



이미지 카빙 자동화 프로그램

융합보안공학과 성연수, 신나연, 한아름

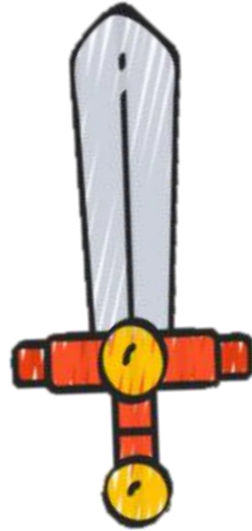
INDEX

- 01 프로젝트 목적
- 02 프로젝트 설명
- 03 프로젝트 진행
- 04 프로젝트 평가

INDEX

- 01 프로젝트 목적
 - 제안 배경(필요성)
 - 아이디어 도출
 - 목적(차별성)
 - 목표

01 프로젝트 목적 | 제안 배경 (필요성 – 문제관점)



디지털포렌식 (Digital Forensic)

- PC나 스마트폰 같은 디지털기기에 들어있는 데이터를 수집 · 추출한 뒤, 이를 바탕으로 범죄의 단서와 증거를 찾아내는 과학수사 기법

이미지 카빙 (Image Carving)

- Carving의 사전적 정의 : 조각하다 깎아내다
- 파일시스템의 경우 파일이 삭제되더라도 데이터를 가지고 있음
→ 이러한 특징을 이용하여 더미파일에서 포맷에 맞게 잘라내어 이미지를 추출가능

01 프로젝트 목적 | 제안 배경 (필요성 – 요구관점)

디지털 포렌식 시대, 이미지·동영상 자동 분석 수요 ↑

이유지 | 2017.05.23

공유 3 댓글 0

언어 선택 ▼ Google 번역에서 제공

가+ 가-

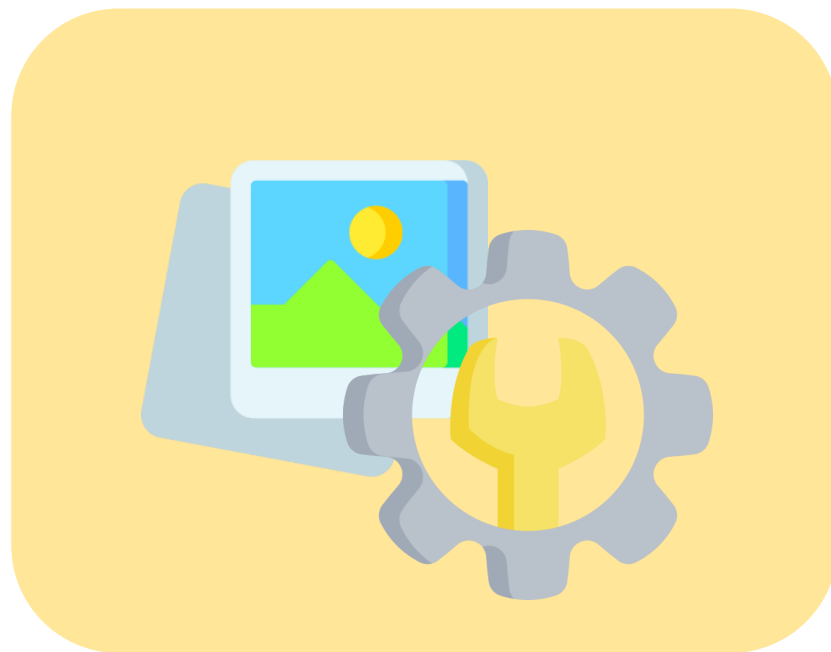
스마트폰과 디지털카메라, 전국 곳곳에 설치돼 있는 CCTV·영상감시 카메라를 통해 생성되고, 컴퓨터와 인터넷상에 저장·게재·배포되는 디지털 이미지와 동영상이 급증하고 있다.

이에 따라 수사기관에서도 사건 해결을 위한 단서나 범죄 증거를 찾기 위해 분석해야 하는 미디어의 양이 기하급수적으로 늘어나는 추세다.

범죄 증거를 찾기 위해
분석해야 할 **이미지의 증가**



디지털포렌식과 **자동화 tool**에 대한
요구 증가



이미지카빙 + 자동분석 및 정리



이미지 카빙 자동화 프로그램

01 프로젝트 목적 | 목적 (차별성)



이미지 포맷 별 설계
(JPG, PNG, GIF 등)



카빙 후 엑셀을 통한
자동화 정리



시간정보(생성, 수정, 삭제),
장소, 이미지 등 자동 정리

01 프로젝트 목적 | 목표

이미지카빙 자동화 프로그램을 제작할 것이다.

파일 시스템의 구조를 이해하고, 이를 프로젝트에 적용할 수 있다.

각 확장자 이미지에 대한 구조 파악 및 이미지 카빙할 수 있다.

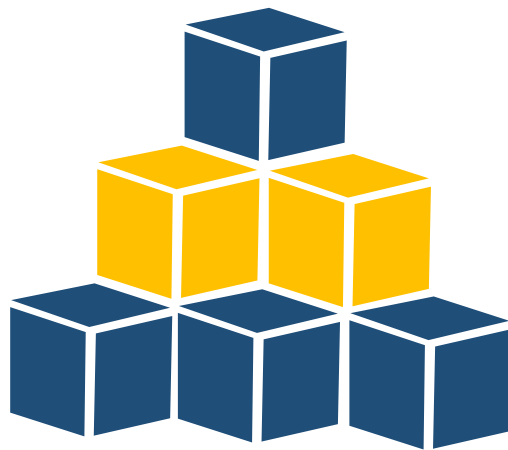
프로그램과 Exel을 연동되도록 프로그램하여, 획득한 정보를 정리할 수 있다.

