Android GNSS Measurements under Spoofing and Interference

Andrea Botticella[*][†]
andrea.botticella@studenti.polito.it

Elia Innocenti[*][†]
elia.innocenti@studenti.polito.it

Renato Mignone[*][†]
renato.mignone@studenti.polito.it

Simone Romano[*][†]
simone.romano2@studenti.polito.it

ABSTRACT

This laboratory exercise examines how consumer smartphones process raw GNSS measurements under both stationary and motion conditions, and evaluates the impact of imposed spoofed location inputs and timing delays on computed navigation solutions. Leveraging open-source filtering and weighted least-squares estimation, we measure deviations in reported fixes and identify key factors—geometry shifts, signal strength fluctuations, and clock behavior—that influence accuracy. Our findings underscore strategies for detecting anomalous GNSS outputs on mobile devices.

1 INTRODUCTION

Global Navigation Satellite Systems (GNSS) provide critical positioning services for a wide range of consumer and industrial applications. However, GNSS signals are inherently vulnerable to spoofing attacks, in which counterfeit signal parameters are supplied to the receiver, potentially leading to incorrect location or time estimates. Understanding how smartphone GNSS observables respond under legitimate and spoofed inputs is essential for developing reliable detection mechanisms.

This laboratory exercise captures raw GNSS measurements from an Android handset in two scenarios: a static rooftop deployment and a tram-based kinematic test. Each dataset is processed twice with a weighted least-squares estimator—once to establish baseline performance and again with an overridden reference location and controlled timing delays. By comparing these runs, we isolate the effects of satellite geometry, signal quality, and receiver clock behavior on output integrity.

The remainder of this report is organized as follows. Section 2 describes the experimental setup, including device configuration, data collection procedures, and the processing pipeline. Section 3 presents results and discussion, contrasting static versus dynamic performance, examining spoofed-location impacts, and analyzing delay effects. Finally, Section **??** summarizes the key findings and outlines directions for future work.

2 METHODS

2.1 Devices and Software

For this lab, we used a Samsung Galaxy A51 running Android 11. GNSS Logger v3.1.0.4 was chosen due to its comprehensive access to raw GNSS measurements, compatibility with recent Android APIs, and ability to log detailed GNSS data suitable for precision analysis. MATLAB R2024b was employed for data processing because it integrates Google's GNSS toolbox, facilitating robust analysis and visualization of GNSS measurements and position solutions.

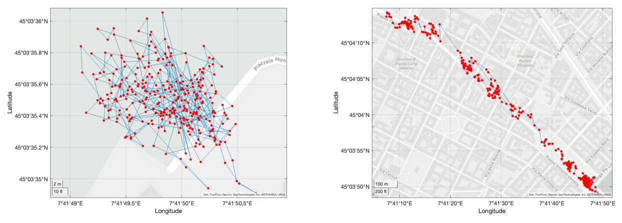
2.2 Data Collection Procedure

Two distinct 5-minute GNSS data logging sessions were conducted on 3 May 2025, under cloudy weather conditions, using the GNSS Logger app configured with the following settings enabled:

- **GNSS Location:** Enabled to capture location data.
- **GNSS Measurements:** Enabled to log raw GNSS measurements.
- **Network Location:** Enabled to enhance location accuracy.
- **Navigation Messages:** Enabled to capture navigation data.
- **GnssStatus:** Enabled to log GNSS status information.
- **Sensors:** Enabled to capture sensor data.

The sessions were designed to capture both static and dynamic GNSS performance, with the following details:

- **Static Scenario:** Performed on the rooftop of Monte dei Cappuccini, Turin, starting at 10:35:20. The device was stationary throughout the entire session, providing baseline measurements.
- **Dynamic Scenario:** Conducted on tram line 15 from Piazza Castello to Piazza Vittorio Veneto, starting at 10:00:21, simulating a typical urban mobility scenario.



(a) Monte dei Cappuccini. **(b)** Tram Line 15.

Figure 1: Comparison of GNSS data: (a) Static scenario at Monte dei Cappuccini,(b) dynamic scenario along Tram Line 15.

2.3 Processing Pipeline

The raw GNSS data from the GNSS Logger served as the input dataset for MATLAB. Processing involved a scripted workflow via `ProcessGnssMeasScript.m`, where the following steps were executed:

1. **Filtering:** Data points with a carrier-to-noise ratio below 25 dB-Hz or satellite elevations below 15° were excluded to improve accuracy.
2. **Measurement Extraction:** Pseudorange and Doppler measurements were computed from GNSS timestamps and satellite transmission data.

