Raspberry Pi Sound and Camera with Python

Rev. R610

이세우 (dltpdn@gmail.com)

세부목차

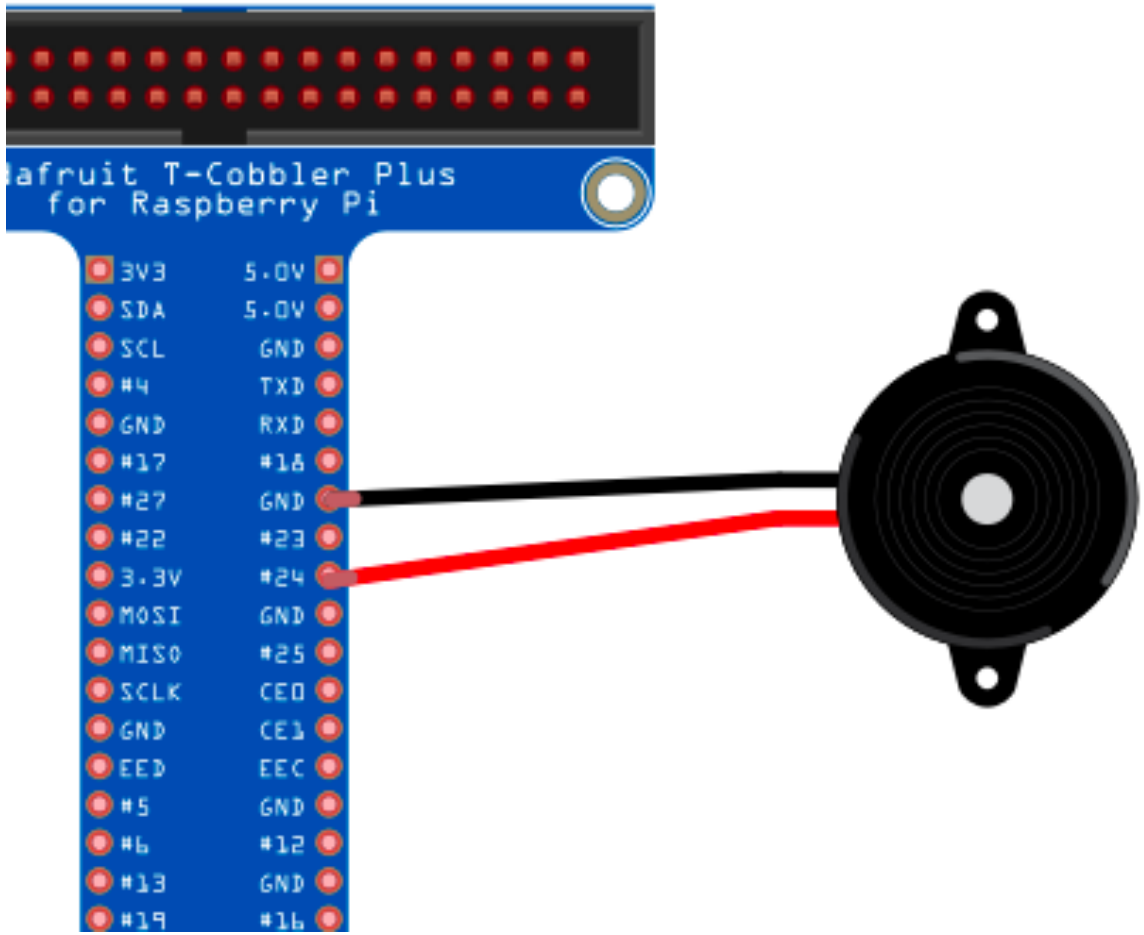
Buzzer

❖ Buzzer

- Digital output
- 필요 부품
  - 부저 스피커 또는 Piezo
  - 가변저항(볼륨조절)
- WiringPi-Python 모듈로만 가능
  - https://github.com/WiringPi/WiringPi-Python
  - 설치
    - git clone --recursive https://github.com/WiringPi/WiringPi-Python.git
    - sudo apt-get install python-dev python-setuptools swig
    - cd WiringPi-Python
    - ./build.sh
  - softToneCreate(PIN)
    - 출력 GPIO Pin 번호
  - softToneWrite(PIN, FREQUENCY)
    - 출력하려는 주파수 값 지정

