In-Production Benchmarking for Automatic Detection of Position Errors in Indoor Localization

Silas Ulrich[1], Janis Tiemann[1], Andreas Lewandowski[1] and Christof Röhrig[2]

[1]Comnovo GmbH, Robert-Schuman-Str. 6, 44263 Dortmund, Germany
[2]University of Applied Sciences and Arts Dortmund, Otto-Hahn-Str. 23, 44227 Dortmund, Germany

Abstract

Indoor positioning systems are crucial in logistics and industrial applications for process optimization, asset tracking, safety features, and automation. However, systems that rely on dead reckoning or other sensor fusion techniques are prone to drift and require frequent recalibration. Ground truth data is often necessary to evaluate these systems, but another approach involves identifying specific features in the data stream that indicate system misbehavior. This paper presents a benchmark architecture for evaluating the performance of indoor positioning systems. This architecture identifies defined events in the data stream that indicate insufficient system performance. It also describes how to recreate the system state during events based on collected data during these events for further analysis. This approach aids in identifying root causes and expediting the development process of indoor positioning systems in real-world operational scenarios. Evaluation results show an increase in position accuracy and improved detection of system misbehavior over time when using the benchmark architecture.

1. Introduction & Related Work

The deployment of Indoor Positioning Systems (IPSs) or Real-Time Locating Systems (RTLSs) in general is crucial in logistics and industrial applications for optimizing processes, tracking assets, enhancing safety, and automating operations. The accuracy and reliability of these systems are paramount, especially in environments where driver behavior impacts the positioning system, as revealed through predefined tests or simulations. In Autonomous Vehicles (AVs) research, similar challenges exist with positioning accuracy in industrial environments. Various studies evaluate AV systems through simulations and test scenarios modeled on the field of application. Thorough evaluation is essential before deploying AVs to the public due to the inefficiency and expense of public road testing, necessitating robust automated testing and simulations [1]. An example is Gomez-Huelamo et al. [2], which use a Robot Operating System (ROS) simulation layer over CARLA [3] for autonomous driving research. This approach provides insights into AV systems' interactions with real-world conditions. The global AGV market, valued at $3.29 billion in 2019, is projected to reach $9.59 billion by 2028, growing at a Compound Annual Growth Rate (CAGR) of 12.62% [4]. In 2019, the market segments based on revenue included forklift trucks at 17.21% and pallet trucks at 13.01%, highlighting their significant impact on improving warehouse operations [4]. Mixed fleet scenarios, combining manual and automated systems, highlight the impact of manual driving on positioning systems, necessitating real-world feedback and robust testing. Recent advancements in Ultra-Wideband (UWB) technology in indoor positioning systems emphasize high accuracy and resilience. Van Herbruggen et al. [5] underscore the need for a multi-metric benchmarking approach to evaluate UWB systems, considering factors like line-of-sight conditions and algorithm selection. Their findings reveal the complexity of optimizing these systems, indicating no general solution exists.

Indoor localization is often evaluated in controlled environments, potentially biasing system performance [6, p.12]. Real-world data is crucial for accurate system evaluation. The Microsoft Indoor Localization Competition highlighted performance variations between controlled and real-world envi-

Proceedings of the Work-in-Progress Papers at the 14th International Conference on Indoor Positioning and Indoor Navigation (IPIN-WiP 2024)
ulrich@comnovo.de (S. Ulrich); tiemann@comnovo.de (J. Tiemann); lewandowski@comnovo.de (A. Lewandowski); christof.roehrig@fh-dortmund.de (C. Röhrig)

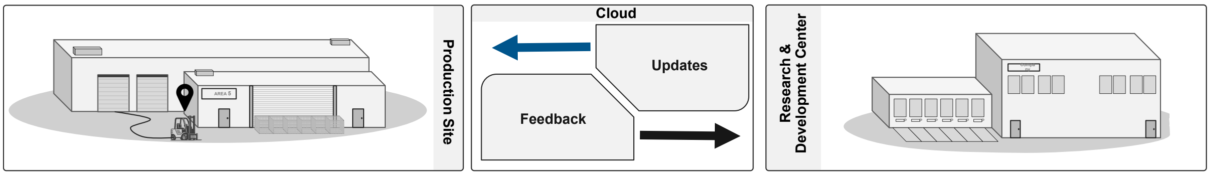


Fig. 1: Benchmarking system detecting and analyzing positioning system behavior in dynamic logistic environments.

