/iot/lec07/mjpg $ wget https://raw.githubusercontent.com/yachentw/y
zucseiot/main/lec07/image_upload.py
--2021-11-25 15:11:43--  https://raw.githubusercontent.com/yachentw/yzucseiot/ma
in/lec07/image_upload.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.1
33, 185.199.111.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.
133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1306 (1.3K) [text/plain]
Saving to: 'image_upload.py'

image_upload.py     100%[===================>]   1.28K  --.-KB/s    in 0.01s

2021-11-25 15:11:44 (131 KB/s) - 'image_upload.py' saved [1306/1306]
```

- camera_pi.py

```python
import cv2
import base64

class Camera(object):
    def __init__(self):
        if cv2.__version__.startswith('2'):
            PROP_FRAME_WIDTH = cv2.cv.CV_CAP_PROP_FRAME_WIDTH
            PROP_FRAME_HEIGHT = cv2.cv.CV_CAP_PROP_FRAME_HEIGHT
        elif cv2.__version__.startswith('3') or cv2.__version__.startswith('4'):
            PROP_FRAME_WIDTH = cv2.CAP_PROP_FRAME_WIDTH
            PROP_FRAME_HEIGHT = cv2.CAP_PROP_FRAME_HEIGHT

        self.video = cv2.VideoCapture(0, cv2.CAP_V4L)
        #self.video.set(PROP_FRAME_WIDTH, 640)
        #self.video.set(PROP_FRAME_HEIGHT, 480)
        self.video.set(PROP_FRAME_WIDTH, 320)
        self.video.set(PROP_FRAME_HEIGHT, 240)

    def __del__(self):
        self.video.release()

    def get_frame(self):
        success, image = self.video.read()
        ret, jpeg = cv2.imencode('.jpg', image)
        return jpeg.tostring()

