Homework #2

- **주제: OpenCV를 사용하여 스마트폰 카메라 보정 (camera calibration) 구현**
  - **내 카메라의 내부변수, 외부변수 구하기**

1. OpenCV 설치 및 사용법

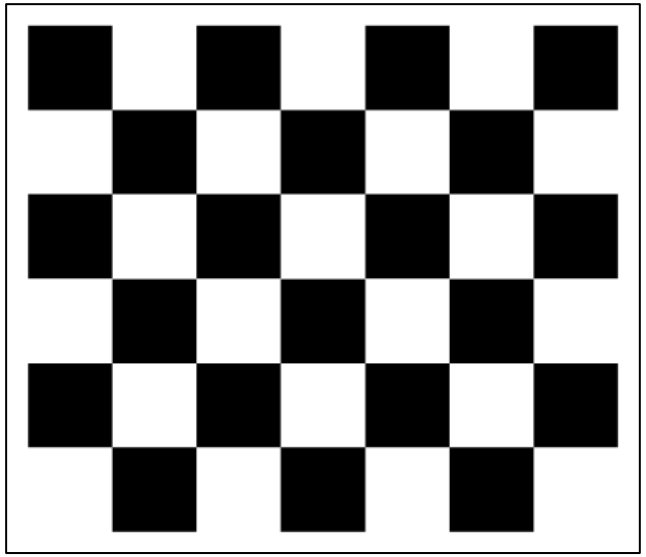
- OpenCV는 영상이나 비디오와 같은 멀티미디어 신호처리에 매우 편리하게 사용할 수 있는 프로그램 library 이다.
- 설치 및 사용법 익히기

  1) https://opencv.org/ 에서 OpenCV를 다운로드한다.
     - 각자 computer의 OS에 맞는 버전 다운로드
     - 버전이 4.* 또는 3.* 두 종류인데 어떤 것이든 상관없음

  2) 인터넷 자료를 이용하여 OpenCV 사용법을 학습
     - 사용법은 https://docs.opencv.org/master/ 를 활용할 것
     - 이외에도 인터넷에 무수히 많은 tutorial 이 있음

Homework #2

2. Chessboard 카메라 보정판 영상 획득



Chessboard

1) LMS에 첨부한 chessboard 그림을 A4에 출력하고, 책상, 마루바닥 등의 평평한 위치에 놓는다. (chessboard 아래쪽 빈칸에 본인의 이름을 작게 적는다)

2) 본인의 스마트폰 (과제 #1에서 사용한 스마트폰 사용 추천) 으로 서로 다른 각도와 위치에서 10장의 영상을 촬영한다. (해상도는 최소로 예를 들어 1000pixel 이하로 촬영, 화질도 너무 고화질 필요없음, 파일 용량 최소화)

- **중요**: 과제 #1에서 사용한 스마트폰 사용 추천, 과제 #1에서 영상획득 때 카메라의 세팅(설정)과 동일하게 영상 획득이 필요함
- chessboard 영역이 영상에서 50% 이상이 되도록 촬영
- chessboard 프린트 출력에서 격자 한 칸의 크기를 측정하여 코너점들의 World 좌표 (X,Y,Z) 계산에 활용

3) 촬영한 영상을 컴퓨터에 내려 받고, OpenCV 를 이용하여 프로그램을 작성하여 (C/C++/Python 등) 10장의 사진을 모니터에 동시에 출력

- chessboard 영상 1개당 윈도우 창을 따로 출력
- 촬영사진은 추후에 사용할 수 있으니 컴퓨터나 폰에 보관

* 보고서 제출자료(1): 모니터에 10개 윈도우 창을 적절히 배치하고 screen capture 후 사진을 보고서에 추가

Homework #2

4) OpenCV에서 제공하는 함수를 사용하여 카메라 보정 프로그램을 작성한다.
   - C/C++/Python 등 사용
   - Internet의 camera calibration program 참고 가능
5) * 보고서 제출자료(2): OpenCV 카메라 보정 결과를 화면 capture 및 .txt 파일로 저장
6) OpenCV의 내부보정값은 초점거리 f와 스케일 S (pixel/mm)가 곱해진 결과를 보여준다.
   - 과제#1에서 찾은 본인 스마트폰의 카메라 센서 스펙에서 스케일값을 결정하고 카메라 렌즈의 focal length를 구하시오.
     ex) f*S = 600, S=240 => f = 2.5mm
   * 보고서 제출(3): 과제 #1에서 본인 카메라 스펙의 focal length 와 비교
     - focal lengh를 mm 또는 pixel 로 비교 가능)
     - 과제 #1에서 영상을 획득할 때와 동일한 카메라 세팅(설정)을 사용한 경우에는 focal length 가 거의 동일하게 나와야 함

7) OpenCV의 undistort ()함수를 사용하여 카메라 영상의 왜곡을 보정하고 보정된 영상을 저장한다.