INDEX

- 02 프로젝트 설명
 - 서비스 개요
 - 프로젝트 동작 원리

JPEG, PNG, BMP 등의 다양한 확장자의 이미지 파일 카빙 기능 제공

이미지 파일 카빙



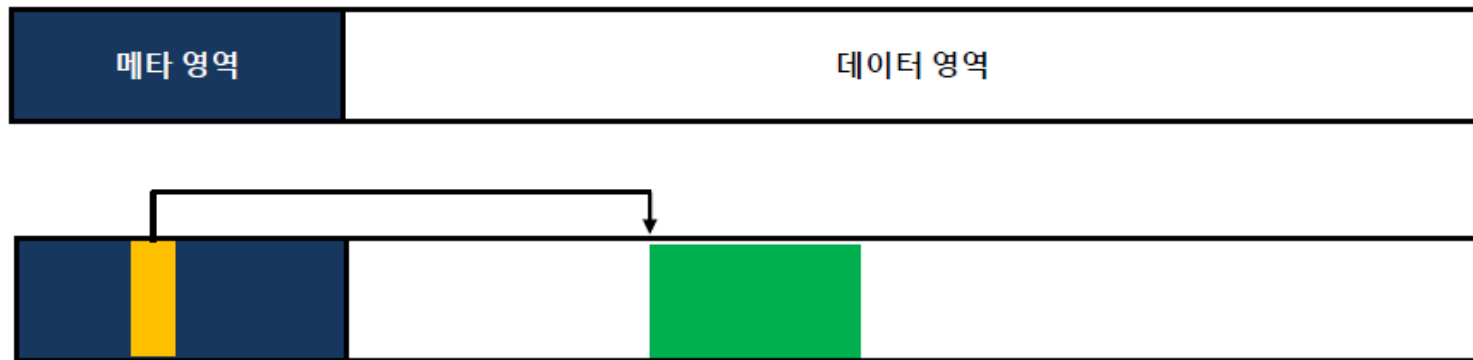
카빙 Tool(exe 파일) 제작

사용하기 쉬운 GUI 형태의 실행파일

이미지 EXIF 추출 & 엑셀 CSV 자동 생성

복구한 이미지 파일의 이름, 크기,
생성/수정/삭제시간, 카메라 모듈 이름,
GPS위치 등의 정보를 엑셀 보고서로 자동 생성

데이터 복구

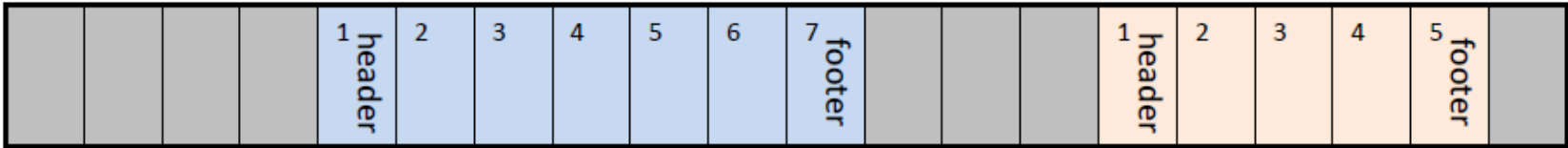


- 파일이 삭제 되었지만 메타 영역과 데이터 영역에 있는 데이터가 덮어쓰워지거나 변질되지 않았다면 메타영역은 올바른 데이터 영역을 가리키고 있다.
- 따라서 메타데이터가 가리키는 곳을 따라가 데이터를 복구하면 된다.

파일 카빙

- 만약 메타 영역의 데이터가 변질되거나 삭제되었을 경우 메타영역이 올바른 데이터 영역을 가리키고 있지 않기 때문에 위의 방법을 사용을 할 수 없다.
- 이때 사용하는 방법이 바로 카빙이다.
- 카빙은 하나의 시그니처나 문자열 등 파일의 내부 구조를 통하여 파일을 복구하는 방법이다.

헤더/푸터 카빙 방식



- 데이터가 연속적으로 저장되어 있을 경우 파일의 header와 footer를 찾아 손쉽게 복구가 가능하다.

0000h:	FF D8	FF E0	00 10 4A 46	49 46 00 01	01 01 01 2C	ÿøÿà...JFIF.....,
0010h:	01 2C 00 00	FF FE 00 25	53 4B 20 43	6F 6D 6D 75		...ÿp.SK Commu
0020h:	6E 69 63 61	74 69 6F 6E	73 20 43 79	49 6D 61 67		nications CyImag
0030h:	65 20 55 70	6C 6F 61 64	65 72 00 FF	E1 00 BB 45		e Uploader.ÿá.»E
0040h:	78 69 66 00	00 49 49 2A	00 08 00 00	00 06 00 00		xif..II*.....
DC30h:	9F F9 7F B7	43 93 FF 00	2F F6 EA 54	AC BA 1D 63		ÿù. -C"ÿ./ôêT¬°.c
DC40h:	C9 FF 00 97	FB 74 39 3F	F2 FF 00 6E	A5 4A F5 FF		Éÿ.-ût9?òÿ.nJöÿ
DC50h:	00 09 EB 0E	B2 25 8F 82	BD 64 21 24	64 9C 7E 5D		..ë..,d!\$dœ~]
DC60h:	4A 95	FF D9				J•ÿù

JPEG 파일 구조

- 이 방식을 사용할 수 있는 대표적인 구조에는 JPEG, PNG, GIF가 있다.
- JPEG의 경우 FFD8로 시작하여 FFD9로 끝나는 값을 가지고 있어 카빙 툴을 만들 때 이에 해당하는 값에서 데이터를 복구하면 된다.