Buzzer

❖ 신호음 재생 회로 구성



❖ 경보음 출력

```
import wiringpi
from time import sleep
pin = 24

wiringpi.wiringPiSetupGpio()

try:
    wiringpi.softToneCreate(pin)
    while True:
        wiringpi.softToneWrite(pin, 392)
        sleep(0.1)
        wiringpi.softToneWrite(pin, 523)
        sleep(0.1)
finally:
    wiringpi.pinMode(pin, 0)
```

Buzzer

❖ 반짝반짝 작은별 스케치

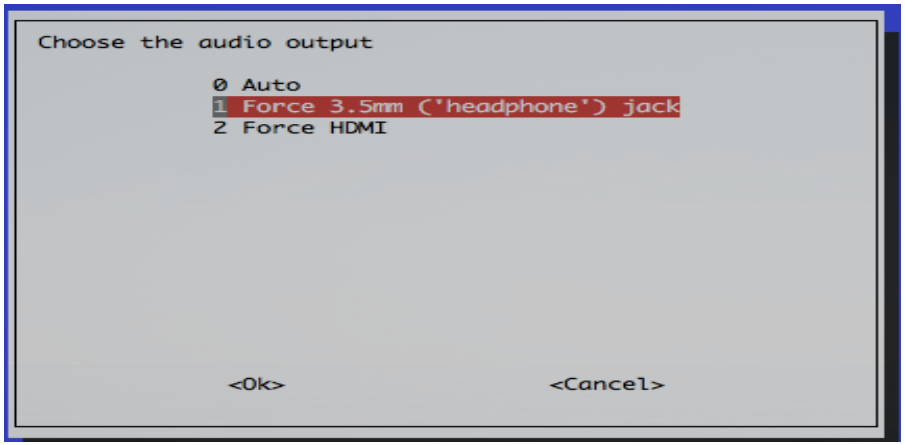
```
import wiringpi
from time import sleep

pin = 24
frequencies = {'c':262, 'd':294, 'e':330, 'f':349, 'g':392, 'a':440, 'b':494}
notes = 'ccggaag ffeeddc ggffeed ggffeed ccggaag ffeeddc'

wiringpi.wiringPiSetupGpio()

try:
    wiringpi.softToneCreate(pin)
    for i in notes:
        if i != ' ':
            wiringpi.softToneWrite(pin, frequencies[i])
        sleep(0.3)
finally:
    wiringpi.pinMode(pin, 0)
```

세부목차

Audio

❖ Audio 출력

- sudo raspi-config
  - Advanced Opiont --> Audio
- Audio Enable
  - /boot/config.txt
  - Dtparam=audio=on
- 다양한 Audio 출력 모듈듈
  - https://wiki.python.org/moin/Audio/
  - pyaudio
  - pygame



Audio

❖ ALSA

- Advanced Linux Sound Architecture
- Linux 커널 버전 2.6 기본 사운드 시스템
- 기존 OSS (Open Sound System) 대체
- 주요 utils
  - amixer
    - 사운드 시스템 설정 및 확인
  - alsamixer
    - 사운드 볼륨 조정
  - alsactl
    - 사운드 드라이버 고급 설정
  - aplay
    - Recorder and Player
    - aplay ~/sample.wav
  - speaker-test
    - 스피커에 화이트 노이즈 출력

Audio

❖ PyAudio

- https://people.csail.mit.edu/hubert/pyaudio/
- 설치
  - sudo apt-get install python-pyaudio
- 기능
  - Recoding(녹음)
  - Wav 출력, mp3 미 지원
- 주요 함수
  - p = pyaudio.PyAudio()
  - stream = p.open()
  - stream.read(chunk)
  - stream.stop_stream()
  - stream.close()
  - p.terminate()

Audio

- **PyAudio**
  - Wave File Play Block

```
import pyaudio, wave

chunk = 1200
wf = wave.open('/home/pi/sample.wav', 'rb')
p = pyaudio.PyAudio()
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
                channels=wf.getnchannels(),
                rate=wf.getframerate(),
                output=True)
data = wf.readframes(chunk)
while len(data) > 0:
   stream.write(data)
   data = wf.readframes(chunk)
stream.stop_stream()
stream.close()
p.terminate()
```

