python™

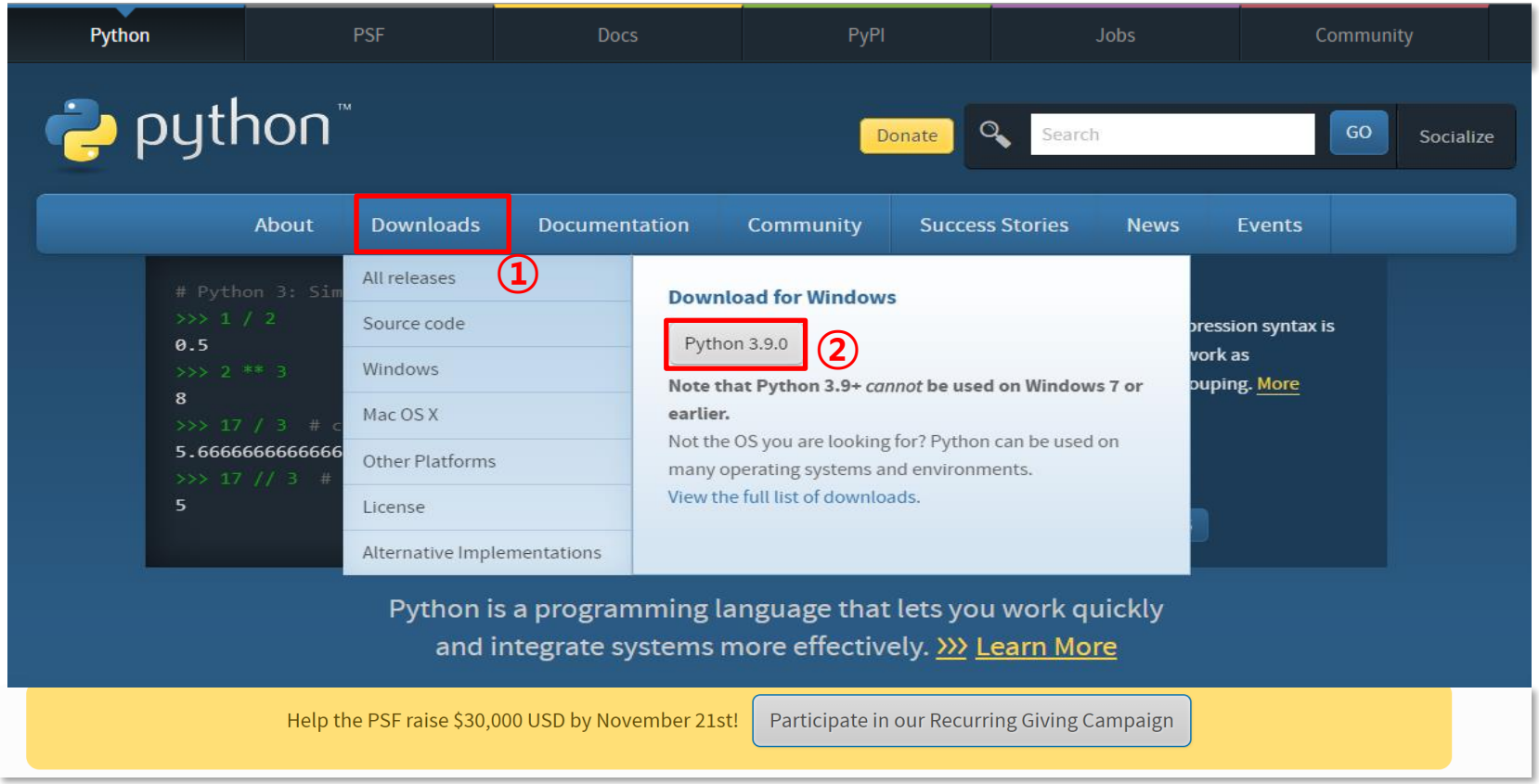
CoDrone 파이썬

Robot Edutainment Company
ROBOLINK

목차

Python 설치

1. Python 설치



https://www.python.org/downloads/

1. Python 설치

https://www.python.org/downloads/windows/

만약 Windows 8.1 이하라면 위 링크를 통해 아래 표시된 Python 버전을 다운로드 받아주세요

Note that Python 3.9.0 *cannot* be used on Windows 7 or earlier.

- Download Windows help file
- Download Windows x86-64 embeddable zip file
- Download Windows x86-64 executable installer
- Download Windows x86-64 web-based installer
- Download Windows x86 embeddable zip file
- Download Windows x86 executable installer
- Download Windows x86 web-based installer

- Python 3.8.6 - Sept. 24, 2020

Note that Python 3.8.6 *cannot* be used on Windows XP or earlier.

- Download Windows help file
- Download Windows x86-64 embeddable zip file
- Download Windows x86-64 executable installer
- Download Windows x86-64 web-based installer
- Download Windows x86 embeddable zip file
- Download Windows x86 executable installer
- Download Windows x86 web-based installer

- Python 3.8.6rc1 - Sept. 8, 2020

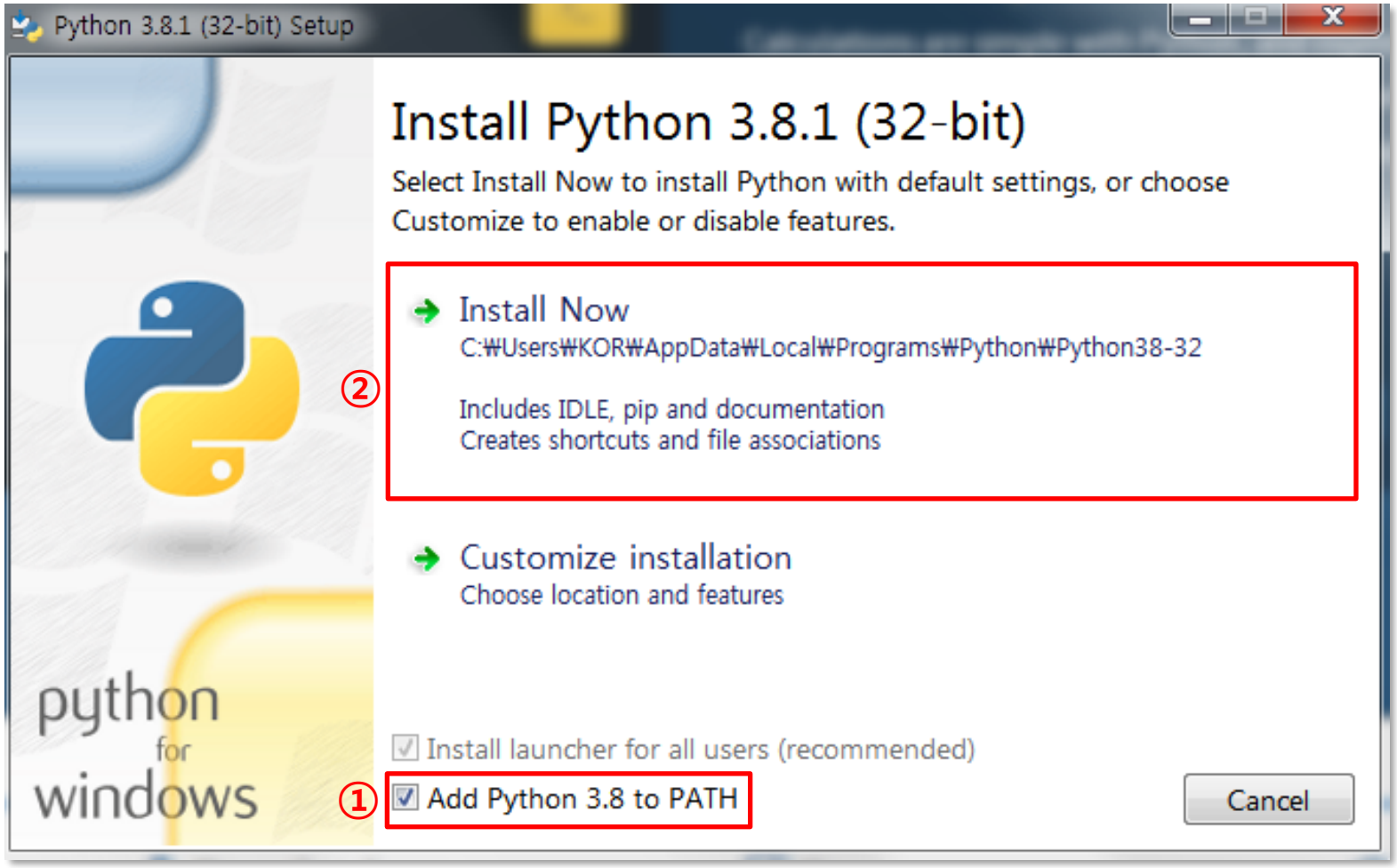
Note that Python 3.8.6rc1 *cannot* be used on Windows XP or earlier.

- Download Windows help file
- Download Windows x86-64 embeddable zip file
- Download Windows x86-64 executable installer
- Download Windows x86-64 web-based installer
- Download Windows x86 embeddable zip file
- Download Windows x86 executable installer

- Download Windows help file
- Download Windows x86-64 embeddable zip file
- Download Windows x86-64 executable installer
- Download Windows x86-64 web-based installer
- Download Windows x86 embeddable zip file
- Download Windows x86 executable installer
- Download Windows x86 web-based installer

- Python 3.9.0rc2 - Sept. 17, 2020
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file
  - Download Windows x86 executable installer
  - Download Windows x86 web-based installer
- Python 3.5.10rc1 - Aug. 22, 2020
  - No files for this release.
- Python 3.9.0rc1 - Aug. 11, 2020
  - Download Windows help file
  - Download Windows x86-64 embeddable zip file
  - Download Windows x86-64 executable installer
  - Download Windows x86-64 web-based installer
  - Download Windows x86 embeddable zip file
  - Download Windows x86 executable installer
  - Download Windows x86 web-based installer
- Python 3.9.0b5 - July 20, 2020

1. Python 설치


Python 3.8.1 (32-bit) Setup
Install Python 3.8.1 (32-bit)
Select Install Now to install Python with default settings, or choose
Customize to enable or disable features.
Install Now
C:₩Users₩KOR₩AppData₩Local₩Programs₩Python₩Python38-32
Includes IDLE, pip and documentation
Creates shortcuts and file associations
②
Customize installation
Choose location and features
python
for
windows
Install launcher for all users (recommended)
①
Add Python 3.8 to PATH
Cancel

Library 설치

2. 라이브러리 설치

PIP Python PIP, 파이썬 패키지 매니저 PIP

◆ **PIP란?**

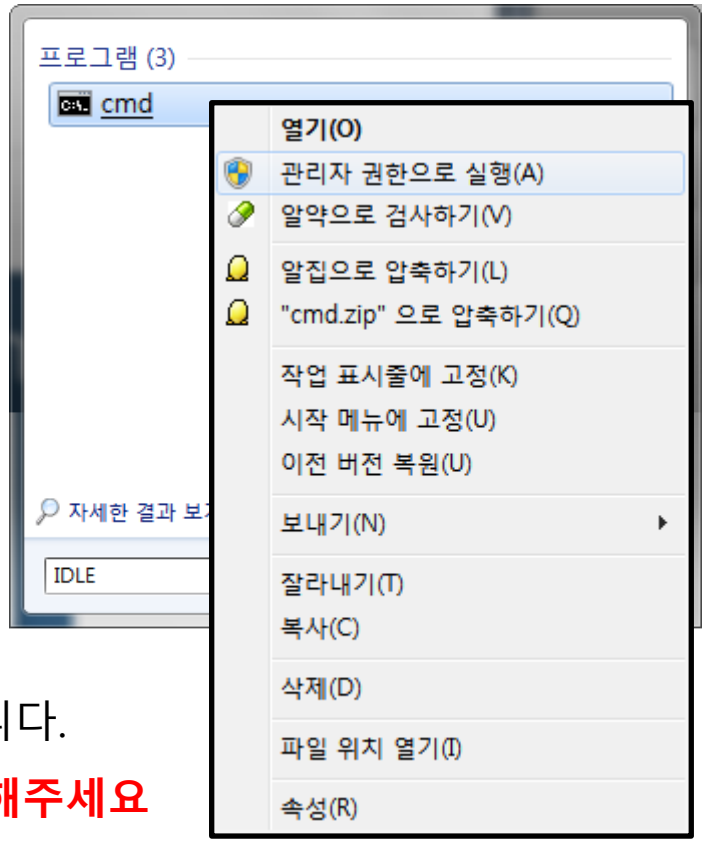
PIP는 파이썬 패키지를 위한 패키지로, Python은 손쉽게 온라인에 접속해 있다면 필요로 하는 다양한 SW,라이브러리들을 쉽게 설치 및 업데이트 할 수 있습니다.

* 파이썬 버전 3.4 이상을 설치하셨다면, PIP는 기본적으로 포함

◆ **패키지란?**

패키지는 모듈을 위해 필요한 모든 것을 포함하며, 모듈은 프로젝트에 포함되기 위한 파이썬 코드의 라이브러리들입니다.

2. 라이브러리 설치



시작 –> **cmd(명령 프롬포트) 클릭**하여 실행합니다.