파일크기 카빙 방식

	시그니처		파일 크기				헤더 크기									
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	78	96	00	00	00	00	00	00	76	00	00	00	28	00
00000010	00	00	40	01	00	00	F0	00	00	00	01	00	04	00	00	00
00000020	00	00	00	00	00	00	12	0B	00	00	12	0B	00	00	00	00

BMP 파일 구조

- 데이터에 헤더는 존재하지만 푸터가 존재하지 않을 경우, 이전의 방식으로 카빙할 수 없다.
- 하지만 헤더에 있는 파일의 크기 정보를 통해 어디까지가 이 프로그램에 해당하는지를 확인할 수가 있다.
- 이러한 방법을 통하여 복구하는 것이 파일크기 카빙 방식으로 BMP가 대표적이다.
- BMP의 경우 42 4D를 통해 파일의 시작점을 확인할 수 있고, 파일 크기와 헤더 크기를 참고하여 전체적인 파일의 크기를 구할 수 있다.

INDEX

- 03 프로젝트 진행
 - 프로젝트 설계 및 구현 과정
 - 프로젝트 예상 동작
 - 프로젝트 활동 일정
 - 역할 분담

03 프로젝트 진행 | 설계 및 구현 과정

프로그램 사용환경

사용 환경

운영체제

Windows 10

파이썬 버전

Python 3.7

에디터

PyCharm

03 프로젝트 진행 | 설계 및 구현 과정

사용 라이브러리

이미지 카빙 라이브러리

imghdr

이미지 유형 판단

binascii

바이너리-ASCII 간의 변환

이미지 정보 추출 라이브러리

PIL

Python Imaging Library, ExifTags

Pillow

PIL 후속 버전

실행파일 생성 라이브러리

PyInstaller

실행 파일 생성

PyQt5

GUI 프로그램

엑셀 CSV 생성 라이브러리

CSV

csv 파일 생성

Pandas

csv 파일 생성

03 프로젝트 진행 | 설계 및 구현 과정

프로그램 동작 과정



03 프로젝트 진행 | 설계 및 구현 과정

헤더 푸터 카빙 방식 | 복구 폴더 생성

```
import os
import binascii #바이너리 값을 hex 값으로 변환하기 위한 모듈
import sys
global jpg_count
global sector

jpg_count=0
sector = 0

def main():
    if len(sys.argv) != 2:
        print "using : carving [imagefile]" #덤프 이미지 파일 대상으로 사용
        sys.exit(2)
    # 복구 폴더 생성
    if os.path.isdir('Recovered_jpg') :
        pass
    else :
        os.mkdir("Recovered_jpg")

    filename=sys.argv[1]
```

03 프로젝트 진행 | 설계 및 구현 과정

헤더 푸터 카빙 방식 | 파일 상태정보 입력 / 진행 상황 확인

```
def carving(filename):
    global jpg_count
    global sector

    statinfo = os.stat(filename) #파일 상태 정보 입력
    f_size=statinfo.st_size
    print sys.argv[1]+" size : "+str(f_size) + " byte("+str(f_size/(1024*1024))+ " Mb)"

    f= open(filename,'rb')

    sector = 0 # 0번섹터
    while sector < f_size :
        Sector_read(f)
        if sector %(1024*1024*100) == 0 : # 100Mb 진행 될때마다 상황 표시
            print "검색중 찾은 파일수jpg : %d" % (jpg_count)
            print "남은 용량 %d / %d (%.2f" %(sector , f_size, float(sector)/float(f_size)*100)+'%')
        f.close

    print "jpg 파일 갯수 : " + str(jpg_count)
```

헤더 푸터 카빙 방식 | 섹터 읽기

```
def file_read(f):  
    global sector  
    sec=f.read(512)  
    f.tell()  
    sector+=512 #다음 섹터 읽기  
    return sec
```