Audio

PyAudio

- Wave File Play Callback

```
import pyaudio, wave, time

def callback(in_data, frame_count, time_info, status):
   data = wf.readframes(frame_count)
   return (data, pyaudio.paContinue)

wf = wave.open('/home/pi/sample.wav', 'rb')
p = pyaudio.PyAudio()
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
            channels=wf.getnchannels(),
            rate=wf.getframerate(),
            output=True,
            stream_callback = callback)
stream.start_stream()
while stream.is_active():
   time.sleep(0.5)
stream.stop_stream()
stream.close()
wf.close()
p.terminate()
```

Audio

❖ Pygame mixer 모듈

- http://www.pygame.org/docs/ref/music.html
- import pygame
- pygame.init()
- pygame.mixer.music.load('sample.mp3')
- pygame.mixer.music.play()
- pygame.mixer.music.pause()
- pygame.mixer.music.unpause()
- pygame.mixer.music.stop()

Audio

Pygame mixer code

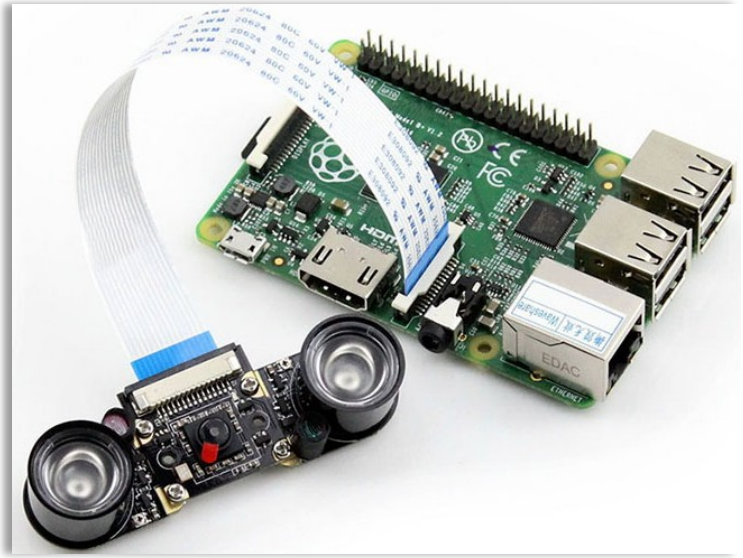
```
import pygame
import time

pygame.init()

#pygame.mixer.music.load("sample.wav")
pygame.mixer.music.load("sample.mp3")
while True:
   cmd = raw_input("cmd{play:p, pause:pp, unpause:up, stop:s}:")
   if cmd == "p":
      pygame.mixer.music.play()
   elif cmd == "pp":
      pygame.mixer.music.pause()
   elif cmd == "up":
      pygame.mixer.music.unpause()
   elif cmd == "s":
      pygame.mixer.music.stop()
    #  exit(0)
   else:
      print "incorrect cmd. try again."
```

