참가팀 선발 메뉴얼

작성자: 2023 Lockheed Martin Falcon Challenger 기술팀장 | 소프트웨어학과 추세빈

들어가기

Tello와 Python

우리가 이번 대회에서 사용할 Tello는 여러 언어로 프로그래밍이 가능하다. 하지만 2023년 Lockheed Martin Falcon Challenger 주제인 “인공지능 기법을 활용한 멀티로더(드론) 자율주행”을 위해서는 파이썬을 사용하도록 하자. 다른 언어도 사용가능하지만, 앞으로의 메뉴얼에서는 파이썬에 대한 것만 제공할 예정이다.

파이썬과 개발 환경에 대한 설명은 생략하도록 한다. 참고로 필자는 Mac OS와 Visual Studio Code를 활용한다. 만약 Python과 Visual Studio Code에 대한 학습이나 자료가 필요하다면 필자가 제작한 Python 강의안을 제공할 것이므로, 메일 남겨주길 바란다.(cntpqls0218@gmail.com)

문의

Instagram @2023falcon_challenger

총괄팀장 정성식 Email | jss8732@naver.com

기술팀장 추세빈 Email | cntpqls0218@gmail.com

환경 세팅

1) pip

Tello drone controller를 제공하는 python 패키지인 tellopy를 활용하여 tello 프로젝트를 수행할 수 있다.

패키지(package)와 라이브러리(library) 이해하기

패키지(package)란?

여러 함수가 들어있는 묶음 즉, 관련 기능과 코드를 모아놓은 묶음이다.

프로그래밍을 처음 배우면 하나의 python 파일에서 일련된 작업을 수행하는 코드만 작성한다. 하지만 본격적으로 프로그래밍 작업을 하게되면 여러 파일, 여러 기능으로 프로그램을 구성한다.

바탕화면에 여러 환경 사진을 다운 받았다고 가정하자. 산, 바다, 강 세가지 폴더에 여러 사진을 저장한 뒤, 환경 폴더에 이 세가지 폴더를 저장한다면 훨씬 효율적으로 사진들을 관리할 수 있다. 만약 폴더 없이 파일들을 바탕화면에 깔아두면 해당 사진을 찾는 데에 비교적 많은 시간이 걸리기 때문이다.

패키지는 이와 비슷한 개념이다. 폴더는 파일들을 구조적으로 관리하기 위해서 사용되는 것처럼, 패키지는 코드를 구조적으로 관리하기 위해 사용된다. 패키지 안에는 여러 모듈(.py)이 포함되어 있으며, 각 모듈은 특정한 기능을 수행하는 코드의 집합이다.

패키지는 종종 **라이브러리**라고도 불리우며, 앞으로 메뉴얼에서는 “라이브러리”라는 명칭을 사용하도록 한다.

파이썬의 다양한 라이브러리를 사용하기 위해서는 pip가 필요하다. pip는 파이썬으로 작성된 라이브러리를 관리해주는 시스템으로, 파이썬 관련 명령어에 다양하게 활용된다. MacOS 유저는 brew, 우분투/리눅스 유저들은 apt와 비슷하다고 생각하면 된다. pip는 Python 2.7.9 이후 버전, Python 3.4 이후 버전에는 내장되어있으므로 따로 설치할 필요가 없다.

