ABeL: A Customizable Open-Source Framework for Evaluating 3D Terrestrial Positioning Algorithms

Simon Huh[1,*], Tristan Itschner[1] and Niclas Zeller[1]

[1]Karlsruhe University of Applied Sciences, Moltkestraße 30, 76131 Karlsruhe, Federal Republic of Germany

Abstract

The increase of readily available compute power of portable devices allows them to use more complex algorithms for positioning. As these algorithms are based on the propagation of a signal through the environment, an evaluation tool that allows for an evaluation of positioning algorithms based on the scenario is needed. With ABeL (*Terrestrial Positioning **A**lgorithm **Be**nchmarking **L**ibrary*), an open-source Python framework for an end-to-end evaluation and benchmarking of indoor and outdoor positioning algorithms, we created a framework satisfying this need. The approach of using Python as a programming language and a modularized structure allows for maximum flexibility in regard to algorithm implementation and evaluation while still maintaining a relatively low barrier of entry. As an initial benchmark, we create a baseline by evaluating two positioning algorithms, thereby demonstrating that the framework can be used for simulating multipath affected environments. The framework is available as OSS (*Open-Source Software*) at https://github.com/RobotVisionHKA/ABeL.git.

Keywords

Indoor Positioning, Simulation Framework, Positioning Benchmark, Nvidia Sionna

1. Introduction

In many terrestrial indoor as well as urban outdoor positioning scenarios, a direct LoS (*Line-of-Sight*) connection between transmitter and receiver cannot be guaranteed. Instead, electromagnetic waves are often reflected at least once to reach the receiver, creating an NLoS (*Non-Line-of-Sight*) environment where the ToA (*Time of Arrival*) is delayed. Furthermore, DMCs (*Diffuse Multipath Components*) are introduced through the interaction of electromagnetic waves with diffuse or rough surfaces as well as edges. As both NLoS and DMC have been shown to have a major impact on the accuracy of positioning algorithms and are highly dependent on the propagation environment, a benchmarking framework is needed allowing for a holistic and comparable testing of positioning and multipath mitigation strategies [1], [2].

To the best of our knowledge, as of today, there exists no such E2E (*End-to-End*) solution, neither commercially nor as OSS (*Open-Source Software*), for terrestrial positioning that is providing a 1. scenario definition, 2. signal generation and data preparation and 3. algorithm evaluation metric in a standardized manner. Instead, datasets, either open-source or self-generated, or simplified channel models are used by researchers in conjunction with custom benchmarking tools. The first drawback arising from this approach is a limited comparability between multiple positioning algorithms, as the data source used for benchmarking the algorithms and the data preprocessing have a major impact on the reported accuracy of the algorithm [3], [4], [5]. Besides, there is a need for a standardized evaluation, either visually or numerically, which cannot be provided by custom benchmarking solutions, leading to only vaguely comparable performance results [6]. Another point to mention is the time needed to create an environment, where positioning algorithms can be realistically tested and benchmarked. At the moment researchers need to either create their own custom environment or augment existing ones to fit their needs, while the primary focus still lies on the research of terrestrial positioning algorithms.

IPIN-WCAL 2025: Workshop for Computing & Advanced Localization at the Fifteenth International Conference on Indoor Positioning and Indoor Navigation, September 15–18, 2025, Tampere, Finland

*Corresponding authors and main contributors.

Simon.Huh@h-ka.de (S. Huh); Tristan.Itschner@h-ka.de (T. Itschner); Niclas.Zeller@h-ka.de (N. Zeller)

https://www.niclas-zeller.de (N. Zeller)

0009-0000-1222-6718 (S. Huh); 0009-0007-4641-0434 (T. Itschner); 0000-0001-7865-1944 (N. Zeller)

With ABeL we would like to overcome the aforementioned limitations and create an easy-to-use framework for rapid prototyping and benchmarking of terrestrial positioning algorithms. Despite having a low barrier of entry, the flexibility, adaptability and expandability of the framework are still maintained.

The remainder of this paper is structured as follows: in section 2 current solutions for simulating signal propagation are compared, section 3 explains the modularized structure of ABeL and how a scenario can be defined and customized, section 4 shows the capabilities of ABeL on the example of a ToA (*Time of Arrival*) and TDoA (*Time Difference of Arrival*) positioning algorithm, section 5 will give an outlook on work-in-progress and planned features and section 6 will summarize the current progress and capabilities of ABeL.

2. Related Work

The propagation of a signal can generally be simulated by either using stochastic or deterministic channel models. While the former is advantageous when temporal change is the main focus, deterministic RT (*Ray Tracing*) and RL (*Ray Launching*) channel models have been shown to be more accurate if the receiver position is (quasi-)static [7]. Furthermore, it was proved that 2-dimensional channel models perform worse than their 3D counterpart with regard to channel capacity, gain and delay spread [8], [9].