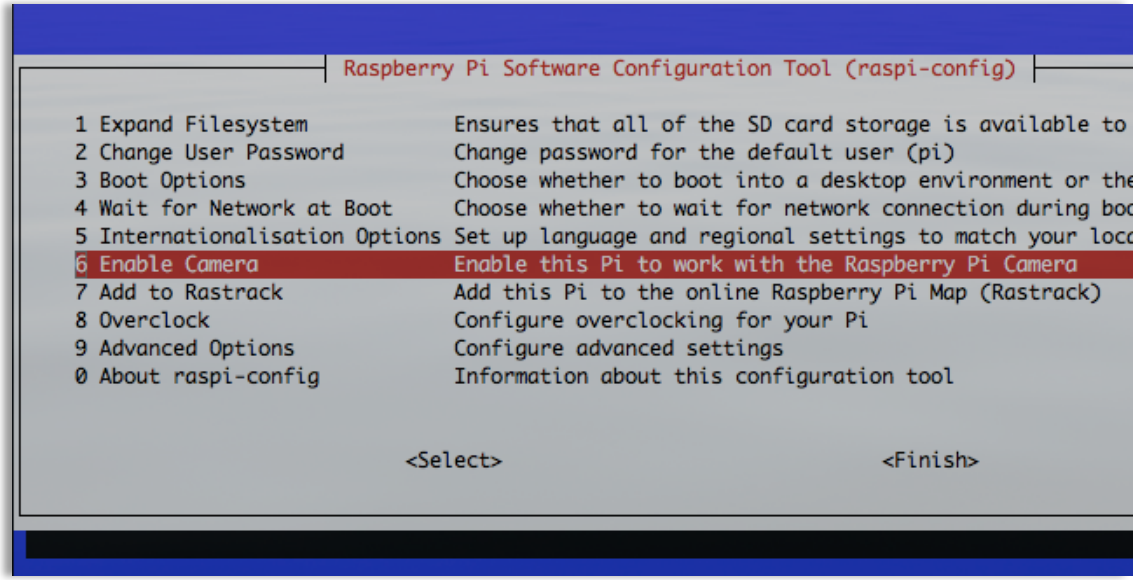
세부목차

Camera

❖ Raspi-Camera

- https://www.raspberrypi.org/documentation/usage/camera/python/README.md
- Camera Interface에 연결
- 라즈베리 카메라 활성화
  - sudo raspi-config
    - 6. Enable Camera





Camera

❖ Raspi-Camera

- 사진 촬영
  - raspistill -o caputre1.jpg
    - 5초후 정지영상 촬영
  - raspistill -0 capture2.jpg -w 1280 -h 720
    - 해상도 1280 x 720
- 동영상 촬영
  - raspivid -o video.h264 -t 5000
    - 5초 동안 동영상 촬영,
    - 영상 확인
      - omxplayer path/video.h264
  - raspivid -o video.mpeg
    - 영상 확인
      - 라즈베리파이의 브라우져에서 확인 가능

❖ Pycamera Module

- Pycamera 모듈 설치
  - https://github.com/waveform80/picamera
  - sudo apt-get install python-pycamera
- 주요 기능
  - camera = picamera.PiCamera()
  - camera.rotation = 90
  - camera.resolution = (1280, 720)
  - camera.start_preview()
  - camera.capture('cam1.jpg')
  - camera.start_recording('video.h264')
  - camera.stop_recoding()
  - camera.stop_preview()
  - camera.close()

Camera

❖ Pycamera Module

```
import time
import picamera

with picamera.PiCamera() as camera:
    try:
        camera.start_preview()
        while True:
            shutter = input('insert key when you are ready to take photo. [photo:1, video:2] ')
            now_str = time.strftime("%Y%m%d-%H%M%S")
            if shutter == 1:
                camera.capture('/home/pi/demo/camera/photo%s.gif'%now_str)
            elif shutter == 2:
                camera.start_recording('/home/pi/demo/camera/video%s.h264'%now_str)
                raw_input('insert key when you want to stop recoding.')
                camera.stop_recording()
    finally:
        camera.stop_preview()
        camera.close()
```

