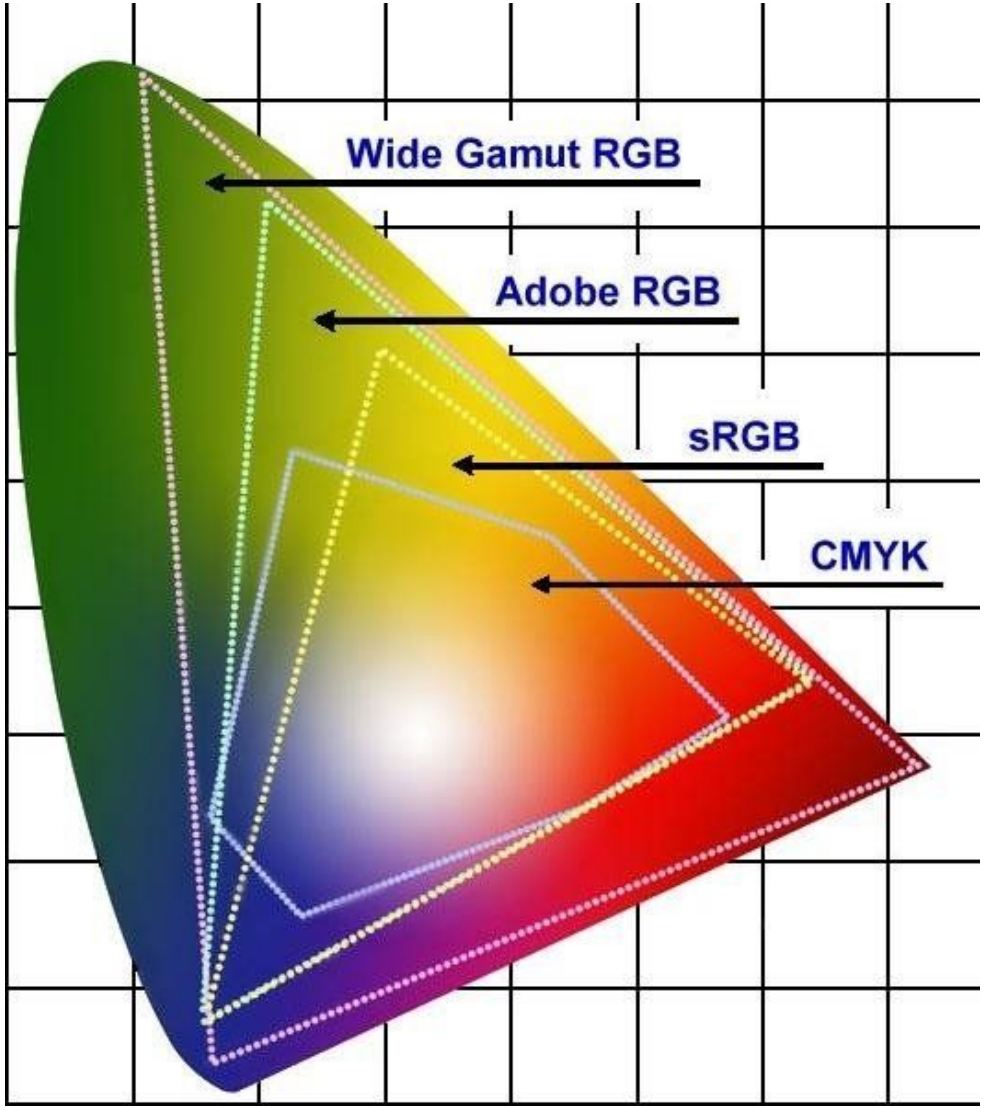
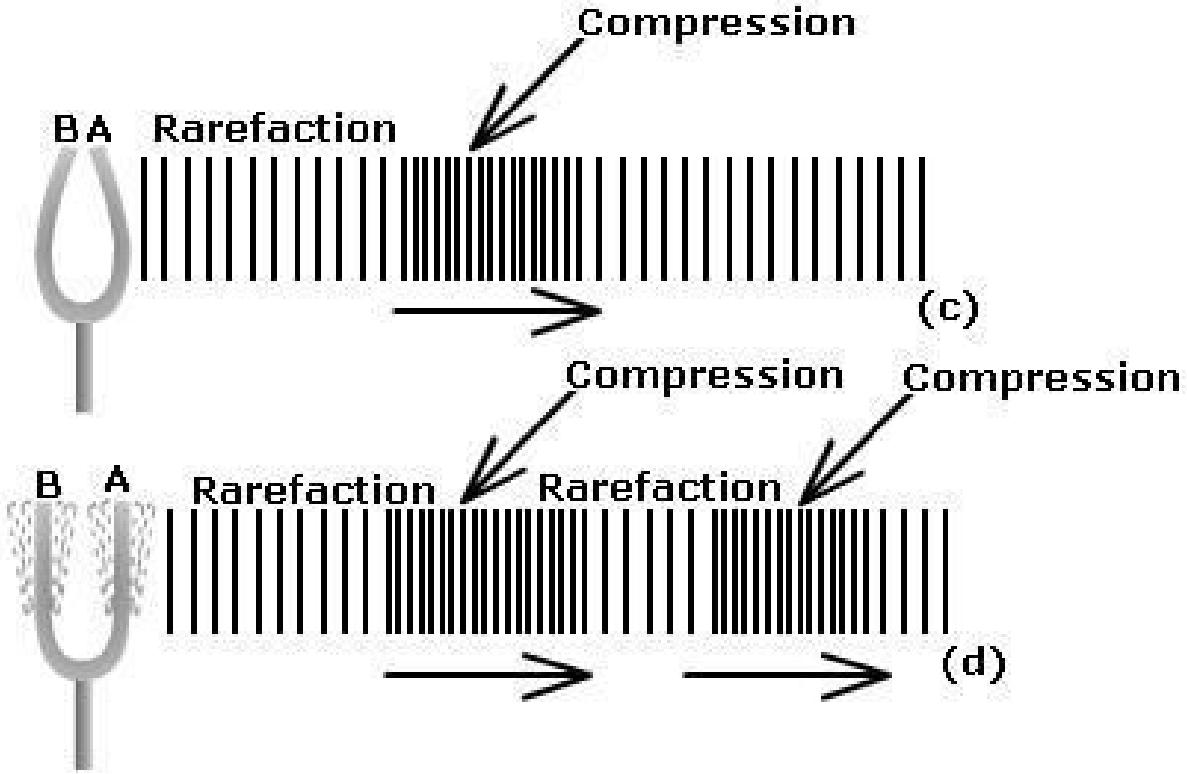
Linux Multimedia Programming

