컴퓨터 비전과 딥러닝
챕터 1, 2 정리

AI 학과 2444337 손주안

Chapter 1

1.1 인간의 시각

1.2 왜 컴퓨터 비전인가?

1.3 컴퓨터 비전은 왜 어려운가?

1.4 컴퓨터 비전의 역사

1.5 컴퓨터 비전 체험 서비스

1.6 컴퓨터 비전 만들기

Chapter 2

2.1 OpenCV 소개

2.2 프로그래밍 킥오프

2.3 객체지향 잘 활용하기

2.4 영상을 읽고 표시하기

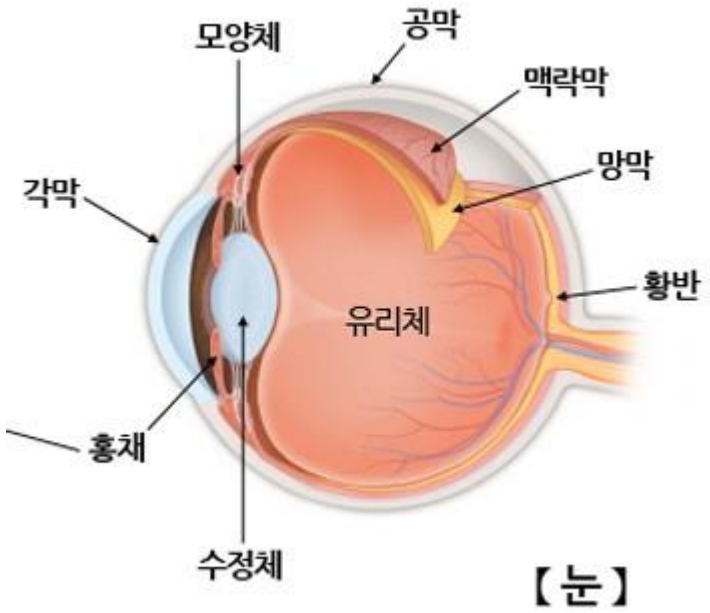
2.5 영상 형태 변환하고 크기 축소하기

2.6 웹 캠에서 비디오 읽기

2.7 그래픽 기능과 사용자 인터페이스 만들기

2.8 페인팅 기능 만들기

인간 시각 | 오감 중 가장 뛰어난 감각

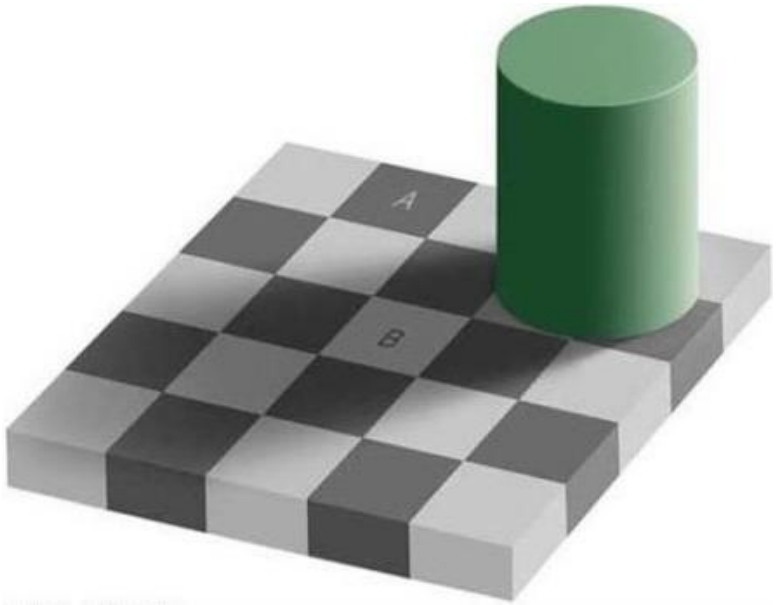


【눈】

- 물체에서 반사된 빛은 수정체를 통해 눈 내부로 들어와 망막에서 이를 화학 신호로 변경한 뒤 시신경의 1차 시각 피질을 통해 이를 동쪽 경로와 배쪽 경로로 전달하는 역할을 한다.

- **분석에 능숙**하며, 2차원 영상을 해석해 **3차원으로 복원**하는 능력이 있고, 과업에 있어 **전환이 매끄럽고 유기적**이라 빠르다는 장점이 있다.