Commercially, there are multiple RT- or RL-based simulators available. *MathWorks Matlab* can be used in conjunction with the *Communication Toolbox* to evaluate the propagation effects based on longitude and latitude. The *Wireless InSite®* software from *remcom* instead follows a more holistic and easy-to-use approach, allowing the user to carry out exhaustive link- and system-level analyses of antennas as well as transmitter and receiver setups in a GUI (*Graphical User Interface*). Besides the aforementioned closed source solutions, there are also many OSS options available. The simulator *WiThRay* [10] implements its own custom ray tracing algorithm and is more inclined towards link-level simulation. Another solution is *Unity*-based simulator presented in [11]. While its unique approach allows simulating time-based scenarios, the drawbacks arise from the use of closed-source tools (*Unity*) and the amount of external dependencies needed for the simulation. While *Opal* [12] is not directly dependent on *Unity*, *Veneris* uses it to create dynamic scenarios. Another double-edged sword is *Opal's* use of pure C++ code: a pure C++ implementation will be faster than comparable Python code, while the ease-of-use is greatly reduced. With *Sionna* [13] a CUDA-based Python simulator is readily available. In comparison to the aforementioned solutions, *Sionna* is actively developed, has exhaustive documentation and is highly customizable. Furthermore, it offers system- and link-level simulation capabilities, which can be used in conjunction with the generated signal propagation data.

The aforementioned simulators have in common that the provided tools are limited to simulating and evaluating signal propagation data. Neither the open- nor closed-source simulators allow 1. to define and create scenarios as well as 2. to implement and benchmark positioning algorithms in a comparable environment. With ABeL we target to fill this gap.

3. Framework Architecture

For ABeL's structure, a modularized approach shown in figure 1 with fixed interfaces between modules was taken. This ensures that future changes can be implemented easily and allows the stages of the framework to be run independently. Having independent stages is especially important, as data generation is the main contributor to the total compute time. The framework itself is separated into three modules, which will be explained in the following chapters.

- *Scenario Definition and Parametrization*: Create a scenario with *Blender*, place transmitters as well as receivers and define the signal and simulation properties.
- *Signal Generation and Data Preprocessing*: Simulate the signal propagation for the scenario defined in the preceding module and prepare the data for saving and later use.

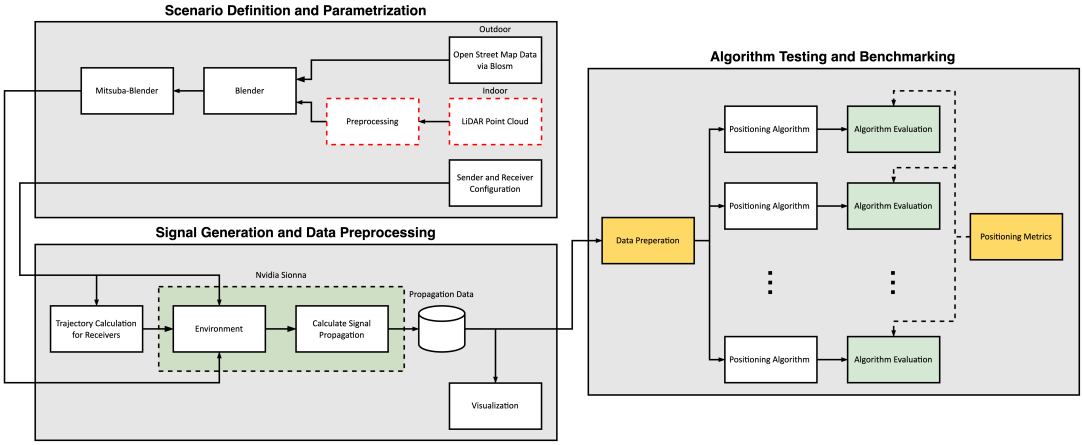


Figure 1: Diagram showing the data flow through ABeL's modules. Features, which will be implemented in the future, are marked by a dashed red border.

- *Algorithm Implementation and Benchmarking*: Use the data generated in the previous module and supply it via a fixed interface. Also provide standardized performance metrics for positioning algorithms, guaranteeing inter-comparability.