- pip vs pip3

  pip: Python2 버전 패키지 매니저

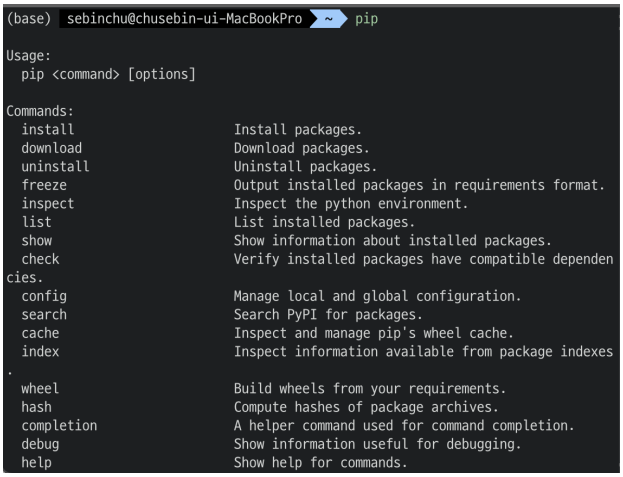
  pip3: Python3 버전 패키지 매니저

- 로컬(내 디바이스)에 pip가 설치되어있는지 확인하는 방법

  command 창(이하 cmd로 줄임)에 다음 코드 입력를 입력해보자.

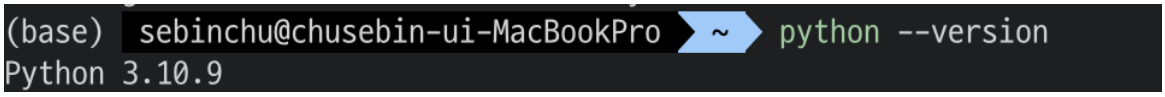
```
pip
pip3
```

아래와 유사한 결과값이 출력되면 pip는 잘 설치되어 있는 것이다. 만약 pip가 설치되어있지 않다면 Python 버전을 업그레이드 하거나, pip를 직접 설치하면 된다.



- 로컬(내 디바이스)에 python이 설치되어있는지 확인하는 방법

```
python --version
```



- pip 설치 방법
  1. 아래 명령어 입력

     ```
     curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
     ```

  2. Python 버전에 맞는 pip 설치
     - 파이썬 버전 확인 명령어는 위에 작성된 바와 같이 `python —version`을 cmd 창에 입력한다.

     ```
     python2 get-pip.py # Python 2.XX 사용자를 위한 pip 설치 명령어
     python3 get-pip.py # Python 3.XX 사용자를 위한 pip 설치 명령어
     ```

     설치가 완료되면 pip를 입력하여 잘 설치되었는지 확인하는 작업까지 잊지않고 하자.