인간 시각 | 오감 중 가장 뛰어난 감각



- 하지만 **착시**가 있고 **정밀 측정이 불가능**하여 물체 길이를 mm단위까지 정확히 알아낼 수 없다는 것, **시야가 한정**되었다는 점 등에서 단점이 있다.

왜 컴퓨터 비전인가? | 인간의 시각을 흉내 내는 컴퓨터 프로그램

- 컴퓨터 비전은 **특정 과업**에선 인간 시각 이상의 능력을 보이나 **일반 환경**에서는 인간 시각에 크게 뒤진다.
- 현재 컴퓨터 비전 기술로 인간에 필적하는 시각을 구현하는 일은 불가능하나, **과업의 한정**으로 인간 이상의 성능의 컴퓨터 비전을 만들어 활용 가능한 분야는 많다.
- 대표적인 응용 분야로 **농업**, **의료**, **교통**, **스마트 공장**, **스포츠** 등이 있다.

컴퓨터 비전은 왜 어려운가?

세상의 변화무쌍함, 넘버 크런처, 미숙한 인공지능

세상의 변화무쌍함

- 영상은 **빛**, 특정 물체를 보는 **방향과 위치** 등에 따라 달라지며 스펙트럼의 모든 수준에서 영상이 발생하므로 컴퓨터 비전이 취급해야 하는 영상의 종류가 방대하다.

- 이런 변화를 표현하는 규칙이나 알고리즘을 만드는 일은 매우 어렵다.

컴퓨터 비전은 왜 어려운가?

세상의 변화무쌍함, 넘버 크런처, 미숙한 인공지능

넘버 크런처

- 넘버 크런처는 엄청난 양의 단순 계산을 반복하는 사람 또는 기계를 말한다.

- 컴퓨터는 영상을 숫자 배열로 나타내어 이를 사칙과 비교 연산을 통해 **중급 특징**을 알아내고, 중급 특징을 결합해 **고급 특징**을 알아내는데 이는 단순하지 않다.

컴퓨터 비전은 왜 어려운가?

세상의 변화무쌍함, 넘버 크런처, 미숙한 인공지능

미숙한 인공지능

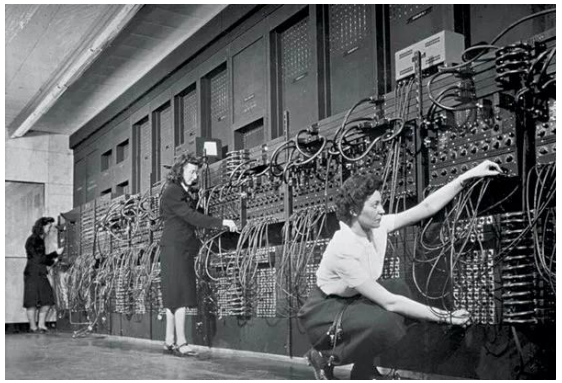
- 70여 년의 역사를 거치며 인공지능은 **학습**이라는 지능 요소에서 상대적으로 크게 발전했지만 **지식 표현**, **추론**, **계획의 연구 성과**는 지지부진하다.

- 현재 인공지능은 약한 인공지능 단계에 머물러 있고, 이런 특성은 컴퓨터 비전의 미숙함으로 전이되었다.

컴퓨터 비전의 역사 | 신문 산업에서 탄생한 디지털 영상

| 1920년

해저 케이블을 통해 사진을 전송하는 **Bartlane** 시스템이 구축되었다.



| 1946년

세계 최초의 전자식 컴퓨터, 에니악이 탄생했다.

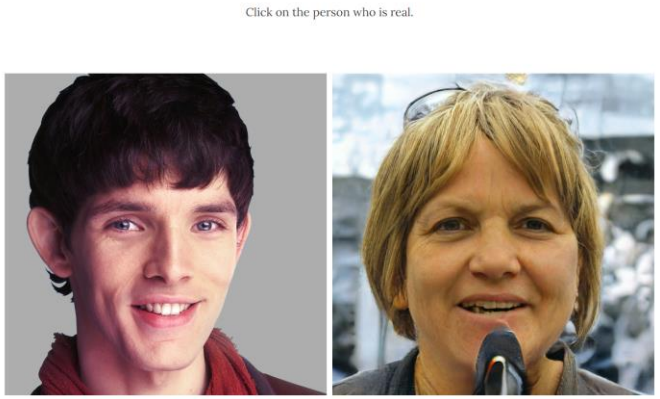
| 1957년

러쉘 커쉬가 스캐너를 개발하여 세계 최초로 디지털 영상을 컴퓨터에 저장했다.