Homework #2

* 유의사항

- 이번 과제 #2에서 카메라의 내부 및 외부 파라메터를 보정하고
- 그 결과를 이번 학기에 이후의 다른 과제에서도 사용 예정임
- 따라서, 이번 과제에서 카메라의 세팅(설정)값 (영상의 해상도, 초점거리 (줌 값)) 등을 이후의 과제에서도 동일하게 사용할 예정임
- 이번 과제의 카메라의 세팅을 기록해 둘 것
- 카메라의 보정 결과에서 이후의 과제에서는 내부파라메터 (K) 를 주로 사용함

Homework #2

- OpenCV document 참조 사이트

  https://docs.opencv.org/3.4/dc/dbb/tutorial_py_calibration.html

- Camera calibration 함수 (C++)

```
double cv::calibrateCamera ( InputArrayOfArrays    objectPoints,   보정판 코너점의 3차원 world 좌표를 저장한 배열
                             InputArrayOfArrays    imagePoints,    보정판 영상에서 코너점의 2차원 좌표를 저장한 배열
                             Size                  imageSize,      영상크기
                             InputOutputArray      cameraMatrix,   내부보정 파라메터 (K= KsKf)
                             InputOutputArray      distCoeffs,     렌즈왜곡 파라메터
                             OutputArrayOfArrays   rvecs,          월드좌표->카메라좌표의 회전행렬
                             OutputArrayOfArrays   tvecs,          월드좌표->카메라좌표의 이동행렬
                             int                   flags = 0 ,
                             TermCriteria          criteria = TermCriteria(TermCriteria::COUNT+TermCriteria::EPS, 30, DBL_EPSILON)
                           )
```

Python:

```
retval, cameraMatrix, distCoeffs, rvecs, tvecs = cv.calibrateCamera( objectPoints, imagePoints, imageSize, cameraMatrix, distCoeffs[, rvecs[, tvecs[, flags[, criteria]]]]
```

Homework #2

- Image undistortion 함수 (C++)

```
void cv::undistort ( InputArray    src,          입력영상
                     OutputArray   dst,          출력영상
                     InputArray    cameraMatrix, 내부보정 파라메터 (K= K_s K_f)
                     InputArray    distCoeffs,   렌즈왜곡 파라메터
                     InputArray    newCameraMatrix = noArray()   새로운 내부보정 파라메터
                   )
```

Python:

```
dst = cv.undistort( src, cameraMatrix, distCoeffs[, dst[, newCameraMatrix]] )
```

Camera calibration

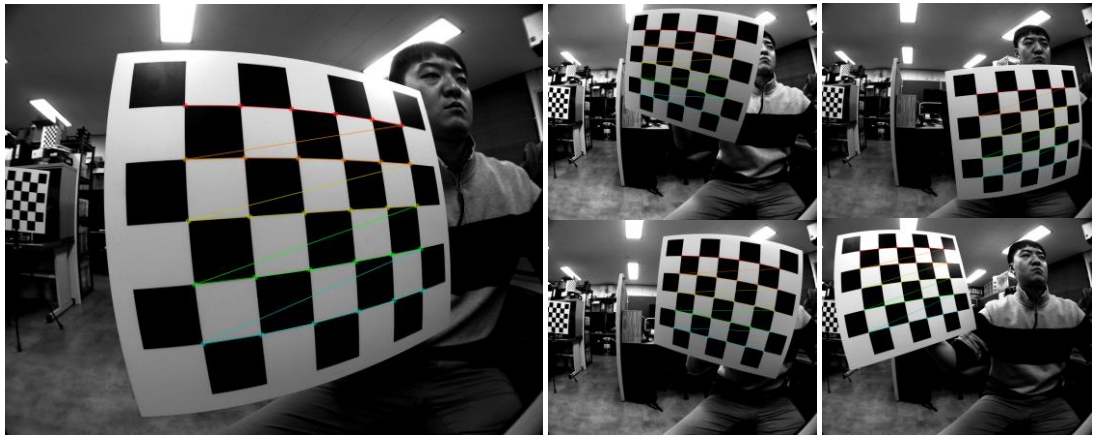
compatibility with OpenCV version : 3.x, 4.x

1. **findChessboardCorners(…)**
   - ➢ internal corners of the chessboard
2. **cornerSubPix(...)**
   - ➢ find the sub-pixel accurate location of corners

