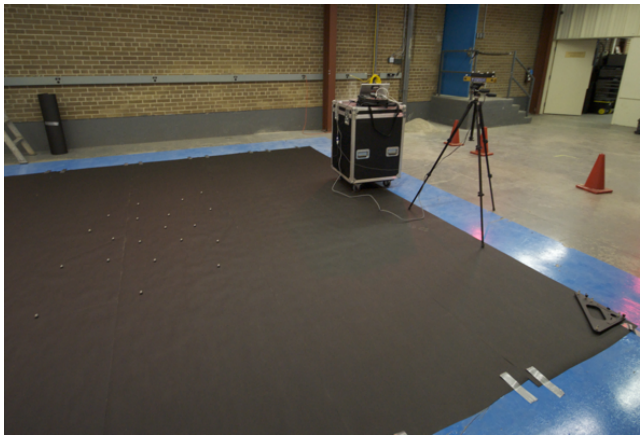
Assignment 3

‘Starry Night Dataset’

In this third assignment we’ll investigate a nonlinear three-dimensional problem consisting of a vehicle equipped with a stereo camera and inertial measurement unit (IMU) flying above a map of point features. We wish to estimate the position/orientation of this vehicle throughout its ≈ 44.3 m traverse. We’ll use the batch Gauss-Newton method to fuse speed measurements from the IMU with point measurements from the stereo camera.

3.1 Experimental Setup

The setup consists of a sensor head being translated/rotated above a set of reflective markers as depicted in Figure 3.1. The sensor head included a stereo camera and IMU rigidly attached to each other. Both the sensor head and the reflective markers were tracked using a ten-camera motion capture system. This motion capture system is able

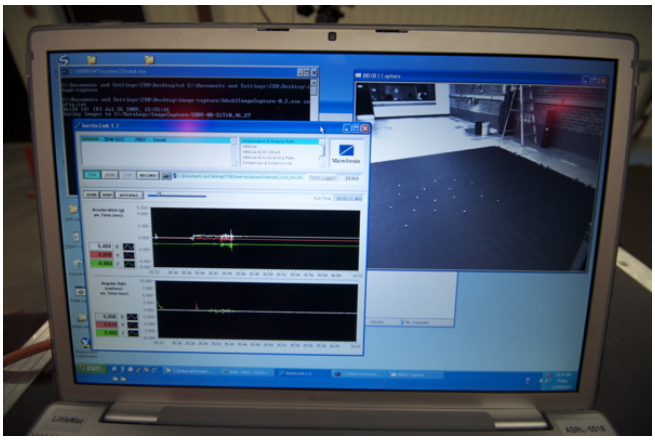


(a) A sensor head on the end of a tripod was translated/rotated above a map of point landmarks (white dots on the black floor).

(b) Vicon motion capture lab. Ten cameras work together to track markers on the sensor head and provide groundtruth position/orientation.

Figure 3.1: Setup for Dataset 3.

(a) The sensor head has an IMU to measure translational/rotational speeds and a stereo camera to measure the positions of point landmarks. Both sensors are noisy.



(b) Laptop screen shows what the stereo camera sees (left image only) as well as the IMU data capture software.

Figure 3.2: Sensor head.

to provide the position of each reflective marker to within a few millimeters. Position estimates from this motion capture system are considered to be quite a bit more accurate than estimates based on the sensor head and thus it serves as the benchmark/groundtruth in our experiment.

The sensor head was translated/rotated at the end of a tripod for approximately 3 minutes and three streams of data were logged:

- stereo image pairs, each image 640×480 resolution, logged at approximately 15 Hz
- the sensor head's velocity based on IMU measurements logged at approximately 100 Hz
- groundtruth position and orientation of a set of markers on the sensor head, logged at approximately 40 Hz

Figure 3.3 shows the sensor head's path over the 3 minute trial. We see that its path is very twisted and looping. We must estimate the sensor head's position and orientation as it moves in this three-dimensional environment.

The 20 landmarks used in this dataset were reflective markers placed on a black background. The positions of these were measured using the motion capture system for 3 minutes and then averaged. The output of these steps can be seen in Figure 3.3.

As with most real datasets, our data streams do not arrive synchronously or with evenly-spaced timesteps. Figure 3.4 shows the variability in the sampling periods of the nominally 15 Hz stereo image capture. Because of the variability in the camera sampling period, we cannot assume a uniform sampling rate. However, as with the other datasets, the velocities derived from the IMU data and the groundtruth values have been interpolated at the image times. Moreover, to avoid having to process the raw stereo images, stereo features were extracted from the images and associated with the landmarks using groundtruth data. Figure 3.5 shows how an example stereo image pair, and the extracted stereo features. Figure 3.6 and 3.7 show histograms of the IMU and stereo camera sensor errors based on groundtruth position/orientation.

