Chapter

2

실습환경구축 및 OpenCV 개요

OpenCV

❖ OpenCV(Open Computer Vision)

- 오픈소스 컴퓨터비전 C/C++라이브러리
  - 2,500개가 넘는 알고리즘으로 구성
    - 영상 처리, 컴퓨터 비전, 기계 학습과 관련된 전통적인 알고리즘
      - » 얼굴 검출과 인식, 객체 인식, 객체 3D 모델 추출, 스테레오 카메라에서 3D 좌표 생성
      - » 고해상도 영상 생성을 위한 이미지 스티칭, 영상 검색, 적목 현상 제거, 안구 운동 추적
      - » 4만 7천 이상의 사용자 그룹과 1,800만 번 이상의 다운로드 횟수
  - 구글, 야후, 마이크로소프트, 인텔, IBM, 소니, 혼다, 도요다와 같은 대기업부터 Applied Minds, Videosurf 및 Zeitera와 같은 신생 기업들까지 사용
  - C, C++, 파이썬(Python), Java, 매트랩 인터페이스 제공
  - 윈도우즈, 리눅스, 안드로이드, 맥 OS 등 다양한 운영체제 지원
  - MX(Multimedia Extension)와 SSE(streaming SIMD Extensions) 명령어 통해 고속의 알고리즘 구현
  - CUDA와 OpenCL 인터페이스 개발

〈표 2.1.1〉 OpneCV 버전별 특징

1.0 버전	2.0 버전	2.2 버전
• C 언어 기반 API • 구조체 기반 데이터 구조 사용 • 비주얼 스튜디오에서 라이 브러리 컴파일 후 사용 • highgui 모듈에서 8비트 PNG, JPEG2000 입출력 지원 • 샘플 예제 파일 추가 (calibrate.cpp, inpaint.cpp, leter_recog. cpp 등)	• C++ 언어 기반 API • 클래스 기반 데이터 구조 도입 • CMake를 이용하여 라이브러리 컴파일 후 사용 가능 • highgui 모듈에서 스테레오 카메라 지원 • 소스 디렉터리 구조 구성	• 템플릿 자료구조 추가 • 기존 5개 라이브러리를 12 모듈로 재구성(opencv_c opencv_imgproc, ope highgui, opencv_ml 등) • 안드로이드 지원 가능 • highgui 모듈에서 16비트 L 압축 TIFF 지원 • GPU 처리 지원
2.4 버전	**3.0 버전**	**3.4 버전**
• cv::Algorithm 클래스 도입 • SIFT와 SURF 모듈 유료화 • SIFT 성능 대폭 개선 • 컬러 영상 캐니 에지 수행	• cv::Algorithm 적극 사용 • 1500개 패치 github 제출 • OpenCL을 사용하는 투명 GPU 가속 레이어 도입 • NEON 내장 함수 사용한 OpenCV 함수 가속화 • Python & Java 바인딩 확장 및 Matlab 바인딩 도입 • Python 3.0 지원 향상 • 안드로이드 지원 향상 • 비디오 캡쳐 및 멀티스레딩 함수 개선	• dnn 모듈 개선 – fast R-CNN 지원 – Javascript 바인딩 – OpenCL 가속화 포함 • OpenCL 커널 바이너리에 디스크 캐시 및 수동 로딩 구현 • GSoC 프로젝트 통합으로 백그라운드 감산 알고리즘 구현

4.0 버전	4.1 버전	4.2 버전
• 1.x 버전 C API 대량 제거 • 효과적인 그래픽 기반 영상처리 엔진으로 G-API 모듈 추가 • OpenVION 딥러닝 툴킷으로 dnn 모듈 업데이트 • 키넥트 퓨전 알고리즘 구현 • QR코드 검출기 추가 • 효과적인 광류 알고리즘 추가	• core와 imgproc 모듈 실행 최적화 • dnn 모듈 개선 – NN Builder API로 교체 – 인텔 Neural ComputerStick2 지원 • 안드로이드 미디어 NDK API 지원 • Hand-Eye 캘리브레이션 추가	• dnn 모듈 개선 – cuda와 통합된 GSoC 프로젝트 • 성능 개선 – SIMD 지원 확대 – pryDown 멀티스레딩 지원 • FSR 알고리즘

C++기반 OpenCV 프로그래밍의 예

```
#include <opencv2\imgproc.hpp>
#include <opencv2\highgui.hpp>

using namespace cv;

void main()
{
   Mat img = imread("lena.jpg");
   Mat gray_img, edge_img;

   imshow("Input Image", img);

   cvtColor(img, gray_img, COLOR_BGR2GRAY);
   imshow("Gray Image", gray_img);

   GaussianBlur(gray_img, gray_img, Size(7, 7), 1.5, 1.5);
   Canny(gray_img, edge_img, 0, 30);
   imshow("Edge Image", edge_img);

   waitKey(0);
}
```

EmguCV

❖ EmguCV

- 크로스플랫폼 영상처리 라이브러리
  - 윈도우, Mac, Android, iPhone 등
- OpenCV의 .NET wrapper
  - Intel이 개발한 OpenCV와 같은 기능을 사용
- C#, VB, Python, VC++ 등의 언어를 지원

❖ 장점

- 크로스 플랫폼 지원
- 범용 컬러와 깊이영상 클래스 제공
- 자동 메모리 관리
- XML 직렬화 영상

OpenCV-Python

❖ OpenCV-Python

- Python언어를 기반으로 사용가능한 OpneCV
- Tutorial
  - https://opencv-python-tutroals.readthedocs.io/en/latest/index.html

❖ Python은 속도가 느리다?

