Exercise 10 for MA-INF 2201 Computer Vision WS23/24
21.01.2024
Submission deadline on 28.01.2024

1. The folder data contains an image pair (Building_facade_01, Building_facade_02):
   - Measure for this image pair the pixel coordinates of at least 6 identical points on the facade interactively (see Fig. 1). You can use the functions (*click_event* and *extract_points*) to extract the coordinates. *(1 point)*
   - Compute the homography transformation **H** using homogeneous coordinates. Compute the errors between the transferred and measured points and print the alignment errors. Use your own implementation following the lecture notes to compute the **H** transformation. *(2 points)*
   - Use the transformation **H** to map the source image into the geometry of the target image. For this task you can use the function *cv2.warpPerspective*. *(1 point)*

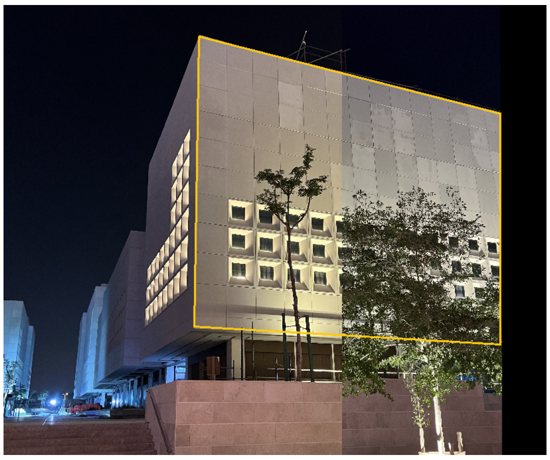

Figure 1: Wrapped source image into the geometry of the target image.

2. Compute the perspective transformation to warp the image (Book) as shown in Fig. 2. Select 4 points interactively and use your own implementation following the lecture notes to compute the perspective transformation. Keep the aspect ratio considering that the book has the following dimension (19.7 cm x 12.9 cm). Visualize the two images as shwon in Fig. 2. You can use the function *cv2.warpPerspective* to wrap the image. *(4 points)*

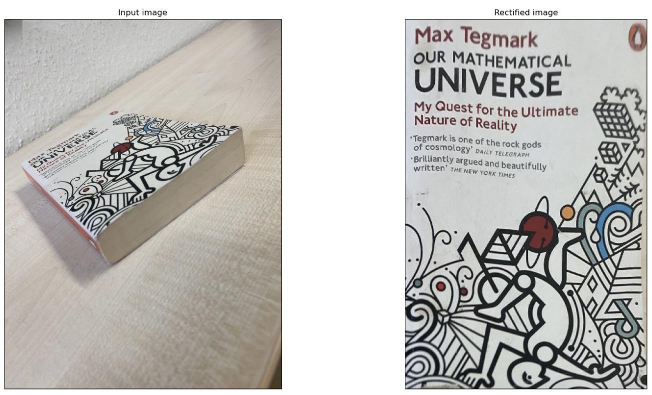
Input image
Rectified image
Max Tegmark
OUR MATHEMATICAL
UNIVERSE
My Quest for the Ultimate Nature of Reality
'Tegmark is one of the rock gods of cosmology' DAILY TELEGRAPH
'Brilliantly argued and beautifully written' THE NEW YORK TIMES

Figure 2: Rectified image.

3. For the image pair (Mountain_01, Mountain_02):

   - Use the SIFT algorithm to extract key-points and their corresponding descriptors for the images. You can use the class *SIFT* from OpenCV. *(0.5 points)*
   - Implement the best match ratio test with the threshold 0.3 as follows: Compute the square of the Euclidean distance between all pairs of key-points across the two images. For each key-point, find the most similar and the second most similar key-point from the other image. Only keep key-points which are consistent in both ways and pass the ratio test. Display your matching key-points using the function *cv2.drawMatchesKnn. (1.5 points)*
   - Implement RANSAC to compute a transformation **H** between the two images Mountain_2 and Mountain_1, where **H** is a projective transformation and can be defined by a set of four pairs of matched key-points. You can use the function *cv2.getPerspectiveTransform* to calculate the projective transformation and *cv2.warpPerspective* to wrap the image. *(1.5 points)*
   - Stitch Mountain_1 and the transformed Mountain_2 to obtain an image mosaic similar to the one in Fig. 3. *(0.5 points)*

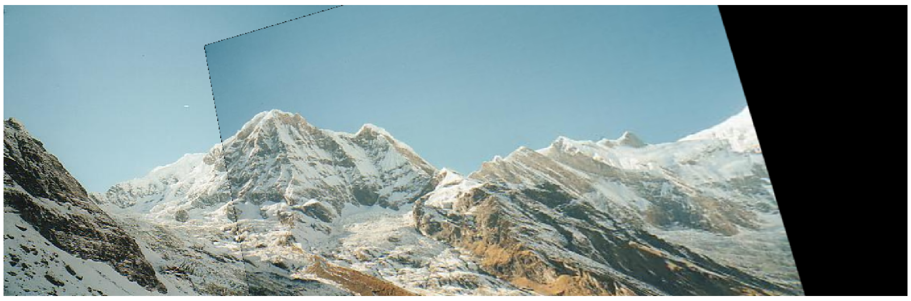

Figure 3: Feature based image rectification.