    def get_frame_b64(self):
        success, image = self.video.read()
        image = cv2.resize(image, (120, 90), interpolation=cv2.INTER_AREA)
        ret, jpeg = cv2.imencode('.jpg', image)
        return base64.b64encode(jpeg)
```

Add this line for resizing.


Devices
Data
Devices
Groups
Types
Functions
Plugins
Search devices
camera_rpi
a day ago
1 Variable
weather-station
a day ago
2 Variables
DEVICES PER PAGE
30
camera_rpi
Description
Change description
API Label
camera_rpi
ID
Token
Tags
Add new tag
6,624.00
image
Last activity:
a day ago
VARIABLES PER PAGE
30

6624.00
image

Description
Change description

API Label
image

ID
619cc8c71d8472771eaebc5d

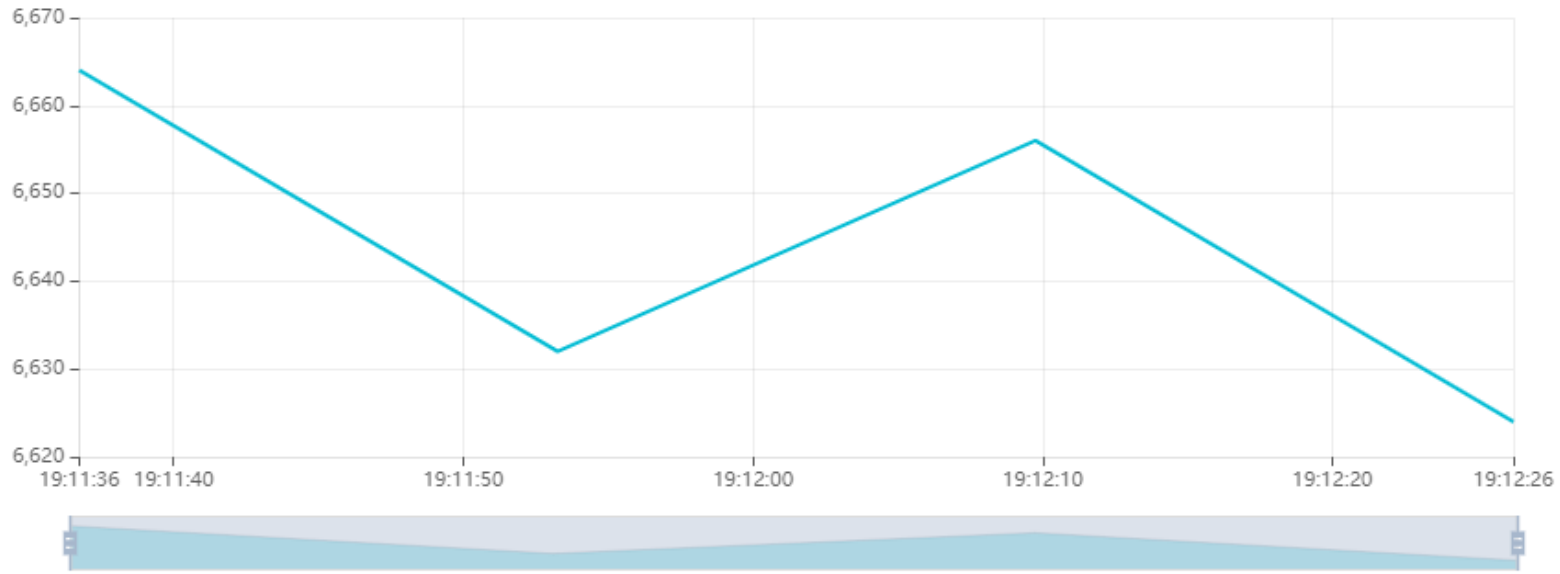
Allowed range
From: Min to: Max

Unit
Add unit

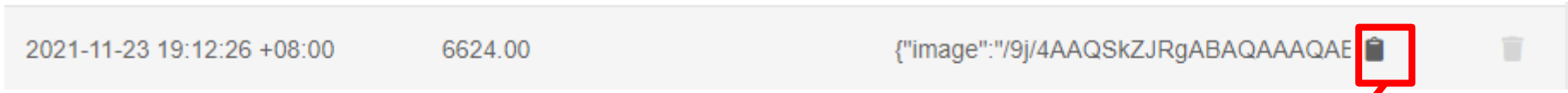
Tags
Add new tag

Last activity
a day ago

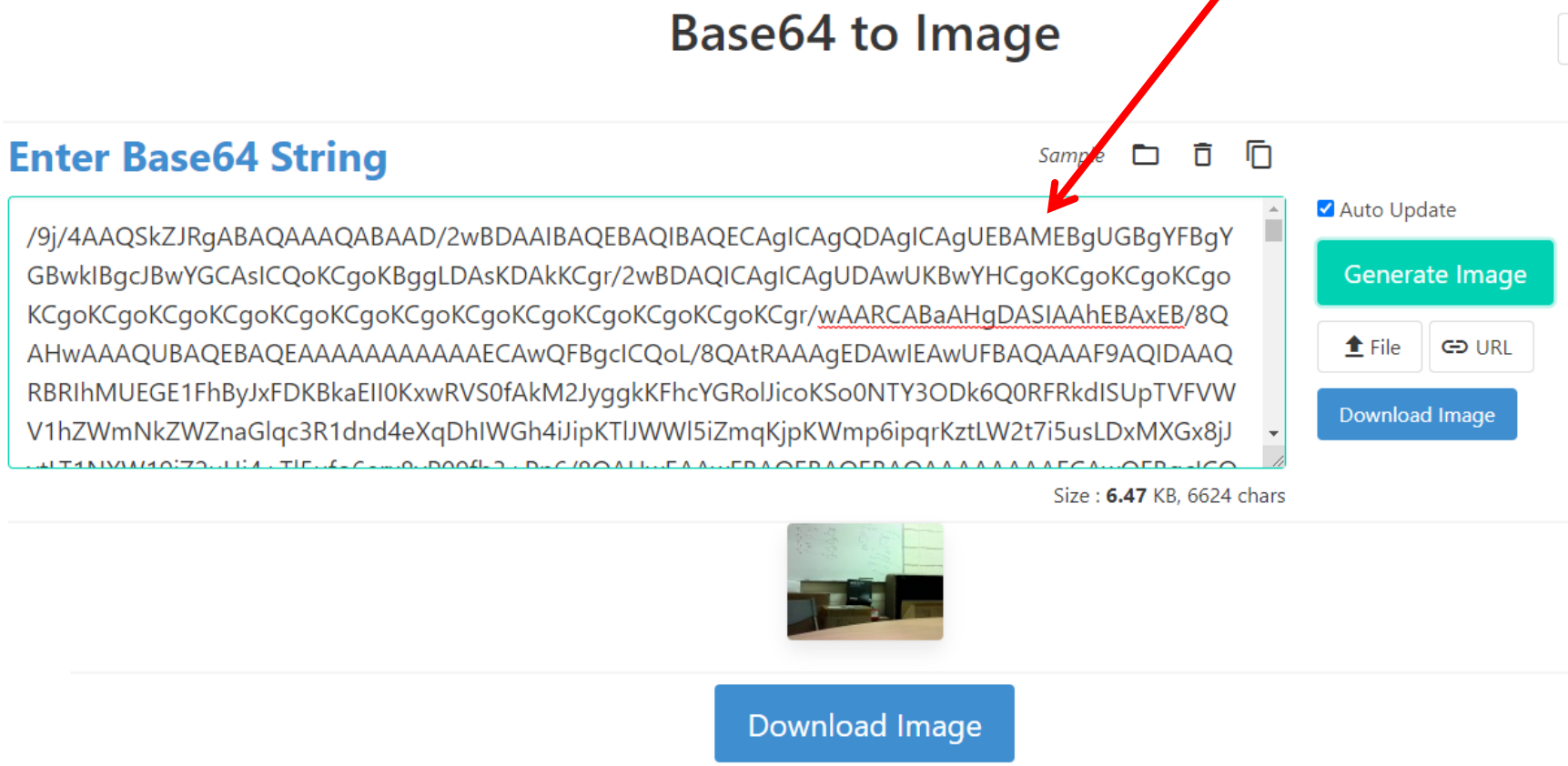
Nov 23 2021 19:11 - Now | Raw



DATE	VALUE	CONTEXT	ACTIONS
2021-11-23 19:12:26 +08:00	6624.00	{"image":"/9j/4AAQSkZJRgABAQAAAQAE	
2021-11-23 19:12:09 +08:00	6656.00	{"image":"/9j/4AAQSkZJRgABAQAAAQAE	



- https://codebeautify.org/base64-to-image-converter
  - Fill the value of key “img” and generate an image.



Get Image

- Restore the uploaded image.
- Modify “ubidots_http_get.py” to process the reply.