ronments, emphasizing the need for comprehensive testing. Using RTLS for collision avoidance with manual vehicles in 5G environments [7] further underscores the importance of real-world testing, demonstrating how advanced technologies can enhance system robustness and safety. Sensor fusion approaches, such as those by [8], combining UWB with odometry data, simplify hardware setups. Their research on an AV car demonstrates the potential for streamlined installations, suggesting that benchmarking systems should also cover installation setups and detect unwanted behavior to improve system quality. As Fig. 1 illustrates, a system detecting and propagating information about system behavior during operation is beneficial. This includes the local machine running the position calculation and a central server for data storage and analysis, designed for real-world scenarios. The Indoor Positioning and Indoor Navigation (IPIN) challenge [9] highlights the complexity of evaluating indoor localization solutions through competitive benchmarking, suggesting diverse datasets and operator knowledge influence the evaluation process. Vedder et al. [10] demonstrate real-time testing with fault injection in AVs using UWB systems, evaluating robustness under faulty conditions. [11] emphasize realistic testing conditions in industrial scenarios, showcasing anchor placement's importance for accuracy. Amini et al. [12] present a simulation framework converting real-world data into a simulated perception-control API, enhancing the simulation's authenticity and underscoring real-world data's importance. Ortega et al. [13] use *rosbag*[1] in ROS for capturing and analyzing log files during service robot testing. This method facilitates precise data capture, aiding performance evaluation and issue identification. Other formats include *rosbag2*[2] and *mcap*[3].

In contrast to other works in the field of indoor localization, we propose an in-production systematic approach, consisting of a system architecture and an automatic method to identify potentially position error-related events. This is done using event definitions to identify such situations and collect data enabling localization engine state reproduction. Our benchmarking system runs parallel to the RTLS, propagating error event data to a central server for analysis, designed for real-world use. Current evaluation approaches using simulations or predefined tests are insufficient due to the varied logistics environments, leading to unaddressed edge cases. Our proposed benchmark architecture aims to detect and analyze system behavior during operation, speeding up development, reducing test complexity, and providing real-world feedback.

2. System Architecture

To build up a benchmarking system that fits with a variety of localization engines used in logistics and industrial applications, we propose a system architecture that can be used in real-world scenarios shown in Fig. 2. It is divided into three main segments: the on-premise production site, the cloud component managing data flow, and the development interface utilizing data from the cloud for further development. The goal of this architecture is to analyze RTLS data in real-time and detect anomalies in the data stream, reducing the amount of diagnostic data to an external server for further analysis and development. The architecture is designed to be scalable and adaptable to various localization engines and sensor setups.

[1]https://wiki.ros.org/Bags/Format
[2]https://wiki.ros.org/Bags/Format/2.0
[3]https://mcap.dev/