4. Consider the following projection matrix:

$$\mathbf{P} = \begin{bmatrix} -14066.493447 & -9945.538111 & 4796.664032 & 8463868675.444821 \\ 9900.140035 & -14625.938658 & 590.458246 & -176902714.499046 \\ 0.113060 & -0.051300 & 0.992263 & -33765.746644 \end{bmatrix}$$

   - Decompose the projection matrix **P** into the extrinsic and intrinsic camera parameters. Consider that the pixel size is 2 µm in the horizontal and vertical directions. *(3 points)*
   - Check the decomposition by reconstructing the projection matrix from the intrinsic and extrinsic camera parameters obtained from the decomposition. *(1 point)*
   - Assume that the given projection matrix corresponds to a camera fixed on a flying UAV above the ground. Given that the flight height is 100 meters and the image width and height are 7360 and 4912 pixels, respectively, compute the ground sampling distance as well as the image footprint on the ground. *(1 point)*

5. Extract the intrinsic camera parameters along with the distortion parameters using a checkerboard-based camera calibration. For this task you have to calibrate your own camera using your own photos of a checkerboard (save the images in a folder *camera_calibration*). Print the results of the calibration in a readable way and compare i.e. the focal length with the given camera specifications. To check the calibration compute the mean re-projection error considering a real scale of objects. You can use the predefined OpenCV functions for the task. *(4 points)*

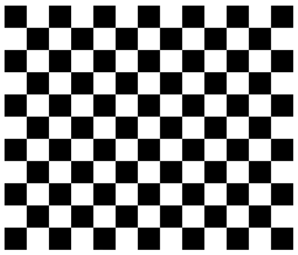
Figure 4: Exemplar checkerboard for camera calibration.

6. In the folder data an image pair is provided (Uni_Bonn_01, Uni_Bonn_02) along with the corresponding points list (corr.txt):
   - Compute the Fundamental matrix **F** using the eight-point algorithm. Use your own implementation following the lecture notes. *(1.5 points)*
   - Check the transformation using the Fundamental matrix **F** and print the corresponding errors. *(0.5 points)*
   - Visualize the corresponding points and the resulting Epipolar lines. *(1 point)*
   - Perform image rectification on the image pair and show the results as in Fig. 5. You can use the functions *cv2.warpPerspective* and *cv2.stereoRectifyUncalibrated*. *(1 point)*

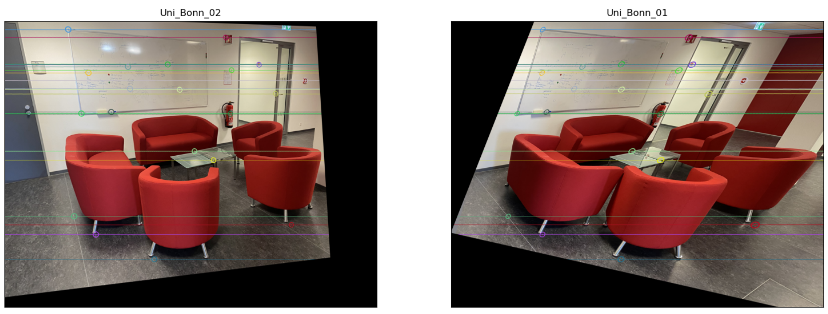

Figure 5: Stereo rectification

How to report on this assignment. Please exclude the data folder from your submission due to its size.

Questions If you have questions, find any ambiguities/mistakes, or if you don't have a camera for the camera calibration (task 5) please contact: M. H. Shams Eddin (shams@iai.uni-bonn.de).

Important: You are **not** allowed to use any additional python modules beyond the ones imported in the template. Otherwise you won't get any points.

Grading: Submissions that generate runtime errors or produce obviously rubbish results (e.g. nans, inf or meaningless visual outputs) will receive at most 50% of the points.

Plagiarism: Plagiarism in any form is prohibited. If your solution contains code directly copied from any source (e.g. other students), you will receive **0 points** for the entire exercise sheet.

Submission: You can complete the exercise in a group of two, but only one submission per group is allowed. Include a *README.txt* file with your group members into each solution. Points for solutions without a README file will only be given to the uploader.