```
reply = json.loads(req.text)
b64img = base64.b64decode(reply['results'][0]['context']['image'])
nparr = np.frombuffer(b64img , np.uint8)
img = cv2.imdecode(nparr, cv2.IMREAD_COLOR)
```

Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Preview

- https://docs.opencv.org/2.4.13.7/doc/tutorials/introduction/display_image/display_image.html

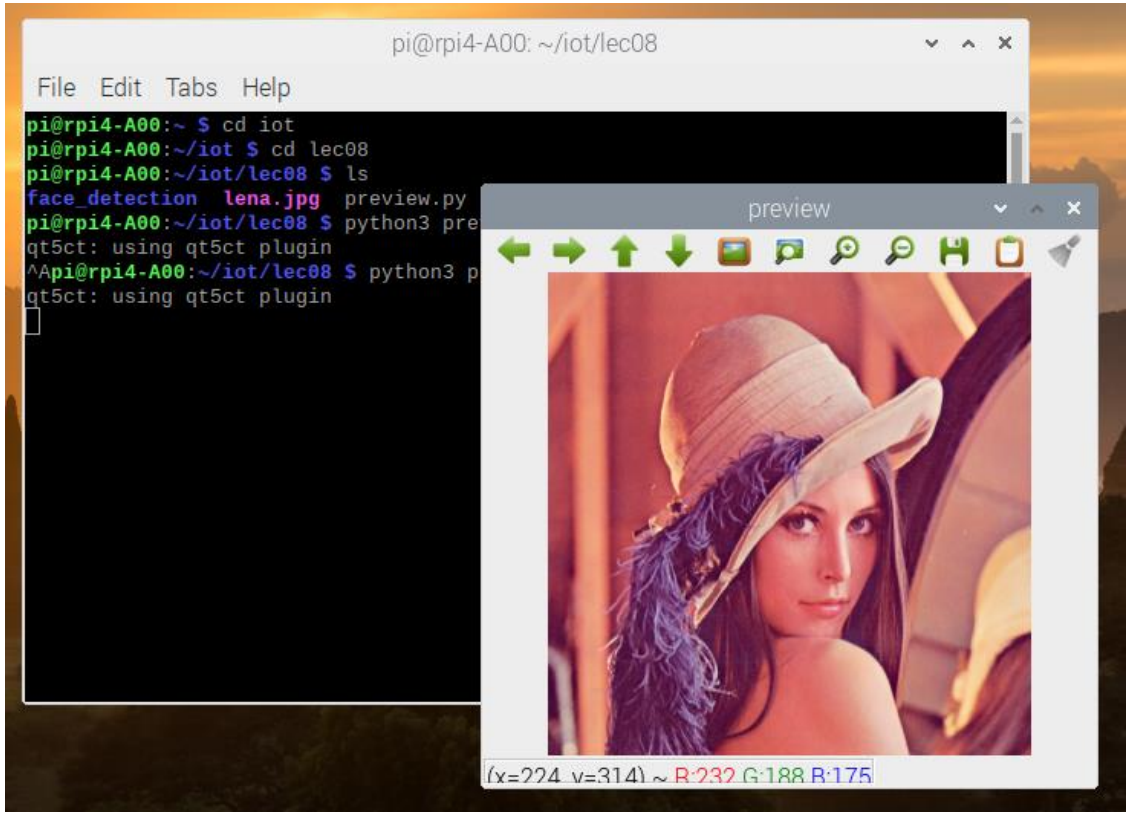
$ wget https://upload.wikimedia.org/wikipedia/zh/3/34/Lenna.jpg

$ mv Lenna.jpg lena.jpg

- preview.py

