Smartphone-based Vehicle Tracking without GPS: Experience and Improvements

Yao Tong[1], Shuli Zhu[1], Qinkun Zhong[1], Ruipeng Gao[1]*, Chi Li[2], and Lei Liu[2]
School of Software Engineering, Beijing Jiaotong University, Beijing, China[1]
DiDi Corporation, Beijing, China[2]
{tongyao, zhushuli, qkzhong, rpgao}@bjtu.edu.cn, {lichi,liuleifrey}@didiglobal.com
* Corresponding author

***Abstract*—Nowadays, GPS and other global positioning systems have been widely developed, enabling accurate and convenient outdoor location-based services for vehicles. However, there are still two percents of areas in urban city that cannot be covered by satellites, e.g., underground parking lots, tunnels, and multi-level flyovers. Current positioning methods always rely on inertial dead-reckoning methods, but the performance is seriously affected by the low-quality inertial sensors embedded in crowdsourced smartphones. Based on our series of experiments with thousands of smartphones, we observe that the accuracy of existing inertial dead-reckoning methods is terribly affected by many factors, e.g., arbitrary and unknown placements of smartphones in car, inconstant inertial noises, and the diversity of smartphones and vehicles. In this paper, we explore a novel smartphone-based inertial sequence learning approach to infer vehicle's location in real time. We also propose a customized model refinement mechanism for individual drivers. Extensive experiments on DiDi ride-hailing platform have proved the effectiveness of our solution.**

***Index Terms*—vehicle tracking, inertial sequence learning, customised training**

I. Introduction

With the wide development of GPS systems, vehicular positioning has become very common for outdoor drivers, e.g., route planning, real-time navigation, and automatic driving. Nowadays, anyone can travel in unfamiliar areas without worrying about getting lost on the map.

However, according to the statistical data, there are still 2% of the area in urban cities that is not covered by satellites, including tunnels, underground parking lots, mountainous areas, etc. Drivers may get confused in such areas and could not find their way out. For example, drivers may forget where they park the car in a multi-level and maze-like parking structure.

Positioning without GPS signals is not a new topic. The current indoor localization methods always rely on WiFi [1], [2] and other RF signatures [3]–[7], but there are many shortcomings when used for vehicles [8]. First, the indoor localization principle is based on signal fingerprints, but there exists serious instability and susceptibility to indoor inferences [9]. Second, it is very time-consuming and labor-intensive to collect and calibrate the RF fingerprints at large scale. Such a high-cost, low-yield, and unstable positioning approach is not a long-term solution after all.

In this paper, we aim to enable a smartphone-only and real-time vehicle tracking solution without GPS or other RF signals. Instead of adopting the traditional inertial dead-reckoning for vehicles, we have conducted extensive experiments with the traffic data from a widely-used ride-hailing platform, analyzed its challenges and weaknesses, and proposed our temporal convolutional learning framework. We have also customized such model for refinement and inference on individual smartphones. Specially, our contributions consist of:

- **Experience**: We present our observations on tracking vehicles with dedicated inertial measurements from smartphones. Since the vehicular motion is typically a combination of rotation and translation movements, we investigate two motion factors, i.e., the angular rates by gyroscope and linear accelerations by accelerometer. We have summarized three technical challenges based on series of experiments: 1) the arbitrary and unknown placement of phones in the car; 2) the multi-factor and inconstant inertial noises; and 3) the diversity of crowdsourced smartphones and vehicles.
- **Improvements**: We explore a coordinate transformation solution to obtain the motion information of vehicles via internal smartphone's inertial readings. We also propose an inertial sequence learning framework to train and infer vehicle's locations and reduce inertial noises. In addition, we customize our model retraining mechanism which derives the accurate vehicle trajectory for individual smartphones.
- **Evaluations**: We collect a large-scale crwodsourced dataset from DiDi ride-hailing platform for model training and testing, in two large cities in China, respectively. Our results outperform traditional EKF-based tracking methods and other sequential learning models. In addition, we retrain our model with individual smartphone's inertial data to further improve the customized accuracy. The experimental results have demonstrated the great improvements of customized learning compared with the general inference model.

