IN2106 Practical Course – Vision-based Navigation: Exercise #3

Min-An Chao (03681062) | TUM MS Informatics | ga83fok@mytum.de

06.05.2018

1 Batch MAP

Task 1 By taking $k = 0, 1, 2$ into the system model, We can obtain

$$\mathbf{H} = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

Task 2 From the system model and Eq. 1, we have,

$$\boldsymbol{e} = \boldsymbol{z} - \mathbf{H}\boldsymbol{x} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} - \begin{pmatrix} x_1 - x_0 \\ x_2 - x_1 \\ x_3 - x_2 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ n_1 \\ n_2 \\ n_3 \end{pmatrix} \sim \mathcal{N}(\mathbf{0}, \mathbf{W}). \tag{2}$$

Obviously, if the noise is independent of each other, we have

$$\mathbf{W} = \mathrm{diag}(Q, Q, Q, R, R, R), \tag{3}$$

otherwise $\mathbf{W}$ could be any covariance matrix with this diagonal terms and other nonzero entries.

Task 3 Yes, if the noise is uncorrelated, then it is independent. In this case there is only one exact solution of $\mathbf{W}$.

2 Iterative Curve Fitting

Task 1 The derivatives of error to parameters could be obtained by

$$\boldsymbol{J}_i = \begin{pmatrix} \frac{\partial e_i}{\partial a} \\ \frac{\partial e_i}{\partial b} \\ \frac{\partial e_i}{\partial c} \end{pmatrix} = \begin{pmatrix} -f(x_i) \cdot x_i^2 \\ -f(x_i) \cdot x_i \\ -f(x_i) \end{pmatrix}, \tag{4}$$

where

$$f(x_i) = \exp(ax_i^2 + bx_i + c). \tag{5}$$

Then we have

$$
\begin{aligned}
\mathbf{H} &= \sum_i \boldsymbol{J}_i \boldsymbol{J}_i^T, \\
\boldsymbol{b} &= \sum_i -e_i \boldsymbol{J}_i, \\
(\Delta a, \Delta b, \Delta c)^T &= \mathbf{H}^{-1}\boldsymbol{b}. \qquad (6)
\end{aligned}
$$

By implementing this, we can see the results:

```
total cost: 3.19575e+06
total cost: 376785
total cost: 35673.6
total cost: 2195.01
total cost: 174.853
total cost: 102.78
total cost: 101.937
total cost: 101.937
total cost: 101.937
total cost: 101.937
total cost: 101.937
total cost: 101.937
cost: 101.937, last cost: 101.937
estimated abc = 0.890912, 2.1719, 0.943629
```

Task 2 With `Google Ceres`, first we have to implement the $e_i = y_i - f(x_i)$ part in the overloaded `()` function, *i.e.* `operator()(const T *const abc, T *residual) const`, inside the `CURVE_FITTING_COST` object. then we instantiate a `Problem` object, adding the cost function of the `CURVE_FITTING_COST` object into a `CostFunction` object by `AutoDiffCostFunction()` method, and using `Problem::AddResidualBlock()` method to add the residual block we have implemented inside the overloaded `()` function. Finally by the solver setups shown in lecture slides and by the solver function call we have the results:

```
iter      cost      cost_change  |gradient|   |step|    ... total_time
   0  1.597873e+06    0.00e+00    3.52e+06   0.00e+00   ...   1.17e-02
   1  1.884440e+05    1.41e+06    4.86e+05   9.88e-01   ...   1.66e-02
   2  1.784821e+04    1.71e+05    6.78e+04   9.89e-01   ...   1.99e-02
   3  1.099631e+03    1.67e+04    8.58e+03   1.10e+00   ...   2.29e-02
   4  8.784938e+01    1.01e+03    6.53e+02   1.51e+00   ...   2.58e-02
   5  5.141230e+01    3.64e+01    2.72e+01   1.13e+00   ...   2.79e-02
   6  5.096862e+01    4.44e-01    4.27e-01   1.89e-01   ...   2.98e-02
   7  5.096851e+01    1.10e-04    9.53e-04   2.84e-03   ...   3.17e-02
Ceres Solver Report: Iterations: 8, Initial cost: 1.597873e+06,
Final cost: 5.096851e+01, Termination: CONVERGENCE
estimated a,b,c = 0.890908, 2.1719, 0.943628
```

