Аннотация

This work proposes a new pipeline for indoor navigation with Inertial Measurement Unit (IMU) sensors. The approach allows estimating the trajectory of the person using IMU data from a smartphone. Paper proposes an algorithm of smartphone orientation correction based on periodicity of the human walk and gravity vector. Experimental results demonstrate the strong performance of the correction method in the first case and novel architecture in the second case. A comprehensive error analysis of the algorithm allows to get insights about deep learning indoor navigation approach.

Содержание

1 Introduction

1.1 Inertial navigation

The task of accurate determination of the position of a smartphone in space is solved with high accuracy in open areas using GPS [1]. Modern technologies demonstrate excellent results with a deviation of less than a few meters [2].

However, the system has a disadvantage: it requires open space between the device and the satellite to pass radio signals. In the real world, we are often surrounded by trees, relief fractures, skyscrapers. The quality of geolocation decreases due to the reflection of radio waves. For example, using GPS to track trajectories inside buildings is almost useless [3].

This disadvantage of GPS navigation system can be solved by an indoor navigation system based on the data from Inertial Measurement Unit (IMU) sensors. Such system is a desirable technology for the research and industrial community, because IMUs are very cheap, efficient and present in every smartphone. Typical IMU contains an accelerometer, a gyroscope and a magnetometer. The signal generated by this sensors theoretically can be used to estimate the full trajectory and position of an IMU.

But the most of IMUs are not enough accurate to derive long trajectory from the raw signal using physic-based approach. There are bias and noise in IMU sensors which lead to accumulation of the positioning errors.

This problem can be reduced by a data-driven approach, which has has significant success in various fields such as natural language processing [4], computer vision [5], and signal processing [6]. In the paper, a new data-driven approach to solve the pedestrian indoor navigation problem is presented.

1.2 Contribution

The main contributions of this paper are:

- New deep learning model for the velocity estimation task.
- An algorithm, which reduces bias in rotation vectors collected from smartphone's IMU during subject's walk.
- An algorithm for the alignment of IMU's first frame and trajectory coordinate systems and show its effectiveness on RuDaCop Dataset.

- An evaluation and error analysis of our algorithm of trajectory estimation, which provides some insights which can be generalized to other DL-based IMU navigation algorithms.

The rest of this paper is organized as follows. In the following section, considered related works and discuss their advantages and disadvantages. In the third section, I give information about the used datasets. The third section provides the description of our solution. The fourth section is devoted to the conducted experiments. The fifth section provides an ablation study of the solution. The last section concludes the results and discusses the contribution of this paper.

2 Related work

There have been many studies that have been done with the focus on research in indoor space [7, 8, 9]. These works aim at different problems: localization, navigation, positioning, and tracking. Part of them is based on the integration of the signal from the accelerometer and gyroscope. Firstly, orientation is extracted from IMU data using such technique as Kalman filter. Then gravity vector is evaluated. After that, residual acceleration is integrated to get velocity, and the velocity is integrated to obtain a trajectory.

Integration of raw signals does not provide us an acceptable result due to inaccuracy of measurement [10]. There are various additional constraints which aimed at an improvement of the physics-based approach. In [11], authors offer a two-step method. The first step is a classification of the phone's position (5 different places). On the second, the mean velocity on the one-second frame is predicted with the regression model. The correction allows for improving the results of double integration significantly. IMU data in the experiments is smoothed with a Gaussian filter. One of heuristics of this work is using the stabilized-IMU coordinate frame, which is achieved by aligning Y-axis with the gravity vector.

The next part of methods is based on the idea that human walking is a repetitive process [12]. We can separate a person's trajectory into repetitive parts (steps). After that, we can predict direction and stride length inside the step. The method demonstrates high performance in experiments that satisfying the following restrictions. The IMU sensors are rigidly linked to a person, and his movement direction does not change relative to the phone position. Distance is proportional to a number of steps. For example, in paper [13] Kalman filter is used to obtain orientation from IMU data. After that, time-series is segmented into steps by the vertical acceleration threshold. The lengths of the steps are determined according to heuristic formulas.

Moreover, there are a lot of exciting works where data-driven approaches were used to extract useful information from IMU data. Authors of the paper [14] offer to use a complex neural network to process time series from sensors. Experiments that were aimed at the prediction trajectory of a car and classification human activities provide us information about the performance of the approach in the area. Let is show architecture in more detail. First of all, sensors data pass throughout the convolutional neural network. It helps to obtain a representation of the time interval. Then the recurrent neural network is used to accumulate information from the local representation. Finally, the output is given with a fully-connected neural network.

The article [15] is devoted to the problem of classifying a person's movement types: standing, walking, jogging, cycling, transport. Standard models are used (AdaBoost, KNN, SVM, DNN). It clearly comes from the title that presents some innovative DL method. However, it is just a fully-connected network with ReLU activation.

In the paper [16], a neural network approach is offered. Their approach is similar to [11], but the regression model is a neural network, and regressed velocities are directly integrated to get a trajectory. They also used a modified stabilized-IMU frame. Their model is agnostic to smartphone placement, and it doesn't deteriorate the quality, and it makes the approach more flexible. Authors of this work also develop a methodology of conducting experiments on indoor navigation with IMU.