※ 마우스 오른쪽을 눌러 관리자 권한으로 실행해주세요

2. 라이브러리 설치

PIP Upgrade

명령 프롬프트

```
C:₩Users₩metal>python -m pip install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/54/0c/d01aa759fdc5
2ce14b5845662c13f3/pip-20.0.2-py2.py3-none-any.whl (1.4MB)
    100% |                                | 1.4MB 3.9MB/s
Installing collected packages: pip
  Found existing installation: pip 10.0.1
    Uninstalling pip-10.0.1:
      Successfully uninstalled pip-10.0.1
Successfully installed pip-20.0.2

C:₩Users₩metal>_
```

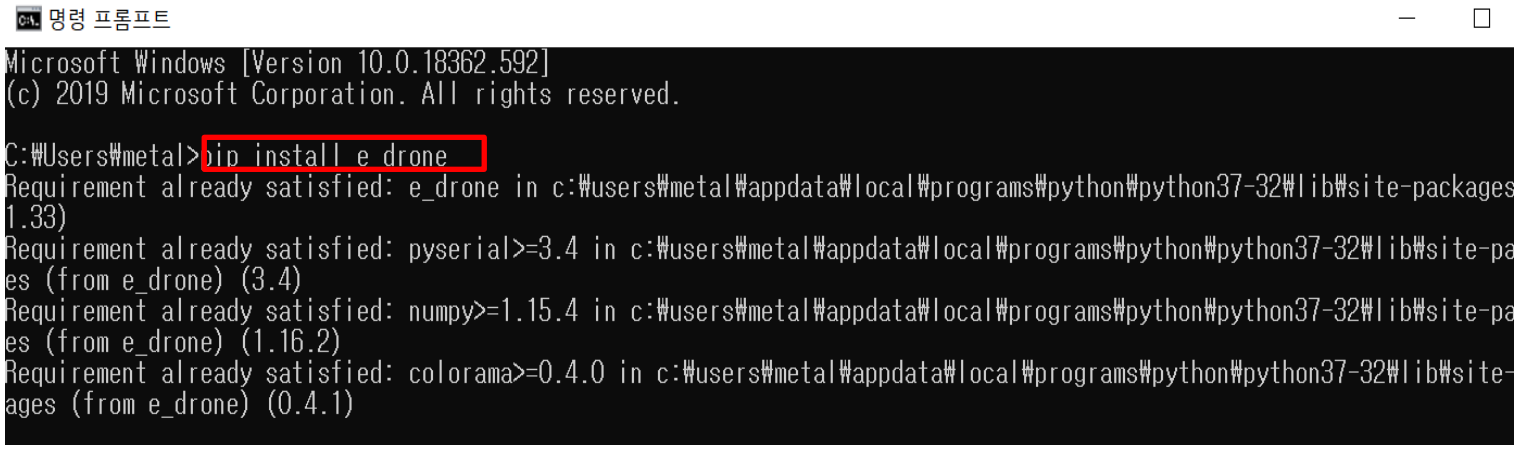
가끔 pip 버전이 이전 버전 이어서 안 되는 경우가 있을 때 업그레이드 줍니다.

python –m pip install —upgrade pip입력하면 pip가 업그레이드 됩니다.

※ 해당 부분은 PC마다 다를 수 있습니다.

2. 라이브러리 설치

Install e_drone



pip install e_drone 입력하면 라이브러리 설치됩니다.

설치되어있는 경우에는 현재 설치된 버전정보가 출력됩니다.

2. 라이브러리 설치

Upgrade e_drone

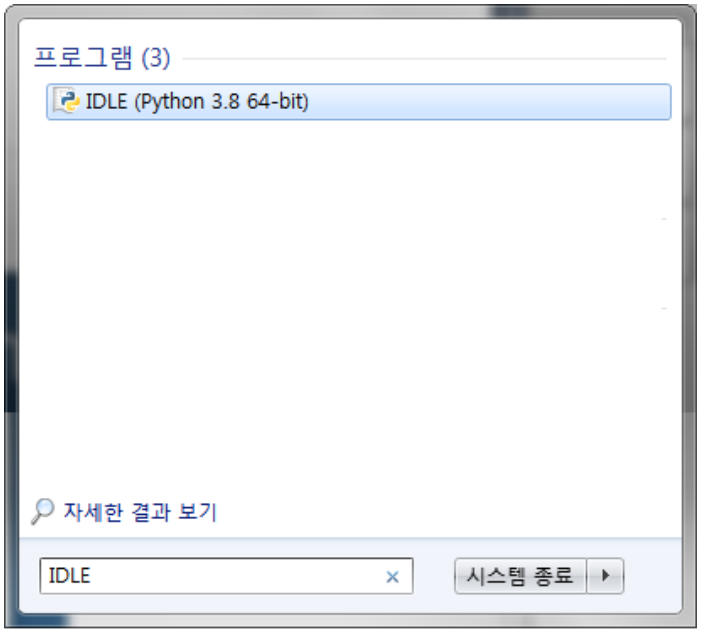
명령 프롬프트

```
C:₩Users₩metal>pip install --upgrade e_drone
Requirement already up-to-date: e_drone in c:₩users₩metal₩appdata₩local₩programs₩
.1.33)
Requirement already satisfied, skipping upgrade: pyserial>=3.4 in c:₩users₩metal₩
32₩lib₩site-packages (from e_drone) (3.4)
Requirement already satisfied, skipping upgrade: numpy>=1.15.4 in c:₩users₩metal₩
32₩lib₩site-packages (from e_drone) (1.16.2)
Requirement already satisfied, skipping upgrade: colorama>=0.4.0 in c:₩users₩meta
7-32₩lib₩site-packages (from e_drone) (0.4.1)

C:₩Users₩metal>_
```

pip install --upgrade e_drone 입력하여 업그레이드 합니다.

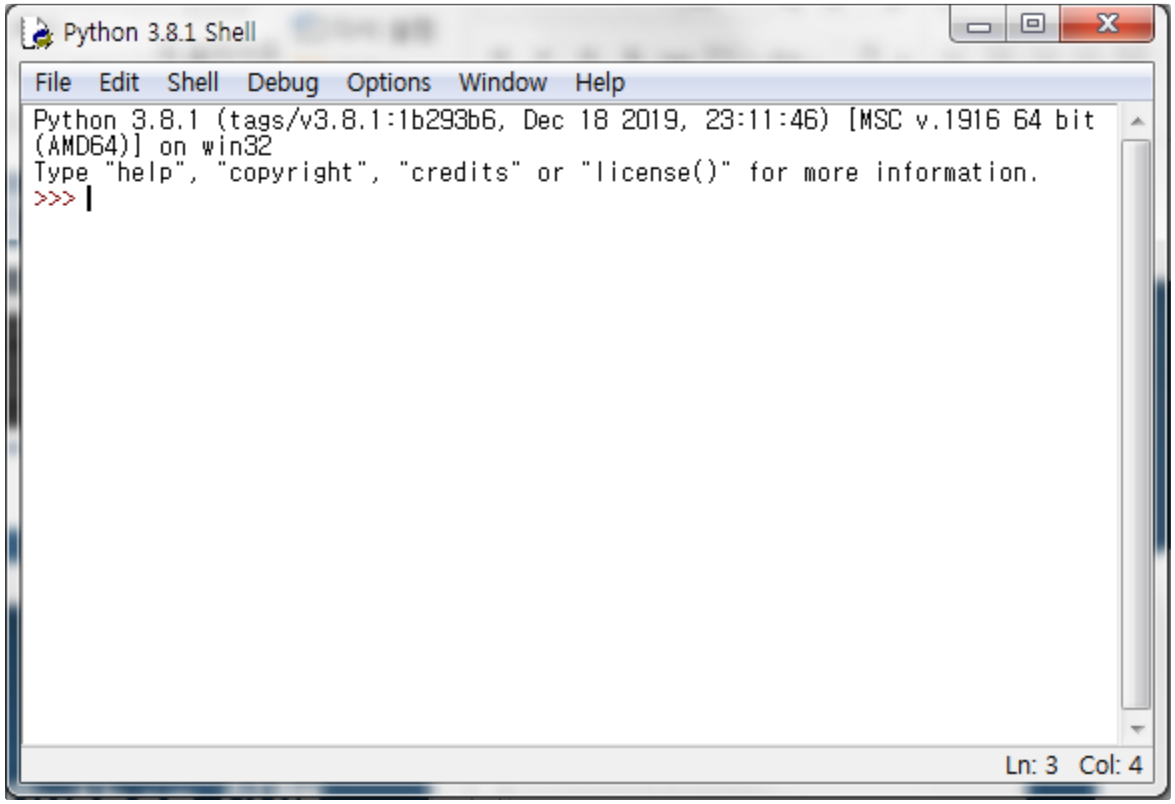
Python 실행

3. 파이썬 실행하기



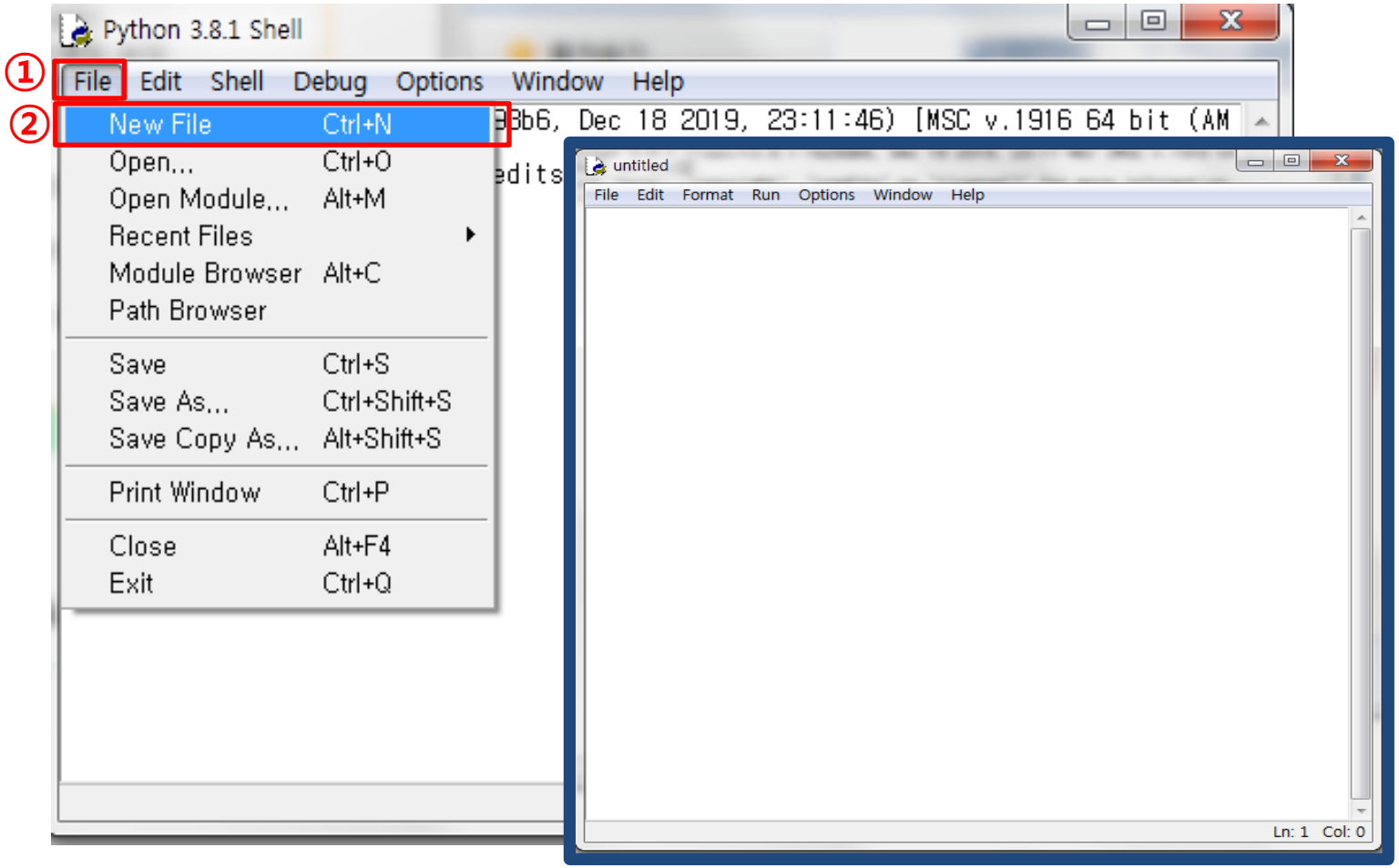
시작 –> IDLE(Python 3.8 64-bit) 클릭하여 실행합니다.

3. 파이썬 실행하기



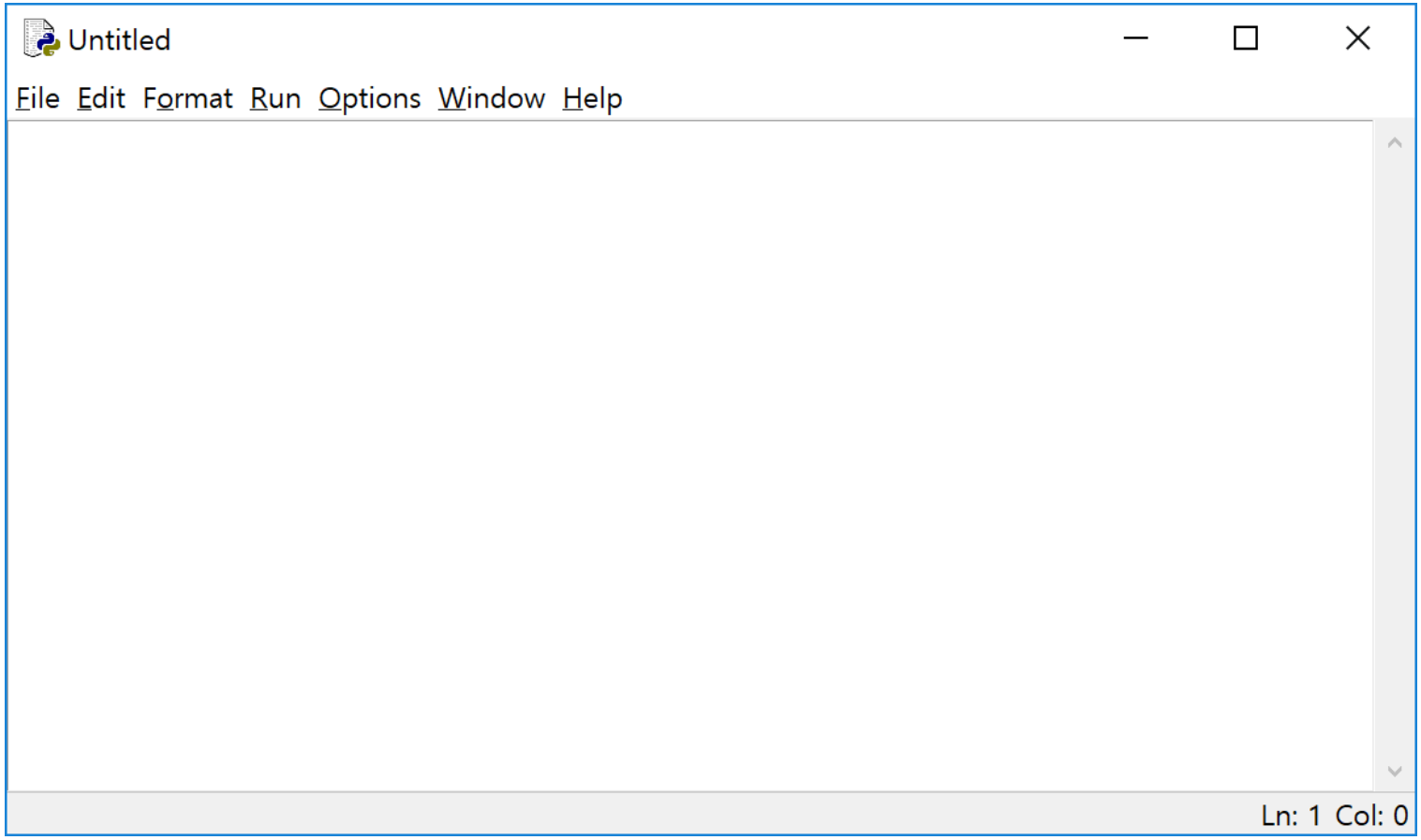
파이썬이 실행되면서 **새 창**이 열립니다.

