COMPGX04: ROBOT VISION AND NAVIGATION

Workshop 1: Mobile GNSS Positioning using Least-Squares Estimation

Aim

The aim of this workshop is to apply least-squares estimation to GNSS positioning using Matlab.

Introduction

A smartphone tracking application logs pseudo-range and pseudo-range rate measurements every minute. Your task is to compute a position and velocity solution. Step by step instructions, including all equations to implement, are provided. Answers will appear on Moodle at the end of the workshop.

Data Files Supplied

You have been provided on Moodle with two data files, all in comma-separated variable (CSV) format:

- In the file Workshop1_Pseudo_ranges.csv, the first column contains the time in seconds and the first row contains the satellite numbers. The remaining rows and columns contain the corresponding pseudo-range measurements in metres.
- In the file Workshop1_Pseudo_range_rates.csv, the first column contains the time in seconds and the first row contains the satellite numbers. The remaining rows and columns contain the corresponding pseudo-range rate measurements in metres per second.

Task 1a: Single-epoch Positioning with Initialisation

Compute the position at time 0 using the first set of pseudo-range measurements in Workshop1_Pseudo_ranges.csv by following these steps

a) Mobile phone cell ID provides an approximate position of latitude -33.821075° , longitude 151.188496°, height 120m. Convert this to a Cartesian ECEF position using the Matlab function pv_NED_to_ECEF.m (supplied on Moodle). Note that the Matlab function inputs latitude and longitude in radians. The Matlab script Define_constants.m defines some useful constants.

b) Compute the Cartesian ECEF positions of the satellites at time 0 using the Matlab function Satellite_position_and_velocity.m (supplied on Moodle). Note that this function needs the satellite numbers given in the first row of the pseudo-ranges file.

c) Predict the ranges from the approximate user position to each satellite using

$$\hat{r}_{aj}^{-} = \sqrt{\left[\mathbf{C}_e^I\left(\hat{r}_{aj}^{-}\right)\hat{\mathbf{r}}_{ej}^{e} - \hat{\mathbf{r}}_{ea}^{e-}\right]^{\mathrm{T}}\left[\mathbf{C}_e^I\left(\hat{r}_{aj}^{-}\right)\hat{\mathbf{r}}_{ej}^{e} - \hat{\mathbf{r}}_{ea}^{e-}\right]} \tag{1}$$

where $\hat{\mathbf{r}}_{ej}^{e}$ is the Cartesian ECEF position of satellite j, $\hat{\mathbf{r}}_{ea}^{e-}$ is the predicted Cartesian ECEF user position and $\mathbf{C}_e^I$ is the Sagnac effect compensation matrix, given by

$$\mathbf{C}_e^I(r_{aj}) \approx \begin{pmatrix} 1 & \omega_{ie} r_{aj}/c & 0 \\ -\omega_{ie} r_{aj}/c & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{2}$$

where the Earth rotation rate, ω_{ie}, is 7.292115×10^{-5} rad s^{-1} and the speed of light, c, is 299792458 m s^{-1}. The recursion is resolved by initially computing the range with the Sagnac effect compensation matrix set to the identity matrix, then using this range to compute the Sagnac effect compensation matrix and then recomputing the range.

d) Compute the line-of-sight unit vector from the approximate user position to each satellite using

$$\mathbf{u}_{aj}^{e} \approx \frac{\hat{\mathbf{r}}_{ej}^{e} - \hat{\mathbf{r}}_{ea}^{e-}}{\hat{r}_{aj}^{-}}. \tag{3}$$

e) Formulate the predicted state vector, $\hat{\mathbf{x}}^{-}$, measurement innovation vector, $\delta\mathbf{z}^{-}$, and measurement matrix, $\mathbf{H}_{G}^{e}$, using

