Technische Universität München
Fakultät für Informatik
PD Dr. Slobodan Ilic

WS 15/16
Exercise Sheet 7
December 10th, 2015

Exercises in Tracking & Detection

In this exercise sheet, we will implement a camera-tracking algorithm based on SIFT and non-linear least square optimization. Specifically, a camera is moving around a *static* object, and we would like to infer the 6 degree-of-freedom camera poses: rotations $\mathbf{R} \in SO(3)$ and translations $\mathbf{T} \in \mathbb{R}^3$. The following exercises lead us to accomplish this goal step by step.

For the computation of SIFT, RANSAC, DLT or homography, you can use either your own implementation from previous exercises or any other implementation on the web (at your own risk). Again, we recommend `vlfeat` for SIFT but you are free to use any toolbox you find comfortable, as long as it is compatible to `MATLAB`.

Exercise 1 Initial Pose Estimation

First, one has to initialize the tracking:

a) Initializing the parameters: we set $\mathbf{R}_0 = \mathbf{I}_{3\times3}$ and $\mathbf{T}_0 = \mathbf{0}$ as the initial pose of the camera, where $\mathbf{I}_{3\times3}$ is the 3×3 identity matrix. The intrinsic matrix $\mathbf{A}$ of the camera is provided as follows and remains fixed for the whole sequence.

$$\mathbf{A} = \begin{bmatrix} 472.3 & 0.64 & 329.0 \\ 0 & 471.0 & 268.3 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

b) Initializing 2D keypoints $\mathbf{m}_i \in \Omega_2$: download the sequence $\{\mathcal{I}_t\}_{t=0}^n$ from the course webpage. The first image $\mathcal{I}_0$ shows a textured object parallel to the image plane of the camera. Please extract SIFT features within the area of the object in $\mathcal{I}_0$.

c) Initializing 3D points $\mathbf{M}_i \in \mathbb{R}^3$: back-project all the above 2D keypoints to 3D. To this end, simply project the obtained 2D SIFT keypoints $\tilde{\mathbf{m}}_i$ (homogeneous coordinates) to the virtual image plane ($z = 1$) by multiplying the inverse of the intrinsic matrix:

$$\mathbf{M}_i = \mathbf{A}^{-1}\tilde{\mathbf{m}}_i. \tag{2}$$

Note: you **do not** need to compute 3D points $\mathbf{M}_i$ for other images $\mathcal{I}_{t=1\ldots n}$, just the first one $\mathcal{I}_0$. They are assumed fixed throughout the whole sequence and will be associated to new 2D keypoints $\mathbf{m}_{i,t=1\ldots n}$ extracted in the next exercise. Now we further denote them as $\mathbf{M}_{i,0}$.

Exercise 2 Tracking SIFT Points

Next, we aim at finding the correctly-matched SIFT points (inliers) between the image $\mathcal{I}_0$ and the image $\mathcal{I}_t$:

a) In the image $\mathcal{I}_t$, compute again all SIFT points $\mathbf{m}_{i,t}$.

b) Match these SIFT points with the SIFT points $\mathbf{m}_{i,0}$ in the image $\mathcal{I}_0$. Use only the points for which you have already computed their 3D coordinates $\mathbf{M}_{i,0}$ in Exercise 1.

c) Compute all inliers using RANSAC, normalized DLT and homography, as you have done in Exercise Sheet 6.

To visualize the results, display $\mathcal{I}_0$ and $\mathcal{I}_{t=1\ldots n}$ side by side and draw lines between the correct point correspondences. Due to the time constraints, please save the image pairs with the visualized correspondences beforehand, such that one does not have to re-run everything during the correction time.

Exercise 3 Non-Linear Optimization and Pose Computation

Finally, we proceed to compute the current pose of the camera. So far, we have an initial pose $[\mathbf{R}_0, \mathbf{T}_0]$, the intrinsic matrix $\mathbf{A}$ and the filtered 2D point correspondence pairs between $\mathcal{I}_0$ and $\mathcal{I}_{t=1\ldots n}$. Additionally, we have the 3D coordinates $\mathbf{M}_{i,0}$ of the 2D feature points in $\mathcal{I}_0$. In order to compute the current camera pose $[\mathbf{R}_t, \mathbf{T}_t]$, we formulate an energy function $\mathbf{f}_t$ and apply non-linear optimization tools. One possible formulation of $\mathbf{f}_t$ is as follows:

$$\mathbf{f}_t(\mathbf{R}_t, \mathbf{T}_t; \mathbf{A}, \mathbf{M}_{i,0}, \mathbf{m}_{i,t}) = \sum_i \left\| \mathbf{A} \left(\mathbf{R}_t \mathbf{M}_{i,0} + \mathbf{T}_t \right) - \tilde{\mathbf{m}}_{i,t} \right\|^2 \tag{3}$$

a) Implement the energy function $\mathbf{f}$ in `MATLAB` that takes as input the rotation parameters (at your own choice), the translation parameters $\mathbf{T}$, the intrinsic matrix $\mathbf{A}$ and the 3D-2D correspondences $\mathbf{M}_i, \mathbf{m}_i$. Three things you should know:

- While being taken as input for $\mathbf{f}$, $\mathbf{A}$ and $\mathbf{M}_i, \mathbf{m}_i$ are known parameters; only rotations and translations are the *unknown* variables to be estimated.
- Although it is not explicitly written in Equ. 3, after projecting 3D points to the 2D image plane, one should always divide all the coordinates by z component.
- 3D rotations have only 3 degrees of freedom. Optimizing directly on a 9-element rotation matrix $\mathbf{R} \in SO(3)$ is not only unnecessary but also redundant. Choose your own parametrization, *e.g.* Euler angle $\mathbf{R}(\alpha, \beta, \gamma)$, unit quaternions $\mathbf{R}(\mathbf{q})$, or the exponential map $\mathbf{R}(\mathbf{w})$. Each of them comes with merits as well as drawbacks. We encourage you to try all of them and compare the outcomes.

b) Estimate the current pose $[\mathbf{R}_t, \mathbf{T}_t]$ at time t by using the 3D-2D correspondences $\mathbf{M}_{i,0}, \mathbf{m}_{i,t}$ and the pose parameter at time $(t-1)$ as initial values. This is done by minimizing $\mathbf{f}_t$ in Equ. 3, which is a non-linear least square problem. Read carefully the documentation of `fminsearch` function in `MATLAB` and apply it to minimize $\mathbf{f}_t$.

c) Repeat Exercise 3b for all images $\mathcal{I}_{t=1\ldots n}$ (namely, construct and solve all $\mathbf{f}_{t=1\ldots n}$). Show the trajectory of the camera in a 3D graph by plotting the camera coordinates in the world coordinate frame. The camera locations in the world coordinate frame at time t are obtained by computing $-\mathbf{R}_t^\top \mathbf{T}_t$. Again, save all these results in advance to avoid re-computing them during the correction.