- pip 업그레이드

  pip는 자주 업데이트되기 때문에 수시로 업데이트를 해주는 것이 좋다.

  ```
  pip install --upgrade pip
  ```

2) TelloPy

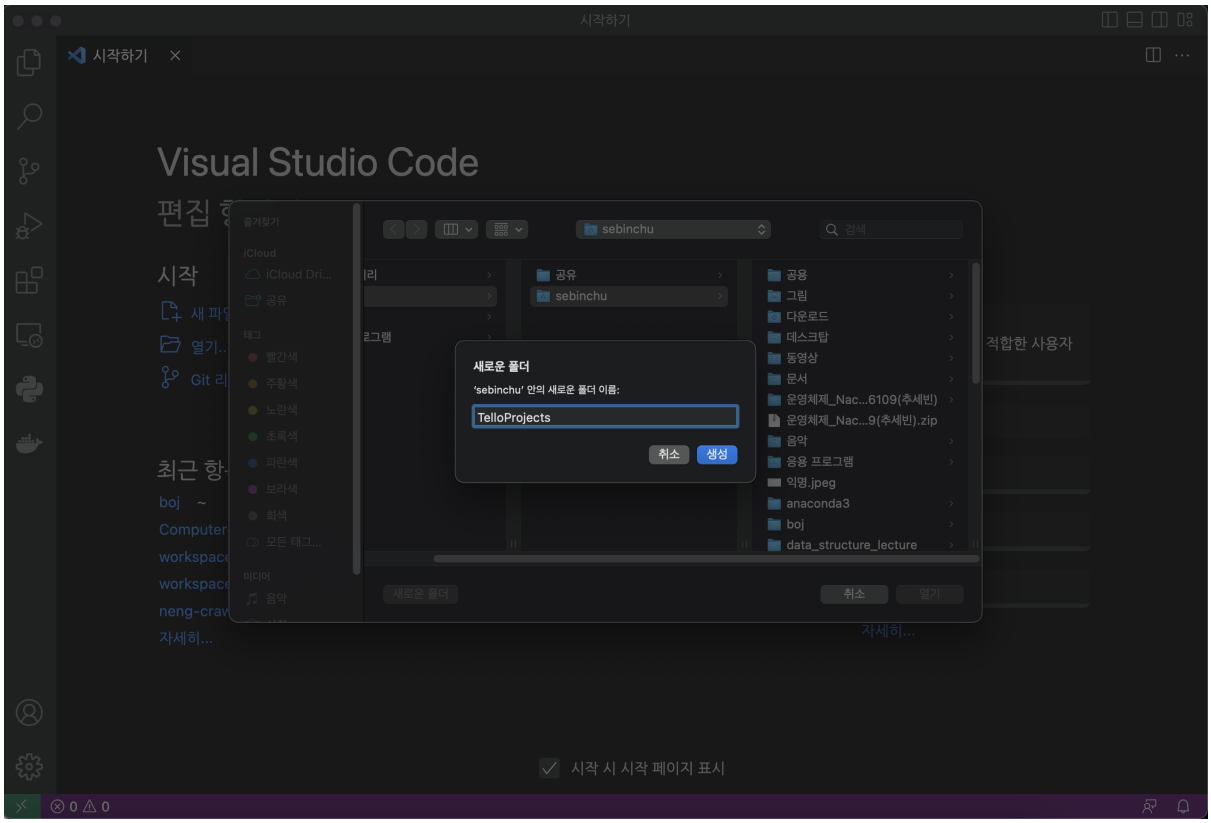
tellopy는 pip 설치가 돼 있다면 터미널에서 아래 명령을 쳐서 쉽게 받을 수 있다.

```
pip3 install tellopy
```

```
(base) sebinchu@chusebin-ui-MacBookPro ~ pip3 install tellopy
Collecting tellopy
  Downloading tellopy-0.6.0-py2.py3-none-any.whl (23 kB)
Installing collected packages: tellopy
Successfully installed tellopy-0.6.0
```

설치 뒤에는 import tellopy를 통해 패키지를 사용할 수 있다.