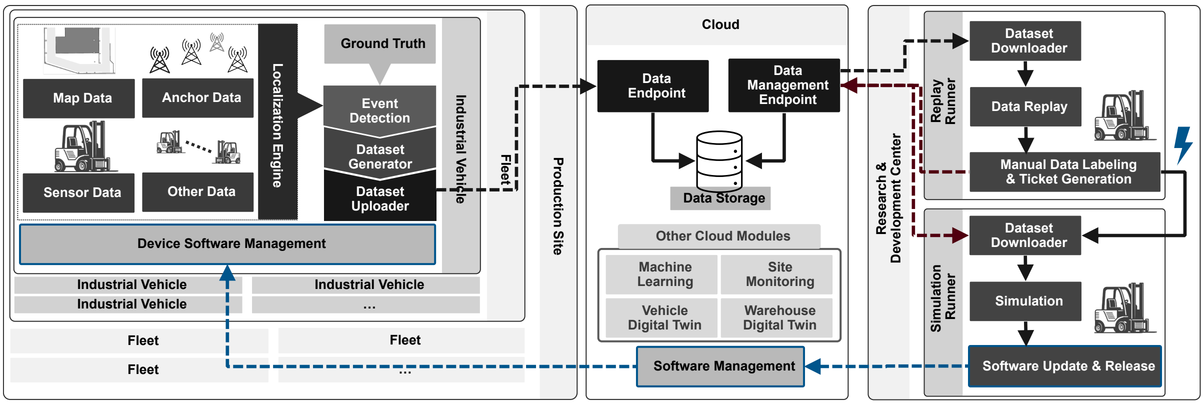


Fig. 2: System architecture of the benchmarking system, seperated into three main segments: the on-premise production site, the cloud component managing data flow, and the development interface utilizing data from the cloud for further development.

The objective of this paper is to present a systematic approach to evaluate and improve system accuracy. Various metrics can identify positional inaccuracies or faulty sensor data, revealing inconsistencies in the position stream. These events can be detected at the sensor level by analyzing the sensor value range, missing values, or large variations. Additionally, map-based collision detection with impassable objects can highlight incorrect sensor data. At a higher abstraction level, position jump detection can also indicate errors. Cross-referencing data from other high-accuracy systems, such as other forklifts, can further enhance error detection capabilities.

At the production site, illustrate in Fig. 2 at the left, the localization engine operates on the vehicle, exemplified by a forklift in the diagram. This engine is an abstract representation of a RTLS that can be implemented using various technologies such as Kalman filter [14] and or other state estimators [15], which are often employed for sensor fusion in indoor positioning systems as discussed in numerous publications [16, 17, 18]. Advanced techniques may integrate sensor data with map information for localization purposes, including Simultaneous Localization and Mapping (SLAM) [19] and camera-based visual SLAM [20] methods.

A ground truth system with higher accuracy than the system under test allows for performance evaluation by comparing ground truth data with system data, thus identifying discrepancies and providing direct feedback on system accuracy.

In our architecture, we combine localization engine data, including incoming sensor data, with ground truth data to detect events that may indicate undesirable system behavior. The Event Detection block implements interrupt functions triggered based on the sensor set and localization technique parameters.

Upon event detection, the data necessary for event recreation, including raw sensor data, optional ground truth data, and system state information, is compiled into a dataset. This dataset is then transmitted to a backend by the Data Uploader for further analysis. By targeting specific error-related data through the Event Detection functions, the volume of data gathered for testing can be minimized, focusing on the essential information needed for accurate error analysis.

The cloud component manages data flow and provides an interface for development. Data from multiple vehicles is uploaded to the cloud, where it is processed by various modules, including data endpoints, data management endpoints, and other cloud modules such as Machine Learning, Site Monitoring, Vehicle Digital Twin, and Warehouse Digital Twin.

Fig. 2 shown on the right, the Replay Runner is utilized by a human operator who labels event scenarios. An operator examines the data to identify false positives that can be ignored and detects issues that require development attention. If an issue is identified, the operator generates a ticket and links the relevant data for the development team to address. The Simulation Runner is used for an in-depth analysis and debugging of scenarios. It provides detailed outputs to help identify and fix issues. Once a software fix is validated in the simulation environment, the update is remotely deployed to the vehicle or fleet. This ensures that the software running on the vehicles is continuously improved based on real-world data and feedback.

This architecture creates a robust feedback and update loop. Initially, the localization engine detects events at the production site, and the relevant data is collected and uploaded to the cloud. The Replay Runner allows for human intervention to label and categorize the events, generating tickets for issues that need development. The Simulation Runner then enables a detailed analysis and debugging process, facilitating the identification and resolution of issues. Once fixes are made, software updates are deployed back to the vehicles, ensuring the system evolves and improves continuously.

By integrating these components, the architecture supports robust error detection and system performance analysis, facilitating continuous improvement in localization accuracy and reliability.

3. Event Detection

Our localization engine is based on a filter that gathers sensor data from a forklift and ranging measurements from a sparse UWB system using Two-Way Ranging (TWR). The vehicle is equipped with a computing unit connected to the sensors via a Controller Area Network (CAN) bus. We equipped a reference-vehicle with a Commercial Off-The-Shelf (COTS) Ground Truth (GT) system. The GT system provides accurate position data, which is used to evaluate the performance of the localization engine. It is connected to the vehicles's computing unit via Ethernet, providing real-time position data to the localization engine. For the event detection several techniques are used to detect errors in the localization engine. The section is split into parts, where the first is dedicated to a setup that can be used without ground truth data available on vehicles, especially in our scenario where an estimating filter is used for localization. The second part is dedicated to the use of ground truth data to detect errors in the localization engine.