```
import cv2
import numpy as np

img = cv2.imread('lena.jpg')
cv2.imshow('preview', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

$ python3 preview.py



Affine Transformation

- https://docs.opencv.org/master/d4/d61/tutorial_warp_affine.html
- Affine Transformation
  - Translations (vector addition)
  - Rotations (linear transformation)
  - Scale operations (linear transformation)

Affine Transformation

The usual way to represent an Affine Transformation is by using a 2×3 matrix.

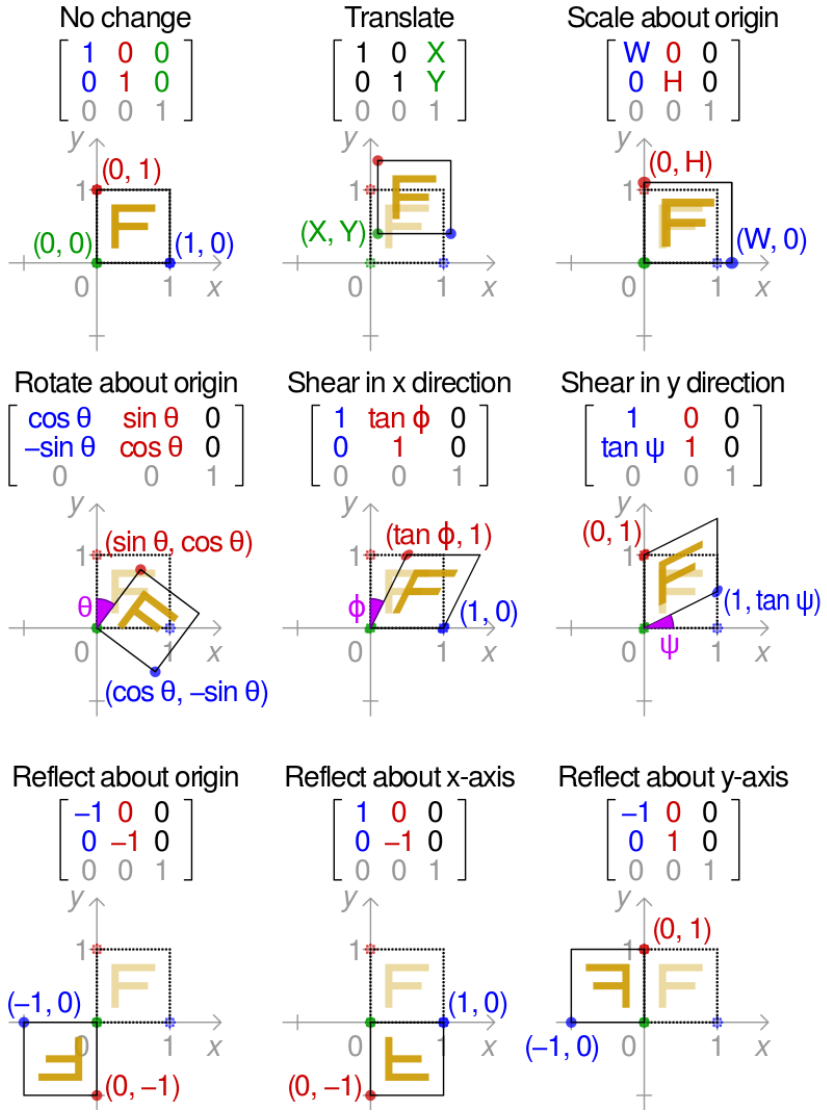
$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2\times 2} \quad B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2\times 1}$$

$$M = [\,A \quad B\,] = \begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}_{2\times 3}$$

Considering that we want to transform a 2D vector $X = \begin{bmatrix} x \\ y \end{bmatrix}$ by using A and B, we can do the same with:

$$T = A \cdot \begin{bmatrix} x \\ y \end{bmatrix} + B \text{ or } T = M \cdot [x, y, 1]^T$$

