L5IN+: From an Analytical Platform to Optimization of Deep Inertial Odometry

Hossein Shoushtari [1], Firas Kassawat [2], Dorian Harder [1], Korvin Venzke [1], Jörg Müller-Lietzkow [3], Harald Sternberg [1]

[1] *HafenCity University, Geodesy and Geoinformatics, Hamburg, Germany*
[2] *Bonn University, Computer Science Institute, Bonn, Germany*
[3] *HafenCity University, Economy and Digitization, Hamburg, Germany*

Abstract

Fifth generation of mobile communications (5G) and Deep Neural Networks (DNN) are two important technologies, which will enable new functions in the field of indoor positioning. This could be seen as the second major development after the innovation of smartphones, as a GNSS/INS alternative for indoor, location based applications. Optimization methods which work as a corrector, and as the uncertainty assessment for real life applications, guided us through the next level of challenges. In this paper, we have opened a novel interpretation of a deep network for inertial odometry which is robust to noisy labelled data that was detected from a 5G network. We also designed and developed analytical platform, which is considered a data collector and cellular positioning simulation. This platform was used to provide the input for the learning and optimization algorithms. The simulation website is implemented and available online under *simulation2evaluation.herokuapp.com* for researchers to generate ground truth trajectories and simulated cellular measurements with assigned quality and exact error values. We have proposed two approaches: (1) deep inertial odometry based on predicting velocity vector elements or relative positions and (2) Kalman Filtering to use, combine and test the absolute positions with the relative ones from the first approach. We finally provide numerical results of our experiments and a discussion of the effectiveness of our approaches.

Keywords

Indoor Localization, 5G Simulation, 5G Correction, Deep Neural Networks, Kalman Filter, Smartphone.

1. Introduction

Inertial Odometry has been a well-known field for the Geodesy and Geoinformatics community, starting with methods to develop a strapdown [1] inertial navigation system (INS) to determine the sensors' position. Combination of GNSS and IMU is still one of the main localization methods, often based on an extended Kalman Filter (KF). This localization core plays a significant role in visual odometry methods in outdoor scenarios [2], [3] for many applications such as autonomous driving, smart vacuum cleaners, UAV, and other robots. However, absence of GNSS or a comparable alternative in indoor areas has been a challenge for indoor applications.

Existing localization methods typically rely on Wi-Fi, Bluetooth, visual sensors such as LiDAR or cameras, but they are usually costly due to the installation requirements for an accurate localization and/or power-hungry [4]. IMUs could be a solution to the above problems as they are energy efficient and environmentally independent. However, these methods require correction after a while, to have a robust performance for a longer time [5]. Combination of two state-of-the-art positioning techniques

IPIN 2022 WiP Proceedings, September 5 – 8, 2022, Beijing, China
EMAIL: hossein.shoushtari@hcu-hamburg.de (H.Shoushtari)

can address the entire challenge, namely fifth generation (5G) positioning and deep learning based IMU localization [6].

On one hand, radio infrastructure can play a vital role in autonomous indoor navigation, thanks to the innovative capabilities of 5G wireless signals. The latest wireless network technology, 5G, also known as New Radio (NR), offers faster data rates, lower latency, higher capacity, lower transmitting power, network slicing and massive connectivity compared to previous generations [7]. Moreover, the related 3rd Generation Partnership Project (3GPP) releases (i.e., release 16 and 17 [8], [9]), are supported by 5G chips embedded in recent smartphones [10]. Considering the 5G as an alternative solution for GNSS in indoor areas, a 5G-based corrector can be used to correct the trajectory once an accurate measurement is available.

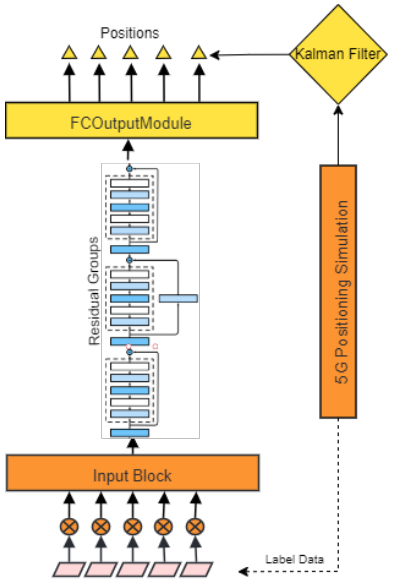

Figure 1: Overview of the proposed method.

On the other hand, recent deep learning approaches such as IONet [11], RoNIN [12], IDOL [4] and CTIN [13] have demonstrated the capability of neural networks to address the problem of accumulating sensor drift and different sensor placements, through the use of supervised learning, to directly estimate the velocity vector when estimating the user pose using an IMU. With a closer look on the RoNIN model for instance, it has been seen that they do not directly use only IMU values but also pre-calculated rotation vectors from the Android position sensor [14]. Moreover, the model still seems not to be robust for unknown users or sensors, presenting perhaps its high dependency on the still limited training data. Finally, a flexible correction for long trajectories seems to be the next step, following these works.