Python

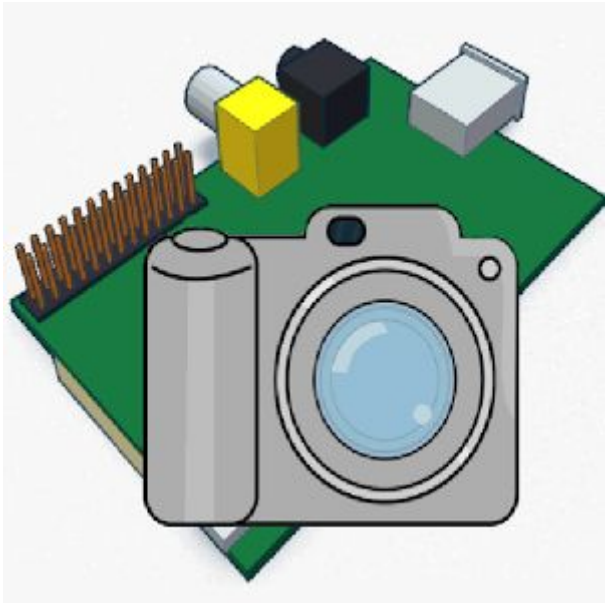
❖ 인간 감각기관 통한 정보 인식과 전달 합성어, 디지털은 주로 시각과 청각.





Pi Camera(1)

- setup
  - Touch Screen
    - Launch Raspberry Pi Configuration from the RaspiCam Remote
    - Navigate to the Interfaces tab
    - Select Enabled next to Camara



~$ sudo rpi-update → update Formware
~$ sudo aptitude update
~$ sudo aptitude upgrade
~$ reboot
~$ sudo aptitude install v4l-utils
~$ sudo modprobe bcm2835-v4l2
~$ lsusb
~$ ls /dev/video0
~$ v4l2-dbg -D /dev/video0 → information

- Auto Kernel Modules

~$ vi /etc/modules

  - add line : bcm2835-v4l2

- RPi-Cam-Web-Interface
  - http://elinux.org/RPi-Cam-Web-Interface

Pi Camera(2)

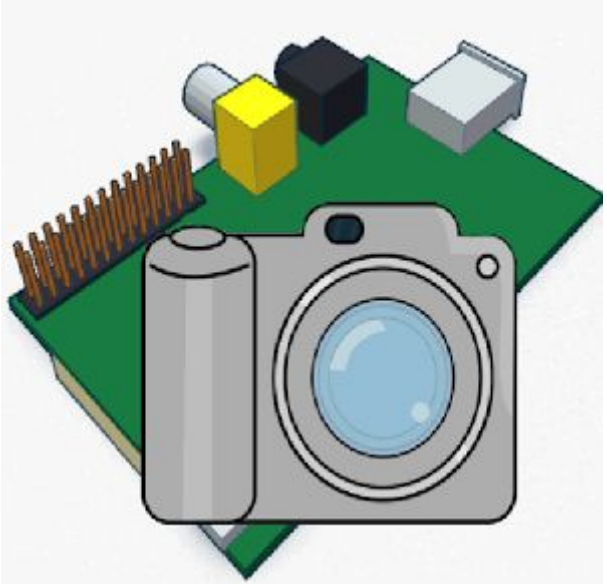
- raspistill -o cam.jpg
  ~$ raspistill -t 2000 -o image.jpg : 2초 후 파일 저장
  ~$ raspistill -t 2000 -o image.jpg –w 640 –h 480 : 이미지 사이즈 지정
  ~$ raspistill -t 2000 -o image.jpg –q 5 : jpeg Quality 지정
  ~$ raspistill -t 2000 -o image.jpg -p 100,100,300,200 : 프리뷰 사이즈 결정 (100,100위치/ 가로 300, 세로 200)
  ~$ raspistill -t 2000 -o image.jpg -n : 프리뷰 사용 않음
  ~$ raspistill -t 2000 -o image.png –e png : PNG 파일 저장
  ~$ raspistill -t 2000 -o image.jpg -ifx emboss : 엠보싱 효과
  ~$ raspistill -t 2000 : 2초 동안 프리뷰 저장 없음
  ~$ raspistill -t 600000 -tl 10000 -o image_num_%d_today.jpg : 일정 간격 저장 (10분 간 매 10초씩 저장)
- raspivid -o video.h264
  ~$ raspivid -t 5000 -o video.h264 : 5초 동안 캡쳐 저장(1080p, 30f)
  ~$ raspivid -t 5000 -o video.h264 -b 3500000 : 3.5Mbits/s 전송률 5초 동안 저장
  ~$ raspivid -t 5000 -o video.h264 -fps 5 : 5fps 프레임 5초 동안 저장

~$ omxplayer video.h264

Pi Camera(3)

```
// rcamera.py
  from picamera import PiCamera
   from time import sleep
   camera = PiCamera()
   camera.start_preview()
   sleep(10)
   camera.capture('/home/pi/Desktop/image.jpg')
   #camera.start_recording('/home/pi/video.h264')
   #sleep(10)
   #camera.stop_recording()
   camera.stop_preview()
```

RaspiCam Remote



~$ omxplayer video.h264 → move with scp command : as file size

Try - App On Raspbian

❖ motion : 영상 모션 움직임 포착하여 이벤트 발생

➢ framerate - Frame Per second, threshold - Minimal For capture image.