$$T = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix}$$



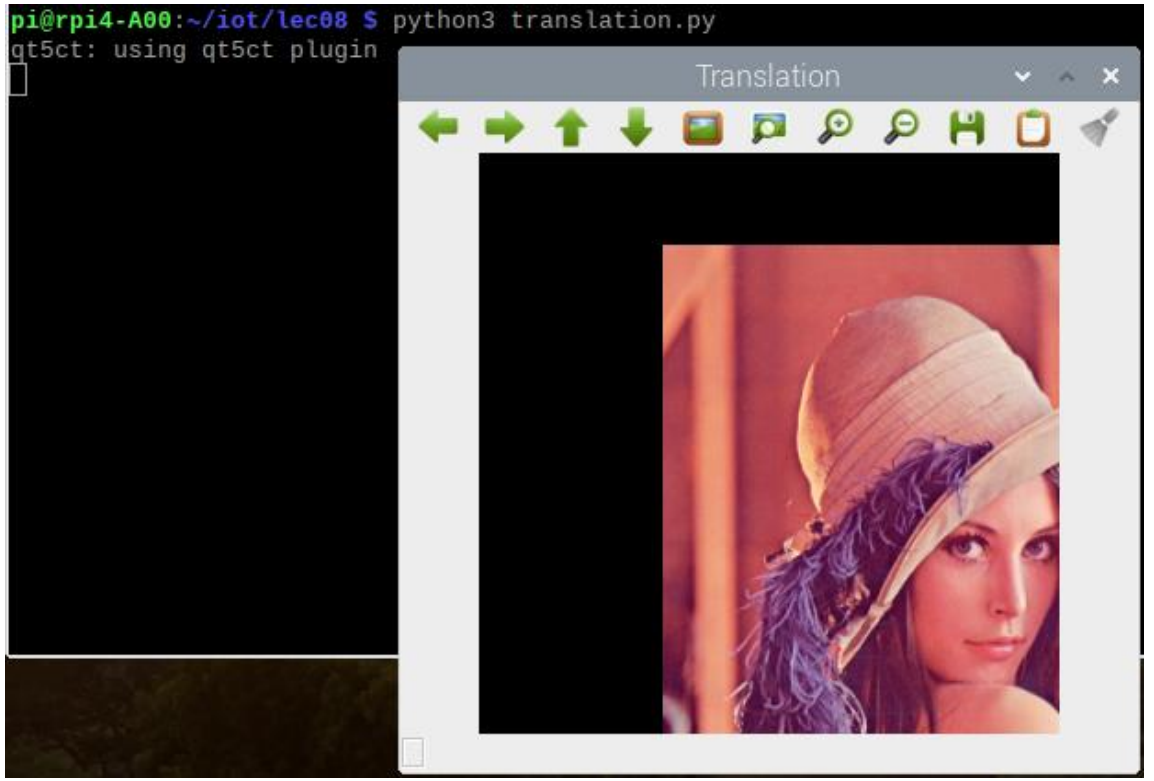
Ref: https://en.wikipedia.org/wiki/Transformation_matrix

Translation

- https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html
- translation.py

```
import cv2
import numpy as np
img = cv2.imread('lena.jpg')

rows, cols = img.shape[:2]
M = np.float32([ [1,0,100], [0,1,50] ])
translation = cv2.warpAffine(img, M, (cols, rows))
cv2.imshow('Translation', translation)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

$ python3 translation.py

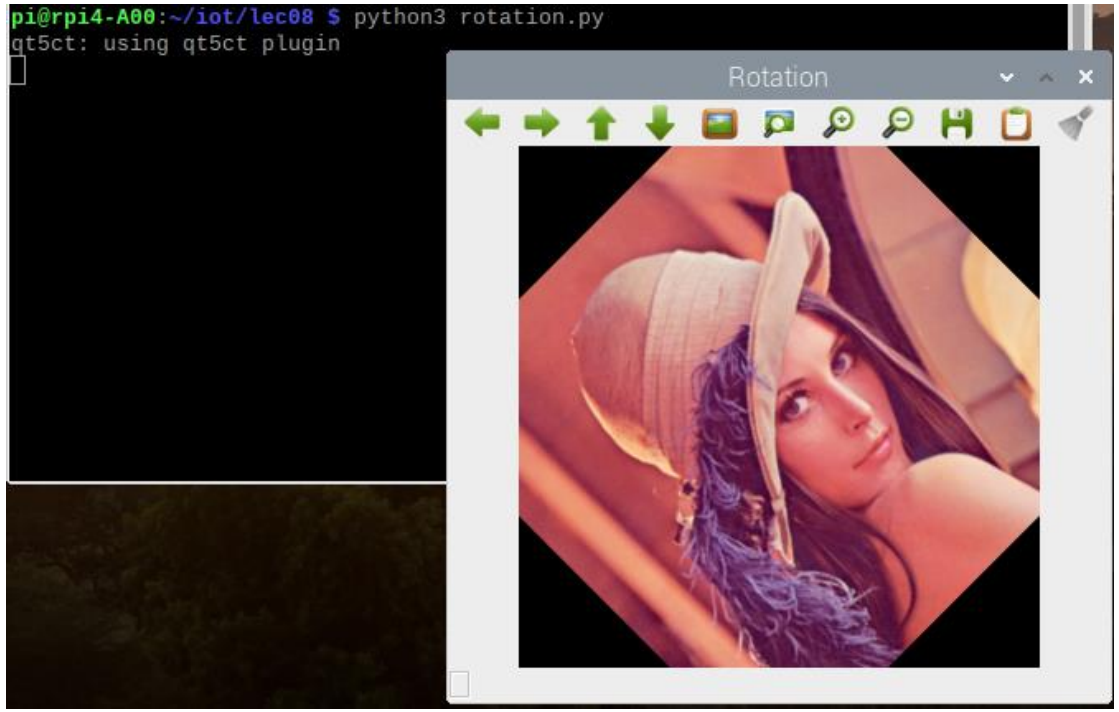


Rotation

- To rotate an image, we need to know two things:
  - The center with respect to which the image will rotate
  - The angle to be rotated. In OpenCV a positive angle is counter-clockwise.
  - Optional: A scale factor
- rotation.py

