On Optimizing Acquisition Functions for Improved Positioning Accuracy in SyDR

Ha My Nguyen[1], Antoine Grenier[1], Aleksandr Ometov[1] and Jari Nurmi[1]

[1]*Tampere Wireless Research Center, Tampere University, Korkeakoulunkatu 1, Tampere, Finland, FI-33720*

Abstract

This work-in-progress paper presents advancements in the acquisition function for processing GNSS signal snapshots, integrated within the System for Dynamic Repositioning (SyDR) framework. The primary objective of this research is to improve the accuracy and efficiency of GNSS signal processing, which is critical for applications such as navigation and positioning. The study addresses the challenges posed by complex wireless environments and the imperative need for energy-efficient solutions in embedded devices. Significant modifications were implemented in the existing positioning function, notably the removal of a Bayes classifier and the simplification of the acquisition function. The performance evaluation demonstrated substantial improvements in positioning accuracy, evidenced by a reduction in median errors and an increase of at least 5% in the percentage of errors below 200 meters for the test data. Despite these advancements, certain challenges persist, particularly concerning processing speed and performance issues encountered when integrating satellites from multiple GNSS systems. This paper contributes to the ongoing discourse on GNSS signal processing, offering insights into the optimization of acquisition functions, and highlighting areas for future research to address the remaining challenges.

Keywords

GNSS, IQ, benchmarking, open-source software

1. Introduction

Over the past decade, Global Navigation Satellite Systems (GNSS) receivers have become a cornerstone of modern industrial and person electronics, seamlessly integrated into a wide array of consumer and industrial devices [1, 2]. With the exponential growth of the Internet of Things (IoT) paradigm, the demand for precise and reliable positioning services has increased significantly. Forecasts suggest that the number of connected IoT devices is expected to reach 26 billion by 2025 [3], further amplifying the need for efficient and robust GNSS solutions.

However, the evolving usage scenarios of GNSS introduce considerable challenges. Embedded devices, particularly those operating in constrained environments, are often subject to adverse wireless conditions, including signal obstruction, multipath propagation, and intentional or unintentional interference. Moreover, unfavorable Dilution of Precision (DOP) metrics can significantly degrade positioning accuracy. The design and optimization of GNSS receivers for such scenarios require innovative strategies that ensure signal acquisition and tracking under limited resources. The modernization of GNSS constellations, offering enhanced signal structures, e.g., as additional frequencies, improved coding schemes, and increased signal power, represents a step forward in mitigating these limitations [4, 5]. Nevertheless, these advancements come at the cost of increased processing complexity [6].

In response to these constraints, the research community has placed significant emphasis on developing novel algorithms to enhance signal processing efficiency. Numerous techniques have been proposed to incorporate the benefits of modernized signals into practical receiver designs. Yet, the trade-offs between measurement accuracy, computational burden, and implementation feasibility remain non-trivial.

WIPHAL'25: Work-in-Progress in Hardware and Software for Location Computation June 10–12, 2025, Rome, Italy

alekasndr.ometov@tuni.fi (A. Ometov); jari.nurmi@tuni.fi (J. Nurmi)
0009-0004-9593-6174 (H. M. Nguyen); 0000-0002-3440-8659 (A. Grenier); 0000-0003-3412-1639 (A. Ometov); 0000-0003-2169-4606 (J. Nurmi)

A critical challenge for embedded GNSS receivers is power consumption, particularly in battery-operated or energy-constrained devices. According to the 2024 GNSS User Consultation Platform, majority of respondents identified power consumption as a major design concern [2]. GNSS modules are often among the most power-intensive components in IoT nodes [7], primarily due to the continuous operation required to maintain accurate positioning. While techniques such as duty cycling have been explored [8, 9], their practical application is limited by startup latency and cold-start acquisition requirements. Offloading strategies, where raw GNSS data are transmitted to remote servers for processing, offer energy savings at the cost of positioning accuracy [10, 11]. These approaches warrant further investigation to assess their effectiveness across diverse application domains [12].

In this context, our ongoing research focuses on enhancing the acquisition function within the System for Dynamic Repositioning (SyDR) framework, first introduced in [13]. The objective is to integrate robust snapshot signal processing capabilities that are well-suited for constrained embedded platforms. Specifically, we target improvements in positioning accuracy, algorithmic efficiency, and support for multi-constellation scenarios.

