Exercise Sheet 7: Matching

Due on 04.07.2025, 10:00

Stefan Baumann (stefan.baumann@lmu.de)

Task 1: Image Matching (8P)

The goal of this task is to estimate the transformation between two given images. Make sure you have installed the *scikit-image*[1] library, as this task makes heavy use of it.

- Load images *view1.jpg* and *view2.jpg* from the data folder and extract keypoints and features with the SIFT extractor[2] (0.5P)
- Match the detected keypoints by using the *match_ descriptors*[3] method (0.5P)
- Implement a least-squares affine transformation solver and use it to estimate the transform between the matched keypoints (6P)
- Look at the visualization of the warped keypoints and the stitched image. Describe what you see. Does the transformation look good? (1P)

The affine transformation is of the following form (subscript s denotes source points, t target points):

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{pmatrix} \cdot \begin{pmatrix} x_s \\ y_s \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = \mathbf{A}\vec{\mathbf{x}}_s + \vec{\mathbf{t}} \tag{1}$$

Task 2: Robust Affine Transformation With RANSAC (6P)

In the lecture, you have already learned the underlying idea and algorithm of RANSAC (Random Sample Consensus). Within the scope of this task, you should implement the RANSAC algorithm on your own and apply it to achieve a more robust affine transformation estimation. We are building on the same data and extracted features from **Task 1**.

- Implement the RANSAC algorithm for the special case of an affine transformation and use it to find a better affine transform than in **Task 1**. (5P)

[1] https://scikit-image.org/

[2] https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_sift.html

[3] https://scikit-image.org/docs/stable/api/skimage.feature.html#skimage.feature.match_descriptors

- Again, look at the visualization of the warped keypoints and the stitched image. Compare these results to those from **Task 1**. What has changed, and why? (1P)

Task 3: Optical Flow Based Image Warping **(6P)**

In this task, we aim to estimate dense optical flow information and warp images based on it. Your starting point is the video file *video.mp4*, which you can find in the data folder.

- Load a pretrained RAFT model (DEFAULT weights) via torchvision and implement flow estimation using it. (1P)
- Implement a warping method that takes source & target frames and backward flow, and then tries to approximate the target frame from the source frame & flow[4]. *Hint: use the grid_sample method from PyTorch.* (4P)
- Which artifacts can you see? Try to explain what caused them. (1P)

Important Note: Submit exactly one ZIP file via Moodle before the deadline. Please submit your solution as a Jupyter Notebook. The ZIP file should contain your notebook and the images/videos you are loading. Make sure that it runs on different operating systems and uses relative paths. Non-trivial sections of your code should be explained with short comments, and variables should have self-explanatory names. The notebook file should contain your written code, all figures, explanations and answers to questions.

[4] detailed description of how this can be achieved: https://arxiv.org/abs/1506.02025