# State Estimation for Robotics
# Assignment 4
# "Starry Night Dataset"

Jiaming Li —— 0501271

April 9, 2024

## 1 Variances of the Noise

1. Base on the data, is the assumption of zero-mean Gaussian noise reasonable? What values of the variances $Q_k$ and $R_k^j$, should we use? Pay attention to the fact that we have variable sampling period.

**Solution:**
According to the data and the histograms shown in the assignment document, we notice that both translational and rotational speeds have close to zero-mean errors. And for the measurements, the image pixels also have close to zero-mean errors.
Since we have variable sampling periods, we need to use the nominal sampling period to normalize the covariance matrices. Hence, the variances $Q_k$ and $R_k^j$ can be written as:

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_{v_x}^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{v_y}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{v_z}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\omega_1}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\omega_2}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\omega_3}^2 \end{bmatrix} T_K^2 = \begin{bmatrix} 0.0026 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0021 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0008 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0090 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0170 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1747 \end{bmatrix} T_K^2$$

$$\mathbf{R}_k^j = \begin{bmatrix} \sigma_{u_l}^2 & 0 & 0 & 0 \\ 0 & \sigma_{v_l}^2 & 0 & 0 \\ 0 & 0 & \sigma_{u_r}^2 & 0 \\ 0 & 0 & 0 & \sigma_{v_r}^2 \end{bmatrix} = \begin{bmatrix} 38.0046 & 0 & 0 & 0 \\ 0 & 129.8544 & 0 & 0 \\ 0 & 0 & 41.9633 & 0 \\ 0 & 0 & 0 & 132.5082 \end{bmatrix}$$

## 2 Objective Function of the LS-Optimization

In our estimation problem, we lift our problem into $SE(3)$, i.e. pose Lie group. So we need the following step to build up the objective function:
Firstly, we combine the translation vector $\mathbf{r}_i^{v_k i}$ and rotation matrix $\mathbf{C}_{v_k i}$ into a pose matrix:

$$\mathbf{T}_k = \mathbf{T}_{v_k i} = \begin{bmatrix} \mathbf{C}_{v_k i} & -\mathbf{C}_{v_k i}\mathbf{r}_i^{v_k i} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{1}$$

So the states we would like to estimate are changed from "translation and rotation separately" to "poses $T_k$".

$$\mathbf{x}_{k_1:k_2} = \left\{ \mathbf{r}_i^{v_{k_1} i}, \mathbf{C}_{v_{k_1} i}, ..., \mathbf{r}_i^{v_{k_2} i}, \mathbf{C}_{v_{k_2} i} \right\} = \left\{ \mathbf{T}_{v_{k_1} i}, ..., \mathbf{T}_{v_{k_2} i} \right\} \tag{2}$$

Correspondingly, we also need to concatenate $\nu_{v_k}^{iv_k}$ and $\omega_{v_k}^{iv_k}$ into a general velocity vector $\varpi$.

$$\varpi = \begin{bmatrix} \nu_{v_k}^{iv_k} \\ \omega_{v_k}^{iv_k} \end{bmatrix}. \tag{3}$$

The velocities(inputs) from timestep $k1$ to $k2$ can be written as:

$$\mathbf{v} = \{\check{\mathbf{T}}_{k_1}, \varpi_{k_1+1}, ..., \varpi_{k_2}\} \tag{4}$$

Where $\mathbf{T}_{k_1}$ is the prior pose at the beginning of the interval, and we use the groundtruth pose as the prior pose.
Then we assume that $\mathbf{M}_k$ represents the visible landmarks at timestep $k$, Then the measurements vector can be written as:

$$\mathbf{y} = \left\{\mathbf{y}_{k_1}^1, ..., \mathbf{y}_{k_1}^{M_{k_1}}, ..., \mathbf{y}_{k_2}^1, ..., \mathbf{y}_{k_2}^{M_{k_2}}\right\} \tag{5}$$

where each $\mathbf{y}_k^i$ represents the coordinates of feature $p_j$, projected into left and right image plane of the stereo camera $(u_l, v_l)$ and $(u_r, v_r)$ at timestep k.
Now we define the input error terms, For prior pose $\mathbf{T}_{k_1}$ and $\varpi_k$, we can write:

$$\mathbf{e}_{v,k}(\mathbf{x}) = \begin{cases} \ln(\check{\mathbf{T}}_{k_1}\mathbf{T}_k^{-1})^\vee & k = k_1 \\ \ln(\boldsymbol{\Xi}_k\mathbf{T}_{k-1}\mathbf{T}_k^{-1})^\vee & k = k_1+1, ..., k_2 \end{cases}. \tag{6}$$

where $\boldsymbol{\Xi}_k = exp(\Delta t_k \varpi_k^\wedge)$
And for measurement errors, we have:

$$\mathbf{e}_{y,jk}(\mathbf{x}) = \mathbf{y}_k^j - \bar{\mathbf{g}}(\mathbf{p}_{c_k}^{p_j c_k}) = \mathbf{y}_k^j - \bar{\mathbf{g}}(\mathbf{D}\mathbf{T}_{cv}\mathbf{T}_k\mathbf{p}_i^{p_j,i}) \tag{7}$$

where $\bar{\mathbf{g}}$ is the observation model that project point $\mathbf{p}_{c_k}^{p_j c_k}$ onto the $4 \times 1$ stereo camera measurements, so that we can subtract the real measurements with the predicted ones. And the other variables and matrices appear in the equation are:

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \mathbf{T}_{cv} = \begin{bmatrix} \mathbf{C}_{cv} & -\mathbf{C}_{cv}\rho_v^{cv} \\ \mathbf{0}^T & 1 \end{bmatrix}, \mathbf{p}_i^{p_j,i} = \begin{bmatrix} \rho_i^{p_j,i} \\ 1 \end{bmatrix} \tag{8}$$

According to the slides, we weight both errors with the inverse of their covariance matrices, namely $\mathbf{Q}_k^{-1}$ and ${\mathbf{R}_k^j}^{-1}$.
Finally we obtain our objective function:

$$J(\mathbf{x}_{k_1:k_2}) := \frac{1}{2}\mathbf{e}(\mathbf{x}_{k_1:k_2})^T\mathbf{W}^{-1}\mathbf{e}(\mathbf{x}_{k_1:k_2}), \tag{9}$$

Where vector $e$ is composed by input errors and measurement errors, and the weighted matrix $W^{-1}$ is a diagonal matrix.

$$\begin{aligned} \mathbf{e}(\mathbf{x}_{k_1:k_2}) &= \Bigg[\underbrace{\mathbf{e}_{v,k_1}(\mathbf{x}_{k_1:k_2}) \; ... \; \mathbf{e}_{v,k_2}(\mathbf{x}_{k_1:k_2})}_{\text{input errors}} \; \overbrace{\underbrace{\mathbf{e}_{y,1k_1}(\mathbf{x}_{k_1}) \; ... \; \mathbf{e}_{y,M_{k_1}k_1}(\mathbf{x}_{k_1})}_{\text{measurement errors at } k_1} \; ... \; \underbrace{\mathbf{e}_{y,1k_2}(\mathbf{x}_{k_2}) \; ... \; \mathbf{e}_{y,M_{k_2}k_2}(\mathbf{x}_{k_2})}_{\text{measurement errors at } k_2}}^{\text{measurement errors}}\Bigg] \\ \mathbf{W}^{-1} &= \text{diag}(\check{\mathbf{P}}_{k_1}^{-1} \; \mathbf{Q}_{k_1+1}^{-1} \; ... \; \mathbf{Q}_{k_2}^{-1} \; {\mathbf{R}_{k_1}^1}^{-1} \; ... \; {\mathbf{R}_{k_1}^{M_{k_1}}}^{-1} \; ... \; ... \; {\mathbf{R}_{k_2}^1}^{-1} \; ... \; {\mathbf{R}_{k_2}^{M_{k_2}}}^{-1}) \end{aligned}$$

# 3 Gauss-Newton Update

Our problem is indeed a Point-cloud tracking problem, and Gauss-Newton method is a good approach to tackle this problem, since Gauss-Newton method can be applied on the optimization problem of some linear system. Obviously the system we use contains exponential map and the nonlinear stereo camera sensor model, which are apparently nonlinear. So we need to linearize both the input error terms and measurement error terms.