The key contributions of this work include: (i) proposing the modification of the existing positioning function to support modern GNSS signals, (ii) initial simulation of representative test cases to validate performance under controlled conditions. Preliminary evaluations indicate measurable improvements in positioning accuracy. However, challenges remain, particularly in managing computational load and maintaining real-time responsiveness when handling signals from multiple GNSS systems, as well as factual integration in SyDR. Overall, this work contributes to the broader effort of developing energy-aware, high-performance GNSS solutions for next-generation IoT and cyber-physical systems.

The rest of the paper is organised as follows. First, we identify the main research objectives in Section 2. Selected numerical results are provided in Section 3. The last section concludes the paper.

2. Research Objectives

A target platform is depicted in Figure 1. SyDR is an open-source SDR framework, implemented in Python and available via GitHub [14], specifically designed to facilitate benchmarking of GNSS algorithms. The design philosophy of SyDR is founded upon five fundamental principles, adapted from GNSS-SDR [15] and developed towards [13, 16], which define its intended functionality and usage for interoperability, usability reproducibility, openness, and efficiency.

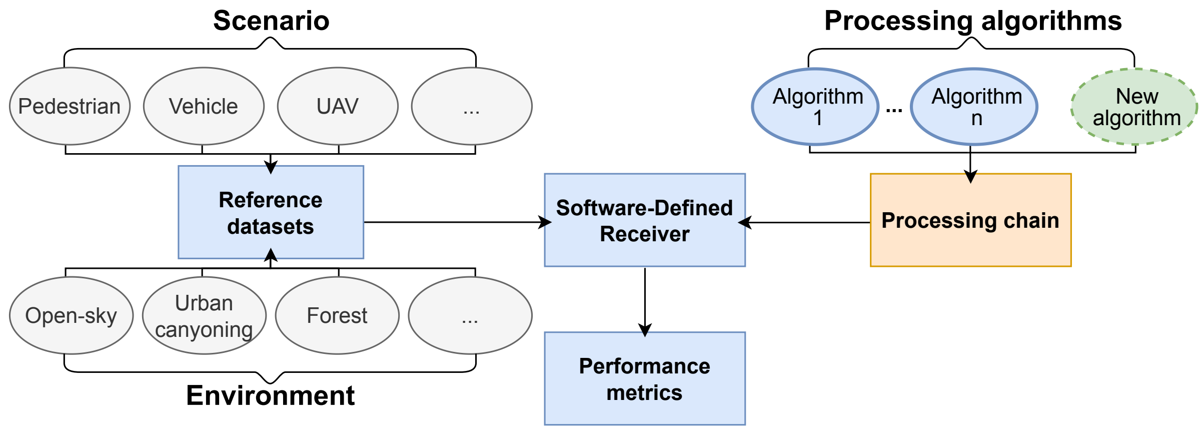

Figure 1: An overview of SyDR's framework

From its inception, SyDR was developed with algorithm benchmarking as a primary objective. It delivers a fully controllable and extensible framework that can be adapted to accommodate specific research requirements. In this respect, SyDR exhibits distinct differences from previously introduced SDR platforms.

First and foremost, SyDR is designed exclusively for post-processing applications. It processes pre-recorded IQ samples while emulating the behavior of a real-time receiver. Post-processing enables the establishment of deterministic receiver behavior, which is essential for rigorous algorithm evaluation. Since GNSS data significantly affects algorithm performance, benchmarking must be conducted within a well-defined and repeatable environment to ensure that observed variations stem solely from the algorithm under test. The pseudo-real-time characteristics of the system allow for deeper insight into the virtual receiver's behavior and the effect of algorithmic modifications across the signal processing pipeline.

Secondly, the framework emphasizes a high-level, modular development environment utilizing an open-source language. While many high-level SDR platforms have historically been implemented in proprietary environments such as Matlab, Python was chosen to maintain an open and community-driven development model while achieving satisfactory code efficiency. Modularity has often been a limiting factor in prior SDR platforms, which were typically tailored for evaluating a narrow set of algorithms rather than serving as comprehensive benchmarking tools.

Thirdly, the evaluation of energy consumption forms a key research direction for SyDR. Absolute power consumption figures are difficult to derive for specific hardware implementations when working with high-level software environments. In this context, algorithmic complexity serves as a practical proxy, offering relative performance comparisons between algorithms. Although this metric does not directly reflect real energy usage, it provides meaningful insights into computational demands. Furthermore, by simulating the full receiver behavior, SyDR allows researchers to assess the impact of algorithmic changes on the overall system.