- 성능이 필요한 부분은 C/C++로 구현한 후, Python Wrapper를 생성하여 해결
- 장점
  - Python코드가 C/C++에 비해 성능이 떨어지지 않게 함
  - Python으로 쉽게 코딩이 가능함

실습을 위한 환경구축

❖ 실습환경

- Python 3.8.x(64-bit)
- IDE: Jupyter Notebook, Visual Studio Code 등을 이용

❖ 설치순서

1. Python 3.8.x 설치
2. Command Prompt에서... 라이브러리 설치
   - numpy, matplotlib, seaborn,
   - scikit-learn, Pandas,
   - opencv-python 등
3. IDE 설치

파이썬 및 라이브러리 설치

- **python 3.8.x 64-bit 버전 설치**
  - python 3.9는 아직 tensorflow 지원 안됨.
- **cmd prompt에서... 다음을 차례로 입력 (참고: anaconda가능)**

```
python -m pip install --upgrade pip
pip install jupyter
pip install numpy
pip install matplotlib
pip install seaborn
pip install pandas
pip install scikit-learn
pip install opencv-python
```

통합개발환경(IDE) 설치

❖ 가능한 IDE

- Python Shell(Python IDLE)
  - python과 동시에 설치됨(반드시 PATH 확인)
- PyCharm Community
  - https://www.jetbrains.com/ko-kr/pycharm/
  - 많이 사용됨
- Visual Studio Code
  - 크로스 플랫폼을 지원하는 editor
  - Windows, macOS, Linux(Ubuntu)를 모두 지원
- **jupyter notebook(또는 colab)**
  - command prompt에서 server 실행 후, Chrome에서 해당 주소로 연결하여 사용

    > jupyter notebook
  - Code와 Markdown 입력 가능

❖ Vscode 설치

- https://code.visualstudio.com/
- 설치시에 반드시 PATH에 경로 추가

❖ Vscode 환경설정

- Extentions(Ctrl+Shift+X)에서 'python'으로 검색 후, Python Extention, Korean Language Pack, Beauty 등을 설치
- 글꼴설정
  - 필요시 D2Coding 폰트 설치
  - 설정 – 텍스트편집기 – 폰트에서 글꼴, 크기 설정

참고: setting.json

```
{
    "terminal.integrated.fontSize": 16,
    "terminal.integrated.fontFamily": "Consolas, D2Coding",
    "editor.fontSize": 18,
    "editor.fontFamily": "Consolas, D2Coding, 'Courier New', monospace",
    "editor.mouseWheelZoom": true
}
```

❖ Vscode 사용방법

- 폴더를 선택/생성
- 파일을 추가(예: test.py)

❖ Vscode 편집을 위한 유용한 단축키

- **Ctrl+C** / **Ctrl+V** / **Ctrl+X**
  - 현재라인 복사하기/붙이기/잘라내기(선택하지 않은 상태)
- **Ctrl+Shift+K**
  - 현재라인 삭제하기(선택하지 않은 상태)
- **Ctrl+Enter(아래)** / **Ctrl+Shift+Enter(위)**
  - 현재라인의 아래 / 위에 빈 라인 추가
- **Alt+ ↑** / **Alt+ ↓**
  - 현재라인을 위 / 아래로 이동
- **Ctrl+D**
  - 현재 커서의 단어와 같은 단어를 차례로 선택하여 한꺼번에 바꾸기
- **Alt+클릭** : 멀티 커서

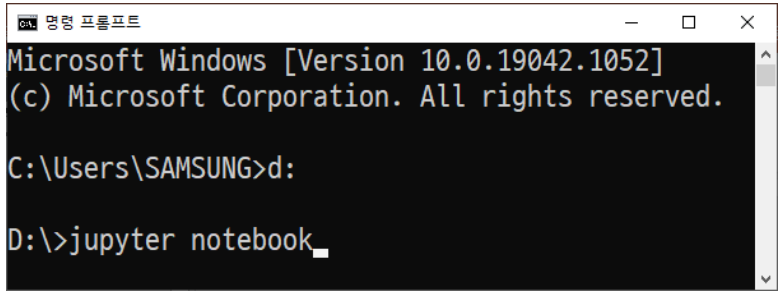
jupyter notebook 설치

❖ 준비

- jupyter notebook은 웹브라우저(IEE, Chrome 등)에서 실행됨
- 가급적 Chrome Browser 설치 권장

❖ 설치

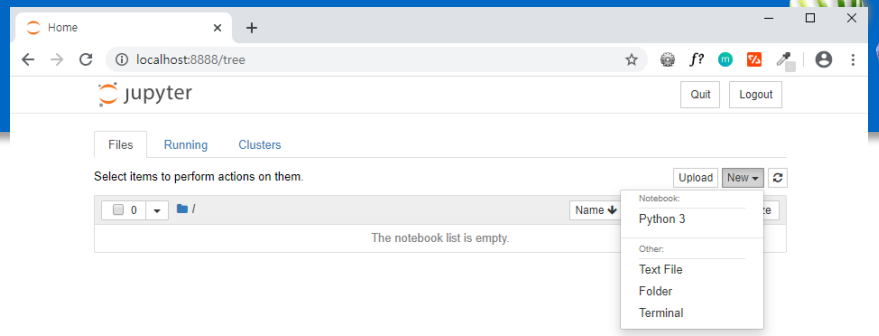
- 검색창(Win+R)에 **cmd** 입력한 후, "명령 프롬프트"에서 설치
- **pip install jupyter**



❖ 실행방법

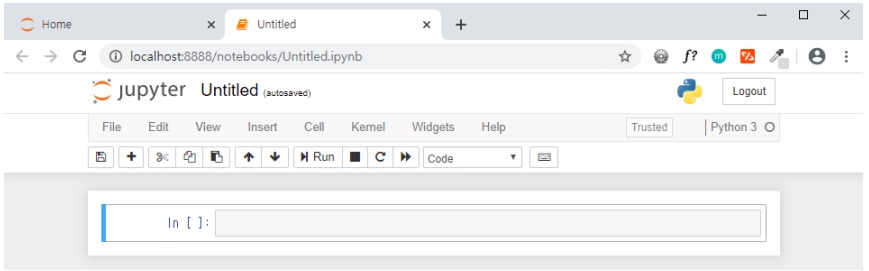
1. 명령 프롬프트 열기(cmd)
2. 노트북파일(*.ipynb)을 저장하기 원하는 디스크로 이동(필요시)
3. jupyter notebook 입력
   - 주의: 명령 프롬프트는 절대 종료시키면 안됨