Firstly we consider the input error terms. The state(i.e. pose) is:

$$\mathbf{T}_k = \exp\left(\epsilon_k^{\wedge}\right) \check{\mathbf{T}}_k. \tag{10}$$

According to the derivation in the slides, for the first input error, we have:

$$\mathbf{e}_{v,k_1}(\mathbf{x}) = \ln(\check{\mathbf{T}}_{k_1}\mathbf{T}_{k_1}^{-1})^{\vee} = \ln(\check{\mathbf{T}}_{k_1}\check{\mathbf{T}}_{\mathrm{op},k_1}^{-1}\exp(-\epsilon_{k_1}^{\wedge}))^{\vee} \approx \mathbf{e}_{v,k_1}(\mathbf{x}_{\mathrm{op}}) - \epsilon_{k_1} \tag{11}$$

For later input error terms, the linearization process can be written as(copied from the slides):

$$\begin{aligned}
\mathbf{e}_{v,k}(\mathbf{x}) &= \ln\left(\boldsymbol{\Xi}_k\mathbf{T}_{k-1}\mathbf{T}_k^{-1}\right)^{\vee} && (12)\\
&= \ln\left(\boldsymbol{\Xi}_k\exp(\epsilon_{k-1}^{\wedge})\mathbf{T}_{\mathrm{op},k-1}\mathbf{T}_{\mathrm{op},k}^{-1}\exp(-\epsilon_k^{\wedge})\right)^{\vee} && (13)\\
&= \ln\left(\boldsymbol{\Xi}_k\mathbf{T}_{\mathrm{op},k-1}\mathbf{T}_{\mathrm{op},k}^{-1}\exp\left(\left(\mathrm{Ad}\left(\mathbf{T}_{\mathrm{op},k}\mathbf{T}_{\mathrm{op},k-1}^{-1}\right)\epsilon_{k-1}\right)^{\wedge}\right)\exp(-\epsilon_k^{\wedge})\right)^{\vee} && (14)\\
&\approx \mathbf{e}_{v,k}(\mathbf{x}_{\mathrm{op}}) + \underbrace{\mathrm{Ad}\left(\mathbf{T}_{\mathrm{op},k}\mathbf{T}_{\mathrm{op},k-1}^{-1}\right)}_{\mathbf{F}_{k-1}}\epsilon_{k-1} - \epsilon_k && (15)
\end{aligned}$$

where $\mathbf{e}_{v,k}(\mathbf{x}_{\mathrm{op}}) = \ln(\boldsymbol{\Xi}_k\mathbf{T}_{\mathrm{op},k-1}\mathbf{T}_{\mathrm{op},k}^{-1})^{\vee}$ is the error evaluate at the operating point.

For measurement errors, we have:

$$\begin{aligned}
\mathbf{e}_{y,jk}(\mathbf{x}) &= \mathbf{y}_k^j - \bar{\mathbf{g}}(\mathbf{p}_{ck}^{p_j c_k}) && (16)\\
&= \mathbf{y}_k^j - \bar{\mathbf{g}}(\mathbf{D}\mathbf{T}_{cv}\mathbf{T}_k\mathbf{p}_i^{p_j,i}) && (17)\\
&\approx \mathbf{y}_k^j - \bar{\mathbf{g}}\left(\mathbf{D}\mathbf{T}_{cv}\exp(\epsilon_k^{\wedge})\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i}\right) && (18)\\
&\approx \mathbf{y}_k^j - \bar{\mathbf{g}}\left(\mathbf{D}\mathbf{T}_{cv}(\mathbf{1}+\epsilon_k^{\wedge})\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i}\right) && (19)\\
&= \mathbf{y}_k^j - \bar{\mathbf{g}}\left(\mathbf{D}\mathbf{T}_{cv}\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i} + \left(\mathbf{D}\mathbf{T}_{cv}(\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i})^{\odot}\right)\epsilon_k\right) && (20)\\
&\approx \underbrace{\mathbf{y}_k^j - \bar{\mathbf{g}}\left(\mathbf{D}\mathbf{T}_{cv}\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i}\right)}_{\mathbf{e}_{y,jk}(\mathbf{x}_{\mathrm{op}})} - \underbrace{\left.\frac{\partial\bar{\mathbf{g}}}{\partial\mathbf{z}}\right|_{\mathbf{z}=\left(\mathbf{D}\mathbf{T}_{cv}\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i}\right)}\left(\mathbf{D}\mathbf{T}_{cv}(\mathbf{T}_{\mathrm{op},k}\mathbf{p}_i^{p_j,i})^{\odot}\right)}_{\mathbf{G}_{jk}}\epsilon_k && (21)
\end{aligned}$$

