소프트웨어 실습 기말 프로젝트

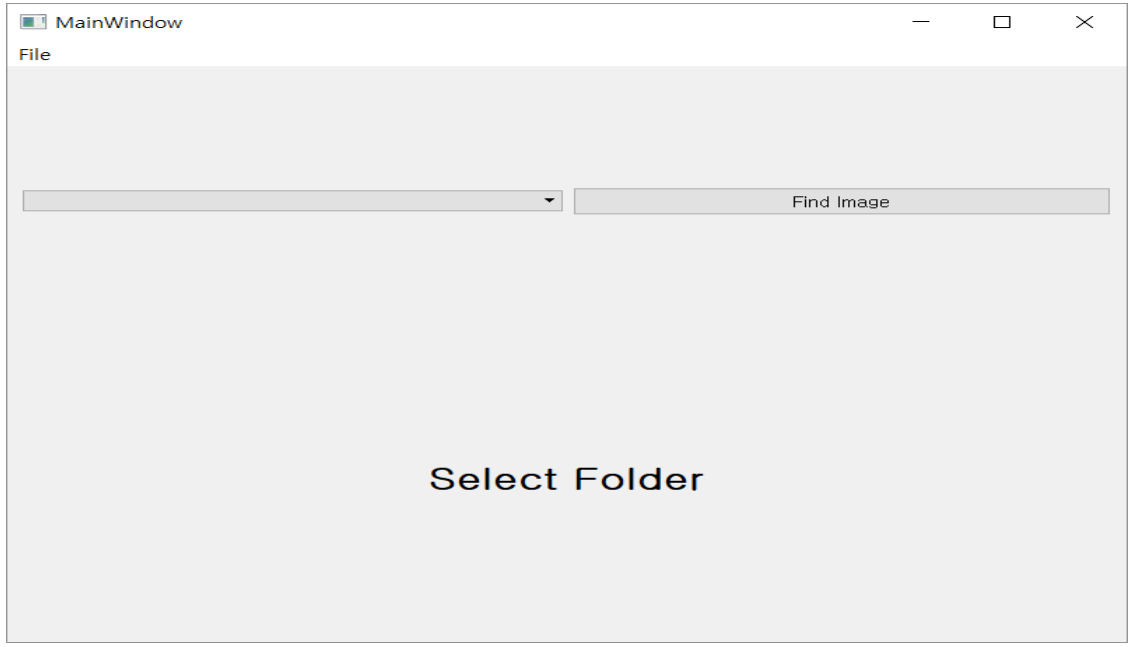
과목명	소프트웨어 실습
학과	수학교육과
학번	12142821
이름	최영효

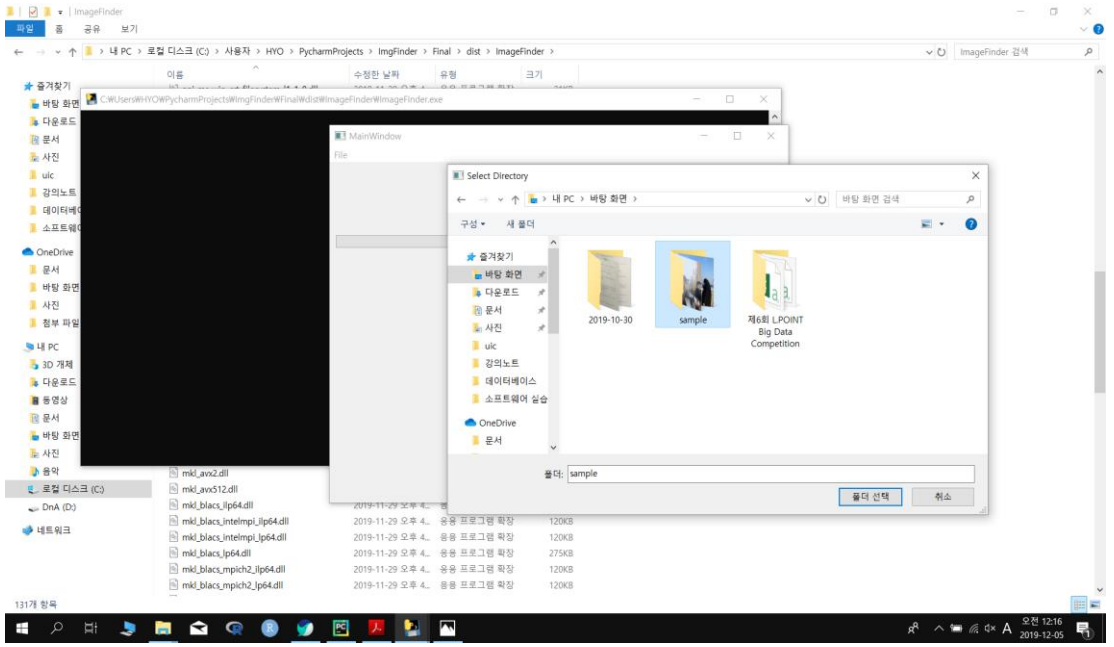
목차

1. 프로그램 소개

사진 폴더에서 비슷한 이미지를 찾고 싶을 때 유사한 이미지를 찾아 따로 저장하고, GPS 태그를 입력해주는 프로그램이다.