~$ sudo aptitude install motion
~$ sudo modprobe bcm2835-v4l2
~$ sudo motion
~$ tail -f /var/log/motion/motion.log → 카메라 앞 움직이기.
~$ ls /var/lib/motion
~$ sudo vi /etc/motion/motion.conf
stream_localhost off # modify on → off
http://192.168.0.30:8081/ # with Browser

❖ vlc : 영상 움직임 리모트 전송.

~$ sudo aptitude install vlc
~$ uv4l --driver raspicam --auto-video_nr --framerate 25 → 활성화

➢ YUV or MJPG 정지 영상 Streaming

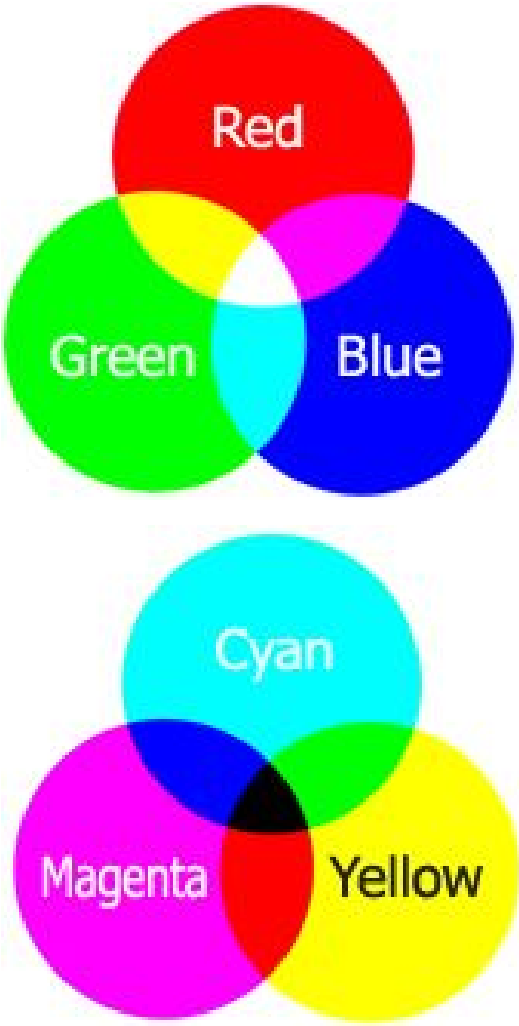
~$ cvlc v4l2:///dev/video0:fps=5:chroma=mjpg --sout
'#standard{access=http,mux=**mpjpeg**,dst=:8083/}'

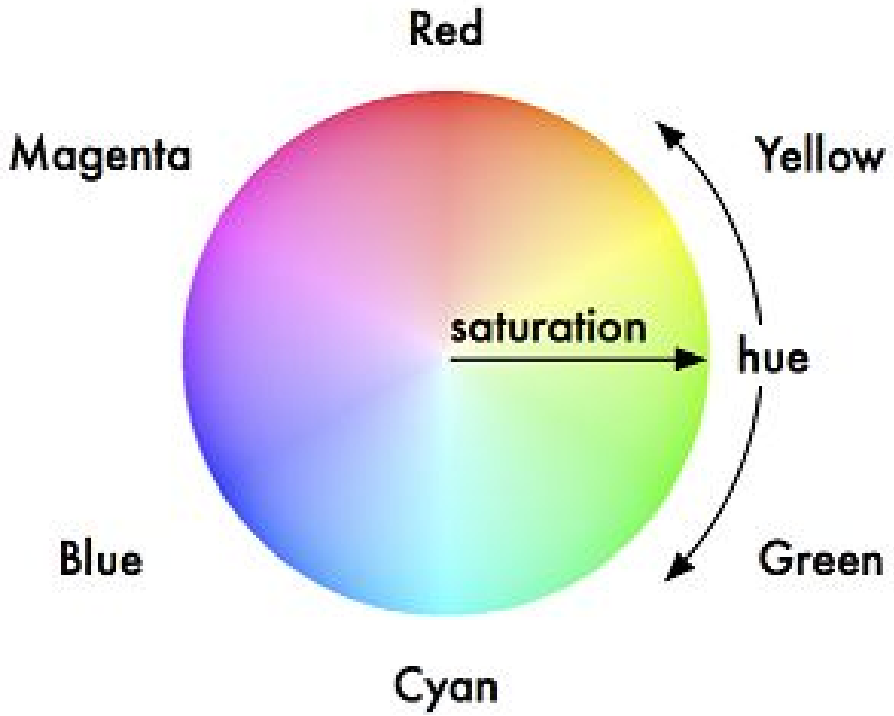
➢ RTP(Real-Time Protocol)