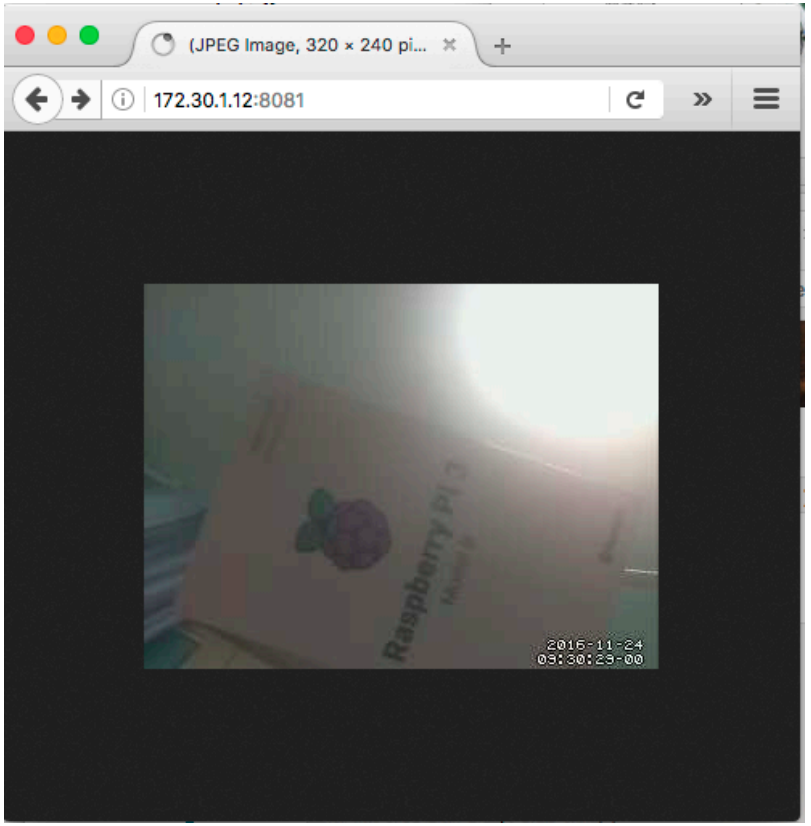
세부목차

❖ Camera Streaming

- 카메라 영상을 원격으로 스트리밍
- 주요 서비스
  - Motion
    - http://www.lavrsen.dk/foswiki/bin/view/Motion
  - MJPG-Streamer
    - https://sourceforge.net/projects/mjpg-streamer/
  - VLC
    - http://www.videolan.org/vlc/

❖ Motion

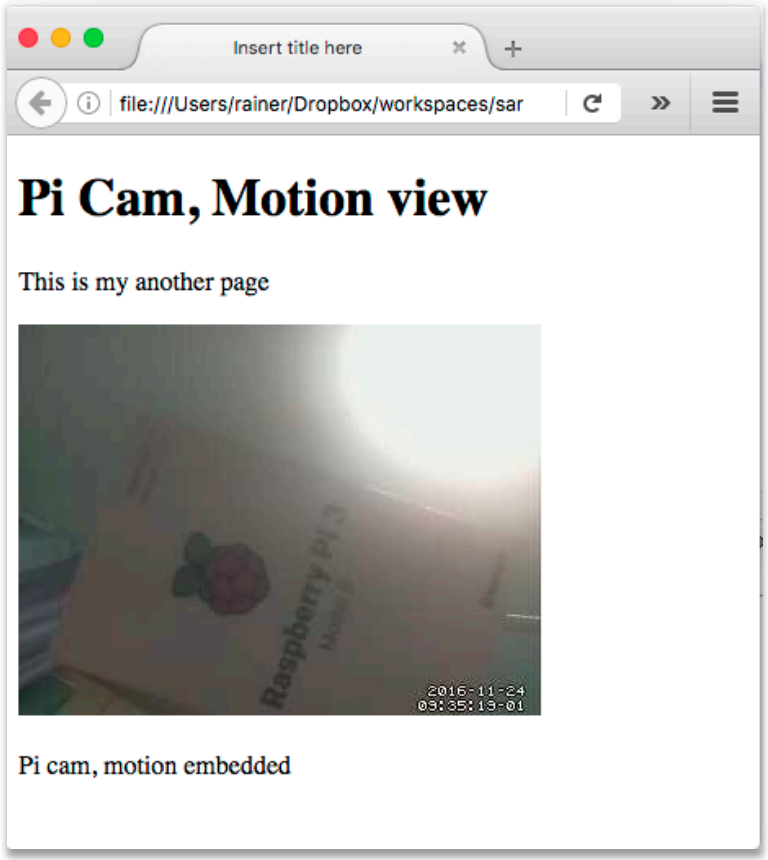
- 동작을 감지해서 촬영, 원격 스트리밍
- 설치
  - sudo apt-get install motion
- 설정
  - /etc/motion/motion.conf
    - stream_localhost on → off
- 실행 및 종료
  - sudo motion
  - sudo service motion stop
- 동작 확인
  - 웹 브라우저 접속
  - http://ip_address:8081