The correction can mathematically be modeled as an optimization problem. Applying the INS approaches using a KF directly on the smartphone sensors values has been tested in [15]. The results do not seem promising because of the poor performances of the sensors and the fix smartphone placement issue. Authors in [16] tried to combine the absolute position from fingerprinting with the relative ones, which are calculated based on inertial sensors, using a weighted least squares approach. Their method needs the same frequency of the absolute position and the relative ones, which would be energy hungry. 5G and odometry combination can estimate a corrected position and compare the performances. A non-linear state estimation by a CNN-based deep learning model for inferring the momentary speed using an extended KF has shown the feasibility of such optimization model [17]. In general, it has been shown that any odometry outputs in formats such as delta x and delta y as well as stride length and stride direction for human steps, and the 5G coordinates with an optimum frequency, in a real time manner is possible [18].

In response to the observations and concerns raised above, a novel interpretation of Deep Neural Networks (DNN) for inertial odometry which is robust to noisy labelled data, for example from a 5G network has been proposed (see Fig. 1). We have developed a research mode to collect more training data for our future 5G network, and in the absence of the 5G network, we have designed a web application to simulate 5G position information. Practically, we extended the ideas of neural networks and KF, to provide a robust combined solution. The major contribution of this paper is summarized as follows:

- Development of a novel combination of DNN and KF for the relative and absolute positioning.
- Design and development of an analytical platform for data collection.
- Extending our cellular positioning simulation and evaluation web application.

The rest of this paper is organized as follows. Section 2 gives background about the inertial odometry. Section 3 reviews the simulation web application as well as the research platform. Our developed models including the DNN and the KF are introduced in Section 4. Evaluation, and later conclusion and outlook are presented in section 5 and section 6.

2. Background

2.1. Inertial Odometry

Technically, 3D angular velocity and 3D acceleration provided by IMUs are the basis measurements. These are subjected to bias and white noises, but our observation shows that calibration of these parameters cannot solve the problem, due to the non-stable and poor performance of such sensors. However, the dynamic acceleration (without gravity) and the angular rate in the navigation coordinate system is essential for each inertial odometry method. We have used the Madgwick [19] to calculate the dynamic acceleration and other sensor values in the navigation frame. The results can be seen in Figure 2. Quaternions are one of the easiest ways to estimate these values:

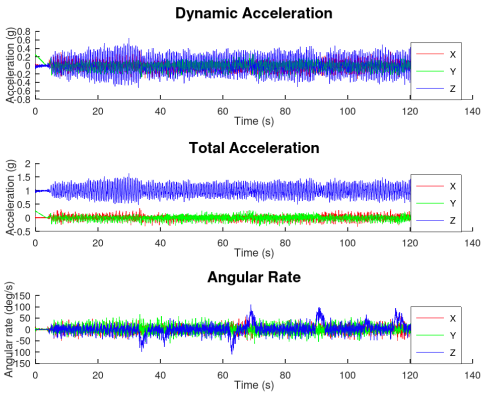

Figure 2: IMU values in the navigation coordinate system.

$$(0, x^p) = q \otimes (0, x^q) \otimes q^{-1} \tag{1}$$

Where x^p, x^q are quaternions with pure vectors i.e., the scaler value is set to be zero since the quaternion product must be applied. q^{-1} is the inverse of the quaternion. These quaternions products are also easy to implement using the well-known matrix below, which would be used to do a quaternion product without any quaternion representations.

$$\mathrm{C(q)} = \frac{1}{|q|}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}, \boldsymbol{x}^p = \mathrm{C(q)}.\boldsymbol{x}^q \tag{2}$$

2.2. Deep Neural Network

With the presence of IMU sensors reading and recording of the resulted velocities which creates a trajectory across time, the use of machine learning and supervised learning is now possible. DNN is one of the most adopted approaches for supervised learning due to the advantages of having high accurate predictions and the ability to deduce relations between the data across time which cannot be deduced by a normal human. Despite not being able to show the determined relations, DNN can be tested for accuracy until we are satisfied with the result. The main objective of using a DNN model for inertial odometery to predicts accurately the velocity and trajectory of a moving subject (e.g., pedestrian) based on the history of the IMU measurements. This is possible due to Newtonian mechanics as shown in [20].