Ultimately, SyDR aims to serve as a reference platform for future research efforts in GNSS algorithm development, mitigating redundant implementation efforts. Although this work has primarily concentrated on the DSP aspects of the receiver, the framework's modular design positions it as a promising foundation for further advancements in areas such as high-precision positioning and advanced signal processing techniques.

3. Selected Experimental Results

With an end-to-end virtual receiver like SyDR, it becomes possible to estimate the complexity of algorithms within the GNSS signal processing chain. While this estimation is inherently coarse, subject to variations stemming from code quality, operating system overhead, and other environmental factors, it still offers a practical means of assessing relative algorithmic complexity through runtime measurements.

To enable this functionality, the `time` module from Python's standard library has been employed, as recommended by the official documentation [17]. It is integrated into the codebase using a Python `decorator`, allowing benchmarking to be seamlessly added to any function. Owing to SyDR's database architecture, every function wrapped with the `benchmarking.time` decorator logs its runtime as part of the intermediate results. This setup provides flexibility in measurement granularity, allowing users to identify the most time-consuming sections of processing with precision.

The primary development in this project involved improving the acquisition function to enhance snapshot positioning and integrating it into the broader SyDR framework. The numerical results obtained from this updated acquisition function demonstrated substantial improvements in positioning accuracy.

3.1. Snapshot Positioning Code Modifications

Only minimal changes were necessary to adapt the existing positioning function. A Bayes classifier was initially implemented to prioritize satellite selection based on their SNR values. However, it offered no advantage over simpler SNR-based ranking and was thus removed. Additionally, support for processing multiple snapshots in a single call was removed to simplify functionality.

The original acquisition function in the library produced suboptimal results, even when supplied with accurate input data such as satellite visibility and expected Doppler values. Its reliance on complex high-dimensional vectorized operations made it difficult to interpret and debug. As a result, a complete reimplementation was undertaken. The new acquisition function combines vectorized 3D array operations with traditional `for`-loops, resulting in clearer and more maintainable code.

This revised implementation, using the same input parameters as the customized version, yielded markedly improved positioning performance. Specifically, the median position error decreased, and the percentage of errors under 200 meters increased by at least 5% on the test datasets. However, these gains came at the cost of reduced processing speed as the new function currently operates approximately ten times slower than the original.

3.2. Performance on Data Obtained in TAU

Initial tests using TAU laboratory data yielded poor results due to an incorrect assumption about the data format. Upon discovering that the recordings were in IQ format (rather than real-valued), appropriate modifications were made. After these adjustments, the updated acquisition function performed well, albeit limited to one GNSS system at a time.

The simulation setup employed Spectracom (a.k.a., Orolia) and a USRP. Early issues involving the unavailability of trajectory and RINEX files [18] were resolved via firmware and software updates. Simulated datasets were created under the following configurations:

- **Static files**: Contain signals from a single satellite.
- **Dynamic files**: Include data from five satellites, with known PRN IDs.
- **Transmit Power and C/N0**: Adjusted for each scenario. Note that power adjustments are possible only on the hardware simulator, not in StudioView.
- **Recording Duration**: One minute for static and five minutes for dynamic scenarios.

Sampling was conducted using 40 MHz and 16 MHz rates, with intermediate frequencies of 10 MHz and 4 MHz, respectively. Two processing phases were developed for pre-processing:

- `resampling`: Implements zero-padding, upsampling, downsampling, resampling, and linear correlation via overlap-and-add.
- `file_to_snapshots`: Converts input recordings into snapshot data suitable for positioning.

These tools collectively enable the generation of consistent and controllable GNSS datasets, aiding in system testing and evaluation.

3.3. Main Findings

A total of 7 static and 4 dynamic simulated files (each with different $CN0$ values) were recorded, dedicated to GNSS data. Visualization of these results is supported through the `TAU data.ipynb` notebook, which provides plots and statistics derived from `results.npy`, see, e.g., Table 1 and Figure 2. These visualizations include error metrics and per-mode performance summaries.

Scenario	Min East	Max East	Median East	RMS East	SD East	Error <50m, %
5MHz_1bit_IQ_gain_25	-20.29	72324.01	0.72	9047.28	8968.86	88
5MHz_8bit_IQ_gain_25	-15.42	19.70	-1.68	6.93	6.82	98
10MHz_1bit_IQ_gain_25	-6.41	7.97	-0.80	3.88	3.87	100
10MHz_1bit_IQ_gain_25	-6.67	6.21	-1.03	3.28	3.18	100

Table 1
Example of statistical data for LS-Single mode `truncatednortherror` in meters.

