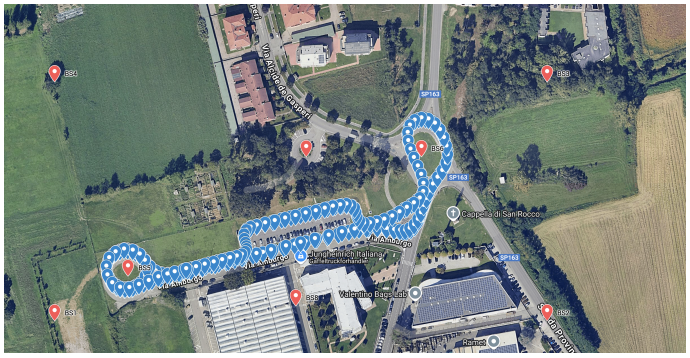
POLITECNICO
MILANO 1863

054183 - GEOINFORMATICS PROJECT

Kalman Estimation of Simulated 5G ToA Measurements for Positioning

11077620 Morten Sandbæk Edvardsen
11075411 Tobias Andresen



20.08.2025

Table of Contents

1 Introduction

This report concludes our coursework in *054183 Geoinformatics Project*, supervised by Professor Ludovico Biagi. We design a synthetic experiment in which a vehicle drives the **T1** loop, an industrial ring road, while eight virtual 5G-like base stations surround the track. Starting from the geodetic coordinates of these "towers," we build a Local Cartesian frame, generate biased, white-noise-corrupted pseudoranges, and recover the trajectory with three estimators. The work is wrapped in a Streamlit web application so users can upload CSV files, tweak noise parameters, run and compare the estimators, and download results.

The deployed web application is available at:

https://exkalman.streamlit.app

The full codebase (source repository) can be found at:

https://github.com/tiltobias/geoinformatics-project

1.1 Objectives

In short, our main objectives are to

- Establish a Local Cartesian frame based on eight virtual base stations placed around the T1 loop.
- Generate 5G-style Time-of-Arrival pseudoranges for every epoch, adding 1 m white noise.
- Model the user-equipment clock offset as a ± 0.5 ns random walk and inject its bias into the pseudoranges.
- Recover the trajectory epoch-by-epoch with a planar Least-Squares solver ($U = 0$).
- Smooth the LS positions with a constant-velocity Kalman filter.
- Estimate position, velocity and clock bias directly from the raw pseudoranges via an Extended Kalman Filter.
- Package into an interface so users can upload CSV files, run and compare estimations, tweak settings and download the results.
- Investigate how results change by different sigma values.

2 Development

2.1 Base stations

The reference trajectory **T1** is a loop along a road in an industrial area in Rosate, located about 25 km southwest of Milan. We started by plotting the trajectory **T1** in *Google My Maps*, and selected eight base stations around it (Figure 1). We then exported the geodetic coordinates of these base stations.

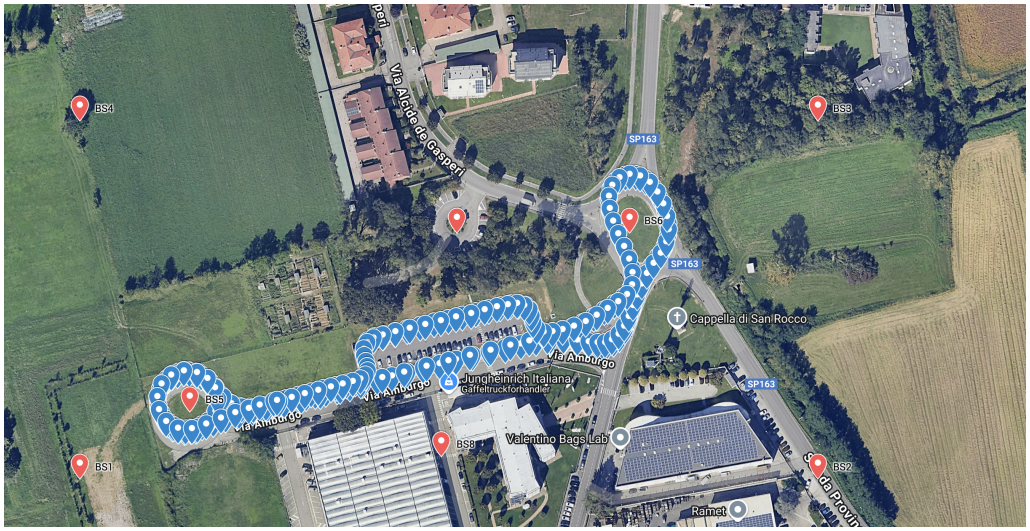


Figure 1: True trajectory in blue, selected base stations in red

2.2 Local Cartesian (ENU) frame

Several of the methods we are about to use rely on linear models and Euclidean geometry, where distances, directions, and state updates are computed using standard vector and matrix operations that assume a flat Cartesian space. We also aim to keep numerical values small to reduce the risk of floating-point errors in our computations. Therefore, we choose a local Cartesian coordinate system based around our area. We identify the axes as

$$\mathbf{x} = E \quad \text{(east)}, \qquad \mathbf{y} = N \quad \text{(north)}, \qquad \mathbf{z} = U \quad \text{(up)},$$

In our initial calculations, we set the origin to be at the first base station (BS1) (Figure 2), but in the published application, we used the mean position of all base stations to better accommodate different base station configurations. The numbering and relative positions of the base stations are the same.

Table 1: Base-station coordinates in the local Cartesian (LC) frame with origin at BS1.

Station	$E[m]$	$N[m]$	$U[m]$
BS1	0.0	0.0	20.0
BS2	368.368	0.0107	20.0
BS3	368.357	177.836	20.0
BS4	-3.332	177.826	20.0
BS5	54.863	33.343	20.0
BS6	274.311	122.261	20.0
BS7	188.099	122.258	20.0
BS8	180.265	11.117	20.0

Table 2: Base-station coordinates in the local Cartesian (LC) frame with origin at the centroid (published application).

Station	$E[m]$	$N[m]$	$U[m]$
BS1	-179.285	-80.5745	20
BS2	189.082	-80.574	20
BS3	189.077	97.251	20
BS4	-179.280	97.251	20
BS5	-124.421	-47.234	20
BS6	95.029	41.679	20
BS7	8.817	41.678	20
BS8	0.980	-69.463	20

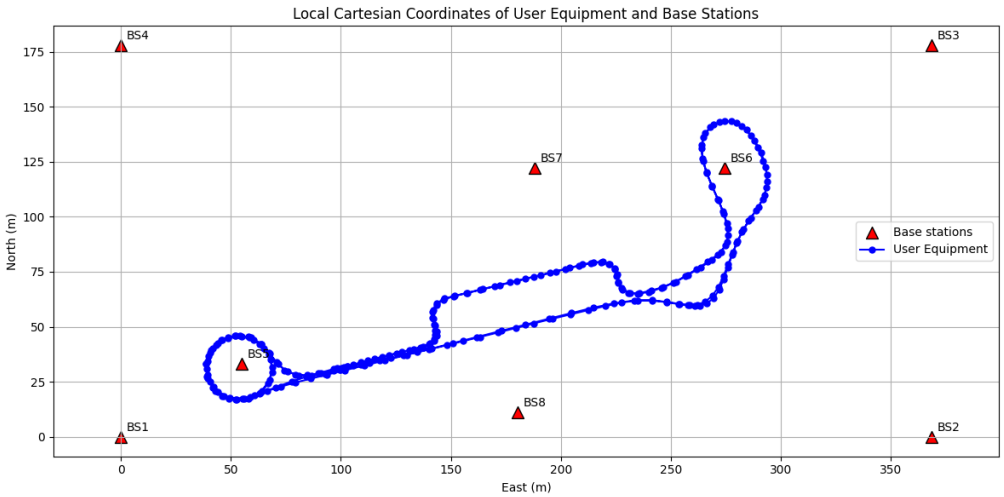


Figure 2: True trajectory in blue, selected base stations in red in local cartesian. (Old coordinate system with origin in BS1)

2.3 Heights

An important assumption in this project is that all base stations lie in the same horizontal plane. This was a crucial design decision, as it introduces a challenge in the position calculation process. Specifically, when all base stations are co-planar, it becomes impossible to solve for the full 3D position using only pseudorange measurements without additional constraints. To address this, we must fix the heights before the estimation. We assume that the user trajectory is quite level, without significant vertical variation. During estimation, we set the heights of the user trajectory to zero in the local reference frame. The zero height in our local system is defined as the mean height of the true user trajectory, so the height error will be minimal according to our assumption. The height of the base stations is chosen to be 20 meters above the ground, i.e. $U = 20\,\mathrm{m}$ for the plane of base stations.