DNNs are different from normal neural networks by the number of hidden layers, as its depth increases with more layers. For some neural networks, it was observed that it makes sense to affirm that "the deeper the better". However, when deep neural networks start converging, degradation problems appear as the accuracy starts to saturate without reaching the state of overfitting. At this point adding more layers to the networks leads to higher training error [21]. Authors in [22] have introduced the Residual learning neural Network (ResNet) to address such problems. The idea of this network is to let the non-linear layers fit a residual mapping, instead of hoping each stacked layer (depth) fits the desired underlying mapping. It means, the network should approximate the identify function in which, the output $H(x)$ become the input X itself.

$$F(X) = X \tag{3}$$

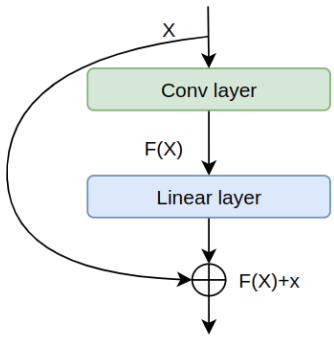

Figure 3: Two layers example of a ResNet.

When the input to the first layer of the model is avoided to be the output of the last layer of the model, the network should be able to predict whatever function it was learning before with the input added to it as follows.

$$H(x) = F(x) + x \tag{4}$$

instead of

$$H(x) = F(x) \tag{5}$$

The intuition is that $F(x) = 0$ is easily learnable, so that the learning is from differences between output and input. The block shown in Figure 3 is adopted to every few stacked layers. The residual mapping to be learned is given as $F(x, \{W_i\})$ which can be identified in case of two layers, similar to Figure 3, as $F(x) = W_2\ \sigma\ (W_1\ x)$.

The convolutional layers mostly have 3x3 filters and are restricted to have the same output feature map size as the number of filters. Finally, the network ends with a global average pooling and a fully connected layer with a Softmax [22] which provides the final max predicted probability of the possible candidates.

2.3. Kalman Filtering

KF is an application of Bayesian estimation method. It obtains optimal estimates using the deterministic and stochastic properties of the system model and measurements to execute recursive state estimation. The current estimate is updated by using the previous best estimates as inputs. The system model provides a prediction of the current state. Measurements are used to correct the predicted state and the measurements are combined by assigning weights to the prediction and the measurements. The new estimate is the weighted mean of the predicted state and the measurements [23]. In kinematic multi sensor systems, which are usually equipped with gyroscopes and accelerometers, KF is often used for the trajectory determination of the moving platform by using the motion model as well as the observation model.

3. It is all about Data

The human behavior patterns are not limited to a few categories. For a robust position estimation, we have tried to combine the datasets from IONet [11] and RoNIN [12] including the ones from RIDI [24]. We have also considered our new dataset with ground truth labels for natural human motions including both real and simulated 5G coordinates. We mainly have used 1 user, error of 1 meter as well as a frequency of 1 Hz as settings, to collect the training data. The data collection is still a work in process until using the developed research mode in real 5G wireless networks. However, the simulation allows researcher to generate noisy label data in absence of a real 5G network for further algorithm development.

3.1. Analytical Platform

During the development of the L5IN app, an analysis of the expected user groups was carried out at the beginning. In the background, the app collects all relevant sensor data from the smartphone and converts it into a position in an analytical platform so called research mode. Based on the user's own position and a previously selected destination, route guidance of the user is possible. The collected

sensor data is available for further development of the positioning algorithms (see Fig. 4). An independent position determination of the mobile device in the 5G network is currently under test and development. As soon as this is locally possible, this way of position determination will be favored.

The L5IN app was developed using the Unity framework to have a common code base for Android and iOS. The abstraction layer, which is used by Unity, in contrast to apps developed natively for mobile devices, the app only indirectly communicates with the base operating system, which offers special challenges with extremely high sampling rates of the sensors, as can be seen in Figure 4.

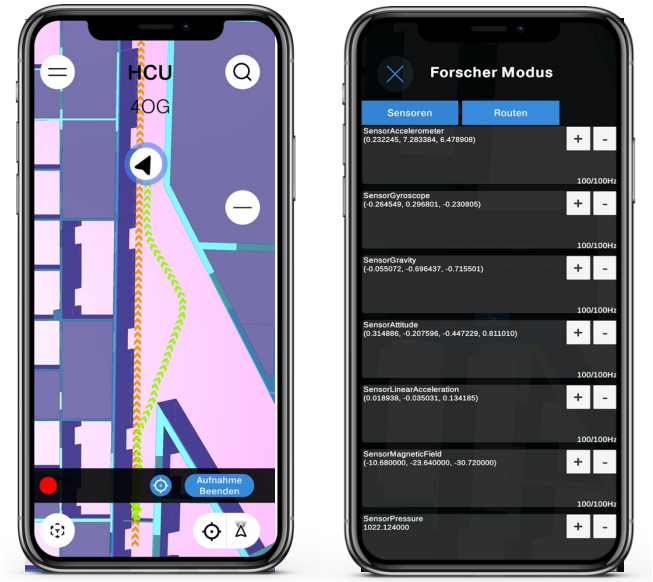

Figure 4: Data collection in the L5IN app (left) and adjustable sampling rate of the sensors (right).