3.1. Scenario Definition and Parametrization

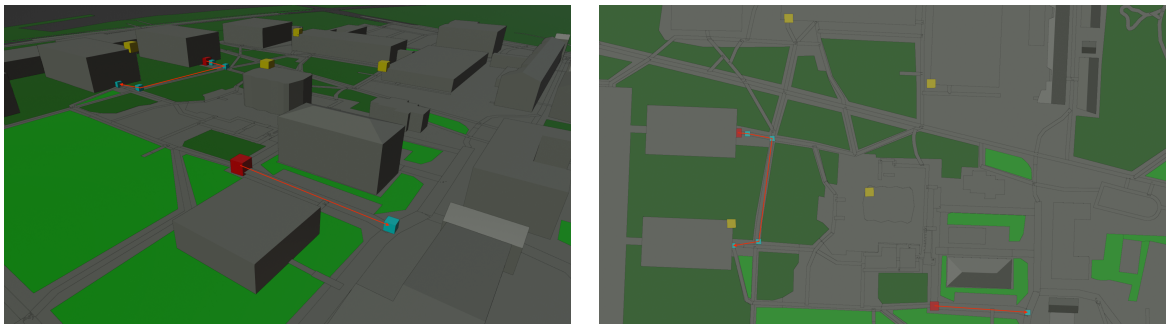
The definition of the scenario can generally be divided into the parts 1. scene creation and 2. signal and simulation parametrization. For the former, *Blender* [14], a 3D rendering OSS, is used to create a rendering of the environment in a multistage process. First, OSM (*Open Street Map*) data is used in conjunction with *Blosm* [15] to create a 3D rendering of the terrain. The objects of the scene, like buildings or streets, are then given a material identifier, which will be used for signal generation with Nvidia's Sionna framework; an aerial view of the scene after the OSM data import and the preprocessing is shown in figure 2. In the last step, the rendering is exported by Mitsuba-Blender [16]. The level of detail of the exported scenario is only limited by the available system resources used for generating the signal propagation describes in the following simulation step.

The second part of the scene definition consists of parametrizing the transmitters, receivers, material and simulation parameters. For transmitter and receiver, the orientation in space and the antenna pattern can be defined, while for receivers there is also an option to configure a speed-dependent trajectory. Furthermore, the electromagnetic properties of the materials need to be set for an accurate simulation of the signal propagation; see [17], [18] for an in-depth description of the propagation models used. Finally, the simulation parameters containing, among other things, the option to set the total amount of bounces or signal frequency, also need to be specified.

3.2. Signal Generation and Data Preprocessing

The generation of the signal propagation data can be generally divided into step 1. setting up the environment, 2. generating signal propagation data and 3. saving and visualizing the data. For the first step, we need to calculate the trajectories of the receivers, which are defined by anchor points containing position and orientation in 3D space. With the knowledge of the simulation step size and the receiver's velocity, we can determine the position and orientation of all receivers for every time step through linear interpolation. Following, a *Nvidia Sionna* scene will be set up with the defined signal and simulation properties, the environment created in Blender and the defined trajectory.

The generation of the signal propagation data is implemented through *Nvidia Sionna* [19], a Python toolbox for researching communication systems. In comparison to other available solutions, it is open-source, still actively developed, has mature ray tracing capabilities and also allows a link- and



(a) 3D aerial side view of the campus.

(b) Orthographic top down view of the campus.

Figure 2: Aerial view of the *Karlsruhe University of Applied Sciences'* campus, which was imported into *Blender* by using OSM data and *Blosm*. The objects of the scene were preprocessed and got assigned material identifiers. Furthermore, the transmitters are shown as yellow cubes, the receivers as red cubes and the anchor points for interpolating the trajectory, shown as red lines, as smaller blue cubes.

system-level channel simulation for the defined scenarios. While there are no plans to directly support the latter in ABeL, users may still use the generated signal propagation data in conjunction with *Sionna*; for this we refer to the documentation [20]. Instead, we use the differential ray tracer in conjunction with the scenario defined in the preceding module to get high resolution signal propagation data containing, among others, the complex amplitude of the signal, the azimuth φ and zenith θ angle of arrival and departure and the time of arrival. *Sionna* itself considers reflection, refraction, (edge) diffraction and scattering in respect to the material and simulation properties [13]. The latter include, but are not limited to, the carrier frequency, bandwith and polarisation of the signal as well as customizable receiver and transmitter antenna arrays.

Following, the data can either be preprocessed and saved for benchmarking or be used directly for visualization and further processing, e.g., for use with *Sionna's* link- and system-level channel simulation. Currently, a visualization of the CIR (*Channel Impulse Response*), the AoA (*Angle of Arrival*) and AoD (*Angle of Departure*) as well as the generation of coverage plots are supported; for the CIR and AOA / ADO plot, see also figure 4.

3.3. Algorithm Implementation and Benchmarking

The comparability between positioning algorithms can be established by using standardized interfaces, namely a joint data preparation and shared performance evaluation. Therefore, the measured performance is only dependent on the positioning algorithms. In code this is realized by the two classes "DataPreperation" and "LocalizationPrecisionMetrics", where the latter currently supports the automatic creation of plots showing the absolute error over time and as a boxplot; for examples, see section 4.