❖ 사용방법 요약



■ 노트북 만들기

- 오른쪽 New 버튼을 클릭한 뒤 Python 3을 클릭
- 노트북이름을 클릭하여, 노트북의 이름을 변경



■ Cell에 Code 또는 Markdown 입력

■ 유용한 단축키

- **편집모드 : 녹색(cell 클릭 / Enter)**
  - **Ctrl+Enter**: 현재 cell 실행
  - **Shift+Enter**: 현재 cell 실행하고, 다음 cell로 이동(없으면 새로 만듦)
- **명령모드 : 청색(cell 바깥쪽 클릭 / ESC)**
  - **m**: markdown, **y**: code
  - **a**: 현재 cell 위에 추가, **b**: 현재 cell 아래에 추가
  - **dd**: 현재 cell 삭제, **z**: 현재 cell 삭제 취소

참고: Jupyter notebook extentions

❖ 설치

- pip install jupyter_contrib_nbextensions && jupyter contrib nbextension install

❖ jupyter notebook에 Nbextensions 탭 생성됨

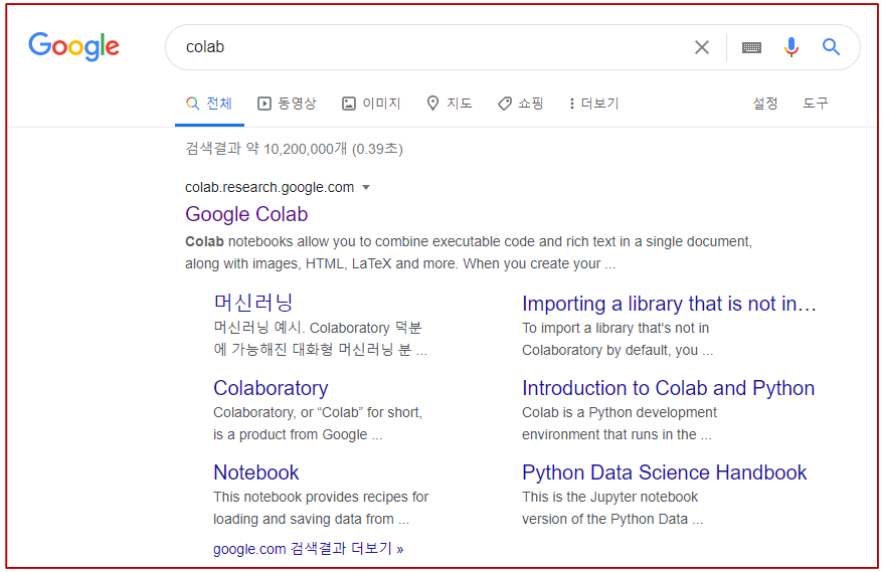
❖ 유용한 확장기능

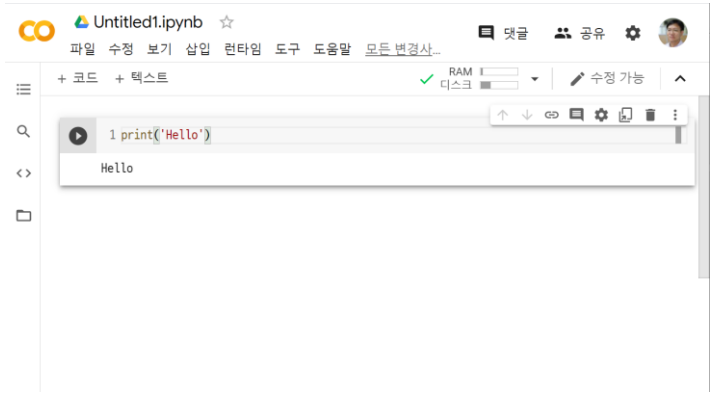
- ExecuteTime
- Hinterland
- Variable Inspector

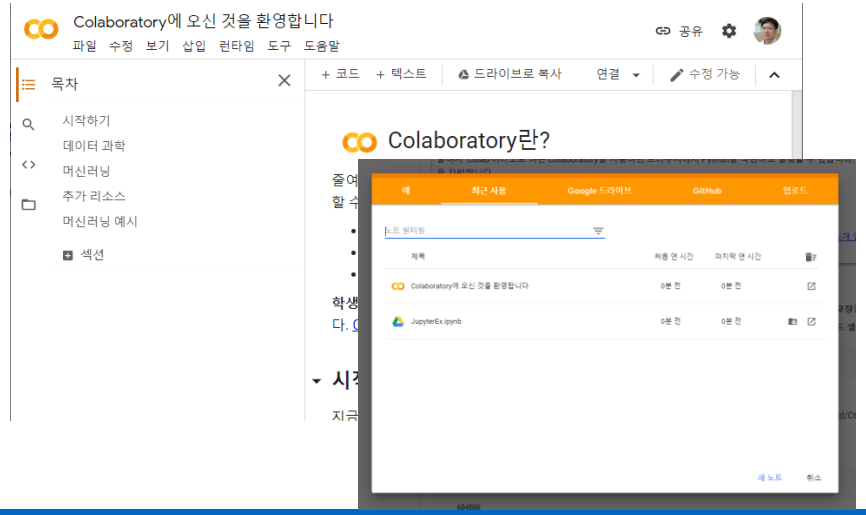
참고: Google Colab

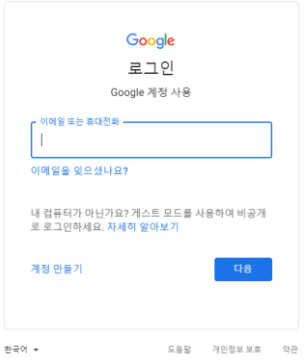
❖ colab 사용하기

- colab 사이트에 접속
  - https://colab.research.google.com/
- google 로그인
- 새노트
  - 새노트 만들기
  - coding 후, 실행(Ctrl+Enter)