With focus on the current state of the app, which is supported in the background by the previously mentioned services, it is possible after the installation of the app to store a starting position in a participating building. Afterwards, the user can move freely, or by entering his destination, guided in the entire building, even over several floors. A 2D or 3D map of the building is displayed, depending on the user's preference. A user login and the use of the research mode is only possible for the research team. It will be soon available as an AWS [25], which can help analyzing a high amount of positioning data.

In explorer mode, previously configured test routes can be loaded and run. In addition, so-called waypoints can be set at prominent points to be able to assign these points during a later analysis of the collected sensor data. An example of a recorded route is shown in Figure 5. The individual sensor values as well as all other recorded data can be downloaded as an export. The standardized data can then be evaluated in subsequent systems.

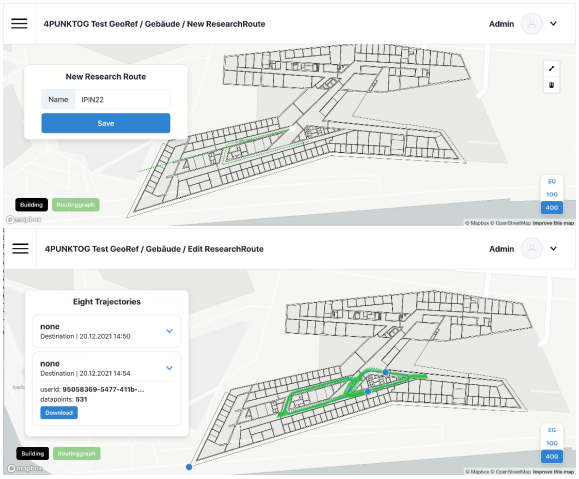

Figure 5: The research Mode test route (above) and data access (down).

3.2. Simulation

To enable the research and development of 5G positioning approaches without an existing 5G network, the following simulation has been developed. The simulation not only permits the generation of measurement data and to add noise to it with the desired measurement frequency, it also uses reference points and turns them into a ground truth trajectory. The measurements that can be simulated include signal sending and receiving time, angle and position coordinates. The accuracy of simulation values, as well as the granularity of the error information can be adjusted. In addition to the white noise error, semantic error that results from environmental conditions can be specified as well.

The main task of the Simulator is to generate data for researchers. If no 5G network is available, this part brings all the tools to simulate different antenna placements and quality scenarios. This requires reference points and sensor data (acceleration and gyroscope). After generating ground truth data, the measurements can be simulated and downloaded (see Fig. 6).

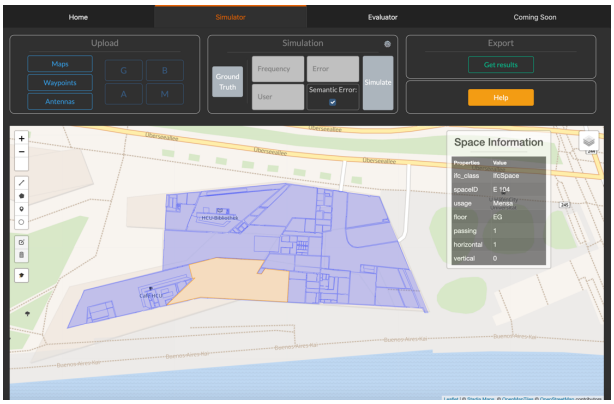

Figure 6: Overview of the simulation web application.

The website is hosted on Heroku's free cloud services [26], which can be connected to a GitHub repository. Heroku automatically updates the site, and the latest updates can be viewed online immediately. Unlike traditional websites, Python is not only used for all functions and calculations, but also for the layout and the entire website using Dash, an open-source framework for data visualization interfaces and predictive analysis [27]. The general workflow includes four main steps: upload, ground truth generation, simulation, and export, as it can be seen in Figure 7. The uploaded data can be viewed on the map. The preferred format is GeoJSON respective to its coordinate reference system. It is important to ensure that the georeferenced coordinates are used for both maps and waypoints.

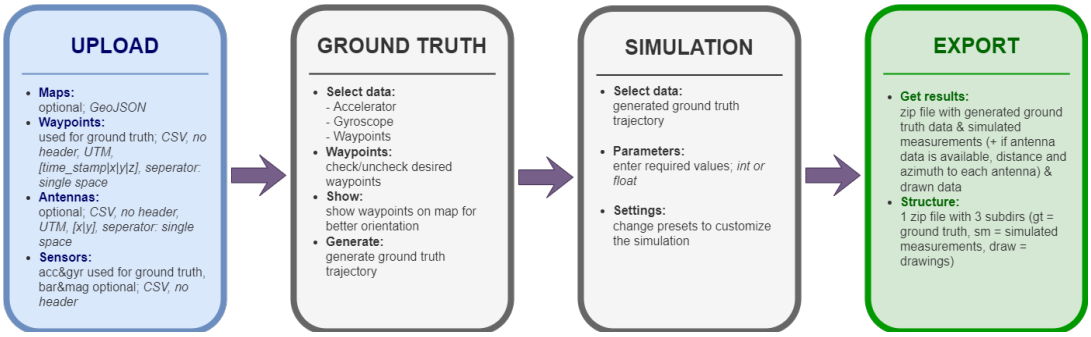

Figure 7: Overview of the simulation web application.