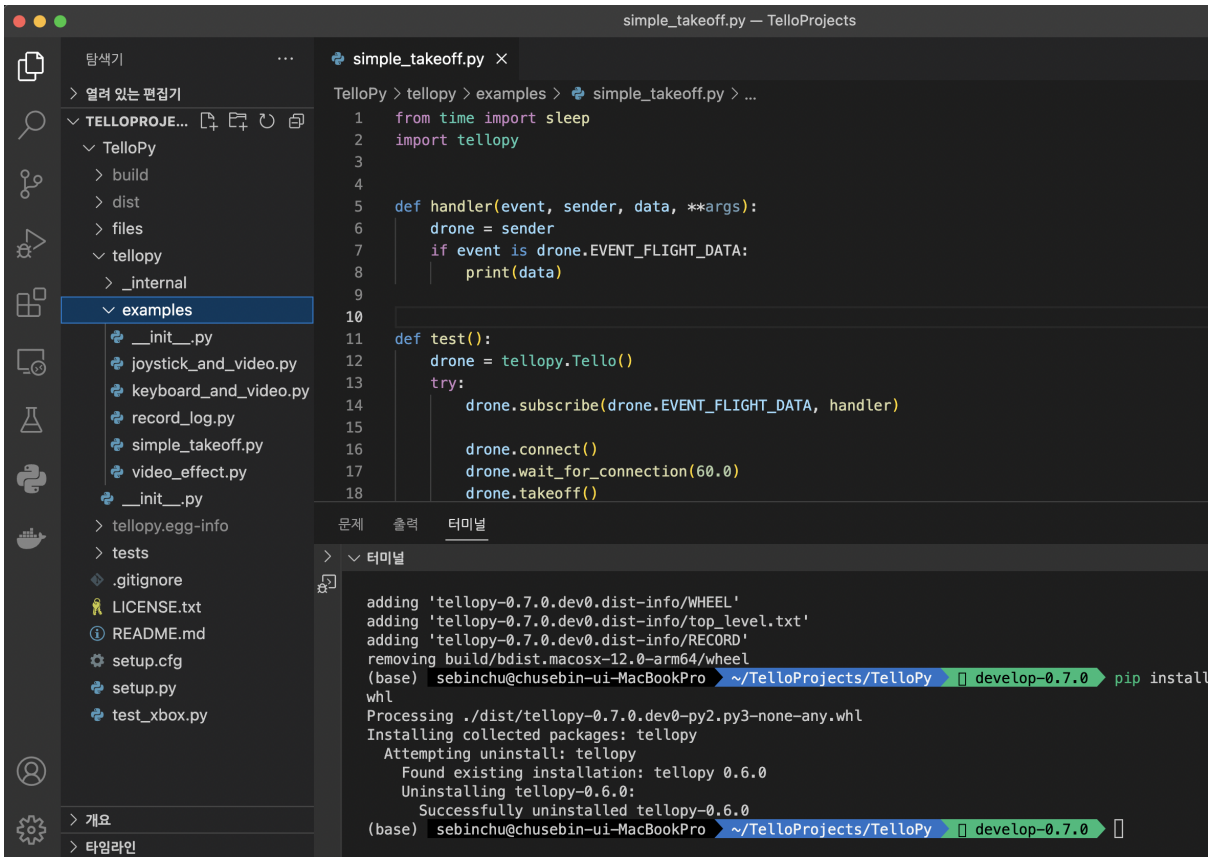
VSC에서 다음과 같이 tello 프로젝트를 진행할 폴더를 생성하자. 필자는 TelloProjects라고 이름을 지어서 폴더를 생성해주었다.



여기서 위 과정을 따라 tellopy 라이브러리를 설치하고 import tellopy를 했음에도 불구하고 라이브러리를 사용할 수 없는 경우에는 tellopy를 직접 빌드하여 설치하는 해결 방법이 있다.

```
git clone https://github.com/hanyazou/TelloPy.git
cd TelloPy
python setup.py bdist_wheel
pip install dist/tellopy-*.dev*.whl
```

위 명령어를 통해 깃허브에서 클론을 받으면 아래와 같은 파일들이 디렉토리(TelloProjects) 내부에 생성된다.



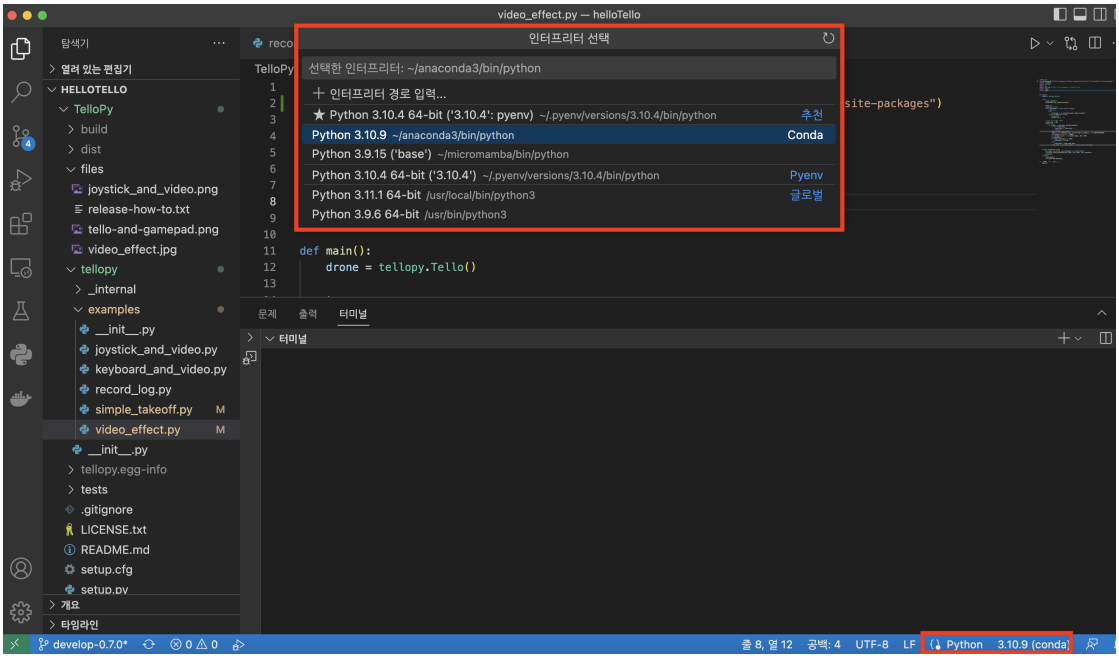
examples 디렉토리에 우리가 tello 조종을 위해 직접적으로 사용할 예제 파일들이 저장되어 있다.