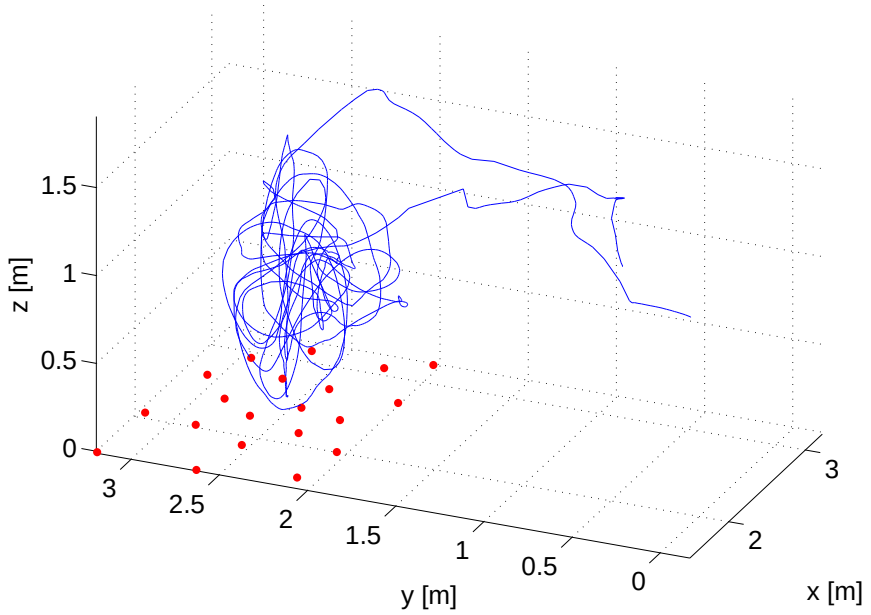


Figure 3.3: Groundtruth sensor head trajectory and 20 landmarks for Dataset 3.

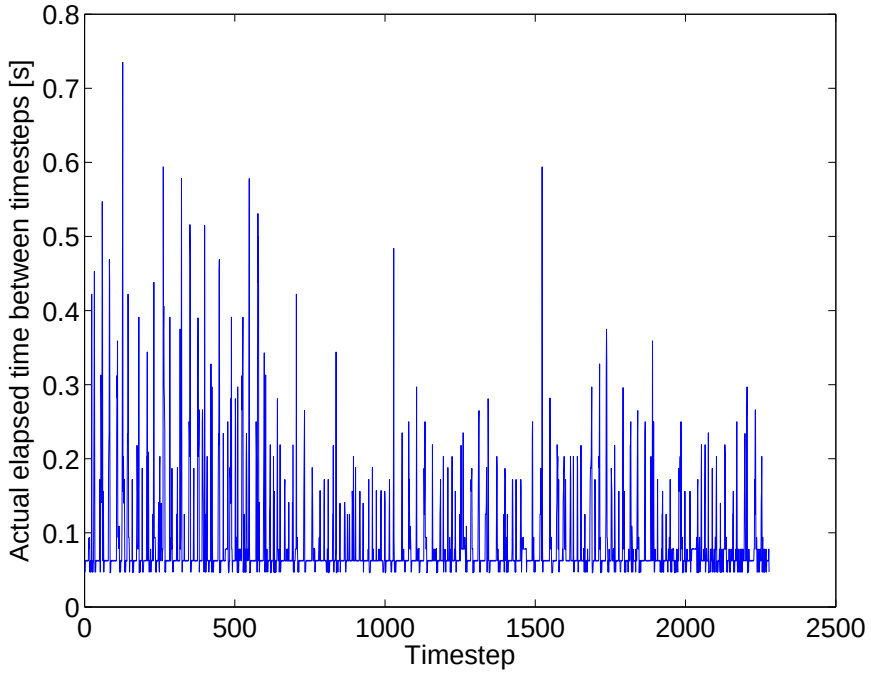


Figure 3.4: Sampling periods of the 15 Hz stereo images. We see that the nominal period is around 0.065 s, but the logging is not very consistent.

