Exercise 1

Deadline: 14 November, 2020

Please send your results as a zip file either to **tewodros_amberbir.habtegebrial@dfki.de** or **torben.fetzer@dfki.de**.
Use the **Latex template** provided in Exercise_0 for the report.
Hand-written results will not be accepted!

Theory

1. Properties of Rotation Matrices

Rotation matrices are orthogonal matrices with determinant 1. Therefore, the following properties hold true for any rotation matrix $\mathbf{R}$:

$$\mathbf{R}^{-1} = \mathbf{R}^{\mathsf{T}} \qquad \text{and} \qquad \det(\mathbf{R}) = 1 \tag{1}$$

(a) Show, that the rows and columns of $\mathbf{R}$ are orthonomal (orthogonal and of length 1).

(b) Any rotation matrix $\mathbf{R}$ can be represented by a sequence of Euler angles, which are rotations around the coordinate axes. Show that the matrices $\mathbf{R}_z(\psi)$, $\mathbf{R}_y(\theta)$ and $\mathbf{R}_x(\phi)$, as defined in the lecture, fulfill the properties of rotation matrices for any angles ϕ, θ and ψ. Prove that properties (1) also hold true for $\mathbf{R} = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)$ as the consecutive execution of these matrices.

(c) What is the geometric interpretation of the determinant of a square 3×3 matrix? Why does a rotation matrix have to have determinant 1? Hint: Compare the determinants of base transformations (e.g. a rotation, translation,...).

2. Transformation Chain

Describe the transformation chain for mapping a point from the world coordinate system to the pixel coordinate system of an intrinsically and extrinsically calibrated camera. Why are homogeneous coordinates used for transforming points between coordinate systems? Use formulas and explain the intermediate steps in words.

Implementation

The images directory contains images of a chessboard that were used for calibrating a camera with high radial distortion. The results of the calibration (intrinsics of the camera and extrinsics for each board) are stored in `data/ex1.mat`. You are asked to

(a) Write a function `project_points` for projecting all 3D-points defined by a chessboard (in the world coordinate system) to the 2D pixel coordinate system. It should optionally regard the radial distortion (k1, k2, k5). The function takes as input a vector of 3D world points, the camera's intrinsic and extrinsic parameters and a flag for considering the distortion. It returns the projected 2D-points.

(b) Write a function `project_and_draw` that takes an image and projects and draws all chessboard points onto that image using `project_points`. **Apply this to all images** after compensating for radial distortion.

(c) In your report, show the first image with the following information:
- projected points without correction of the distortion in red
- projected points with correction of the radial distortion (k1, k2 and k5) in green

Remarks:

- Use the template `main.py` to load the data and fill in your code.
- Extend function `project_and_draw` to take color into account.
- Directory `data/sample_results` shows the expected output applied to one of the images.
- Distortion coefficients influence the points in the following way:
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_d \\ y_d \end{pmatrix} \cdot (1 + k_1 r^2 + k_2 r^4 + k_5 r^6)$$
- Distortion must be applied in the normalized image coordinate system. $\mathbf{K}^{-1}(x, y, 1)^\mathsf{T}$ transforms the point $(x, y)^\mathsf{T}$ from the pixel coordinate system to the normalized image coordinate system. After the distortion was applied, the point needs to be transformed back to the pixel coordinate system (multiplication by $\mathbf{K}$).
- Remember: we must be able to run your application by calling `python3.x main.py`.

Good Luck!