~$ cvlc v4l2:///dev/video0:width=640:height=480:chroma=**h264** --sout
'#rtp{sdp=rtsp://:8083/}'
http://192.168.0.30:8083/ # with Browser

Color Models

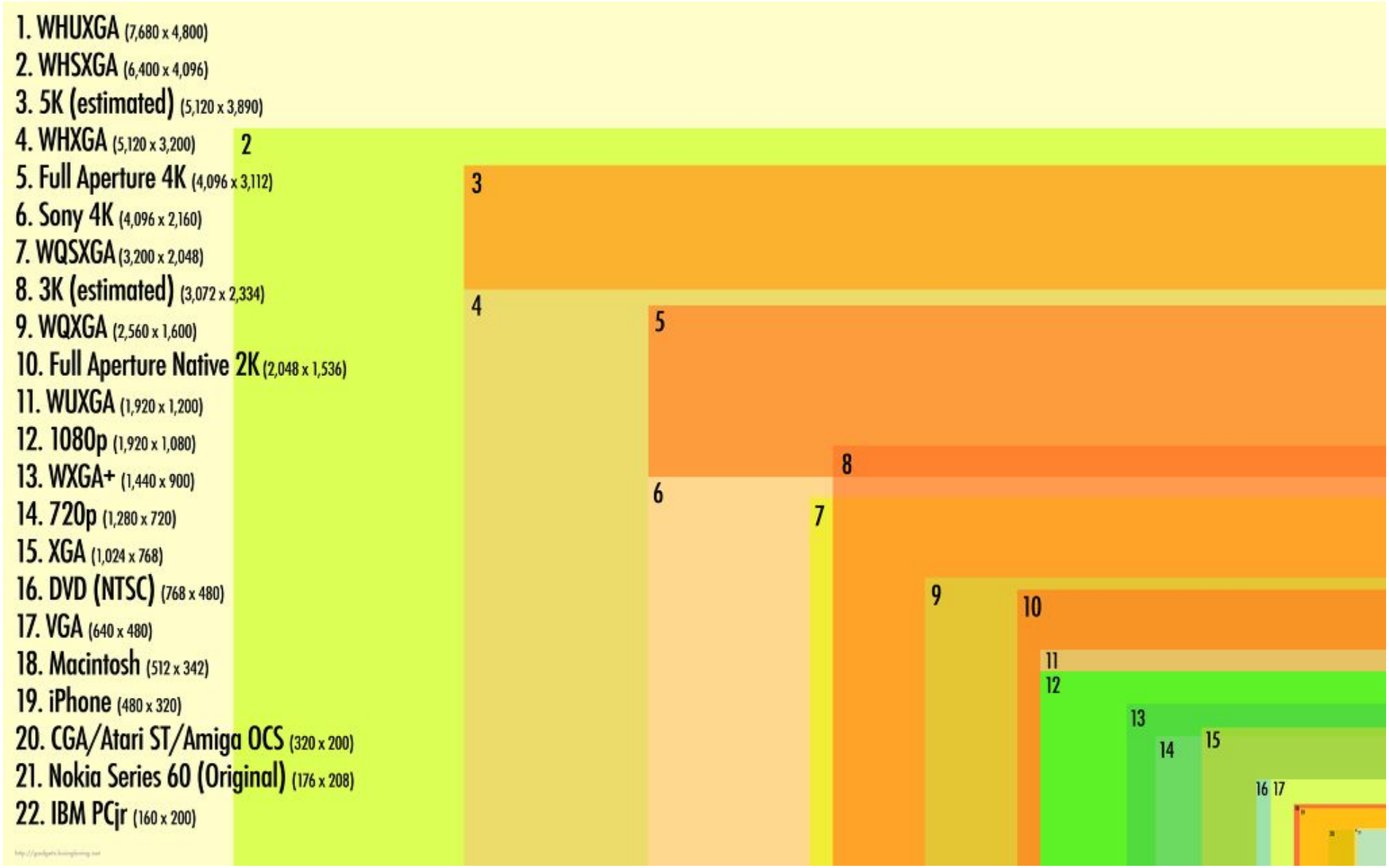
- RGB : 빨강,녹색,파랑 빛의 삼원색, 모니터나 영화관 등
- CMYK : 청록,자홍,노랑,검은색, 출판
- HSI : 채도(Hue),채도(Saturation),명도(Intensity),영상
- YUV : 밝기휘도(Luminace), 색상(Chrominance), JPEG/MPEG코덱 디지털방송.