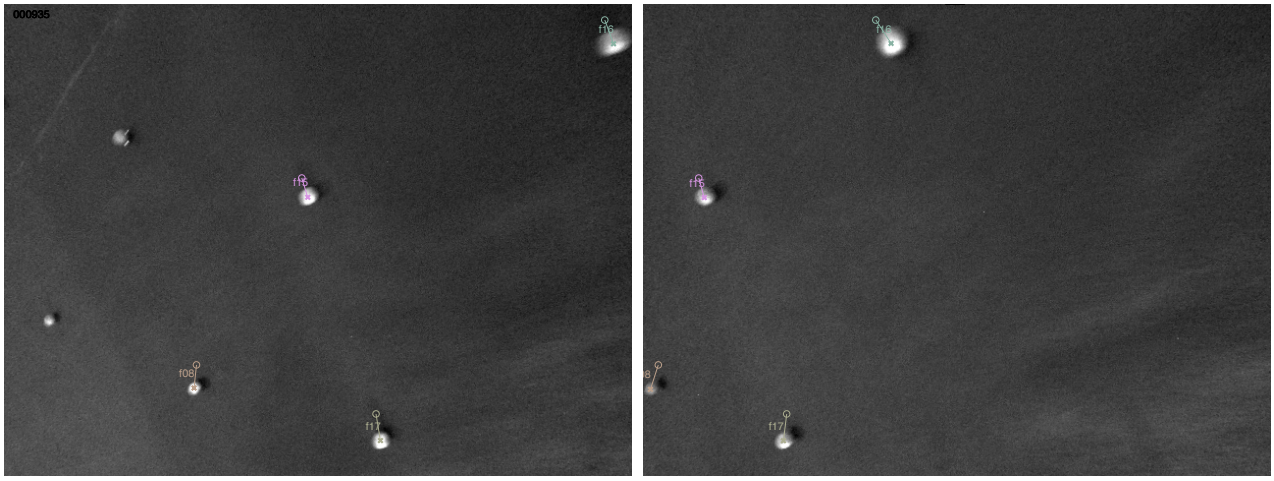
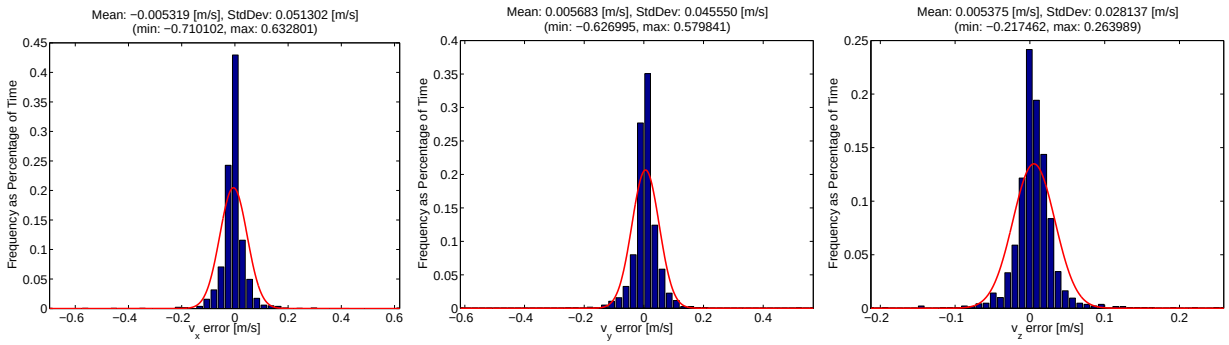
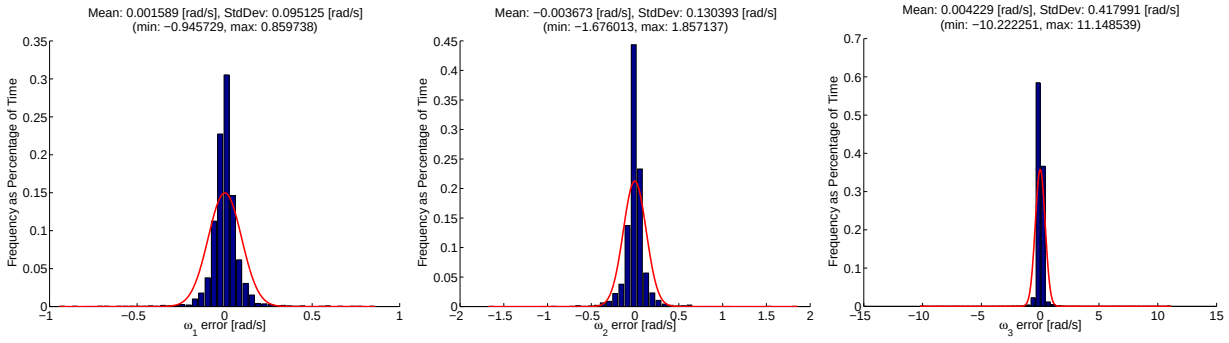


Figure 3.5: A sample pair of stereo images. The 'x' is the measurement, the 'o' indicates the predicted position using the groundtruth position and orientation.



(a) Translational speed errors.



(b) Rotational speed errors.

Figure 3.6: Histograms of IMU sensor errors. Red curves are Gaussians with standard deviation fit to the data. Both translational and rotational speeds have close to zero-mean errors.