헤더 푸터 카빙 방식 | 푸터 찾기

```
def Sector_read(f):
    global jpg_count
    global sector

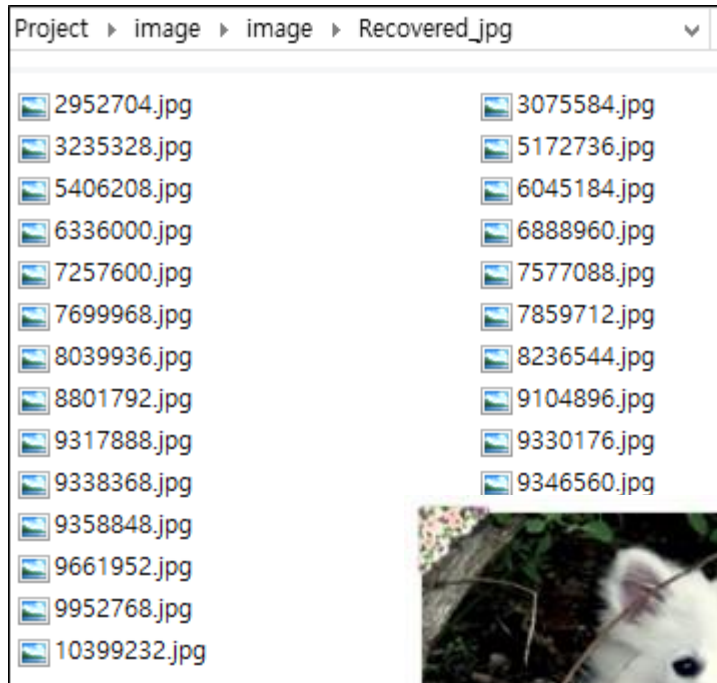
    sec=file_read(f)
    Hex_b= binascii.hexlify(sec)    #바이너리를 hex 값으로 변환
    footerFlag=False #푸터 찾기용 플래그
    if Hex_b[:8] == 'ffd8ffe0' : #섹터 처음 부분에 jpg 헤더값 찾기
        jpg_count+=1
        output_file=open("Recovered_jpg/"+str(sector)+".jpg","wb") #sector 번호로 파일명 설정
        output_file.write(sec)
        while not footerFlag : #푸터를 찾을때까지 계속 섹터를 읽으면서 쓰기
            sec = file_read(f)
            Hex_b= binascii.hexlify(sec)
            output_file.write(sec)
            if Hex_b.find('ffd900') != -1 or Hex_b[:-6]=='ffd900': #푸터를 찾으면 쓰기 완료
                footerFlag =True
        output_file.close()
        footerFlag=False
```

03 프로젝트 진행 | 예상 동작

헤더 푸터 카빙 방식 | 예상 동작

```
D:\Project\모바일 포렌식 Project\image\image>carving.py
using : carving [imagefile]

D:\Project\모바일 포렌식 Project\image\image>carving.py fat.001
fat.001 size : 1503657984 byte(1434 Mb)
[0] 파일수jpg : 27
[0] 104857600 / 1503657984 (6.97%)
[0] 파일수jpg : 27
[0] 209715200 / 1503657984 (13.95%)
[0] 파일수jpg : 27
[0] 314572800 / 1503657984 (20.92%)
[0] 파일수jpg : 27
[0] 419430400 / 1503657984 (27.89%)
[0] 파일수jpg : 27
[0] 524288000 / 1503657984 (34.87%)
[0] 파일수jpg : 27
[0] 629145600 / 1503657984 (41.84%)
[0] 파일수jpg : 27
[0] 734003200 / 1503657984 (48.81%)
[0] 파일수jpg : 27
[0] 838860800 / 1503657984 (55.79%)
[0] 파일수jpg : 27
[0] 943718400 / 1503657984 (62.76%)
[0] 파일수jpg : 27
[0] 1048576000 / 1503657984 (69.74%)
[0] 파일수jpg : 27
[0] 1153433600 / 1503657984 (76.71%)
[0] 파일수jpg : 27
[0] 1258291200 / 1503657984 (83.68%)
[0] 파일수jpg : 27
[0] 1363148800 / 1503657984 (90.66%)
[0] 파일수jpg : 27
[0] 1468006400 / 1503657984 (97.63%)
jpg 파일 갯수 : 27
```



이미지 정보 추출 | EXIF 데이터에서 GPS 정보 추출

```
from PIL import Image
from PIL.ExifTags import TAGS

filename = "image.jpg"
extension = filename.split('.')[-1]
if (extension == 'jpg') | (extension == 'JPG') | (extension == 'jpeg') | (extension == 'JPEG'):
    try:
        img = Image.open(filename)
        info = img._getexif()
        exif = {}
        for tag, value in info.items():
            decoded = TAGS.get(tag, tag)
            exif[decoded] = value
        # from the exif data, extract gps
        exifGPS = exif['GPSInfo']
        latData = exifGPS[2]
        lonData = exifGPS[4]
```

이미지 정보 추출 | 위도 경도 계산

```
# calculate the lat / long
latDeg = latData[0][0] / float(latData[0][1])
latMin = latData[1][0] / float(latData[1][1])
latSec = latData[2][0] / float(latData[2][1])
lonDeg = lonData[0][0] / float(lonData[0][1])
lonMin = lonData[1][0] / float(lonData[1][1])
lonSec = lonData[2][0] / float(lonData[2][1])
# correct the lat/lon based on N/E/W/S
Lat = (latDeg + (latMin + latSec / 60.0) / 60.0)
if exifGPS[1] == 'S': Lat = Lat * -1
Lon = (lonDeg + (lonMin + lonSec / 60.0) / 60.0)
if exifGPS[3] == 'W': Lon = Lon * -1
```