❖ Camera Streaming

- motion view를 웹페이지에 삽입

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>Pi Cam, Motion view</h1>
<p>This is my another page</p>

<img src="http://172.30.1.12:8081"/>

<p>Pi cam, motion embedded</p>
</body>
</html>
```

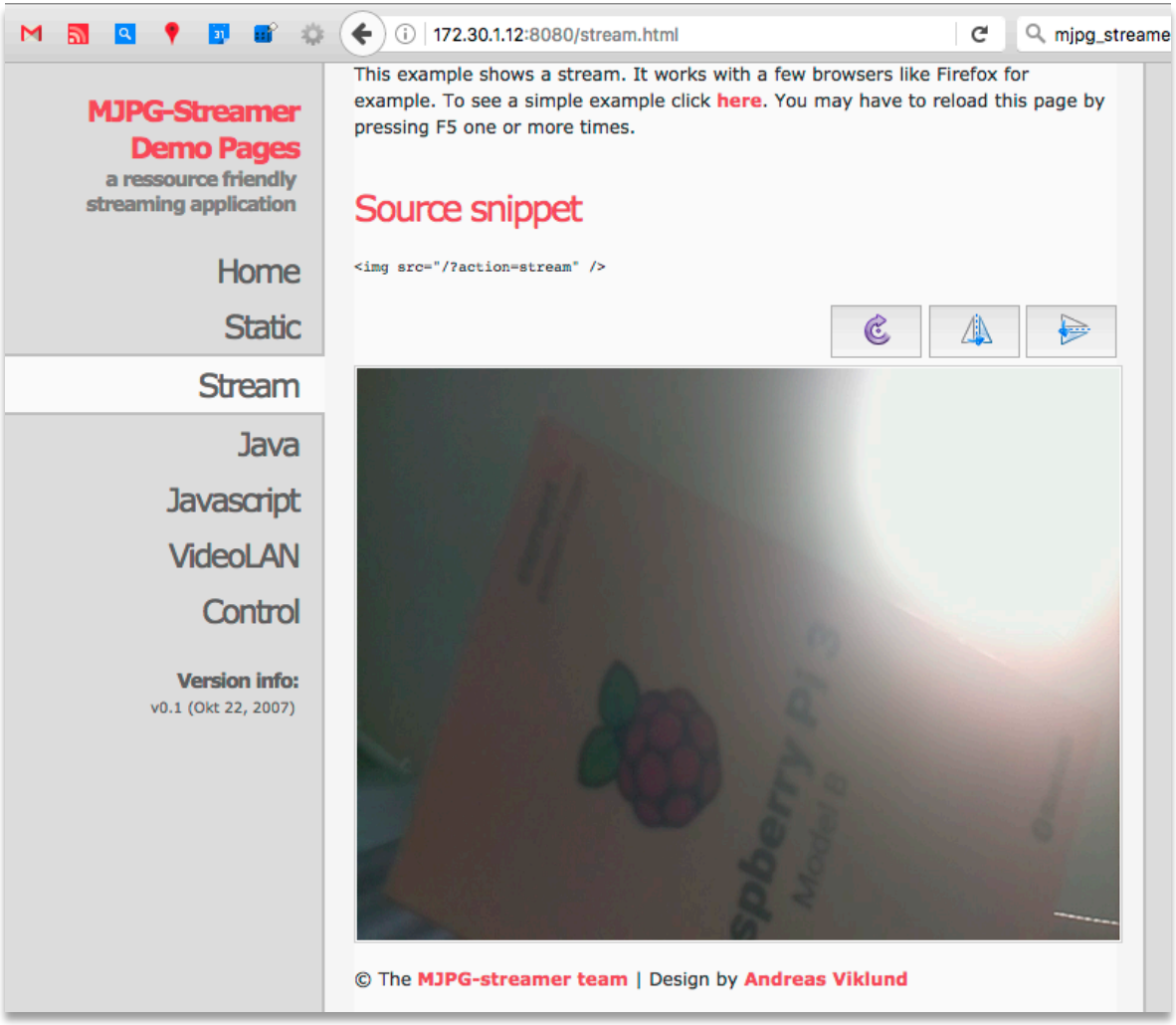


❖ MJPG-Streamer

- 의존 패키지 설치
  - sudo apt-get install libjpeg8-dev imagemagick libv4l-dev
- 소스코드 다운로드
  - wget http://sourceforge.net/code-snapshots/svn/m/mj/mjpg-streamer/code/mjpg-streamer-code-182.zip
  - unzip mjpg-streamer-code-182.zip
  - cd mjpg-streamer-code-182/ mjpg-streamer
- 패치적용
  - wget https://github.com/swkim01/RaspberryPiWithIOT/raw/master/ch7/input_uvc_patch
  - patch -p0 < input_uvc_patch
- 빌드 및 설치
  - make USE_LIBV4L2=true clean all
  - sudo make DESTDIR=/usr install
- 실행