컴퓨터 비전 체험 서비스 | 현재 기술 수준 확인하기

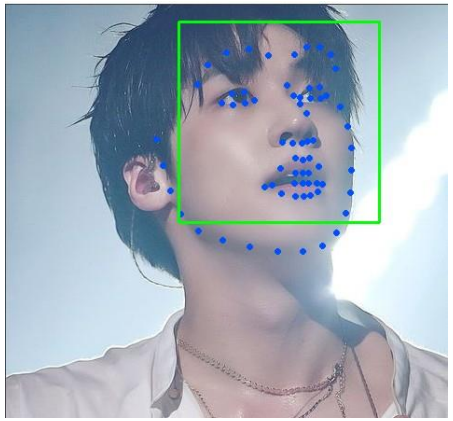
| 구글 렌즈

물체나 텍스트를 촬영하면
인식하여 관련 내용을 보여 준다.



| Which face Is real?

가짜 얼굴을 생성하여 진짜 얼굴과
함께 배치해 진짜를 골라내도록
하는 서비스다.



| 얼굴 랜드마크 검출

웹캠을 통해 얼굴을 보여 주면
랜드마크를 검출해 3차원 깊이
정보를 추정해 보여 준다.

컴퓨터 비전 만들기

OpenCV

- 컴퓨터 비전 전문 라이브러리

텐서플로

- 딥러닝 라이브러리

파이썬

- 풍부한 배열 연산을 제공
- x, y, z가 길이가 n인 1차원 배열일 때, x, y를 요소별로 더해 z에 저장하기 위해 C 언어에서는 반복문을 사용해 각 요소를 더하는 방식으로 코딩하지만 파이썬에서는 x와 y를 그냥 더해서 z에 저장하면 되므로 편리하다.

C 언어

```
for(i=0; i<n; i++)   z[i] = x[i] + y[i];
```

파이썬

```
z = x + y
```

OpenCV 소개

- 인텔이 개발했으며, 구성 함수와 클래스는 C와 C++ 언어로 개발되었다.
- 전체 코드는 180만 라인 이상이다.
- 함수 호출 시에 사용되는 인터페이스 언어는 C, C++, 자바, 자바스크립트, 파이썬이 있다.

프로그래밍 킥오프 | 파이썬을 쓰기 위해 설치할 3가지 소프트웨어

파이썬 컴파일러

- 파이썬으로 작성된 소스 프로그램을 컴퓨터가 실행할 수 기계어 프로그램으로 번역하는 프로그램이다.

통합 개발 환경

- 프로그래밍할 때 프로그램 편집, 관리, 번역, 디버깅 등의 작업을 통합적으로 지원하는 프로그램이다.

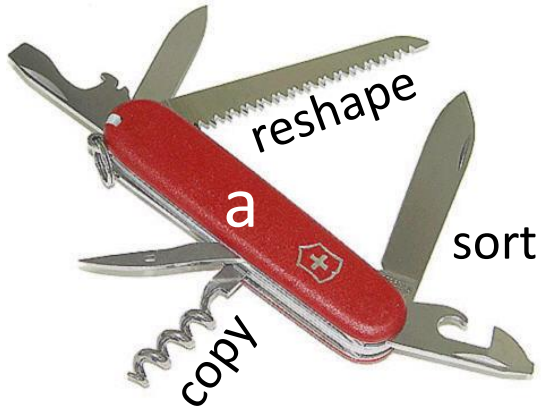
라이브러리

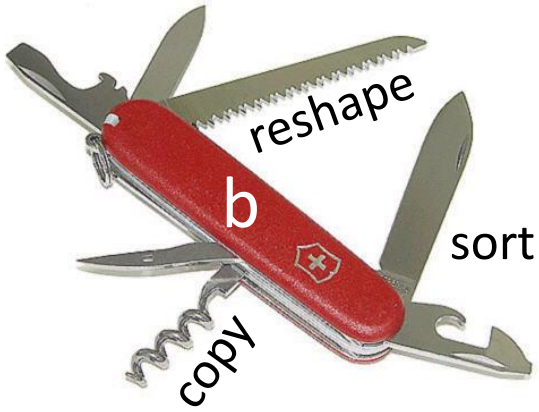
- 특정한 일을 지원하는 변수, 함수, 클래스의 뭉치다.