사용법은 먼저 File 메뉴를 통해 사진 폴더를 연다.

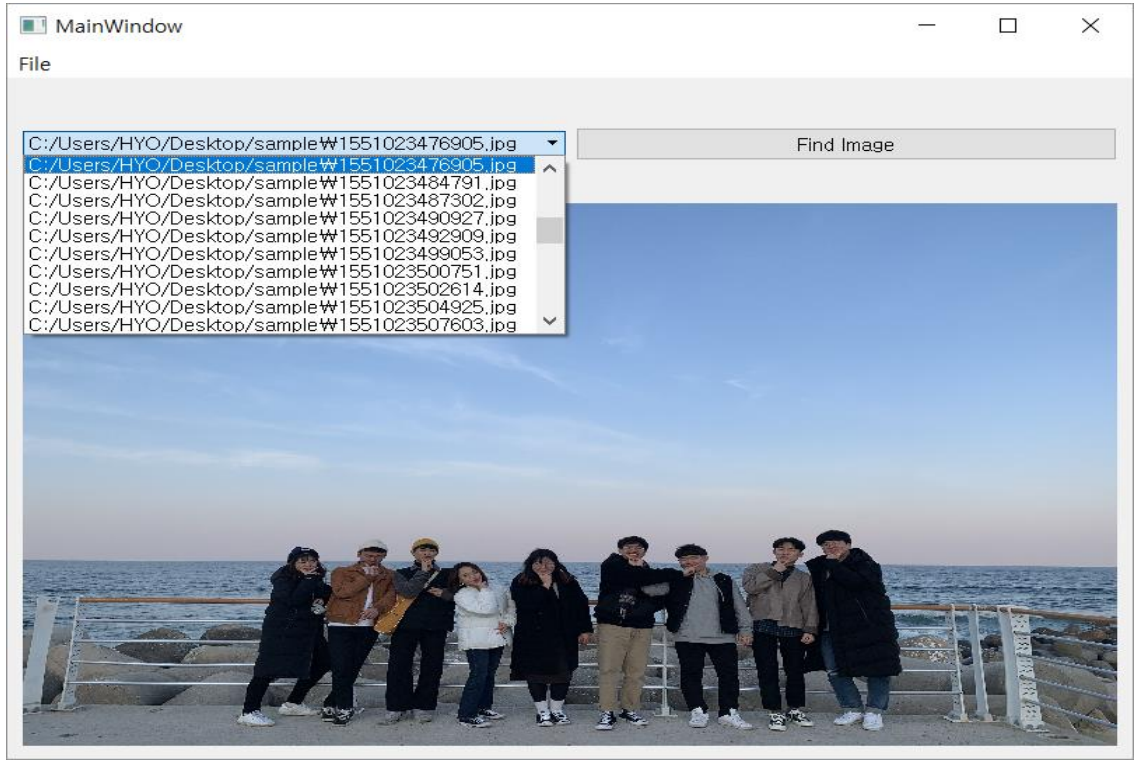


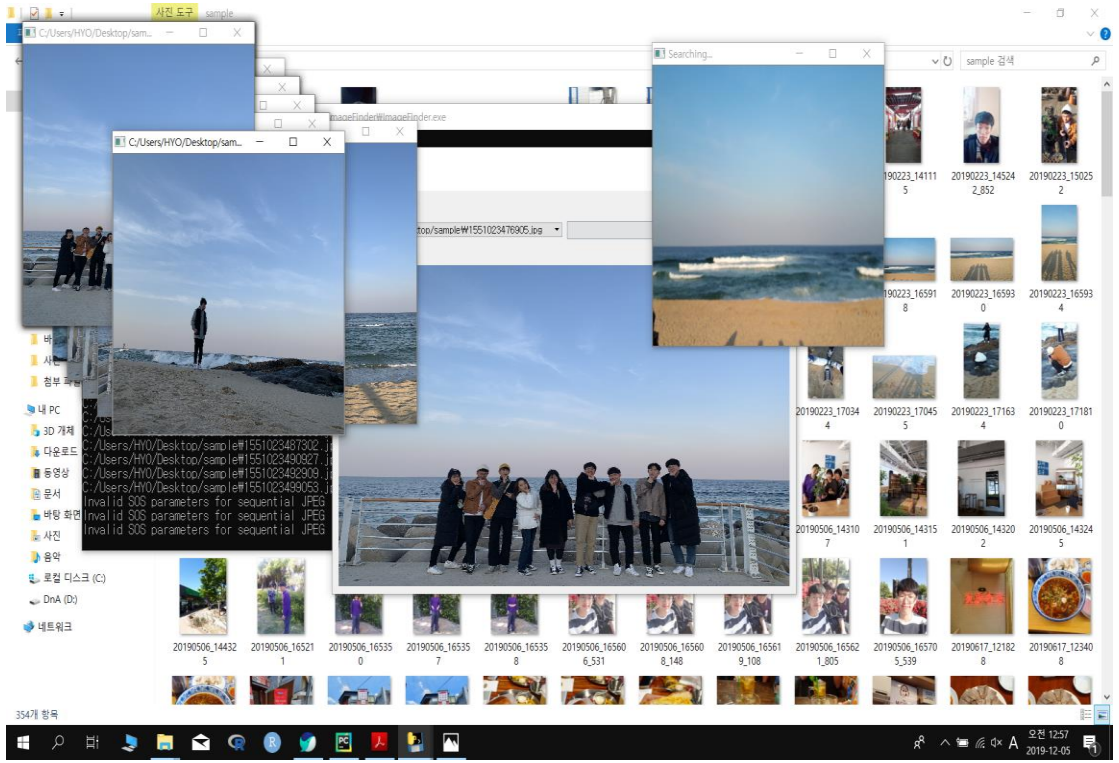


1. 프로그램 소개

2) Combo Box를 이용하여 비슷한 사진을 찾고 싶은 사진을 선택한다.

3) Find Image 버튼을 클릭하면 유사한 사진을 탐색하게 된다.