● Example.

findChessboardCorners() + coernerSubPix()

3. **drawChessboardCorners(…)**
   - ➢ Renders the detected chessboard corners
4. **calibrateCamera(…)**
   - ➢ find intrinsic and extrinsic parameters from several views of a chessboard pattern



drawChessboardCorners()

보정결과화면
(이 화면에는 외부변수는 없는데 외부변수도 같이 출력하기 바람)

```
intrinsic
[681.7775224765145, 0, 630.3440830813091;
 0, 682.0975983136947, 515.8322849792697;
 0, 0, 1]
distCoeffs
[-0.3232040816955161, 0.111713703179034, -0.0002184692480051047, 0.0001799130213567006, -0.01757012985795507]


error_rating : 0.467865
Error_rating : 0.467865
```

calibrateCamera()

Camera undistortion

compatibility with OpenCV version : 3.x, 4.x

1. **undistort(…)**

- Example.

Input image

Undistort

Output image

Homework #2 (Part 2)

Homework #2의 Part 2 : Calibration SW를 이용하여 OpenCV와 비교

- Camera calibration SW
  - 인터넷에서 구할 수 있는 SW tool 사용

    ex1) GML C++ Camera Calibration Toolbox

    https://web.archive.org/web/20190103151950/http://graphics.cs.msu.ru/en/node/909

    ex2) DarkCamCalibrator

    https://darkpgmr.tistory.com/139
  - 또는 기타의 SW tool 사용 가능

Homework #2 (Part 2)

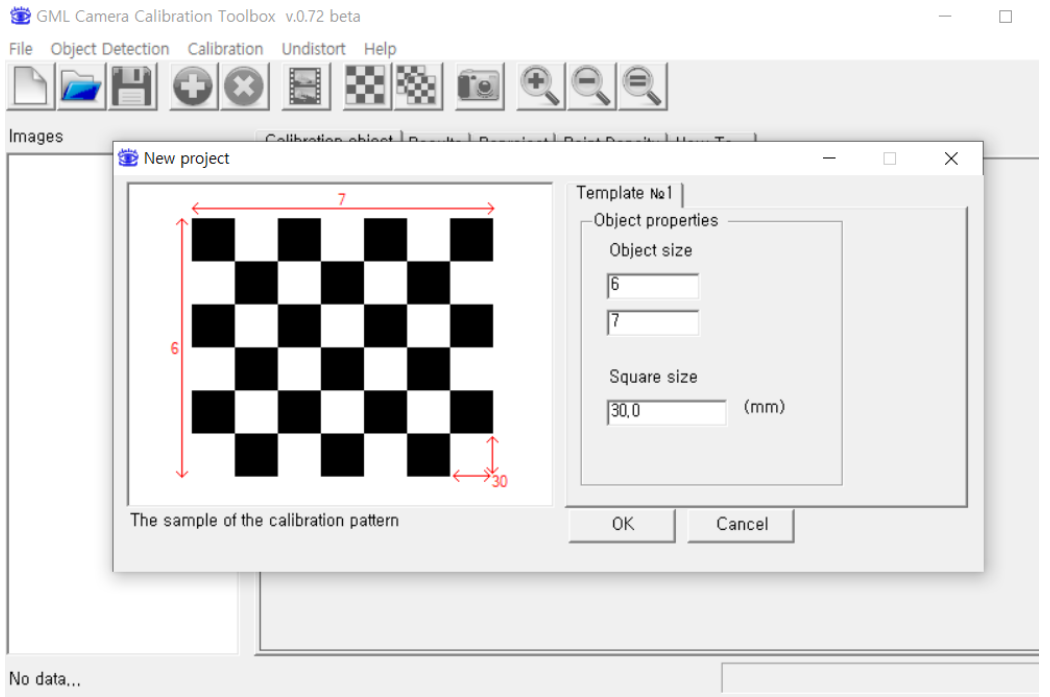
GML C++ Camera Calibration Toolbox 기준으로 설명

1) LMS에서 ver 0.75를 다운 받아서 설치

2) SW시작 방법
   - New project 먼저 실행
   - Number of template은 1개
   - 보정판의 size는 6x7로 입력
   - Square size도 실제 크기 입력

3) 보정판 영상을 load

4) 카메라 보정을 실행



도움말이나 https://web.archive.org/web/20190103151950/http://graphics.cs.msu.ru/en/node/909 에서 사용법 확인

Homework #2 (Part 2)