3.1. Techniques for Internal Localization Engine Detection

Given a sensor fusion system like our filter, having vehicle and UWB data, the following techniques can be used to detect errors in the localization engine.

Sliding Window of Sensor Values and Speed Influence

A significant increase in the standard deviations of position $p = (x, y)^\intercal$ and orientation (θ) can indicate potential errors in the localization engine. Implementing a sliding window technique ensures consistent error detection and mitigation over time. By concurrently monitoring orientation and position variance, the system can more accurately identify potential errors. For each time step t, the standard deviations of the position and orientation of the state estimations are calculated as follows:

$$\sigma_{p,\text{state},t} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((x_i - \mu_{x,t})^2 + (y_i - \mu_{y,t})^2)}, \tag{1}$$

$$\sigma_{\theta,\text{state},t} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\theta_i - \mu_{\theta,t})^2}. \tag{2}$$

Subsequently, a sliding window is applied to these standard deviations:

$$\mu_{\sigma_p,t} = \frac{1}{w}\sum_{i=0}^{w-1}\sigma_{p,\text{state},t-i} \quad \text{and} \quad \mu_{\sigma_\theta,t} = \frac{1}{w}\sum_{i=0}^{w-1}\sigma_{\theta,\text{state},t-i}. \tag{3}$$

The conditions for detecting potential errors are defined as instances when the sliding window mean standard deviations of position and orientation exceed predefined thresholds:

$$\mu_{\sigma_p,t} > \sigma_{p,\text{threshold}} \wedge \mu_{\sigma_\theta,t} > \sigma_{\theta,\text{threshold}}. \tag{4}$$

These equations represent the methodology for calculating the standard deviation of position and orientation for state in our filter, followed by the application of a sliding window to these standard deviations. It should be noted, that estimators typically already provide such a metric such as covariance-matrix, or belief that could be utilized after or without some form of sliding window filtering. Based on these results, the system flags potential errors when the computed sliding window standard deviations and the vehicle speed exceed predefined thresholds.

Plausibility Checks Filter

Incorporating plausibility with our filter can enhance accuracy by eliminating colliding state estimates with pre-known non-plausible ones. However, due to odometry drift, extensive non-plausible state depletion or collisions leading to total state loss can provide insights into position reliability. By monitoring the ratio of remaining state N_{current} estimates after a plausibility check to the total count N_{max}, error scenarios can be identified. The deletion rate of the current state count can be calculated as:

$$\Delta_{\text{rate}} = \frac{N_{\text{max}} - N_{\text{current}}}{N_{\text{max}}}. \tag{5}$$

This deletion rate Δ_{rate} is then added to the total state ratio deleted since the start:

$$\Delta_{\text{total}} = \sum_{t=0}^{T} \Delta_{\text{rate},t}. \tag{6}$$

The total deletion ratio provides a cumulative measure of non-plausible state loss over time, which is indicative of the system's position reliability. It is important to note that this total deletion ratio is reset whenever a UWB position calculation is performed. This reset ensures that our filter can reinitialize with fresh data, thereby maintaining the accuracy and reliability of the localization system. By systematically monitoring these metrics, the system can identify potential error scenarios and make necessary adjustments to improve position reliability.