In [17], another data-driven pipeline was introduced, L-IONet, a lightweight deep neural network framework to efficiently learn and infer inertial odometry. The model predicts a change of position and orientation with gyroscope and accelerometer data and consists of convolution and recurrent parts. Compared with other studies, the model uses a relative pose representation.

In [18], the authors presented a method for odometry. Inertial measurements and ground truth data were input to a neural network to learn movement from sensor data. The final neural network consists of a fully-connected part and a recurrent part. The method makes no assumption about either sensor placement or user motion. Performance evaluations demonstrated the neural network outperformed competing algorithms.

To sum up, various methods are used to predict movement patterns and estimate

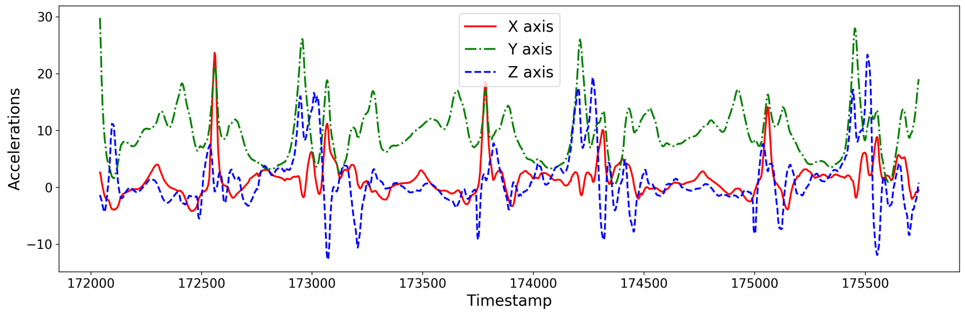

Рис. 1: Example of accelerometer signals

the trajectory with sensors data. The approaches include integration based on the laws of motion, finding corrections for integrable values using machine learning methods, fully end-to-end solutions using deep learning methods. In general, the research area has some serious problems. The absence of accessible publicly-available datasets limits the research. The results are often compared with trivial solutions on custom data.

3 Problem statement

Given 3 multidimensional time-series: accelerometer $\boldsymbol{a}$, gyroscope $\boldsymbol{w}$ and trajectory $\boldsymbol{r}$ of the person with IMU. Example illustration of the signals are shown on fig **??**. The task is to estimate $\boldsymbol{r}$ given sensors $(\boldsymbol{a}, \boldsymbol{w})$ with help of velocity model.

On time-series semgents of length Δt a machine learning model is trained:

$$M(\boldsymbol{a}^i_{t:t+\Delta t}, \boldsymbol{w}_{t:t+\Delta t}) : \mathbb{R}^{\Delta t \times 3} \times \mathbb{R}^{\Delta t \times 3} \to \mathbb{R}^3$$

$$W = \arg\min \sum_{e^i \in \mathcal{D}_L} \sum_{t \in \tau} (M(\boldsymbol{a}^i_{t:t+\Delta t}, \boldsymbol{w}^i_{t:t+\Delta t}) - \overline{\dot{\boldsymbol{r}}^i_{t:t+\Delta t}})^2 \to min$$

For trajecotry estimation $\hat{\dot{\boldsymbol{r}}}_{t:t+\Delta t}$ mean velocity vector estimates are integrated and moved to the original time grid.

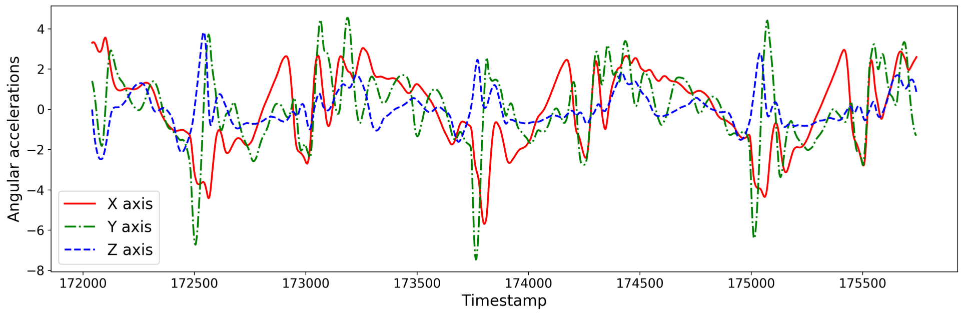

Рис. 2: Example of gyroscope signals

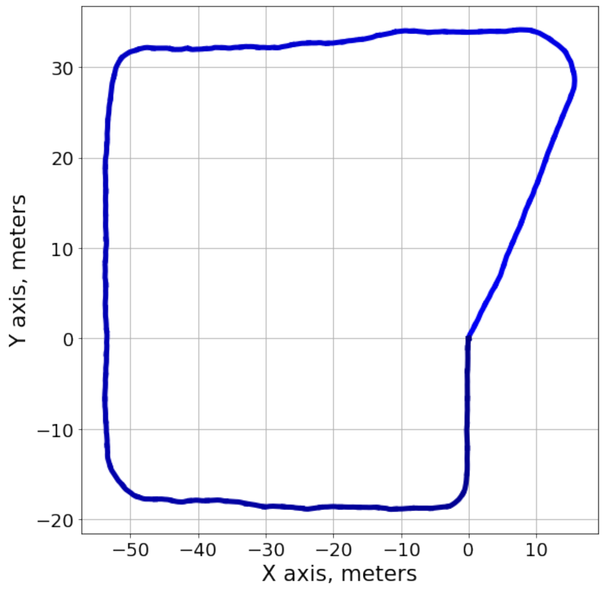

Рис. 3: Example of trajectory

4 Method

4.1 Preprocessing

4.1.1 Coordinate frames

There are four coordinate systems that are proposed to be used in this work. Also, there are methods of moving IMU readings from one systems to another.