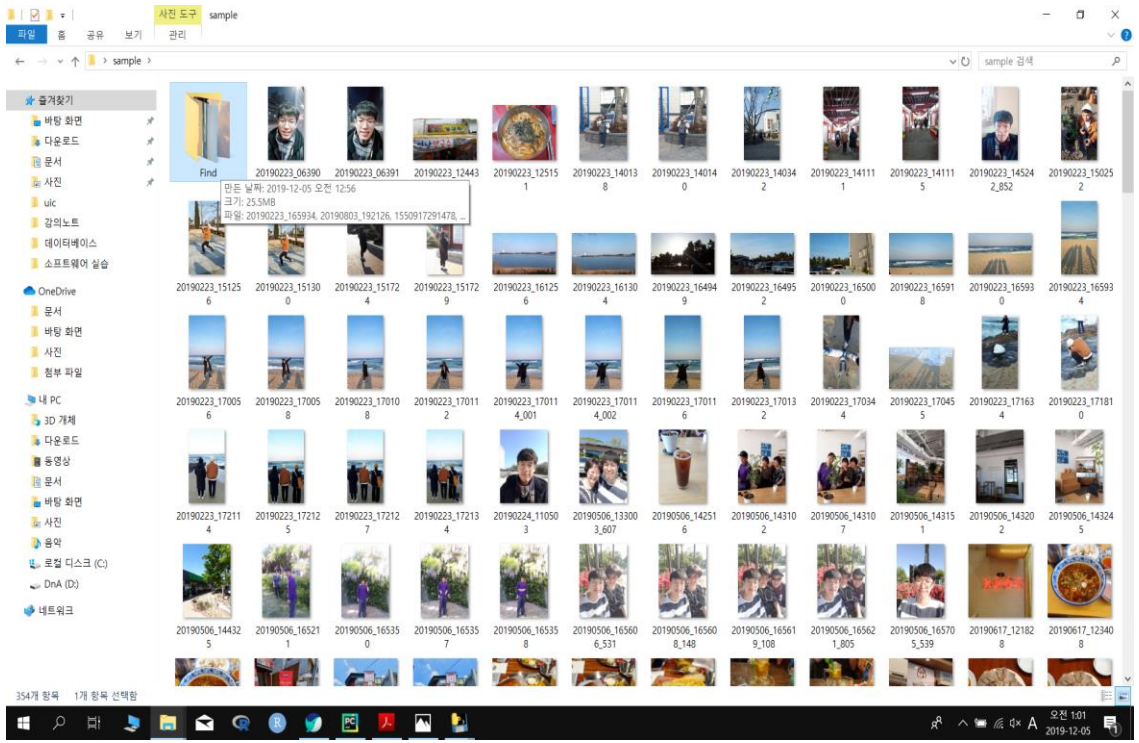


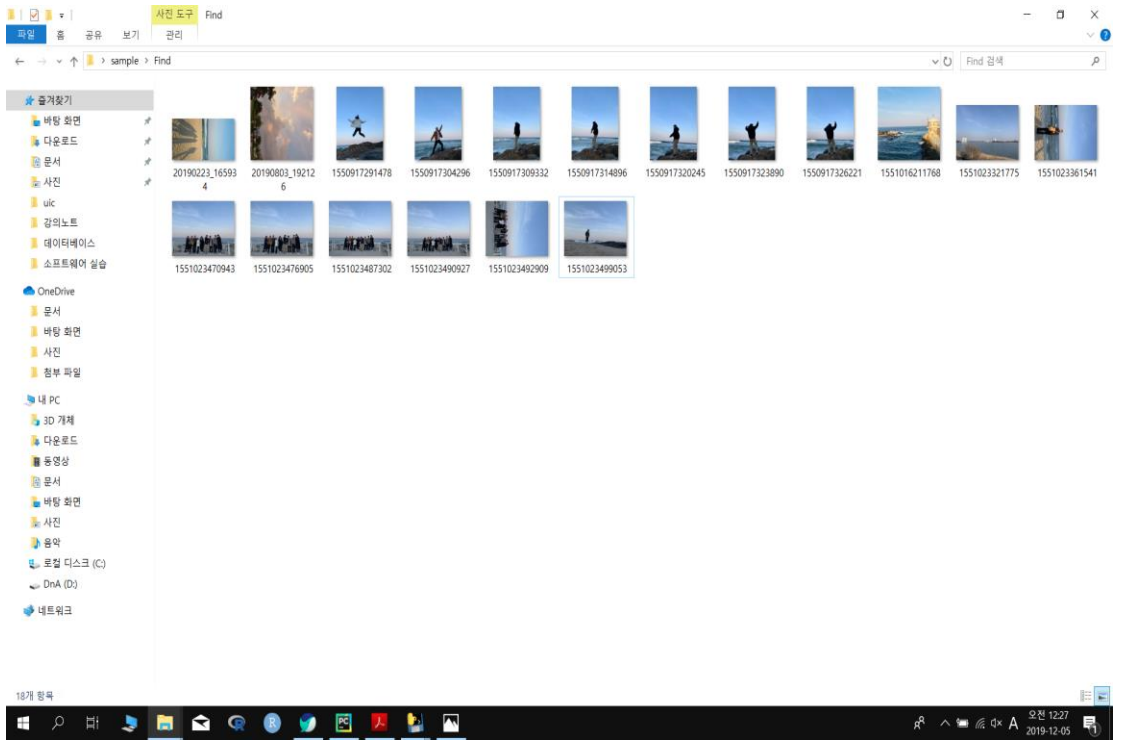


1. 프로그램 소개

4) 찾은 유사 이미지들은 다른 사진을 찾는 동안 잠시 보여준 뒤,

5) Find라는 이미지 폴더에 따로 저장된다.





2. 개발 동기

- 수업 내용에서의 이미지 관련 내용들 (이미지 태그 정보, Auto Encoder, Object Detection, …)을 많이 다뤘고,

- 스마트폰 내에서 사진 이동시 사진 정렬이 제대로 이루어지지 않는 경우가 있는데 어머니 핸드폰의 사진을 정리해드리다가 사진이 섞인 경험이 떠올라 개발을 생각하게 되었다.

- Google photos를 사진 백업에 이용하는데, GPS 태그를 이용해 장소 별로 분류해주는 기능에서 착안하여 GPS 태그 정보를 이용하기로 하였다.