Where $\frac{\partial\bar{\mathbf{g}}}{\partial\mathbf{z}}$ is the Jacobian of the stereo camera measurement model. And $\frac{\partial\bar{\mathbf{g}}}{\partial\mathbf{z}}$ can be written as:

$$\frac{\partial\bar{\mathbf{g}}}{\partial\mathbf{z}} = \begin{bmatrix} \frac{f_u}{z} & 0 & -\frac{f_u x}{z^2} \\ 0 & \frac{f_v}{z} & -\frac{f_v y}{z^2} \\ \frac{f_u}{z} & 0 & -\frac{f_u(x-b)}{z^2} \\ 0 & \frac{f_v}{z} & -\frac{f_v y}{z^2} \end{bmatrix} \text{ with } \quad \mathbf{z} = \begin{bmatrix} x & y & z \end{bmatrix}^T \tag{22}$$

Then we define the following stacked quantities for Gauss-Newton update:

$$\delta\mathbf{x} = \begin{bmatrix} \epsilon_{k_1} & \epsilon_{k_1+1} & \dots & \epsilon_{k_2} \end{bmatrix}^T \tag{23}$$

$$\mathbf{e}(\mathbf{x}_{\text{op}}) = \left[\mathbf{e}_{v,k_1}(\mathbf{x}_{\text{op}}) \dots \mathbf{e}_{v,k_2}(\mathbf{x}_{\text{op}}) \;\; \mathbf{e}_{y,1k_1}(\mathbf{x}_{\text{op}}) \dots \mathbf{e}_{y,M_{k_1}k_1}(\mathbf{x}_{\text{op}}) \;\; \dots \;\; \mathbf{e}_{y,1k_2}(\mathbf{x}_{\text{op}}) \dots \mathbf{e}_{y,M_{k_2}k_2}(\mathbf{x}_{\text{op}})\right]^T \tag{24}$$

$$\mathbf{H} = \begin{bmatrix}
\mathbf{1} & & & & & & \\
-\mathbf{F}_{k_1} & \mathbf{1} & & & & & \\
 & -\mathbf{F}_{k_1+1} & \mathbf{1} & & & & \\
 & & \ddots & \ddots & & & \\
 & & & -\mathbf{F}_{k_2-1} & \mathbf{1} & & \\
\mathbf{G}_{1,k_1} & & & & & & \\
\mathbf{G}_{2,k_1} & & & & & & \\
\vdots & & & & & & \\
\mathbf{G}_{M_{k_1},k_1} & & & & & & \\
 & \mathbf{G}_{1,k_1+1} & & & & & \\
 & \mathbf{G}_{2,k_1+1} & & & & & \\
 & \vdots & & & & & \\
 & \mathbf{G}_{M_{k_1+1},k_1+1} & & & & & \\
 & & \ddots & & & & \\
 & & \ddots & & & & \\
 & & \ddots & & & & \\
 & & \ddots & & & & \\
 & & & \ddots & & & \\
 & & & \ddots & & & \\
 & & & \ddots & & & \\
 & & & \ddots & & & \\
 & & & & & \mathbf{G}_{1,k_2} & \\
 & & & & & \mathbf{G}_{2,k_2} & \\
 & & & & & \vdots & \\
 & & & & & \mathbf{G}_{M_{k_2},k_2} &
\end{bmatrix}, \tag{25}$$

$$\mathbf{W} = \text{diag}\left(\check{\mathbf{P}}_{k_1} \; \mathbf{Q}_{k_1+1} \dots \mathbf{Q}_{k_2} \;\; \mathbf{R}_{k_1}^1 \dots \mathbf{R}_{k_1}^{M_{k_1}} \;\; \dots \;\; \dots \;\; \mathbf{R}_{k_2}^1 \dots \mathbf{R}_{k_2}^{M_{k_2}}\right) \tag{26}$$