II. Background and Data Source

In this section, we introduce application scenarios of city-level vehicle tracking, and present our date source collected via crowdsensing.

2690-5965/21/$31.00 ©2021 IEEE
DOI 10.1109/ICPADS53394.2021.00032

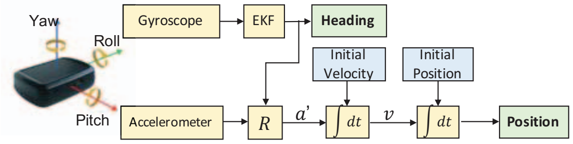


Fig. 1. Architecture of traditional inertial dead-reckoning. R is the acceleration projection matrix, and a' denotes the 2D accelerations on the ground.

A. City-level vehicle tracking

Location-based service (LBS) providers enable accurate city-level vehicle tracking service by GPS inside smartphones. Nevertheless, with the rapid development of urban cities, drivers frequently pass through many GPS blocked environments, e.g, multi-level overpass, underground parking lots, tunnel and urban canyons. Such loss of GPS information seriously affects the experience of vehicular navigation.

To satisfy the location requirements in such environments, providers mainly focus on strap-down inertial navigation systems (INS) for vehicle dead-reckoning, via micro-machined electromechanical systems (MEMS) devices inside smartphones. They exploit smartphone's inertial data as inputs, i.e., the 3-axis linear accelerations by accelerometer and the 3-axis angular rates by gyroscope, both measured at the phone's coordinate system. Instead of directly double integrating phone's accelerations into distance $\left(\Delta x_t = \int\int a_t dtdt\right)$, this method estimates the vehicular heading H_t and projects the phone's accelerations as its forwarding accelerations $a'_t = H_t \cdot a_t$, then the location changes are calculated as $\Delta x_t = \int\int a'_t dtdt$ to derive the user's location.

In practice, drivers place the smartphones with arbitrary postures in the vehicle, thus the phone's coordinate system (X^p, Y^p, Z^p) are not aligned with the vehicle (X^v, Y^v, Z^v). Therefore, they always combine the Extended Kalman Filter (EKF) with INS to estimate the smartphone's posture in advance (shown in Figure 1). Specially, the EKF algorithm constructs the vehicular attitude states and measurement equations, and leverages the gyroscope to continually updates the attitude states owing to the measurement observations, thus deriving the heading direction H_t.

However, due to the low-quality inertial sensors embedded in commodity smartphones, large drifts and noises appear frequently and they are easily accumulated to extreme location errors.

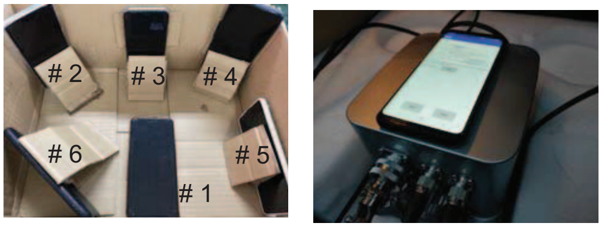
B. Data source

In order to explore the influence of inertial sensors for crowdsourced vehicle tracking, we have collected millions of driving trajectories by the DiDi ride-hailing platform. Our data source can be summarized into two parts, i.e., the crowdsourced dataset and dedicated dataset. Details are shown in Table I.

Part 1: Crowdsourced dataset. Our large-scale real-world crowdsourced dataset is collected by DiDi platform in urban cities in China between Dec. 1 2020 and Dec. 30 2020. The training data are gathered via crowdsensing from 876 phones during 308 hours, covering $17195km$ distances. The test data are from 829 phones during 191 hours, covering $5591km$ distances.