For the implementation of positioning algorithms, we recommend the use of the provided template, whose call structure is shown in figure 3. This is a further step towards standardization, as the compliance between the class and interfaces of the algorithm is guaranteed, while an easy implementation of new algorithms can be ensured. The latter can be traced to the decorator "iterateMethod", which allows iterating over a decorated method depending on an instance variable. The result is a dimensionality reduction, in other words, if we iterate over all simulation steps, we only need to implement an algorithm for a system of one or multiple transmitters and receivers. One may take this even a step further and use the decorator in conjunction with the "__algorithm" method to iterate over all receivers, hence creating a constellation of multiple transmitters and one receiver.

After creating and populating the positioning algorithm class, the performance metrics can easily be called by dot-notation, accessing the inherited evaluation metrics from "LocalizationPrecisionMetrics".

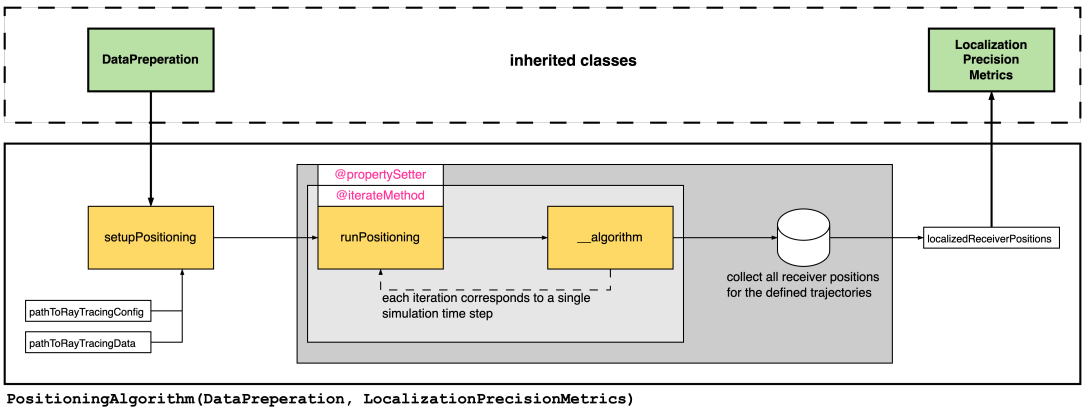


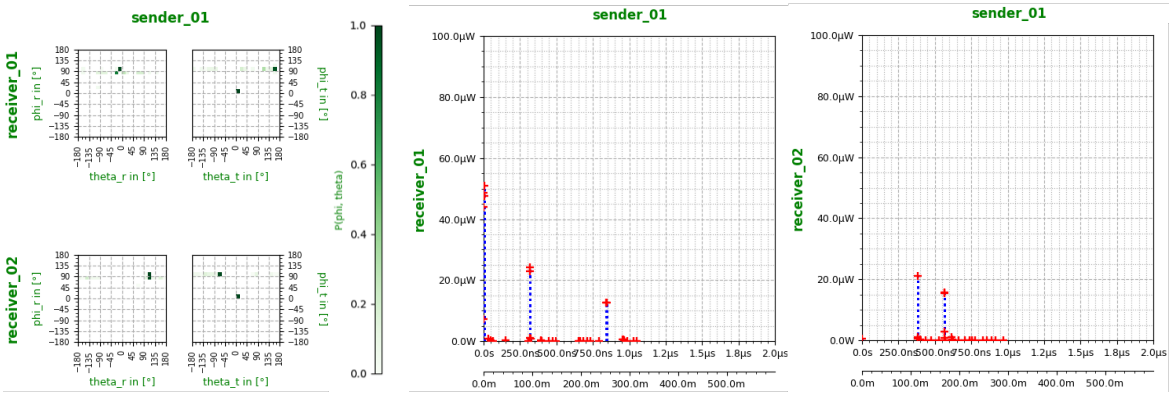
Figure 3: Visualization of the call structure of the "PositioningAlgorithm" class. Its methods are shown in yellow, the decorators in pink and inherited classes in green.

4. Reference Benchmark Implementation

As we would like to keep the barrier of entry low, we have chosen to show ABeL's benchmarking capabilities for a ToA [21] and TDoA [22] based positioning algorithm in a 5G environment with a carrier frequency of 5 GHz and bandwidth of 10 MHz. Both will be used in conjunction with a minimal ToF (*Time of Flight*) selector, picking the received signal with the shortest ToF, for a rudimentary mitigation of multipath effects. Furthermore, a minimal set of four transmitters and two receivers were selected for the scenario. This decision was made to create a minimal working benchmark in 3D space. The receivers themselves will follow trajectories with 1. direct LoS and 2. no or only partial LoS to all transmitters with a speed of 1 m/s, mimicking a human's normal walking speed [23]. A visualization of the setup and configuration is also shown in figure 2. Finally, the main simulation parameters were chosen as follows: a step size of 1 s, minimizing the total compute time while still ensuring a high positioning fidelity, the use of electromagnetic reflection and diffraction as well as scattering and a maximum amount of 20 bounces, guaranteeing a high resolution even in dense and complex environments.