3. 파이썬 실행하기



File -> New File 클릭 합니다.

3. 파이썬 실행하기



새 창이 열리면 여기에 **파이썬 프로그램을 작성**합니다.

Python 코딩

예제 목차

- LED를 빨강-녹색-파랑으로 변하게 하기
- LED를 랜덤으로 변하게 하기
- 이륙 - 착륙
- 이륙 – 전진 - 착륙
- 사각형 모양 패턴 비행
- 원 모양 패턴 비행
- 자이로 센서 값 읽기
- 고도 센서 값 읽기

4. CoDrone Programming

- 주의사항

※ 파이썬 프로그래밍을 하기 전에 드론이 흐르지 않게 미세조정 해주세요

롤 (ROLL) 미세 조정

드론이 오른쪽으로 흐른다면 왼쪽 버튼을 눌러 조정합니다.

드론이 왼쪽으로 흐른다면 오른쪽 버튼을 눌러 조정합니다.

피치 (PITCH) 미세 조정

드론이 앞쪽으로 흐른다면 아래 버튼을 눌러 조정합니다.

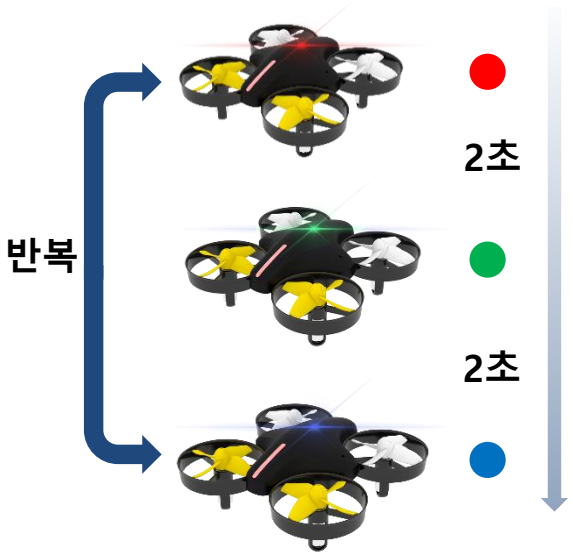
드론이 뒤쪽으로 흐른다면 위 버튼을 눌러 조정합니다.

4. CoDrone Programming

- LED R,G,B 2초간 반복점멸 패턴

```
def sendLightDefaultColor(self, lightMode, interval, r, g, b):
```

변수 이름	형식 또는 범위	설명
lightMode	UInt8	LED 동작 모드
interval	0 ~ 65535	내부 밝기 제어 함수 호출 주기
r	0 ~ 255	Red
g	0 ~ 255	Green
b	0 ~ 255	Blue



untitled

File Edit Format Run Options Window Help

```
import random
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


if __name__ == '__main__':

    drone = Drone(True, True, True, True, True)
    drone.open()

    while True:

        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 255, 0, 0)
        sleep(2)
        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 0, 255, 0)
        sleep(2)
        drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, 0, 0, 255)
        sleep(2)

    drone.close()
```

LED RGB 입니다.
(0~255)

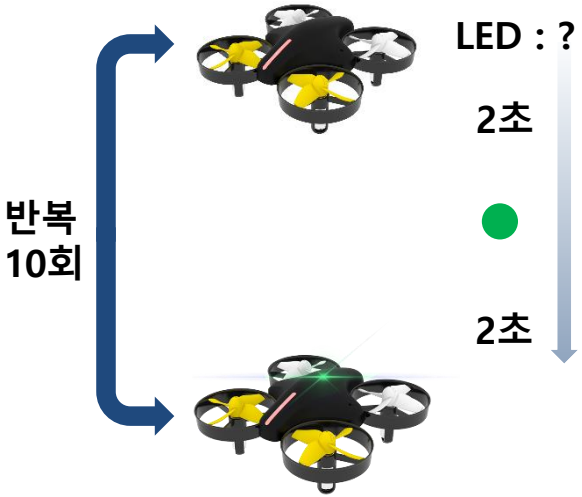
Ln: 21 Col: 0

4. CoDrone Programming

- LED 랜덤한 색으로 점점 밝아졌다 어두워 지게 하는 명령 10회 실행

```
def sendLightDefaultColor(self, lightMode, interval, r, g, b):
```

변수 이름	형식 또는 범위	설명
lightMode	UInt8	LED 동작 모드
interval	0 ~ 65535	내부 밝기 제어 함수 호출 주기
r	0 ~ 255	Red
g	0 ~ 255	Green
b	0 ~ 255	Blue



untitled

File Edit Format Run Options Window Help

```
import random
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


if __name__ == '__main__':

    drone = Drone(True, True, True, True, True)
    drone.open()

    for i in range(0, 10, 1):

        r    = int(random.randint(0, 255))
        g    = int(random.randint(0, 255))
        b    = int(random.randint(0, 255))

        dataArray = drone.sendLightDefaultColor(LightModeDrone.BodyDimming, 1, r, g, b)
        print("{0} / {1}".format(i, convertByteArrayToString(dataArray)))

        sleep(2)

    drone.close()
```

Ln: 21 Col: 0

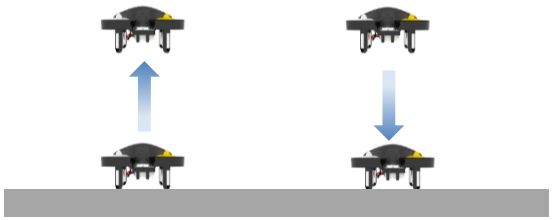
4. CoDrone Programming

- 이륙, 호버링, 착륙

```
def sendControlWhile(self, roll, pitch, yaw, throttle, timeMs):
```

변수 이름	형식 또는 범위	설명
roll	-100 ~ 100	Roll
pitch	-100 ~ 100	Pitch
yaw	-100 ~ 100	Yaw
throttle	-100 ~ 100	Throttle

- sendTakeOff()
- sendControlWhile()
- sendLanding()



untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 5000)

    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)


    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

Ln: 21 Col: 0

드론 명령어는 약 0.01초 정도의 delay를 줘야 합니다.

드론 5초간 호버링 비행으로 멈추는 루틴

1초 기다리기

드론 landing 함수는 2번 정도씩 호출하여 멈추게 합니다.

4. CoDrone Programming

- 이륙, 호버링, 전진, 착륙

- sendTakeOff()
- sendControlWhile()
- sendLanding()

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 5000)

    print("Go Start")
    drone.sendControlWhile( 0, 50, 0, 0, 2000)


    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)


    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

이륙 명령

호버링 명령

2초간 전진 (Pitch) 50% 속도 명령

1초 기다리기

착륙 명령

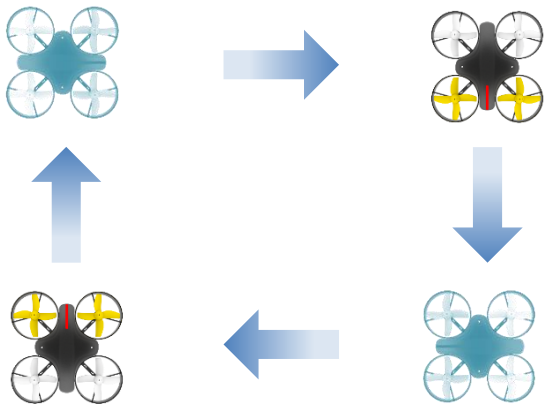
Ln: 21 Col: 0

4. CoDrone Programming

- 사각 패턴 비행

- sendTakeOff()
- sendControlWhile()
- sendLanding()

속도와 시간을 조절하여 정교하게 비행 시켜 봅시다.



untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 4000)

    print("Go Start")
    drone.sendControlWhile( 0, 50, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( 50, 0, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( 0, -50, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    drone.sendControlWhile( -50, 0, 0, 0, 1000)
    drone.sendControlWhile(0, 0, 0, 0, 1000)
    print("Go Stop")

    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

이륙 명령

호버링 명령

1초간 전진 (Pitch 50)

1초간 우측 (Roll 50)

1초간 후진 (Pitch -50)

1초간 좌측 (Roll -50)

착륙 명령

Ln: 21 Col: 0

4. CoDrone Programming

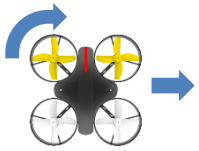
- 원형 패턴 비행

- sendTakeOff()
- sendControlWhile()
- sendLanding()

속도를 변형하여 원의 크기를
조절하여 정교하게 비행 시켜 봅시다.

시계방향 회전

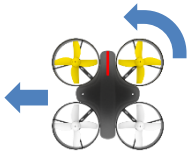
```
drone.sendControlWhile(50, 0, -50, 0, 4000)
```



Roll (50) **Yaw** (-50)

반시계방향 회전

```
drone.sendControlWhile( -50, 0, 50, 0, 4000)
```



Roll (-50) **Yaw** (50)

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)

    print("Hovering")
    drone.sendControlWhile(0, 0, 0, 0, 4000)

    print("Go Start")
    drone.sendControlWhile( 50, 0, -50, 0, 4000)

    print("Go Stop")
    drone.sendControlWhile(0, 0, 0, 0, 1000)

    print("Landing")
    drone.sendLanding()
    sleep(0.01)
    drone.sendLanding()
    sleep(0.01)

    drone.close()
```

이륙 명령

호버링 명령

4초간 회전비행

착륙 명령

Ln: 21 Col: 0

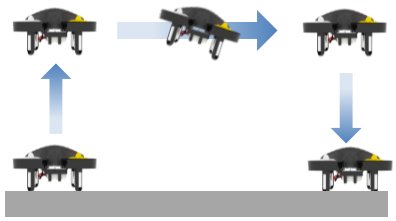
4. CoDrone Programming

- 이륙, 호버링, 전진, 착륙 테스트

```
def sendControl(self, roll, pitch, yaw, throttle):
```

- sendControl()

변수 이름	형식 또는 범위	설명
roll	-100 ~ 100	Roll
pitch	-100 ~ 100	Pitch
yaw	-100 ~ 100	Yaw
throttle	-100 ~ 100	Throttle



```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    print("TakeOff")
    drone.sendTakeOff()
    sleep(0.01)
    for i in range(3, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Hovering")
    for i in range(1, 0, -1):
        print("{0}".format(i))
        drone.sendControlWhile(0, 0, 0, 0, 1000)
        sleep(1)
```

이륙 명령

비행시간 화면 출력 루틴

호버링 명령

```
    print("Go Start")
    drone.sendControl(0, 50, 0, 0)
    for i in range(3, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Go Stop")
    drone.sendControl( 0, 0, 0, 0)
    for i in range(2, 0, -1):
        print("{0}".format(i))
        sleep(1)

    print("Landing")
    drone.sendLanding()
    sleep(0.01)

drone.close()
```

1초간 전진

착륙 명령

4. CoDrone Programming

- 자이로, 가속도 센서 값 읽기

- Motion
- sendRequest()

변수 이름	형식 또는 범위	설명
deviceType	DeviceType	전송할 대상 장치
dataType	DataType	데이터의 타입

```
def sendRequest(self, deviceType, dataType):
```

untitled

File Edit Format Run Options Window Help

```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


def eventMotion(motion):
 print("eventMotion()")
 print("- Accel: {0:5}, {1:5}, {2:5}".format(motion.accelX, motion.accelY, motion.accelZ))
 print("-  Gyro: {0:5}, {1:5}, {2:5}".format(motion.gyroRoll, motion.gyroPitch, motion.gyroYaw))
 print("- Angle: {0:5}, {1:5}, {2:5}".format(motion.angleRoll, motion.anglePitch, motion.angleYaw))


if __name__ == '__main__':

    drone = Drone()
    drone.open()

    # 이벤트 핸들링 함수 등록
    drone.setEventHandler(DataType.Motion, eventMotion)

    while True:

        # Range 정보 요청
        drone.sendRequest(DeviceType.Drone, DataType.Motion)
        sleep(1)

    drone.close()
```

Ln: 21 Col: 0

4. CoDrone Programming

- 고도 센서 값 읽기

- Attitude
- sendRequest()

변수 이름	형식 또는 범위	설명
deviceType	DeviceType	전송할 대상 장치
dataType	DataType	데이터의 타입