Colab 사용시 꼭 알아야 할 것들

❖ 경고안내문 무시하기

```
import warnings

warnings.filterwarnings("ignore")
```

❖ 한글폰트 설치하기

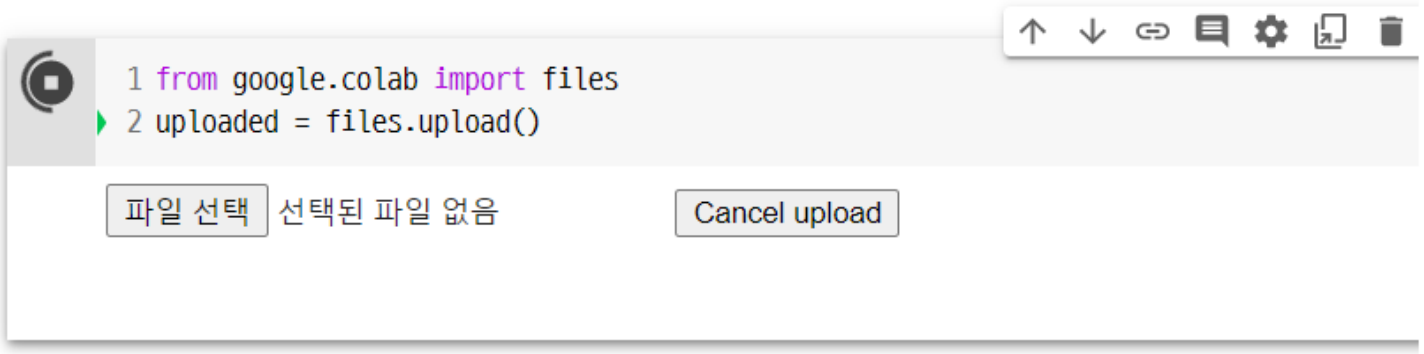
```
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf
```

❖ matplotlib에서 한글폰트 사용하기

```
import matplotlib.pyplot as plt

plt.rc('font', family='NanumBarunGothic')
```

❖ Google drive로 로컬파일 업로드하기

```
from google.colab import files
uploaded = files.upload()
```

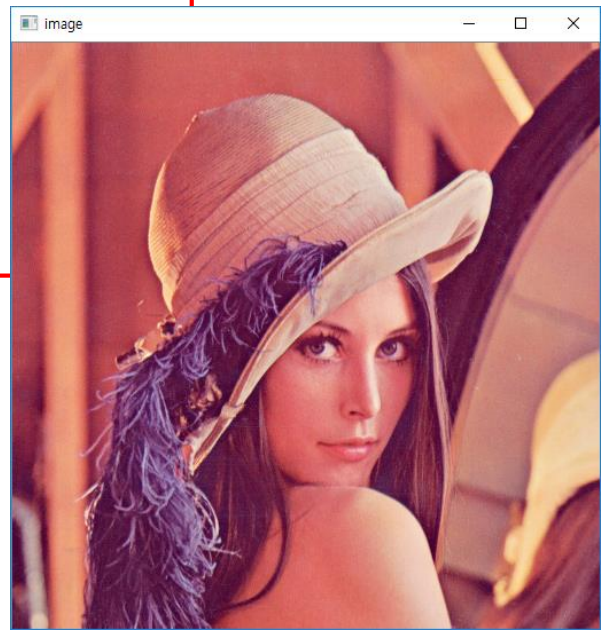


❖ Google drive로 부터 csv파일 불러오기

```
import io
import pandas as pd

data = pd.read_csv(io.BytesIO(uploaded['tips.csv']))
data
```

이미지 읽고 출력하기

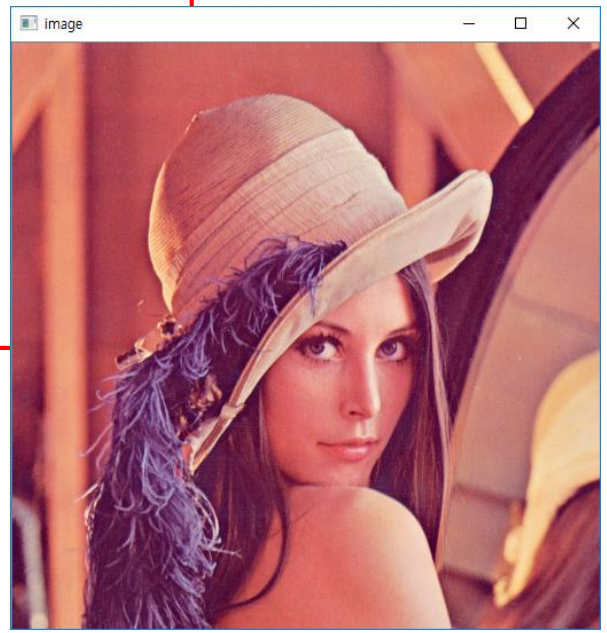
```
import cv2
import numpy as np

def showImage():
    filename = "lena.jpg"
    img = cv2.imread(filename, cv2.IMREAD_COLOR)

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

showImage()
```



- cv2.IMREAD_COLOR
- cv2.IMREAD_GRAYSCALE
- cv2.IMREAD_UNCHANGED

크기조절이 가능한 윈도우창 만들기

```
import cv2
import numpy as np

def showImage():
    filename = "lena.jpg"
    img = cv2.imread(filename, cv2.IMREAD_COLOR)

    cv2.namedWindow('image', cv2.WINDOW_NORMAL)
    cv2.imshow('image', img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

showImage()
```



- cv2.WINDOW_AUTOSIZE
- cv2.WINDOW_NORMAL

imread

❖ cv2.imread()함수

- 이미지를 읽은 후, numpy의 ndarray 형식으로 리턴

❖ 각 픽셀들의 값은 [B, G, R]의 값

❖ 모드

- cv2.IMREAD_COLOR
  - 컬러 이미지로 읽음
- cv2.IMREAD_GRAYSCALE
  - 그레이 이미지로 읽음
- cv2.IMREAD_UNCHANGED
  - 알파채널을 포함하여 그대로 읽음

비디오 캡쳐

❖ 종료 : ESC

```
import cv2
import numpy as np

def showVideo():
    try:
        cap = cv2.VideoCapture(0)
    except:
        return

    while True:
        ret, frame = cap.read()
        if not ret:
            break
        cv2.imshow('Video', frame)

        key = cv2.waitKey(1) & 0xFF
        if key == 27:
            break
    cap.release()
    cv2.destroyAllWindows()

showVideo()
```

기존의 Video를 캡쳐하려면 여기에 filename넣어준다.