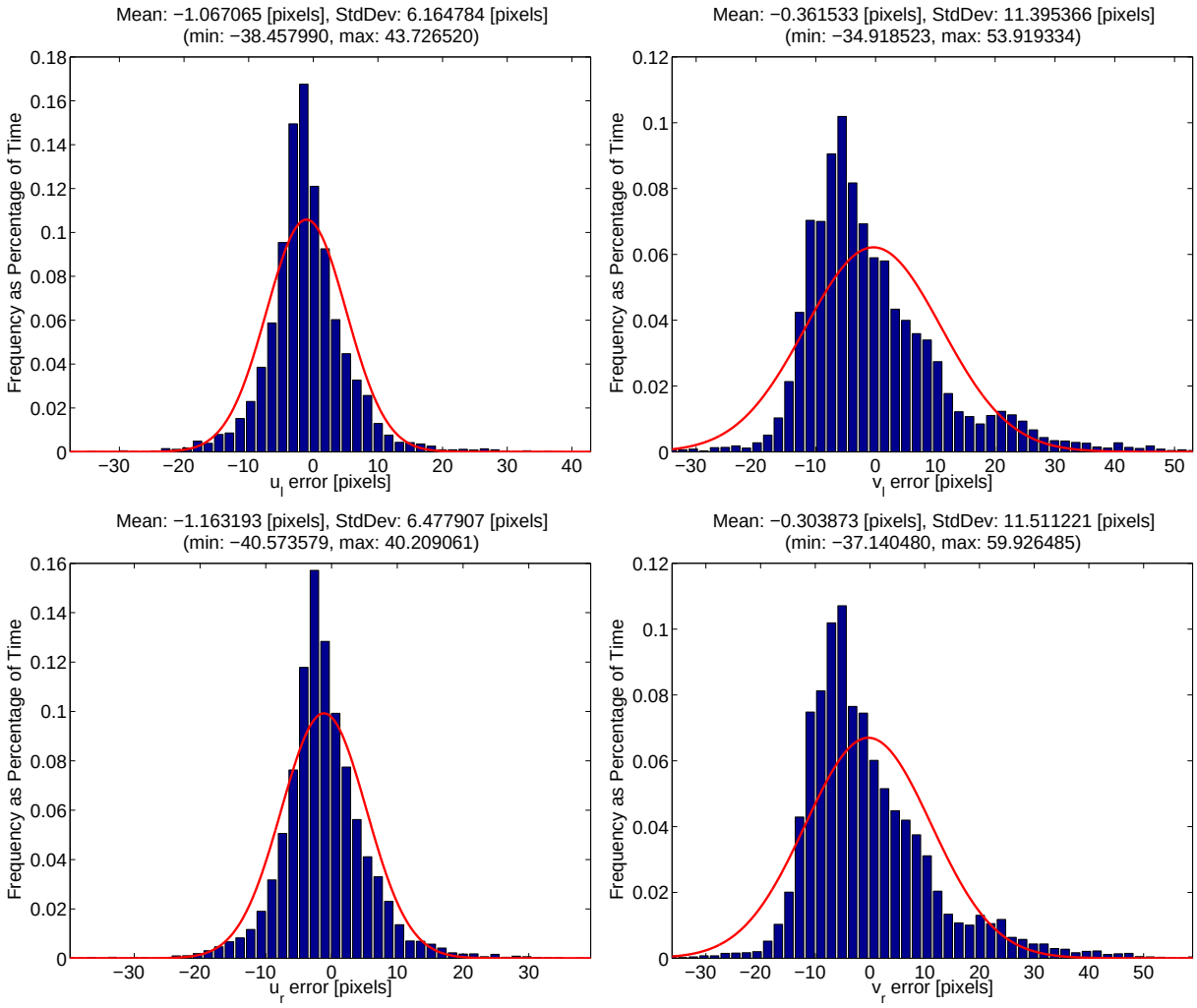


Figure 3.7: Histograms of stereo camera sensor errors. Red curves are Gaussians with standard deviation fit to the data.

The file containing the final processed data for this assignment is called `dataset3.mat`, which is a Matlab binary file. You can view the contents by typing

```
load dataset3.mat
who
```

at the Matlab (or Octave) command prompt. The following 17 Matlab variables will be listed:

`theta_vk_i` : a $3 \times K$ matrix where the k^{th} column is the axis-angle representation of the groundtruth value of $\mathbf{C}_{v_k i}$. Use this vector as $\boldsymbol{\psi}_k$ in (3.3c) to recover the rotation matrix.