The approximation of the objective function $J(\mathbf{x})$ is:

$$J(\mathbf{x}) \approx J(\mathbf{x}_{\text{op}}) - \mathbf{b}^T\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T\mathbf{A}\delta\mathbf{x} \tag{27}$$

where

$$\mathbf{A} = \mathbf{H}^T\mathbf{W}^{-1}\mathbf{H}, \quad \mathbf{b} = \mathbf{H}^T\mathbf{W}^{-1}\mathbf{e}(\mathbf{x}_{\text{op}}) \tag{28}$$

With Gauss-Newton method we minimizing the objective function with respect to $\delta\mathbf{x}^*$, we have:

$$\mathbf{A}\delta\mathbf{x}^* = \mathbf{b} \tag{29}$$

Solving this set of linear equations we have:

$$\delta\mathbf{x}^* = \begin{bmatrix}\epsilon_{k_1}^* & \epsilon_{k_1+1}^* & \dots & \epsilon_{k_2}^*\end{bmatrix} \tag{30}$$

We can use Cholesky decomposition or some other methods to obtain the optimal perturbation:
Finally we retract the optimal perturbation on the left to get the new operating point $\mathbf{T}_{\text{op},k}$.

$$\mathbf{T}_{\text{op},k} \leftarrow \exp\left(\epsilon_k^{*\wedge}\right)\mathbf{T}_{\text{op},k} \tag{31}$$

# 4 Plot of visible Landmarks

The numbers of visible landmarks at each timestep is shown in Fig 1. We use green dots for those timesteps for which there are at least 3 visible landmarks and red dots otherwise.

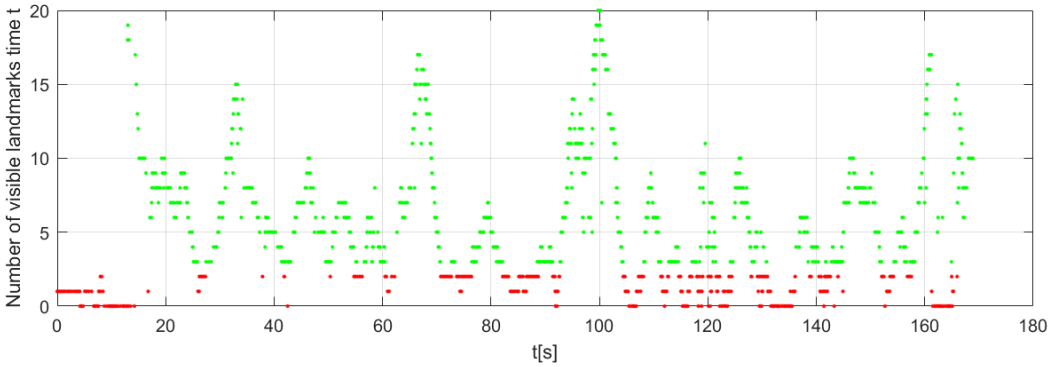


Figure 1: Number of visible landmarks at each time-step.

# 5 Gauss-Newton Updating method

The results of all 3 methods are shown in the following plots.