Sensor Calibration Age

Vehicle data, when used for odometry calculations, necessitates in-operation calibration to mitigate drift. This approach is akin to the one employed in Pedestrian Dead Reckoning (PDR) [21, 22], and can be similarly applied to vehicle localization systems. By integrating the last calculation time into error detection, it becomes feasible to correlate potential inaccurate measurements with drift caused by inadequate calibration. This correlation is represented by κ_γ, with thr serving as the threshold for calibration age.

$$\kappa_\gamma = \sqrt{\sum_{j=1}^{3} \left(\frac{1}{\gamma_{\text{measured},j}} - \frac{1}{\gamma_{\text{bias},j}} \right)^2}, \tag{7}$$

$$\kappa_{\gamma,\text{last}} > \kappa_{\text{thr}}. \tag{8}$$

Time Since Last Reliable Measurement

In parallel to the calibration detection κ_γ, the time since the last absolute position calculation based on UWB range measurements can be used to detect potential errors. This time is represented by Ω_{last}, where Ω denotes the time of the last reliable UWB measurement. The threshold for the maximum allowable time since the last reliable UWB measurement is represented by Ω_{thr} with

$$\Omega_{\text{last}} > \Omega_{\text{thr}}. \tag{9}$$

Discrepancy Detection in Position Updates upon UWB Zone Entry

Abrupt variations in position updates, especially during the transition from relative to absolute positioning via UWB measurements, may signify potential inaccuracies. Such discrepancies, surpassing a predetermined threshold associated with the minimum accuracy requisites of the RTLS, warrant further scrutiny.

The positional change at each time step t is computed as the absolute difference between the current position $\mathbf{p}_{curr}$ and the position at the most recent UWB measurement $\mathbf{p}_{\text{UWB,last}}$:

$$\Delta_{\text{pos}} = \left|\mathbf{p}_{curr} - \mathbf{p}_{\text{UWB,last}}\right|. \tag{10}$$

Should this positional change surpass a predetermined threshold, it necessitates further examination:

$$\Delta_{\text{pos}} > \Delta_{\text{threshold}}. \tag{11}$$

Moreover, the surveillance of abrupt orientation alterations is advantageous, as they can be indicative of implementation inaccuracies.

3.2. Techniques for Ground Truth-Based Error Detection

Having access to accurate GT data allows to rate the system accuracy on a position level. The following techniques can be used to detect errors in the localization engine based on the GT data.

Position Offset Detection

The Euclidean distance between the estimated and GT positions serves as a measure of localization error. This metric is particularly useful in assessing the installation quality of UWB setups in UWB enabled areas, as factors such as improper mounting or obstructive objects can result in erroneous measurements. Furthermore, the GT provides a valuable benchmark for evaluating the parameterization of the sensor fusion system, supplementing other error metrics to isolate and identify specific components of the sensor fusion system that may be contributing to errors.

$$d_{\text{Euclidean}} = \sqrt{(x_{\text{GT}} - x_{\text{est}})^2 + (y_{\text{GT}} - y_{\text{est}})^2}. \tag{12}$$

This can be extended to the orientation error between the estimated and GT positions as well.

$$\theta_{\text{error}} = \left|\theta_{\text{filter}} - \theta_{\text{gt}}\right|. \tag{13}$$

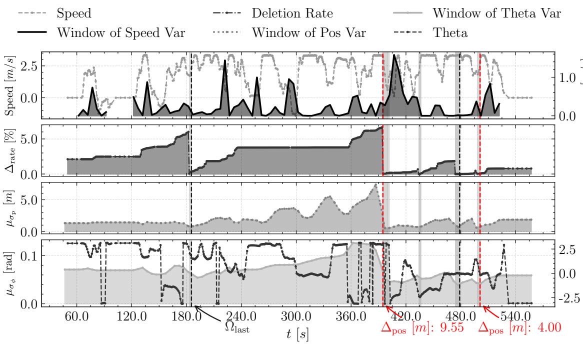
Mahalanobis Distance for State-Estimate Inclusion

The Mahalanobis distance provides an error metric that is more related to the filter by considering the entire state estimate distribution. Unlike the Euclidean distance, which only measures the direct spatial offset between the estimated and GT positions, the Mahalanobis distance takes into account the spread and shape of the state distribution. This metric measures how many standard deviations away the GT position is from the mean of the state-estimate distribution, thereby offering a deeper insight into the system's accuracy and is calculated as follows:

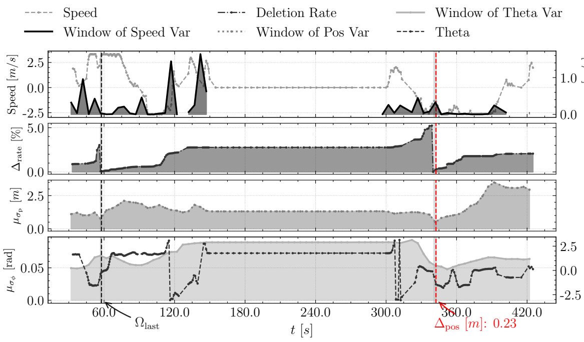
$$d_{\text{Mahalanobis}} = \sqrt{(x_{\text{GT}} - \mu_x)^T S^{-1} (x_{\text{GT}} - \mu_x)}. \tag{14}$$

Where μ_x is the mean vector of the state positions and S is the positive-definite covariance matrix of the state estimate positions. By focusing on the state-estimates, the Mahalanobis distance accounts for the distribution and variance within a filter, providing a more robust metric for error detection in comparison to the Euclidean distance. This approach is particularly useful for identifying outliers and assessing the overall reliability of the estimated position relative to the ground truth when the state-estimate-based position is used during unsupported estimation.

4. Experimental Results



(a) Sensor data 1



(b) Sensor data 2

Fig. 3: (a) and (b) depict sensor data for two distinct runs. The shaded regions represent UWB measurements, while the black vertical lines signify the commencement of unsupported estimation. The red lines denote the position difference Δ_{pos}, instances where relative positioning is rectified using absolute measurements. The window size μ is set to 5 s.

In the given scenario, sensor data is passively collected from multiple vehicles operating on-site during the production process, as detailed in Section 3. The dataset, representing one month of data collection from multiple vehicles at a single site, comprises a total of 4,129,602 position events. Each position event encapsulates a timestamp, current speed, pose information (x, y, θ), and the standard deviation (σ) for each value. Additionally, the dataset includes the vehicle's speed and the current state-estimate depletion ratio.

The illustration in Fig. 3 presents two selected traces that underscore the data. A more detailed analysis of the data reveals several insights. Firstly, the magnitude of the discrepancy is not readily discernible by merely observing the quantity of state-estimate wipe, standard deviation of position, and theta. Secondly, alterations in theta and speed influence the standard deviation of position. Finally, based on the map settings at the production site, in narrow corridors, the jump size is constrained to the width of the corridor. Consequently, errors may occur even when jumps are relatively small, underscoring the necessity of a combination of error detection methods. Utilizing the dataset provided, the benchmark pipeline is used to manually processes the data to ascertain the thresholds for the event detection methods, as described in Section 2. Moreover, datasets associated with localization engine issues are employed in the simulation segment of the benchmark pipeline. Fig. 4 shows the monthly event count of a real production site, having multiple manually driven vehicle in operation. Because this is a work in progress, thresholds are manually picked and not validated in combination with the ground truth data. The presented data shows an example of one RTLS system in a real-world scenario. During the dataset generation multiple software changes were made to the localization engine settings, resulting in a variation of event counts. The link between event count and adaptions in the localization engine settings is visible in the data. To have this evaluated in a controlled test scenario, we plan to test different loaclization settings during specific time frames, resulting in a variation of event counts. This will be validated with a ground truth system to ensure the thresholds are set correctly.

5. Conclusion & Discussion

This paper presented a preliminary architecture for in-production benchmarking of indoor localization systems, focusing on real-time error detection through event-driven data collection. While the architecture shows promise in identifying system misbehaviors, it currently relies on manually calibrated thresholds, which have not been validated against ground truth data. As such, the detection accuracy is not yet reliable enough for fully autonomous operation.