TABLE I
DATASET INFORMATION

Source	Crowdsouced by DiDi	Dedicated by ourselves
Location	Beijing, Shenzhen and so on	Beijing
Time	December 1 ∼ 30, 2020	April 1 ∼ May 30, 2021
Distance	(17195 + 5591) km	62 km
Smartphones	(876 + 829) phones	6 phones
Ground truth	GNSS (1Hz)	Dedicated INS (100Hz)



(a) Mould with 6 phones

(b) Dedicated IMU device

Fig. 2. Dedicated measurement by a mould with six smartphones.

Part 2: Dedicated dataset. Since the crowdsourced data lacks the ground truth of smartphone's posture, we build a mould to hold six smartphones with different placements, as shown in Figure 2(a). Meanwhile, we leverage a dedicated IMU device in the same vehicle and collect its measurement results as the inertial ground truth.

III. Observation and Challenges

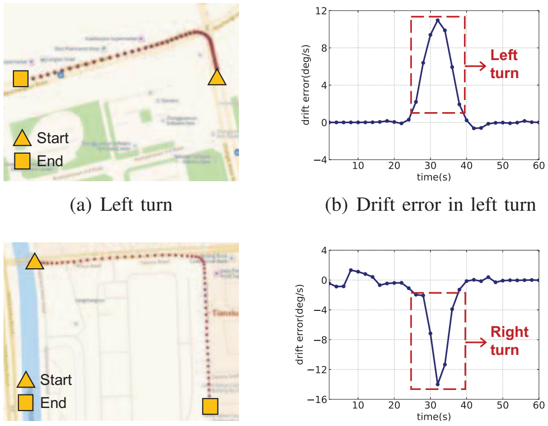
In this section, we conduct a series of experimental studies to investigate the inertial impacts on vehicle's tracking accuracy. All observations are based on our dedicated measurements proposed in Table I.

A. Observation of inertial noises

The vehicular motion is typically a combination of rotation and translation movements. Thus we investigate two motion factors, i.e., the angular velocities measured by the gyroscope and linear acceleration measured by the accelerator, both in the smartphone's coordinate system. In order to explore the error caused by inertial noises, we fix the smartphone in vehicle and align it with the vehicle, thus the smartphone's inertial readings are approximated as the vehicle's.

Gyroscope. Gyroscope measures the real-time angular rates (w^p_X, w^p_Y, w^p_Z) around the X^p-axis,Y^p-axis and Z^p-axis in smartphone coordinate system. Theoretically, with continuous integration on angular velocities, we can derive the vehicle's rotational motion. However, due to the low quality of MEMS's gyroscope, its readings are affected by numerous noise, such as the constant bias, thermo mechanical white noise and flicker noise [10].

To investigate the impact of gyroscope errors on rotational motion, we perform continuous integration on angular rates in



(a) Left turn

(b) Drift error in left turn

(c) Right turn

(d) Drift error in right turn

Fig. 3. Two GPS trajectory with gyroscope drift errors in 1 minute.

TABLE II
THE SKEWNESS AND KURTOSIS IN DIFFERENT ROADS.

Phones	Left turn		Right turn		Straight	
	skew	kurt	skew	kurt	skew	kurt
#1	1.94	3.96	-2.79	8.04	1.01	7.21
#2	1.96	4.06	-2.81	8.17	1.01	7.23
#3	1.94	3.96	-2.79	8.03	1.01	7.2
#4	1.93	3.89	-2.80	8.06	0.99	7.29
#5	1.96	4.05	-2.81	8.15	1.0	7.09
#6	1.95	3.98	-2.79	8.04	1.01	7.2

TABLE III
THE SKEWNESS AND KURTOSIS DURING ACCELERATING AND DECELERATING.