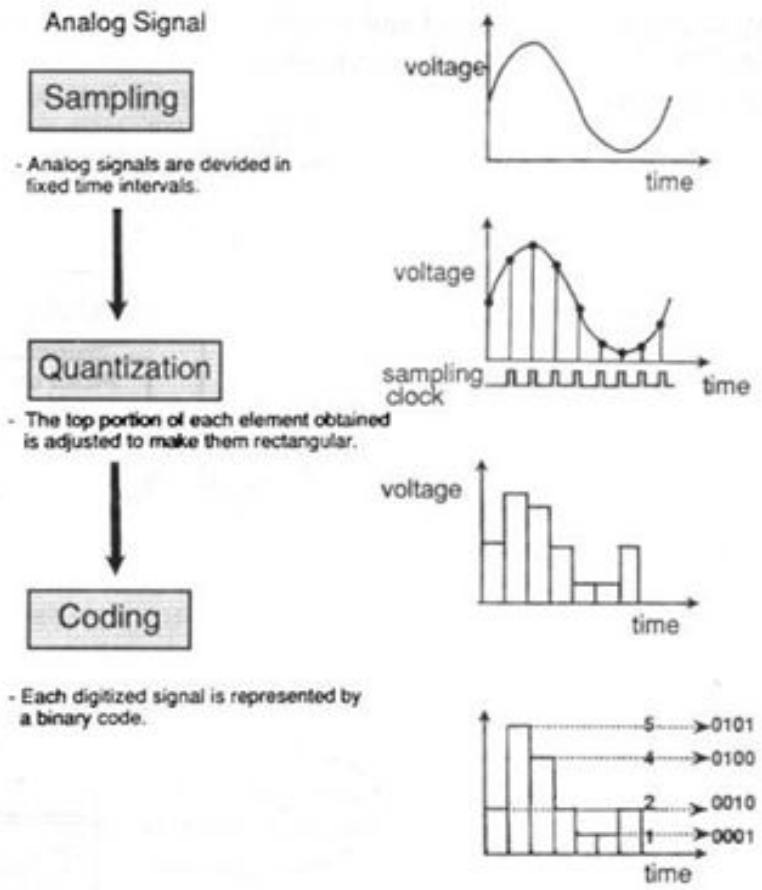
Pixel

- 불연속적 점.
- CIF(Common Intermediate Format) : 화상 회의 시스템 비디오 형식



이미지 아날로그/디지털 변환

- 표본화(Sampling) : 해상도(픽셀) 산출
- 양자화(Quantization) : 픽셀 대한 수치 산출(컬러->RGB, 흑백 -> 밝기), 1/8/15/16/24/32 등 비트 표현.
- 부호화(Encoding) : 사용 색상(Indexed) 대한 매핑 팔레트 이용 용량 줄임.
-



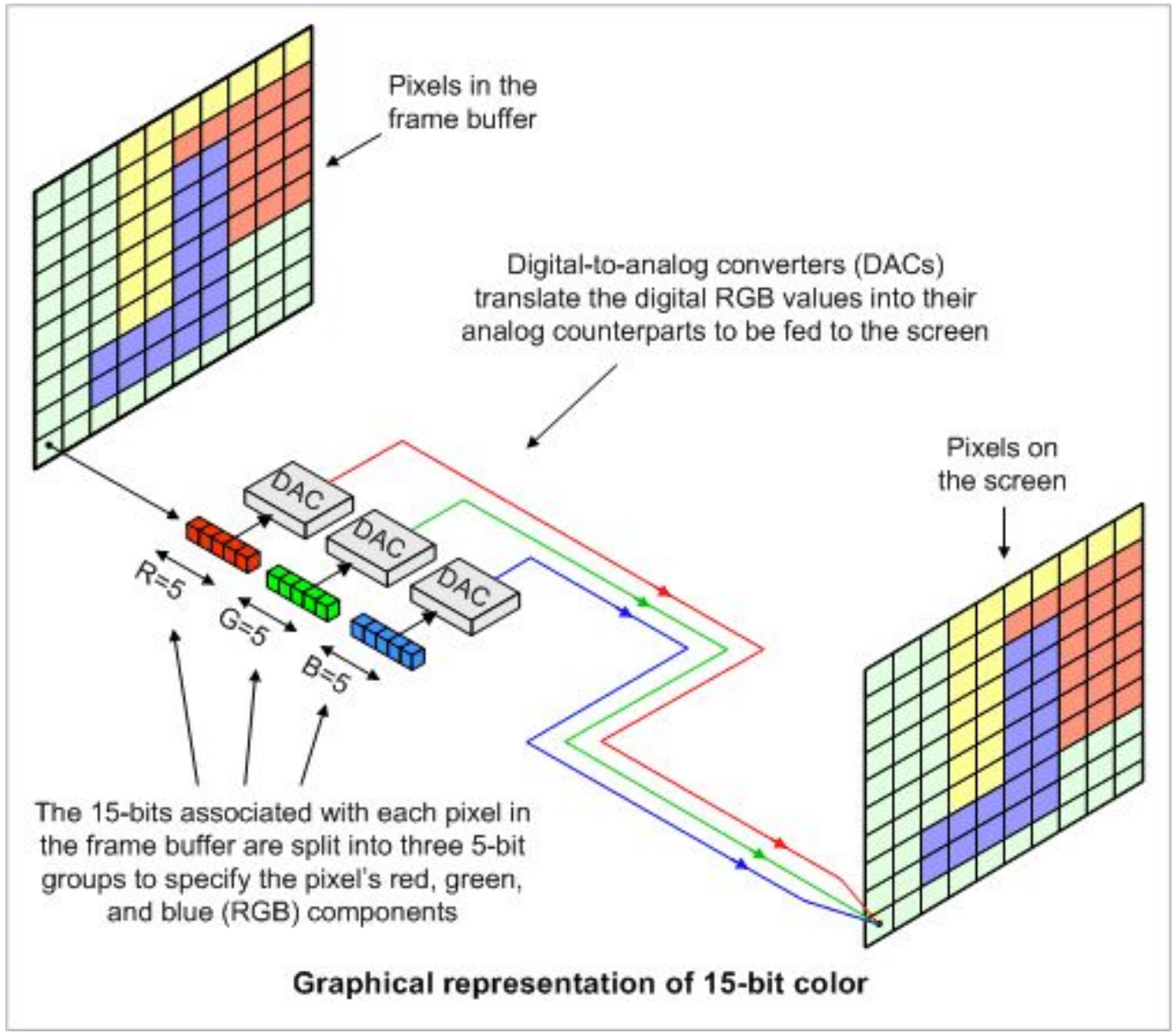
24 bits

- 256 × 256 × 265 = 16,777,216 colors

32 bits

Frame Buffer

- 화면 출력 위한 메모리 공간, 생성된 래스터 이미지가 모니터 전달 전 잠시 저장.
- PC -> 그래픽 카드, 임베디드 -> LCD 컨트롤러



Graphical representation of 15-bit color

Setup OpenCV3.3 with Python3.5(opencv2_install.sh)