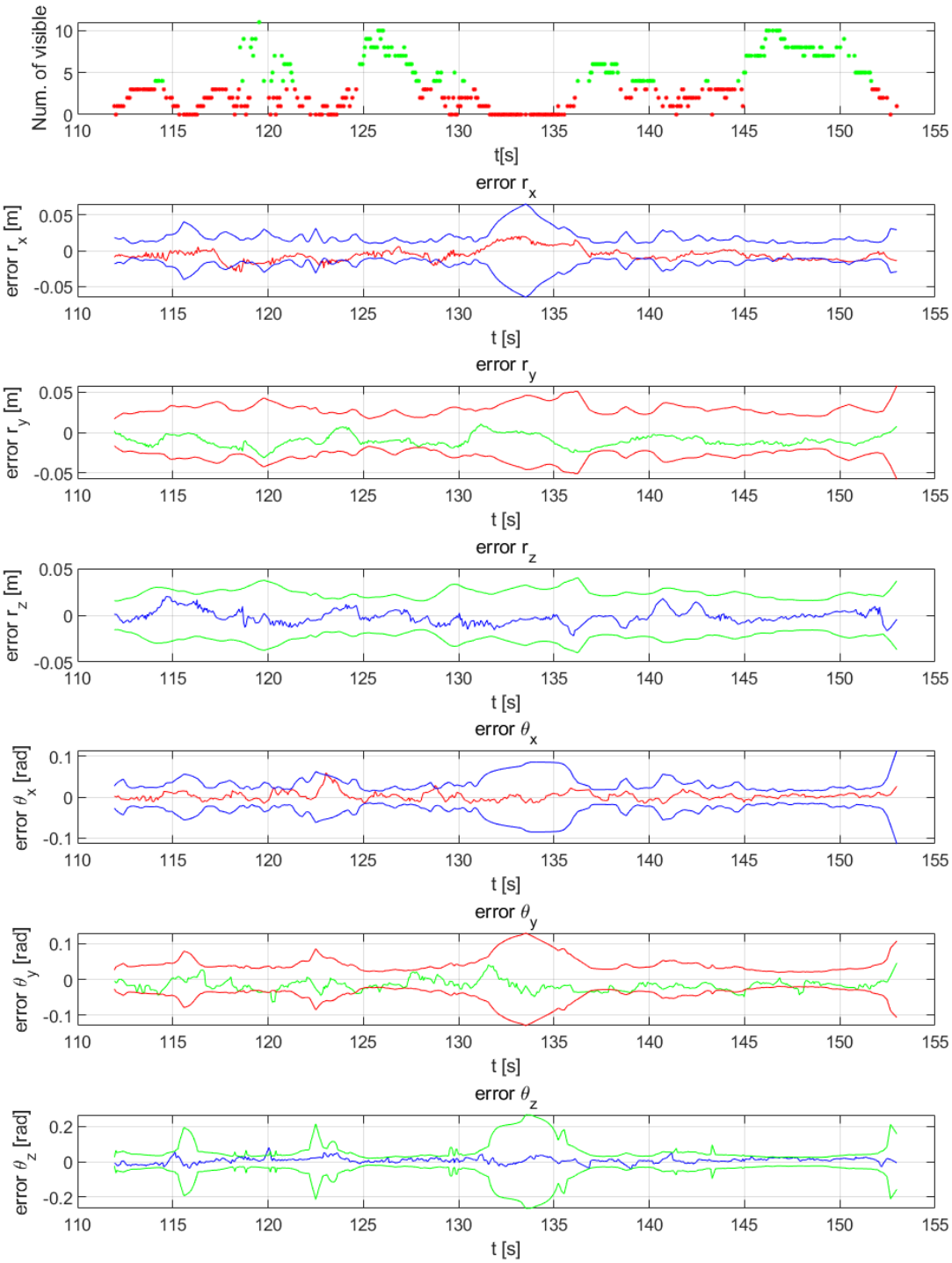


Figure 2: Results analysis of Batch solution.