2.4 Simulation of pseudoranges and Clock offset

To simulate pseudoranges we calculate the distance between each base station and each position of the true trajectory, and then add random noise and clock offset error. For each pair of base station and trajectory epoch we find the distance between them in 3D space. The random noise is then added using the Gaussian distribution with the expected value zero and standard deviation of 1 meter.

2.4.1 Clock offset random walk

To simulate the clock offset of the user equipment we used the Random Walk stochastic process, where each step in the walk corresponds with one epoch in the trajectory. We started the walk from the base value of 1 microsecond, and each step changes the clock offset by 0.5 nanoseconds, either adding or subtracting from the previous epoch's value (Figure 3). We add the product of the speed of light multiplied by the clock offset to the pseudoranges in each epoch.

Let epochs be indexed by $t = 0, 1, \ldots, n-1$. We model the receiver clock offset in *seconds* by a symmetric random walk starting at

$$dt_0 \;=\; 1\ \mu\mathrm{s} \;=\; 10^{-6}\ \mathrm{s},$$

with step size

$$\delta \;=\; 0.5\ \mathrm{ns} \;=\; 0.5 \times 10^{-9}\ \mathrm{s}.$$

At each epoch,

$$dt_t \;=\; dt_{t-1} \;+\; \delta\,\xi_t, \qquad \xi_t \in \{+1, -1\}\ , \quad \mathbb{P}(\xi_t{=}1) = \mathbb{P}(\xi_t{=}-1) = \tfrac{1}{2}, \tag{1}$$

equivalently

$$dt_t \;=\; dt_0 \;+\; \delta \sum_{k=1}^{t} \xi_k.$$

The pseudorange observation at epoch t then becomes

$$P_{UE}^{BS_i}(t) \;=\; \rho_{UE}^{BS_i}(t) \;+\; c\,dt_t \;+\; \varepsilon_{i,t}, \qquad \varepsilon_{i,t} \sim \mathcal{N}(0, 1^2\ \mathrm{m}^2). \tag{2}$$

2.5 Position estimation by Least Squares Method

As a baseline method for position estimation, we implemented the iterative Least Squares Method (LSM). For each epoch, we solve for the user's horizontal position and clock offset based on the pseudorange measurements to all base stations. Since all base stations are assumed to lie in the same horizontal plane, we constrain the height of the estimated trajectory to zero in the local coordinate system. The first epoch starts with an initial guess at the centroid of the base stations and a clock bias of zero. Each subsequent epoch uses the previous result as its starting point and iterates until the change in position is negligible (Figure 4 and 5).

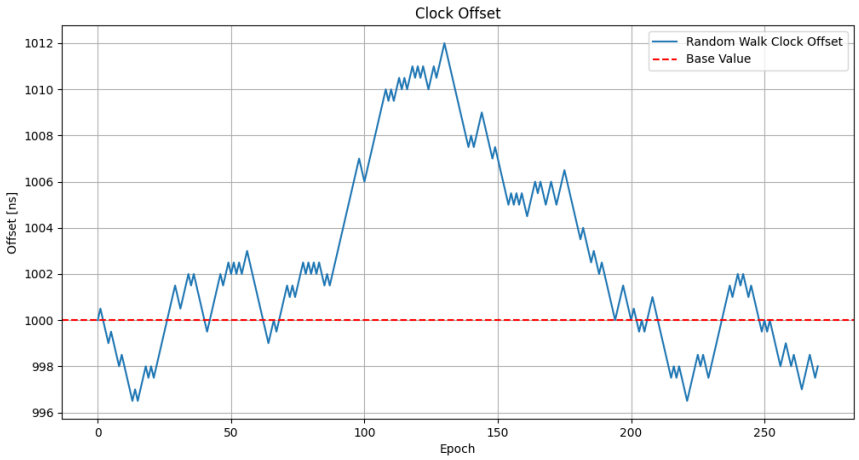


Figure 3: Random Walk

2.5.1 Observation Equations

The pseudorange observation for user equipment UE to base station BS_i is

$$P_{UE}^{BS_i} \;=\; \rho_{UE}^{BS_i} \;+\; b_{UE} \;+\; \varepsilon_i, \tag{3}$$

where $\rho_{UE}^{BS_i} = \|\mathbf{x}_{UE} - \mathbf{x}^{BS_i}\|$ is the geometric distance, $b_{UE} = c\,dt_{UE}$ is the clock offset in metres, and $\varepsilon_i \sim \mathcal{N}(0, 1^2\,\mathrm{m}^2)$ models measurement noise.

2.5.2 Linearisation and design matrix

Let $\hat{\mathbf{x}}_{UE}^{(0)} = [\hat{E}, \hat{N}, 0]^\mathsf{T}$ and $\hat{b}_{UE}^{(0)}$ be an a priori state for the current epoch. Linearising (3) about this state yields the model

$$\boldsymbol{\ell} = \mathbf{A}\,\Delta\mathbf{X}_{UE} + \boldsymbol{\varepsilon}, \qquad \Delta\mathbf{X}_{UE} = \begin{bmatrix} \Delta E \\ \Delta N \\ \Delta b_{UE} \end{bmatrix},$$

with rows of the design matrix

$$\mathbf{a}_i^\mathsf{T} = \begin{bmatrix} \dfrac{\hat{E} - E^{BS_i}}{\hat{\rho}_i} & \dfrac{\hat{N} - N^{BS_i}}{\hat{\rho}_i} & 1 \end{bmatrix}, \qquad \hat{\rho}_i = \left\|\hat{\mathbf{x}}_{UE}^{(0)} - \mathbf{x}^{BS_i}\right\|,$$

so that $\mathbf{A} \in \mathbb{R}^{m\times 3}$ is assembled by stacking the $\mathbf{a}_i^\mathsf{T}$.

2.5.3 Iterative least-squares solution

We initialise with $\mathbf{x}_{UE}^{(0)} = [E_c, N_c, 0]^\mathsf{T}$ (centroid of base stations) and $b_{UE}^{(0)} = 0$. At each iteration,

$$\Delta\mathbf{X}_{UE} = (\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A}^\mathsf{T}\boldsymbol{\ell}, \qquad \mathbf{x}_{UE} \leftarrow \mathbf{x}_{UE} + \begin{bmatrix} \Delta E \\ \Delta N \\ 0 \end{bmatrix}, \qquad b_{UE} \leftarrow b_{UE} + \Delta b_{UE},$$

until the position update satisfies $max(|\Delta E, \Delta N|) < 10^{-6}$m. The height is fixed to zero in the local frame, and the solution at epoch t is used to initialise epoch t+1.

Statistics can be found in Table 3.