이미지 정보 추출 | GPS 출력

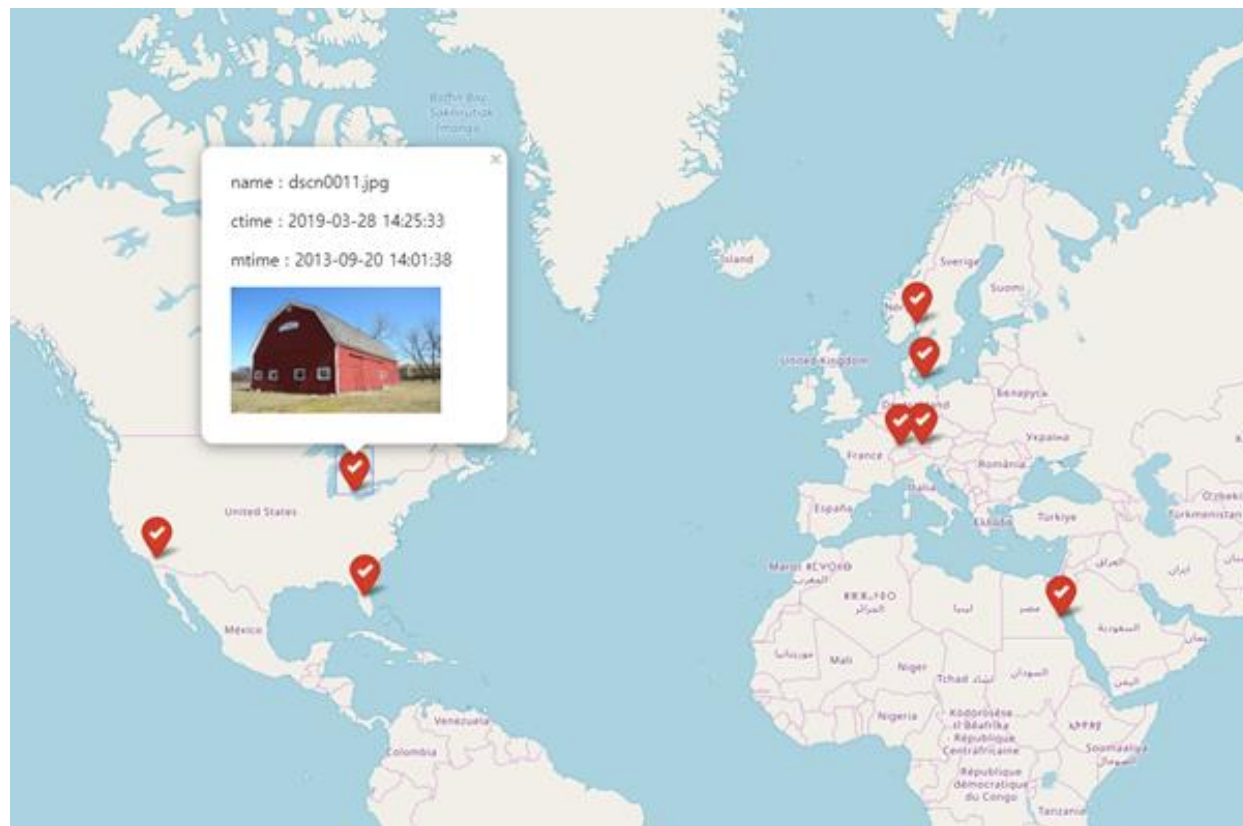
```
# GPS 출력
msg = "GPS is " + str(Lat) + "," + str(Lon)
print(msg)

#CSV 파일에 GPS 정보 입력
Gpswrite.write(str(file)+"," +str(Lat)+"," +str(Lon)+"," +str(ctime)+"," +str(mtime)+"\n")
print ('csv created ' )

if __name__ == '__main__':
    Gpswrite.write("NAME,"+"LAT,"+"LON,"+"CTIME,"+"MTIME,"+"\\n")
    jkmap = folium.Map(location=[47.975,-40], zoom_start = 3)
    GPSScript(path_dir)
```