카메라 보정 결과 확인

- 아래 그림과 같은 형태로 결과를 확인 (파일출력 가능)
  - 보고서에도 출력해야함

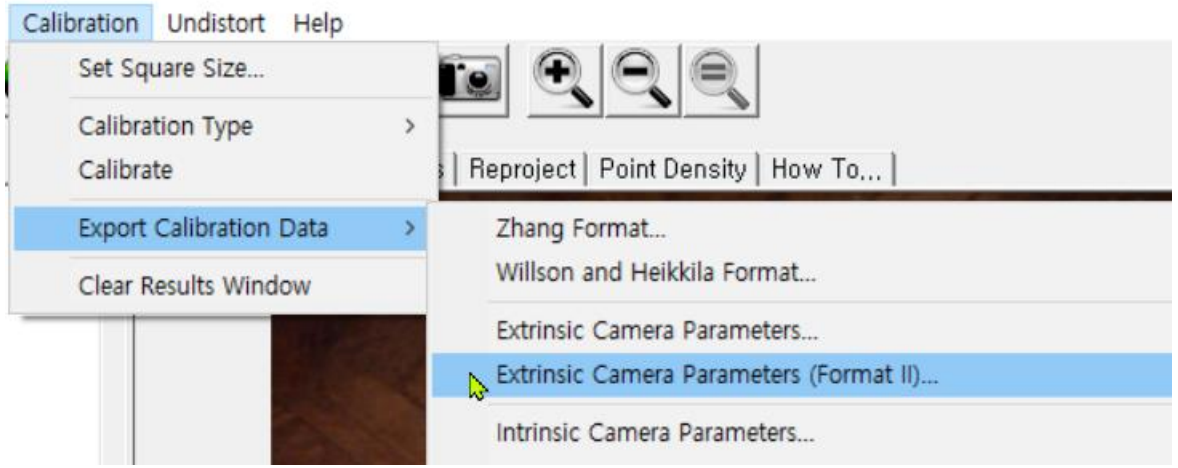
결과화면 capture하여 보고서 출력

Parameter	Value
Calibration date	11/18/2011 6:02:29 PM
Number of images	8
Base template	1
Square size	30.000 (mm)
Focal length	[3616.607 3617.186] ± [28.911 27.793]
Principal point	[1113.987 761.476] ± [21.102 30.388]
Distortion	[-0.007101 -0.182392 0.005124 0.007720] ± [0.036440 1.024242 0.001490 0.001670]
The camera matrix	[3616.607 0 1113.987; 0 3617.186 761.476; 0 0 1]
Pixel error	[0.40 0.45]

왜곡파라미터

내부파라미터

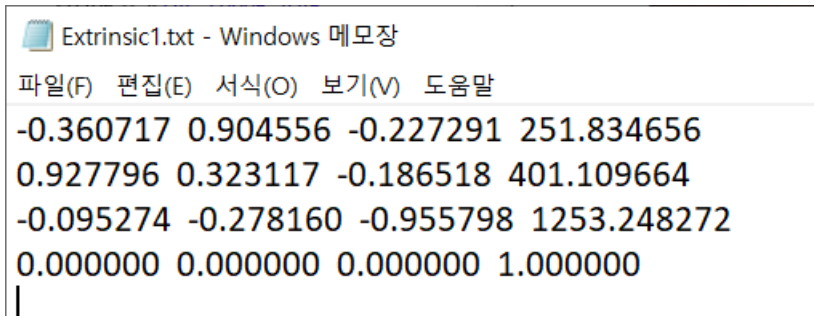
- 외부파라메터 출력 (format II 사용)

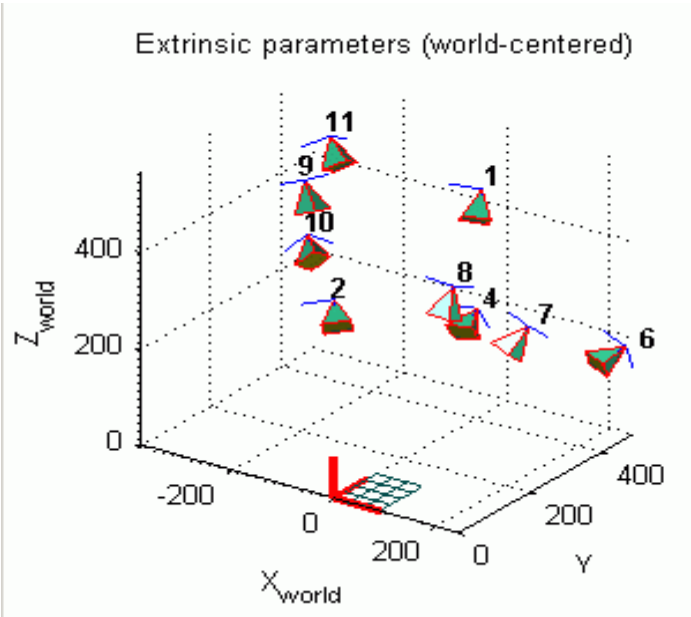


Homework #2 (Part 2)