이미지 픽셀에 접근하기

❖ 픽셀의 값에 직접접근

- img[340, 200] = [100, 150, 200]
  - 340, 200위치의 픽셀을 B=100, G=150, R=200의 값으로 변경
- 상대적으로 느림

❖ numpy의 item(), itemset()함수로 접근

- 최적화되어 있어 빠르다
- B, G, R 각각의 값에 개별적으로 접근해야 함
- (예)
  - `b = img.item(340, 200, 0)`
  - `g = img.item(340, 200, 1)`
  - `r = img.item(340, 200, 2)`

❖ itemset()함수로 픽셀값 변경하기

- B, G, R 각각의 변경해야 함
- (예)
  - `img.itemset((340, 200, 0), 100) # B = 100`
  - `img.itemset((340, 200, 1), 150) # G = 150`
  - `img.itemset((340, 200, 2), 200) # R = 200`

이미지의 속성 얻기

❖ 주요 이미지 속성

- img.shape
  - 이미지의 (높이, 너비, 채널수)
- img.size
  - 이미지의 크기(바이트)
- img.dtype
  - 이미지 픽셀의 데이터타입

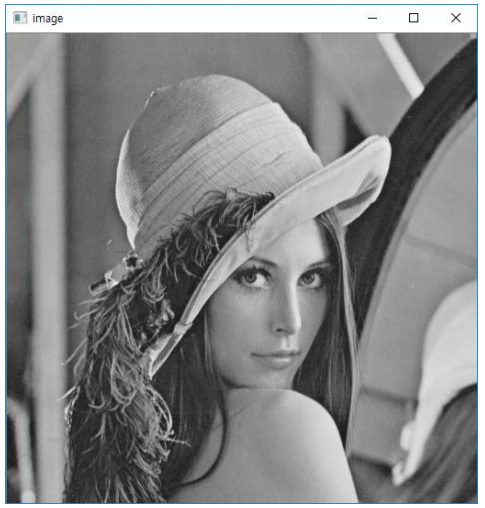
❖ (예)

```
print(img.shape)
print(img.size)
print(img.dtype)
```

```
(512, 512, 3)
786432
uint8
```

이치화(픽셀처리)

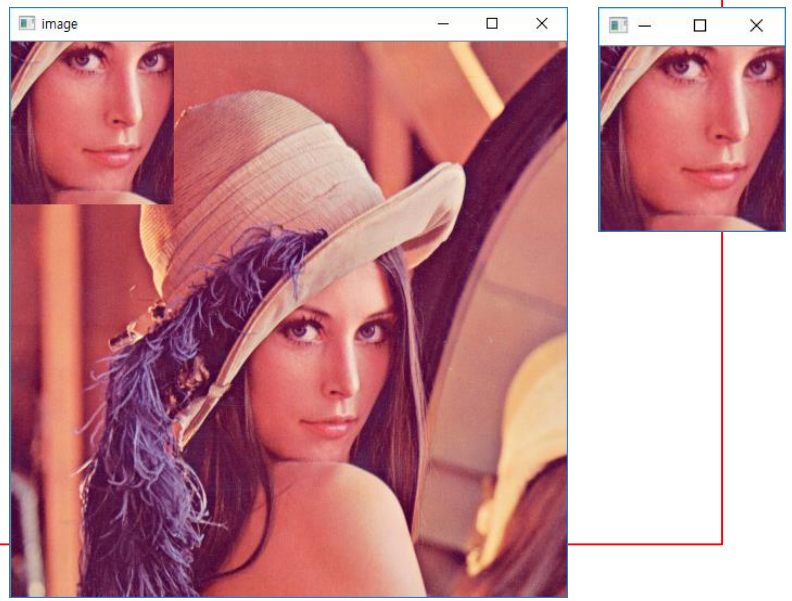
```
import cv2
import numpy as np

def showImage():
    filename = "lena.jpg"
    img = cv2.imread(filename, cv2.IMREAD_GRAYSCALE)
    cv2.imshow('image', img)

    ysize = img.shape[0]
    xsize = img.shape[1]
    for y in range(ysize):
        for x in range(xsize):
            if img.item(y, x) < 128:
                img.itemset((y, x), 0)
            else:
                img.itemset((y, x), 255)
    cv2.imshow('bin-image', img)

    cv2.waitKey(0)
    cv2.destroyAllWindows()

showImage()
```



이미지에 ROI 설정하기

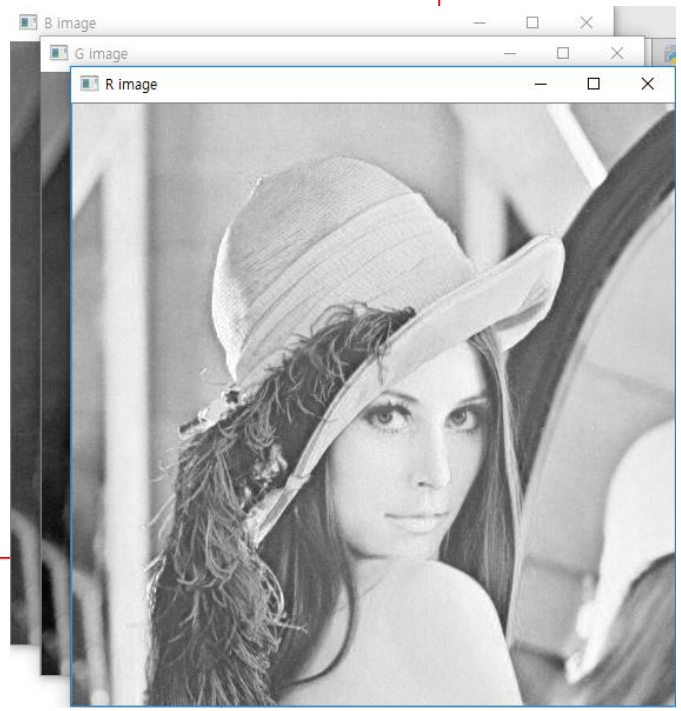
❖ ROI(Region Of Image)는 numpy의 indexing과 slicing을 이용

```
import cv2
import numpy as np

def showImage():
    filename = "lena.jpg"
    img = cv2.imread(filename, cv2.IMREAD_COLOR)

    subimg = img[250:400, 200:350]
    cv2.imshow('subimage', subimg)

    img[0:150, 0:150] = subimg
    cv2.imshow('image', img)

    cv2.waitKey(0)
    cv2.destroyAllWindows()

showImage()
```



