Python Programming with ENVITASK

Intro

본 문서에서는 ENVITASK 함수를 활용한 Python 프로그래밍 실습을 진행하고자 합니다.

ENVITASK

ENVI 내 대부분의 기능은 ENVITASK 함수를 통해 IDL에서 제공되며, 사용자는 ENVITASK 함수를 활용하여 일련의 자동화된 처리를 수행할 수 있습니다. 이때, ENVITASK는 ENVI와 동일한 형태의 Workflow 형식을 가지며, [ENVI 기능 선택]-[입력 파일 설정]-[파라미터 설정]-[출력 경로 및 파일명 설정] 순으로 이루어집니다.

IDL-Python Bridge

IDL-Python Bridge 기능은 IDL 8.5 버전부터 제공되고 있습니다. 이는 IDL에서 Python의 기능을 불러오거나 (IDL to Python), Python에서 IDL의 기능을 불러오도록 (Python to IDL) 합니다. IDL-Python Bridge에 대한 자세한 설명은 IDL 도움말 또는 아래의 링크에서 확인할 수 있습니다.

⇒ http://blog.daum.net/swrush/560

Tutorial

본 실습에서는 ENVITASK 함수 및 IDL-Python Bridge를 활용하여 Python에서 일련의 자동화된 처리를 수행하고자 하며, 1) Landsat-8 데이터 입력 2) Radiometric Calibration 3) QUAC 순으로 이루어집니다.

Step 1. Setting

첫번째 과정에서는 패키지 호출, ENVI 실행 및 입출력 경로 지정을 수행합니다.

```
from idlpy import IDL
import glob

e = IDL.envi(HEADLESS = 1)

root_dir = "C:\\Users\\USER\\Downloads\\"
out_dir = root_dir + "Result\\"
```

코드 내 각 과정에 대한 설명은 다음과 같습니다.

1) 패키지호출

IDL 및 glob 패키지를 호출합니다. 이때, glob 패키지를 통해 root 경로 하부 내 Landsat-8 메타데이터 파일인 *_MTL.txt 파일을 검색할 수 있습니다.

2) ENVI 실행

HEADLESS를 통해 백그라운드로 ENVI를 실행합니다.

3) 입출력 경로 지정

Step 2. Main body

다음 과정은 전처리에 대한 Main body를 구현하는 것입니다. 이를 위해, class로 l8_envitask_preprocess 객체를 정의하였고, __init__ 를 통해 입출력 경로 (input_dir, output_dir)를 클래스 변수로 지정하였습니다.

```
class l8_envitask_preprocess():
    def __init__(self, input_dir, output_dir):
        self.input_dir = root_dir
        self.output_dir = out_dir
```

그 후에는, 전처리의 각 단계를 메소드로 호출합니다.

```
def do_import(self, data):
    l8 = e.OpenRaster(data)
    MS = l8[0]
    return MS

def do_rad_cal(self, data):
    task_rad = IDL.ENVITask("RadiometricCalibration")
    task_rad.INPUT_RASTER = data
    task_rad.CALIBRATION_TYPE = 'Top-of-Atmosphere ' \
                                'Reflectance'
    task_rad.execute()
    Rad = task_rad.OUTPUT_RASTER
    return Rad

def do_atm_cal(self, data, datestamp):
    task_atm = IDL.ENVITask('QUAC')
    task_atm.INPUT_RASTER = data
    task_atm.OUTPUT_RASTER_URI = self.output_dir \
                                 + datestamp + "_BOA"
    task_atm.Execute()
    return True
```

각 과정에 대하여 살펴보면 다음과 같습니다.

1) Landsat-8 데이터 입력

OpenRaster 함수를 통해 Landsat-8 데이터를 ENVI에 입력하며, l8[0]을 통해 첫 번째 객체인 다중분광 밴드만을 MS로 입력합니다.

2) ENVITASK

① IDL.ENVITask를 통한 기능 호출
② 파라미터 선정
③ 실행 (task.execute())

3) Return

각 처리의 결과를 반환합니다.