3. 개발 과정

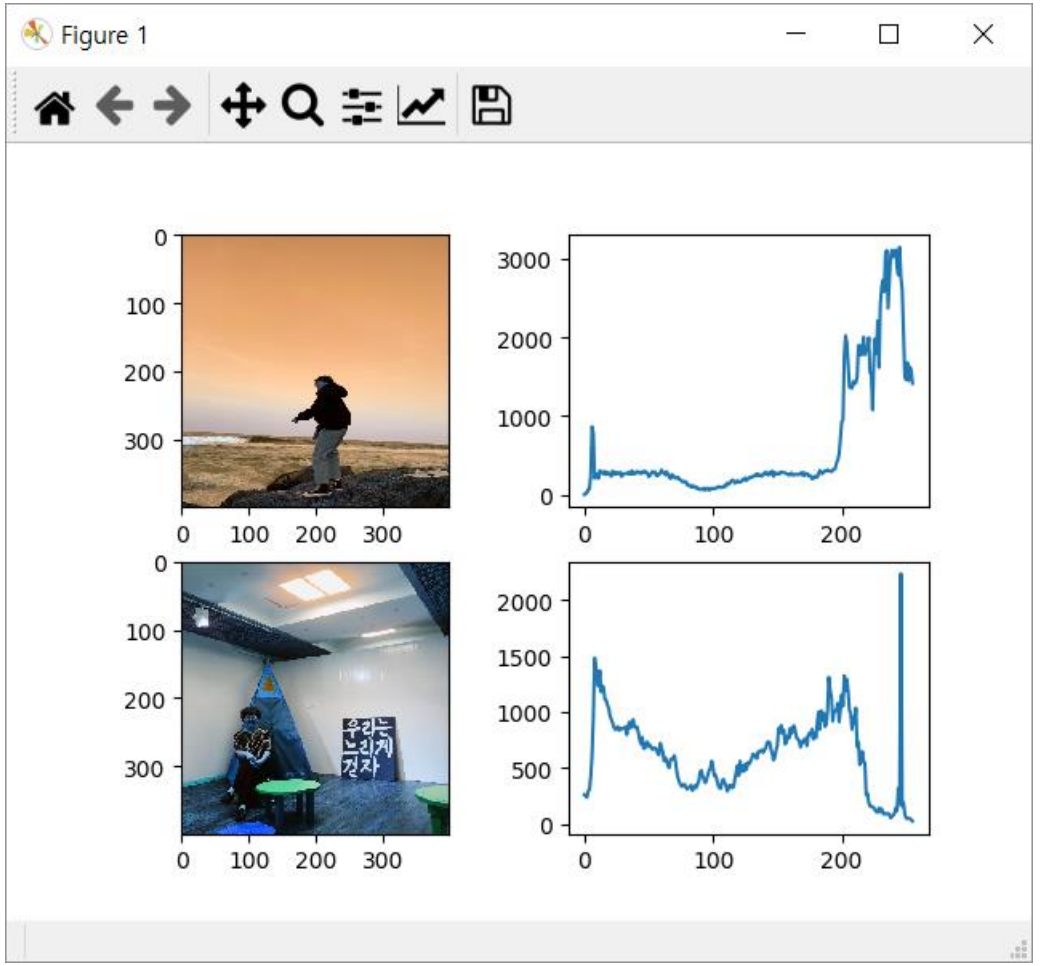
- 내부 프로그램과 GUI를 나눠 내부 프로그램을 먼저 개발하고 GUI를 개발하였다.

- 내부 프로그램은 유사 이미지를 찾는 기능과 GPS 태그 정보를 입력해주는 알고리즘을 사용하였다.

- 유사 이미지 찾는 알고리즘에는 **Image Histogram** 방법과 **Hash matching** 방법 두 가지를 시도해 보았다.

3. 개발 과정 – Image Histogram

- **Image Histogram**
  - 이미지의 밝기에 따른 픽셀 분포를 그래프로 표현한 것을 말한다.
  - 비슷한 이미지라면 비슷한 밝기 분포를 가질 것이라고 예상하고 시도해보았다.
  - 하지만 이미지 히스토그램을 이용한 결과, 유사 이미지 간의 픽셀 분포 값 비교가 잘 이루어지지 않았다.