ABeL supports evaluating 1. the signal propagation itself and 2. the performance of positioning algorithms. The former is solely dependent on the signals' propagation through the environment and thus needs to be carried out only once per data generation. While not directly contributing to the performance evaluation, the plots collected in 1. allow researchers to make sense of results of the positioning algorithms and choose an appropriate algorithm depending on the environment. As the propagation data changes over time, we decided to create an animation made by linking together the CIR (*Channel Impulse Response*) as well as AoA (*Angle of Arrival*) and AoD (*Angle of Departure*) plots created for a single simulation time step. Figure 4a visualizes the change of the AoA and AoD by plotting the relative frequency of azimuth φ and zenith θ angle. If high spreading occurs or multiple points of high density are shown, the transmitter-receiver connection must be NLoS and an angle-based positioning algorithm might be infeasible. With the CIRs shown in figure 4b, we can get an even clearer view of the situation: "receiver_02" must have moved into a (partial) NLoS environment, as the signal power is either very low, resulting from a loss of power due to multiple material interactions, or the signal is arriving delayed, a consequence of the additional distance traveled due to reflection. Furthermore, there is also an option to generate a coverage map of the scenario.

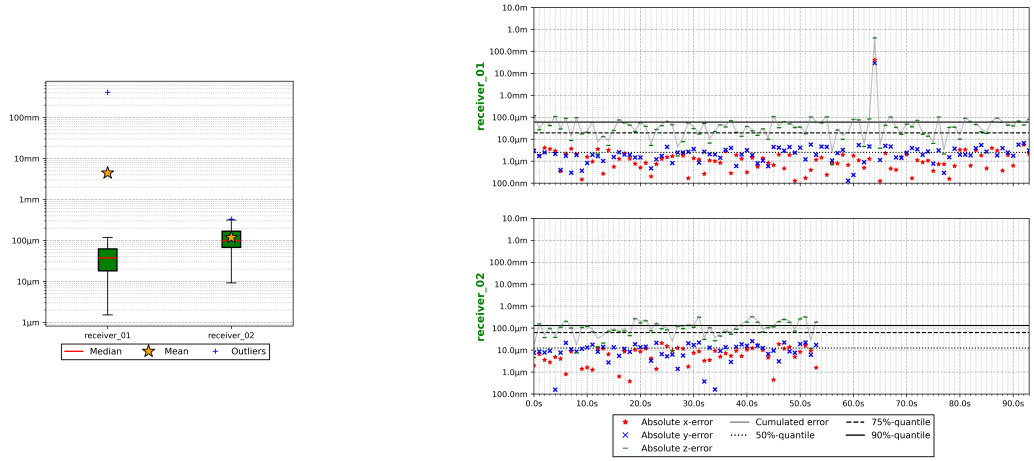
After the simulation of the signal propagation and the implementation of the algorithms as shown in figure 3, the error between actual and calculated receiver position $\vec{e} = \vec{p}_{\text{actual}} - \vec{p}_{\text{calculated}}$, with $\vec{p} \in \mathbb{R}^3$ describing the receiver position, can either be evaluated visually or tabulary. The former is shown in figure 5 and gives a statistical overview of the error in two differing ways. In figure 5b the $\mathcal{L}_2$-norm of the error $\vec{e}$ is shown as a boxplot with an inter-quantile range lying between the 25 %- and 75 %-quantile and whiskers of size 1.5 times the inter-quantile range. This allows for a quick assessment of the algorithm used, e.g., the plot shown lets us know that the positioning accuracy is



(a) AoA and AoD plots, showing azimuth φ and zenith θ angle.

(b) CIR plot, showing the signal power and delay.

Figure 4: Visualization of a transmitter-receiver-pair for a single frame of the animation.



(a) Boxplot showing the $\mathcal{L}_2$-norm of the errors between actual and localized receiver positions.

(b) Plot showing the change of the absolute errors of the spatial axes.