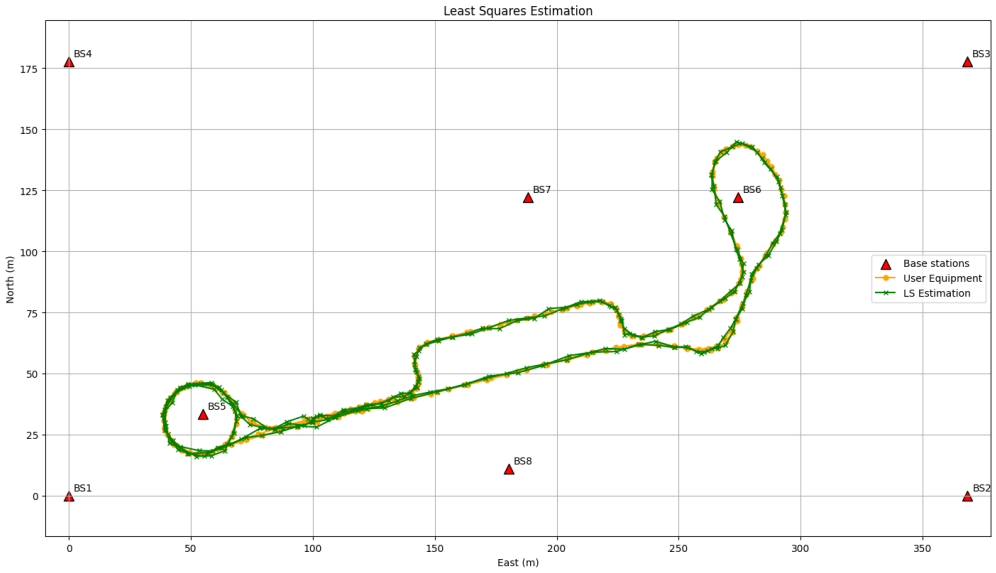


Figure 4: Least Squares estimated coordinates

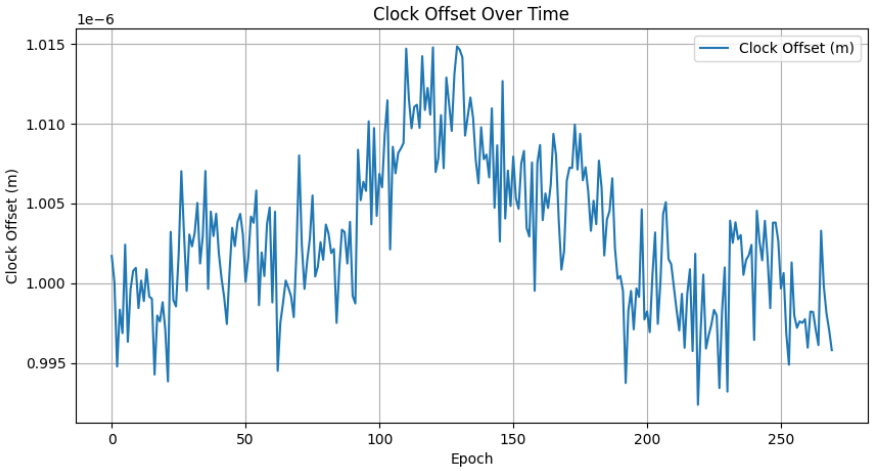


Figure 5: Least Squares estimated Clock Offset

2.6 Position smoothing with Kalman filter

As part of our goal to compare different position estimation techniques, we implemented a standard Kalman filter using a constant-velocity model. The state vector at epoch k is defined as

$$\mathbf{x}_k = \begin{bmatrix} E_u & N_u & V_E & V_N \end{bmatrix}^\mathsf{T},$$

where (E_u, N_u) are the local plane coordinates derived from an initial Least Squares Method (LSM) estimate, and (V_E, V_N) the corresponding velocities.

2.6.1 State transition model

We assume constant velocity with unit time step. The state propagation is given by

$$\mathbf{x}_{k|k-1} = \mathbf{T}\,\mathbf{x}_{k-1|k-1}, \qquad \mathbf{T} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The driving noise of the dynamic model is represented by

$$\mathbf{C}_{\text{model}} = \text{diag}\big(\sigma^2_{\text{pos}},\ \sigma^2_{\text{pos}},\ \sigma^2_{\text{vel}},\ \sigma^2_{\text{vel}}\big).$$

2.6.2 Observation model

At each epoch, the observation consists of the position estimate (E_u, N_u) from the LSM:

$$\mathbf{y}_k = \begin{bmatrix} E_u \\ N_u \end{bmatrix} = \mathbf{A}\,\mathbf{x}_k + \boldsymbol{\varepsilon}_k, \qquad \mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{\varepsilon}_k \sim \mathcal{N}(0, \mathbf{C}_{\text{obs}}),$$

with measurement noise covariance

$$\mathbf{C}_{\text{obs}} = \text{diag}(\sigma^2_{\text{obs}},\ \sigma^2_{\text{obs}}).$$

2.6.3 Recursive filter update

The Kalman filter recursion consists of the following steps:

Prediction:

$$\begin{aligned} \mathbf{x}_{k|k-1} &= \mathbf{T}\,\mathbf{x}_{k-1|k-1}, \\ \mathbf{K} &= \mathbf{T}\,\mathbf{C}_{k-1|k-1}\mathbf{T}^\mathsf{T} + \mathbf{C}_{\text{model}}. \end{aligned}$$

Update:

$$\begin{aligned} \mathbf{G}_k &= \mathbf{K}\mathbf{A}^\mathsf{T}\Big(\mathbf{A}\mathbf{K}\mathbf{A}^\mathsf{T} + \mathbf{C}_{\text{obs}}\Big)^{-1}, \\ \mathbf{x}_{k|k} &= (\mathbf{I} - \mathbf{G}_k\mathbf{A})\,\mathbf{x}_{k|k-1} + \mathbf{G}_k\,\mathbf{y}_k, \\ \mathbf{C}_{k|k} &= (\mathbf{I} - \mathbf{G}_k\mathbf{A})\,\mathbf{K}. \end{aligned}$$

This formulation provides a smoothed trajectory by balancing noisy LSM position measurements with the kinematic model (Figure 6). The relative influence of model versus measurements is controlled through the noise parameters $\sigma_{\text{obs}}, \sigma_{\text{pos}}$, and σ_{vel}.

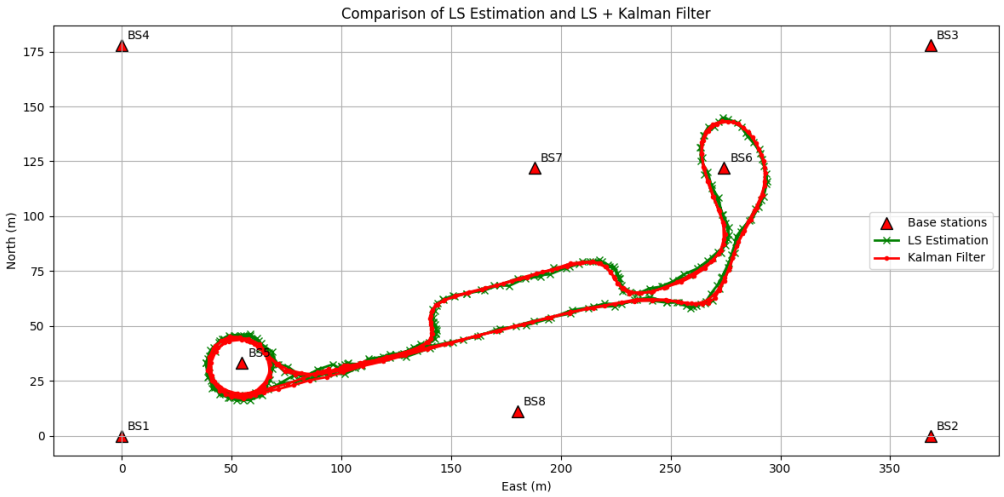


Figure 6: Kalman filter smoothed trajectory

2.7 Position estimation with Extended Kalman filter

To complement the standard Kalman filter, we also implemented an Extended Kalman filter (EKF) for direct estimation from pseudorange measurements. Unlike the standard Kalman filter, which operates on linearly estimated positions (such as those from LSM), the EKF incorporates the full nonlinear measurement model and estimates position, velocity, and clock offset in a single framework. The state vector at epoch k is defined as

$$\mathbf{x}_k = \begin{bmatrix} N_u & E_u & V_N & V_E & c\,dt_u \end{bmatrix}^{\mathsf{T}},$$

where (N_u, E_u) are user coordinates in the local plane, (V_N, V_E) the corresponding velocities, and $c\,dt_u$ the receiver clock bias expressed in meters.