3 Camera Pose Estimation by Gauss-Newton Method

Task 1 Suppose the rotation matrix of the estimated pose $\mathbf{T}_k$ is $\mathbf{R}_k$ and the translation part is $\boldsymbol{t}_k$, where k stands for the iteration count. The projected position of 3D points $\boldsymbol{p}_i$ would be

$$
\begin{aligned}
\boldsymbol{q}_{k,i} &= \mathbf{R}_{k-1}\boldsymbol{p}_i + \boldsymbol{t}_{k-1} \\
&= (x_{k,i}, y_{k,i}, z_{k,i})^T. \qquad (7)
\end{aligned}
$$

The re-projection error $\boldsymbol{e}_{k,i}$ is obtained by applying the intrinsic matrix to the projected position with the normalization term $z_{k,i}$, or equally, $q_{k,i}^{'(2)}$, which is,

$$
\begin{aligned}
\boldsymbol{q}'_{k,i} &= \mathbf{K}\boldsymbol{q}_{k,i}, \\
\boldsymbol{e}_{k,i} &= \boldsymbol{u}_i - \frac{1}{z_{k,i}} \cdot (q_{k,i}^{'(0)}, q_{k,i}^{'(1)})^T,
\end{aligned} \tag{8}
$$

Task 2 The Jacobian matrix of error could be derived, based on the slides, as

$$
\mathbf{J}_{k,i} = \begin{pmatrix} -\frac{f_x}{z_{k,i}} & 0 & \frac{f_x x_{k,i}}{z_{k,i}^2} & \frac{f_x x_{k,i} y_{k,i}}{z_{k,i}^2} & -f_x - \frac{f_x x_{k,i}^2}{z_{k,i}^2} & \frac{f_x y_{k,i}}{z_{k,i}} \\ 0 & -\frac{f_y}{z_{k,i}} & \frac{f_y y_{k,i}}{z_{k,i}^2} & f_y + \frac{f_y y_{k,i}^2}{z_{k,i}^2} & -\frac{f_y x_{k,i} y_{k,i}}{z_{k,i}^2} & -\frac{f_y x_{k,i}}{z_{k,i}} \end{pmatrix}, \tag{9}
$$

where $x_{k,i}, y_{k,i}, z_{k,i}$ comes from Eq. 7, and $f_x = K_{0,0}, f_y = K_{1,1}$.

Task 3 Updates on estimation pose are made iterative by

$$
\begin{aligned}
\mathbf{H}_k &= \sum_i \boldsymbol{J}_{k,i} \boldsymbol{J}_{k,i}^T, \\
\boldsymbol{b}_k &= \sum_i -e_{k,i} \boldsymbol{J}_{k,i}, \\
\Delta \mathbf{T}_k &= \mathbf{H}_k^{-1} \boldsymbol{b}_k, \\
\mathbf{T}_k &= (\Delta \mathbf{T}_k)^\wedge \mathbf{T}_{k-1}.
\end{aligned} \tag{10}
$$

$\Delta \mathbf{T}_k$ here is expressed as a $\mathfrak{se}(3)$ 6-by-1 vector. Updates can be applied by calculating its exponential map, and multiplying it to the previous pose $\mathbf{T}_{k-1}$. The results are shown below.

```
points: 76
iteration 0 cost=45538.2
iteration 1 cost=413.209
iteration 2 cost=302.649
iteration 3 cost=301.357
iteration 4 cost=301.351
iteration 5 cost=301.351
iteration 6 cost=301.351
iteration 7 cost=301.351
iteration 8 cost=301.351
cost: 301.351, last cost: 301.351
estimated pose:
  0.997866 -0.0516724  0.0399128  -0.127227
 0.0505959    0.99834  0.0275274 -0.0075068
-0.0412689 -0.0254492   0.998824  0.0613861
         0          0          0          1
```