아나콘다

- 필요한 소프트웨어 대부분을 한꺼번에 설치해 주며 가상 환경을 제공한다.

객체 지향 잘 활용하기 | 객체 다루기





- 객체는 ‘ . ’을 찍어 자신이 가진 함수를 능동적으로 호출한다.
- 객체 내부에 있는 함수를 멤버 함수 또는 메서드라고 한다. 객체 내부에는 데이터를 저장할 멤버 변수도 있다.
- 객체는 무한정 생성할 수 있다.np.ndarray 클래스로 만든 객체를 스위스 칼에 비유할 수 있다.
- 객체 여러 개가 서로 다르지만 멤버 함수는 같다.

객체 지향 잘 활용하기 | 객체 확인하기

type, dir

- 객체가 어떤 클래스인지, 사용 가능한 멤버 함수 목록을 알기 위해 사용하는 명령어

```
type(a)
dir(a)
```

help

- 멤버 함수가 어떤 일을 하며 어떻게 사용하는지 알기 위해 사용하는 함수다.

```
help(a.sort)
```

영상을 읽고 표시하기

```
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('파일이 없습니다.')

cv2.imshow('image display', img)

cv2.waitKey()
cv2.destroyAllWindows()
```

imread

- CV2 모듈이 제공하는 함수로, 소스 파일이 저장되어 있는 폴더 내의 이미지 파일을 인수로 제공한다.

imshow

- 영상을 윈도우에 디스플레이 한다. 첫 번째 인수는 윈도우의 이름, 두 번째 인수는 디스플레이할 영상이다.

waitKey

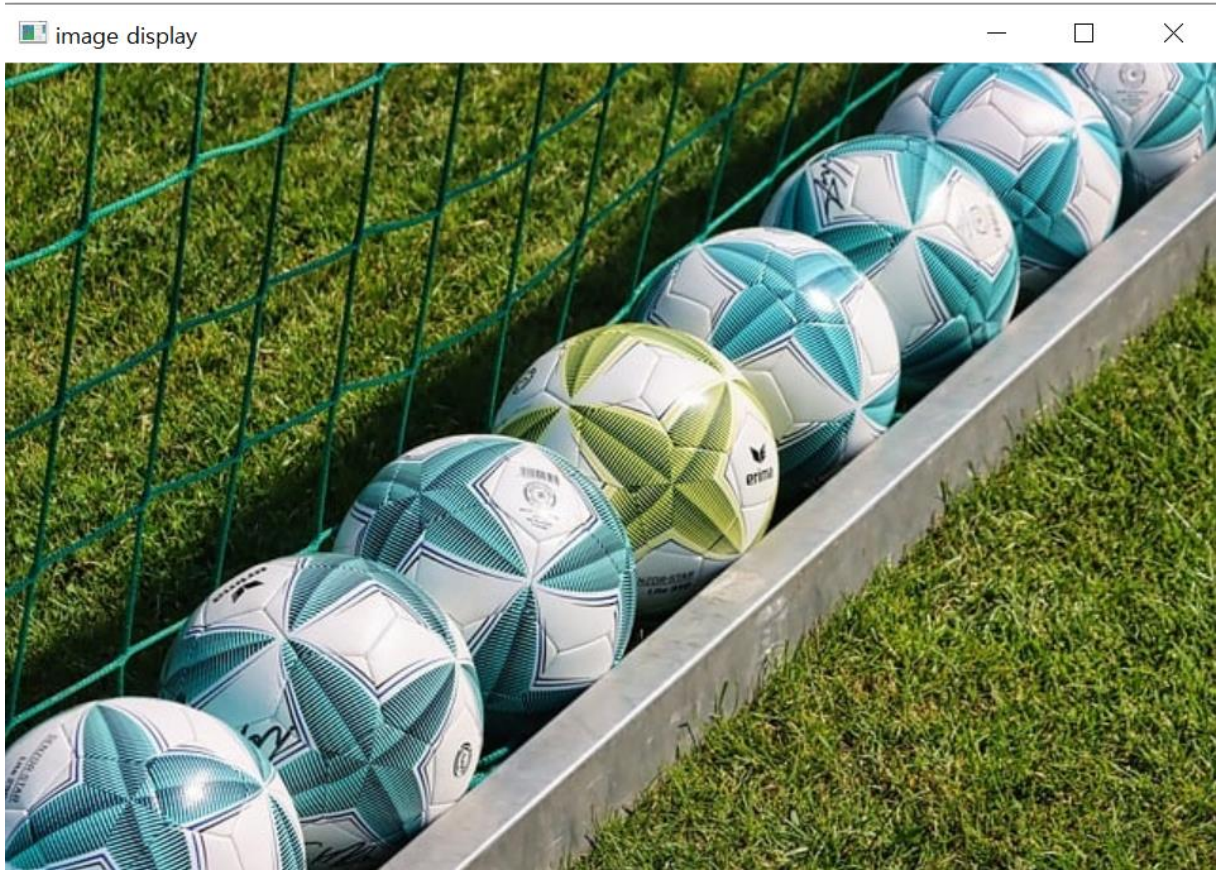
- 키보드의 키가 눌릴 때까지 기다리다가 눌리면 해당 키의 유니코드 값을 반환한다. 시간을 지정하려면 인수로 밀리초 단위로 정수를 주면 되며, 0으로 설정하면 무한정 기다린다.