IMU coordinate frame (I)

All the IMU readings are taken in the IMU coordinate frame. This coordinate frame is rigidly bound with the device. The system I is rotating and accelerating with the smartphone, which makes it hard to interpret raw data from the IMU system: readings and the system is constantly changing. To make the signal more interpretable, we need to transform readings to a more stable coordinate frame.

World coordinate frame (W)

To obtain sensors data in the world coordinate frame, we calculate quaternions in each moment according to the algorithm described below. First of all, we need to determine rule of constructing quaternion from angular velocity vector. Quaternion that represents rotation around axis is defined by the formula:

$$q = (\cos\alpha/2, \mathbf{v}\cdot\sin\alpha/2)\,,$$

where α denotes angle of rotation and $\mathbf{v}$ is rotation axis vector.

Thus we are able to construct instant quaternion q_j^i at moment t_j using $\mathbf{v} = \boldsymbol{w}_j \cdot (t_j - t_{j-1})$, $\alpha = |\boldsymbol{w}_j|$.

Final array of quaternions $\{q_i^W\}_{i=1}^n$ represents rotations that are needed to transform vector from I to world coordinate frame. Also we have to determine first quaternion q_0^W that aligns gravity vector in I system during immobility period $\boldsymbol{g}_0^I$ with vector $\mathbf{g'} = (0, 0, -1)$. Gravity vector $\boldsymbol{g}_0^I$ was calculated by averaging accelerometer readings during initial period of immobility. There is still uncertainty of horizontal rotation, which is not important for us due to circumstances described below in GT frame section. The following formulas were used to construct array:

$$q_0^W = (\cos\alpha_0/2, \mathbf{v}_0\cdot\sin\alpha_0/2)\,, \mathbf{v}_0 = \frac{\mathbf{g'}\times\boldsymbol{g}_0^I}{|\mathbf{g'}\times\boldsymbol{g}_0^I|}, \alpha_0 = \frac{\mathbf{g'}\cdot\boldsymbol{g}_0^I}{|\mathbf{g'}||\boldsymbol{g}_0^I|} \tag{4.1}$$

$$q_j^W = q_{j-1}^W \cdot q_j^i \tag{4.2}$$

This formula becomes clear if we imagine ourselves in a moving frame with the knowledge that quaternions represent rotation to the world coordinate frame.

We also determine rotation quaternions $\{q_i^{rot}\}_{i=1}^n$ that slightly differ from world quaternions and will be used lately. For them we let $q_0^{rot} = (1, 0, 0, 0)$ that correspond to zero rotation.

Then quaternions are applied to accelerations in I frame:

$$\boldsymbol{a}_i^w = Im\left(q_i^{W*}\cdot(0, \boldsymbol{a}_i)\cdot q_i^W\right)$$

After this procedure we obtain accelerometer data in world coordinate frame: $\boldsymbol{a}^w = (\boldsymbol{a}x^w, \boldsymbol{a}y^w, \boldsymbol{a}z^w)$.

IMU-stabilized coordinate frame (S)

The IMU-stabilized coordinate frame is proposed in [11]. It is obtained by the alignment Z axis of I frame with gravity vector in each timestamp, so the yaw angle axis is always oriented parallel to the gravity.

Instant gravity vectors array $\boldsymbol{g}_j^I$ for each timestamp t_j in I frame are obtained by applying quaternions $\{q_i^{rot}\}$ to $\boldsymbol{g}_0^I$. For each timestamp t_j we obtain quaternion q_j^{SI} of the minimal rotation from vector $\mathbf{g}'$ to vector $\boldsymbol{g}_j^I$. These quaternions transform vector from I to S frame. In this case, we bind Z axis with gravity vector not only for the period of immobility, but we try to keep Z axis vertical for the whole experiment.

The effects of pitch and roll rotations of the system are minimized in such a coordinate frame.

Ground truth coordinate frame (GT)

Ground truth coordinate frame is one in which the ground truth data is measured. Its Z axis is aligned with gravity, so the movement of the subject is only along the X, Y plane. This frame is different from the W frame by an arbitrary rotation along Z axis.

4.1.2 Gyroscope data correction

It is known that cheap gyroscopes always suffer from a drift, which leads to a compounding error in trajectory estimation. To reduce the accumulating error, we need a process of gyroscope readings correction.

One of the ways to do this is to correct gyroscope readings such that the gravity vector in W frame, calculated with the corrected gyroscope, is as close as possible to the vector $(0, 0, 9.8)$ which is true gravity vector in W.