Figure 5: Visualization of the error between actual and localized receiver positions for the trilateration algorithm used for ToA positioning. The error can be evaluated either for all receiver positions at once (a) or every individual position (b).

acceptable, while "receiver_01" has a significant outlier. For a more in-depth view, we can consult figure 5b, visualizing the absolute error for each spatial axis and simulation time step. In the case of the ToA-based positioning algorithm, the z-axis can be determined as the main contributor to the mean error, indicating an imprecision of the model described in [21]. Furthermore, the outlier affects all spatial axes, pointing to a loss of an LoS connection between the receiver and one or multiple of the transmitters; a possible solution has been described in [24]. Finally, with a median positioning resolution of $\approx 40\,\mu\text{m} \propto 13\,\text{fs}$ for "receiver_01" and $\approx 100\,\mu\text{m} \propto 33\,\text{fs}$ for "receiver_02", the error introduced by ABeL is multiple orders of magnitude below the accuracy of any practical implementation of positioning algorithms [25], [26].

For a direct comparison between multiple positioning algorithms, the metrics shown in table 1 can be returned by ABeL. The shown column arrangement allows for either a general or per-axis performance analysis, guaranteeing that both an easy and fast as well as in-depth evaluation can be conducted. As for the metrics, the RMSE (*Root Mean Square Error*) measures the standard deviation of the residuals, i.e., the error $\vec{e}$. The mean in conjunction with the median allows us instead to estimate the scale of the average

Table 1
Statistical values measured for the ToA and TDoA positioning algorithm for all combined and individual spatial axes; the structure shown is also returned by the "LocalizationPrecisionMetrics" class. The upper group of rows are assigned to LoS "receiver_01", the lower to NLoS "receiver_02".

Metric [m]	ToA				TDoA			
	all axes	x-axis	y-axis	z-axis	all axes	x-axis	y-axis	z-axis
RMSE	+4.244e-02	+4.213e-03	+2.998e-03	+4.212e-02	+4.164e+00	+1.440e+00	+1.864e+00	+3.434e+00
MEAN	-1.697e-03	-4.345e-04	-3.095e-04	-4.348e-03	-5.853e-01	-3.781e-01	-4.932e-01	-8.847e-01
MEDIAN	-3.069e-07	+9.495e-08	-3.497e-07	-7.768e-06	-1.375e-06	-4.934e-07	-1.046e-06	-8.454e-06
ABS-MEA.	+1.712e-03	+4.359e-04	+3.115e-04	+4.387e-03	+5.853e-01	+3.781e-01	+4.932e-01	+8.847e-01
ABS-MED.	+2.505e-06	+1.096e-06	+2.022e-06	+3.728e-05	+7.912e-06	+2.907e-06	+3.957e-06	+2.596e-05
RMSE	+3.226e-01	+3.202e-02	+2.279e-02	+3.202e-01	+9.873e-02	+3.164e-02	+5.998e-02	+7.175e-02
MEAN	-1.292e-02	-3.302e-03	-2.358e-03	-3.310e-02	-3.428e-03	+3.292e-03	-6.210e-03	-7.367e-03
MEDIAN	-4.612e-06	+1.584e-06	-1.385e-05	-1.101e-04	+1.272e-05	+3.464e-05	-3.558e-05	+6.362e-05
ABS-MEA.	+1.295e-02	+3.309e-03	+2.365e-03	+3.318e-02	+5.666e-03	+3.309e-03	+6.229e-03	+7.461e-03
ABS-MED.	+1.720e-05	+1.584e-06	+1.720e-05	+1.886e-04	+5.796e-05	+5.176e-05	+5.463e-05	+8.165e-05

error without outliers as well as measure the influence of the outliers themselves. Furthermore, we also calculate the absolute value of both metrics, as the error is a signed value. While the sign introduces directionality, it also distorts the mean and median, in the sense that the positioning error is moved closer to zero, thus causing an additional error; e.g., see the x-axis of "receiver_01" for the ToA algorithm. Finally, the values shown in table 1 for positioning algorithms shall also serve as a reference for future works and while we only considered the positioning precision, ABeL can easily be extended by custom performance metrics, e.g., for measuring the availability or efficiency of a positioning algorithm.

5. Discussion and Outlook

While ABeL can already be used for benchmarking positioning algorithms, improvements can still be carried out in regard to the integration of *Blender*. Currently, the creation of the environment is done in *Blender*, while the transmitter and receiver properties are defined independently by hand. This results in 1. unnecessary labor and 2. may also lead to transfer errors of the properties from *Blender* to the config file, which could introduce positioning or simulation errors.

Another important area needing further improvement, is the dependence on material and electromagnetic properties for high-frequency systems defined in [27]. Especially for dense and complex environments, i.e., when applying indoor positioning, the provided materials may not be sufficient to accurately model the scenario. A possible solution is discussed by the creators of *Sionna* themselves: calibrating the material properties in such a way that the simulated signal matches the real signal closely [13], [28]. Closing the topic on *Sionna's* implementation into ABeL, we strive to provide further tools and metrics for evaluating the data generated by our framework.