```
def sendRequest(self, deviceType, dataType):
```

untitled

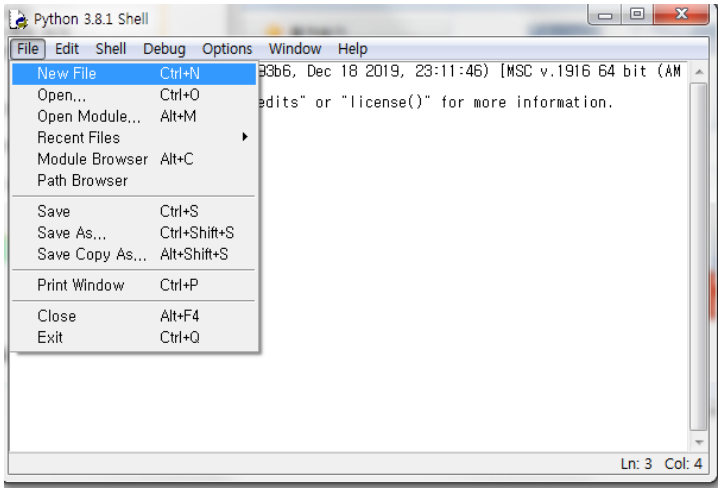
File Edit Format Run Options Window Help

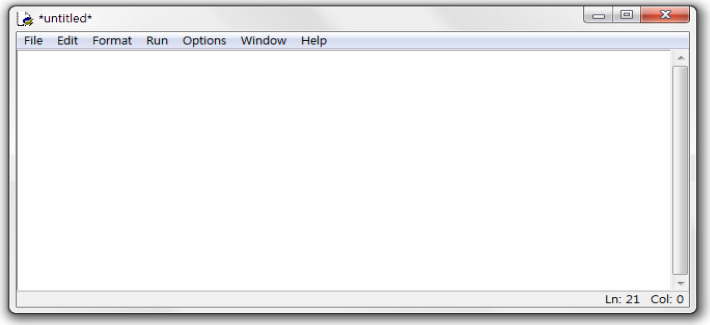
```
from time import sleep

from e_drone.drone import *
from e_drone.protocol import *


def eventAltitude(altitude):
    print("eventAltitude()")
    print("-  Temperature: {0:.3f}".format(altitude.temperature))
    print("-     Pressure: {0:.3f}".format(altitude.pressure))
    print("-     Altitude: {0:.3f}".format(altitude.altitude))
    print("- Range Height: {0:.3f}".format(altitude.rangeHeight))

if __name__ == '__main__':

    drone = Drone()
    drone.open()

    # 이벤트 핸들링 함수 등록
    drone.setEventHandler(DataType.Altitude, eventAltitude)

    while True:

        # Altitude 정보 요청
        drone.sendRequest(DeviceType.Drone, DataType.Altitude)
        sleep(1)

    drone.close()
```

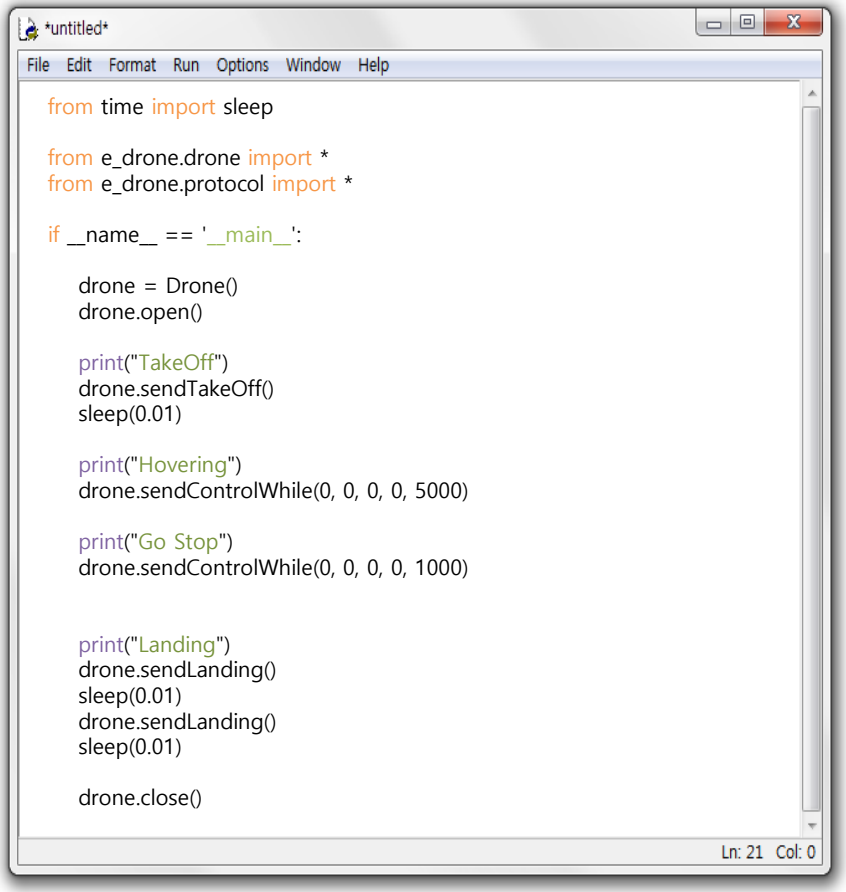
Ln: 21 Col: 0

Python 디버깅

5. CoDrone Programming

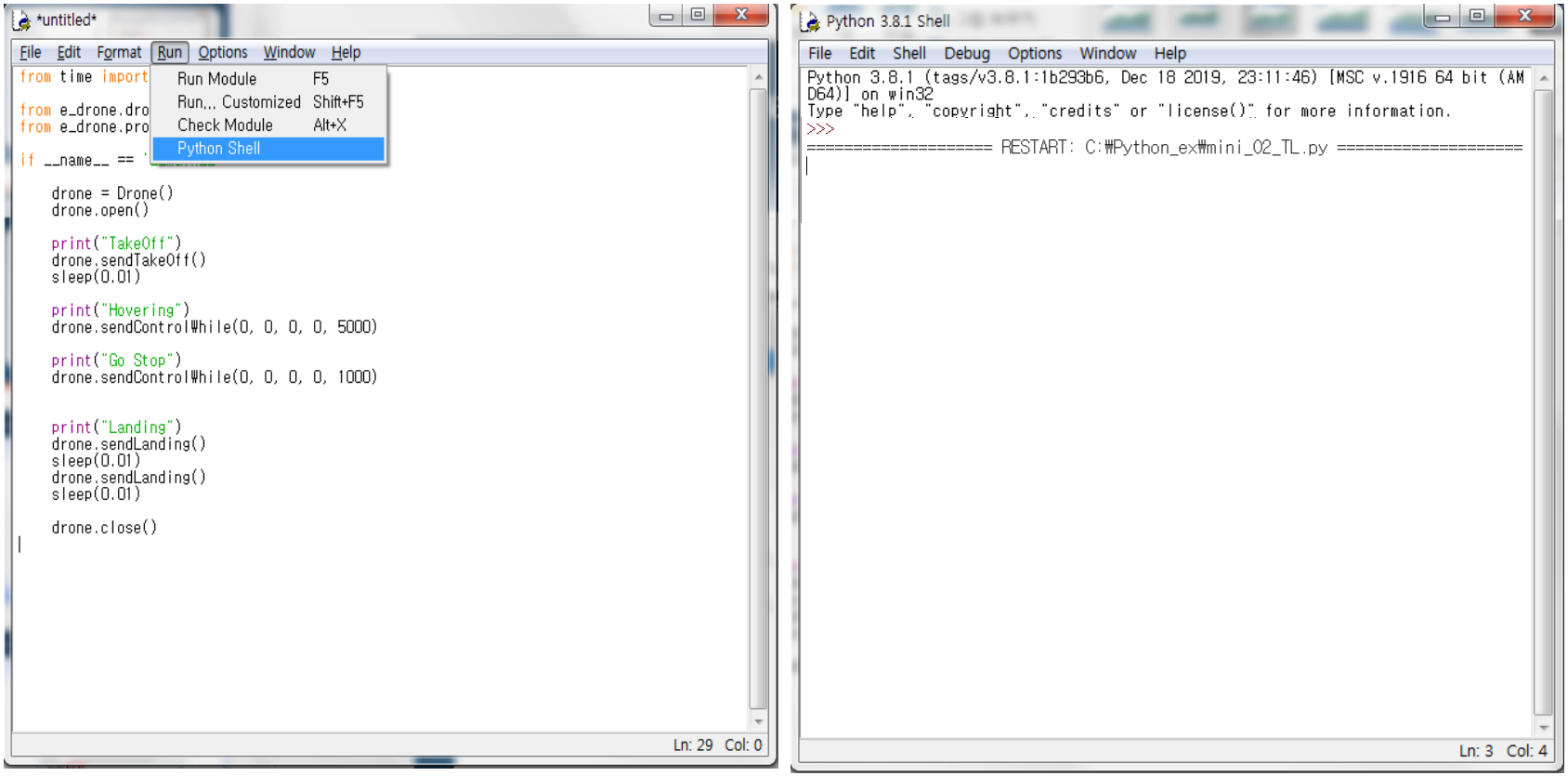






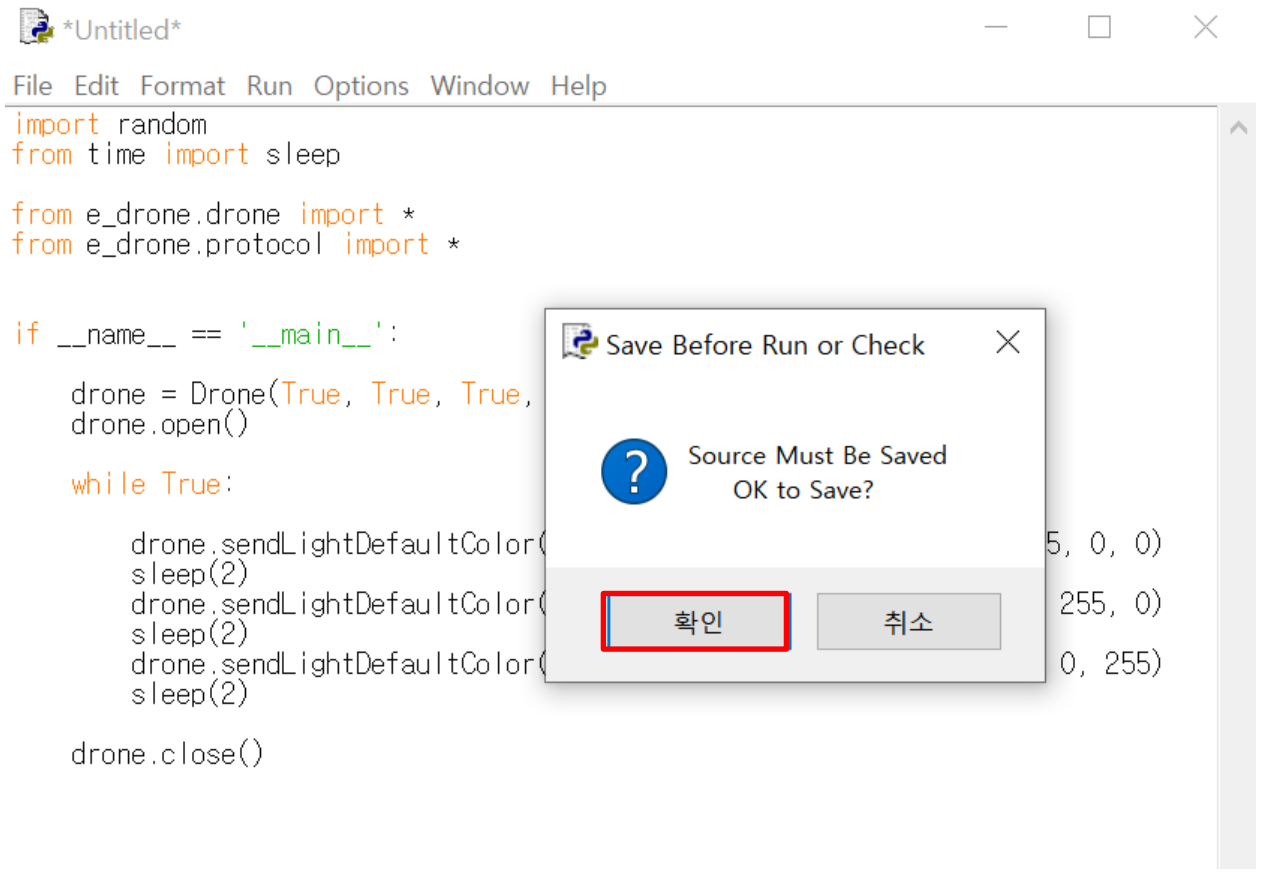
Python IDLE 프로그램에서 New File을 열고
프로그램을 타이핑하거나 복사합니다.

5. CoDrone Programming



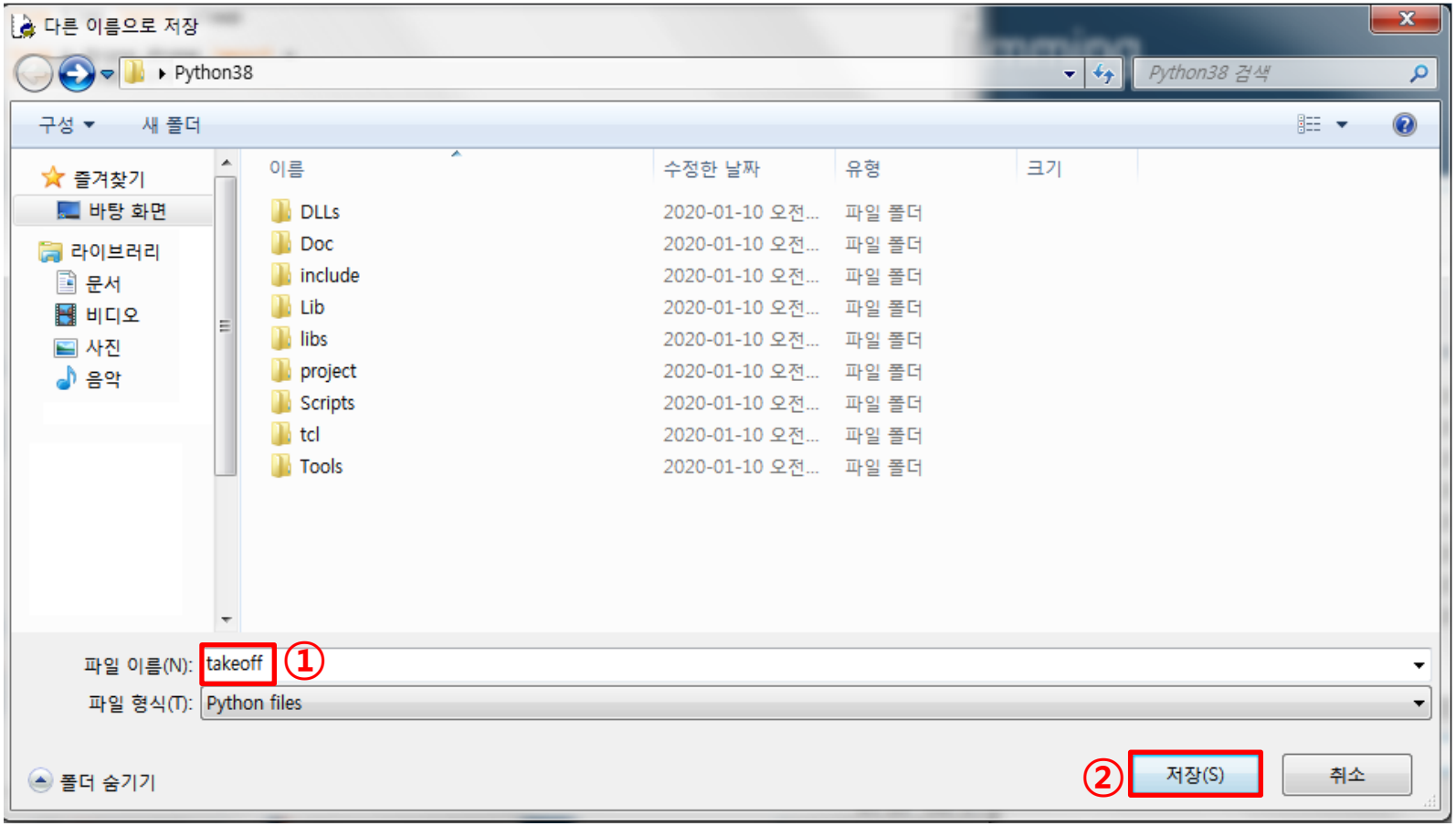
Run – Run Module 클릭 또는 **F5**눌러서 프로그램 실행한다.

5. CoDrone Programming



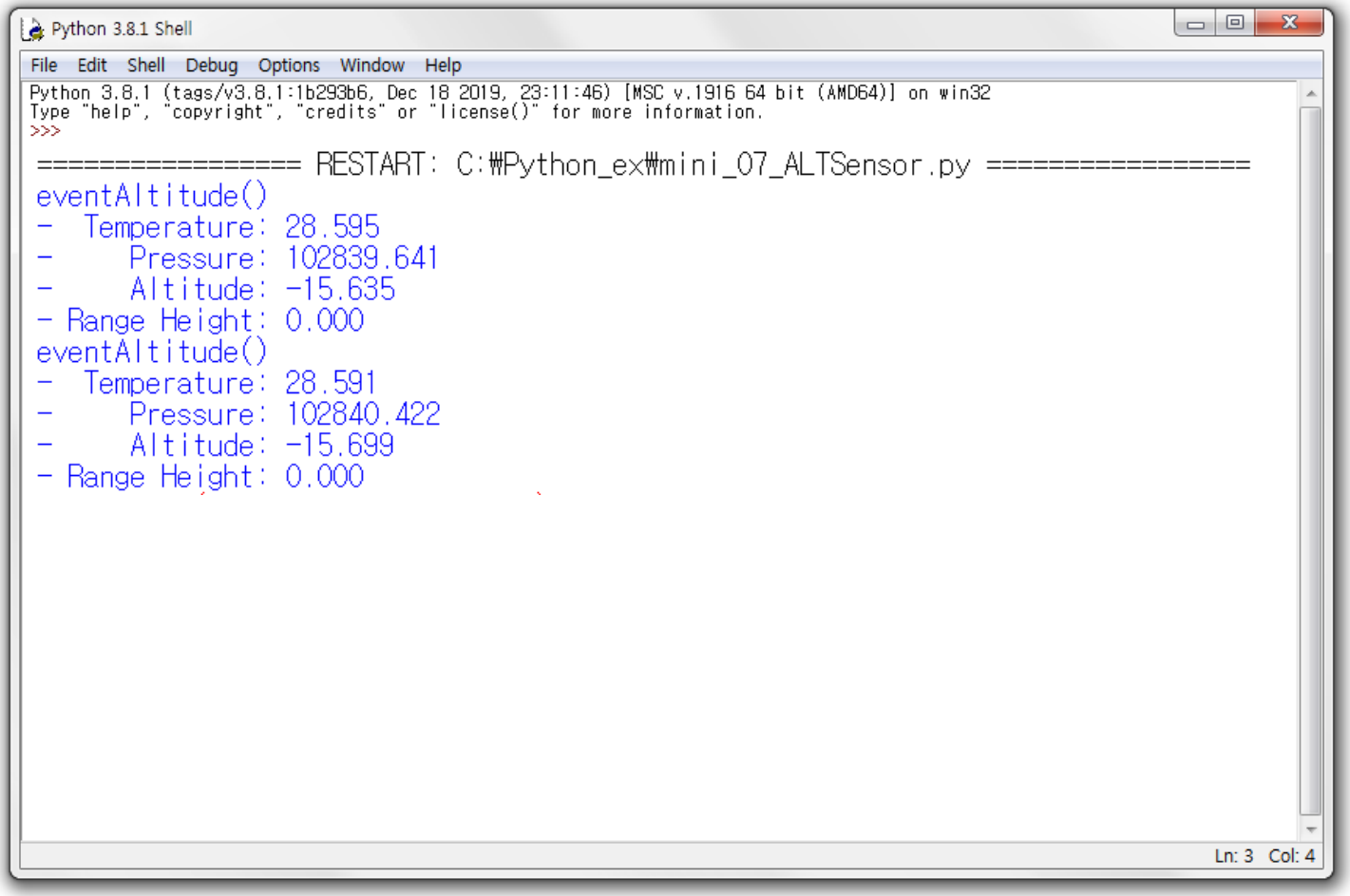
프로그램이 저장되어 있어야 실행이 되므로 확인을 클릭합니다.

5. CoDrone Programming



파일 이름을 입력 후 저장을 클릭한다.

5. CoDrone Programming



Shell 창이 열리고 화면 출력 후 프로그램이 실행됩니다.

5. CoDrone Programming

파이썬 라이브러리 프로토콜은 아래 링크를 통해 함수 및 프로토콜을 확인 할 수 있습니다.

http://dev.byrobot.co.kr/documents/kr/products/e_drone/library/python/e_drone/

e_drone for python

1. Intro
2. System
3. Protocol
4. Drone
5. Examples - Ping
6. Examples - Information
7. Examples - Pairing
8. Examples - Control
9. Examples - Sensor
10. Examples - Motor
11. Examples - Setup
12. Examples - Buzzer
13. Examples - Vibrator
14. Examples - Light
15. Examples - Display
16. Examples - Input
17. Examples - Error

Examples - Ping

Examples - Information

Examples - Pairing

Examples - Control

Examples - Sensor

Examples - Motor

Examples - Setup

Examples - Buzzer

Examples - Vibrator

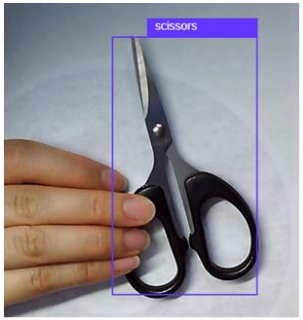
Examples - Light

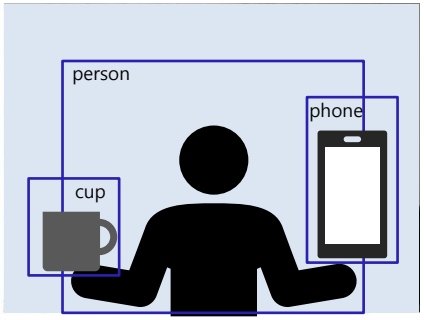
Examples - Display

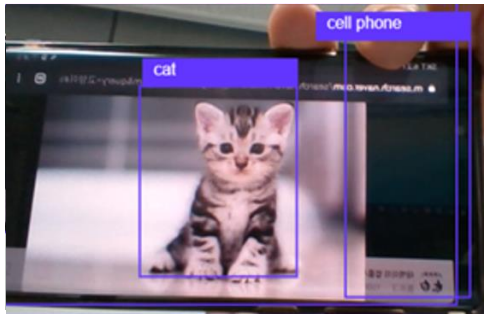
Examples - Input

Open CV를 사용한 영상 인식

1. 영상인식과 드론







Opencv 이것은 아마 영상인식에 대한 관심이 있는 모든 사람들이 한번씩 들어본 용어일 겁니다.

OpenCV는 Open Source Computer Vision의 약자이며 실시간 컴퓨터 비전을 목적으로 한 프로그래밍 라이브러리입니다. 실시간으로 이미지 프로세싱을 쉽게 할 수 있도록 인텔에서 만든 라이브러리이고 OpenCV는 사물의 색깔, 모양, 사람의 얼굴, 제스쳐등 기본적인 영상인식뿐아니라 TensorFlow, Torch등의 딥러닝 프로엠워크까지 지원합니다.

즉 Opencv는 영상으로 사물의 특성을 분석하여 인식하게하는 도구이며, 이를 라이브러리화 하여 우리가 쉽게 영상인식에 활용할 수 있도록 만든 기술입니다. 이를 통해 주자창의 번호판 인식,
자동차의 불법단속, CCTV등 다양한 서비스로 활용되고 있는 겁니다.

1. 영상인식과 드론

이중에서 TensorFlow는 머신 러닝을 위한 오픈소스 소프트웨어로 구글(Google)사에서 개발한 기계 학습(machine learning) 엔진. 검색, 음성 인식, 번역 등의 구글 앱에 사용되는 기계 학습용 엔진으로, 2015년에 공개 소스 소프트웨어(open source software)로 전환되었다.

텐서플로는 C++ 언어로 작성되었고, 파이선(Python) 응용 프로그래밍 인터페이스(API)를 제공하여

Opencv의 영상인식을 머신러닝으로 융합하는데 사용되기 때문에 인공지능에 관심있는 분은 기억해두면 좋을 것 같습니다.

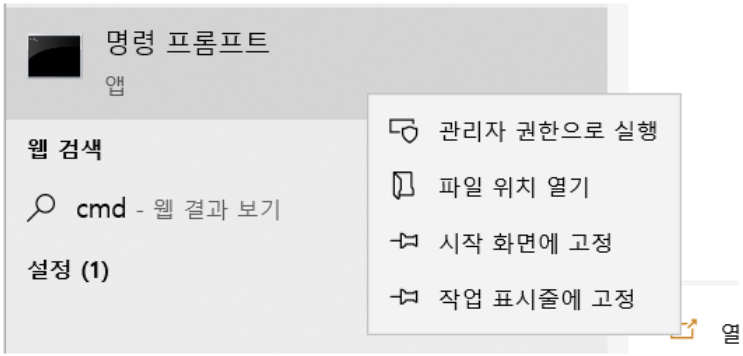
OpenCV는 C/C++ 프로그래밍 언어로 개발 되었으며 파이썬 , 자바 및 매트랩 / OCTAVE에 바인딩 되어

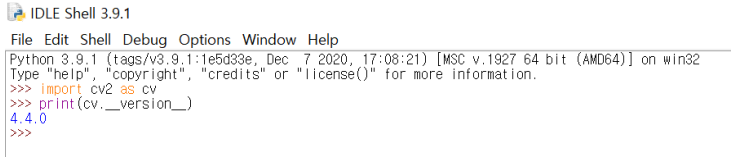
프로그래머에게 개발 환경을 지원합니다.

특히 요즘 파이썬에서 많이 사용되고 있고 그중에서도 인공지능쪽으로 특히 활용되고있습니다.

2. 파이썬에서 OpenCE 설치 방법

컴퓨터의 검색창에 cmd 를 검색해 오른쪽 마우스 클릭 --> 관리자 권한으로 실행 을 클릭 해 준다.





파이썬에서 Opencv를 하기 위해서는 필요한 작업이 있습니다.

앞서서 파이썬은 pip를 통해 다양한 라이브러리를 손쉽게 설치할 수 있었습니다.

Opencv도 이와 마찬가지로 관련 라이브러리를 설치해야합니다.

Pip install opencv-python opencv의 메인 모듈 설치

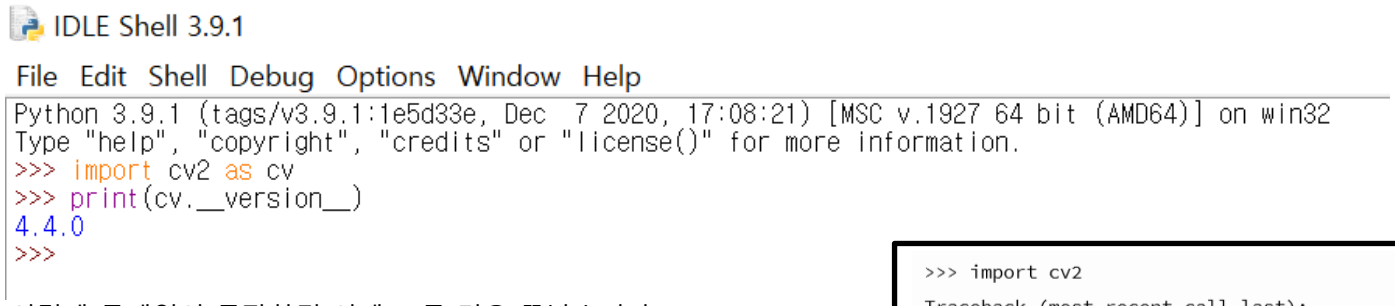
Pip install opencv-contrib-python contrib 모듈(래퍼 패키지)

Pip install numpy 데이터 분석 환경에서 많이 사용되는 행렬 연산을 위한 라이브러리

Pip install matplotlib 도표, 차트, 그래프 등을 구현할 수 있도록 해주는 그래픽 라이브러리

2. 파이썬에서 OpenCE 설치 방법

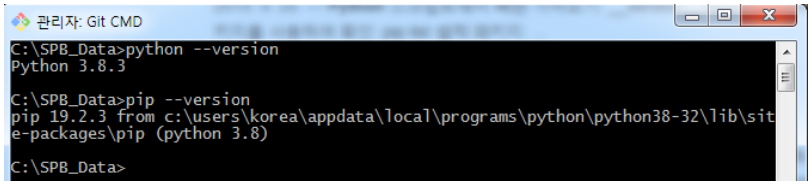
이렇게 문제없이 설치가 되었는지 확인하려면 python idle를 통해 다음과 같이 입력합니다.



이렇게 문제없이 동작하면 이제 모든 것은 끝났습니다.

만약 위 cv 버전확인에서 에러가 난다면, 다음 사항을 체크합니다. →