We assume that $\boldsymbol{a}^W = \boldsymbol{g}^w + \boldsymbol{\mathcal{S}}^w$, where $\boldsymbol{g}^w$ is a constant gravity component and $\mathcal{S}^w$ is a high-frequency steps component. We derive $\boldsymbol{g}^w$ by filtering $\boldsymbol{\mathcal{S}}^w$ from $\boldsymbol{a}^W$ by deleting high frequencies from a fast Fourier transformation of $\boldsymbol{a}^W$. Then we compute quaternions $\mathbf{q}^{corr}$ of the smallest rotation from $\boldsymbol{g}_i^w$ to the constant vector $(0, 0, 9.8)$. Then we use corrected quaternions $\mathbf{q_i}^{corr} q_i^{rot}$ instead of $\mathbf{q_i^{rot}}^w$.

4.2 Trajectory estimation

Trajectory of the subject is restored with estimations of the average velocities during fixed time intervals, which include 6D IMU data. Average velocities are estimated by the neural network model. Input of the model is a matrix containing IMU data during an interval. Output of the model is a 2D vector which represents average velocity of the subject on the plane in S coordinate system. Target velocities in training set are numerical derivatives of the ground truth trajectories.

The whole sequence of IMU data is split time intervals, each of which is an input to the model. Model estimates average velocities over those intervals. All the average velocities are moved to W systems and integrated to obtain a trajectory.

4.2.1 Deep learning model

Convolutional neural networks are good at detecting local patterns in the data. To aggregate all the patterns in a given time-series and make an estimation it is appropriate to use LSTM layer.

We process IMU readings with a 1D version of ResNet-18 without the last convolutional layer stacked with 2 LSTM layers. The last hidden state of the second LSTM layer is processed by the fully connected layer, which outputs the velocity in S system.

We use ResNet-18 without the last layer stacked with two LSTM layers. Its score is not the best, but it differs from ResNet-18 very little, and it needs 3.7x less memory, so we decide to use it as a more practical variant. Input is 9-dimensional: x, y, z axes of rotation rates, accelerations, and low-frequency component of rotation rates in S coordinate system. Gyroscope readings are corrected with the method described in section 4.1.2 on the inference stage. The architecture of the resulting network is illustrated on 4.

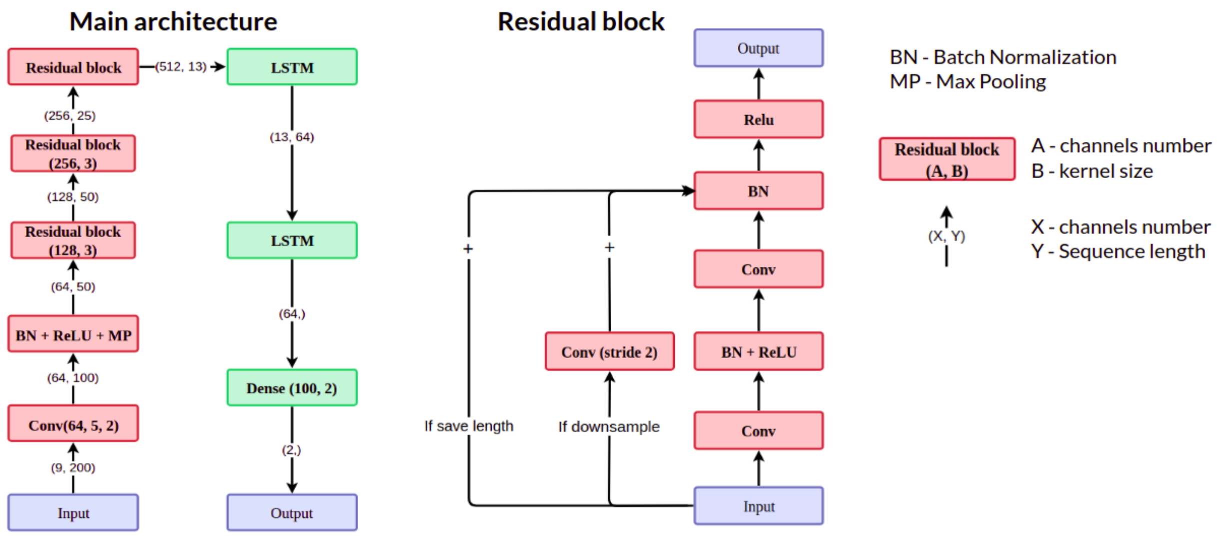

Рис. 4: Illustration of the final neural network architecture. Numbers in parentheses on arrows represent shape of the tensor. Numbers in parentheses in blocks represent layer parameters.

5 Experiments

The problem is to find the superposition of functions that we denote as F_{tr}, which transforms sensors data to trajectory estimation that will be close to the ground truth (5.1).

$$\arg\min_{F} \mathcal{L}\left(F_{\text{tr}}\left(\boldsymbol{a}, \boldsymbol{w}\right), \boldsymbol{r}\right) \tag{5.1}$$

5.1 Metrics

For evaluation and model analysis, we use the following metrics: **RMSE** (5.2), Mean integral distance (**MPE**) (5.3).

Absolute trajectory error (**RMSE**) defined as the root mean square error between predicted and ground truth trajectories:

$$RMSE(\hat{\boldsymbol{r}}, \boldsymbol{r}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T} |\boldsymbol{r}_t - \hat{\boldsymbol{r}}_t|^2} \tag{5.2}$$

Mean Positional Error defined as mean distance error divided by the length of the trajectory.

$$MPE(\hat{\boldsymbol{r}}, \boldsymbol{r}) = \frac{1}{T}\sum_{t=1}^{T} \frac{|\boldsymbol{r}_t - \hat{\boldsymbol{r}}_t|}{|\boldsymbol{r}_t|} \tag{5.3}$$

5.2 Data

Evaluation experiments were conducted on RuDaCoP [19] and RONIN [16]. Error analysis was conducted on RuDaCoP.