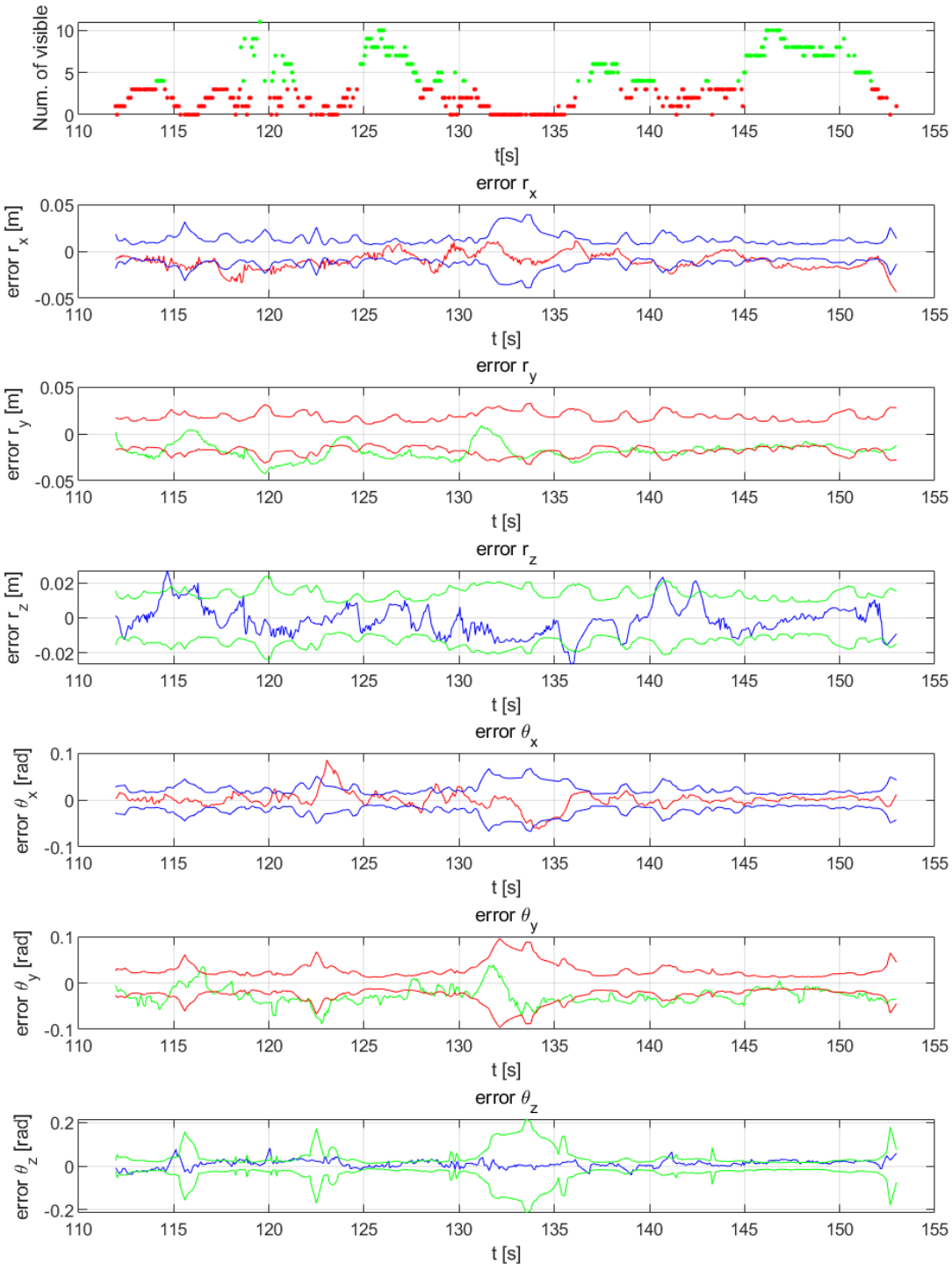


Figure 3: Results analysis of Sliding Window ($\kappa = 50$) solution.