`r_i_vk_i` : a $3 \times K$ matrix where the k^{th} column is the groundtruth value of $\mathbf{r}_i^{v_k i}$ [m]

`t` : a $1 \times K$ matrix of time values $t(k)$ [s]

`w_vk_vk_i` : a $3 \times K$ matrix where the k^{th} column is the measured rotational velocity, $\boldsymbol{\omega}_{v_k}^{v_k i}$ [rad/s]

`w_var` : a 3×1 matrix of the computed variances (based on groundtruth) of the rotational speeds [rad^2/s^2]

`v_vk_vk_i` : a $3 \times K$ matrix where the k^{th} column is the measured translational velocity, $\mathbf{v}_{v_k}^{v_k, i}$ [m/s]

`v_var` : a 3×1 matrix of the computed variances (based on groundtruth) of the translational speeds [m^2/s^2]

`rho_i_pj_i` : a 3×20 matrix where the j^{th} column is the position of feature j, $\boldsymbol{\rho}_i^{p_j i}$ [m]

`y_k_j` : a $4 \times K \times 20$ array of observations, $\mathbf{y}_k^j$ [pixels]. All components of `y_k_j(:,k,j)` will be -1 if the observation is invalid.

`y_var` : a 4×1 matrix of the computed variances (based on groundtruth) of the stereo measurements [pixels^2]

`C_c_v` : a 3×3 matrix giving the rotation from the vehicle frame to the camera frame, $\mathbf{C}_{cv}$

`rho_v_c_v` : a 3×1 matrix giving the translation from the vehicle frame to the camera frame, $\boldsymbol{\rho}_v^{cv}$ [m]

`fu` : the stereo camera's horizontal focal length, f_u [pixels]

`fv` : the stereo camera's vertical focal length, f_v [pixels]

`cu` : the stereo camera's horizontal optical center, c_u [pixels]

`cv` : the stereo camera's vertical optical center, c_v [pixels]

`b` : the stereo camera baseline, b [m]

3.2 Motion and Observation Models

3.2.1 Reference Frames

We need to define some reference frames for this problem. Figure 3.8 shows these reference frames. The frames are briefly described as follows:

- $\underrightarrow{\mathcal{F}}_i$: The inertial frame. Our estimate of the vehicle's translation/rotation will be computed in this frame. Additionally, the positions of the points on the floor, P_j, will be provided in this frame.
- $\underrightarrow{\mathcal{F}}_v$: The vehicle frame. This will be the main reference frame attached to the sensor head. We will be estimating the position/orientation of this frame with respect to the inertial frame. For convenience we choose to use a frame attached to the IMU for this frame.
- $\underrightarrow{\mathcal{F}}_c$: The camera frame. Since the stereo camera and the IMU can't be in the same location, we need an extra frame attached to the camera. For convenience this is attached to the left camera in the stereo pair. The translation/rotation between the vehicle frame and the camera frame, $\underrightarrow{\rho}^{vc}$ and $\mathbf{C}_{cv}$, are fixed and determined by calibration in advance.

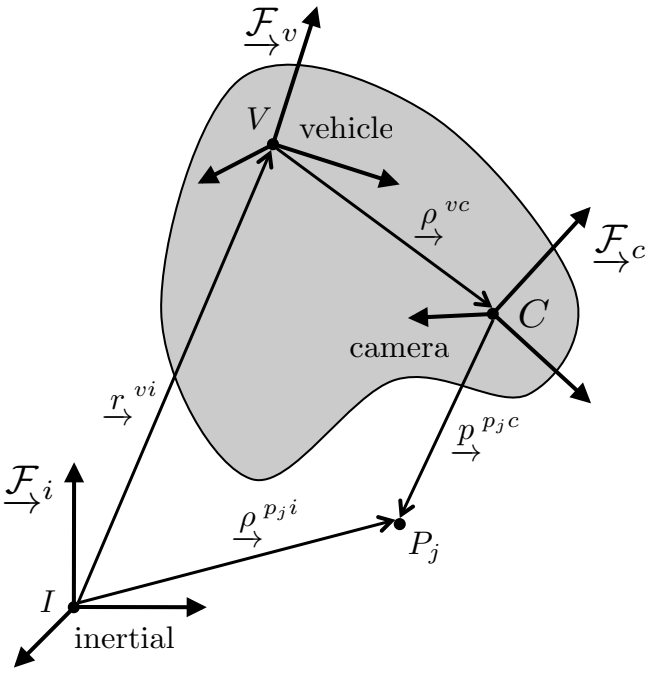


Figure 3.8: Reference frames used in Dataset 3.

3.2.2 Motion Model

This motion model we will use is very similar to the one derived by Barfoot (2017). It has been modified to account for a variable sampling period:

$$T_k := t(k) - t(k-1) \; . \tag{3.1}$$

We define the discrete-time motion model to be

$$\mathbf{r}_i^{v_k i} = \mathbf{r}_i^{v_{k-1} i} + \mathbf{C}_{v_{k-1} i}^T \mathbf{d}_{v_{k-1}}^{v_k v_{k-1}}, \tag{3.2a}$$

$$\mathbf{C}_{v_k i} = \boldsymbol{\Psi}_{v_k v_{k-1}} \mathbf{C}_{v_{k-1} i}, \tag{3.2b}$$

where k is the discrete-time index. The state of the system at timestep k is

$\mathbf{r}_i^{v_k i}$: Translation of vehicle frame with respect to inertial frame, expressed in inertial frame,
$\mathbf{C}_{v_k i}$: Rotation of vehicle frame with respect to inertial frame.

