Visual Navigation for Flying Robots
D. Cremers, X. Gao, V. Usenko
Summer Semester 2018

Computer Vision Group
Department of Informatics
Technical University of Munich

Exercise Sheet 3

Topic: State Estimation

Submission deadline: Sunday, 06.05.2018, 23:59 pm
Hand-in via email to visnav_ss2018@vision.in.tum.de

General Notice

The exercises should be done by yourself, but the final project should be done in teams of two to three students. We will use Ubuntu 16.04 in this lab course. It is already installed on the lab computers. If you want to use your own laptop, you will need to install Ubuntu by yourself.

Files related with the exercise will be placed in directories like 'paper/' or 'doc/'. Please read these materials before start answering the questions.

Exercise 1: Batch MAP

Consider a linear discrete time system:

$$\begin{aligned} x_k &= x_{k-1} + v_k + w_k, & w_k &\sim \mathcal{N}(0, Q) \\ y_k &= x_k + n_k, & n_k &\sim \mathcal{N}(0, R) \end{aligned} \tag{1}$$

which could represent a car moving back and forth along the x axis. The initial state is x_0 and time is from $k = 0, \ldots, 3$. Please derive the batch MAP (Maximum A Posteriori) for this system. First, we set batch state variable as $\mathbf{x} = [x_0, x_1, x_2, x_3]^{\mathrm{T}}$ and the batch observation data is: $\mathbf{z} = [x_0, v_1, v_2, v_3, y_1, y_2, y_3]^{\mathrm{T}}$. Then we can:

1. Define a matrix $\mathbf{H}$ such that the batch error can be written as:

$$\mathbf{e} = \mathbf{z} - \mathbf{H}\mathbf{x}. \tag{2}$$

   Please derive the exact form of $\mathbf{H}$.

2. With these notations we can turn the batch state estimation problem into a least square problem:

$$\mathbf{x}^* = \arg\min \frac{1}{2} (\mathbf{z} - \mathbf{H}\mathbf{x})^{\mathrm{T}} \mathbf{W}^{-1} (\mathbf{z} - \mathbf{H}\mathbf{x}), \tag{3}$$

   where $\mathbf{W}$ is the information matrix which can be derived from the MAP. Please show the exact form of $\mathbf{W}$ here.

3. Discuss if this problem has a unique solution if each noise item is uncorrelated.

Exercise 2: Iterative Curve Fitting

Consider a curve fitting problem where the curve model is:

$$y = \exp(ax^2 + bx + c) + w, \tag{4}$$

where a, b, c are parameters, x, y are data and w is the noise. We have $N = 100$ simulated data and want to estimate the parameters a, b, c with them. Please use the code/gn-curve-fitting.cpp to generate the data.

1. We want to use Gauss-Newton method to estimate the parameters. Define the error of i-th data as:

   $$e_i = y_i - \exp(ax_i^2 + bx_i + c). \tag{5}$$

   Then (a, b, c) can be estimated by solving a least square problem:

   $$\min_{a,b,c} \frac{1}{2} \sum_{i=1}^{N} \left\| y_i - \exp\left(ax_i^2 + bx_i + c\right) \right\|^2. \tag{6}$$

   Please complete the code/gn-curve-fitting.cpp to write a Gauss-Newton method.

2. Then let's try using an optimization library to solve the same problem. You can either use Google Ceres [1]or g2o[2] or both. The code frame work is provided in code/ceres-curve-fitting.cpp and code/g2o-curve-fitting.cpp. Please complete the code example to solve the problem, and compare the results with your own Gauss-Newton method.

Exercise 3: Camera Pose Estimation by Gauss-Newton Method

The camera pose can be estimated by minimizing the re-projection error. Assume we have a set of 3D points $\mathbf{P} = \{\mathbf{p}_i\}$ and their pixel positions $\mathbf{U} = \{\mathbf{u}_i\} \forall i = 1, \ldots n$. They are stored in code/p3d.txt and code/p2d.txt as text files. The intrinsic matrix is:

$$\mathbf{K} = \begin{bmatrix} 520.9 & 0 & 325.1 \\ 0 & 521.0 & 249.7 \\ 0 & 0 & 1 \end{bmatrix}.$$

Please use the provided data, write a Gauss-Newton iterative method to solve the camera pose. The initial guess is $\mathbf{T}_0 = \mathbf{I}$. The code framework is provided in code/gn-pose-estimate.cpp. When completing the codes, please answer the following questions:

1. How to define the re-projection error?
2. What is the Jacobian matrix of the error?
3. How to update the estimate?

[1] `http://ceres-solver.org/`

[2] `https://github.com/RainerKuemmerle/g2o`

Submission instructions

A complete submission consists both of a PDF file with the solutions/answers to the questions on the exercise sheet and a ZIP file containing the source code that you used to solve the given problems. Note all names of your team members in the PDF file. Make sure that your ZIP file contains all files necessary to compile and run your code, but it should not contain any build files or binaries. Please submit your solution via email to `visnav_ss2018@vision.in.tum.de`.