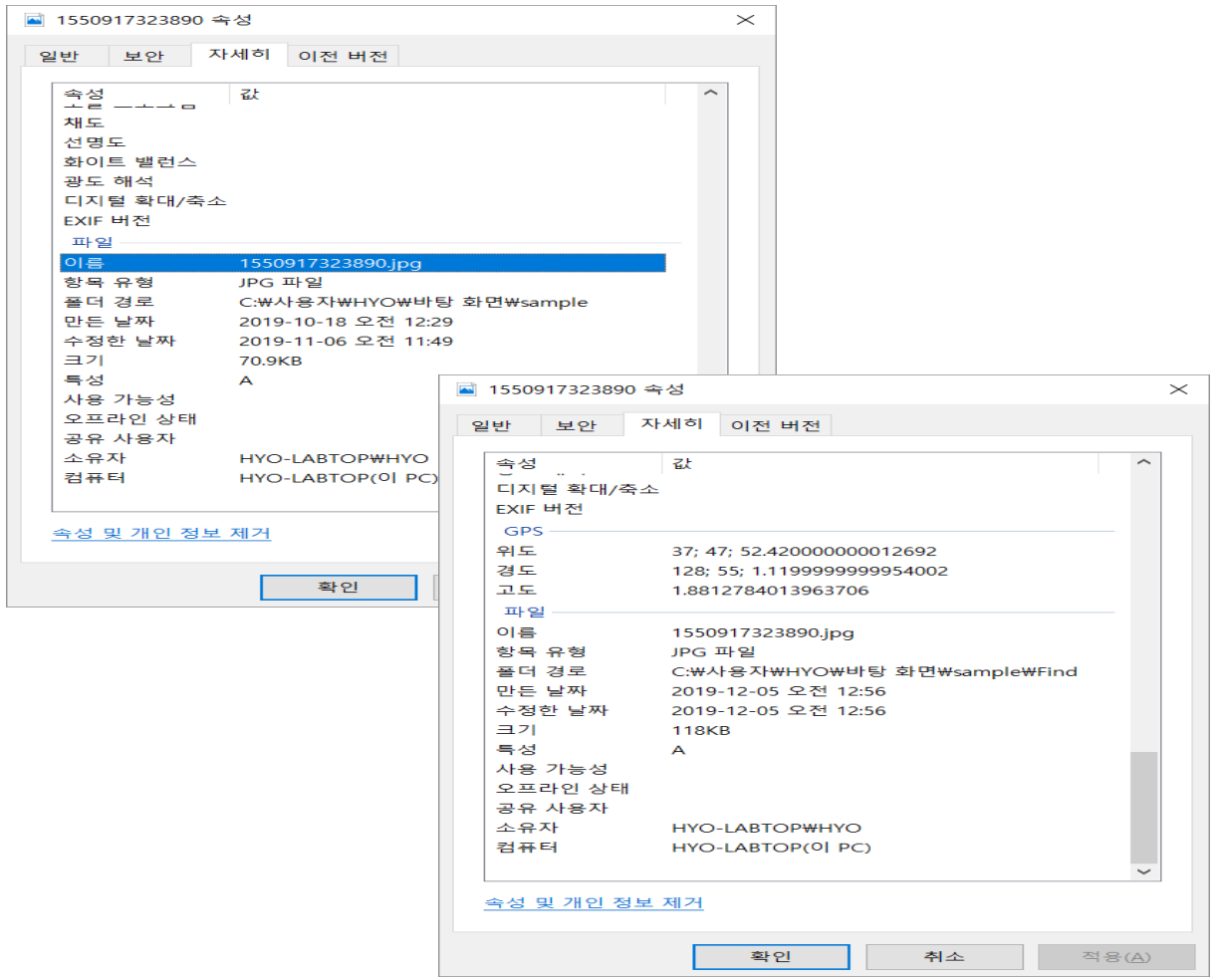
3. 개발 과정 – Hash Matching

- **Hash Matching**
  - 이미지들을 같은 사이즈로 변환한 뒤, 이미지 전체 픽셀의 평균값을 이용해 각 픽셀들을 1 또는 0 값으로 바꾼다.
  - 바꾼 뒤 두 이미지의 같은 인덱스에 있는 픽셀 값을 비교하여 결과를 리턴 하였다.
  - 이미지들의 해시 값이 80% 이상 일치할 경우 유사 이미지로 판단하였다.

```python
# (16, 16) 사이즈 binary image 변환
def img2hash(image):
    img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    img = cv2.resize(img, (16, 16))
    avg = img.mean()  # 픽셀 평균값 구하기
    bi = 1 * (img > avg)  # 픽셀 평균값보다 큰 픽셀 갯수 구하기
    return bi


def hamming_distance(a, b):
    a = a.reshape(1, -1)
    b = b.reshape(1, -1)
    # 같은 자리의 값이 서로 다른 것들의 합
    distance = (a != b).sum()
    return distance
```

3. 개발 과정 – GPS Tagging

- **GPS 태그**
  - 최신 스마트폰의 기본 카메라를 사용한 사진은 태그 정보를 잘 담고 있다.
  - 하지만 옛날 스마트폰이나 어플을 사용한 사진의 경우 태그 정보를 담고 있지 않다.
  - 정리한 사진들의 관리를 위하여 선택한 이미지의 GPS 정보를 유사 이미지들 중 GPS 정보가 없는 사진들에 입력하였다.
  - Piexif 패키지를 이용하였다.