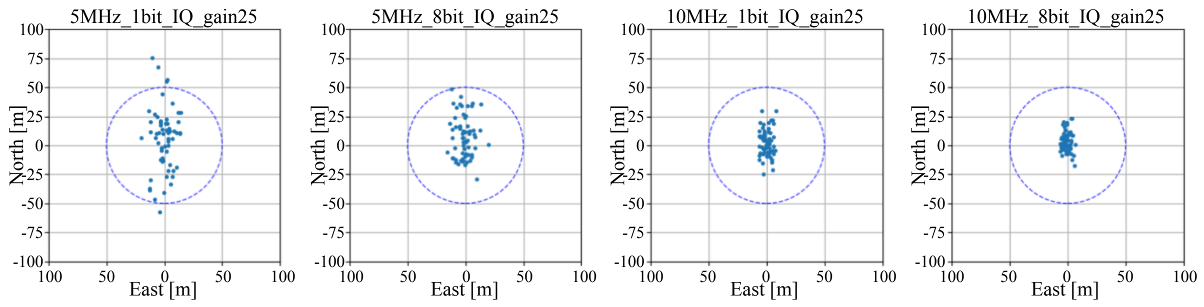

Figure 2: Visualization of positioning errors

The results demonstrate substantial improvements in positioning accuracy, particularly in the later test cases. First, we observe a very high maximum error, suggesting significant outliers in the dataset, but the low median error indicates that most errors are negligible. Further cases show better performance, with maximum errors below 10 meters and median errors close to zero, indicating high accuracy. The high percentages of errors within 50 meters in these rows highlight the effectiveness of the updated acquisition function. However, the presence of extreme outliers underscores the need for further refinement and testing to ensure consistent performance across all scenarios.

The results align with expectations, i.e., positioning errors decrease with higher sampling rates and greater quantization depth. However, a significant issue arises when multiple GNSS systems are processed simultaneously. While the function performs well with a single GNSS system, accuracy degrades substantially when combining satellites from different systems. This issue also affects the *ls-combo* satellite selection mode, which evaluates all possible combinations of visible satellites and has been found to underperform relative to simpler selection schemes. This is a concerning outcome, as combining data from more visible satellites is theoretically expected to improve, not worsen, positioning accuracy. Overall, these findings highlight the potential of the proposed modifications to enhance positioning accuracy while also identifying areas for further optimization.

4. Conclusions and Future Work

This research successfully enhanced the acquisition functionality for snapshot-based GNSS signal processing and integrated it into the SyDR framework. The improvements achieved, particularly in positioning accuracy, demonstrate the potential of the proposed modifications. However, challenges remain, especially when processing multiple GNSS constellations concurrently, which has led to degraded performance and requires further investigation.

The key findings from this work include improved positioning accuracy, with the updated acquisition function significantly reducing the median position error and increasing the percentage of errors under 200 meters. These improvements highlight the effectiveness of the new implementation in enhancing positioning accuracy. However, while the function performs well with a single GNSS system, accuracy degrades substantially when combining satellites from different systems. This issue also affects the *ls-combo* satellite selection mode, which underperforms relative to simpler selection schemes. This unexpected outcome indicates the need for further optimization and testing.

Future work will focus on addressing the identified limitations in multi-constellation processing and implementing SyDR on real hardware [16]. Specific areas for future research include ensuring the decoupling of acquisition pipelines for each GNSS system to improve performance when processing multiple constellations. Furthermore, resampling and signal preparation routines should be moved outside the acquisition block to reduce computational overhead and enhance overall system efficiency. Implementing SyDR on real hardware will validate the framework's performance in practical scenarios and further refine the acquisition function.

In summary, this work has established a solid foundation for further research in GNSS signal processing using SyDR. The insights gained and the system improvements realized contribute meaningfully to the evolution of the SyDR platform and its applicability in real-world virtual receiver implementations.

Acknowledgments

The authors gratefully acknowledge funding from European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska Curie grant agreement No. 956090 (APROPOS: Approximate Computing for Power and Energy Optimisation, http://www.apropos-itn.eu/)

Declaration on Generative AI

During the preparation of this work, the authors used Microsoft Copilot in order to: styling, grammar and spelling checks. After using these tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] S. Jin, A. Camps, Y. Jia, F. Wang, M. Martin-Neira, F. Huang, Q. Yan, S. Zhang, Z. Li, K. Edokossi, et al., Remote Sensing and Its Applications Using GNSS Reflected Signals: Advances and Prospects, Satellite Navigation 5 (2024) 19.