4) 저장

Main 함수에서는 glob 패키지를 통해 입력 경로 하부에서 Landsat-8 메타데이터 파일을 검색하였고, 앞선 과정을 통해 정의된 전처리의 각 단계에 대한 메소드를 순서대로 배열하였습니다.

대기보정이 완료된 결과는 출력 경로 하부에 "촬영날짜_BOA"로 저장합니다. 이때, 촬영날짜는 datestamp의 지역변수로 선언하였습니다.

```
def main(self):
    for source in glob.glob(self.input_dir + "LC08_**"
                            + "\\**_MTL.txt"):
        datestamp = source[-31:-23]
        MS = self.do_import(source)
        Rad = self.do_rad_cal(MS)
        BOA = self.do_atm_cal(Rad, datestamp)
    print('finished')
```

각 과정에 대한 설명은 다음과 같습니다.

1) glob

입력 경로 하부 내 각 LC08_* 폴더 내 *_MTL.txt 파일을 glob 함수를 통해 검색하며, 이를 source로 정의합니다.

2) datestamp

Landsat-8 데이터 폴더에서 촬영 일자를 획득하는 과정으로, source[-31:-23]을 통해 반환된 *_MTL.txt 경로에서 string으로 읽습니다.

3) 전처리

메소드로 정의한 전처리의 각 단계를 순서대로 진행합니다. 이때, 각 단계의 입력자료는 이전 단계의 결과물로 정의합니다.

4) for문

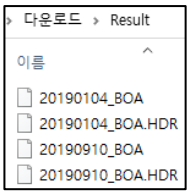
for 문을 통해 glob에서 검색된 모든 파일에 대하여 전처리를 수행합니다.

Step 4. Execute

마지막은 실행 단계로, l8_envitask_preprocess 클래스를 통해 L8 인스턴스를 생성한 후, main 함수를 호출합니다.

```
L8 = l8_envitask_preprocess(root_dir, out_dir)
L8.main()
```

모든 과정이 완료되면, 아래의 그림과 같이 결과물이 저장된 것을 확인할 수 있습니다.



아래의 그림은 앞서 기술한 각 과정에 대한 Python 코드를 통합한 것입니다. IDL-Python Bridge를 통해 Python에서 ENVI 기능을 호출하여 사용할 수 있으며, 이를 활용하면 원격탐사 자료의 자동화된 처리에 효과적으로 사용될 수 있으리라 여겨집니다.

```
from idlpy import IDL
import glob

e = IDL.envi(HEADLESS = 1)

root_dir = "C:\\Users\\USER\\Downloads\\"
out_dir = root_dir + "Result\\"

class l8_envitask_preprocess():
    def __init__(self, input_dir, output_dir):
        self.input_dir = root_dir
        self.output_dir = out_dir

    def main(self):
        for source in glob.glob(self.input_dir + "LC08_**" + "\\**_MTL.txt"):
            datestamp = source[-31:-23]
            MS = self.do_import(source)
            Rad = self.do_rad_cal(MS)
            BOA = self.do_atm_cal(Rad, datestamp)
        print('finished')

    def do_import(self, data):
        l8 = e.OpenRaster(data)
        MS = l8[0]
        return MS

    def do_rad_cal(self, data):
        task_rad = IDL.ENVITask("RadiometricCalibration")
        task_rad.INPUT_RASTER = data
        task_rad.CALIBRATION_TYPE = 'Top-of-Atmosphere Reflectance'
        task_rad.execute()
        Rad = task_rad.OUTPUT_RASTER
        return Rad

    def do_atm_cal(self, data, datestamp):
        task_atm = IDL.ENVITask('QUAC')
        task_atm.INPUT_RASTER = data
        task_atm.OUTPUT_RASTER_URI = self.output_dir + datestamp + "_BOA"
        task_atm.Execute()
        return True

L8 = l8_envitask_preprocess(root_dir, out_dir)
L8.main()
```

또한, 처리 결과를 ENVI에 표출할 경우, 아래와 같은 영상을 획득할 수 있습니다.