Velocity	Skewness		Kurtosis	
	skew	z-score	kurt	z-score
Accelerating	0.24	1.49	-0.74	-2.28
Decelerating	0.19	2.0	-0.68	-3.51

straight road segments and turning road segments, respectively. Straight segments are defined as the angular change less than 10 degrees within one minute and turning segments denote the ones with more than 45 degrees. The ground truth is measured by a dedicated IMU device as described in our dedicated dataset. We compared their angular drift error (the angular difference between ground truth and integration result) in unit time.

An interesting finding is that the drift error of left turn and right turn have the same amplitude but inverse value, as Figure 3 depicts. Based on the skewness and kurtosis value in Table II, we find that no matter driving on turning road or straight road, the skewness and kurtosis of drift error are not closed to zero. In addition, six phones have the same potential manifestation. All phenomenons demonstrate that the gyroscope's drift errors don't follow the normal distribution.

Accelerometer. Accelerometer measures the 3-axis linear accelerations (a^p_X, a^p_Y, a^p_Z) in smarpthone's coordinate system. The types of accelerator' drift errors are analogous to gyroscope, expect the arising errors due to the double integration for distance [10].

To investigate the impact of linear acceleration, we extract the accelerating and decelerating segments with 5-second intervals in straight roads, and calculate the integrated drift errors on six phones. The ground truth is also supported by the dedicated IMU device. Although its skewness and kurtness are closed to zero (Table III), their corresponding z-sore does not satisfy the hypothetical conditions ($-1.96 \leq$z-score≤ 1.96) while the inspection level $\alpha = 0.05$. As a result, the drift error of accelerometer also disobey the normal distribution .

Inertial sensor with the same type. In order to analyze the impact of the same inertial sensor in different smartphones, we collect two smartphones with the same type in the same vehicle. We leverage a 10s sliding window to process the temporal sequence and calculate its accumulated errors. As Figure 4 depicts, the linear acceleration errors on Phone #1 and Phone #2 have an opposite error trend. When acceleration changes in $(0, 1)$, Phone 1 gets a lower error while Phone 2 obtains more. They also have different maximum orientation errors when the average angular velocity is 0.3 deg/s. This experimental result demonstrates that different smartphones with the same type performs differently in acceleration and angular rates even in the same vehicle.

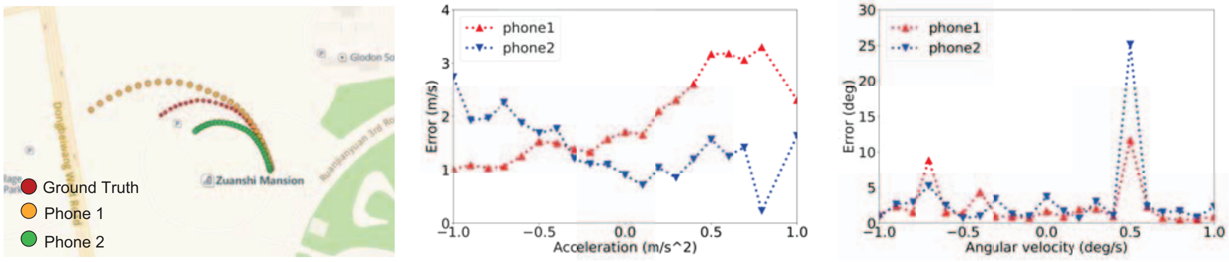
B. Challenges and design guidelines

Based on our observations, the challenges of city-level vehicle tracking are summarized as follows:

1) Arbitrary posture: Drivers have their own preference to place the smartphone in vehicle, thus the posture of smartphone in car is arbitrary and we are ignorant of the relationship between smartphone's coordinate system and vehicle's. Exiting EKF-base heading estimation method is affected by the low-quality of MEMS sensors in smartphone. To deal with such a challenge, we propose a PCA-based coordinate transformation method in section IV C.