destroyAllWindows

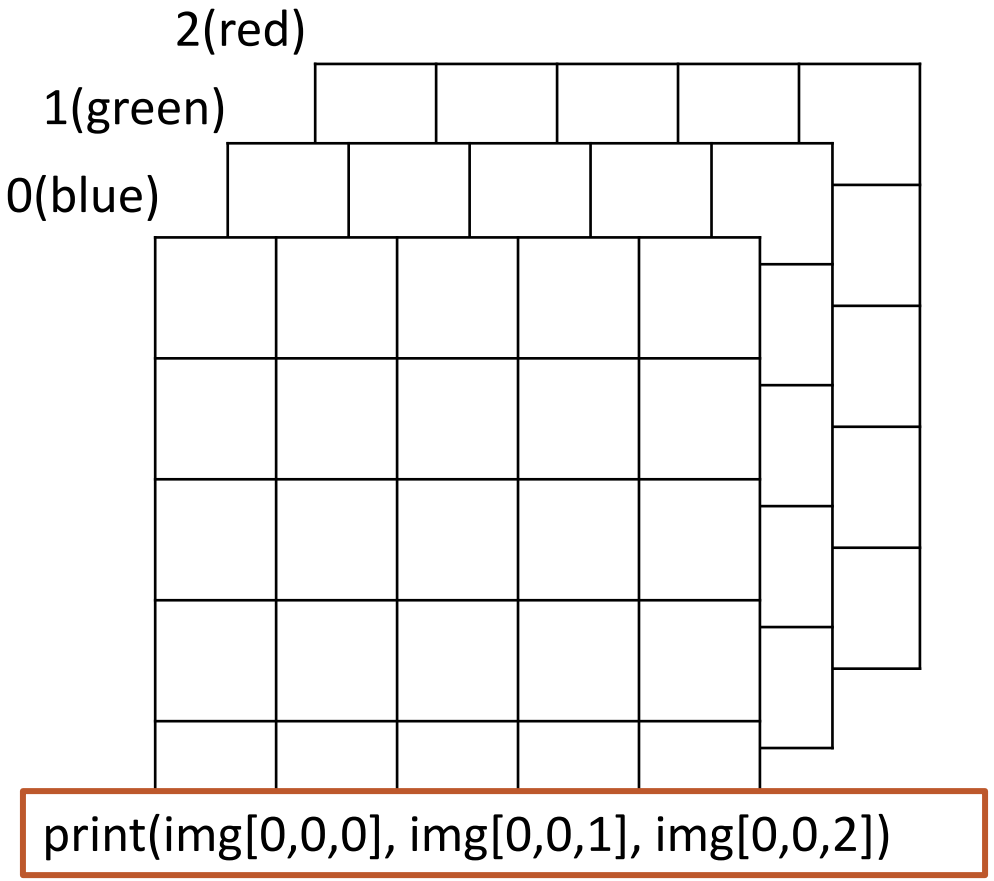
- 윈도우를 닫는 역할을 한다.