The quantities $\mathbf{d}_{v_{k-1}}^{v_k v_{k-1}}$ and $\boldsymbol{\Psi}_{v_k v_{k-1}}$ are given by

$$\mathbf{d}_{v_{k-1}}^{v_k v_{k-1}} := \mathbf{v}_{v_{k-1}}^{v_{k-1} i} T_k + \delta \mathbf{d}_k, \tag{3.3a}$$

$$\boldsymbol{\Psi}_{v_k v_{k-1}} := \cos\psi_k \mathbf{1} + (1 - \cos\psi_k) \left(\frac{\boldsymbol{\psi}_k}{\psi_k}\right) \left(\frac{\boldsymbol{\psi}_k}{\psi_k}\right)^T - \sin\psi_k \left(\frac{\boldsymbol{\psi}_k}{\psi_k}\right)^\times, \tag{3.3b}$$

$$\boldsymbol{\psi}_k := \boldsymbol{\omega}_{v_{k-1}}^{v_{k-1} i} T_k + \delta \boldsymbol{\psi}_k, \tag{3.3c}$$

$$\psi_k := |\boldsymbol{\psi}_k|. \tag{3.3d}$$

The interoceptive measurements (and associated noise variables) come from the IMU in this case:

$\mathbf{v}_{v_k}^{v_k i}$: Translational velocity of vehicle frame with respect to inertial frame, expressed in vehicle frame
$\boldsymbol{\omega}_{v_k}^{v_k i}$: Rotational velocity of vehicle frame with respect to inertial frame, expressed in vehicle frame
$\delta \mathbf{d}_k$: Translational process noise
$\delta \boldsymbol{\psi}_k$: Rotational process noise

3.2.3 Observation Model

The observation model, $\mathbf{g}(\cdot)$, is a nonlinear function that projects points expressed in the left camera frame into the camera's image coordinates. The position of a point on the floor, P_j, expressed in the camera frame, $\underrightarrow{\mathcal{F}}_c$, is denoted $\mathbf{p}_{c_k}^{p_j c_k}$ and is given by

$$\mathbf{p}_{c_k}^{p_j c_k} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{C}_{cv} \left(\mathbf{C}_{v_k i} \left(\boldsymbol{\rho}_i^{p_j i} - \mathbf{r}_i^{v_k i} \right) - \boldsymbol{\rho}_v^{cv} \right), \tag{3.4}$$

where $\left\{ \mathbf{r}_i^{v_k i}, \mathbf{C}_{v_k i} \right\}$ is the 'state' of the vehicle, $\{\boldsymbol{\rho}_v^{cv}, \mathbf{C}_{cv}\}$ is the known translation/rotation from the vehicle frame to the camera frame, and $\mathbf{p}_i^{p_j i}$ is the known position of P_j in the inertial frame.

The observation model, $\mathbf{g}(\cdot)$, projects $\mathbf{p}_{c_k}^{p_j c_k}$ into the rectified images of an axis-aligned stereo camera:

$$\mathbf{y}_k^j := \mathbf{g}\left(\mathbf{p}_{c_k}^{p_j c_k}\right) = \begin{bmatrix} u_l \\ v_l \\ u_r \\ v_r \end{bmatrix} = \frac{1}{z}\begin{bmatrix} f_u x \\ f_v y \\ f_u(x-b) \\ f_v y \end{bmatrix} + \begin{bmatrix} c_u \\ c_v \\ c_u \\ c_v \end{bmatrix} + \delta\mathbf{n}_k^j, \tag{3.5}$$

where the exteroceptive measurement and associated measurement noise are

- $\mathbf{y}_k^j$: The pixel coordinates of the point, P_j, projected into the left and right images of the stereo camera, (u_l, v_l) and (u_r, v_r), respectively
- $\delta\mathbf{n}_k^j$: Measurement noise

This calibrated, parallel stereo camera model, has the following fixed calibration parameters:

- c_u, c_v : The horizontal and vertical optical center from the top left of the image [pixels]
- f_u, f_v : The horizontal and vertical focal length [pixels]
- b : The camera baseline (i.e., distance between the two centers of projection) [m]

3.3 Assignment

We'll use the batch Gauss-Newton method to estimate the sensor head's position/orientation using both the IMU and stereo camera measurements. As usual, you should ask yourself do we really need both? Figure 3.9 shows what the sensor head's path looks like using only the interoceptive measurements (i.e., the IMU). A quick visual comparison with the groundtruth path in Figure 3.3 shows this is way off. And, although there are a lot of sensor camera measurements, there are many timesteps when there are less than three such observations. Figure 3.10 shows a histogram of the number of landmarks seen simultaneously. In fact, there are over 650 out of 1900 timesteps when there are less than three landmarks observed. This means that it is impossible to solve for the sensor head's position/orientation from the stereo camera measurements alone for many timesteps. Thus, in this situation, we actually need both the IMU and stereo camera measurements to create a reasonable estimator.