2.7.1 State transition model

We adopt a constant-velocity kinematic model with sampling interval $dt = 1$. The state transition is expressed as

$$\mathbf{x}_{k|k-1} = \mathbf{T}\,\mathbf{x}_{k-1|k-1}, \qquad \mathbf{T} = \begin{bmatrix} 1 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The process noise covariance is modelled as

$$\mathbf{C}_{\text{model}} = \text{diag}\big(\sigma^2_{\text{pos}},\, \sigma^2_{\text{pos}},\, \sigma^2_{\text{vel}},\, \sigma^2_{\text{vel}},\, (c\,\sigma_{dt})^2\big).$$

2.7.2 Nonlinear measurement model

For each base station BS_i with coordinates (E^i, N^i, U^i), the pseudorange observation is

$$P_{UE}^{BS_i}(k) \;=\; \rho_{UE}^{BS_i}(k) \;+\; c\,dt_u(k) \;+\; \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2_{\text{pr}}), \tag{4}$$

where $\rho_{UE}^{BS_i}(k) = \sqrt{(N^i - N_u)^2 + (E^i - E_u)^2 + (U^i - 0)^2}$ is the geometric range, and σ_{pr} is the pseudorange measurement noise covariance. The corresponding Jacobian matrix $\mathbf{A}_k$ (linearisation of the measurement function) has rows

$$\mathbf{a}_i^{\mathsf{T}} = \begin{bmatrix} \dfrac{N_u - N^i}{\rho_i} & \dfrac{E_u - E^i}{\rho_i} & 0 & 0 & 1 \end{bmatrix},$$

for $i = 1, \ldots, m$ base stations.

2.7.3 Recursive filter update

At each epoch, the EKF proceeds as follows:

Prediction:

$$\begin{aligned}\mathbf{x}_{k|k-1} &= \mathbf{T}\,\mathbf{x}_{k-1|k-1},\\ \mathbf{K}_k &= \mathbf{T}\,\mathbf{C}_{k-1|k-1}\,\mathbf{T}^{\mathsf{T}} + \mathbf{C}_{\text{model}}.\end{aligned}$$

Update:

$$\begin{aligned}\mathbf{G}_k &= \mathbf{K}_k\,\mathbf{A}_k^{\mathsf{T}}\Big(\mathbf{A}_k\,\mathbf{K}_k\,\mathbf{A}_k^{\mathsf{T}} + \mathbf{C}_{\text{obs}}\Big)^{-1},\\ \mathbf{x}_{k|k} &= \mathbf{x}_{k|k-1} + \mathbf{G}_k\big(\mathbf{y}_k - \hat{\boldsymbol{\rho}}_k\big),\\ \mathbf{C}_{k|k} &= \big(\mathbf{I} - \mathbf{G}_k\mathbf{A}_k\big)\,\mathbf{K}_k,\end{aligned}$$

where $\mathbf{y}_k$ are the measured pseudoranges and $\hat{\boldsymbol{\rho}}_k$ the predicted pseudoranges.

2.7.4 Initialization

The initial state $\mathbf{x}_{0|0}$ is obtained from the LSM solution at the first epoch, with zero initial velocities and clock bias. The covariance $\mathbf{C}_{0|0}$ is set according to the assumed initial error variance.

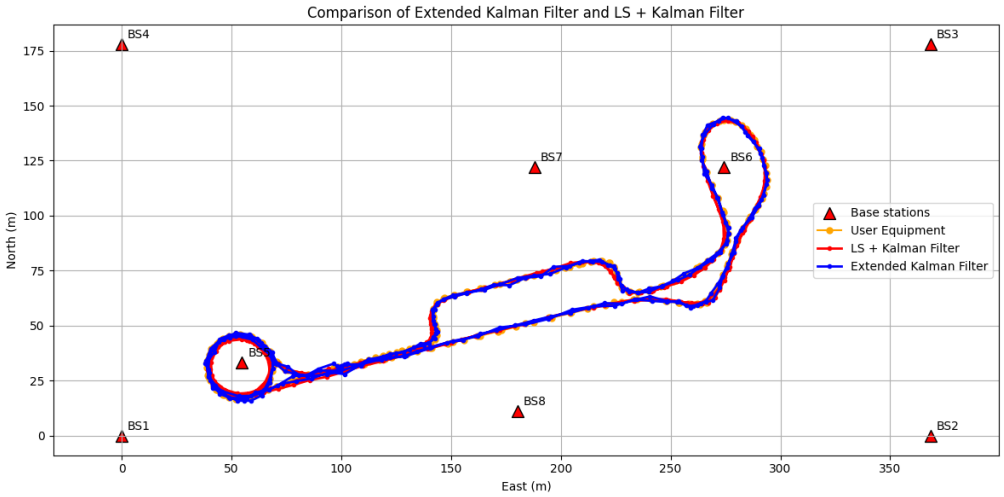


Figure 7: Extended Kalman filter estimated trajectory

The EKF framework thus provides a recursive solution that incorporates both dynamics and nonlinear measurement geometry (Figure 7). While more sensitive to initialization and parameter choice than LSM, it has the potential to yield smoother and more consistent trajectory estimates.

3 User Interface

The user interface is a two-page web application built using the open source Python framework Streamlit[1]. On the *Upload* page (Figure 8) the user can download template CSV files, read a short guide, and drag-and-drop their base-station and pseudorange files to be used in the position estimations. Optionally, a true-trajectory file can also be uploaded to compare with the plotted results. If there are no errors, the user can continue to the *Estimation* page.

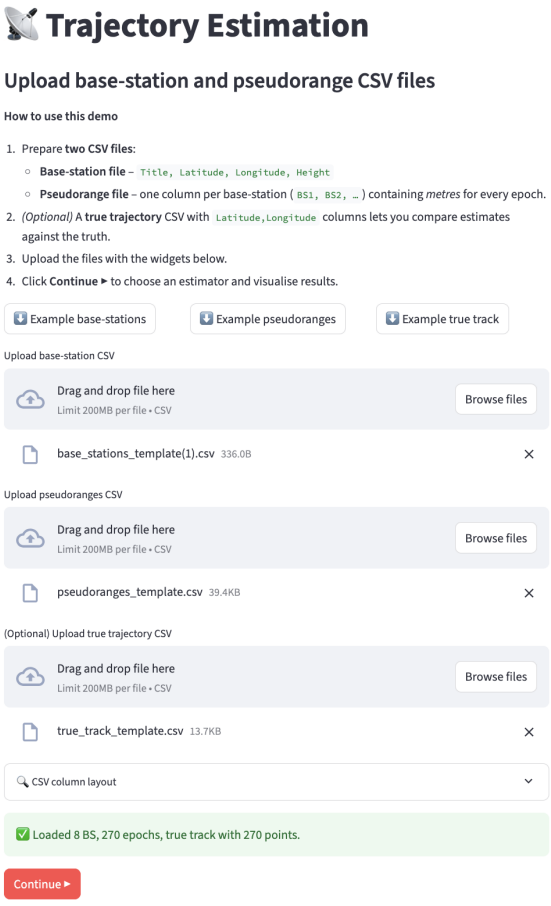


Figure 8: App: Upload page

The *Estimation* page (Figure 9) embeds a Leaflet map through streamlit-folium[2]. Base-station markers and, if present, the true trajectory appear by default. Solution tracks pop up automatically after a run, each in its own colour. Visibility is controlled in two places: a small layer panel in the sidebar for stations and truth, and three check-boxes above the map for the individual estimator outputs.

The sidebar itself is divided into three blocks. The top block hosts the run buttons for Least-Squares, LS + Kalman and EKF, plus input parameters for LS + Kalman and EKF. The middle block toggles base station and true trajectory visibility. The bottom block offers CSV exports for every finished solution in both local ENU and geodetic coordinates, as well as computed residuals for the Least Squares and Extended Kalman filter solutions.