```
import cv2
import numpy as np

img = cv2.imread("lena.jpg")
rows, cols = img.shape[:2]

M = cv2.getRotationMatrix2D((cols/2, rows/2), 45, 1)
rotation = cv2.warpAffine(img, M, (cols, rows))

cv2.imshow('Rotation', rotation)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

$ python3 preview.py



Resize (1/2)

- https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html
- resize.py

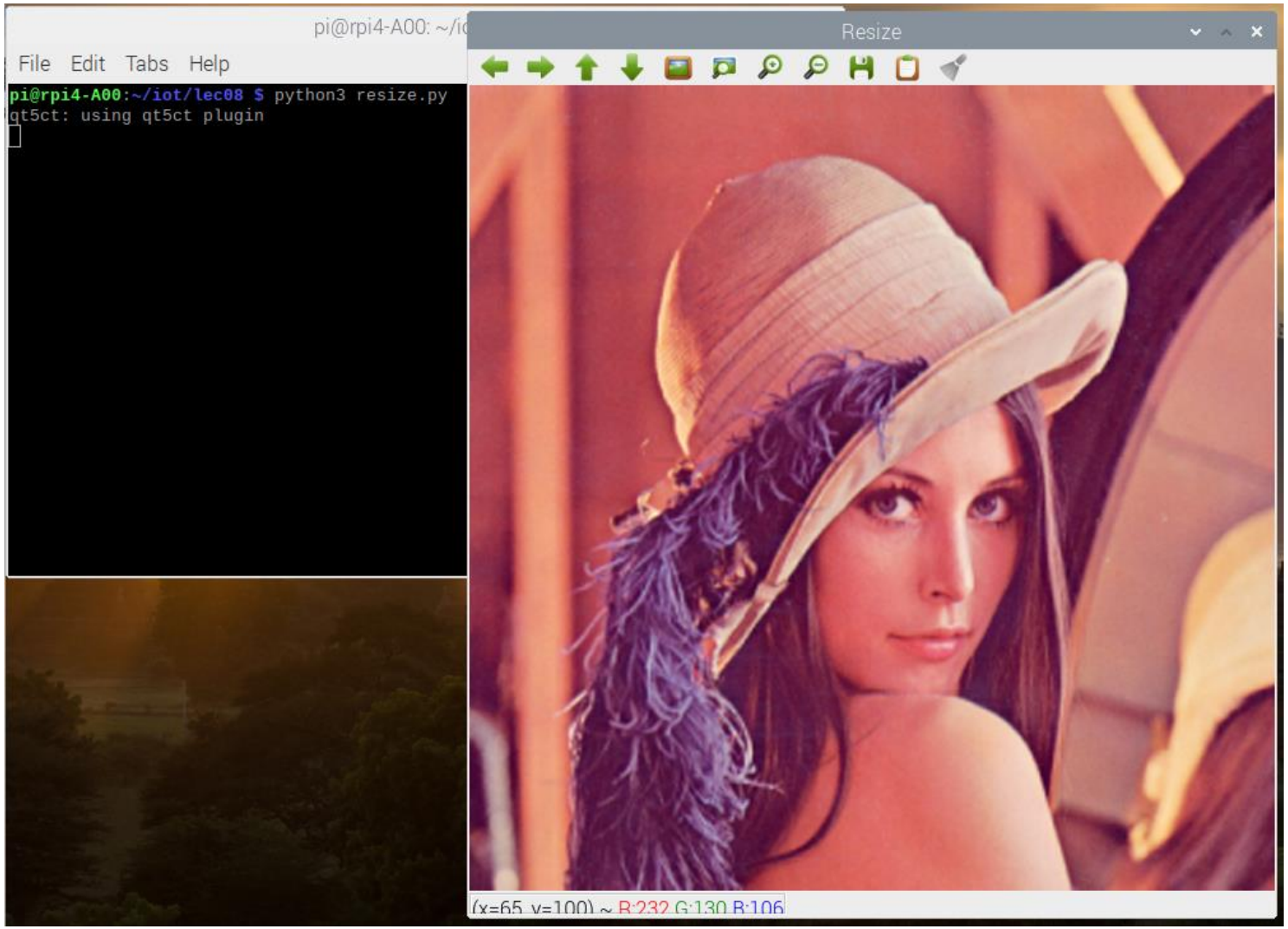
```
import cv2
import numpy as np

img = cv2.imread("lena.jpg")
rows, cols = img.shape[:2]
resize = cv2.resize(img, (2*rows, 2*cols), interpolation = cv2.INTER_CUBIC)

cv2.imshow('Resize', resize)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

interpolation method:

- **INTER_NEAREST** - a nearest-neighbor interpolation
- **INTER_LINEAR** - a bilinear interpolation (used by default)
- **INTER_AREA** - resampling using pixel area relation. It may be a preferred method for image decimation, as it gives moire'-free results. But when the image is zoomed, it is similar to the INTER_NEAREST method.