RuDaCoP dataset consists of about 1200 sets of inertial measurements from sensors of several smartphones. The trajectories are collected by walking up to 10 minutes long on a horizontal surface. Each trajectory started with 10-15 seconds staying to be suitable for initial gravity detection. Ground truth and IMU sensor data are synchronized with the same timestamp. Step data comprises timestamps and coordinates of each step with the information about which leg has touched the ground. Further, we use only data from accelerometer and gyroscope of the smartphone and ground truth trajectory to train and evaluate the model.

In RONIN dataset authors utilize game rotation vectors to derive orientations of the device. We used their gyroscope orientations instead, because we test our system on raw IMU output.

5.2.1 Ground truth alignment problem

There is a problem with ground truth data in RuDaCoP dataset. Smartphones have different placements and positions, and it is possible to find out their position based on IMU sensors. Gravity vector, calculated according to the accelerometer data and directions to the North, calculated according to the magnetometer data, allows us to identify phone orientation in World coordinate frame uniquely. But the orientation of Ground Truth relative to the smartphone is unknown since the North direction is not specified in ground truth data.

In this regard, we decided not to take into account North direction and tried to find out this unknown rotation angle.

The problem with ground truth is that we can't bring it to the same frame as data (W or S) based on data. Frame GT is arbitrarily rotated relative to W, and the angle of rotation differs from experiment to experiment. We need to find those angles and rotate GT to W system.

5.2.2 Ground truth alignment algorithm

Firstly, we bring sensors readings to a W frame using gyroscope data as in 4.1.1. As a result, readings and ground truth trajectories are in such frames (W and GT) that Z axis is aligned with the gravity vector. Then we use a rough trajectory estimation model. Prediction of such a model is expected to be by average in the same frame as input data. Then it is possible to calculate the angle between ground truth and prediction and rotate ground truth on this angle to align coordinate frames.

To eliminate overfitting from this process, we split a dataset into three parts. We train a model on the first and second parts and correct ground truth on the third part. Then train on the second and third and correct the first etc. Every iteration of ground truth alignment and model training decreases the misalignment and increases the accuracy of the model. This process converges for the most of the trajectories.

Model	RMSE	MPE
RoNIN	13.0	11.2
RNL	12.1	10.1
RNL + OC	10.9	8.9
RNL + OC + Aug	**10.5**	**8.5**

Таблица 1: RoNIN — analogous method, RNL - ResNetLSTM architecture, OC - oreintation correction, Aug - random horizontal rotation of sensors and trajectory in training phase

We used only those trajectories, which were adequately rotated by the ground truth correction algorithm. We used two neural architectures in this algorithm and checked whether both of them rotated ground truth trajectory on the same angle. Neural networks were trained for 3 epochs for each fold iteration of the algorithm. Totally there were 5 iterations of trajectory rotation procedure. Sixth iteration didn't change trajectories. If the trajectory rotation angles from different models are too different, we remove those trajectories from the training and validation samples. In total we removed 32 experiments from the dataset.

5.3 Evaluation

Our final model was trained on intervals of 200 timestamps (1 second). Consequential intervals differ in 20 timestamps, so intervals has overlap of 180. Sequences of IMU readings are moved to S system as well as target velocities.

Neural network was trained on GPU Nvidia GeForce GTX 1080 Ti. Training for 30 epochs with batch size equal to 128 took about 5 hours. Optimization were conducted with Adam optimizer with learning rate of 0.001.

Evaluation results of our solution is provided in table 5.3.

The results show that our gyroscope bias correction techniques improves quality of the model on both datasets and with both models. Improvement on RONIN dataset is much more vivid due to the higher gyroscope biases in this dataset. Our technique is effective for the worst case of gyroscope bias.

Also they demonstrate, that our model being 3.7x less memory demanding performs slightly better on RuDaCoP and slightly worse on RONIN dataset.