03 프로젝트 진행 | 예상 동작

이미지 정보 추출 | 예상 동작



이미지가 촬영된 곳의 위치정보를 구글에 표시

03 프로젝트 진행 | 설계 및 구현 과정

실행 프로그램 만들기

PyInstaller 설치

```
>> pip install pyinstaller
```

exe 파일 만들기

```
>> pyinstaller 파이썬_파일_이름.py
```

exe 파일 변환

```
>> pyinstaller 파이썬_파일_이름.py
```

콘솔창 안뜨게 하기

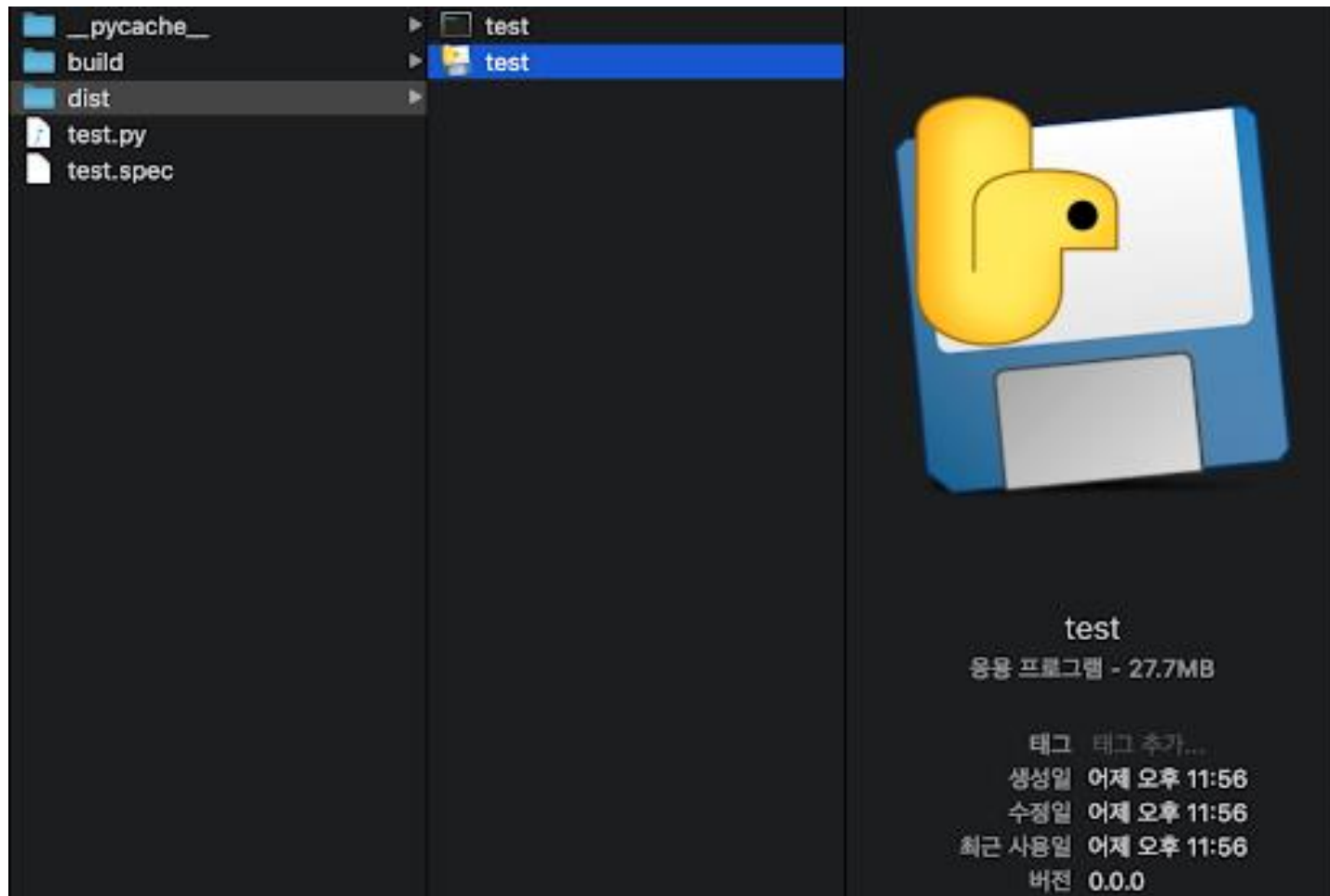
```
>> pyinstaller --noconsole 파이썬_파일_이름.py
```

exe 파일에 모든 라이브러리 압축하기

```
>> pyinstaller --onefile --noconsole 파이썬_파일_이름.py
```

03 프로젝트 진행 | 예상 동작

실행 프로그램 만들기 | 예상 동작



03 프로젝트 진행 | 설계 및 구현 과정

CSV 파일 생성 | 데이터 프레임 생성

```
import pandas as pd
df1 = pd.DataFrame({'Color': ['blue', 'green', 'red', 'black'], \
                      'Product ID': [1, 2, 34], \
                      'Product Name': ['t-shirt', 't-shirt', 'skirt', 'skirt']})
print(df1)
```

CSV 파일 생성 | CSV 파일 읽기

```
import pandas as pd
df = pd.read_csv('sample.csv')
print(df)
```

CSV 파일 생성 | CSV 파일로 저장하기

```
df.to_csv('sample.csv')
```

03 프로젝트 진행 | 예상 동작

CSV 파일 생성 | 예상 동작

	Color	Product ID	Product Name
0	blue	1	t-shirt
1	green	2	t-shirt
2	red	3	skirt
3	black	4	skirt

NAME	LAT	LON	CTIME	MTIME
1.txt	is No Image		2019-03-28 17:06	06:21.0
2.txt	is No Image		2019-03-28 17:06	06:24.7
Biking.jpg	33.875461	-116.302	2019-03-28 14:25	2013-09-20 14:07
Castle.JPG	55.007338	11.91093	2019-03-28 14:25	2013-09-20 14:08
Cat.jpg	59.924755	10.6956	2019-03-28 14:25	2013-09-20 14:22
Deutschland.JPG	47.975	7.829667	2019-03-28 14:25	2013-09-20 15:04
Disney.jpg	28.41879	-81.581	2019-03-28 14:25	2013-09-20 14:26
dscn0011.jpg	42.501235	-83.2507	2019-03-28 14:25	2013-09-20 14:01
kinderscout.jpg	NOT	NOT	2019-03-28 14:25	2013-09-20 14:14
Munich.JPG	48.141333	11.57667	2019-03-28 14:25	2013-09-20 14:58