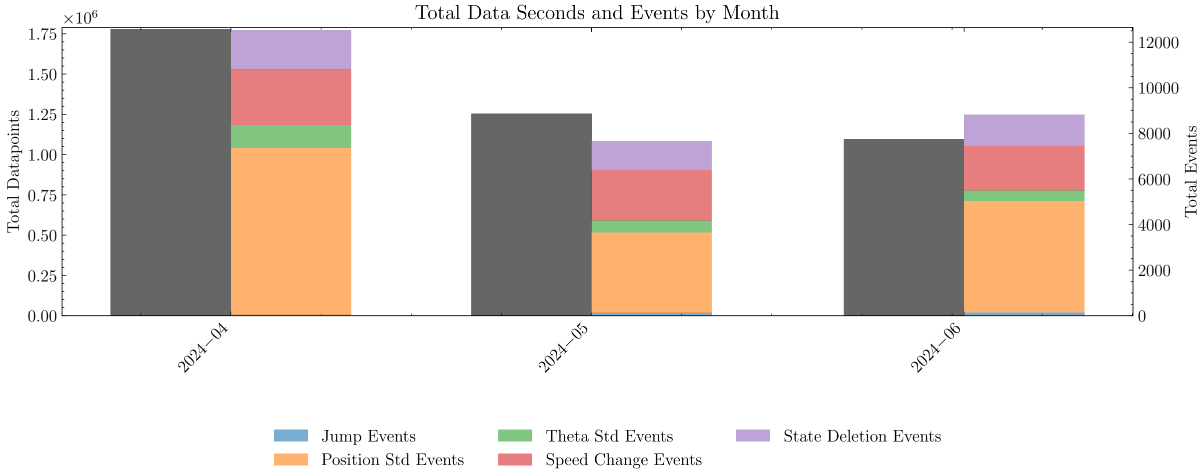


Fig. 4: Total amount of events by category and datapoints for the given production site by month. The categories are calculated over a slinding window of 5 seconds. The thresholds are manually set: $45°$ for the theta standard deviation, $5\mathrm{m}$ for the position standard deviation, $2.77\mathrm{m\,s^{-1}}$ for the speed change, $10\mathrm{m}$ for the jump event, and 60% for the state deletion event with the plausibility check.

The next steps involve refining these thresholds using ground truth data to enhance the system's real-time capabilities. Additionally, validation against established benchmarking systems is essential to assess the overall effectiveness of the system. We invite the research community to explore questions such as: How can this system be validated and compared against existing methods in diverse operational environments, especially when running in operational scenarios?

These discussions are crucial as we aim to create a more robust, scalable solution for real-world industrial applications. By addressing these challenges collaboratively, we can significantly advance the reliability and performance of indoor localization systems.

References

[1] H. Alghodhaifi, S. Lakshmanan, Autonomous Vehicle Evaluation: A Comprehensive Survey on Modeling and Simulation Approaches, IEEE Access 9 (2021) 151531–151566. doi:10.1109/ACCESS.2021.3125620, conference Name: IEEE Access.

[2] C. Gómez-Huélamo, J. Del Egido, L. M. Bergasa, R. Barea, E. López-Guillén, F. Arango, J. Araluce, J. López, Train here, drive there: ROS based end-to-end autonomous-driving pipeline validation in CARLA simulator using the NHTSA typology, Multimed Tools Appl 81 (2022) 4213–4240. doi:10.1007/s11042-021-11681-7.

[3] C. Team, CARLA, 2024. URL: http://carla.org//.

[4] A. Thakur, A Conceptual Market Analysis of Automated Vehicles for Logistics in Future, Journal of Supply Chain Management Systems 11 (2022) 24.

[5] B. Van Herbruggen, J. V.-V. Gerwen, S. Luchie, Y. Durodié, B. Vanderborght, M. Aernouts, A. Munteanu, J. Fontaine, E. D. Poorter, Selecting and Combining UWB Localization Algorithms: Insights and Recommendations From a Multi-Metric Benchmark, IEEE Access 12 (2024) 16881–16901. doi:10.1109/ACCESS.2024.3358274.

[6] D. Lymberopoulos, J. Liu, The Microsoft Indoor Localization Competition: Experiences and Lessons Learned, IEEE Signal Process. Mag. 34 (2017) 125–140. doi:10.1109/MSP.2017.2713817.

[7] S. Ulrich, T. Luong, C. Moldovan, J. Tiemann, A. Lewandowski, C. Röhrig, System Architecture for Digital Twin Based Collision Avoidance Through Private 5G Networks, in: 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), IEEE, Dortmund, Germany, 2023, pp. 199–204. doi:10.1109/IDAACS58523.2023.10348730.

[8] D. Grzechca, A. Ziębiński, K. Paszek, K. Hanzel, A. Giel, M. Czerny, A. Becker, How Accurate Can UWB and Dead Reckoning Positioning Systems Be? Comparison to SLAM Using the RPLidar System, Sensors (Basel) 20 (2020) 3761. doi:10.3390/s20133761.