[1] https://streamlit.io/
[2] https://folium.streamlit.app/

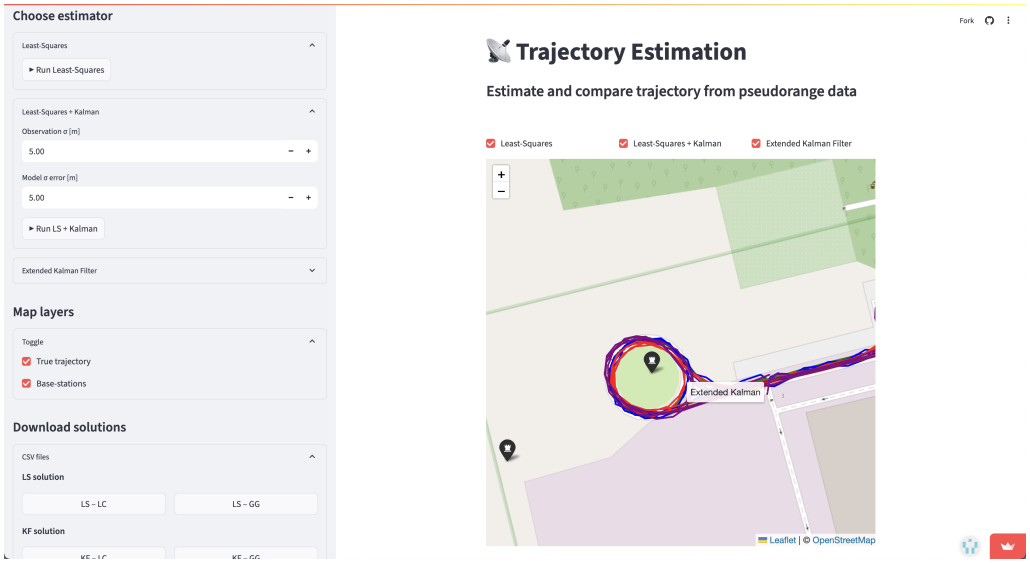


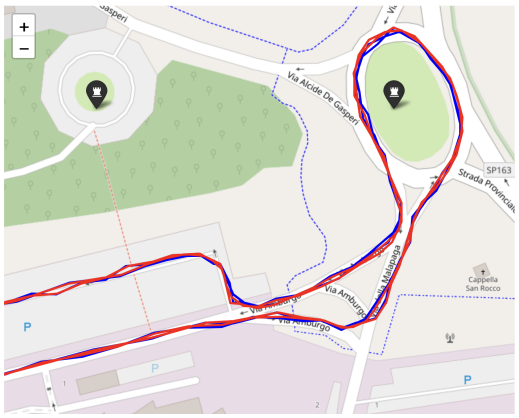
Figure 9: App: Estimation page

4 Parameter comparison: Kalman filter

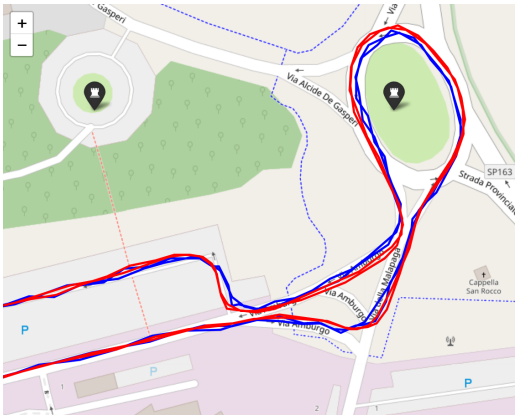
On page 13 and 14 we compare Kalman filtered results (in red) against pure Least Squares results (in blue). We chose to test different values of the uncertainty of the observations σ_{obs} and the position in the dynamic model σ_{pos}. For σ_{obs} we tested the values 1, 2.5, 5, 10 meters, and for σ_{pos} the values 0.1, 1, 5, 10 meters. To simplify the testing process and visualization, we don't want to change the value of the velocity in the dynamic model σ_{vel}, so we kept the value at 1. We chose to fix the σ_{vel} value because we believe we can find a satisfactory set of parameters without changing it. The images are screenshots from our online application, where we input the different σ values. The displayed images are zoomed onto a part of the track to get a more detailed view, while containing enough different parts of the trajectory to compare.

We observed that for very small values of σ_{obs} the filter trusts the observations way too much, and we get almost no smoothing effect. However, when we adjust σ_{obs} to be higher without also increasing the uncertainty of σ_{pos}, we get a very smooth trajectory that doesn't follow the original path at all.

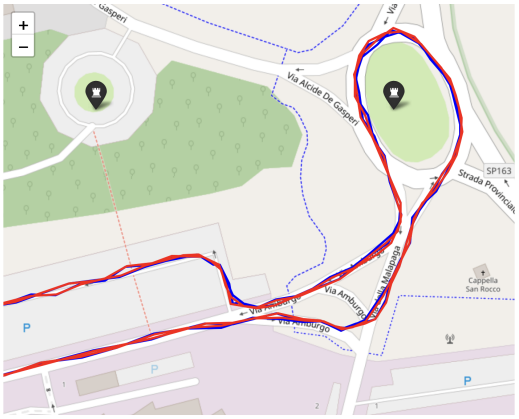
Hence, we try to opt for a balance between smoothness while still following the trajectory quite well. This way, the results do not depend too much on either the model or the jittery least squares path. Our favorite is Figure 10d on page 13.



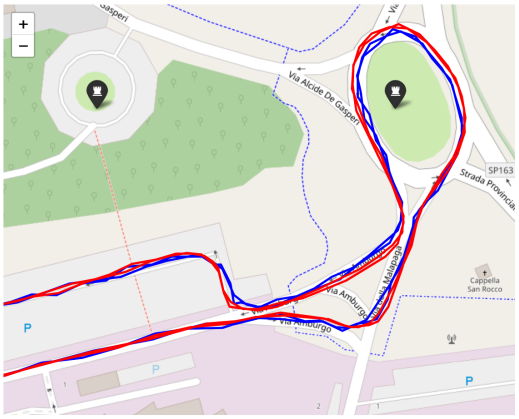
(a) $\sigma_{obs} = 1m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



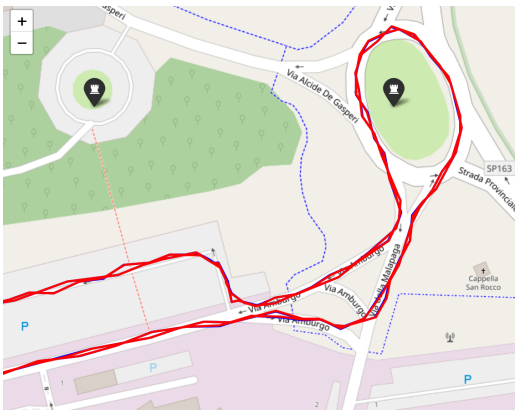
(b) $\sigma_{obs} = 2.5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



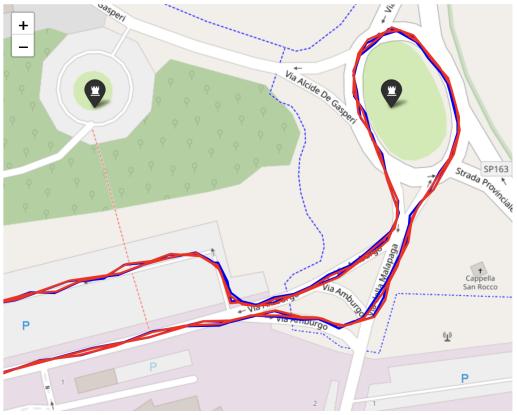
(c) $\sigma_{obs} = 1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



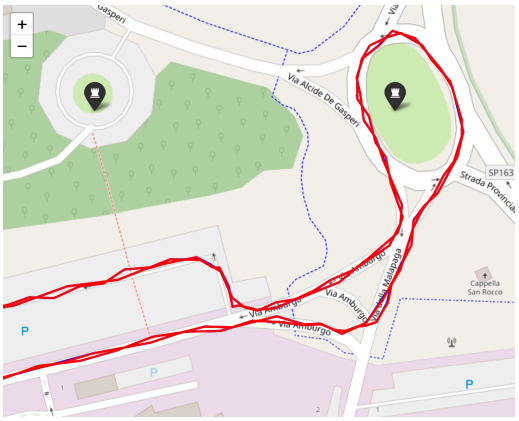
(d) $\sigma_{obs} = 2.5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