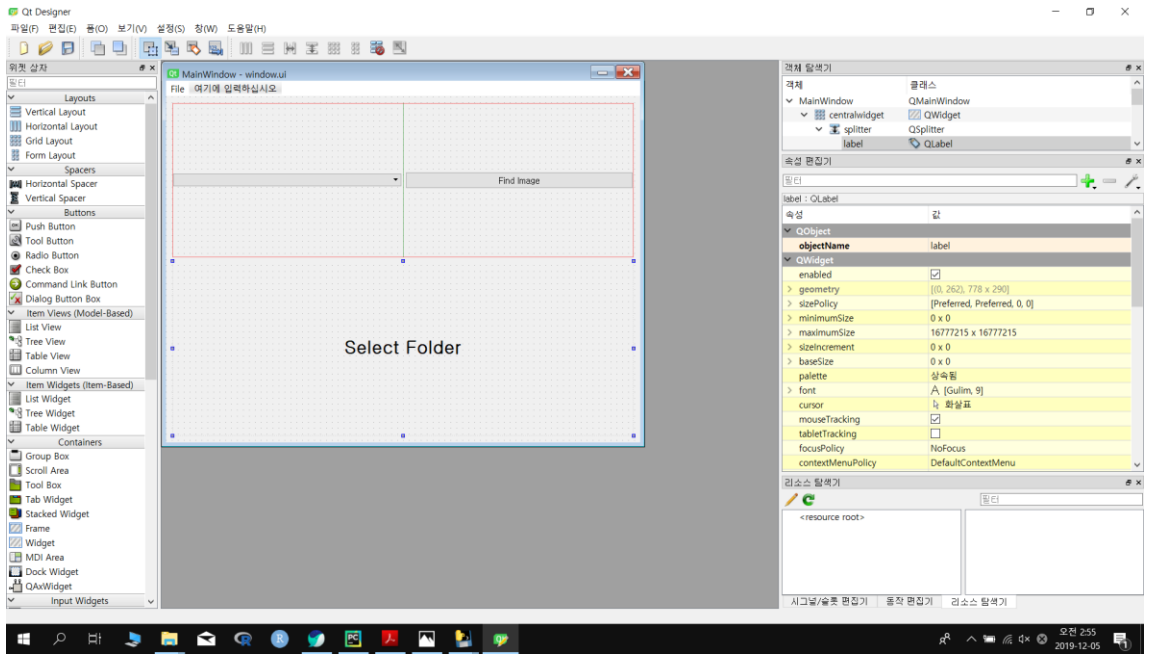
3. 개발 과정 – 내부 프로그램

- 유사 이미지 찾는 알고리즘과 GPS 태그를 입력하는 합쳐주어 찾은 유사 이미지들을 새로운 폴더에 저장하여 태그 정보를 입력할 수 있도록 하였다.

```python
def findImg(folder_path, query, save_dir):
    # 대상 이미지 불러오기
    query_img = cv2.imread(query, cv2.IMREAD_COLOR)
    query_img = cv2.resize(query_img, (400, 400)) # 이미지 사이즈 바꾸고
    query_hash = img2hash(query_img) # 해쉬 이미지로 변환

    qTag = getTag(query) # 쿼리 이미지 태그
    _createFolder(save_dir) # 저장할 폴더 생성

    for path in folder_path:
        # 이미지 읽기
        img = cv2.imread(path, cv2.IMREAD_COLOR)
        reimg = cv2.resize(img, (400, 400))

        # 이미지 찾는 동안 띄워놓기
        cv2.imshow("Searching...", reimg)
        cv2.waitKey(5)

        # hamming distance 를 통해 비교하기
        img_hash = img2hash(reimg)
        dst = hamming_distance(query_hash, img_hash) / 256 # hamming distance 비율로 바꾸기 (0 ~ 1)

        if dst < 0.20: # hamming distance 가 80퍼 이상 유사한 이미지를 찾는다.
            print(path, dst)
            cv2.imshow(path, reimg)

            tag = getTag(path) # 유사 이미지의 태그 정보
            # gps 태그가 없는 이미지들의 태그를 저장
            if tag['GPS'] == {}:
                tag["GPS"] = qTag["GPS"]

            # 유사 이미지들을 gps태그를 수정해서 새로 저장
            # 저장할 경로로 파일 이름 변경
            tmp = path.find('\\')
            sPath = save_dir + path[tmp:]
            cv2.imwrite(sPath, img) # 새 경로로 이미지 저장

            exif_bytes = piexif.dump(tag)
            piexif.insert(exif_bytes, sPath) # 저장한 이미지에 수정된 태그 삽입

    cv2.destroyWindow("Searching...")
    cv2.destroyAllWindows()
```