  - mjpg_streamer -i "input_uvc.so -d /dev/video0 -n -f 30 -r 1280x720" -o "output_http.so -n -w /usr/www"

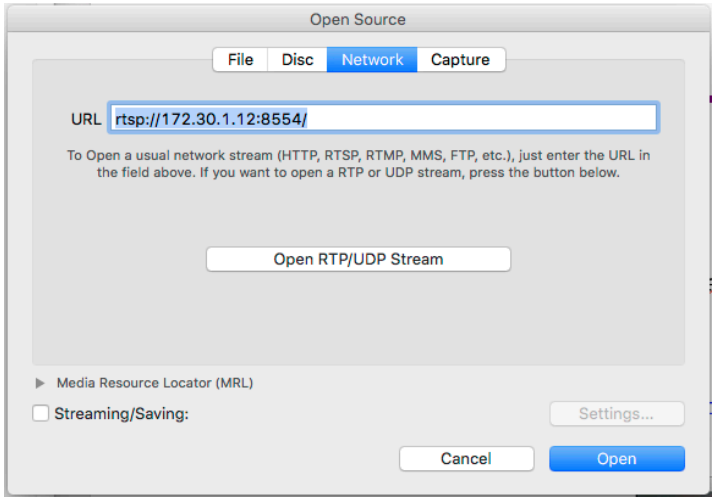
❖ MJPG-Streamer

- 결과 확인
  - 웹 브라우저 접속
  - http://ip_address:8080



❖ VLC

- 설치
  - sudo apt-get install vlc
- 스트리밍 시작
  - raspivid -o - -t 0 -n | cvlc -vvv stream:///dev/stdin --sout '#rtp{sdp=rtsp://:8554/}' :demux=h264
- VLC 영상 확인
  - PC 용 VLC 설치
  - https://www.videolan.org/vlc/download-windows.ko.html
  - File > openNetwork > rtsp://IP_Address:8554/



세부목차

OpenCV 영상처리

❖ OpenCV

- http://opencv.org/
- Open Source Computer Vision
- 실시간 컴퓨터 비전 라이브러리
- 인텔에서 개발
- 영상처리에 필요한 다양한 기능 제공
  - 필터
  - 분류기
  - 특징 추출
  - 움직임 검출
  - 비디오 캡쳐
- 설치
  - sudo apt-get install libopencv-dev
  - sudo apt-get install python-opencv
- v4l2 로딩
  - sudo apt-get install v4l-utils
  - sudo modprobe bcm2835-v4l2
  - ls /dev/video0

❖ 비디오 캡쳐