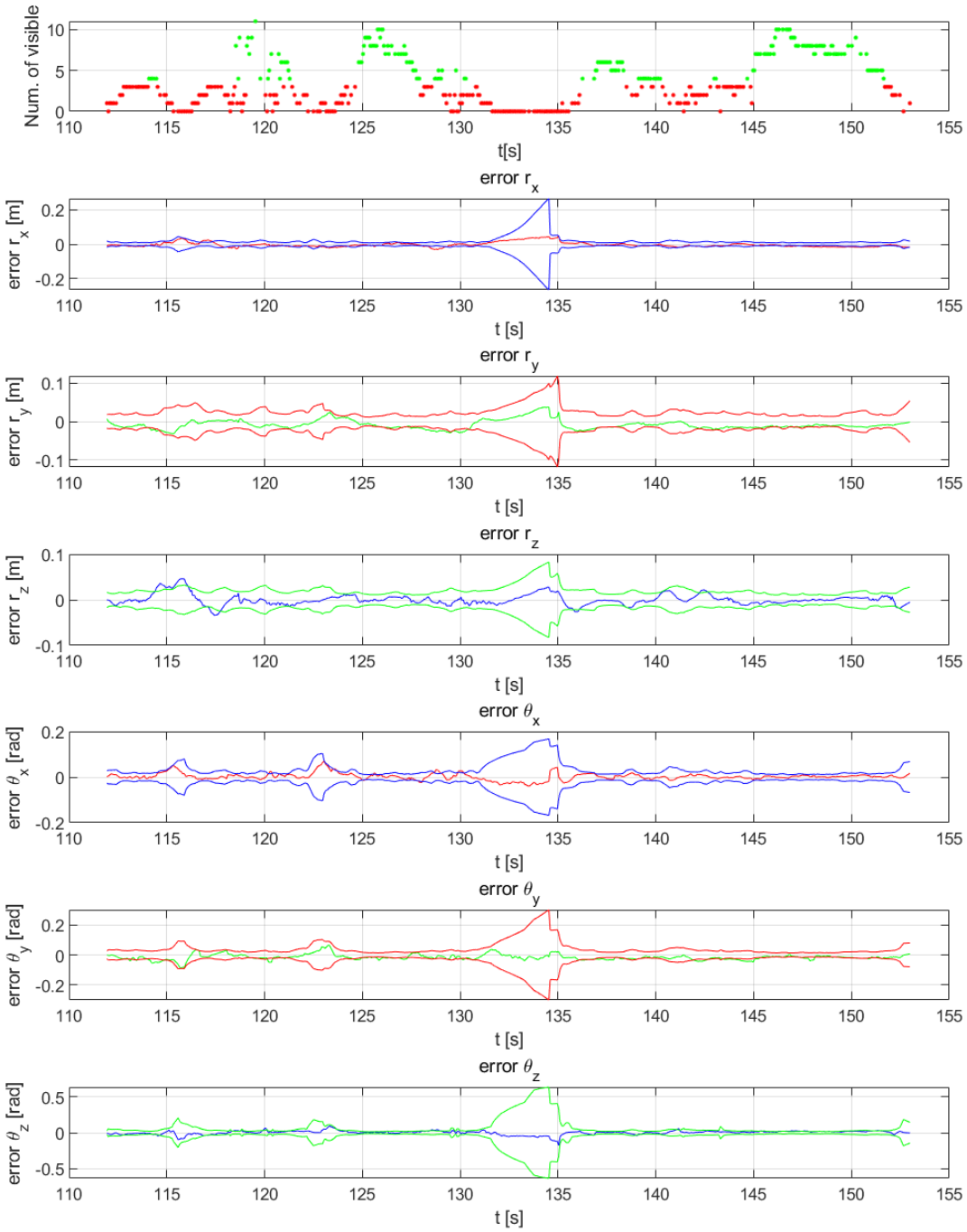


Figure 4: Results analysis of Sliding Window ($\kappa = 10$) solution.

**From these plots we can obtain several conclusions:**

(1) The relationship between error magnitude and number of visible landmarks
From all 3 plots we can see that the uncertainty envelope becomes much larger when there are fewer visible landmarks. The reason is that we don't have enough measurements to correct the dead reckoning results. Especially in time interval 133 126 [s], there are fewest visible landmarks during this period and the uncertainty is especially large.

(2) The advantages of BATCH solution

By comparing all 3 plots we can observe that BATCH solution has the best accuracy, since BATCH solution uses all information in the past and doesn't apply any Markov property. However, the uncertainty envelope of BATCH solution is much larger than the SLIDING-WINDOW ones. That is because BATCH solution propagates the uncertainty through the whole trajectory in each iteration and the uncertainty accumulates.

(3) The advantages of SLIDING-WINDOW solution

By comparing all 3 plots we can observe that SLIDING-WINDOW solution has the worse accuracy, since SLIDING-WINDOW solution applies Markov property and this will cause inaccurate results. However, the uncertainty envelope of SLIDING-WINDOW solution is much smaller than the BATCH one. That is because SLIDING-WINDOW solution propagates the uncertainty only through part of the trajectory in each iteration. Hence, the uncertainty of our estimation won't accumulate that much.

(4) Computational efficiency

The computation time of each method are shown in Table.1. We can see that the SLIDING-WINDOW method is more efficient than BATCH solution. Besides, the smaller window size we set, the faster the code runs. But we can observe that for $\kappa = 50$, SLIDING-WINDOW method is slower than BATCH method. The reason is we actually did more iterations totally than BATCH method. So window size $kappa = 50$ is not appropriate for this problem, since both the accuracy and the computational efficiency drops.

| | **BATCH** | **SLIDING-WINDOW** $\kappa = 50$ | **SLIDING-WINDOW** $\kappa = 10$ |
|---|---|---|---|
| computation time [s] | 50.28 | 100.97 | 9.20 |

Table 1: Computation time of each method.