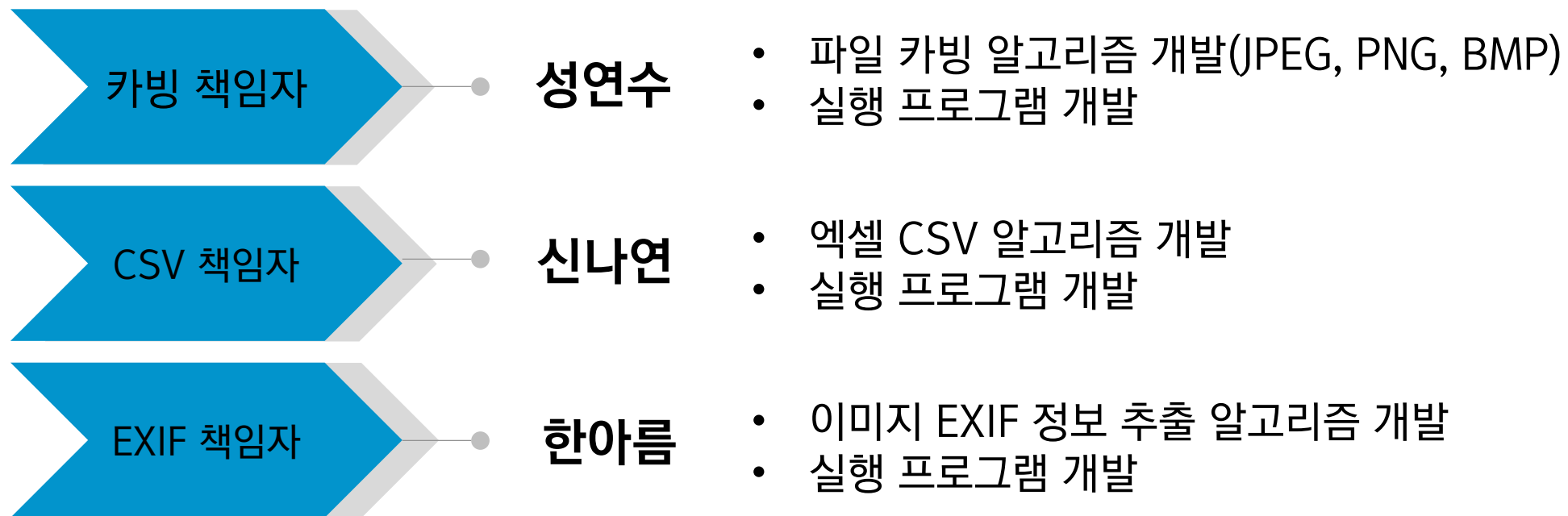
03 프로젝트 진행 | 프로젝트 활동 일정

	7주차							8주차							9주차							10주차						
자료 조사 및 분석																												
기본 틀 작성																												
Jpeg 카빙 툴 작성																												
Png 카빙 툴 작성																												
Bmp 카빙 툴 작성																												
GUI / Excel 연동																												
구동 및 버그 수정																												

	11주차							12주차							13주차							14주차						
자료 조사 및 분석																												
기본 틀 작성																												
Jpeg 카빙 툴 작성																												
Png 카빙 툴 작성																												
Bmp 카빙 툴 작성																												
GUI / Excel 연동																												
구동 및 버그 수정																												

*일정은 상황에 따라 변동 가능성 있음

03 프로젝트 진행 | 역할 분담

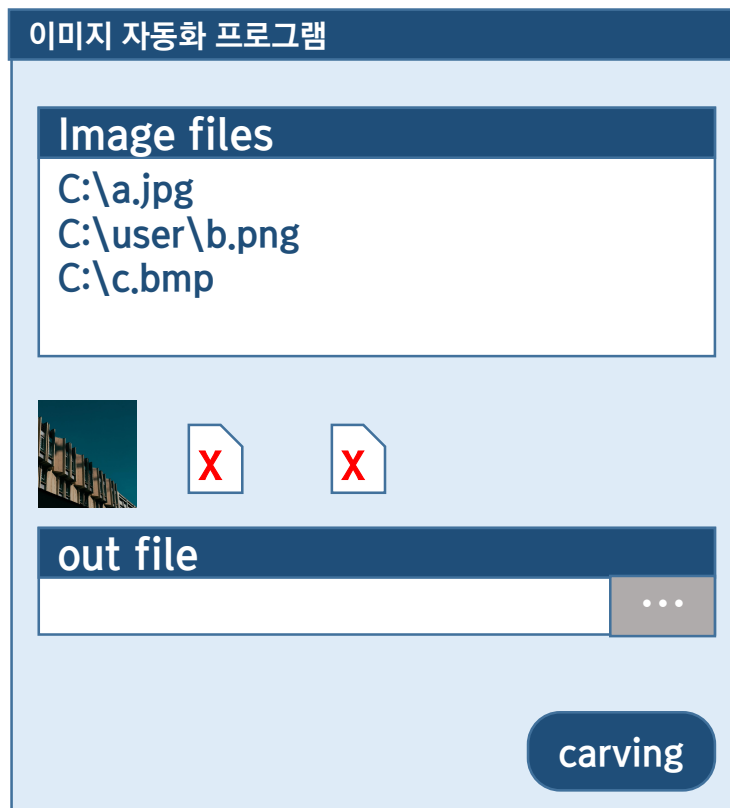


역할분담 : 책임자를 두지만 다같이 구현 진행

INDEX

- 04 프로젝트 평가
 - 예상 결과물
 - 결과물의 기대효과
 - 평가 방법/기준

이미지 카빙 자동화 프로그램(GUI)



이미지 카빙 자동화 프로그램(CMD)