❖ Download opencv-3.3.0.zip , opencv_contrib-3.3.0.zip

```
~$ sudo aptitude install -y build-essential cmake pkg-config \
    ffmpeg libavcodec-dev libavformat-dev libswscale-dev \
    libv4l-dev libxvidcore-dev libx264-dev libgtk2.0-dev libatlas-base-dev \
    gfortran python3-dev opencv-data libopencv-dev python3-numpy \
    x264 v4l-utils qt5-default
~$ cd ~/?/opencv-3.3.0/ ;       mkdir build;       cd build
~$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
    -D INSTALL_PYTHON_EXAMPLES=ON \
    -D INSTALL_C_EXAMPLES=OFF -D WITH_GSTREAMER=ON \
    -D OPENCV_EXTRA_MODULES_PATH=~/?/opencv_contrib-3.3.0/modules \
    -D PYTHON_EXECUTABLE=/usr/bin/python3.5 \
     -D WITH_TBB=ON -D BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON \
     -D WITH_QT=ON -D WITH_OPENGL=ON \
    -D BUILD_EXAMPLES=ON ..
~$ make -j4        ~$ sudo make install
~$ cd /usr/local/lib/python3.5/dist-packages
~$ sudo ln -s cv2.cpython-35m-x86_64-linux-gnu.so cv2.so
    >>> import cv2
    >>> cv2.__version__
    >>> cap = cv2.VideoCapture(0);   cap.isOpened();  cap.release()
```

Try - Test OpenCV with Camera (SimpleCamera.py)

❖ 실행결과

press ESC

❖ 알아가기

~$ lsusb ~$ ls -l /dev/video*

❖ 따라하기

```
import cv2
cap = cv2.VideoCapture(0)                    # double check /dev/video?
# Read until video is completed
while(cap.isOpened()):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # Display the resulting frame
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) == 27:
        break
# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

The Mat Class

- **Mat**(int rows, int cols, int type)
- **Mat**(Size size, int type)
- **Mat**(int rows, int cols, int type, const Scalar& s)
- **Mat**(Size size, int type, const Scalar& s)
- **Mat**(const Mat& m)
- **Mat**(int rows, int cols, int type, void* data, size_t step=AUTO_STEP)
- **Mat**(Size size, int type, void* data, size_t step=AUTO_STEP)
- **Mat**(const Mat& m, const Range& rowRange, const Range& colRange=Range::all())
- **Mat**(const Mat& m, const Rect& roi)
- **Mat**(const CvMat* m, bool copyData=false)
- **Mat**(const IplImage* img, bool copyData=false)
- **Mat**(int ndims, const int* sizes, int type)
- **Mat**(int ndims, const int* sizes, int type, const Scalar& s)
- **Mat**(int ndims, const int* sizes, int type, void* data, const size_t* steps=0)
- **Mat**(const Mat& m, const Range* ranges)

Try - read / write image(ReadWriteImage.py)

❖ 실행결과

press ESC or press S

images/load_gray.png

❖ 따라하기

```
import cv2
# Loads image in grayscale mode
img = cv2.imread('images/load_image.jpg',cv2.IMREAD_GRAYSCALE)
cv2.imshow('image',img)
k = cv2.waitKey(0) & 0xFF          # 64 bit
if k == 27:         # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'): # wait for 's' key to save and exit
    cv2.imwrite('images/load_gray.png',img)
    cv2.destroyAllWindows()
```

❖ 해보기 (Read an image flag)

- ➢ cv2.**IMREAD_GRAYSCALE** (0): Loads image in grayscale mode
- ➢ cv2.**IMREAD_COLOR** (1): Loads a color image. Any transparency of image will be neglected. It is the default flag.
- ➢ cv2.**IMREAD_UNCHANGED** (-1): Loads image as such including alpha channel

Try - Drawing Functions(DrawingFunctions.py)

❖ 실행결과
각 도형과 글자 출력.

❖ 따라하기

```
import numpy as np
img = np.zeros((512,512,3), np.uint8) # Create a black image
# Draw a diagonal blue line with thickness of 5 px
img = cv2.line(img,(0,0),(511,511),(255,0,0),5)
img = cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)
img = cv2.circle(img,(447,63), 63, (0,0,255), -1)
img = cv2.ellipse(img,(256,256),(100,50),0,0,180,255,-1)
pts = np.array([[10,5],[20,30],[70,20],[50,10]], np.int32)
pts = pts.reshape((-1,1,2))
img = cv2.polylines(img,[pts],True,(0,255,255))

font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(img,'OpenCV',(10,500), font, 4,(255,255,255),2,cv2.LINE_AA)
```

Try - Mouse as a Paint-Brush(MousePaint-Brush.py)

❖ 실행결과

Click Lfe press ESC

❖ 따라하기

```
import cv2
import numpy as np
# mouse callback function
def draw_circle(event,x,y,flags,param):
    if event == cv2.EVENT_FLAG_LBUTTON:
        cv2.circle(img,(x,y),10,(255,0,0),-1)
# Create a black image, a window and bind the function to window
img = np.zeros((512,512,3), np.uint8)
cv2.namedWindow('image')
cv2.setMouseCallback('image',draw_circle)
while(1):
    cv2.imshow('image',img)
    if cv2.waitKey(20) & 0xFF == 27:        break
cv2.destroyAllWindows()
```

Try - Trackbar as ColorPalette(TrackbarColorPalette.py)

❖ 실행결과

Move Trackbar.

❖ 따라하기

