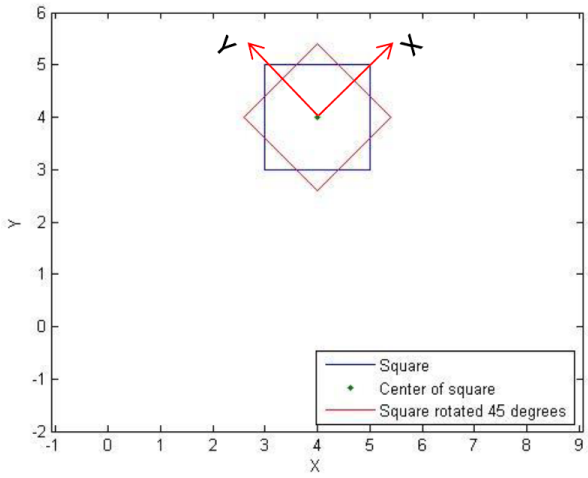
Computer Vision

Exercises of Lab 3

Exercise 3.1: Rotating a square in 2D

The goal of this exercise is quite simple; rotate a square by 45° around its center point (xc, yc) = (4, 4). To do this, read and modify the Matlab script Exercise3.1.m

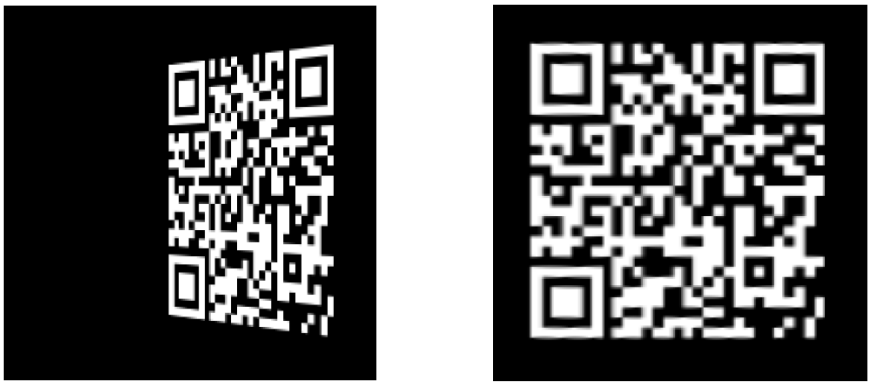
Hint: You can think of the exercise this way: suppose we are given two coordinate systems (X,Y) and (X',Y'), e.g., the world and camera coordinate systems. Coordinate system (X',Y') is centered at (xc, yc) and rotated 45° with respect to (X,Y). The exercise involves having to find a transformation that maps points in (X,Y) to points in (X',Y'), as depicted in the figure below.



Try to expand the code to handle 3D-coordinates. Uncomment the code at the bottom of *Exercise3.1.m* and implement a transformation that rotates a cube around its center point. The rotation should be a combination of a 70° rotation about the z-axis, a 45° rotation about the y-axis, and a 20° rotation about the x-axis. Does is matter in which order we apply the three rotation matrices? More information here.

Exercise 3.2: Image rectification

The goal of this exercise is to rectify the QR barcode displayed below. To do this, read and modify the Matlab script Exercise3.2.m.



Exercise 3.3: Points and lines

The goal of this exercise is to understand and utilize the point and line duality. We want to find two vanishing points and the horizon in an image based on corner points of a building.

Run the Matlab script Exercise3.3.m and implement the missing code in step 1-4. Find the height of the camera above the ground. As a reference, the wall marked on the image is measured to be 3.2 meters.

Exercise 3.4: Camera calibration

The goal of this exercise is to calibrate a camera using multiple correspondences of 2D image coordinates and 3D world coordinates of a known object.

First, download the Camera Calibration Toolbox for Matlab from vision.caltech.edu. Also, download Peter Kovesis' Computer Vision Matlab functions – direct link here.

Open Exercise3.4.m and change the variables *VLFEATROOT*, *CALIBROOT*, and *MATLABFNSROOT* to the unpacked paths at your hard drive.

Run the code and point out the four corners (clockwise from the upper left corner) in the first frame. Read the code and understand exactly what happens in the subsequent frames.

Inspect Figure 2 (3D) carefully and click the button "Switch to world-centered view". When would you use a camera-centered view and when would you use a world-centered-view?

Plot a projection of an (x,y,z)-axis on top of the images. Use the estimated intrinsic and extrinsic parameters from the camera calibration. Slide 17-19 in Lecture3.1-2_ImageFormation describe the order of transformations required for transforming from 3D world coordinates to 2D image coordinates. The parameters returned from the camera calibration toolbox are described here.

NOTE! The convention of the extrinsic parameters in the toolbox differs from the convention in slide 13-14. The toolbox uses the following convention:

$$P_{camera} = R \times P_{world} + T$$

where R is a rotation matrix, T is a translation vector, and P_{world} and P_{camera} are nonhomogeneous 3D coordinate vectors.

How does this influence the formulas in slide 17-19?