In addition, modeling indoor environments is 1. time-consuming and 2. imprecise when done manually, causing a distortion between the simulated and actual signal propagation. We want to remedy this error by creating a *Blender* interface, which allows us to import point clouds captured by LiDAR sensors, e.g., generated by automatic mapping algorithms like [29]. Thus, we will be able to recreate the environment by a point cloud to mesh conversion, allowing an easy and accurate representation of the real-world environment [30]. Based on this approach and the aforementioned material calibration, we want to validate ABeL with actual terrestrial data.

6. Conclusion

In this paper we have introduced ABeL, a novel benchmarking framework for 3D terrestrial positioning algorithms. We discussed the framework's structure as well as modules in necessary detail such that the

readers can effectively use ABeL and its features as well as being able to create their own scenarios and evaluation metrics. Furthermore, we have explained the latter by example of a ToA- and TDoA-based positioning algorithm, showcasing how ABeL can help researchers in developing positioning algorithms and create a baseline for future works.

With ABeL, we hope to support the ongoing research for accurate indoor positioning. Furthermore, we hope that the provided framework will guarantee reproductivity and comparability for the evaluation of (indoor) positioning algorithms.

Acknowledgments

This work was funded by the German *Federal Ministry of the Interior* represented by the *Federal Agency for Public Safety Digital Radio* (BDBOS) in the *KoPa_45* program.

We want to thank Janis Bernauer for supporting the technical realization of this paper as part of his work as student assistant.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] D. Weng, Z. Hou, Y. Meng, M. Cai, Y. Chan, Characterization and mitigation of urban GNSS multipath effects on smartphones, Measurement 223 (2023) 113766. doi:10.1016/j.measurement.2023.113766.

[2] L. Xie, Y. Zhang, C. Zhao, C. Zhang, M. Zhou, X. Yang, Positioning Under Multipath Environments in Wireless Network: Survey, Design, and Opportunities, IEEE Network 38 (2024) 476–484. doi:10.1109/MNET.2024.3435087.

[3] J. Medbo, I. Siomina, A. Kangas, J. Furuskog, Propagation channel impact on LTE positioning accuracy: A study based on real measurements of observed time difference of arrival, in: 2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications, IEEE, Toyko, Japan, 2009, pp. 2213–2217. doi:10.1109/PIMRC.2009.5450144.

[4] I. Sharp, Kegen Yu, T. Sathyan, Positional Accuracy Measurement and Error Modeling for Mobile Tracking, IEEE Transactions on Mobile Computing 11 (2012) 1021–1032. doi:10.1109/TMC.2011.119.

[5] I. Lapin, G. Seco-Granados, J. Samson, J. A. Garcia-Molina, Base Station Clock Offset of Cellular Measurements and Its Impact on Positioning, IEEE Transactions on Vehicular Technology 73 (2024) 15553–15566. doi:10.1109/TVT.2024.3409088.

[6] H. Liu, H. Darabi, P. Banerjee, J. Liu, Survey of Wireless Indoor Positioning Techniques and Systems, IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews) 37 (2007) 1067–1080. doi:10.1109/TSMCC.2007.905750.

[7] Stochastic versus Ray Tracing Wireless Channel Modeling for 5G and V2X Applications: Opportunities and Challenges, IntechOpen, 2022. doi:10.5772/intechopen.101625.

[8] N.-r. Jeon, C.-h. Lee, N.-g. Kang, S.-c. Kim, Performance of Channel Prediction Using 3D Ray-tracing Scheme Compared to Conventional 2D Scheme, in: 2006 Asia-Pacific Conference on Communications, IEEE, Busan, 2006, pp. 1–6. doi:10.1109/apcc.2006.255882.

[9] R. Almesaeed, A. S. Ameen, A. Doufexi, N. Dahnoun, A. R. Nix, A comparison study of 2D and 3D ITU channel model, in: 2013 IFIP Wireless Days (WD), IEEE, Valencia, Spain, 2013. doi:10.1109/wd.2013.6686494.

[10] H. Choi, J. Oh, J. Chung, G. C. Alexandropoulos, J. Choi, WiThRay: A Versatile Ray-Tracing Simulator for Smart Wireless Environments, IEEE Access 11 (2023) 56822–56845. doi:10.1109/access.2023.3283610.

[11] W. Tärneberg, A. Fedorov, G. Callebaut, L. V. der Perre, E. Fitzgerald, Towards Practical Cell-Free 6G Network Deployments: An Open-Source End-to-End Ray Tracing Simulator, 2023. doi:10.48550/arXiv.2401.08624. arXiv:2401.08624.