영상을 읽고 표시하기 | 실행 결과

```python
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('파일이 없습니다.')
cv2.imshow('image display', img)

cv2.waitKey()
cv2.destroyAllWindows()
```



영상을 읽고 표시하기



화소

- 영상을 구성하는 한 점
- 컴퓨터 비전에서는 왼쪽 위를 원점으로 하여 행 좌표로 y축, 열 좌표로 x축을, 즉, (y,x)의 형태로 사용한다.
- 컬러 영상은 한 화소가 blue, green, red 3개 값을 가지며, OpenCV에서는 RGB가 아닌, BGR의 순서를 가진다.
- 다음 명령어를 통해 (0,0)에서의 화솟값을 알 수 있다.

영상 형태 변환하고 크기 축소하기

```
img = cv2.imread('soccer.jpg')

gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
gray_small = cv2.resize(gray, dsize = (0,0), fx = 0.5, fy = 0.5)
```

- cvtColor 함수를 통해 첫 번째 인수로 쓰인 객체를 두 번째 인수를 통해 명암 영상으로 변환한다.
- resize 함수를 사용할 경우 영상의 크기를 변환한다. 첫 번째 인수는 입력할 영상이며, 두 번째 인수의 경우 변환할 크기를 지정하는데, (0,0)이므로 비율을 지정하는 fx와 fy에 따라 결정한다.

영상 형태 변환하고 크기 축소하기

```
cv2.imwrite('soccer_gray.jpg', gray)

cv2.imwrite('soccer_gray_small.jpg', gray_small)
```

$$I = round(0.299 * R + 0.587 * G + 0.114 * B)$$

- imwrite 함수는 지정한 영상을 지정한 이름으로 지정한 파일에 저장한다. 두 번째 인수로 쓰인 객체를 첫 번째 인수에 저장하며, 파일 확장자를 jpg가 아닌 png로 저장하는 것도 가능하다.
- 영상을 명암 영상으로 변환하는 건 다음 공식으로 제시된다.
- BGR의 값을 식에 대입하여 round를 통해 반올림 한 후, I를 구한다. 이런 연산을 모든 화소에 적용한다.

영상 형태 변환하고 크기 축소하기 | 실행 결과

```python
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('파일이 없습니다.')

gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
gray_small = cv2.resize(gray, dsize = (0,0), fx = 0.5, fy = 0.5)

cv2.imwrite('gray.jpg', gray)
cv2.imwrite('gray_small.jpg', gray_small)
```

|gray.jpg

|gray_small.jpg

웹 캠에서 비디오 읽기

```
cv2.imwrite('soccer_gray.jpg', gray)

Cap = cv2.VideoCapture(0,
cv2.CAP_DSHOW)

While True:
    ret, frame = cap.read()
    if not ret:
        break
    cv2.imshow('video display', frame)
    key = cv2.waitKey(1)
    if key == ord('q')
        break
Cap.release()
Cv2.destroyAllWindows()
```

- VideoCapture 함수는 웹 캠과의 연결을 시도하고 그 결과를 cap 객체에 저장한다. 첫 번째 인수는 웹캠의 번호로, 웹 캠이 **하나인 경우**에 **0**을 입력한다. 두 번째 인수의 경우 비디오가 화면에 바로 나타나게 한다.
- Read 함수는 호출한 순간의 프레임을 획득하여 성공 여부와 프레임을 반환한다. 각각 ret와 frame에 저장한다.
- imshow 함수를 통해 윈도우에 영상을 디스플레이한다.
- 키보드에서 q가 입력될 경우 루프를 탈출하도록 되어 영상이 끝나게 된다.

웹 캠에서 비디오 읽기 | 비디오에서 영상 수집하기

```
import numpy as np
frames = []
```

```
if key == ord('c')
    frames.append(frames)
```

```
if len(frames) >0:
    imgs = frames[0]
    for i in range(1, min(3, len(frames)))
        imgs = np.hstack((imgs, frames[i]))
    cv2.imshow('colledted images', imgs)
```

- 수집한 영상을 이어 붙이기 위해서 numpy의 hstack 함수를 사용한다.
- 키보드의 c를 누를 경우 프레임을 frames에 추가한다.
- 수집한 영상이 있을 경우 min 함수를 사용하여 3개까지 이어 붙이도록 설정한다.