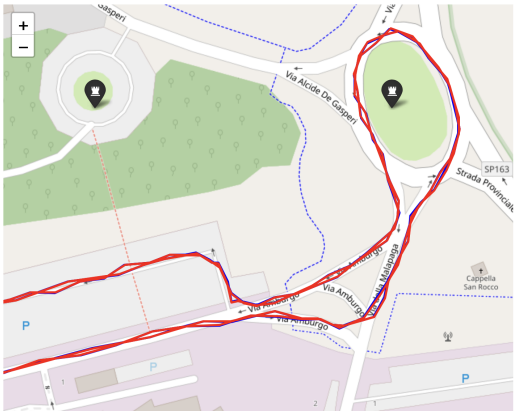
(e) $\sigma_{obs} = 1m,\ \sigma_{pos} = 5m,\ \sigma_{vel} = 1\frac{m}{s}$



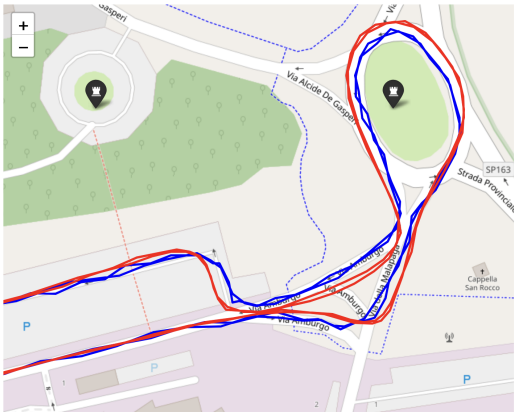
(f) $\sigma_{obs} = 2.5m,\ \sigma_{pos} = 5m,\ \sigma_{vel} = 1\frac{m}{s}$



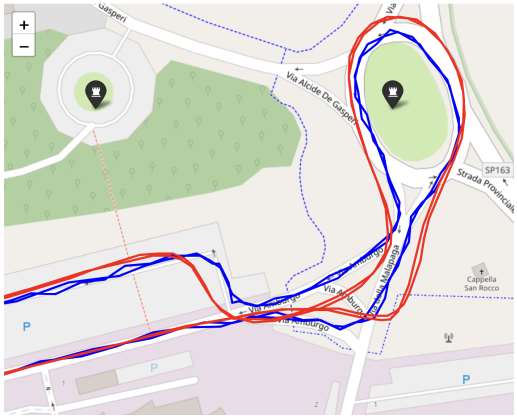
(g) $\sigma_{obs} = 1m,\ \sigma_{pos} = 10m,\ \sigma_{vel} = 1\frac{m}{s}$



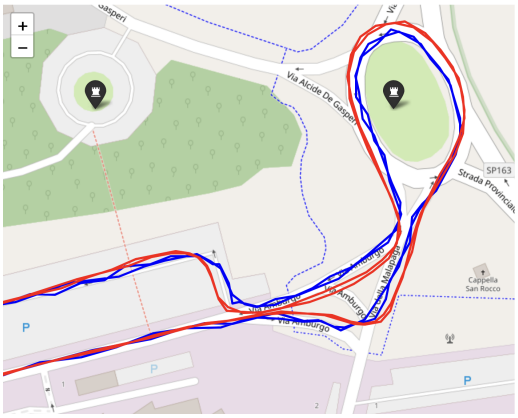
(h) $\sigma_{obs} = 2.5m,\ \sigma_{pos} = 10m,\ \sigma_{vel} = 1\frac{m}{s}$



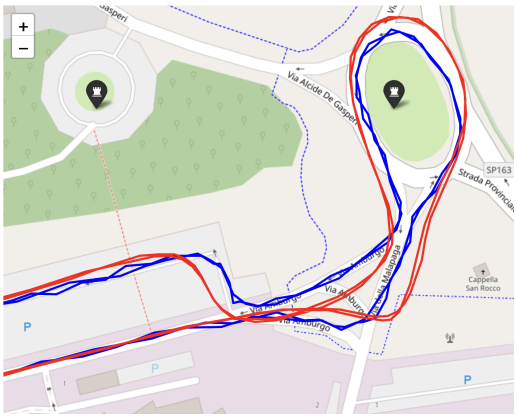
(a) $\sigma_{obs} = 5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



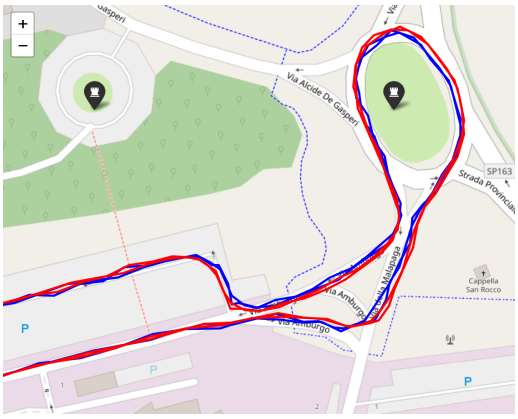
(b) $\sigma_{obs} = 10m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



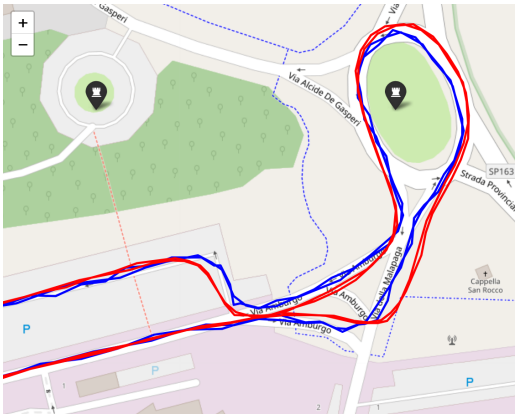
(c) $\sigma_{obs} = 5m\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



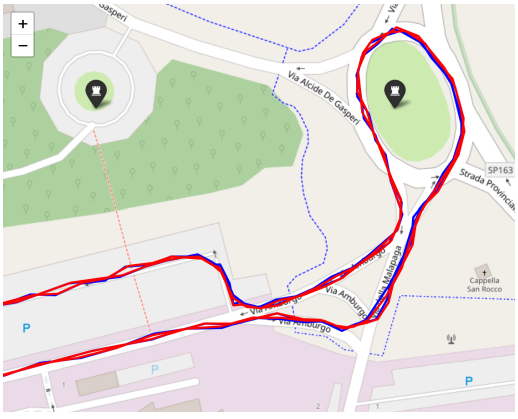
(d) $\sigma_{obs} = 10m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



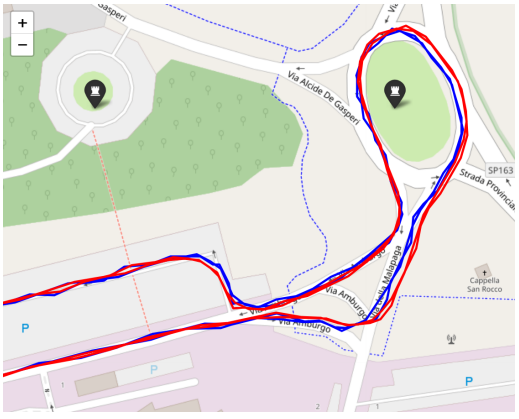
(e) $\sigma_{obs} = 5m,\ \sigma_{pos} = 5m,\ \sigma_{vel} = 1\frac{m}{s}$



(f) $\sigma_{obs} = 10m,\ \sigma_{pos} = 5m,\ \sigma_{vel} = 1\frac{m}{s}$



(g) $\sigma_{obs} = 5m,\ \sigma_{pos} = 10m,\ \sigma_{vel} = 1\frac{m}{s}$



(h) $\sigma_{obs} = 10m,\ \sigma_{pos} = 10m,\ \sigma_{vel} = 1\frac{m}{s}$

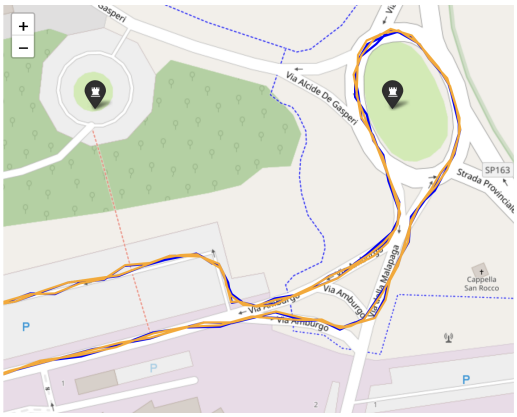
5 Parameter comparison: Extended Kalman filter

On page 16 and 17 we compare the Least Squares estimation (Figure 4) against different settings of the Extended Kalman parameters. All values are in the unit of meters.