```
>>> import cv2
Traceback (most recent call last):
   File "<stdin>", line 1, in <module>
ModuleNotFoundError : No module named 'cv2'
```



1. 파이썬 버전이 여러개 설치되어있는 경우

파이썬이 여러 버전 설치되어있다면, pip version을 통해 지금 사용하는 파이썬 버전에 설치되었는지 확인하고, 해당 파이썬 버전을 설치하거나, opencv를 재설치한다.

2. opencv 설치가 제대로 안된 경우

pip를 uninstall한 후 모두 다시 install을 해줍니다.
이때도 역시 파이썬 버전을 확인하고 설치하기바랍니다.

Pip uninstall opencv-python
Pip uninstall opencv-contrib-python
Pip uninstall numpy
Pip uninstall matplotlib

3. 이미지 사진에서 원하는것 검출하기

코드론 미니 카메라 표시

우선 우리는 우리가 원하는 사진(이미지)를 통해 opencv 기능을 확인해보도록 하겠습니다.

일단 특정경로에 이미지파일을 옮겨 놓고 그 파일을 열어보는 예제를 만들어 보겠습니다.

C:₩img₩codrone.jpg

```
import cv2 as cv
img = cv.imread('C:₩img₩codrone.jpg')
#경로에 있는 이미지를 불러오고 뒤에 0을 입력하면 흑백이 됩니다.
print(img.shape)  #이미지의 사이즈가 반환됨
cv.imshow('window_title',img)  #이미지를 실행시킴
```

이렇게 코딩을 하고 실행을 시키면 다음과 같이 원하는 경로의 이미지가 불러와집니다.

3. 이미지 사진에서 원하는것 검출하기

코드론 미니 흑백 처리

이번에는 흑백처리를 하여 이미지를 불러와보겠습니다. 가끔 환경에 따라 흑백처리를 하면 더욱 사물의 경계나 모양이 뚜렷하게 나올 수 있어 더 쉽게 인식할 수 있습니다.

C:₩img₩codrone.jpg

```
import cv2 as cv
img = cv.imread('C:₩img₩codrone.jpg')
print(img.shape)
img_gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
# 이미지를 흑백으로 바꾸어줍니다.
cv.imshow('window_title',img_gray)
```

이렇게 하면 흑백 이미지로 실행이 됩니다.

3. 이미지 사진에서 원하는것 검출하기

여러 드론 중에 코드론 미니 인식시키기

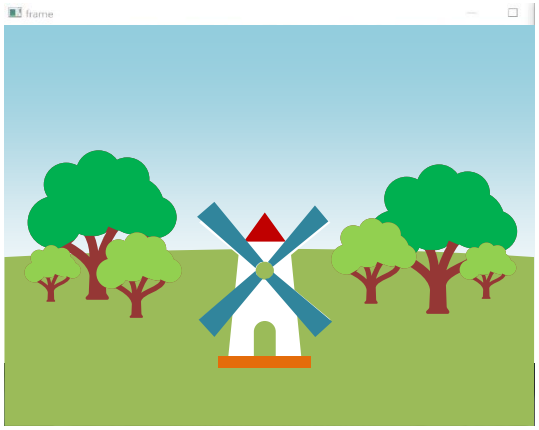
지금까지는 단순히 이미지를 불러오는 예제였습니다. 이번에는 여러 가지 객체가 나오는 사진에서 원하는 사진만 인식하는 본격적인 예제를 만들어 보겠습니다.


```
import cv2 as cv
img = cv.imread(r'C:₩img₩drones.jpg',0)
# 여러사진이 있는 사진을 흑백으로 불러옵니다.
a = cv.imread(r'C:₩img₩codrone.jpg',0)
b = cv.imread(r'C:₩img₩drones.jpg') #컬러
result = cv.matchTemplate(img, a, cv.TM_SQDIFF)
# 원하는 이미지를 특정화
minVal, maxVal, minLoc, maxLoc = cv.minMaxLoc(result)
# 원하는 이미지의 위치 설정
x, y = minLoc
h,w = a.shape
b = cv.rectangle(b, (x, y), (x + w, y + h), (0,0,255), 2)
# 원하는 이미지에 사각형 표시
cv.imshow("result",b)
cv.waitKey(0)
cv.destroyAllWindows()
```

4. 사람 얼굴 인식과 키보드 드론 제어

카메라 켜고 흑백으로 만들기

영상인식하면 가장 떠오르는 것이 바로 얼굴 인식일 것입니다. 이번에는 pc의 웹캠 이나 노트북에 내장 되어있는 카메라를 통해 얼굴인식을 하며 프로그램하는 방법을 알아보도록 하겠습니다.

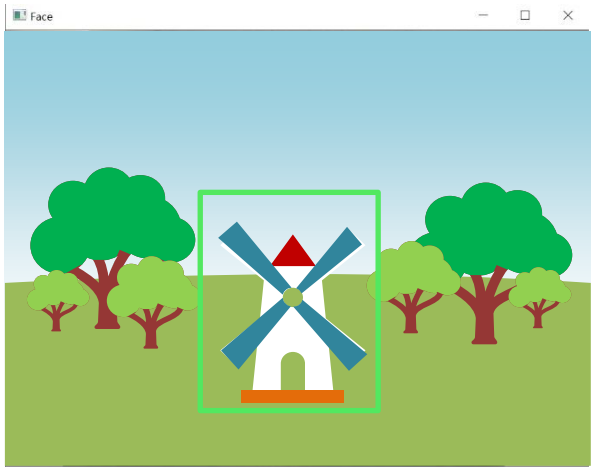



```
import numpy as np
import cv2
cap = cv2.VideoCapture(0)
#0은 cam의 id로 여러개 있으면 0,1,2로 고유id가 생깁니다.
cap.set(3,640) # set Width # 영상 가로 넓이 설정
cap.set(4,480) # set Height # 영상 세로 높이 설정
while(True):
#while(cap.isOpened()):
ret, frame = cap.read()
frame = cv2.flip(frame, 1)
# Flip camera vertically #영상 상하반전유무에 따라 0,1을 사용
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY) #영상처리를 위한 흑백처리
cv2.imshow('frame', frame)
cv2.imshow('gray', gray)
k = cv2.waitKey(30) & 0xff
if k == 27: # press 'ESC' to quit
break
cap.release()
cv2.destroyAllWindows()
```

4. 사람 얼굴 인식과 키보드 드론 제어

사람 얼굴 인식

카메라를 구동하는 방법을 배웠으니 이제 실제 카메라에서 사람의 얼굴을 인식하는 방법을 알아보도록 하겠습니다.

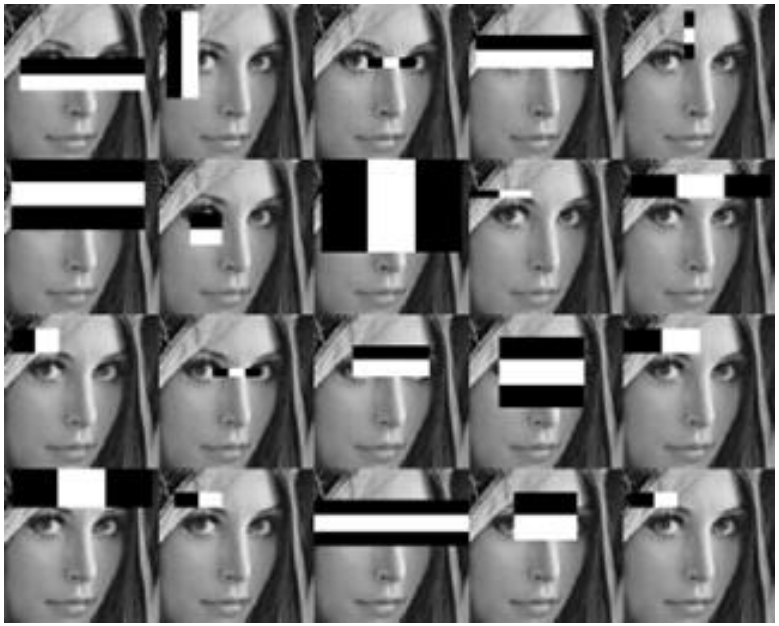


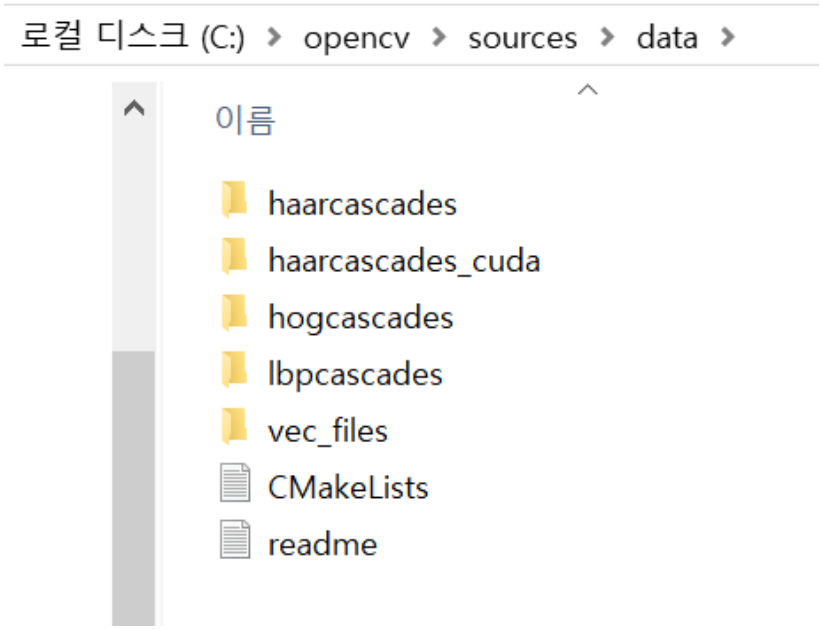