채널 분리와 합성

```
import cv2
import numpy as np

def showImage():
    filename = "lena.jpg"
    img = cv2.imread(filename, cv2.IMREAD_COLOR)

    b, g, r = cv2.split(img)
    cv2.imshow('B image', b)
    cv2.imshow('G image', g)
    cv2.imshow('R image', r)

    cv2.waitKey(0)
    cv2.destroyAllWindows()

showImage()
```



❖ numpy를 이용한 채널분리

- 매우 효율적임
  - `b = img[:, :, 0]`
  - `g = img[:, :, 1]`
  - `r = img[:, :, 2]`

❖ 채널의 합성

- cv2.split()함수와 반대로 cv2.merge()함수 사용
- (예)
  - `mergedimg = cv2.merge((b, g, r))`

Color의 개요

❖ 컬러 표현

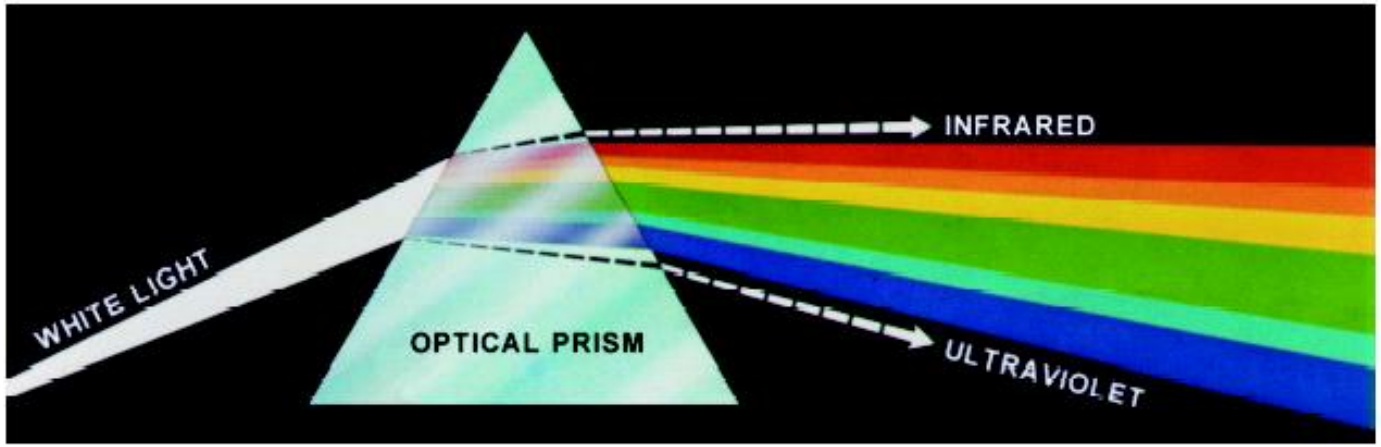
- 1802년, Tomas Young이 제안한 3색 이론
- 컬러는 세가지 기본 컬러를 적당한 비율로 조합하여 만들어진다고 주장(사람의 색인식구조와 일치)

$$C = aC_1 + bC_2 + cC_3$$

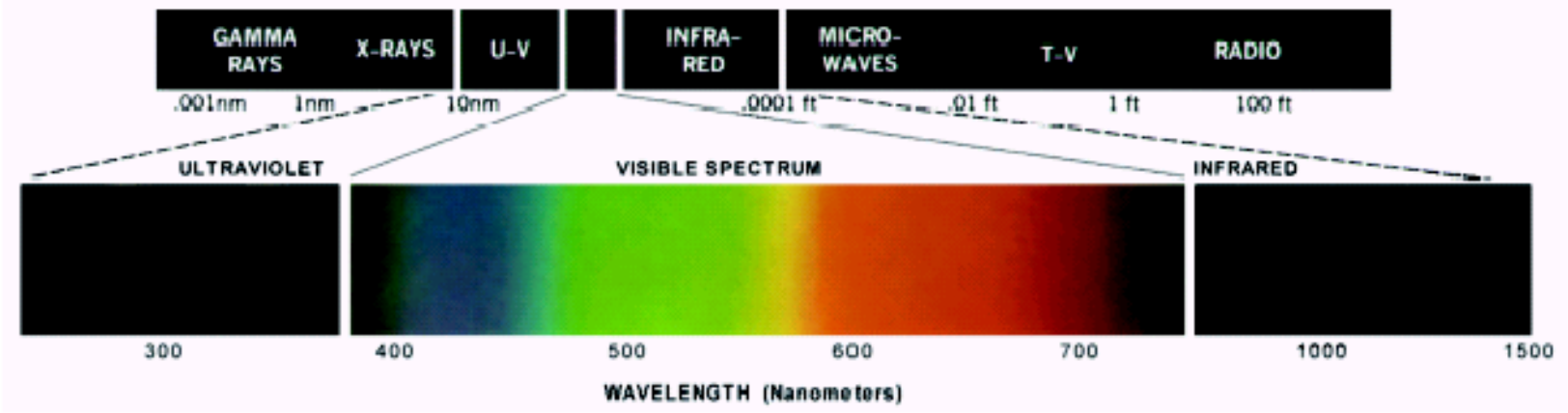
❖ 삼중자극(tristimulus)

- 인간의 컬러 인지 능력은 세가지 추상체들의 반응에 의해 나타남
  - 휴먼 망막(retina) 내부: 색을 감지하는 추상체(cone)이 3개 존재
  - 추상체: 빛을 감지하는 3개의 센서로 각각 적(red),녹(green),청(blue)영역 감지

❖ 컬러 스펙트럼(spectrum)