```
import cv2

cam = cv2.VideoCapture(0)

while True:
    ret,img = cam.read()
    cv2.imshow('Video Capture', img)
    key = cv2.waitKey(10)
    if key==27:
        break
    if key== ord(' '):
        cv2.imwrite('capture.jpg', img)
```

❖ 움직임 감지

- 차영상

```
import cv2
import numpy as np

def diffImage(i):
    diff0 = cv2.absdiff(i[0], i[1])
    diff1 = cv2.absdiff(i[1], i[2])
    return cv2.bitwise_and(diff0, diff1)

def getGrayCameraImage(cam):
    img=cam.read()[1]
    gimg=cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    return gimg

def updateCameraImage(cam, i):
    i[0] = i[1]
    i[1] = i[2]
    i[2] = getGrayCameraImage(cam)
```

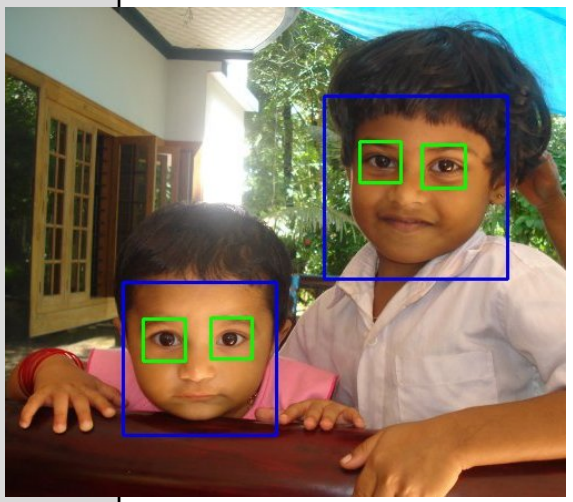
❖ 움직임 감지 <앞에서 계속>

```
if __name__ == "__main__":
    thresh = 32
    cam = cv2.VideoCapture(0)
    i = [None, None, None]
    for n in range(3):
        i[n] = getGrayCameraImage(cam)

    while True:
        diff = diffImage(i)
        ret,thrimg=cv2.threshold(diff, thresh, 1, cv2.THRESH_BINARY)
        count = cv2.countNonZero(thrimg)
        if (count > 20):
            nz = np.nonzero(thrimg)
            cv2.rectangle(diff,(min(nz[1]),min(nz[0])),(max(nz[1]),max(nz[0])),(255,0,0),2)
            cv2.rectangle(i[0],(min(nz[1]),min(nz[0])),(max(nz[1]),max(nz[0])),(0,0,255),2)
            cv2.imwrite('detect.jpg',i[0])

        cv2.imshow('Detecting Motion', diff)
        updateCameraImage(cam, i)
        key = cv2.waitKey(10)
        if key == 27:
            break
```

❖ 사진 얼굴 인식

- https://github.com/opencv/opencv/tree/master/data/haarcascades

```
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
img = cv2.imread('children.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



❖ 카메라 얼굴 인식 <다음에 계속>

```
import numpy as np
import cv2
import cv2.cv as cv

def clock():
    return cv2.getTickCount() / cv2.getTickFrequency()

def draw_str(dst, (x, y), s):
    cv2.putText(dst, s, (x+1, y+1), cv2.FONT_HERSHEY_PLAIN, 1.0, (0, 0, 0), thickness = 2, lineType=cv2.CV_AA)
    cv2.putText(dst, s, (x, y), cv2.FONT_HERSHEY_PLAIN, 1.0, (255, 255, 255), lineType=cv2.CV_AA)

def detect(img, cascade):
    rects = cascade.detectMultiScale(img, scaleFactor=1.1, minNeighbors=3, minSize=(80, 80), flags =
cv.CV_HAAR_SCALE_IMAGE)
    if len(rects) == 0:
        return []
    rects[:,2:] += rects[:,:2]
    return rects

def draw_rects(img, rects, color):
    for x1, y1, x2, y2 in rects:
        cv2.rectangle(img, (x1, y1), (x2, y2), color, 2)
```

❖ 카메라 얼굴 인식 <앞에서 계속>

```
if __name__ == '__main__':
    cascade_fn = 'haarcascade_frontalface_default.xml'
    cascade = cv2.CascadeClassifier(cascade_fn)
    cam = cv2.VideoCapture(0)
    cam.set(cv2.cv.CV_CAP_PROP_FRAME_WIDTH, 320)
    cam.set(cv2.cv.CV_CAP_PROP_FRAME_HEIGHT, 240)

    while True:
        ret, img = cam.read()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        gray = cv2.equalizeHist(gray)
        t = clock()
        rects = detect(gray, cascade)
        vis = img.copy()
        draw_rects(vis, rects, (0, 255, 0))

        dt = clock() - t
        draw_str(vis, (20, 20), 'time: %.1f ms' % (dt*1000))
        cv2.imshow('facedetect', vis)
        if 0xFF & cv2.waitKey(5) == 27:
            break
    cv2.destroyAllWindows()
```