The next part consists of ground truth generation followed by the simulation of measurements. The ground truth calculation mainly uses the acceleration and gyroscope data as well as some reference points. The waypoints can be individually selected and displayed on the map. The trajectory can also be viewed on the map. To simulate measurements, ground truth data and values for frequency, error and the number of users is required as input. The settings button in the simulation section allows to change default parameters to further customize the semantic errors.

The website also offers the possibility to upload antenna coordinates. Simulating measurements such as the distance and azimuth to all antennas are also calculated and can be seen in the output file. In addition, various shapes such as lines, polygons and markers can be drawn directly on the map and be exported. The output file will be downloaded as a compressed file.

4. Models Implementation

4.1. Simulation

In general, a ground truth trajectory including the timestamps and coordinates is the basis of the simulation models. The user can then define an error range and measurement frequency. In the absence of a ground truth trajectory, this can be done within the web application. The simulated result with the same format of the timestamps and the corresponding coordinates and measurements, according to the chosen frequency, are then calculated from the ground truth by linear regression. For every coordinate pair a normal distributed random error is generated, using a random value from the specified error range as standard deviation. This error value is then added to the coordinate values. Here the generated error is assigned to a quality class, based on the granularity of the information about the measurement quality that is expected from the simulated network. The simulation then returns the timestamps and noisy 5G position measurements such as coordinates, distances, and angles, as well as the chosen error value and the assigned quality class, for each generated measurement.

For the semantic errors, the user can specify a range of time interval lengths, a range of the amount of time intervals and an error range for the semantic errors (see Fig. 7). The number of time intervals with semantic errors, as well as the duration of each interval and the assigned errors value is then randomly chosen from the specified ranges. The start times of the intervals are distributed randomly along the given ground truth trajectory. During the simulation process, the errors for each position are determined as explained above, until the start time for a semantic error interval is reached. During the semantic error interval, for each measurement the error is randomly generated from a normal distribution, using the semantic error value assigned to the interval as standard deviation for the error distribution.

Further, the number of users requesting a position from the network can be specified. This way, the potential loss of accuracy due to the lower frequency of received position information can be examined. The lower frequency of received position information is the result of the number of users being greater than the possible number of queries for each position measurement. For this, the duration between two received positions for a user is determined by dividing the number of users by the query frequency. The user receives the position according to the time stamp from the last measurement plus the estimated delay. If this calculated duration exceeds the duration between two measurements a time lag accumulates, resulting in a higher inaccuracy of the simulated estimation of the propagation of the user. In this way, one can roughly model the number of users. However, further investigation needs to provide an accurate effect of the number of users on the positioning performance.

4.2. Deep Neural Network

IMU values tend to be affected by bias and noise. This may lead to a major drawback to the whole prediction efficacy of the DNN model. We tackle this drawback, partially inspiring from the ResNet model from [12], by 1) improving the model's robustness to noise. That is by providing large and sparse data to better generalize the model and reduce the effects of noise measurements. 2) keeping track of the predicted velocities and detecting the error difference and faulty measurements and finally compensating them while predicting, considering the mentioned improvement above.

In our ResNet 18 model, one fully connected layer with 512 units is added at the end to regress a 2D vector. The network takes IMU values and creates a feature vector of size 6 for every timestamp t and calculates the velocity for a flexible time window of [t-n, t]. The network inputs tensor would have the shape of 6x200 and predicts the velocity tensor at t. From the predicted velocities we predict the current position based on the initial position provided by velocity integration. Figure 8 shows selected visualizations of the reconstructed trajectories against the ground truth.