❖ 가시광선 영역



컬러모델

❖ 컬러모델

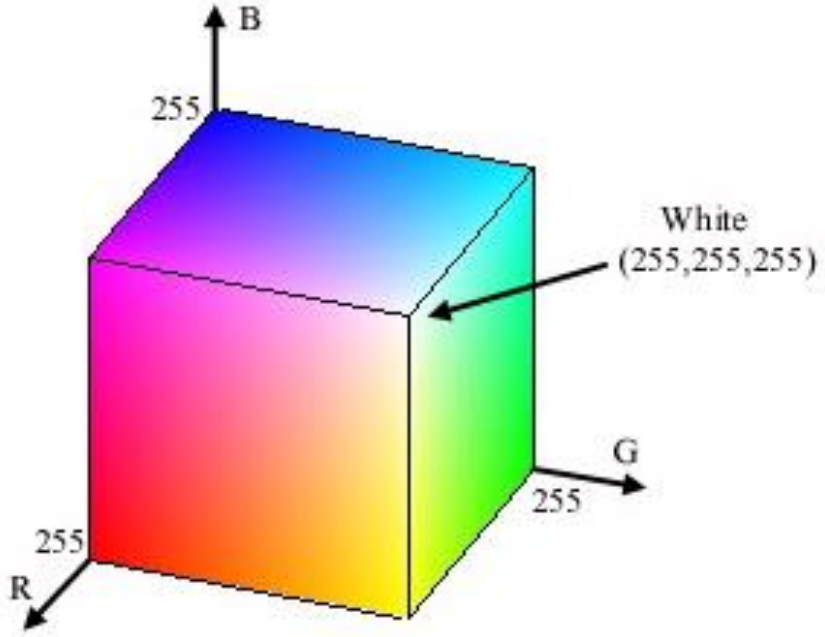
- 색상을 표현하는 체계

❖ 종류

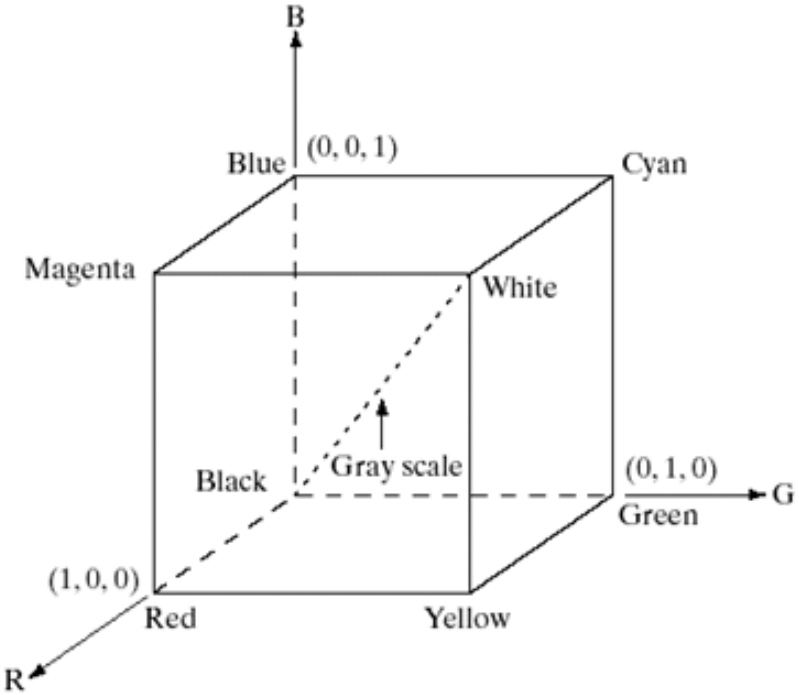
- OpenCV에서는 150여개의 컬러모델을 지원
- 주요 컬러모델
  - RGB
  - CMY
  - HSI
  - YIQ

RGB모델

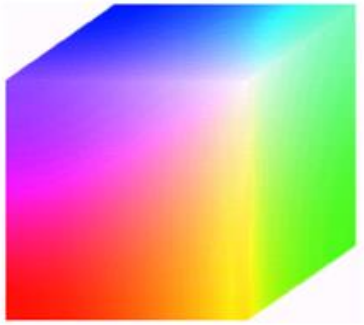
- 개인용 PC의 컬러 CRT 모니터, 컴퓨터 그래픽스
- “빛”의 삼원색
- 각각을 더해 원하는 컬러를 만들어 냄
- 더하기 삼원색(additive primaries)라고 부름



- 각각의 좌표축은 R,G,B축을 나타냄
- 좌표 (0,0,0): 검은색
- 좌표 (1,0,0): 빨강
- (0,0,0)-(1,1,1) 연결 대각선: R,G,B 비율이 동일한 회색(gray) 등급



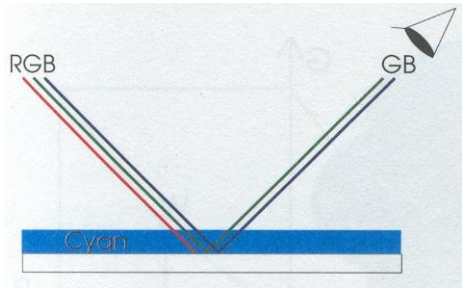
명암도 = 0.299R + 0.587G + 0.114B
명암도 = 0.333R + 0.333G + 0.333B



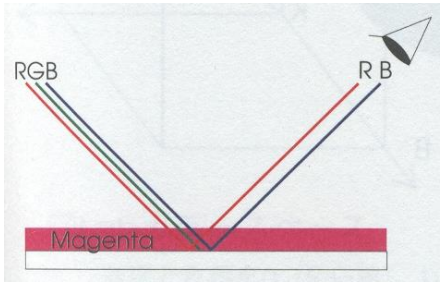
CMY모델

- ‘색’의 삼원색이며, 청록(Cyan), 자홍(Magenta), 노랑(Yellow)로 구성
- RGB모형과 반대의 공간, C,M,Y는 각기 R,G,B의 보색(complement)
- 빼기 삼원색(subtractive primaries): 백색광에서 특정색을 뺌에 의해 원하는 색깔을 만듬
- 컬러 복사기, 프린트와 같은 출력장치에 사용

C = 1-R M = 1-G Y = 1-B



(R흡수 = Cyan)



(G흡수=Magenta)

(B흡수=Yellow)