$$\hat{\mathbf{x}}^{-} = \begin{pmatrix} \hat{\mathbf{r}}_{ea}^{e-} \\ \delta\hat{\rho}_{c}^{a-} \end{pmatrix} \quad \delta\mathbf{z}^{-} = \begin{pmatrix} \tilde{\rho}_{a}^{2} - \hat{r}_{a2}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{17} - \hat{r}_{a17}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{18} - \hat{r}_{a18}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{22} - \hat{r}_{a22}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{23} - \hat{r}_{a23}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{26} - \hat{r}_{a26}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{27} - \hat{r}_{a27}^{-} - \delta\hat{\rho}_{c}^{a-} \\ \tilde{\rho}_{a}^{28} - \hat{r}_{a28}^{-} - \delta\hat{\rho}_{c}^{a-} \end{pmatrix} \quad \mathbf{H}_{G}^{e} = \begin{pmatrix} -u_{a2,x}^{e} & -u_{a2,y}^{e} & -u_{a2,z}^{e} & 1 \\ -u_{a17,x}^{e} & -u_{a17,y}^{e} & -u_{a17,z}^{e} & 1 \\ -u_{a18,x}^{e} & -u_{a18,y}^{e} & -u_{a18,z}^{e} & 1 \\ -u_{a22,x}^{e} & -u_{a22,y}^{e} & -u_{a22,z}^{e} & 1 \\ -u_{a23,x}^{e} & -u_{a23,y}^{e} & -u_{a23,z}^{e} & 1 \\ -u_{a26,x}^{e} & -u_{a26,y}^{e} & -u_{a26,z}^{e} & 1 \\ -u_{a27,x}^{e} & -u_{a27,y}^{e} & -u_{a27,z}^{e} & 1 \\ -u_{a28,x}^{e} & -u_{a28,y}^{e} & -u_{a28,z}^{e} & 1 \end{pmatrix} \tag{4}$$

where $\tilde{\rho}_{a}^{j}$ is the measured pseudo-range from satellite *j* to the user antenna and $\delta\hat{\rho}_{c}^{a-}$ is the predicted receiver clock offset (any value may be used as it has a linear relationship with the measurements).

f) Compute the position and receiver clock offset using unweighted least-squares:

$$\begin{pmatrix} \hat{\mathbf{r}}_{ea}^{e+} \\ \delta\hat{\rho}_{c}^{a+} \end{pmatrix} = \hat{\mathbf{x}}^{+} = \hat{\mathbf{x}}^{-} + \left(\mathbf{H}_{G}^{e\,\mathrm{T}}\mathbf{H}_{G}^{e}\right)^{-1}\mathbf{H}_{G}^{e\,\mathrm{T}}\delta\mathbf{z}^{-}. \tag{5}$$

g) Convert this Cartesian ECEF position solution to latitude, longitude and height using the Matlab function pv_ECEF_to_NED.m (supplied on Moodle). Note that the Matlab function outputs latitude and longitude in radians.

Task 1b: Single-epoch Positioning without Initialisation

Repeat Task 1a without using the initial position from cell ID. Instead, you should assume an initial position at the centre of the Earth (Cartesian ECEF coordinates: 0, 0, 0). Then iterate your least-squares position computation, using the solution from the previous iteration as the initial position for subsequent iterations. Keep iterating until your position solution is within 10cm of that from Task 1a.

Task 2: Multi-epoch Positioning

Now use the same method to compute the position at all of the epochs in Workshop1_Pseudo_ranges.csv. You can use the position and receiver clock offset solution from each epoch as the predicted position and receiver clock offset for the following epoch.

Task 3: Outlier Detection

Add residual-based outlier detection at each epoch.

a) Compute the residuals vector using

$$\mathbf{v} = \left[\mathbf{H}_G^e\left(\mathbf{H}_G^{e\,\mathrm{T}}\mathbf{H}_G^e\right)^{-1}\mathbf{H}_G^{e\,\mathrm{T}} - \mathbf{I}_m\right]\delta\mathbf{z}^-, \tag{6}$$

where $\mathbf{I}_m$ is the $m{\times}m$ identity matrix, where m is the number of measurements (8 in this case).

b) Compute the residuals covariance matrix using

$$\mathbf{C}_\mathbf{v} = \left[\mathbf{I}_m - \mathbf{H}_G^e\left(\mathbf{H}_G^{e\,\mathrm{T}}\mathbf{H}_G^e\right)^{-1}\mathbf{H}_G^{e\,\mathrm{T}}\right]\sigma_\rho^2, \tag{7}$$

where σ_ρ is the measurement error standard deviation; a suitable value is 5m. Note that this equation only applies to unweighted least-squares estimation.

c) Compute the normalised residuals and compare each with a threshold. Measurement *j* is an outlier when the following condition is met

$$\left|v_j\right| > \sqrt{C_{\mathbf{v}jj}}\,T, \tag{8}$$

Where C_{vjj} is the *j*th diagonal element of $\mathrm{C_v}$ and T is the outlier detection threshold; a suitable value is 6.