웹 캠에서 비디오 읽기 | 실행 결과

```
import cv2
import sys
import numpy as np

frames = []

cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)

if not cap.isOpened():
    sys.exit('실패')

while True:
    ret,frame = cap.read()
    if not ret:
        print('없음.')
        break
    cv2.imshow('video',frame)
    key = cv2.waitKey(1)
    if key == ord('q'):
        break
    if key == ord('c'):
        frames.append(frame)

cap.release()
cv2.destroyAllWindows()
```

```
if len(frames) > 0:
    imgs = frames[0]
    for i in range(1, min(3,len(frames))):
        imgs = np.hstack((imgs,frames[i]))
    cv2.imshow('img',imgs)
    cv2.waitKey()
    cv2.destroyAllWindows()
```

그래픽 기능과 사용자 인터페이스 만들기

```
Cv2.rectangle(img,(100, 100),(200, 200),(0, 0, 255), 2)

Cv2.putText(img, ‘laugh‘, (300, 300), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
```

- rectangle 함수를 이용해 직사각형을 그린다.
- 첫 번째 인수는 직사각형을 그릴 영상이고, 두, 세 번째 인수는 각각 직사각형의 왼쪽 위, 오른쪽 아래 구석점의 좌표이다. 네 번째 인수는 색이며, 다섯 번째 인수는 선의 두께다.
- putText 함수는 문자를 입력한다. 첫 번째 인수는 문자를 적을 영상이다. 두 번째 인수는 어떤 문자를 적을지, 세 번째 인수는 문자열의 왼쪽 아래 구석점을 지정한다. 네 번째는 폰트 종류, 다섯 번째 인수는 글자 크기, 여섯 번째 인수는 색, 일곱번째 인수는 글자 두께다.

그래픽 기능과 사용자 인터페이스 만들기 | 실행 결과

```
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('nothing')

cv2.rectangle(img,(100,100),(200,200),(0,0,255),2)
cv2.putText(img,'laugh',(300,300),cv2.FONT_HERSHEY_SIMPLEX,1,(255,255,255),2)
cv2.imshow('a',img)
cv2.waitKey()
cv2.destroyAllWindows()
```

그래픽 기능과 사용자 인터페이스 만들기

```
def draw(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDOWN:
        cv2.rectangle(img, (x, y), (x + 200,y + 200), (255,255,255), 2)

cv2.setMouseCallback('drawing ', draw)
```

```
while True:
    if cv2.waitKey(1) == ord('q'):
        cv2.destroyAllWindows()
        break
```

- setMouseCallback 함수는 마우스에 이벤트가 발생하면 첫 번째 인수의 이름이 사용된 윈도우에 두 번째 인수의 함수를 실행한다.
- Draw 함수 내의 event_LBUTTONDOWN의 경우 마우스 왼쪽 버튼을 클릭할 경우 참이 되어 너비와 높이가 200인 직사각형을 그린다.
- 콜백 함수 등록 후에 프로그램이 끝나지 않게 하기 위해 무한 루프를 사용한다.

그래픽 기능과 사용자 인터페이스 만들기

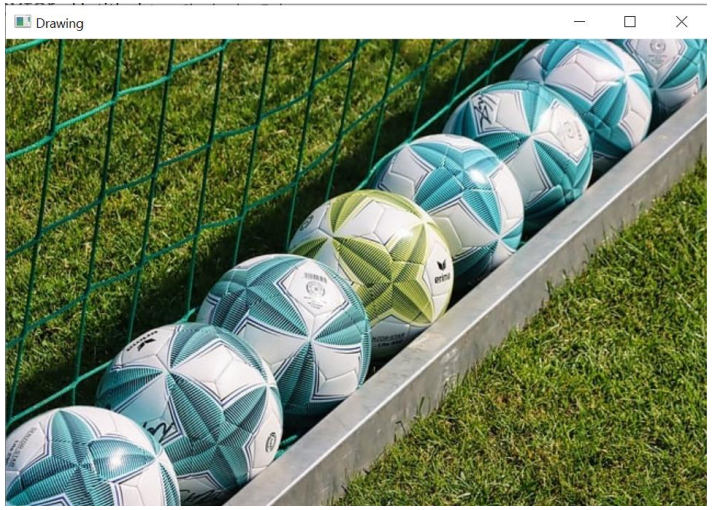
```
def draw(event, x, y, flags, param):
    global ix, iy

    if event == cv2.EVENT_LBUTTONDOWN:
        ix, iy = x, y
    elif event == cv2.EVENT_LBUTTONUP:
        cv2.rectangle(img, (ix, iy), (x, y) ,(255, 255, 255), 2)
```