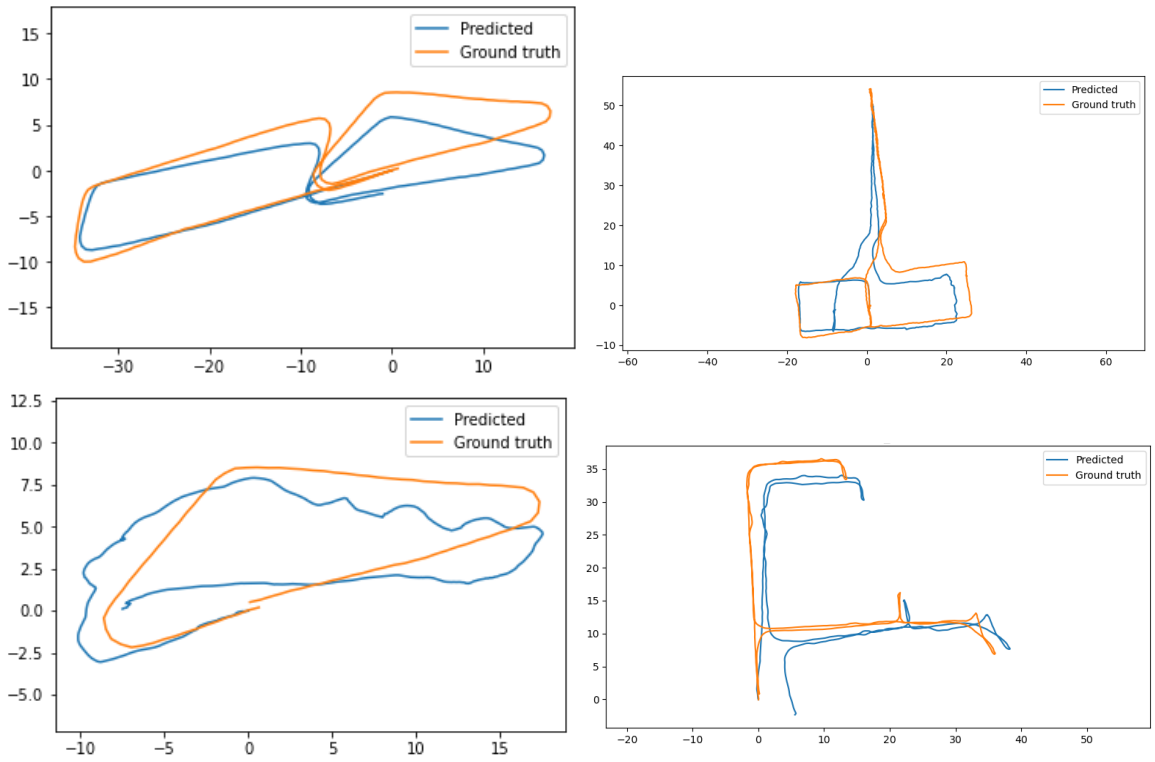

Figure 8: Selected visualizations from the deep inertial odometry model on the L5IN⁺ dataset including the handheld (top-left) and bag placements (bottom-left) as well as on the RoNIN (top-right) and RIDI (bottom-right).

4.3. Kalman Filtering

In the KF used for the state estimation of the device, the absolute and relative positions are used as the observation variables. To make the model simple and flexible, we have calculated the displacement outside the model. The movement model is defined as follows.

$$\begin{bmatrix} X_t \\ Y_t \\ \Delta X \\ \Delta Y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{t-1} \\ Y_{t-1} \\ \Delta X \\ \Delta Y \end{bmatrix} \tag{6}$$

The observation model is as follows.

$$Z_t = \begin{bmatrix} X_t \\ Y_t \\ \sum \Delta X \\ \sum \Delta Y \end{bmatrix} \tag{7}$$

where X, Y are the absolute positioning and the $\Delta X, \Delta Y$ are the inertial odometry output. The initial values for the state vector have been achieved from the label data. We assume no correlation between the four-state component. The initial covariance matrix of the process noise is

$$P_0 = \sigma^2 I_{4\times 4}. \tag{8}$$

The covariance of the system state is

$$P_S = E[(X - \hat{X})(X - \hat{X})^T] \tag{9}$$

The noise of the observed 5G coordinates and the inertial odometry could also be modeled as white Gaussian Noise. Therefore, the initial observation covariance matrix is

$$P_O = \sigma^2 I_{4\times4}. \quad (10)$$

Implementation results seems to be promising and have been illustrated in Figure 9.

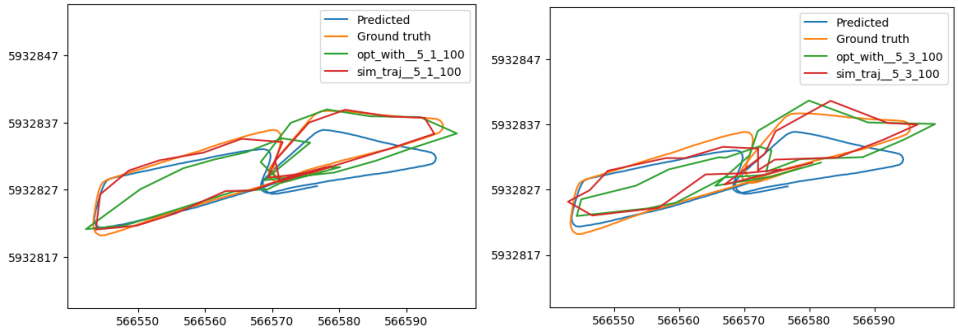

Figure 9: Selected visualizations of absolute Positioning in UTM 32 coordinate system (WGS 84) with the KF optimization.

5. Evaluation

We conduct evaluations on three tasks. First the DNN model comparison with the benchmark, using the same metrics as proposed in [12]. Absolute Trajectory Error (ATE), defined as the Root Mean Squared Error (RMSE) between the estimated and ground truth trajectories and Relative Trajectory Error (RTE), defined as the average RMSE over a fixed time interval of 1 minute, have been used. Table 1 illustrates our model performance on our test dataset. We believe that our network performance can be even better, due to the flexibility of the input data windows and the increased amount of training dataset. We also found out the estimated orientations and sensor values are severely corrupted on the above-mentioned datasets. However, our observations show that it is not related to the sensor calibration, but poor stability of the smartphone sensor values.