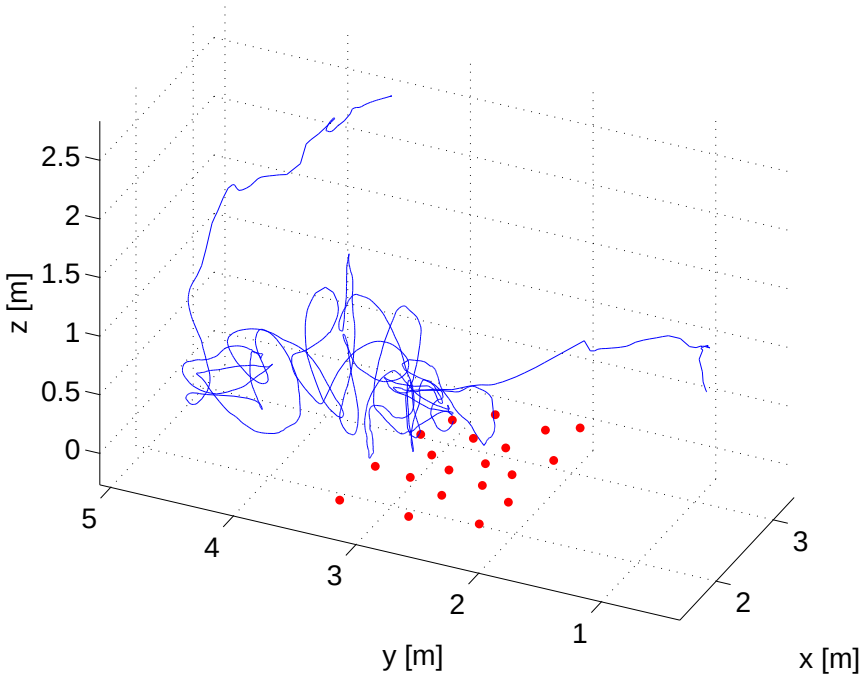


Figure 3.9: Sensor head path for Dataset 3 as computed using only IMU readings (no stereo camera measurements).

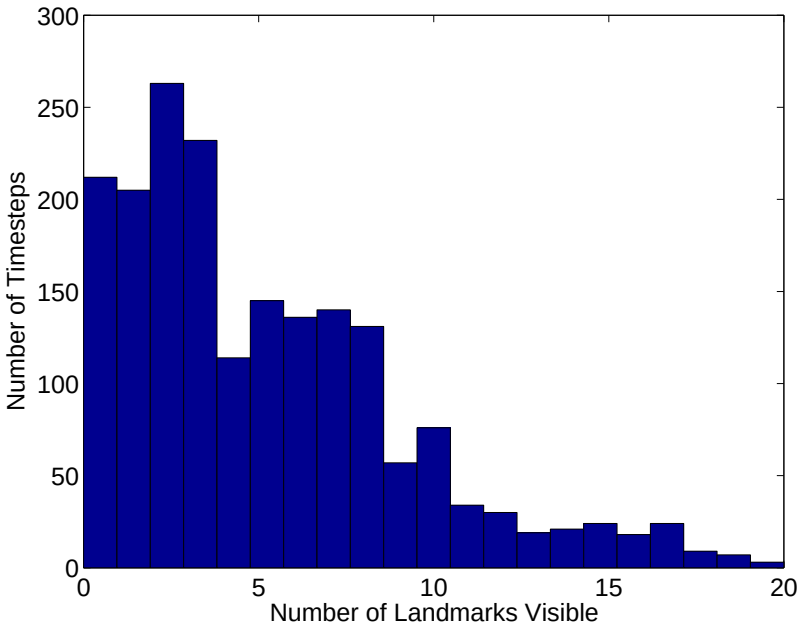


Figure 3.10: Histogram of number of landmarks seen simultaneously.