- Ix와 iy에는 처음 마우스를 클릭했을 때의 x,y 값이 저장된다.
- 두 번째 마우스를 클릭할 경우 그때의 x,y 좌표까지 직사각형이 그려진다.

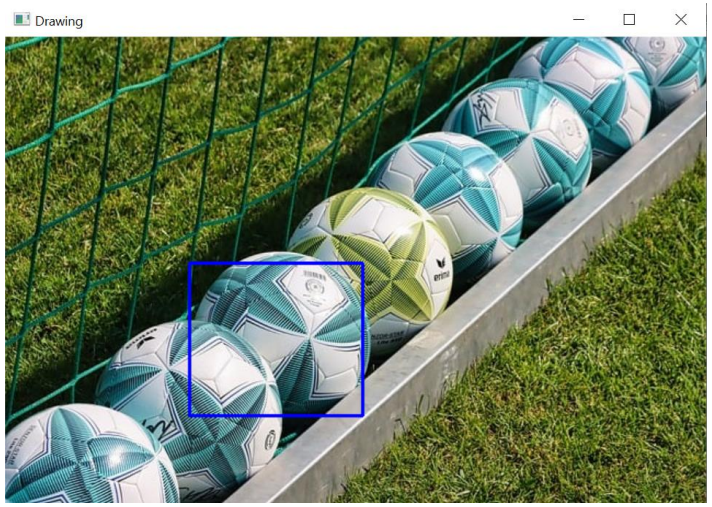
그래픽 기능과 사용자 인터페이스 만들기 | 실행 결과

```
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('nothing')

def draw(event,x,y,flags,param):
    global ix, iy

    if event == cv2.EVENT_LBUTTONDOWN:
        ix, iy = x,y

    elif event == cv2.EVENT_LBUTTONUP:
        cv2.rectangle(img,(x,y),(ix,iy),(255,0,0),2)

    cv2.imshow('Drawing',img)

cv2.namedWindow('Drawing')
cv2.imshow('Drawing',img)

cv2.setMouseCallback('Drawing',draw)
```

```
while True:
    if cv2.waitKey(1) == ord('q'):
        cv2.destroyAllWindows()
        break
```



| 마우스를 눌렀을 때



| 마우스를 놓았을 때

페인팅 기능 만들기

```
brushsiz = 5
color = (255, 0, 0)
```

```
def painting(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDOWN:
    cv2.circle(img, (x, y), brushsiz, color, -1)
elif event == cv2.EVENT_MOUSEMOVE and
flags ==  cv2.EVENT_FLAG_LBUTTON:
    cv2.circle(img, (x, y), brushsiz, color, -1)
```

- 마우스 왼쪽 버튼을 클릭할 경우 원을 그리며, 그대로 이동할 경우에도 원을 그리기에 색칠한 것처럼 나타난다.
- Circle 함수의 다섯 번째 함수는 두께를 지정하는데, -1로 지정하면 내부를 채운다.

페인팅 기능 만들기 | 실행 결과

```
import cv2
import sys

img = cv2.imread('soccer.jpg')

if img is None:
    sys.exit('nothing')

brushsiz = 5
lcolor,rcolor=(255,0,0),(0,0,255)

def painting(event,x,y,flags,param):

    if event == cv2.EVENT_LBUTTONDOWN:
        cv2.circle(img,(x,y),brushsiz,lcolor,-1)

    elif event == cv2.EVENT_MOUSEMOVE and flags == cv2.EVENT_LBUTTONDOWN:
        cv2.circle(img,(x,y),brushsiz,lcolor,-1)

    cv2.imshow('painting',img)

cv2.namedWindow('painting')
cv2.imshow('painting',img)

cv2.setMouseCallback('painting',painting)
```

```
while True:
    if cv2.waitKey(1) == ord('q'):
        cv2.destroyAllWindows()
        break
```