```
import cv2
CAM_ID = 0
#Open the CAM
cap = cv2.VideoCapture(CAM_ID) #카메라 생성
#create the window & change the window size
#윈도우 생성 및 사이즈 변경
cv2.namedWindow('Face')
face_cascade = cv2.CascadeClassifier()
# opencv의 인식모듈을 사용하기위해 사용해야하는 라이브러리
face_cascade.load(r'C:₩opencv₩sources₩data₩haarcascades₩haarcascade_frontalface_default.xml')
# opencv 설치후 해당 xml의 경로를 자신의 컴퓨터에 맞도록 확인해야합니다.
while(True):
#read the camera image
#카메라에서 이미지 얻기
ret, frame = cap.read()
grayframe = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
grayframe = cv2.equalizeHist(grayframe)
#회색으로 좀더 얼굴 인식 구별 확인 / 얼굴 인식이 안되면 0 , 되면 1
faces = face_cascade.detectMultiScale(grayframe, 1.1, 3, 0, (30, 30))
if faces is():
print("0")
else:
print("1")
#얼굴에 사각형을 표시
for (x,y,w,h) in faces:
cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),3, 4, 0)
cv2.imshow('Face',frame)
#10ms 동안 키입력 대기
if cv2.waitKey(10) >= 0:
break;
#close the window
#윈도우 종료
cap.release()
cv2.destroyWindow('Face')
```

4. 사람 얼굴 인식과 키보드 드론 제어

haarcascades <여기서 잠깐>

Opencv에서 오브젝트를 인식하는 대표적인 방법 중의 하나는 Haarcascade 라이브러리 모듈을 사용하는 방법입니다.
Haarcascade 라이브러리를 사용하여 인식할 수 있는 오브젝트는 사람의 얼굴의 특징점 , 얼굴 안의 눈, 동물 얼굴, 사람의 몸 각 부분들, 컬러 및 차량을 포함하며, 흑백 현상 사진 현상에서 얻을 수 있는 이미지들로부터 각 특징들을 구분할 수 있도록 되어있습니다.
주의할 점은 파이썬 코딩시에 해당 경로를 잘 확인하고 적어야 동작됩니다.





4. 사람 얼굴 인식과 키보드 드론 제어

드론 키보드 조종

조종기가 아닌 키보드의 키를 통해 드론이 조종되는 것을 알아보도록 하겠습니다.

import keyboard를 이용하면 키보드의 입력기능을 사용할 수 있습니다.

숫자 1,0으로 이착륙제어를 W,S 키를 드론의 Throttle을 상하를 조절하고 키보드의 방향키 ▲▼◀▶를 이용하여 드론의 전,후,좌,우를 제어하도록 하겠습니다.

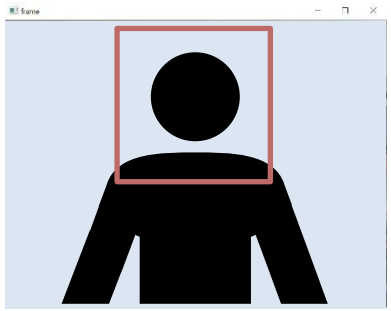
```
from time import sleep
from e_drone.drone import *
from e_drone.protocol import *
import keyboard #키보드 입력을 위한 keyboard 모듈 호출
drone = Drone()
drone.open()
#키보드 입력을 통해 드론을 제어하는 루틴입니다.
#1: 이륙, 0: 착륙, w: 상승, s: 하강, 화살표 ▲ :전진, ▼: 후진, ▶: 우이동 ◀:좌이동
while(True):
if keyboard.is_pressed("1"):
print("TakeOff")
drone.sendTakeOff()
sleep(0.01)
drone.sendControlWhile(0, 0, 0, 0, 4000)
elif keyboard.is_pressed("0"):
print("Landing")
drone.sendLanding()
sleep(0.01)
elif keyboard.is_pressed("W"):
print("Up")
drone.sendControl(0, 0, 0, 50)
elif keyboard.is_pressed("S"):
print("Down")
drone.sendControl(0, 0, 0, -50)
elif keyboard.is_pressed("Up"):
print("Forward")
drone.sendControl(0, 50, 0, 0)
elif keyboard.is_pressed("Down"):
print("Backward")
drone.sendControl(0, -50, 0, 0)
elif keyboard.is_pressed("Left"):
print("Left")
drone.sendControl(-50, 0, 0, 0)
elif keyboard.is_pressed("Right"):
print("Right")
drone.sendControl(50, 0, 0, 0)
elif keyboard.is_pressed("Space"):
print("Space")
drone.sendControl(0, 0, 0, 0)
```

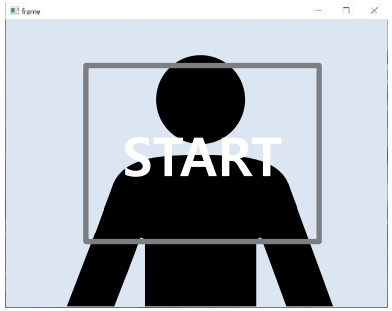
5. 손 제스쳐로 드론 조종하기

손 제스쳐로 드론 조종하기

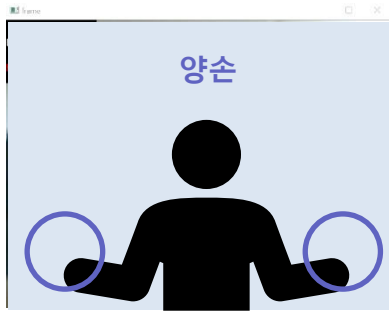
이제 최종 관문입니다. 얼굴인식과 제스쳐인식을 통해 드론을 움직이는 예제입니다. 일단 이전에 배웠던 얼굴인식과 키보드로 드론을 조종하는 예제를 포함하여 얼굴을 인식하면 드론이 뜨면서 시작되고 드론을 제대로 된 위치에 놓기위해 키보드 제어부분을 포함시켰습니다.

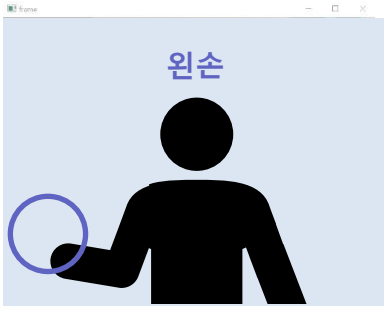
실행을 하면 처음에 빨간 네모가 나타납니다. 그안에 얼굴을 위치하면 드론이 Take off 되면서 start 되게 되고 사람의 몸통가운데가 기본적으로 인식되며, 두팔의 날개짓을 하면 그 모션을 원으로 표시 되게됩니다.
양쪽손을 파닥거리면 드론이 위로 올라가며, 오른손만 파닥거리면 왼쪽으로, 왼손만 파닥거리면 오른쪽으로 드론이 날아가게됩니다.
이상한곳으로 가게되면 키보드의 w,s 키로 상하 , 키보드 상하좌우키로 드론을 전후좌우로 이동 시키면됩니다.





1) 카메라를 켜고 빨간 네모안에 얼굴을 인식하도록 대기
2) 네모안에 얼굴이 인식되면 드론이 이륙하면서 Start 메시지 생성
3) 몸통 가운데를 기준으로 인식
4) 양쪽손의 모션움직임을 통해 드론 제어





5. 손 제스쳐로 드론 조종하기

손 제스쳐로 드론 조종하기

실습 코드는 크게 다음과 같이 9개 부분으로 되어있고, 각 해당 구문에 주석이 포함되어 있습니다.
앞서 opencv 예제들과 더불어 그동안 배워왔던 것들을 상기하면서 따라와주시기 바랍니다.

-> 다운로드 (txt)(클릭)

 # 1) 얼굴을 인식하게 하는 haarcascades 기능 사용

 # 2) 카메라영상의 특정 위치에 4각형을 만들고 그안에서 얼굴이 인식되는지 확인하는 face함수

 # 3) 화면에 표시되는 도형의 색상과 표시할 글씨 폰트 크기 변수 값 설정

 # 4) 드론의 움직임에 관련한 모든 변수값 설정

 # 5) 실제 정해진 사각형안에 얼굴을 인식하는 함수부분

 # 6) 이전 프레임과 비교하여 변화한 좌우 차이 값을 반환합니다.

 # 7) 현재 감지된 상태와 사용된 변수들을 화면에 출력합니다.

 # 8) 실제 메인 구동 함수 : 얼굴을 인식하고 드론이 take off 되면서 카메라영상의 모션을

인식하여 드론을 제어하는 루틴

 # 9) 이상한곳으로 날라가는 것을 방지하기 위한 드론 키보드 제어 부분

5. 손 제스쳐로 드론 조종하기

1) 얼굴을 인식하게 하는 haarcascades 기능 사용

```
from e_drone.drone import *
from e_drone.protocol import *
from time import sleep
import cv2
import keyboard
# 1) 얼굴을 인식하게 하는 haarcascades 기능 사용
face_detector = cv2.CascadeClassifier()
face_detector.load(r'C:₩opencv₩sources₩data₩haarcascades₩haarcascade_frontalface_default.xml')
```

5. 손 제스쳐로 드론 조종하기

2) 카메라영상의 특정 위치에 4각형을 만들고 그안에서 얼굴이 인식되는지 확인하는 face함수

```
def face(image, window, ROI, W, H):
# 관심영역 : Region Of Interesting (ROI))
# roi = image[H1 :H2, W1 : W2] H1 좌표부터 H2 좌표의 높이까지,
# W1 좌표부터 W2 좌표까지의 폭을 잘라냅니다.
roi = image[H - ROI :H + ROI, W - ROI:W + ROI] # 이미지를 원하는 위치의 크기로 자르기
# 잘라낼 높이 좌표 : H - ROI부터 H + ROI 값만큼의 높이(H를 기준으로 ROI 값만큼의 범위)
# 잘라낼 폭의 좌표 : W - ROI부터 W + ROI 값만큼의 폭(W를 기준으로 ROI 값만큼의 범위)
faces = face_detector.detectMultiScale(roi, 1.3, 5, 10) # 얼굴을 인식합니다.
if(len(faces) == 0): return False # 인식된 얼굴이 없다면 False 를 반환합니다.
else: return True # 인식된 얼굴이 있다면 True 를 반환합니다.
```

5. 손 제스쳐로 드론 조종하기

3) 화면에 표시되는 도형의 색상과 표시할 글씨 폰트 크기 변수 값 설정

```
point_color = (0,0,255)  #  (B,G,R) 붉은색으로 색상 설정
font_big = 0.5 # 폰트 크기
simple_color = (255,255,255)   # (B,G,R) 흰색으로 색상 설정
font_small = 0.5 # 폰트 크기

# 특정 위치에 4각형을 만들고 변화된 이미지의 수치들을 표시하는 함수
    def putTextonFrame(frame, differ_ = 0, sum_ = 0, THRE_DIFFER = 0, THRE_SUM = 0):
    font = cv2.FONT_HERSHEY_DUPLEX # 사용할 폰트 선택
    h,w,_ = frame.shape  # 이미지의 높이와 폭의 좌표를 가져옵니다.
    w = int(w/15)  # 폭의 길이를 기준으로 15로 나눕니다.
    cv2.rectangle(frame, (0,0), (150, 2 * w + 20), (0,0,0), -1)
                # cv2.rectangle(img, start, end, color, thickness)
                # img – 그림을 그릴 이미지(ex; (frame))
                # start – 시작 좌표(ex; (0,0))
                # end – 종료 좌표(ex; (150, 2 * w + 20))
                # color – BGR형태의 Color(ex; 0,0, 0) -> black)
                # thickness (int) – 선의 두께. pixel ( -1 이면 안쪽을 채움)

    differ_ = abs(differ_)  # 왼쪽과 오른쪽의 이미지 차이를 절대 값으로 나타냅니다.
    # 변화된 좌우이미지 차이가 설정된 변화량의 크기보다 크다면 폰트를 변화하여 표시합니다.
```

5. 손 제스쳐로 드론 조종하기

3) 화면에 표시되는 도형의 색상과 표시할 글씨 폰트 크기 변수 값 설정

```
if(THRE_SUM < sum_ and THRE_DIFFER < differ_):
        color_ = point_color
        font_size = font_big
    else:
        color_ = simple_color
        font_size = font_small

    if(THRE_DIFFER < 1): THRE_DIFFER = 1
# DIFFER 파라미터 값을 화면에 표시합니다. (이전 프레임에서 변화된 좌우 이미지의 차이)
 cv2.putText(frame, "DIFFER : "  + "{:3d}".format(int(differ_/THRE_DIFFER * 100),2) + "%", (0, 1*w), font, font_size, color_, 1, cv2.LINE_AA)

# cv2.putText(img, text, org, font, fontSacle, color, thickness, lineType)
# img – image (ex; (frame))
# text – 표시할 문자열 (ex; ("DIFFER : "  + "{:3d}".format(int(differ_/THRE_DIFFER * 100),2) + "%"))
# org – 문자열이 표시될 위치. 문자열의 좌측 하단  (ex; (0, 1*w))
# font – 폰트 타입 CV2.FONT_XXX
# fontSacle – 폰트 크기
# color – 폰트 색상
# thickness – 폰트 두께 (ex; (1))
# lineType – 선 종류 (default cv.Line_8) (ex; (cv2.LINE_AA))
# 변화된 좌우이미지의 합이 변화량이 설정된 크기보다 크다면 폰트를 변화하여 표시합니다.
```

5. 손 제스쳐로 드론 조종하기

3) 화면에 표시되는 도형의 색상과 표시할 글씨 폰트 크기 변수 값 설정

```
if(THRE_SUM < sum_):
        color_ = point_color
        font_size = font_big
    else:
        color_ = simple_color
        font_size = font_small

# POWER  파라미터 값을 화면에 표시합니다. (이전 프레임에서 변화된 좌우이미지의 합)
    cv2.putText(frame, "POWER : " + "{:3d}".format(int(sum_/THRE_SUM * 100)) + "%", (0, 2*w), font , font_size, color_, 1, cv2.LINE_AA)

# cv2.putText(img, text, org, font, fontSacle, color, thickness, lineType)
# img – image (ex; (frame))
# text – 표시할 문자열 (ex; ("POWER : " + "{:3d}".format(int(sum_/THRE_SUM * 100)) + "%""))
# org – 문자열이 표시될 위치. 문자열의 좌측 하단  (ex; (0, 1*w))
# font – 폰트 타입 CV2.FONT_XXX
# fontSacle – 폰트 크기
# color – 폰트 색상
# thickness – 폰트 두께 (ex; (1))
# lineType – 선 종류 (default cv.Line_8) (ex; (cv2.LINE_AA))
```

5. 손 제스쳐로 드론 조종하기

4) 드론의 움직임에 관련한 모든 변수값 설정

```
class FlappyBird:
    def __init__(self):

        # 고정 파라미터
        self.ROI = 100  # 잘라낼 이미지의 관심영역 크기 변수
        self.resize_num = 0.3 # 이미지 처리를 위해 이미지를 축소할 때 사용할 비율
        self.DIFFER_PERCENT = 0.7 # 변화된 이미지의 좌우 차이의 기준 값을 만들어내기 위한 변수
        self.MOVE = 3000  # 변화된 이미지의 좌우 합의 기준 값을 확인하기 위한 변수

        # drone motor
        # 이 값이 실제 드론에 전달됩니다.
        self.PITCH = 0 # 드론을 앞뒤 이동할 때의 값
        self.ROLL = 30 # 드론을 좌우 이동할 때의 값
        self.THRO_UP = 80 # 드론을 상승 시킬 때의 값
        self.THRO_DOWN = -30  # 드론을 하강시킬 때의 값
        self.LOW_THRO_UP = 20  # 드론이 움직일 때 고도를 보정해주는 상승 값

        #self.drone = Drone()
        self.cap = cv2.VideoCapture(0)
        self.frame_name = "frame“ # 종료 상태 표시
        self.quit = 0 # 드론의 현재 움직임 명령 상태
```

5. 손 제스쳐로 드론 조종하기

4) 드론의 움직임에 관련한 모든 변수값 설정

```
self.now_roll = 0 # 현재 roll에 입력된 값을 저장하기 위한 변수
self.now_pitch = 0 # 현재 pitch에 입력된 값을 저장하기 위한 변수
self.now_yaw = 0 # 현재 yaw 에 입력된 값을 저장하기 위한 변수
self.now_throttle = 0 # 현재 throttle에 입력된 값을 저장하기 위한 변수

# 프레임 측정과 좌우 변화 상태를 기록
self.frame_cnt = 0 # 일정 프레임마다 이미지처리를 하도록 현재 프레임을 세는 변수
self.right_area = 0 # 화면에서 나누어진 우측영역
self.left_area = 0 # 화면에서 나누어진 좌측영역
```

5. 손 제스쳐로 드론 조종하기

5) 실제 정해진 사각형안에 얼굴을 인식하는 함수부분