[12] E. Egea-Lopez, J. M. Molina-Garcia-Pardo, M. Lienard, P. Degauque, Opal: An open source ray-tracing propagation simulator for electromagnetic characterization, PLOS ONE 16 (2021) e0260060. doi:10.1371/journal.pone.0260060.

[13] J. Hoydis, F. A. Aoudia, S. Cammerer, M. Nimier-David, N. Binder, G. Marcus, A. Keller, Sionna RT: Differentiable Ray Tracing for Radio Propagation Modeling, in: 2023 IEEE Globecom Workshops (GC Wkshps), IEEE, Kuala Lumpur, Malaysia, 2023, pp. 317–321. doi:10.1109/GCWkshps58843.2023.10465179.

[14] Blender, Blender Foundation, 2025. URL: https://www.blender.org/.

[15] Blosm, prochitecture, 2025. URL: https://github.com/vvoovv/blosm.

[16] Mitsuba-Blender, Realistic Graphics Lab, 2025. URL: https://github.com/mitsuba-renderer/mitsuba-blender.

[17] V. Degli-Esposti, F. Fuschini, E. M. Vitucci, G. Falciasecca, Measurement and Modelling of Scattering From Buildings, IEEE Transactions on Antennas and Propagation 55 (2007) 143–153. doi:10.1109/TAP.2006.888422.

[18] V. Degli-Esposti, V.-M. Kolmonen, E. M. Vitucci, P. Vainikainen, Analysis and Modeling on co- and Cross-Polarized Urban Radio Propagation for Dual-Polarized MIMO Wireless Systems, IEEE Transactions on Antennas and Propagation 59 (2011) 4247–4256. doi:10.1109/TAP.2011.2164226.

[19] Nvidia Sionna, Nvidia Corporation, 2025. URL: https://github.com/NVlabs/sionna.

[20] Sionna — Sionna 1.1.0 documentation, 2025. URL: https://nvlabs.github.io/sionna/.

[21] W. Xiong, J. Bordoy, A. Gabbrielli, G. Fischer, D. J. Schott, F. Hoflinger, J. Wendeberg, C. Schindelhauer, S. J. Rupitsch, Two Efficient and Easy-to-Use NLOS Mitigation Solutions to Indoor 3-D AOA-Based Localization, in: 2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, Lloret de Mar, Spain, 2021, pp. 1–8. doi:10.1109/IPIN51156.2021.9662529.

[22] N. K. Inamdar, Time Difference of Arrival Source Localization: Exact Linear Solutions for the General 3D Problem, 2025. doi:10.48550/arXiv.2501.01076. arXiv:2501.01076.

[23] R. W. Bohannon, A. Williams Andrews, Normal walking speed: A descriptive meta-analysis, Physiotherapy 97 (2011) 182–189. doi:10.1016/j.physio.2010.12.004.

[24] E. Navon, B. Bobrovsky, An efficient outlier rejection technique for Kalman filters, Signal Processing 188 (2021) 108164. doi:10.1016/j.sigpro.2021.108164.

[25] Z. Abu-Shaban, X. Zhou, T. Abhayapala, G. Seco-Granados, H. Wymeersch, Error Bounds for Uplink and Downlink 3D Localization in 5G Millimeter Wave Systems, IEEE Transactions on Wireless Communications 17 (2018) 4939–4954. doi:10.1109/TWC.2018.2832134.

[26] A. Shahmansoori, G. E. Garcia, G. Destino, G. Seco-Granados, H. Wymeersch, Position and Orientation Estimation Through Millimeter-Wave MIMO in 5G Systems, IEEE Transactions on Wireless Communications 17 (2018) 1822–1835. doi:10.1109/TWC.2017.2785788.

[27] Recommendation ITU-R P.2040-3 (08/2023) - Effects of building materials and structures on radiowave propagation above about 100 MHz (2023).

[28] J. Hoydis, F. A. Aoudia, S. Cammerer, F. Euchner, M. Nimier-David, S. T. Brink, A. Keller, Learning Radio Environments by Differentiable Ray Tracing, IEEE Transactions on Machine Learning in Communications and Networking 2 (2024) 1527–1539. doi:10.1109/TMLCN.2024.3474639.

[29] T. Guadagnino, B. Mersch, S. Gupta, I. Vizzo, G. Grisetti, C. Stachniss, KISS-SLAM: A Simple, Robust, and Accurate 3D LiDAR SLAM System With Enhanced Generalization Capabilities, 2025. doi:10.48550/arXiv.2503.12660. arXiv:2503.12660.

[30] H.-W. Lin, C.-L. Tai, G.-J. Wang, A mesh reconstruction algorithm driven by an intrinsic property of a point cloud, Computer-Aided Design 36 (2004) 1–9. doi:10.1016/S0010-4485(03)00064-2.