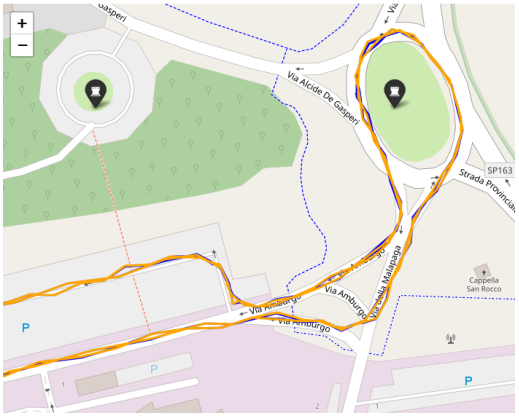
From testing with different values, our experience is that the measurement-noise standard deviation $\sigma_{\text{pseudorange}}$ provides a significant smoothness effect at higher values, while the estimated position follows the trajectory more tightly at smaller values. In particular, increasing $\sigma_{\text{pseudorange}}$ reduces the Kalman gain and makes the filter rely more on the constant-velocity model, yielding a smoother, but more lagged, response to turns. A small $\sigma_{\text{pseudorange}}$ produces a very responsive estimate that may exhibit jitter when measurements are noisy. We see that when smoothness increases, the extended Kalman estimate overshoots turns more than the Kalman filter applied to Least-Squares positions. This phenomenon is clearly visible in Figure 12c on page 16, where the Extended Kalman trajectory swings too wide around the roundabout and arrives late at the subsequent 90° turn.

Equally important are the process-noise parameters σ_{pos} and σ_{vel}, which govern how much the filter allows deviations from the constant-velocity model in position and velocity, respectively. A larger σ_{pos} permits the position estimate to jump more between time steps, helpful for capturing rapid maneuvers but at the cost of noisier trajectories, while a smaller σ_{pos} enforces a smoother path. Similarly, increasing σ_{vel} allows the estimated speed to change quickly (useful for acceleration), whereas reducing it constrains the filter to near-constant speed. We observe that high values of $\sigma_{\text{pseudorange}}$ can destabilize the solution if σ_{pos} and σ_{vel} remain low, but when σ_{pos} and σ_{vel} are raised (as shown in the second column on page 17), the trajectory remains well-behaved despite aggressive measurement smoothing.

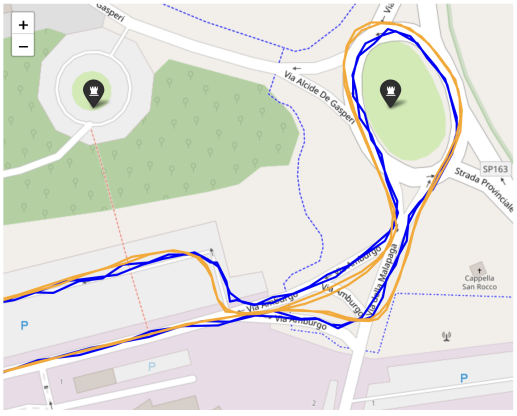
Our experience is that finding a middle ground between a jagged path and oversmoothing is much more difficult with this method than with the Kalman filter applied to Least-Squares output. It is therefore hard to choose only one preferred set of parameters. The two best estimations, in our opinion, are shown in Figure 12d and Figure 13h, which balance responsiveness and smoothness: the first is among the closest estimations yet smoother than some of the tighter options, while the second employs a very high $\sigma_{\text{pseudorange}}$ together with high σ_{pos} and σ_{vel}, yielding one of the smoothest trajectories that still follows the path closely.



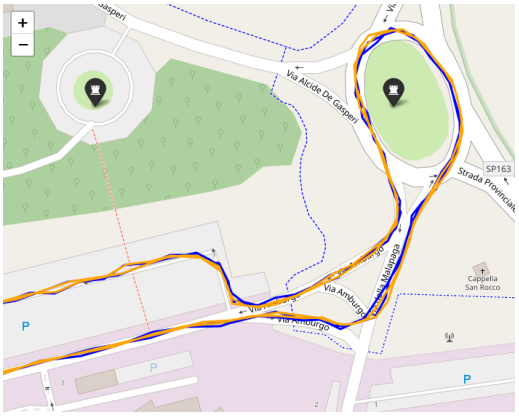
(a) $\sigma_{pr} = 0.1m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



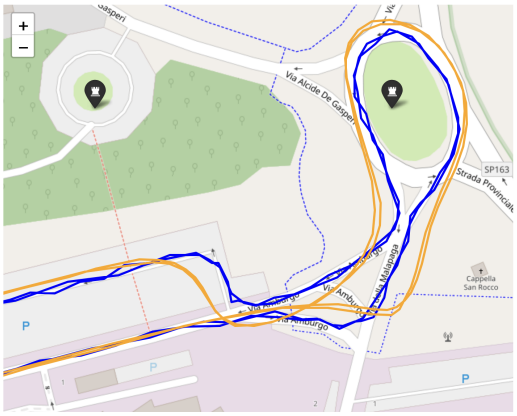
(b) $\sigma_{pr} = 0.1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



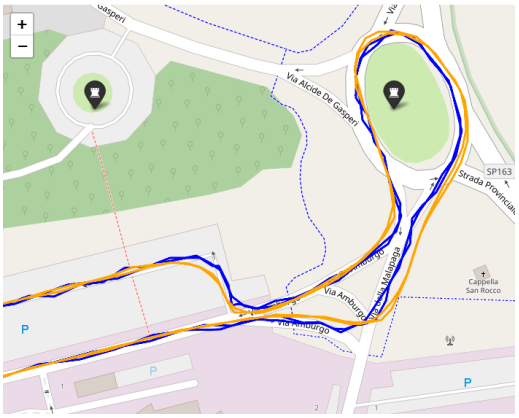
(c) $\sigma_{pr} = 1m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



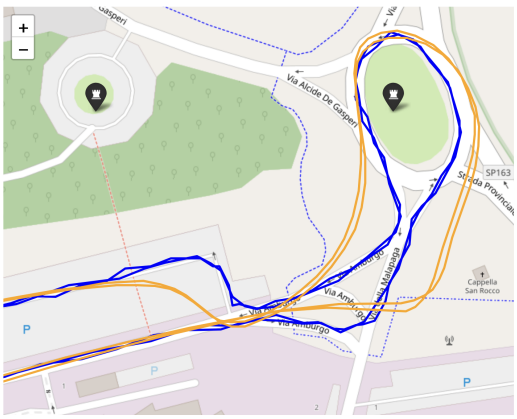
(d) $\sigma_{pr} = 1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



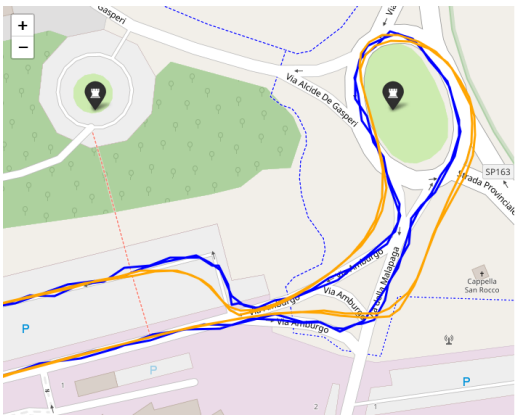
(e) $\sigma_{pr} = 2.5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