```
import cv2;    import numpy as np
def nothing(x):     pass
img = np.zeros((300,512,3), np.uint8)
cv2.namedWindow('image')
cv2.createTrackbar('R','image',0,255,nothing)
cv2.createTrackbar('G','image',0,255,nothing)
cv2.createTrackbar('B','image',0,255,nothing)
while(1):
    cv2.imshow('image',img)
    k = cv2.waitKey(1) & 0xFF
    if k == 27:
        break
    r = cv2.getTrackbarPos('R','image')
    g = cv2.getTrackbarPos('G','image')
    b = cv2.getTrackbarPos('B','image')
    img[:] = [b,g,r]
cv2.destroyAllWindows()
```

Try - Accessing & Modifying pixel values(AccessingModifying_Pixels.py)

❖ 실행결과

```
pixel img[100,100] values [ 0  5 12]
accessing only blue pixel 0
Modifying img[100,100] pixel values to [255,255,255] [255 255 255]
accessing RED value 173
modifying RED value img.itemset((10,10,2),100) 173
Image Properties - shape (360, 500, 3)
Image Properties - size 540000
Image Properties - dtype uint8
```

❖ 따라하기

```
import cv2; img = cv2.imread('images/load_image.jpg')
print ('pixel img[100,100] values',img[100,100])
print('accessing only blue pixel',img[100,100,0])
img[100,100] = [255,255,255]
print('Modifying img[100,100] pixel values to [255,255,255]',img[100,100])
print('accessing RED value',img.item(10,10,2))
print('modifying RED value img.itemset((10,10,2),100)',img.item(10,10,2))
print('Image Properties - shape',img.shape)
print('Image Properties - size', img.size)
print('Image Properties - dtype',img.dtype)
```

Try - Splitting and Merging Image Channels(SplittingMerging_Channels.py)

❖ 실행결과

6 개 dialog 생성.

❖ 따라하기

```
img_tmp = img
b,g,r = cv2.split(img)  #is a costly operation
cv2.imshow('blue image Channel',b)
cv2.imshow('green image Channel',g)
cv2.imshow('red image Channel',r)

merge_img = cv2.merge((b,g,r))
cv2.imshow('merge image',merge_img)
img_tmp[:,:,2]=0    #isn't a costly operation
cv2.imshow('blue image Channel by Numpy',img_tmp)

roi = img[180:340, 330:490]     # region of images
img_tmp[173:333, 100:260] = roi
cv2.imshow('Image ROI',img_tmp)
```

Try - Image Addition(ImageAddition.py)

❖ 실행결과

5개 dialog 생성

❖ 따라하기

```
img = cv2.imread('images/load_image.jpg')
img2 = cv2.imread('images/window_image.jpg')
cv2.imshow('load image',img)
cv2.imshow('window image',img2)
cv2.imshow('load + window image',img+img2)
cv2.imshow('add (load,window) image',cv2.add(img,img2))
def nothing(x):
    pass
cv2.namedWindow('Blending image')
cv2.createTrackbar('blending','Blending image',0,100,nothing)
mix = 1
while(True):
    blending_img=cv2.addWeighted(img2, float(100-mix)/100, img, float(mix)/100,0)
    cv2.imshow('Blending image',blending_img)
    k = cv2.waitKey(1) & 0xFF
    if k == 27:
        break
    mix = cv2.getTrackbarPos('blending', 'Blending image')
```

Try - Bitwise Operations(BitwiseOperations.py)

❖ 실행결과

1개 dialog 생성.

❖ 따라하기

```
img = cv2.imread('images/load_image.jpg')
img2 = cv2.imread('images/opencv_logo.png')
rows,cols,channels = img2.shape
roi = img[0:rows, 0:cols ]
# Now create a mask of logo and create its inverse mask also
img2gray = cv2.cvtColor(img2,cv2.COLOR_BGR2GRAY)
ret, mask = cv2.threshold(img2gray, 10, 255, cv2.THRESH_BINARY)
mask_inv = cv2.bitwise_not(mask)
# Now black-out the area of logo in ROI
img1_bg = cv2.bitwise_and(roi,roi,mask = mask_inv)
# Take only region of logo from logo image.
img2_fg = cv2.bitwise_and(img2,img2,mask = mask)
# Put logo in ROI and modify the main image
dst = cv2.add(img1_bg,img2_fg)
img[0:rows, 0:cols ] = dst
cv2.imshow('add(img1_bg,img2_fg)',dst)
```

Try - find HSV values to track(findHSVToTrack.py)

❖ 실행결과

변환 가능 종류 list

BGR to HSV valuses

❖ 따라하기

```
import numpy as np
flags = [i for i in dir(cv2) if i.startswith('COLOR_')]
print (flags)
#BGR --> Gray and BGR --> HSV
blue = np.uint8([[[ 255,0,0 ]]])
green = np.uint8([[[ 0,255,0 ]]])
red = np.uint8([[[ 0,0,255 ]]])
hsv_blue = cv2.cvtColor(blue,cv2.COLOR_BGR2HSV)

print('Convert HSV for blue', hsv_blue)
hsv_green = cv2.cvtColor(green,cv2.COLOR_BGR2HSV)
print('Convert HSV for green', hsv_green)
hsv_red = cv2.cvtColor(red,cv2.COLOR_BGR2HSV)
print('Convert HSV for red', hsv_red)
```

Try - Object Tracking(ObjectTracking_inRange.py)

❖ 실행결과

Blue color Tracking

❖ 따라하기