Note down any outliers that your software detects.

Task 4: Velocity determination

Use least-squares estimation to compute the velocity at all of the epochs using the pseudo-range rate measurements in Workshop1_Pseudo_range_rates.csv. You can assume that the initial predicted user velocity and clock drift are zero.

The predicted range rates are given by

$$\hat{\dot{r}}_{aj}^- = \hat{\mathbf{u}}_{aj}^{e-\mathrm{T}}\left[\mathbf{C}_e^I\left(\hat{r}_{aj}^-\right)\left(\hat{\mathbf{v}}_{ej}^e + \mathbf{\Omega}_{ie}^e\hat{\mathbf{r}}_{ej}^e\right) - \left(\hat{\mathbf{v}}_{ea}^{e-} + \mathbf{\Omega}_{ie}^e\hat{\mathbf{r}}_{ea}^{e-}\right)\right] \tag{9}$$

where $\hat{\mathbf{v}}_{ej}^e$ is the Cartesian ECEF velocity of satellite *j*, $\hat{\mathbf{v}}_{ea}^{e-}$ is the predicted Cartesian ECEF user velocity and the skew symmetric matrix of the Earth rotation rate is

$$\mathbf{\Omega}_{ie}^e = \begin{pmatrix} 0 & -\omega_{ie} & 0 \\ \omega_{ie} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{10}$$

The predicted state vector, measurement innovation vector and measurement matrix are given by

$$\hat{\mathbf{x}}^- = \begin{pmatrix} \hat{\mathbf{v}}_{ea}^{e-} \\ \delta\hat{\dot{\rho}}_c^{a-} \end{pmatrix} \quad \delta\mathbf{z}^- = \begin{pmatrix} \tilde{\dot{\rho}}_a^2 - \hat{\dot{r}}_{a2}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{17} - \hat{\dot{r}}_{a17}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{18} - \hat{\dot{r}}_{a18}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{22} - \hat{\dot{r}}_{a22}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{23} - \hat{\dot{r}}_{a23}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{26} - \hat{\dot{r}}_{a26}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{27} - \hat{\dot{r}}_{a27}^- - \delta\hat{\dot{\rho}}_c^{a-} \\ \tilde{\dot{\rho}}_a^{28} - \hat{\dot{r}}_{a28}^- - \delta\hat{\dot{\rho}}_c^{a-} \end{pmatrix} \quad \mathbf{H}_G^e = \begin{pmatrix} -u_{a2,x}^e & -u_{a2,y}^e & -u_{a2,z}^e & 1 \\ -u_{a17,x}^e & -u_{a17,y}^e & -u_{a17,z}^e & 1 \\ -u_{a18,x}^e & -u_{a18,y}^e & -u_{a18,z}^e & 1 \\ -u_{a22,x}^e & -u_{a22,y}^e & -u_{a22,z}^e & 1 \\ -u_{a23,x}^e & -u_{a23,y}^e & -u_{a23,z}^e & 1 \\ -u_{a26,x}^e & -u_{a26,y}^e & -u_{a26,z}^e & 1 \\ -u_{a27,x}^e & -u_{a27,y}^e & -u_{a27,z}^e & 1 \\ -u_{a28,x}^e & -u_{a28,y}^e & -u_{a28,z}^e & 1 \end{pmatrix} \tag{11}$$

where $\tilde{\dot{\rho}}_a^j$ is the measured pseudo-range from satellite *j* to the user antenna and $\delta\hat{\dot{\rho}}_c^{a-}$ is the predicted receiver clock offset (any value may be used as it has a linear relationship with the measurements).

The velocity and receiver clock drift solution, computed using unweighted least-squares, is

$$\begin{pmatrix} \hat{\dot{\mathbf{r}}}_{ea}^{e+} \\ \delta\hat{\dot{\rho}}_c^{a+} \end{pmatrix} = \hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \left({\mathbf{H}_G^e}^{\mathrm{T}} \mathbf{H}_G^e\right)^{-1} {\mathbf{H}_G^e}^{\mathrm{T}} \delta\mathbf{z}^- \,. \qquad (12)$$

The Matlab function pv_ECEF_to_NED.m (supplied on Moodle) can be used to obtain the Earth-referenced velocity resolved along the north, east and down directions.