[2] European Union Agency for the Space Programme (EUSPA), EUSPA: EO and GNSS Market Report, 2024. URL: https://www.euspa.europa.eu/european-space/euspace-market/gnss-market, issue 2.

[3] 6GWorld, Ericsson: Short-range IoT Set to Reach More than 20 Billion Connected Devices by 2026 (2024). URL: https://www.6gworld.com/exclusives/ericsson-short-range-iot-set-to-reach-more-than-20-billion-connected-devices-by-2026/, accessed: June 10, 2025.

[4] X. Luo, H.-H. Chen, Q. Guo, LEO/VLEO Satellite Communications in 6G and Beyond Networks–Technologies, Applications, and Challenges, IEEE Network 38 (2024) 273–285.

[5] I. Fernandez-Hernandez, A. Chamorro-Moreno, S. Cancela-Diaz, J. Calle-Calle, P. Zoccarato, D. Blonski, T. Senni, F. Blas, C. Hernández, J. Simón, A. Mozo, Galileo High Accuracy Service: Initial definition and Performance, GPS Solutions 26 (2022). doi:10.1007/s10291-022-01247-x.

[6] A. Grenier, E. S. Lohan, A. Ometov, J. Nurmi, On the Integration of Approximate Computing in GNSS Signal Processing for Improved Energy-Efficiency, in: Proc. of 11th Workshop on Satellite Navigation Technology (NAVITEC), IEEE, 2024.

[7] S. Narayana, R. V. Prasad, V. Rao, L. Mottola, T. V. Prabhakar, Hummingbird: Energy Efficient GPS Receiver for Small Satellites, in: Proc. of the 26th Annual International Conference on Mobile Computing and Networking, MobiCom '20, Association for Computing Machinery, New York, NY, USA, 2020. URL: https://doi.org/10.1145/3372224.3380886. doi:10.1145/3372224.3380886.

[8] V. Bellad, Intermittent GNSS Signal Tracking for Improved Receiver Power Performance, Ph.D. thesis, University of Calgary, 2015. URL: https://prism.ucalgary.ca/handle/11023/2667. doi:10.11575/PRISM/10182.

[9] T. Everett, T. Taylor, D.-K. Lee, D. M. Akos, Optimizing the use of RTKLIB for smartphone-based GNSS measurements, Sensors 22 (2022) 3825.

[10] A. Grenier, E. S. Lohan, A. Ometov, J. Nurmi, A Survey on Low-Power GNSS, IEEE Communications Surveys & Tutorials (2023).

[11] P. Misra, W. Hu, Y. Jin, J. Liu, A. S. de Paula, N. Wirström, T. Voigt, Energy Efficient GPS Acquisition with Sparse-GPS, in: Proc. of the 13th International Symposium on Information Processing in Sensor Networks (IPSN-14), 2014, pp. 155–166. doi:10.1109/IPSN.2014.6846749.

[12] H. M. Nguyen, Review of Off-loading Processing Strategies for GNSS Positioning: Theoretical Review and Visual Analysis Implementation, 2024. URL: https://trepo.tuni.fi/handle/10024/156652, examiners: Antoine Grenier, Simona Lohan.

[13] A. Grenier, E. S. Lohan, A. Ometov, J. Nurmi, An Open-Source Software-Defined Receiver for GNSS Algorithms Benchmarking, in: Proc. of 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), IEEE, 2022, pp. 31–38.

[14] A. Grenier, SyDR, [Online] https://github.com/aproposorg/sydr, 2023. [Accessed on June 10, 2025].

[15] C. Fernández–Prades, J. Arribas, P. Closas, C. Avilés, L. Esteve, GNSS-SDR: An Open Source Tool For Researchers and Developers, in: Proc. of the 24th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2011), Portland, OR, 2011, pp. 780–794.

[16] A. Grenier, J. Lei, H. J. Damsgaard, E. S. Quintana-Ortí, A. Ometov, E. S. Lohan, J. Nurmi, Hard SyDR: A Benchmarking Environment for Global Navigation Satellite System Algorithms, Sensors 24 (2024) 409.

[17] Python, Time Access and Conversions, https://docs.python.org/3/library/time.html, 2025. Last accessed: June 10, 2025.

[18] International GNSS Service, RINEX Working Group, https://igs.org/wg/rinex/, 2025. Accessed: June 10, 2025.