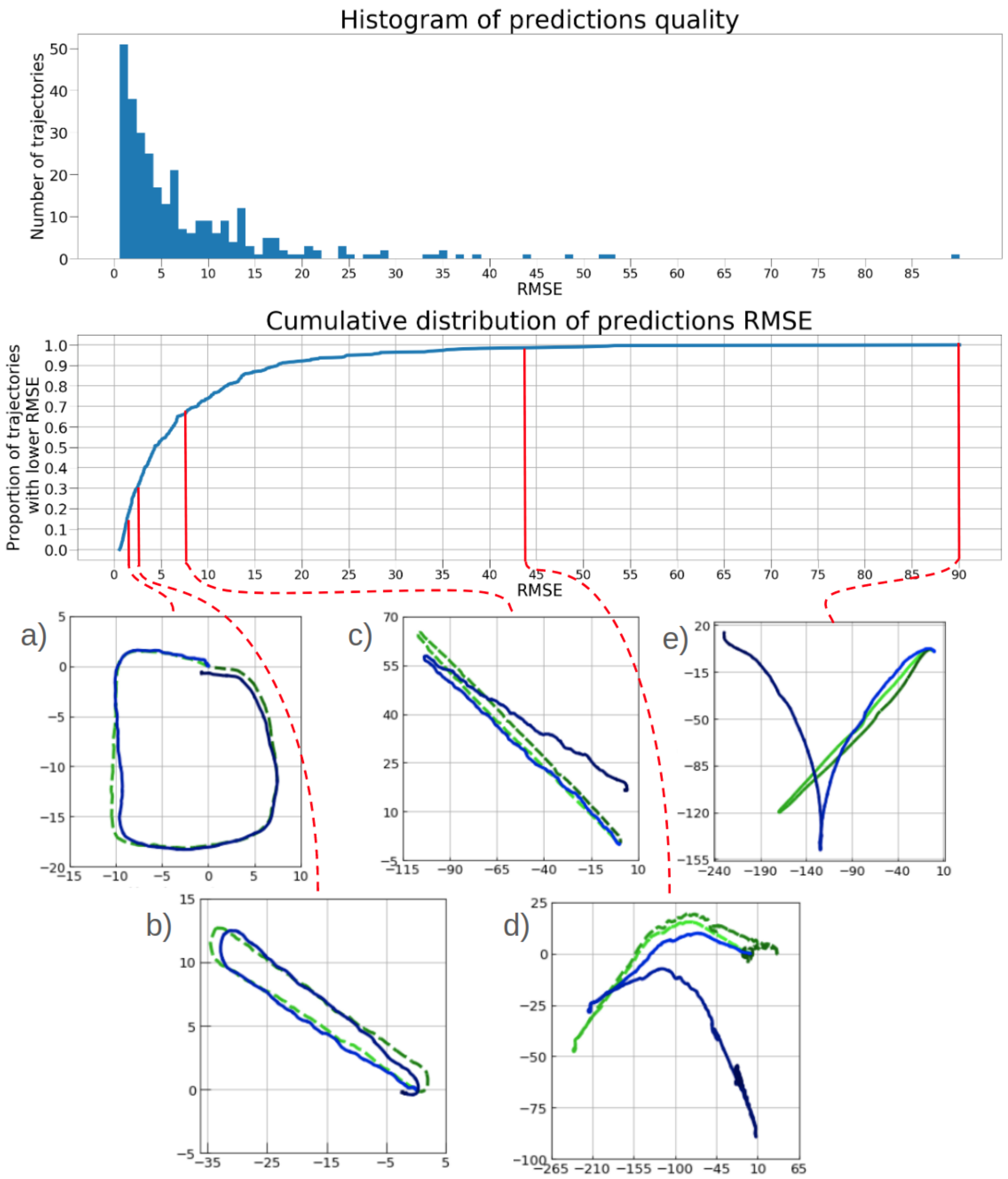

Рис. 5: Visualization of the model predictions quality obtained using 3-fold strategy. The histogram represents count of trajectories with the given prediction $\mathcal{L}_{\text{tr}}$ (5.2) score on X axis. The cumulative distribution represents how many predictions of trajectories has lower $\mathcal{L}_{\text{tr}}$ than the given number on X axis. The lower part of the figure is some examples of trajectory prediction with certain $\mathcal{L}_{\text{tr}}$, where blue line is prediction and green is ground truth. For example, percentage of predictions with lower $\mathcal{L}_{\text{tr}}$ than the (c) prediction is 67%, so the most of predictions are better, than (c). Inaccurate predictions like (d) and (e) are very rare, which can be seen on the histogram. The main source of prediction error is gyroscope drift, which is illustrated on the figure.

5.4 Error analysis

The rest of the experiments are devoted to analyzing the proposed method. Firstly, the authors investigate the overall quality of the final model on the whole dataset. The quality was estimated with 3-fold cross-validation strategy [20], see Figure 5. The proposed model demonstrates excellent performance. However, there are noticeable patterns of error. Most of them correspond to wrong course angle prediction, especially short-term drift at sharp turns. Long-term drift errors are also presented in the results. The model suffers from pedestrians with unstable speed because of the lack of data with the pattern. This type of error can be fixed using more various training data.

Then we estimate the impact of a holding position on overall model quality. The next positions:`upsidedown_right_overcoat`, `front_right_overcoat`, `bag`, `jacket`, `backpack` was excluded from the experiment due to lack of trajectories. Dataset splits for every holding position (as in k-fold) was created. For every experiment, one holding position was excluded from the training set. Data from each holding position is presented in the test part. Then we train and evaluate models for each holding position using the described splits (stratified by phone position k-fold strategy). We use this strategy to ensure that phone positions distributions match on the train and test datasets. According to the described procedure, we obtained results on all data. One can see the results for various metrics in the confusion matrices, see Figure 6. We can see that data from a particular position does not result in explicit dependence on overall model performance. The maximum values are placed on the diagonal, and it is easy to see that the presence of trajectories with `back_*` positions in the data is critical. In a particular column, metrics can be less than the corresponding value in `all` row what can be explained by the increase in the share of trajectories with a particular phone model. It is easy to see different values in columns — dependence on phone position that is not much considerable. Therefore if a sufficient dataset is obtained, the general trajectory estimation model can be constructed. Finally, we estimate the dependency of the model performance on the smartphone model. We split the source data in the same way with the previous experiment. The results of the experiment are presented in Figure 7. We can notice from the figure that maximum in each column on the diagonal. Data from the target smartphone is necessary to predict trajectories with higher accuracy. In the conducted experiment, we can see a more significant difference between metric values from different columns. That shows dependence on phone models. One can notice that the smallest values in the columns correspond `google_sailfish` and `samsung_dreamltexx` phones. It means that data obtained using these devices is well stabilized and has better quality. According to metrics on the diagonal, the presence of `samsung_startltexx` in the data is crucial.