- joystick_and_video.py: 조이스틱 조종기와 비디오 예제
- keyboard_and_video.py: 키보드 활용 조종기와 비디오 예제
- record_log.py: log 데이터 저장 예제
- simple_takeoff.py: 간단한 이착륙 하는 예제
- video_effect.py: tello가 촬영하고 있는 비디오를 pc에서 스트리밍 하는 예제

여기서 simple_takeoff.py와 video_effect.py에 작성된 예제들이 앞으로 우리가 사용할 Python으로 Tello를 조작하는 방법론에 대한 시발점이다. 따라서 이번 메뉴얼에서는 이 두 파일을 집중적으로 공부해보자.

파이썬 여러 버전 설치 후 사용 시 주의점

파이썬의 라이브러리는 버전 간 공유되지 않으므로 라이브러리를 설치한 파이썬 버전과 실행할 파이썬 버전을 맞춰주는 것이 중요하다. VSC 오른쪽 아래에서 내가 작업할 프로젝트의 Python 버전을 선택할 수 있으므로, 내가 설치한 라이브러리와 버전, 경로가 맞는 파이썬 버전을 선택하여 사용하자.



아래와 같이 sys 모듈을 통해 경로를 지정하는 방법도 있다.

1. tellopy 경로 찾기

```
pip show tellopy
```

2. sys를 통해 경로 지정하기

```
import sys
sys.path.append("/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages")
```

Python으로 Tello 조종하기

1) simple_takeoff.py

[간단한 이착륙 코드]

```
from time import sleep
import sys
sys.path.append("/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages")
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```

- drone = tellopy.Tello()

  tellopy 내부에 정의 되어있는 Tello를 drone이라는 객체로 선언한다.

- drone.connect

  선언한 drone 객체의 connect 메소드를 활용하여 tello에 connection을 요청한다.

- drone.takeoff

  takeoff 메소드를 통해 이륙 명령을 내린다.

- drone.land

  land 메소드를 통해 착륙 명령을 내린다.

- sleep

  몇 초동안 기다리게 하는 함수이다.

2. video_effect.py

해당 코드를 사용하기 위해서는 ac, opencv, image 패키지를 다운 받아야 한다.

[패키지 다운로드 명령어]

```
pip install av
pip install opencv-python
pip install image
python -m tellopy.examples.video_effect
```

- av: python으로 오디오 및 비디오 파일 CRUD를 위한 라이브러리이다.
  - open: 비디오 or 비디오 파일 열기
  - av 라이브러리를 사용하기 위해서는 FFmpeg 설치 필수

- opencv: 이미지 및 비디오 처리, 컴퓨터 비전 작업, 기계 학습 등에 사용되는 오픈 소스 컴퓨터 비전 라이브러리이다. 실시간 컴퓨터 비전 애플리케이션 개발에 널리 활용된다.

[Tello-비디오 스트림 출력 코드]

```
import sys
import traceback
import tellopy
import av
import cv2  # for avoidance of pylint error
import numpy
import time


def main():
    drone = tellopy.Tello()

    try:
        drone.connect()
        drone.wait_for_connection(60.0)

        retry = 3
        container = None
        while container is None and 0 < retry:
            retry -= 1
            try:
                container = av.open(drone.get_video_stream())
            except av.AVError as ave:
                print(ave)
                print('retry...')