Answer the following questions in a short typeset report (marks in the margin - 25 total):

3 1. Based on the data above, is the assumption of zero-mean Gaussian noise reasonable? What values of the variances, $\mathbf{Q}_k$ and $\mathbf{R}_k^j$, should we use? Pay attention to the fact that we have a variable sampling period.

$$\begin{bmatrix} \delta\mathbf{d}_k \\ \delta\boldsymbol{\psi}_k \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), \qquad \delta\mathbf{n}_k^j \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k^j) \tag{3.6}$$

5 2. Write out an expression for the batch nonlinear least-squares objective function that we will seek to minimize using the Gauss-Newton method:

$$J\left(\bar{\mathbf{x}}_{k_1:k_2}\right) := \frac{1}{2}\mathbf{e}(\bar{\mathbf{x}}_{k_1:k_2})^T\mathbf{T}^{-1}\mathbf{e}(\bar{\mathbf{x}}_{k_1:k_2}), \tag{3.7}$$

where $\bar{\mathbf{x}}_{k_1:k_2}$ is the set of all poses from timestep k_1 to timestep k_2, i.e., $\bar{\mathbf{x}}_{k_1:k_2} = \left\{\bar{\mathbf{r}}_i^{v_{k_1}i}, \bar{\mathbf{C}}_{v_{k_1}i}, \ldots, \bar{\mathbf{r}}_i^{v_{k_2}i}, \bar{\mathbf{C}}_{v_{k_2}i}\right\}$. Note that we want to be able to handle only a window of poses in order to try different cases below.

4 3. Write out the Gauss-Newton algorithm to find

$$\bar{\mathbf{x}}_{k_1:k_2}^* = \arg\min_x J\left(\bar{\mathbf{x}}_{k_1:k_2}\right). \tag{3.8}$$

3 4. Make a plot of the number of visible landmarks at each given timestep, i.e., M_k vs. t_k. Use green dots for those timesteps for which there are at least three visible landmarks and red dots otherwise. We'll use this to interpret our results later.

10 5. Write a Matlab or Python script to implement your Gauss-Newton algorithm above, keeping k_1 and k_2 general. Note, the groundtruth sensor head position and orientation, $\left\{\mathbf{r}_i^{v_k i}, \mathbf{C}_{v_k i}\right\}$ are contained in the `r_i_vk_i` and `theta_vk_i` arrays.

(a) BATCH: First apply GN to the entire interval $k_1 = 1215$ to $k_2 = 1714$. To initialize the GN algorithm, you will need an initial guess for $\bar{\mathbf{x}}_{k_1:k_2}$. For this use the groundtruth at timestep k_1 and then dead reckon using only the interoceptive measurements and the motion model up to timestep k_2. This should look like a portion of Figure 3.9. Note you are estimating 3000 states since each timestep has 6 scalar states in all.

The estimation errors at timestep k can be computed according to

$$\text{translational errors:} \quad \delta\mathbf{r}_k = \begin{bmatrix} \delta r_{x,k} \\ \delta r_{y,k} \\ \delta r_{z,k} \end{bmatrix} := \bar{\mathbf{r}}_i^{v_k i^*} - \mathbf{r}_i^{v_k i},$$

$$\text{rotational errors:} \quad \delta\boldsymbol{\theta}_k^\times = \begin{bmatrix} \delta\theta_{x,k} \\ \delta\theta_{y,k} \\ \delta\theta_{z,k} \end{bmatrix}^\times := \mathbf{1} - \bar{\mathbf{C}}_{v_k i}^* \mathbf{C}_{v_k i}^T.$$

Plot the following:

- the error, $\delta r_{x,k}$ vs. t_k, as a solid line.

  - the uncertainty envelope, $\pm 3\sigma_{r_{x,k}}$ vs. t_k, as two dotted lines. Note, the variance, $\sigma^2_{r_{x,k}}$, can be extracted from the last iteration of the GN least-squares solution (see the notes).

  Repeat for $\delta r_{y,k}$, $\delta r_{z,k}$, $\delta\theta_{x,k}$, $\delta\theta_{y,k}$, and $\delta\theta_{z,k}$. That's 6 plots total.

(b) SLIDING WINDOW I: Start by apply GN to the interval $k_1 = k$ to $k_2 = k + \kappa$, where $k = 1215$ and $\kappa = 50$. Store only the pose from timestep k. Then repeatedly slide the window along by making $k = 1216, 1217, \ldots, 1714$, solving a GN problem at each k and storing only the pose from timestep k each time. To initialize GN, use a similar procedure to the BATCH case, but dead reckon from your most recent stored pose (and groundtruth only for the first pose, $k = 1215$). Make the same plots as in (a).

(c) SLIDING WINDOW II: Repeat (b) with $\kappa = 10$.

Comment on the differences in accuracy and computational effort between the BATCH and SLIDING WINDOW cases. Discuss any trends you see between your error plots and the plot you made in question 4. Attach your code in an Appendix.

Bibliography

Barfoot, T. D. (2017). "State Estimation for Robotics". Cambridge University Press, 2017.