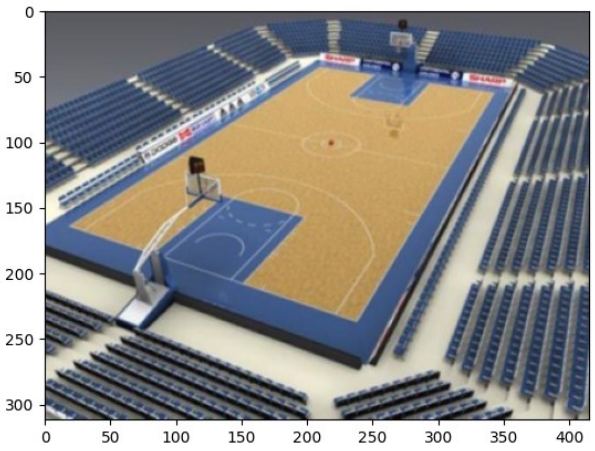
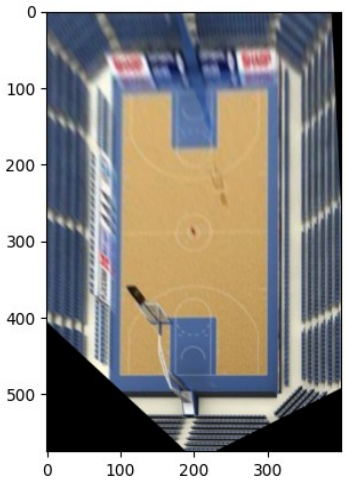
HOMEWORK SET 4 – Víctor Mira Ramírez

Exercise 1

To solve this exercise we first plot the image with matlpotlib, get the position of the four corners of the basketball court, the blue ones. I made a written calculation and estimated the final position of the four corners mantaining the proportion of a basketball court and adding a margin.






```
path=r'/home/victor/fisicaua/tercero/SIUE/robotic_vision/entregas/hw4/
basketball_court.jpg'
img = cv2.imread(path, cv2.IMREAD_COLOR)
RGB_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) # translate to rgb
pts1 = np.float32([[210, 41], [343, 62], [239, 237], [19, 164]]) # clockwise starting from NW
pts2 = np.float32([[100, 100], [300, 100], [300, 475], [100, 475]])
# Apply Perspective Transform Algorithm
matrix = cv2.getPerspectiveTransform(pts1, pts2)
result = cv2.warpPerspective(RGB_img, matrix, (400, 575))
# added 100px size around the court so that the image is narrower
# the proportion of a basketball court is approximately 1.87
plt.figure()
plt.imshow(result)
plt.figure()
plt.imshow(RGB_img)
plt.show()
```

Exercise 2

For this exercise I had to figure out how matlab works in the first place, as I didn't have it installed. An online version was used to ease on this matter. The code is just copied from the book and played around some parameters.

Increasing noise on the cam variable will make the maximum residual increases over 0.1 pixels. If noise is set to 0, then the maximum residual is zero too. The minimum number of calibration points is 6, and with less points there is slightly lower maximum residual.

Code:

```
% Repeat the homogeneous camera calibration exercise of Sect. 11.2.1.
% Investigate the effect of the number of calibration points, and noise
% on the calibration residual.

% This part is a repeat of MATLAB code in textbook, there is no easy
% replacement for Python, I would recommend you use MATLAB for this
% problem

P = mkcube(2);

% P = [[-1,-1,1,1,-1,-1];[-1,1,1,-1,-1,1,];[-1,-1,-1,-1,1,1]];

T_unknown = SE3(0.1, 0.2, 1.5) * SE3.rpy(0.1, 0.2, 0.3);
cam = CentralCamera('focal', 0.015, 'pixel', 10e-6, ...
        'resolution', [1280 1024], 'noise', 0.05); % gaussian noise

p = cam.project(P, 'objpose', T_unknown);
C = camcald(P, p);
```