Table 1
The Model performance in comparison

Model	Losses	ATE	RTE
RONIN	0.040185, 0.040000	2.776	3.227
L5IN +	0.021484, 0.020613	2.741	2.524

The performance of the simulation web application with default semantic error adjustment including the network capacity of 500 request per second has been shown in Figure 10 using normal Cumulative Distribution Functions (CDFs). We have also considered the novel evaluation metrics as point in polygons (pip) percentage, initially defined by us in [5]. The simulation parameters are listed as error, frequency and the number of users in the figure legend. One can use the evaluation section in the web application, for further online investigation.

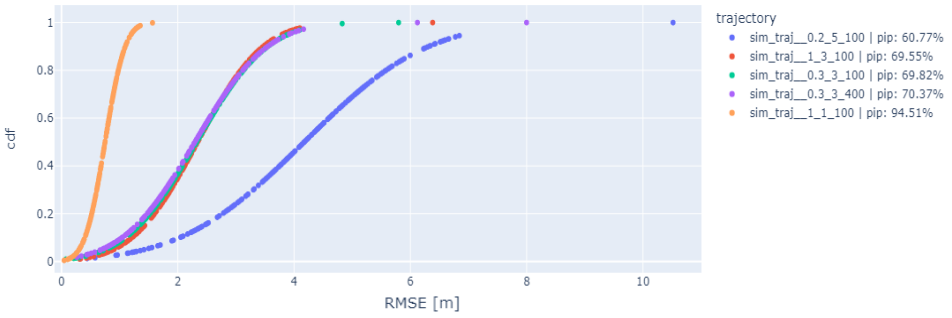

Figure 10: CDF of the simulation on RMSE

Finally, Figure 11 demonstrates the improvement based on the KF optimization in selected settings.

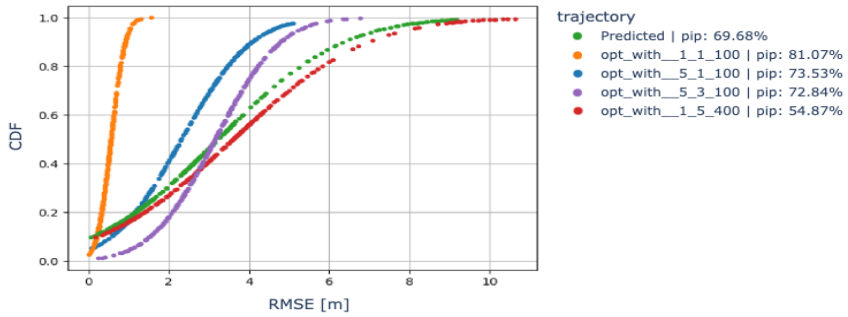

Figure 11: CDF of the KF based optimization on RMSE.

6. Conclusion and outlook

Indoor positioning is facing two big innovations, namely 5G positioning based on the soon promised implementation of release 16 and the fast development of data science approaches such as DNN. In this paper we have presented a platform for collecting smartphone sensor motion data and cellular network measurements, which works under the L5IN mobile application on the spot. Then by developing a web application, we could extend the cellular positioning simulation up to even the simulation of latencies due to a high number of users, providing a new benchmark dataset that is large and sparse. We then trained a new ResNet neural model with the combination of the data we collected and a benchmark dataset. The result was a model that can estimate a position based on IMU readings which is more robust to noise from sensor readings and can use the noisy label data coming from a 5G network, for instance. In the future, the proposed model would be tested through an ablation study, in which the relation between the countless human walking patterns and the amount of training data would be explained. We also optimized the inertial odometry model result with absolute positions, using a KF which showed improvement in the overall evaluation. The KF model is light enough to work on the smartphones in real time applications. In the future, we will also focus on the stochastic model. The main limitations of current machine learning approaches come from the start pose initialization, which we intend to improve in future works using the 5G network readings.

Acknowledgment

This research and L5IN project were funded by the Federal Ministry of Transport and Digital Infrastructure (BMVI), grant number VB5GFHAMB. We would like to thank the other project members, who did not participate as authors of this work.

References

[1] P. G. Savage, “Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms,” J. Guid. Control. Dyn., vol. 21, no. 1, pp. 19–28, 1998, doi: 10.2514/2.4228.

[2] K. Koide, J. Miura, and E. Menegatti, “A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement:,” vol. 16, no. 2, Apr. 2019, doi: 10.1177/1729881419841532.

[3] X. Chen, A. Milioto, E. Palazzolo, P. Giguˋ, J. Behley, and C. Stachniss, “SuMa++: Efficient LiDAR-based Semantic SLAM,” Accessed: May 09, 2022. [Online]. Available: https://github.com/PRBonn/semantic_suma/.