3. 개발 과정 – GUI

- GUI는 PyQt5 패키지를 이용하였다.
- Qt Designer를 이용하여 GUI를 개발하였고, 이미지 폴더를 불러와 원하는 사진을 미리 보는 기능을 추가하였다.




```python
class MyWindow(QMainWindow, form_class):
    def __init__(self):
        super().__init__()
        self.setupUi(self)

        self.actionOpen.triggered.connect(self.openDir)
        self.actionExit.triggered.connect(qApp.quit) # quitAction
        self.comboBox.currentIndexChanged.connect(self.imgPreview)

    # 폴더 불러와서 comboBox 에 파일 표시
    def openDir(self):
        dname = str(QFileDialog.getExistingDirectory(self, "Select Directory")) # dir 폴더 불러오기
        self.dir_path = glob.glob(dname + '/*.jpg', recursive=True) # 이미지 파일들 경로

        # 불러온 뒤 comboBox에 삽입
        self.comboBox.clear()
        self.comboBox.addItems(self.dir_path) # 경로에 있는 파일 이름 추가

    # comboBox 에서 선택할 때마다 이미지 보여줌
    def imgPreview(self):
        fname = self.comboBox.currentText() # comboBox 에서 선택할 때마다

        # 이미지 프리뷰
        pixmap = QPixmap(fname).scaled(782, 458) # 이미지 불러와서 label 사이즈만큼 이미지 크기 조절
        self.label.setPixmap(pixmap)
        self.resize(pixmap.width(), pixmap.height())

        self.resize(800, 600) # window 크기 조절
```

3. 개발 과정 - 마무리

- 내부 프로그램으로 만든 기능을 GUI에 버튼을 통한 이벤트로 추가하였다.
- OpenCV, piexif, PyQt5 등의 패키지를 사용하였다.

- 개발 과정에 관한 자세한 코드와 최종 코드는

https://github.com/choi-yh/ImgFinder/tree/master/Final 를 통해 확인할 수 있다.

4. 개선점 및 어려웠던 점

- **유사 이미지를 찾는 알고리즘의 개선**
  - 실행 결과를 보았을 때 유사 이미지를 생각만큼 완벽하게 찾아주는 것은 아니었다. 이미지 히스토그램 방식의 수정, Object Detection 방식, CNN 기법들을 적용해보면 효율이 올라갈 것이라는 생각이 들었다.
- **UI 디자인에 대한 문제**
  - PyQt5 라는 패키지를 처음 사용해봤기 때문에 문법 요소에서의 어려움이 많이 있었다.
  - 스스로 GUI를 개발하는 과정에서 디자인 요소도 많은 고민이 됐다.
- **코드에 대한 정리**
  - 필요한 기능들을 하나씩 개발하고 합쳐주는 작업을 반복하였는데 그 결과 전체 코드가 지저분하다는 느낌을 많이 받았다.

5. 느낀 점

- **완성에 대한 뿌듯함**
  - 소프트웨어 융합공학 연계전공을 시작하여 공부한 뒤 처음으로 하나의 프로그램을 완성하였고, 개발 과정에서 많은 시행착오를 겪었지만 내부 코드들을 합쳐 최종 프로그램이 작동했을 때 굉장한 희열을 느낄 수 있었다.
  - 어려운 수학 문제를 해결했을 때와 비슷한 느낌이었는데 개발한 프로그램을 실질적으로 사용할 수 있다는 점에서 더 높은 만족을 얻을 수 있었고 앞으로 개발 관련된 일을 하더라도 만족하면서 할 수 있겠다는 생각이 들었다.

5. 느낀 점

- **전공에서 배운 수학과 실제로 적용하는 수학적 개념**
  - 전공에서 배운 수학 및 통계 내용은 이론적인 측면만을 다루고 문제 풀이에만 초점이 맞춰져 있어 학생들이 수학 공부에 대한 이유를 물어본다면 실질적으로 납득이 가는 대답보다는 교과서적인 대답밖에 해주지 못했다.
- **공부의 필요성**
  - 이미지 찾는 알고리즘과 GPS 태그에 대한 공부가 되지 않았다면 개발에 어려움이 있었을 것이다.
  - 이미지 히스토그램 방식에 대한 공부가 더 되었다면 적용하여 더 나은 결과를 얻을 수 있을 것이다.
  - 현실에서 적용할 수 있는 수학, 통계적 개념에 대해서 공부할 필요성을 많이 느꼈고, 지금까지 공부해온 이론적인 공부가 앞으로 도움이 될 것이라고 생각했다.