[9] F. Potorti, P. Barsocchi, M. Girolami, J. Torres-Sospedra, R. Montoliu, Evaluating indoor localization solutions in large environments through competitive benchmarking: The EvAAL-ETRI competition, in: 2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, Banff, AB, Canada, 2015, pp. 1–10. doi:10.1109/IPIN.2015.7346970.

[10] B. Vedder, B. J. Svensson, J. Vinter, M. Jonsson, Automated Testing of Ultrawideband Positioning for Autonomous Driving, Journal of Robotics 2020 (2020) 1–15. doi:10.1155/2020/9345360.

[11] A. Schjørring, A. L. Cretu-Sircu, I. Rodriguez, P. Cederholm, G. Berardinelli, P. Mogensen, Performance Evaluation of a UWB Positioning System Applied to Static and Mobile Use Cases in Industrial Scenarios, Electronics 11 (2022) 3294. doi:10.3390/electronics11203294.

[12] A. Amini, T.-H. Wang, I. Gilitschenski, W. Schwarting, Z. Liu, S. Han, S. Karaman, D. Rus, VISTA 2.0: An Open, Data-driven Simulator for Multimodal Sensing and Policy Learning for Autonomous Vehicles, in: 2022 International Conference on Robotics and Automation (ICRA), IEEE, Philadelphia, PA, USA, 2022, pp. 2419–2426. doi:10.1109/ICRA46639.2022.9812276.

[13] A. Ortega, N. Hochgeschwender, T. Berger, Testing Service Robots in the Field: An Experience Report, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2022, pp. 165–172. doi:10.1109/IROS47612.2022.9981789.

[14] R. E. Kalman, A New Approach to Linear Filtering and Prediction Problems, Journal of Basic Engineering 82 (1960) 35–45. doi:10.1115/1.3662552.

[15] J. S. Liu, R. Chen, Sequential Monte Carlo Methods for Dynamic Systems, Journal of the American Statistical Association 93 (1998) 1032–1044. doi:10.1080/01621459.1998.10473765.

[16] F. Zafari, A. Gkelias, K. K. Leung, A Survey of Indoor Localization Systems and Technologies, IEEE Communications Surveys & Tutorials 21 (2019) 2568–2599. doi:10.1109/COMST.2019.2911558, conference Name: IEEE Communications Surveys & Tutorials.

[17] R. F. Brena, J. P. García-Vázquez, C. E. Galván-Tejada, D. Muñoz-Rodriguez, C. Vargas-Rosales, J. Fangmeyer, Evolution of Indoor Positioning Technologies: A Survey, Journal of Sensors 2017 (2017) 1–21. doi:10.1155/2017/2630413.

[18] L. Mainetti, L. Patrono, I. Sergi, A survey on indoor positioning systems, in: 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM), IEEE, Split, Croatia, 2014, pp. 111–120. doi:10.1109/SOFTCOM.2014.7039067.

[19] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping: part I, IEEE Robotics & Automation Magazine 13 (2006) 99–110. doi:10.1109/MRA.2006.1638022, conference Name: IEEE Robotics & Automation Magazine.

[20] K. Yousif, A. Bab-Hadiashar, R. Hoseinnezhad, An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics, Intell Ind Syst 1 (2015) 289–311. doi:10.1007/s40903-015-0032-7.

[21] W. Zhang, X. Li, D. Wei, X. Ji, H. Yuan, A foot-mounted PDR system based on IMU/EKF+HMM+ZUPT+ZARU+HDR+compass algorithm, in: 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, Sapporo, 2017, pp. 1–5. doi:10.1109/IPIN.2017.8115916.

[22] A. R. Jimenez Ruiz, F. Seco Granja, J. Carlos Prieto Honorato, J. I. Guevara Rosas, Pedestrian indoor navigation by aiding a foot-mounted IMU with RFID Signal Strength measurements, in: 2010 International Conference on Indoor Positioning and Indoor Navigation, IEEE, Zurich, Switzerland, 2010, pp. 1–7. doi:10.1109/IPIN.2010.5646885.