[4] S. Sun, D. Melamed, and K. Kitani, “IDOL: Inertial Deep Orientation-Estimation and Localization,” 2021, [Online]. Available: http://arxiv.org/abs/2102.04024.

[5] H. Shoushtari, C. Askar, D. Harder, T. Willemsen, and H. Sternberg, “3D Indoor Localization using 5G-based Particle Filtering and CAD Plans,” 2021, doi: 10.1109/IPIN51156.2021.9662636.

[6] H. Shoushtari, T. Willemsen, and H. Sternberg, “Many ways lead to the goal—possibilities of autonomous and infrastructure-based indoor positioning,” Electron., vol. 10, no. 4, pp. 1–17, 2021, doi: 10.3390/electronics10040397.

[7] A. A. Abdallah, K. Shamaei, and Z. M. Kassas, “Assessing real 5G signals for opportunistic navigation,” in ION GNSS, Sep. 2020, pp. 2548–2559, doi: 10.33012/2020.17702.

[8] “Release 16.” https://www.3gpp.org/release-16 (accessed Nov. 05, 2020).

[9] “Release 17.” https://www.3gpp.org/release-17 (accessed Dec. 09, 2020).

[10] A. García, S. Maier, and A. Philips, Location-Based Services in Cellular Networks: from GSM to 5G NR. 2020.

[11] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni, “Deep-Learning-Based Pedestrian Inertial Navigation: Methods, Data Set, and On-Device Inference,” IEEE Internet Things J., vol. 7, no. 5, pp. 4431–4441, 2020, doi: 10.1109/JIOT.2020.2966773.

[12] S. Herath, H. Yan, and Y. Furukawa, “RoNIN: Robust Neural Inertial Navigation in the Wild: Benchmark, Evaluations, & New Methods,” in 2020 IEEE International Conference on Robotics and Automation (ICRA), May 2020, pp. 3146–3152, doi: 10.1109/ICRA40945.2020.9196860.

[13] B. Rao, E. Kazemi, Y. Ding, D. M. Shila, F. M. Tucker, and L. Wang, “CTIN: Robust Contextual Transformer Network for Inertial Navigation,” in 2022 Preceedings of the AAAI conference on Artificial Intelligence.

[14] “Position sensors | Android Developers.” https://developer.android.com/guide/topics/sensors/sensors_position (accessed May 09, 2022).

[15] T. Willemsen, F. Keller, and H. Sternberg, “A topological approach with MEMS in smartphones based on routing-graph,” 2015, doi: 10.1109/IPIN.2015.7346952.

[16] C. Kjellson, M. Larsson, K. Astrom, and M. Oskarsson, “Accurate Indoor Positioning Based on Learned Absolute and Relative Models,” 2021 Int. Conf. Indoor Position. Indoor Navig. IPIN 2021, 2021, doi: 10.1109/IPIN51156.2021.9662534.

[17] S. Cortes, A. Solin, and J. Kannala, “Deep Learning Based Speed Estimation for Constraining Strapdown Inertial Navigation on Smartphones,” IEEE Int. Work. Mach. Learn. Signal Process. MLSP, vol. 2018-September, Aug. 2018, doi: 10.48550/arxiv.1808.03485.

[18] D. Harder, H. Shoushtari, and H. Sternberg, “Real-Time Map Matching with a Backtracking Particle Filter Using Geospatial Analysis,” Sensors 2022, Vol. 22, Page 3289, vol. 22, no. 9, p. 3289, Apr. 2022, doi: 10.3390/S22093289.

[19] S. O. H. Madgwick, “An efficient orientation filter for inertial and inertial / magnetic sensor arrays,” 2010.

[20] M. Kok, J. D. Hol, and T. B. Schön, “Using inertial sensors for position and orientation estimation,” Found. Trends Signal Process., vol. 11, no. 1–2, pp. 1–153, 2017, doi: 10.1561/2000000094.

[21] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Highway Networks,” 2015, Accessed: May 11, 2022. [Online]. Available: http://arxiv.org/abs/1507.06228,.

[22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-Decem, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[23] G. Wang et al., “A GNSS/INS Integrated Navigation Algorithm Based on Kalman Filter,” IFAC-PapersOnLine, vol. 51, no. 17, pp. 232–237, Jan. 2018, doi: 10.1016/J.IFACOL.2018.08.151.

[24] H. Yan, Q. Shan, and Y. Furukawa, “RIDI: Robust IMU double integration,” arXiv. pp. 1–16, 2017.

[25] “AWS | Cloud Server.” https://aws.amazon.com/ (accessed May 15, 2022).

[26] “Cloud Application Platform | Heroku.” https://www.heroku.com/ (accessed May 09, 2022).

[27] “Plotly: The front end for ML and data science models.” https://plotly.com/ (accessed May 09, 2022)