- **INTER_CUBIC** - a bicubic interpolation over 4x4 pixel neighborhood
- **INTER_LANCZOS4** - a Lanczos interpolation over 8x8 pixel neighborhood

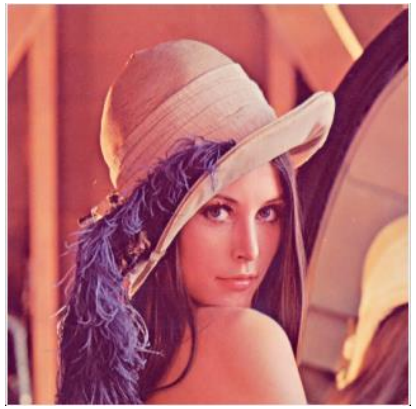
Resize (2/2)

$ python3 resize.py



Crop

```python
import cv2
import numpy as np

img = cv2.imread("lena.jpg")
cv2.imshow("Normal", img)
cv2.waitKey(0)

face = img[90:240, 125:225]
cv2.imshow("Face", face)
cv2.waitKey(0)

body = img[20:, 40:240]
cv2.imshow("Body", body)
cv2.waitKey(0)

cv2.destroyAllWindows()
```



Outline

- Introduction
- Image/video capture
  - Capture by tools
  - Python program via system interface
  - Python program via OpenCV
- Video streaming
  - Motion JPEG
- Uploading onto Ubidots
- OpenCV
- Lab

Lab

- Modify from MJPG lab.
- Push button one time to rotate the image 45 degrees counter clockwise.
- You may have to modify “app-camera.py” and “camera_pi.py”