        # skip first 300 frames
        frame_skip = 300
        while True:
            for frame in container.decode(video=0):
                if 0 < frame_skip:
                    frame_skip = frame_skip - 1
                    continue
                start_time = time.time()
                image = cv2.cvtColor(numpy.array(frame.to_image()), cv2.COLOR_RGB2BGR)
                cv2.imshow('Original', image)
                cv2.imshow('Canny', cv2.Canny(image, 100, 200))
                cv2.waitKey(1)
                if frame.time_base < 1.0/60:
                    time_base = 1.0/60
                else:
                    time_base = frame.time_base
                frame_skip = int((time.time() - start_time)/time_base)
```

```
    except Exception as ex:
        exc_type, exc_value, exc_traceback = sys.exc_info()
        traceback.print_exception(exc_type, exc_value, exc_traceback)
        print(ex)
    finally:
        drone.quit()
        cv2.destroyAllWindows()

if __name__ == '__main__':
    main()
```

- container = av.open(drone.get_video_stream()) : 비디오 스트림을 저장하기 위한 AVContainer 변수를 선언한다. av.open 메소드를 통해 해당 파일을 연다.
- #skip first 300 frames 코드를 살펴보자.

```
# skip first 300 frames
        frame_skip = 300
        while True:
            for frame in container.decode(video=0):
                if 0 < frame_skip:
                    frame_skip = frame_skip - 1
                    continue
```

  frame_skip은 비디오 프레임을 건너뛸 횟수를 나타내는 변수이다. 이 변수를 300으로 초기화하여 처음 300개의 프레임을 건너뛴다. 초기 스트림의 시작 부분에는 보통 초기화되는 노이즈 또는 불안정한 프레임이 포함될 수 있으므로 시작 300 부분을 건너뛰고, 정상적인 비디오 프레임부터 처리하기 위해 처리해주는 코드이다.

- 비디오 처리를 위한 무한 루프