2) Inconstant error distribution: The inertial error distributions are seriously varying via crowdsensing. The drift error of accelerometer or gyroscope don't follow the normal distribution, and even the drift distribution of left turns is not the same as right turns. Thus, we cannot build a general error distribution model to meet all smartphones. A possible approach is to leverage the deep learning method such as RNN for time sequence learning. They capture temporal dependencies instead of double integration with a general error distribution. In section IV C, we will introduce the details of our inertial sequence learning method.

3) The difference of smartphones: Due to the different commodity inertial sensor, we have realized that different smartphone proverbially has the difference localization accuracy. A straightforward but naive thought is to customize the inference model for each type of smartphone. However, to our surprise, two smartphones with the same type still have diverse localization performance when driving in same vehicle (Figure 4). Thus, model customization for each smartphone is a wise and brief approach. Detailed introduction is in section IV D.



(a) Trajectories on the map

(b) Linear acceleration errors

(c) Angular rate errors

Fig. 4. The velocity and orientation comparison of two smartphones with the same type.

IV. Solutions

To solve the above three challenges, we propose a novel customized vehicle inertial tracking architecture. As depicted in Figure 5, it is comprised of 3 phases: (1) a coordinate transformation which transforms the smartphone's inertial data into vehicle's; (2) inertial sequence learning which uses historical data to train location inference model and (3) customized model retraining which derives the accurate vehicle trajectory for individual smartphones.

A. Coordinate transformation

Intuitively, the direction of majority accelerations indicates the vehicle's forwarding orientation on the horizontal plane (excluding the gravity). Thus, we explore a three-dimensional PCA algorithm to extract the direction vector of the largest acceleration variances, i.e., the first component direction produced by PCA (shown in Figure 6).

Next, we transfer the acceleration readings from the phone to the vehicle by establishing a transformation matrix $C^{v \to p}$. It is a set of mutually orthogonal three-dimensional coordinate axes produced by PCA, i.e.,

$$C^{v \to p} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \tag{1}$$

Then, we convert the inertial readings from mobile phone to the corresponding vehicle, i.e.,

$$a_v = a_p (C^{v \to p})^T \tag{2}$$

B. Inertial Sequence Learning

Since the inertial error is always volatile and inconstant, a general error distribution model is hard to construct for crowdsourced smartphones. Instead of using traditional inertial dead-reckoning method, we explore an inertial sequence learning framework to produce a neural network that minimizes the differences between actual inertial readings and corresponding location observations.

1) Model formulation: Intuitively, we aim to estimate the location variance ($\Delta lon_{0:t}, \Delta lat_{0:t}$) during each time interval. We split the velocity vector into the change of speed and bearing ($\Delta v_{0:t}, \Delta o_{0:t}$). Specially, the $\Delta o_{0:t}$ is an intricate nut to crack, thus we suppose the angular change of bearing is in (0, 360), and the clockwise change is positive while the counterclockwise change is negative. Since the turning angles with more than 180 degrees in a very short duration is rare, we derive the angular change of bearing $\Delta o_{0:t} = \min\{\Delta o_{0:t}, 2\pi - \Delta o_{0:t}\}$.

2) Inertial sequence input: Based on the DiDi ride-hailing platform, we collect the raw inertial data from millions of travelling orders. It consists of the smartphone's 3-axis accelerations (a_X^p, a_Y^p, a_Z^p), 3-axis angular velocities (w_X^p, w_Y^p, w_Z^p), 3-axis gravity accelerations (g_X^p, g_Y^p, g_Z^p), GPS location (longitude and latitude), speed and bearing (lon, lat, v, o). The sampling frequency of the first three sensors are $50Hz$ and last one is $1Hz$. Instead of using the raw concatenated 9-dimension inertial data as model input, it is absolutely a grievous scheme for a deep learning network to deploy in smartphones, thus the real-time performance of location inference can not be promised.