CMD

```
Microsoft Windows [Version 10.0.18368.336 ]  
(c) 2019 Microsoft Corporation. All rights reserved
```

```
C:\ > 이미지자동화카빙프로그램.py -i a.jpg -o b.jpg -e  
Request succeed!  
Excel file → C:/a(1).xlsx
```

```
C:\ > 이미지자동화카빙프로그램.py -i c.bmp -o d.bmp  
Request failed...
```

Out file (CSV파일)

	A	B	C	D	E
1	In file name	In file image		Out file name	Out file image
2	a.jpg			A_1.jpg	
3					
4					
5					
6					
7					

1 확장자 분류

복원하고자 하는 이미지의 확장자를 몰라도 이미지를 분류하여 카빙할 수 있도록 지원

2 이미지 카빙

JPEG, PNG, BMP 이미지 파일을 파일 구조에 따라 두 가지 방법으로 카빙

3 이미지 EXIF 정보 제공

카빙 완료 후 속성정보를 추출하여 이미지 EXIF 정보를 제공

4 CSV 파일 제공

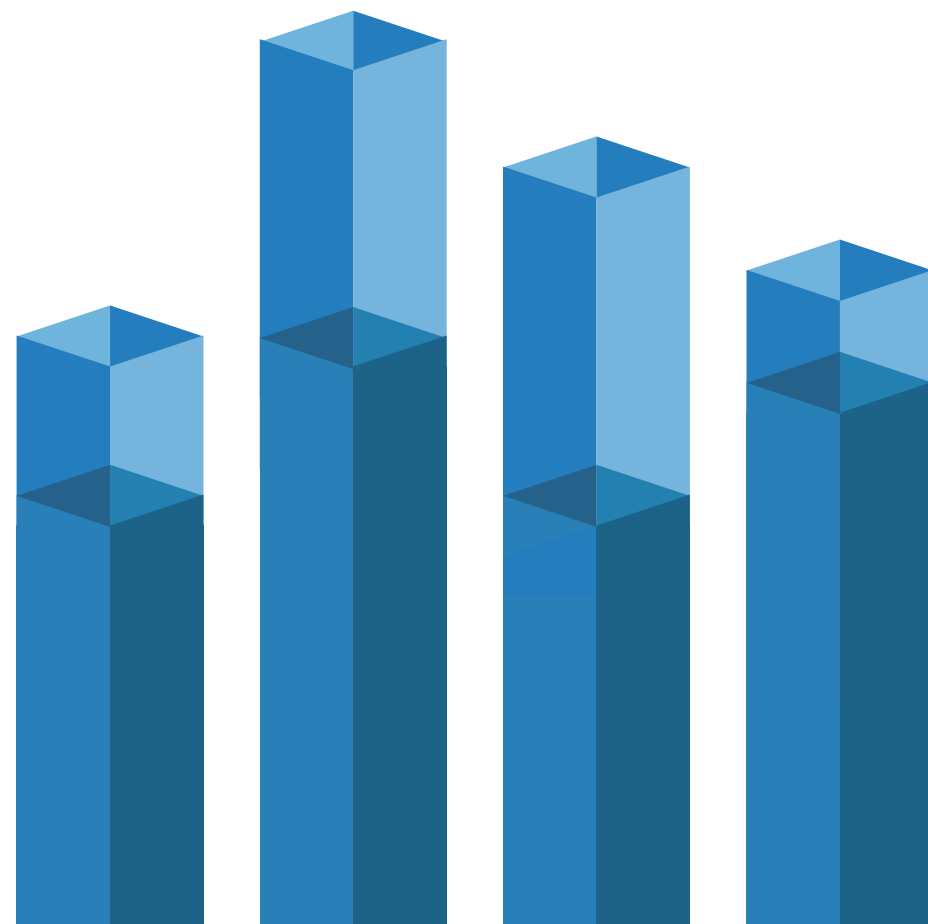
카빙 결과를 엑셀 프로그램의 CSV로 확인할 수 있음

완성도

JPEG, PNG, BMP 이미지 카빙 알고리즘이 구현되었는가?
이미지 EXIF 정보 추출 기능이 구현되었는가?
실행 파일로 이미지 카빙 프로그램을 만들었는가?
실행 결과를 CSV로 확인할 수 있는가?

기능 구현

여러 이미지의 확장자를 구분할 수 있는가?
이미지 카빙 과정에서 걸리는 속도는 적절한가?
이미지 카빙 과정에서 발생하는 오류는 없는가?



- 기사 이미지 : <http://www.bloter.net/archives/280325>
- 안랩 이미지 카빙 : <https://whitesnake1004.tistory.com/248>
- 파일 카빙 : <https://whitesnake1004.tistory.com/248>
- 데이터 복구, 파일 카빙 방법 : <https://kali-km.tistory.com/entry/filerecovery>
- 파일 크기 카빙 코드 : <http://blog.naver.com/bitnang/220218322877>
- 실행파일 만들기 : https://wikidocs.net/21952#_3
- 이미지 GPS 정보 추출 : <https://cpuu.postype.com/post/23100>
<https://www.trossjo.kr:444/wordpress/index.php/2019/04/05/imagegps-python/>
<https://www.trossjo.kr:444/wordpress/index.php/2019/04/05/imagegps-python/>
- BMP 파일 구조 :
http://blog.daum.net/_blog/BlogTypeView.do?blogid=0tsgd&articleno=5&_bloghome_menu=recentthumb
- PANDAS CSV 생성 : <http://hleecaster.com/python-pandas-creating-and-loading-dataframes/>

THANK YOU!