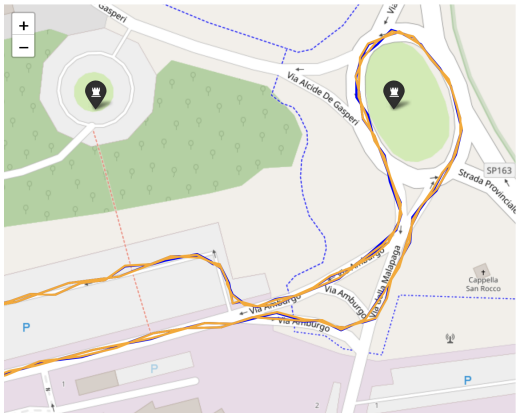
(f) $\sigma_{pr} = 2.5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



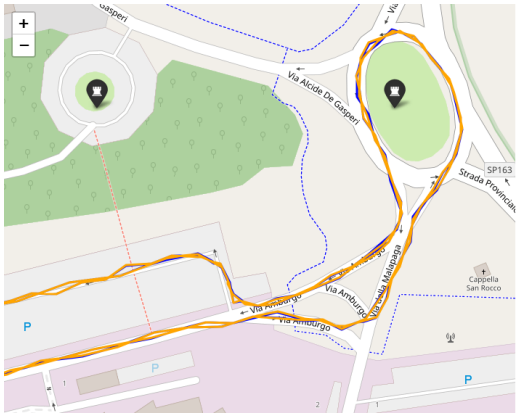
(g) $\sigma_{pr} = 5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



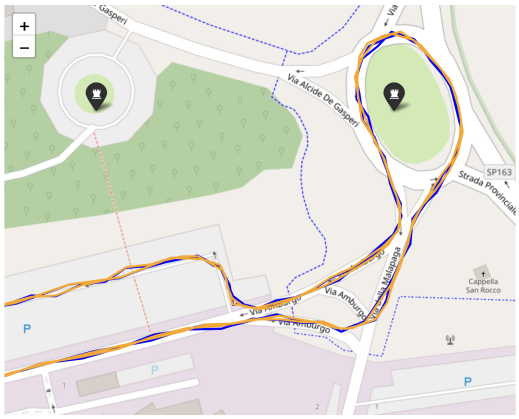
(h) $\sigma_{pr} = 5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 0.1\frac{m}{s}$



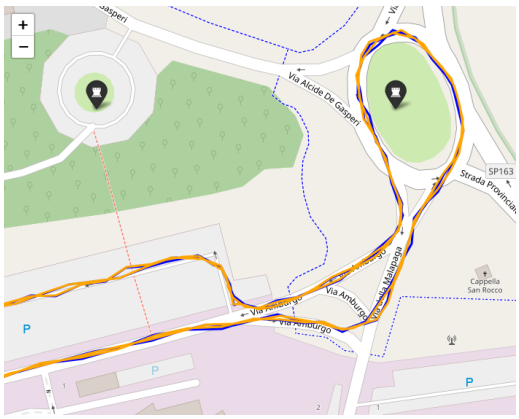
(a) $\sigma_{pr} = 0.1m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



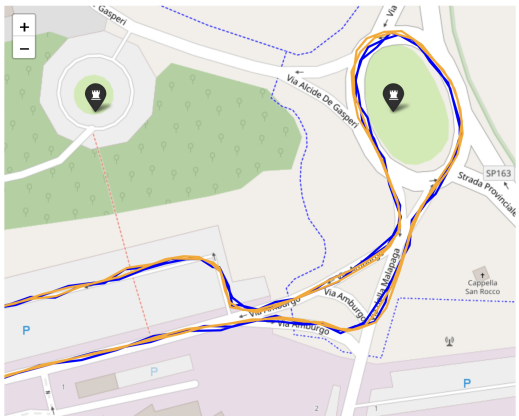
(b) $\sigma_{pr} = 0.1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



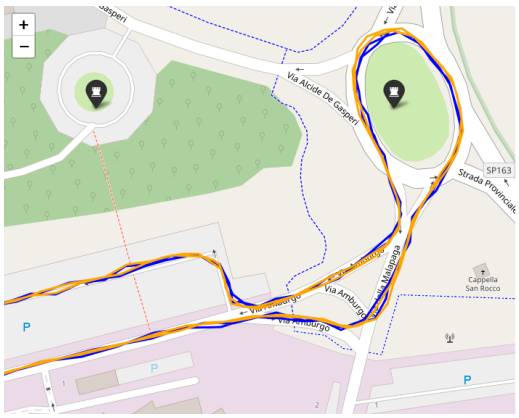
(c) $\sigma_{pr} = 1m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



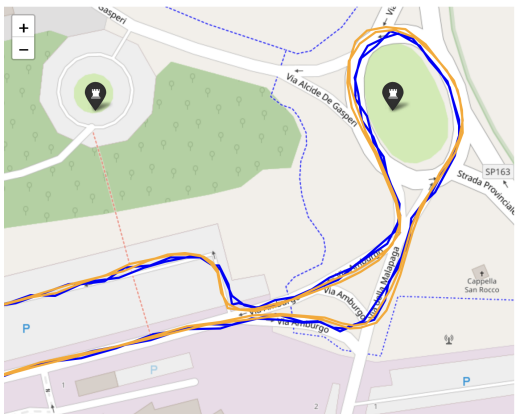
(d) $\sigma_{pr} = 1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



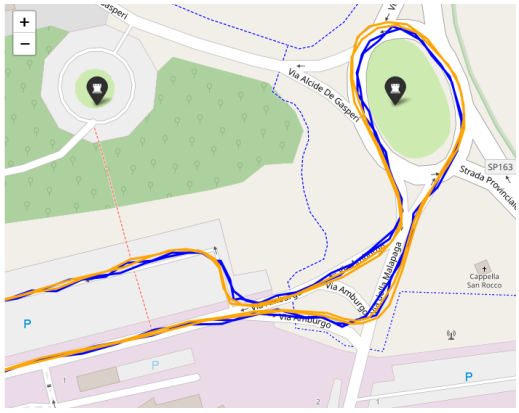
(e) $\sigma_{pr} = 2.5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



(f) $\sigma_{pr} = 2.5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$



(g) $\sigma_{pr} = 5m,\ \sigma_{pos} = 0.1m,\ \sigma_{vel} = 1\frac{m}{s}$



(h) $\sigma_{pr} = 5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$

6 Accuracy statistics

6.1 Least-Squares Method (LSM)

Table 3: LSM horizontal-error statistics

Metric	East [m]	North [m]	2-D [m]
Mean error	−0.017	−0.025	0.799
Standard deviation	0.476	0.807	0.489
Maximum |error|	1.801	2.682	2.844

6.2 Kalman filter

Table 4: KF horizontal-error statistics

$\sigma_{obs} = 2.5m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 1\frac{m}{s}$

Metric	East [m]	North [m]	2-D [m]
Mean error	−0.046	−0.033	1.712
Standard deviation	1.442	1.379	1.027
Maximum |error|	4.233	4.309	4.770

6.3 Extended Kalman filter

Table 5: EKF horizontal-error statistics

$\sigma_{pr} = 1m,\ \sigma_{pos} = 1m,\ \sigma_{vel} = 0.1\frac{m}{s}$

Metric	East [m]	North [m]	2-D [m]
Mean error	0.001	0.022	1.146
Standard deviation	0.666	1.217	0.782
Maximum |error|	2.197	4.380	4.523

7 Conclusion

We have established a local East–North–Up Cartesian frame around the T1 loop, simulated 5G-style pseudoranges incorporating a realistic clock-offset random walk, and recovered the user-equipment trajectory using three estimators: a planar least-squares solver, a constant-velocity Kalman smoother, and an extended Kalman filter. The epoch-by-epoch least-squares solution is unbiased but noisy, while the classical Kalman smoother produces a much smoother track, achieving a balance between responsiveness and oversmoothing. The extended Kalman filter, which jointly estimates pseudorange, velocity, and clock bias in a single nonlinear framework, offers comparable overall accuracy but requires tuning of its process and measurement noise parameters. Our Streamlit interface integrates these methods into an interactive application, allowing users to upload CSV files, adjust noise settings, execute each estimator, visualize the results on a map, and export both ENU and geodetic coordinates.