To get rid of the complexity of training and inference, we extract the most efficient features instead of raw inertial data to track vehicles. 1) The initial speed feature $init\_speed$ and bearing feature $init\_bearing$ (vehicle's heading direction) at initial point ($init\_lon, init\_lat$) in $t = 0$, i.e., last valid GPS's information. 2) The current features consist of accelerometer features, gyroscope features and gravity features, and we calculate the corresponding $mean, min, max, std, var$ and sum at 1-second interval. Therefore, the length of 1-second inertial sequence is reduced from $3 \times 3 \times 50 = 450$ to $3 \times 3 \times 6 = 54$, and the memory of train set is only about 1/5 of the original.

3) TCN Architecture: We use a TCN (Temporal Convolutional Network) architecture with residual blocks to learn the inertial sequence. Although recurrent neural networks (e.g., LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit)) have become the most popular architectures for sequence modeling, they are inefficient on mobile devices due to the large number of parameters. Instead, we adopt the TCN which is more suitable for deployment on smartphones with a long-range receptive field. Specifically, residual blocks used in TCN architecture have been proved to be an effective way to train deep networks, which enables networks to transmit information in a cross-layer manner.

As shown in Figure 7, the hyper-parameter settings of our network are: 9 fully connected layers and 3 TCN residual

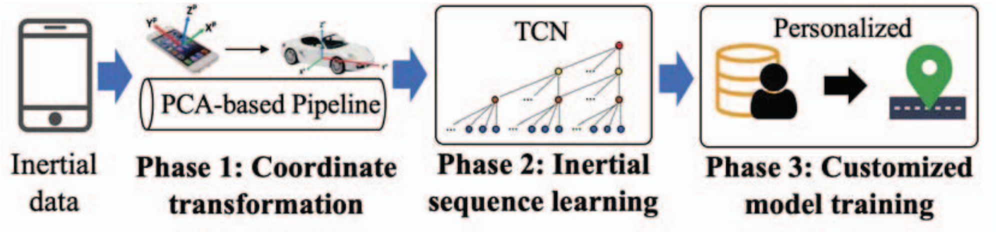

Inertial data
PCA-based Pipeline
Phase 1: Coordinate transformation
TCN
Phase 2: Inertial sequence learning
Personalized
Phase 3: Customized model training

Fig. 5. System overview.

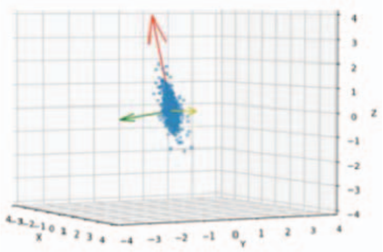


Fig. 6. 3D Principal Component Analysis. The red arrow denotes the direction of majority accelerations, i.e., vehicle's heading direction.

TABLE IV
THE DIFFERENT WAYS TO TRAIN PERSONALIZATION MODELS.

Period	Models	Has PCA	Training set	Testing set
Offline	general model	×	①	②
	model #2	×	① + ②	②
	model #3	√	① + ②	②
Online	model #3	√	① + ②	③
	model #4	√	① + ② + ③	③

blocks (batch size of 128 and the learning rate of 0.001). We use the ReLU activation function in each TCN residual block, whereas the Leaky-ReLU is used for the first six layers of FC network. In each TCN residual block, we apply the dropout rate of 0.5 to mitigate overfitting. Meanwhile, we use the weight normalization and batch normalization to accelerate model convergence, and the Adam optimizer to iteratively update network weights.