카메라 보정 결과 확인

- 외부파라미터 출력 (.txt)
  - 입력한 영상의 수만큼 txt파일 생성됨
  - 파일 출력 예





  - 외부파라미터 파일이 많은 이유는 오른쪽 그림과 같이 보정판이 월드좌표계 기준이고 영상을 촬영한 각 카메라 위치에서 world->camera transformation 이 다르기 때문

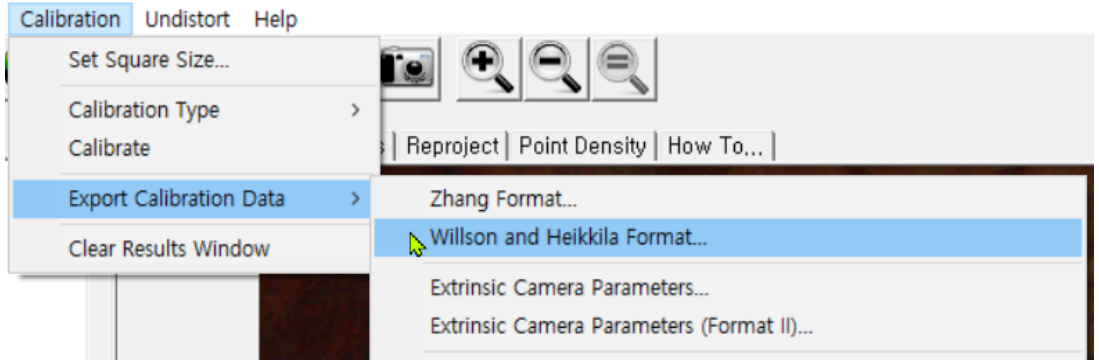
* 보고서 제출(4): Calibration SW의 결과를 보고서에 출력
- 앞 페이지의 결과 화면 capture
- 10개 카메라의 extrinsic parameter
- Zhang's calibration과의 차이 설명

Homework #2 (Part 2)

카메라 보정 결과 확인

- 보정판의 코너점들의 월드좌표와 영상에서 코너점의 픽셀좌표확인

1) export calibration data에서 willson and heikkila format을 선택하여 아래와 같은 txt 파일을 생성함



2) 이 파일은 보정판 코너점의 월드좌표와 픽셀좌표를 보여줌

PICT0001_WH.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말

0.000000E+000	0.000000E+000	0.000000E+000	6.517557E+002	4.615783E+002
0.000000E+000	3.000000E+001	0.000000E+000	7.212286E+002	4.861078E+002
0.000000E+000	6.000000E+001	0.000000E+000	7.893201E+002	5.111600E+002
0.000000E+000	9.000000E+001	0.000000E+000	8.589738E+002	5.360905E+002
3.000000E+001	0.000000E+000	0.000000E+000	6.232687E+002	5.287567E+002
3.000000E+001	3.000000E+001	0.000000E+000	6.921730E+002	5.538378E+002
3.000000E+001	6.000000E+001	0.000000E+000	7.609679E+002	5.788935E+002

Homework #2

- 보고서 제출자료

1) chessboard 10장 영상을 opencv로 10개의 윈도우로 출력하고 화면 capture
   - chessboard 아래쪽에 본인의 이름이 적혀져 있어야 함
   - 10개 윈도우를 컴퓨터 화면을 동시에 출력하면 글자가 작게 보일 수 있으나 식별 가능하면 됨

2) program source file 1개, 보고서에 capture 또는 ctrl^c, ctrl^v (.h 파일 등은 필요없음)

3) 보정결과 화면 capture (image file) (내부값, 왜곡값, 보정오차값)

4) 보정결과 .txt
   - 카메라 내부값, 렌즈왜곡값, 보정오차값
   - 카메라외부값 (rotation, translation) (보정판 영상 1개당 외부변수가 1set 있음. 모든 보정판에 대한 결과값 출력)

5) 계산된 스마트폰의 focal length와 스펙 상의 focal length 비교
   (iphone 등 focal length 스펙을 찾을 수 없으면 비교부분 제외 (인터넷확인바람))

6) 원영상 및 왜곡 보정 영상 (1 pair)

7) GML camera calibration SW 의 결과

- 위 모든 자료를 hwp, word 등 워드프로세서=> 하나의 PDF 파일로 변환하여 제출