To sum up, the experimental result demonstrates that the performance of the model depends on the phone model. Therefore it is imperative to obtain sufficient training datasets for different devices to increase the robustness of the model.

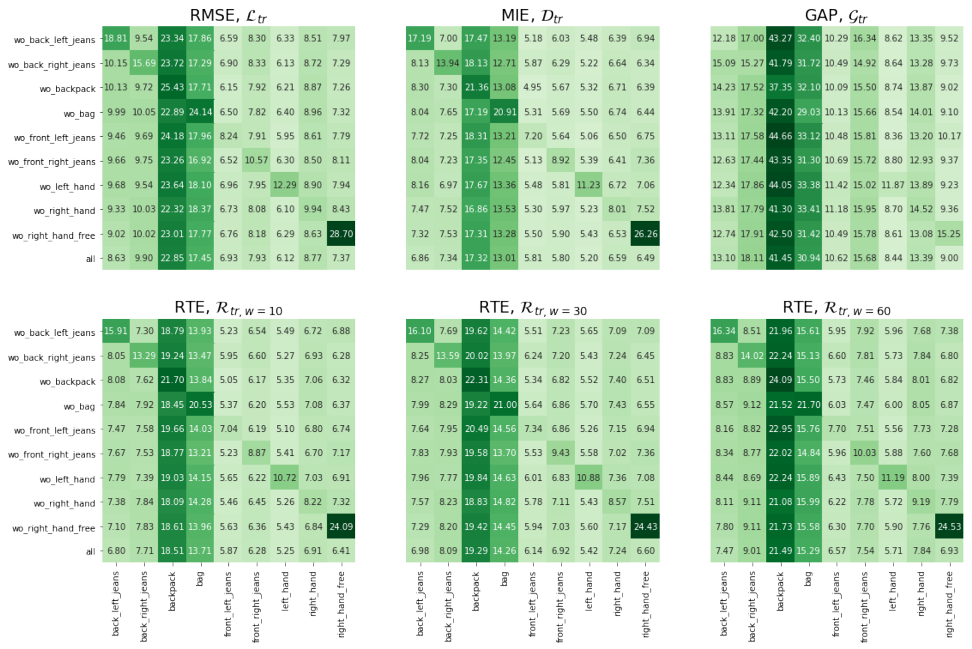

Рис. 6: Mean results on trajectories with particular phone position (along axis x) for various excluded from training phone positions (axis y with prefix `wo` — without, `all` means that all data was used). To obtain the results ResNet-18 and LSTM (RL) with bias reduction on inference (pc) and adding trend of gyroscope signal to the network input (trend) was used.

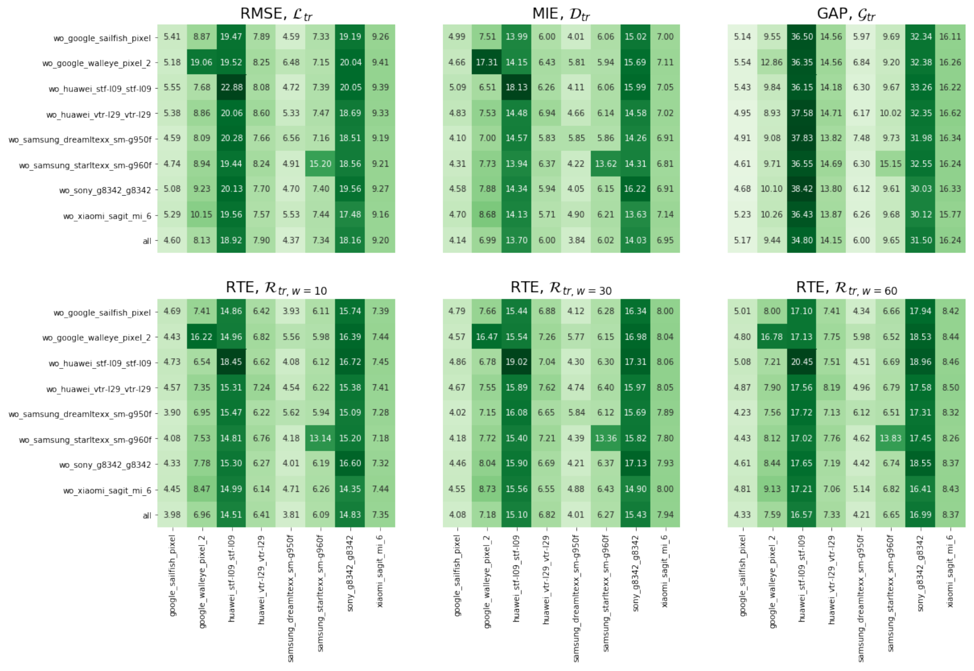

Рис. 7: Mean results on trajectories with particular phone model (along axis x) for various excluded from training phone models (axis y with prefix `wo` — without, `all` means that all data was used). To obtain the results ResNet-18 and LSTM (RL) with bias reduction on inference (pc) and adding trend of gyroscope signal to the network input (trend) was used.