❖ CMYK 모델

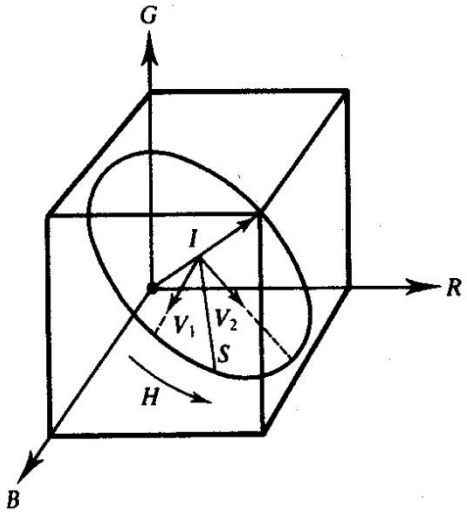
- CMY 모델 + 검정색(black) K = CMYK
- 검정색을 만들기 위해 CMY를 조합하는 것은 문제
- 비용 증가, 검정색의 질적 수준이 떨어지는 것을 방지
- 실용적 프린트 모델

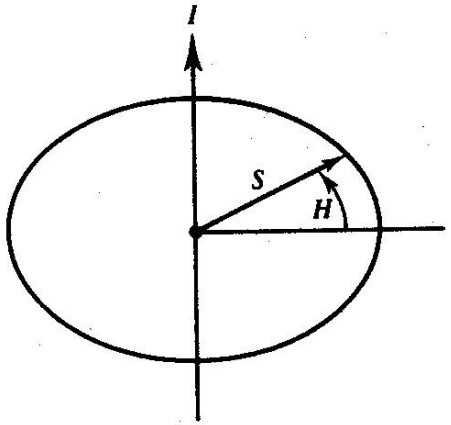
$$C = 1-R$$
$$M = 1-G$$
$$Y = 1-B$$
$$K = \min(C,M,Y)$$

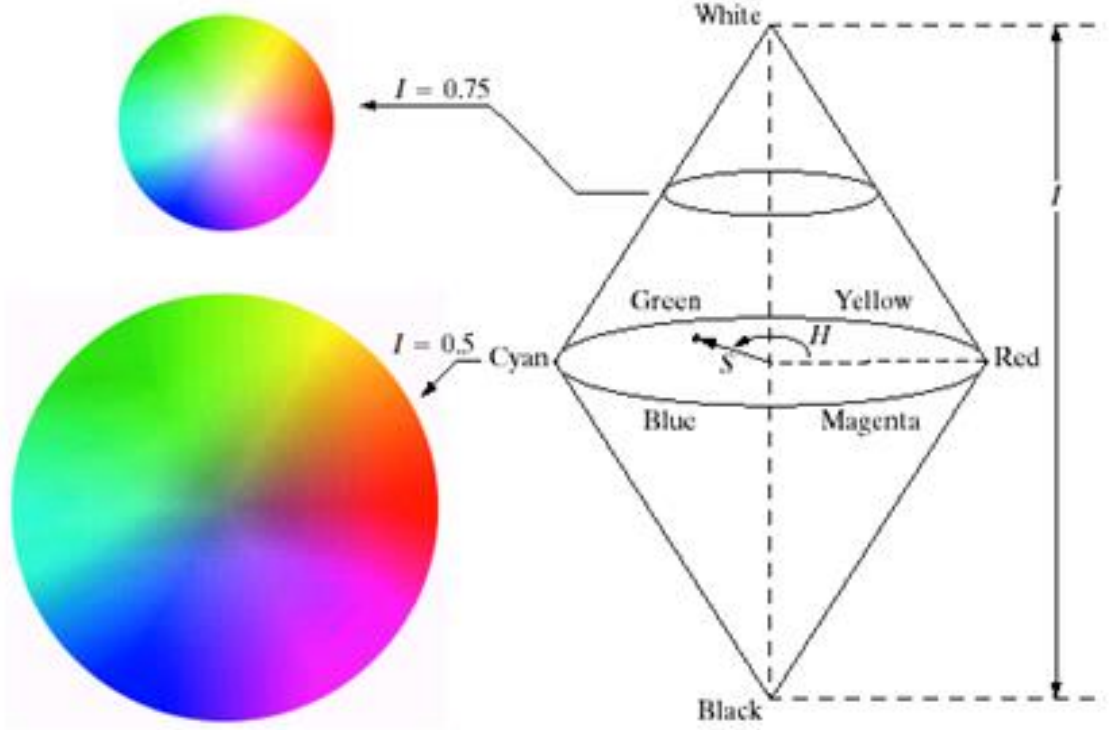
HSI모델

- 인간의 색인지에 기반한 모델
- (RGB,YIQ,CMY,CMYK는 시스템이나 하드웨어에서 사용을 위한 모델)
- 색상(Hue: H), 채도(Saturation: S), 명도(Intensity: I) 모델
- 구체적 컬러를 만들기 위해 색 조합이 불필요(H 좌표값 자체가 바로 색상값)

  - H(색상) : 0~360° 범위, 빨강,파랑, 노랑 등의 색을 부별하는 축
  - S(채도) : 0~1 범위, 순색에 첨가된 백색광의 비율
  - I(명도) : 0~1의 범위,0은 검정, 1은 흰색













원본 hue saturation intensity

YIQ모델

- TV방송국에서 사용하는 모델
- 방송국에서는 가정의 TV가 흑백,컬러 인지 무관하게 YIQ신호 발송
- 흑백TV는 Y신호만 취함, 컬러TV는 YIQ모두 취함
- Y: 명암도(Luminance), I,Q: 색신호로 색상(hue)과 채도(saturation)

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.273 & -0.647 \\ 1.000 & -1.104 & 1.701 \end{pmatrix} \begin{pmatrix} Y \\ I \\ Q \end{pmatrix}$$

- 사람 눈은 컬러값 보다 밝기값에 더 민감

- 신호 전송 시, Y값은 덜 압축 I,Q값은 많이 압축하여 전송 가능

컬러모델 바꾸기

❖ OpenCV

- cv2.cvtColor()함수를 이용

❖ cv2.cvtColor()함수의 사용 예

- hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

- 주요모드
  - **cv2.COLOR_BGR2GRAY** **cv2.COLOR_GRAY2BGR**
  - **cv2.COLOR_BGR2HSV** **cv2.COLOR_HSV2BGR**
  - **cv2.COLOR_BGR2LAB** **cv2.COLOR_LAB2BGR**
  - cv2.COLOR_BGR2RGB
  - cv2.COLOR_GRAY2RGB
  - cv2.COLOR_HSV2RGB