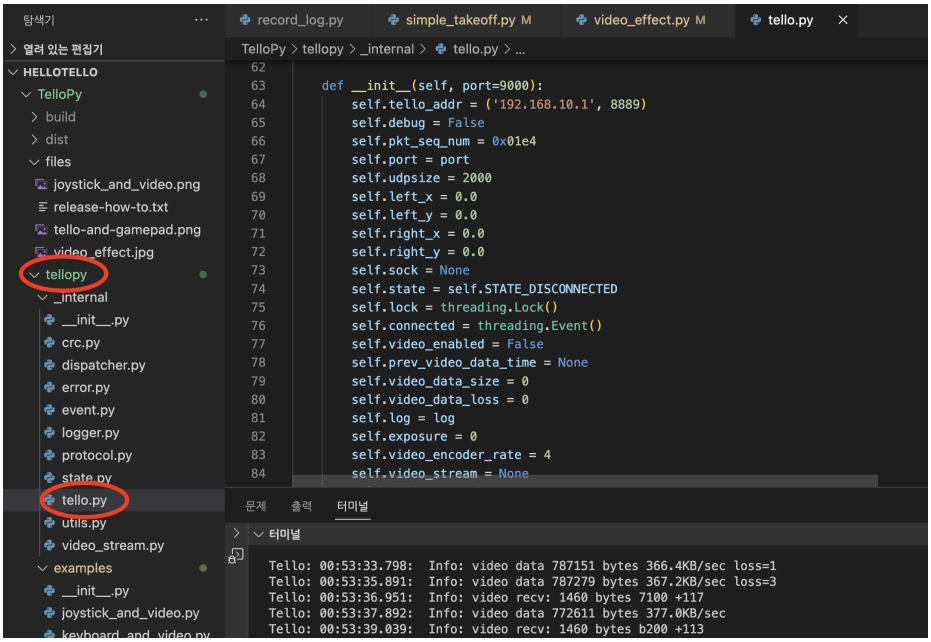
  위 코드는 while 루프 내에서 OpenCV를 사용하여 비디오 프레임을 처리하고 화면에 표시하는 방식으로 영상 프레임을 조작한다. 비디오는 정적 이미지의 연속된 파일이라고 생각하면 되기에 정적 이미지를 연속적으로 처리하기 위해 무한 루프를 사용한다.

- image = cv2.cvtColor(numpy.array(frame.to_image()), cv2.COLOR_RGB2BGR): Tello에서 받아온 프레임을 이미지로 변환한 후, 이미지를 Numpy(파이썬 라이브러리) 배열로 변환한다. 이후 cv2.cvtColort 메소드를 사용하여 이미지의 컬러 채널 순서를 RGB → BGR로 변경하는 역할을 한다.

  numpy.array로 변환한 이미지는 기본적으로 RGB 컬러 순서를 가지고 있지만, OpenCV에서는 기본적으로 BGR 컬러 순서를 사용하기 때문에 cv2.COLOR_RGB2BGR로 컬러 채널 순서를 변경해준다.

- cv2.imshow('Original', image): 받아온 이미지를 화면에 띄우는 메소드이다. Original은 영상을 띄울 창의 이름이고 image는 띄울 이미지(ndarray)이다.

다양한 Tello Method



환경세팅에서 clone한 내용 중 tello.py를 살펴보면 다양한 tello 관련 함수들이 정의되어 있다. 해당 파일에 정의된 함수들은 이번 대회에서도 자주 사용될 주요 함수들이 포함되어 있으니, 주의깊게 살펴보길 바란다.

OpenCV

이미지 및 비디오 처리, 컴퓨터 비전 작업, 기계 학습 등에 사용되는 오픈 소스 컴퓨터 비전 라이브러리이다. 실시간 컴퓨터 비전 애플리케이션 개발에 널리 활용된다.

파이썬에서는 opencv 라이브러리를 다운하고(pip) cv2를 import하여 사용할 수 있다.

이미지 픽셀 데이터는 스칼라가 아닌 벡터이므로 기본적으로 (Height * Width * Channel)의 3차원 행렬로 표현된다.

- Channel?

  Channel은 이미지의 컬러 채널 정보를 나타낸다. 컬러 이미지의 경우, 보통 Red(빨강), Green(초록), Blue(파랑) 세 가지 채널을 가진다. 이러한 컬러 채널을 RGB 채널이라고도 한다.

OpenCV Method

- cv2.imread(fileName, flag): 이미지 읽기
- cv2.imshow(title, image): 이미지 화면에 출력하기
- cv2.imwrite(fileName, image): 이미지 저장
- cv2.destroyAllWindows() : 화면의 모든 창을 닫기
- GaussianBlur : 가우시안 필터링 혹은 블러링 함수

- Sobel , Canny : 대표적인 에지 검출 알고리즘 함수 (위 예시 코드에서도 Canny를 통해 에지를 검출하는 내용이 구현되어있다.)
- Sobel , Canny : 대표적인 에지 검출 알고리즘 함수
- cvtColor: 색공간 변환 함수
- Threshold: 이진화 수행 함수
  Sobel , Canny : 대표적인 에지 검출 알고리즘 함수이다.

Opencv에 함수는 상당히 많고 각 예제에 대해 공식 홈 또는 구글에 많은 사람들이 수행한 경우가 많기 때문에 찾아보면서 응용하면 될 것이다.