6 Discussion

Our research clarifies several questions as well as demonstrates some new and classical problems in the IMU navigation problem and data-driven approach to it.

One of the main problems is gyroscope drift, which we partly solved in section 4.1.2, but this solution is hardly appropriate for the long trajectories. Cheap smartphone gyroscopes tend to develop a bias, which leads to compounding error in the trajectory estimation.

Another problem we faced is the insufficiency of walk regimes in the dataset. When we tested the model, trained on the given dataset, on our data, we noticed that it is not robust enough to changes in walking speed and a way of carrying a smartphone in hand. It is necessary to train in on the dataset that is as diverse as possible to use the model in real life.

The strongest bottleneck is a necessity of gyroscope integration for coordinate frame transformations. To use the full potential of neural networks it is necessary to move from the velocity regression on short intervals of IMU to some more global, trajectory and user conditioned approach.

7 Future work

We assume that the error can be reduced automatically by the neural network, because we observed, that successfully chosen architecture of the network can learn to ignore imperfections in data.

The first area of our future research is a more natural neural network for such kind of data. Increasing the vision field of the model can let it correct gyroscope bias earlier which can lead to a less cumulative error. However, addition of new convolutional layer or just changing the length of input IMU interval doesn't help.

The second problem to focus on is conditioning of the model output on the particular walk: gait, phone placement, speed etc. It can be implemented by dynamical calculation of context vector and usage of it in regression model. Such vector can be constructed as a hidden state of the IMU neural autoregression model.

The last problem to focus on in this area is making coordinate frame transformation an end-to-end procedure. It is necessary to remove the bottleneck of gyroscope integration and allow neural network to correct gyro bias as early as possible.

Список литературы

[1] AH Mohamed and KP Schwarz. Adaptive kalman filtering for ins/gps. *Journal of geodesy*, 73(4):193–203, 1999.

[2] Wan Rahiman and Zafariq Zainal. An overview of development gps navigation for autonomous car. In *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, pages 1112–1118. IEEE, 2013.

[3] George Dedes and Andrew G Dempster. Indoor gps positioning-challenges and opportunities. In *VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference, 2005.*, volume 1, pages 412–415. Citeseer, 2005.

[4] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[5] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[6] Dong Yu and Li Deng. Deep learning and its applications to signal and information processing [exploratory dsp]. *IEEE Signal Processing Magazine*, 28(1):145–154, 2010.

[7] Leading Edge and G Jobs. Centimeter-accuracy indoor navigation using gps-like pseudolites. 2001.

[8] Alessandro Mulloni, Daniel Wagner, Istvan Barakonyi, and Dieter Schmalstieg. Indoor positioning and navigation with camera phones. *IEEE Pervasive Computing*, 8(2):22–31, 2009.

[9] Gabriel Girard, Stéphane Côté, Sisi Zlatanova, Yannick Barette, Johanne St-Pierre, and Peter Van Oosterom. Indoor pedestrian navigation using foot-mounted imu and portable ultrasound range sensors. *Sensors*, 11(8):7606–7624, 2011.

[10] YK Thong, MS Woolfson, JA Crowe, BR Hayes-Gill, and DA Jones. Numerical double integration of acceleration measurements in noise. *Measurement*, 36(1):73–92, 2004.

[11] Hang Yan, Qi Shan, and Yasutaka Furukawa. Ridi: Robust imu double integration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 621–636, 2018.

[12] Stephane Beauregard and Harald Haas. Pedestrian dead reckoning: A basis for personal positioning. In *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, pages 27–35, 2006.

[13] Rui Zhang, Amir Bannoura, Fabian Höflinger, Leonhard M Reindl, and Christian Schindelhauer. Indoor localization using a smart phone. In *2013 IEEE Sensors Applications Symposium Proceedings*, pages 38–42. IEEE, 2013.

[14] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 351–360. International World Wide Web Conferences Steering Committee, 2017.

[15] Shih-Hau Fang, Yu-Xaing Fei, Zhezhuang Xu, and Yu Tsao. Learning transportation modes from smartphone sensors based on deep neural network. *IEEE Sensors Journal*, 17(18):6111–6118, 2017.

[16] Hang Yan, Sachini Herath, and Yasutaka Furukawa. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, and new methods. *CoRR*, abs/1905.12853, 2019.

[17] João Paulo Silva do Monte Lima, Hideaki Uchiyama, and Rin-ichiro Taniguchi. End-to-end learning framework for imu-based 6-dof odometry. *Sensors*, 19(17):3777, 2019.

[18] Changhao Chen, Xiaoxuan Lu, Johan Wahlstrom, Andrew Markham, and Niki Trigoni. Deep neural network based inertial odometry using low-cost inertial measurement units. *IEEE Transactions on Mobile Computing*, 2019.

[19] Andrey Bayev, Ivan Chistyakov, Alexey Derevyankin, Ilya Gartseev, Alexey Nikulin, and Mikhail Pikhletsky. Rudacop: The dataset for smartphone-based intellectual pedestrian navigation. In *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8. IEEE, 2019.

[20] Juan D Rodriguez, Aritz Perez, and Jose A Lozano. Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):569–575, 2009.