<모바일응용 소프트웨어 설계>

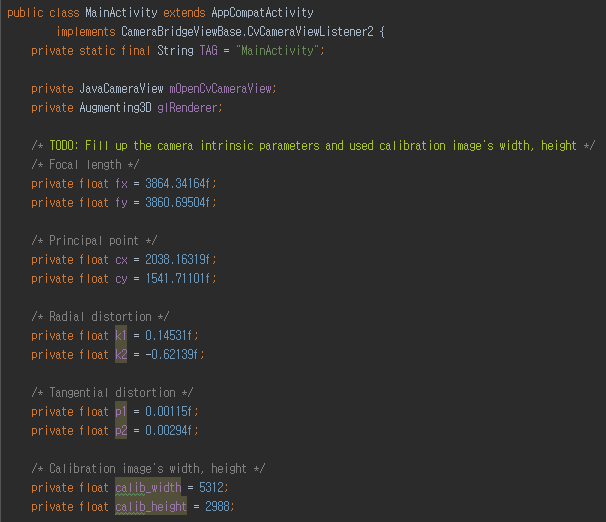
# 9주차 실습 과제

정보통신공학과

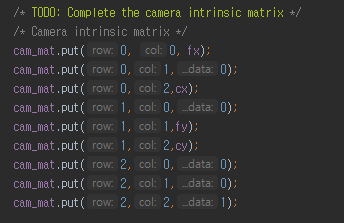
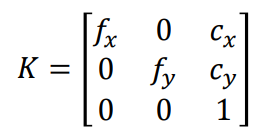
12161774 이 채 은

**[소스 코드]**

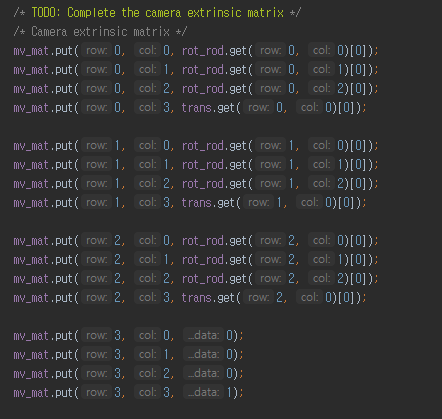
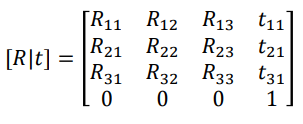
1. MainActivity.java



=> Camera intrinsic matrix 를 계산하기 위해 필요한 원소 fx, fy, Cx, Cy, distortion coefficients 를 입력해준다. Matlab 을 이용해서 캘리브레이션한 값을 넣는다. 또한 카메라로 찍은 이미지의 width, height 를 설정해준다.

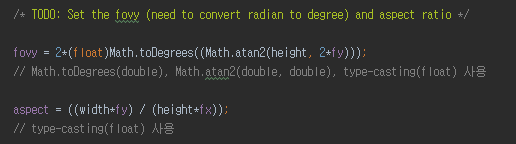


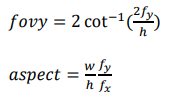
**=>** put 을 이용해서 Camera intrinsic matrix 를 완성해준다. Matrix 의 행과 열은 0부터 시작한다. 따라서 1행 1열부터 row:0, clo:0으로 시작한다.



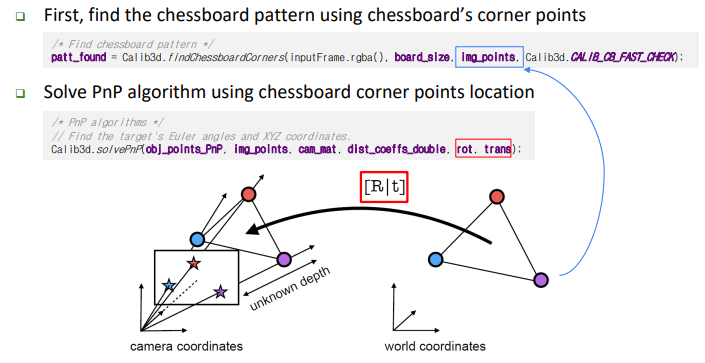
**=>** put 을 이용해서 Camera extrinsic matrix 를 완성해준다. 왼쪽의 matrix에서 보듯이 R은 rot\_rod 의 원소이고, t는 trans의 원소이다.

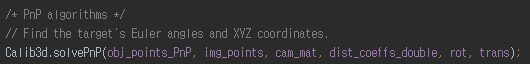
2. Augmenting3D.java



**=>** fovy, aspect 를 계산하는 식이다. Fovy 식을 나타내기 위해서 각도를 라디안으로 변경해주는 Math.toDegress와, 역탄젠트를 계산해주는 Math.atan2를 이용하고 float로 type-casting 을 한다. Aspect는 w = width, h = height 로 나타낸다.

**[ 실습 결과 및 고찰 ]**





=> 이번 실습에서는 chessboard 를 카메라로 찍어서 카메라 캘리브레이션한 값을 직접 넣어서 x, y, z 축을 띄워보았다. 이 때 chessboard 가 이미지 마커로 쓰인다.

Rotation, translate 를 구하기 위해서 PnP algorithms 의 solvePnP를 이용한다. World 좌표계에 나타나져 있는 물체의 위치를 [R|t] 를 이용해서 Camera 좌표계에 나타내야 한다. 따라서 World에서 Camera 좌표계로 바꾸기 위해서 rotation, translate가 필요한데 그 값들을 구해주는 역할을 한다. 어떤 translate 와 어떤 rotation 이 일어나는 지 알 수 있다. 따라서 solvePnP에는 총 5개의 인자가 있다. 앞의 세 인자는 input이고, 뒤의 두 인자는 output이다. Obj\_points\_PnP는 어떤 체스 보드를 쓸건지 나타낸다. Imag\_points 는 image의 좌표계, cam\_mat는 카메라 intrinsic matrix를 나타낸다. Dist\_coeffs\_double 은 왜곡 계수 matrix이다. 이들을 이용해 최종적으로 rot, trans가 나오는 것이다. 그리고 여기에 사용되는 intrinsic parameter, 왜곡 계수들은 카메라 캘리브레이션을 통해서 구한 값들을 나타낸다. (fx, fy, cx, cy, k1, k2, p1, p2)

완성한 코드를 직접 실행시켰더니, 체커 보드에 x, y, z 축이 고정되어 나타나는 것을 확인할 수 있었고, 큐브 또한 체커 보드 위에 제대로 나타났다. 하지만 체커보드의 시작점과 다른 위치에 축이 생기는 것을 보았는데, 이는 카메라의 기종 별로 화면 배율과 resolution 이 다르기 때문에 shifting 현상이 일어난 것이다. 이번 과제를 통해 pipeline에서 local => world => camera 여러 좌표계 사이의 변환 과정을 알 수 있었고, 그 변환 사이에 쓰이는 matrix와 계산 인자, 값들이 어떻게 구성되는지 확인할 수 있었다.