```
def face_detect(self, frame):
h, w, _ = frame.shape # 이미지의 높이와 폭의 좌표를 가져옵니다.
w_half = int(w / 2)              # 320 : 폭을 기준으로하여 절반으로 줄인 값을 저장
h_half = int(h / 2)              # 240 : 높이를 기준으로 하여 절반으로 줄인 값을 저장
h_quar = int(h / 4)              # 120 : 높이를 기준으로 하여 1/4로 줄인 값을 저장
font = cv2.FONT_HERSHEY_DUPLEX # 사용할 폰트를 저장 #설정된 관심영역이 너무 크게 되어있는 경우에 메시지 출력
if (h_quar < self.ROI or w_half < self.ROI): print("ROI IS TOO BIG!")
# self.ROI = 100 이므로, 높이와 너비는 100보다 작으면 안됨
#얼굴이 인식되지 않을동안 반복대기하는 부분
# face 함수를 사용하여 얼굴이 인식되었는지 확인합니다.
# (감지가 되지 않았다면 감지가 될 때까지 반복합니다.) 감지가 되면 while 반복문을 빠져나갑니다.
while (not face(frame, self.frame_name, self.ROI, w_half, h_quar)):
cv2.rectangle(frame, (w_half - self.ROI, h_quar - self.ROI), (w_half + self.ROI, h_quar + self.ROI), (100, 100, 200), 8)
# self.ROI 만큼의 사각형을 그립니다.
# cv2.rectangle(img, start, end, color, thickness)
                # img – 그림을 그릴 이미지(ex; (frame))
                # start – 시작 좌표(ex; (w_half - self.ROI, h_quar - self.ROI))
                # end – 종료 좌표(ex; (w_half + self.ROI, h_quar + self.ROI))
                # color – BGR형태의 Color(ex; 100, 100, 200))
                # thickness (int) – 선의 두께. pixel (ex;  8 )
cv2.imshow(self.frame_name, frame)   # 화면에 이미지를 표시
cv2.waitKey(1) # 키보드 입력 확인(1ms 대기)
ret, frame = self.cap.read() # 카메라에서 화면을 가져옵니다.
frame = cv2.flip(frame, 1) # 가져온 화면을 반전시킵니다. #반복문 빠져나감
# 얼굴이 인식된 후 화면에 사각형을 그리고 Start라는 문구를 표시하는 부분
# Show the starting image :
cv2.rectangle(frame, (int(w_half - w_half/2), int(h_half - w_half/2)), (int(w_half + w_half/2), int(h_half + w_half/2)), (150, 150, 150), 10)
# 화면에 사각형을 그립니다. w_half 와 h_half 값에 계산식을 넣어서 원하는 크기의 사각형을 만듭니다.
cv2.putText(frame, "START ", (w_half - 135, h_half + 20) , font , 3, (255, 255, 255), 3)
# 중앙을 기준으로 (- 135, + 20) 위치에 "START" 글씨를 표시합니다.
cv2.imshow(self.frame_name, frame) # 화면에 이미지를 표시
cv2.waitKey(1) # 키보드 입력 확인(1ms 대기)
```

5. 손 제스쳐로 드론 조종하기

6) 이전 프레임과 비교하여 변화한 좌우 차이 값을 반환합니다.

```
def divide_screen(self, frame, pre_frame): #이미지의 사이즈를 변경합니다. resize_num 변수의 비율만큼 변화
    frame = cv2.resize(frame, None, fx=self.resize_num, fy=self.resize_num)
    # 이미지의 색상을 흑백으로 변화
    now_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # 두 프레임 사이의 다른 부분 절대값 계산
    frame = cv2.absdiff(pre_frame, now_frame)
    # 임계 값을 설정하여 설정한 값 이상인 부분은 검은색(255)으로 표시
    frame = cv2.threshold(frame, 40, 255, cv2.THRESH_BINARY)[1]
    # 이미지의 x와 y 좌표를 가져옵니다.
    h, w = frame.shape
    # 이미지의 오른쪽만을 잘라옵니다.
    right_frame = frame[:, int(w * 0.55):]
    # 이미지의 왼쪽만 잘라옵니다.
    left_frame = frame[:, 0:int(w * 0.45)]
     # 오른쪽 이미지 픽셀에서 0이 아닌 값을 가져옵니다. (오른쪽 변화한 값의 수치)
    right_num = cv2.countNonZero(right_frame)
    # 왼쪽 이미지 픽셀에서 0이 아닌 값을 가져옵니다.
    left_num = cv2.countNonZero(left_frame)
    ee1 = cv2.getTickCount()  # 디버깅을 위한 시간 측정
    # 흑백으로 변환된 현재 이미지와 오른쪽과 왼쪽 오른쪽의 변화한 이미지 값을 반환
    return now_frame, right_num, left_num
```

5. 손 제스쳐로 드론 조종하기

7) 현재 감지된 상태와 사용된 변수들을 화면에 출력합니다.

```
def draw_for_debug(self, frame, sum, differ): # 이미지 좌우반전
frame = cv2.flip(frame, 1) # 이미지의 x와 y 좌표를 가져옵니다.
h, w, _ = frame.shape # right # 드론에게 오른쪽으로 이동 명령을 내렸다면 실행
if (self.now_roll < 0): #화면에 명령에 해당하는 도형을 표시합니다. #cv2.rectangle(frame, (w - 10, 0), (w-1, h), (200, 200, 200), 5)
cv2.circle(frame, (int(w/2 - 240) , int(h/2)), 60, (200, 100, 100), 5) # left # 드론에게 왼쪽으로 이동 명령을 내렸다면 실행

elif (self.now_roll > 0):
#화면에 명령에 해당하는 도형을 표시합니다. #cv2.rectangle(frame, (1, 0), (10, h), (200, 200, 200), 5)
cv2.circle(frame, (int(w/2 + 240) , int(h/2)), 60, (200, 100, 100), 5) # up # 드론에게 상승 명령을 내렸다면 실행
elif (self.now_throttle > self.LOW_THRO_UP):
#화면에 명령에 해당하는 도형을 표시합니다.
#cv2.rectangle(frame, (0, 1), (w, 10), (200, 200, 200), 5)
cv2.circle(frame, (int(w/2 - 240) , int(h/2)), 60, (200, 100, 100), 5)
cv2.circle(frame, (int(w/2 + 240) , int(h/2)), 60, (200, 100, 100), 5)  # down # 드론에게 하강명령을 내렸다면 실행
elif (self.now_throttle == self.THRO_DOWN): #화면에 명령에 해당하는 도형을 표시합니다.
#cv2.rectangle(frame, (0, w - 10), (w-1, h), (200, 200, 200), 5)
cv2.circle(frame, (int(w/2) , int(h/2 + 160)), 60, (200, 100, 100), 5) # 화면에 측정 값과 상태를 표시합니다.
# putTextonFrame()함수를 실행하여 화면에 변화된 좌우 차이값과 합을 표시합니다.
putTextonFrame(frame=frame, differ_=differ, sum_=sum,
THRE_DIFFER=self.DIFFER_PERCENT * sum,
THRE_SUM=self.MOVE)
return frame
```

5. 손 제스쳐로 드론 조종하기

8) 실제 메인 구동 함수 : 얼굴을 인식하고 드론이 take off 되면서 카메라영상의 모션을 인식하여 드론을 제어하는 루틴

```
def run(self, port_name=None, drone_name=None):
# 시작시 얼굴 인식을 사용할지 설정하는 변수로 1로 설정하면 얼굴이 인식되어야만 드론 조종이 작동하며, 0으로 설정하면 얼굴을 인식하지 않아도 바로 조종이 가능합니다.
face_detection = 1    # 종료 변수 값이 0이 아니라면 반복
while (self.quit is 0): # 일정한 간격으로 작동하도록 현재 시간을 측정
e1 = cv2.getTickCount() # 드론의 통신 포트를 열기

drone = Drone()
drone.open()

### video capture and save to previous frame


        if (self.cap.isOpened()): # 정상적으로 video 동작
            ret, frame = self.cap.read() # 카메라 화면 가져오기
            frame = cv2.flip(frame, 1) # 이미지의 좌우 반전 # 이미지의 사이즈를 변경합니다.  resize_num 변수의 비율만큼 변화
            pre_frame = cv2.resize(frame, None, fx=self.resize_num, fy=self.resize_num)  # 이미지의 색상을 흑백으로 변화 (이미지 처리와 비교를 쉽게 하기 위해)
            pre_frame = cv2.cvtColor(pre_frame, cv2.COLOR_BGR2GRAY)
        else:  # 정상적으로 video 동작하지 않을때
            print("video is not opened!") # 화면에 보일 창을 설정합니다.
        cv2.namedWindow(self.frame_name, cv2.WND_PROP_FULLSCREEN)
        cv2.setWindowProperty(self.frame_name,cv2.WND_PROP_VISIBLE,
        cv2.WINDOW_FULLSCREEN)
        cv2.imshow(self.frame_name, frame)
        cv2.waitKey(1)

        ### Start with face detection : 얼굴이 특정 사각형안에 인식되는 루틴

print("Start with face detection!") # face_detection 변수 값이 1이라면 얼굴을 인식해야만 드론이 작동합니다.
        if (face_detection == 1):
            self.face_detect(frame)

        ### Start with takeoff  : 얼굴이 인식되면 자동으로 드론 이륙 명령

        drone.sendTakeOff()
        sleep(0.1)
        drone.sendTakeOff()
        sleep(0.1)
        drone.sendTakeOff()
        sleep(0.1)
        drone.sendControlWhile(0, 0, 0, 0, 4000)

        print("Control!!") # 카메라가 켜있고, 종료 변수 값이 0이 아니라면 반복
```

5. 손 제스쳐로 드론 조종하기

8) 실제 메인 구동 함수 : 얼굴을 인식하고 드론이 take off 되면서 카메라영상의 모션을 인식하여 드론을 제어하는 루틴

```
while (self.cap.isOpened() and self.quit is 0):
    ret, frame = self.cap.read()  # 카메라 화면 가져오기
    self.frame_cnt += 1 # 프레임 변수 증가 # 흑백으로 변환된 현재이미지와 오른쪽과 왼쪽의 변화한 이미지 값을 가져오기
    pre_frame, r, l = self.divide_screen(frame, pre_frame)

        self.right_area += r  # 오른쪽 화면의 변화 값 변수( self.right_area)에 측정된 r을 더합니다.
        self.left_area += l # 왼쪽 화면의 변화 값 변수(self.right_area)에 측정된 l을 더합니다.


        if (self.frame_cnt == 4): # 5프레임이 될 때마다 명령을 보냅니다.

        sum = self.right_area + self.left_area # 합 값에 왼쪽과 오른쪽의 화면 변화 값을 더하여 입력
        differ = self.left_area - self.right_area # 차이 값에 왼쪽에서 오른쪽의 화면 변화 값을 빼서 입력


        self.frame_cnt, self.right_area, self.left_area = 0, 0, 0 # 프레임을 초기화
        roll_flag = 1  # 좌우 이동을 사용할지 설정하는 변수 #print("Find face!!")

        DIFFER_THRESH = self.DIFFER_PERCENT * sum # 합 값에 self.DIFFER_PERCENT 만큼을 곱하여 차이의 기준 값으로 사용합니다


        if (roll_flag and differ > DIFFER_THRESH and sum > self.MOVE / 2):
        print("right")
        #drone.sendControl(self.ROLL, 0, 0, self.LOW_THRO_UP)
        self.now_roll = self.ROLL
        self.now_pitch = 0
        self.now_yaw = 0
        self.now_throttle = self.LOW_THRO_UP

                                                                    # right
                                                                    # go left : 왼쪽 손의 움직임만 커지면 드론은 오른쪽으로 이동
                                                                    # 움직임의 차이 값이 기준 값보다 크면 왼쪽의 감지 값이 더 큰 걸로 인식
                                                                    # 움직임의 합 값이 설정된 elf.MOVE 값의 절반 이상보다 크다면 작동


        elif (roll_flag and differ < -DIFFER_THRESH and sum > self.MOVE / 2):
        print("left")
        #drone.sendControl(-self.ROLL, 0, 0, self.LOW_THRO_UP)
        self.now_roll = -self.ROLL
        self.now_pitch = 0
        self.now_yaw = 0
        self.now_throttle = self.LOW_THRO_UP

                                                                    # left
                                                                    # go right : 오른쪽 손의 움직임만 커지면 드론은 왼쪽으로 이동
                                                                    # 움직임의 차이 값이 기준 값보다 작다면 오른쪽의 감지 값이 더 큰 걸로 인식
                                                                    # 움직임의 합 값이 설정된 elf.MOVE 값의 절반 이상보다 크다면 작동
```

5. 손 제스쳐로 드론 조종하기

8) 실제 메인 구동 함수 : 얼굴을 인식하고 드론이 take off 되면서 카메라영상의 모션을 인식하여 드론을 제어하는 루틴

```
                # up : 양손의 움직임이 커지면 드론을 상승하게하는 루틴
                # 움직임의 전체 합(양 손) 값이 self.MOVE에 설정한 값보다 크다면 실행
                elif (sum > self.MOVE):
                    print("up")
                    #drone.sendControl(0, self.PITCH, 0, self.THRO_UP)
                    self.now_roll = 0
                    self.now_pitch = self.PITCH
                    self.now_yaw = 0
                    self.now_throttle = self.THRO_UP

                # down : 양손의 움직임이 없으면 드론을 하강 하게하는 루틴
                else:
                    print("down")
                    #drone.sendControl(0, 0, 0, self.THRO_DOWN)
                    self.now_roll = 0
                    self.now_pitch = 0
                    self.now_yaw = 0
                    self.now_throttle = self.THRO_DOWN

                drone.sendControl(self.now_roll, self.now_pitch, self.now_yaw, self.now_throttle)
                print(self.now_roll, self.now_pitch, self.now_yaw, self.now_throttle)

                # 시간을 측정하고 프레임을 계산합니다.
                e2 = cv2.getTickCount()
                timePassed = (e2 - e1) / cv2.getTickFrequency()
                e1 = e2
                fps = 1 / timePassed
                print("fps : ", format(fps,".2f"))

                frame = self.draw_for_debug(frame, sum, differ)

                cv2.imshow(self.frame_name, frame)
                cv2.waitKey(1)

        #키보다 "q"를 누르면 드론과 동작이 멈춤

                if cv2.waitKey(1) & 0xFF == ord("q"):
                    drone.sendStop()
                    self.quit = 1
                    break
```