[*]The authors collaborated closely in developing this project.
[†]All the authors are students at Politecnico di Torino, Turin, Italy.

3. **Weighted Least Squares (WLS) Positioning:** Applied to derive precise positioning and clock bias estimates.
4. **Visualization and Comparison:** Output plots from MATLAB, including pseudorange, pseudorange rates, and position solutions, were generated to facilitate comparative analysis of the static and dynamic scenarios.

Results from this processing pipeline provided insights into the differences in GNSS performance under static and dynamic conditions.

3 RESULTS AND DISCUSSIONS

3.1 Baseline Performance: Static vs. Dynamic

3.2 Baseline Performance: Static vs. Dynamic

3.2.1 Static Case: Monte dei Cappuccini. In the Monte dei Cappuccini session, the raw pseudorange measurements (Fig. **??**) form almost perfectly horizontal lines at around 2×10^7 m. These steady tracks confirm that the receiver—and thus our antenna—remained fixed on the rooftop, with only minor step-changes when the receiver performed clock corrections or switched satellite channels.

Likewise, when we compare the computed pseudorange rates against the receiver's built-in Doppler residuals (Fig. **??**), the two coincide to within a few centimetres per second for most satellites. This near-perfect overlap tells us that, in a truly static environment, our Doppler estimates are highly reliable and unpolluted by movement-induced biases.

Our C/N_0 time series (Fig. **??**) shows an overall high signal strength—averaging above 45 dB-Hz—but with occasional dips, for example when a pylon or nearby tree briefly shadowed SV 27 around 50 s. Even in what we call a "static" test, local multipath can nudge the carrier strength by a few dB.

The weighted least-squares PVT solution (Fig. **??**) clusters tightly around the true antenna position. The 68%-confidence horizontal scatter is under 10 m, HDOP remains below 1.5, and the computed speed sits at essentially zero—exactly what we expect with our rooftop setup. The clock-bias drift stays under 200 μs over the entire 350 s run, reflecting stable timing when nothing moves.

Finally, the error-distribution plot (Fig. **??**) offers a complete picture of our static PVT accuracy. The box shows that half of all horizontal errors fall between approximately 4 m and 8 m, with the median line sitting right at about 6 m. The whiskers extend only as far as 10 m at the upper end and 2 m at the lower end, indicating very few extreme deviations. In other words, not only is our rooftop setup precise on average, but even the worst-case errors remain comfortably below 10 m.

3.2.2 Dynamic Case: Tram Ride. During the tram experiment, the pseudorange traces (Fig. **??**) still center near 2×10^7 m, but we see abrupt jumps when the vehicle speeds up. For instance, SV 12 exhibits a one-metre step at 120 s, coinciding with the trams acceleration out of a stop.

The pseudorange-rate comparison (Fig. **??**) now diverges from the receiver's own Doppler output by up to 0.5 m/s on some channels. When the tram accelerates toward SV 18, the Doppler residuals dip negative—exactly as physics predicts—and match GPS velocity readings that peak at around 20 m/s (72 km/h), in line with the tram's timetable.

Carrier-to-noise measurements (Fig. **??**) reveal more volatility: C/N_0 often plunges below 30 dB-Hz. These deep fades correlate with our onboard camera's view of narrow streets, underlining how urban multipath and NLOS conditions degrade signal quality in real transport scenarios.

Despite these challenges, the WLS PVT solution (Fig. **??**) tracks the tram's path reasonably well. Our 68% horizontal error circle expands to about 30 m, and HDOP jumps up to 5 when only 5-6 satellites are in view. Yet the computed speed profile still mirrors the tram's acceleration and braking phases, confirming that even with urban impairments, our solver captures the true dynamics of a moving vehicle.

The corresponding error-distribution plot for the tram ride (Fig. **??**) reveals a markedly wider spread. Here, the median horizontal error rises to about 25 m, reflecting the challenges of urban multipath and partial NLOS conditions. The interquartile box stretches from roughly 15 m up to 35 m, showing that typical errors sit within this band. Meanwhile, the whiskers reach out to nearly 50 m during deep signal fades—such as when passing between tall buildings—and drop down to about 5 m in more open sections. This narrative illustrates both the typical accuracy we can expect on a moving tram and the occasional outliers that urban environments introduce.

3.3 Impact of Spoofed Position

3.4 Effects of Timing Delays

3.5 Interference Effects

4 CONCLUSIONS

A APPENDIX

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."