```
import numpy as np
cap = cv2.VideoCapture(0)
while(True):
    _, frame = cap.read()      # Take each frame
   # Convert BGR to HSV
   hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
   # define range of blue color in HSV
   lower_blue = np.array([110,50,50])
   upper_blue = np.array([130,255,255])
   # Threshold the HSV image to get only blue colors
   mask = cv2.inRange(hsv, lower_blue, upper_blue)
   # Bitwise-AND mask and original image
   result = cv2.bitwise_and(frame,frame, mask= mask)
   cv2.imshow('frame',frame)
   cv2.imshow('mask',mask)
   cv2.imshow('result',result)
```

Try - Simple Thresholding(SimpleThresholding.py)

❖ 실행결과

각 Threshold 적용 표시.

❖ 따라하기

```
from matplotlib import pyplot as plt
img = cv2.imread('images/radial_gradient.png',0)
ret,thresh1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)
ret,thresh3 = cv2.threshold(img,127,255,cv2.THRESH_TRUNC)
ret,thresh4 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO)
ret,thresh5 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO_INV)
titles = ['Original Image','BINARY','BINARY_INV',
                'TRUNC','TOZERO','TOZERO_INV']
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]

for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```

Try - Adaptive Thresholding(AdaptiveThresholding.py)

❖ 실행결과

❖ 따라하기

```
from matplotlib import pyplot as plt
img = cv2.imread('images/sudoku.jpg',cv2.IMREAD_GRAYSCALE)
img = cv2.medianBlur(img,5)
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,\
            cv2.THRESH_BINARY,11,2)
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,\
            cv2.THRESH_BINARY,11,2)
titles = ['Original Image', 'Global Thresholding (v = 127)',
            'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']
images = [img, th1, th2, th3]
for i in range(4):
    plt.subplot(2,2,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```

Try - Otsu's Binarization(OtsusBinarizationThresholding.py)

```
img = cv2.imread('images/noisy.jpg',0)
ret1,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
# Otsu's thresholding
ret2,th2 = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
# Otsu's thresholding after Gaussian filtering
blur = cv2.GaussianBlur(img,(5,5),0)
ret3,th3 = cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
# plot all the images and their histograms
images = [img, 0, th1,img, 0, th2,blur, 0, th3]
titles = ['Original Noisy Image','Histogram','Global Thresholding (v=127)',
          'Original Noisy Image','Histogram',"Otsu's Thresholding",
          'Gaussian filtered Image','Histogram',"Otsu's Thresholding"]
for i in range(3):
    plt.subplot(3,3,i*3+1),plt.imshow(images[i*3],'gray')
    plt.title(titles[i*3]), plt.xticks([]), plt.yticks([])
    plt.subplot(3,3,i*3+2),plt.hist(images[i*3].ravel(),256)
    plt.title(titles[i*3+1]), plt.xticks([]), plt.yticks([])
    plt.subplot(3,3,i*3+3),plt.imshow(images[i*3+2],'gray')
    plt.title(titles[i*3+2]), plt.xticks([]), plt.yticks([])
```

Try - Image Transformations (ImageTransformations.py)

❖ 실행결과

4개 Dialog 생성.

❖ 따라하기

```
img = cv2.imread('images/window_image.jpg')
cv2.imshow('Original Image',img)

res = cv2.resize(img,None,fx=2, fy=2, interpolation = cv2.INTER_CUBIC)
cv2.imshow('Scaling Image',res)

rows,cols,tmp = img.shape
M = np.float32([[1,0,100],[0,1,50]])
dst = cv2.warpAffine(img,M,(cols,rows))
cv2.imshow('Shifting Image',dst)

M2 = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst2 = cv2.warpAffine(img,M2,(cols,rows))
cv2.imshow('Rotating Image',dst2)
```

Try - Capture Video from Camera(CaptureVideofromCamera.py)

❖ 실행결과

press key ‘q’

❖ 따라하기

```
import cv2
cap = cv2.VideoCapture(1)
#cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
#cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
#cap.set(cv2.CAP_PROP_FPS, 7)
if (cap.isOpened()== False):
    print("Error opening video stream or file")
while(cap.isOpened()):                  # Read until video is completed
    ret, frame = cap.read()             # Capture frame-by-frame
    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow('frame',gray)            # Display the resulting frame
    if cv2.waitKey(0) == ord('q'):
        break
cap.release()                           # When everything done, release the capture
cv2.destroyAllWindows()
```

Try - save image from diff motion(motion_cv2.py)

```
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)
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 = numpy.nonzero(thrimg)
            cv2.imwrite('detect'+str(time.time())+'.jpg',i[0])
        cv2.imshow('Detecting Motion', diff)
        updateCameraImage(cam, i)        # process next image
```

Try - Web Server Push With image(mjpgstream_flask.py)

❖ 실행결과

http://127.0.0.1:5000/

❖ 따라하기

```
class VideoCamera(object):
    def __init__(self):        self.video = cv2.VideoCapture(0)
    def __del__(self):         self.video.release()
    def get_frame(self):
        success, image = self.video.read(); ret, jpeg = cv2.imencode('.jpg', image)
        return jpeg.tobytes()
@app.route('/')
def index():
    return "<HTML><BODY><img src="+url_for('video_feed')+"></BODY></HTML>"
def gen(camera):
    while True:
        frame = camera.get_frame()
        yield (b'--frame\r\nContent-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
@app.route('/video_feed')
def video_feed():
    return Response(gen(VideoCamera()),
                    mimetype='multipart/x-mixed-replace; boundary=frame')
```


수고하셨습니다.