4) Loss Computation: Since our learning objective combines both velocity and orientation estimates, we use the weighted *SmoothL1Loss* as our loss metric, which is less sensitive to outliers than the MSE (Mean Squared Error) and prevents exploding gradients in some cases. The *SmoothL1Loss* is a popular metric to calculate the loss value, i.e.,

$$L(\hat{x}, x) = \begin{cases} |\hat{x} - x| - 0.5 & |\hat{x} - x| > 1, \\ 0.5\,|\hat{x} - x|^2 & |\hat{x} - x| \leq 1. \end{cases} \tag{3}$$

C. Customized Training

With the large-scale database collected via crowdsensing, we have established a general location inference model by all users. However, based on our observation in Section III, vehicle's tracking accuracy differs a lot with the same tracking method, even on the same smartphone as Figure 4 shows. A straightforward approach is to customize the location inference model to fit individual smartphones. Though appealing, it has two inherent limits. First, people upload their location information and it will lead to privacy concerns. Besides, personalized learning per user will cost dozens of minutes on smartphones. Thus, on-cloud personalization is not realistic for location service providers to confirm real-time and iterative vehicular tracking.

To address the preceding issues, we propose our customized model training mechanism as depicted in Figure 9. Its training process comprises three phases:

1) Cloud training: First, we use the crowdsensing dataset to train a general location inference model. This phase is typically performed on the cloud. The trained model is called as a general model. After achieving great inference performance on the crowdsourced dataset, we dispatch the general model to all users' devices.

2) Offline training: On each mobile device, we leverage its private dataset and retrain the general model. The personalization is mainly achieved by synthesizing the dataset which user refuse to upload. Finally, we deploy the personal model on individual smartphones.

3) Online training: During vehicle's outdoor driving, each smartphone collects the latest inertial data with reliable GPS. We use them to training data in real time and continuously reinforce our personal model.

Given this customized training mechanism, we derive a better individual tracking model without incurring privacy issues. Note that offline training and online training are both scheduled when devices are in charging mode, minimizing the impact of battery life.

V. EVALUATION

A. Methodology

Training and testing datasets. The detailed dataset is shown in Table I. As a supplement, training dataset is at the sample length of 10 seconds, while testing dataset is 60 seconds. To distinguish crowdsourced dataset and private dataset, we name the crowdsourced dataset as ①, the private historical data in April as ②, and the private online data in May as ③ (shown in Figure 10), where ② and ③ come from the dedicated dataset with six smartphones. We regard the data in April as historical input for offline training, and treat the private data in May as the latest trajectory for online training. Detailed descriptions about private model training are shown in Table IV.

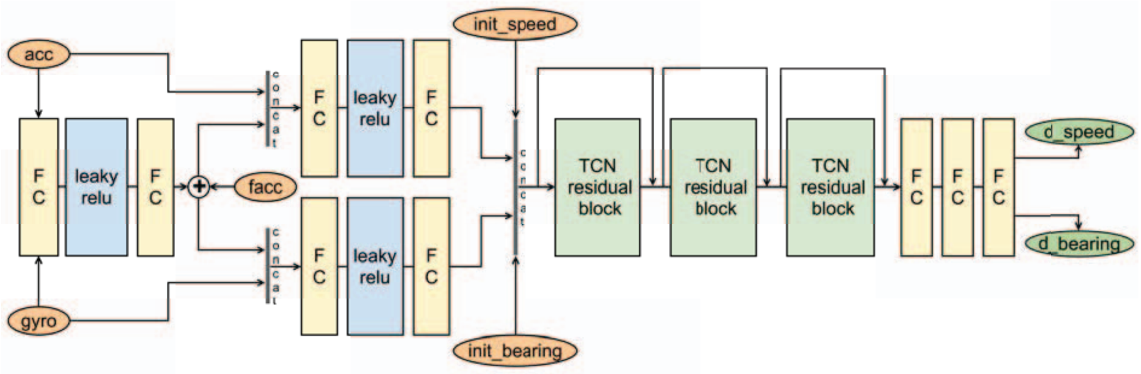


Fig. 7. TCN framework for vehicle tracking.

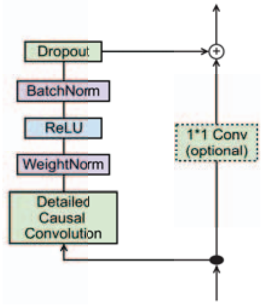


Fig. 8. The details of a TCN residual block

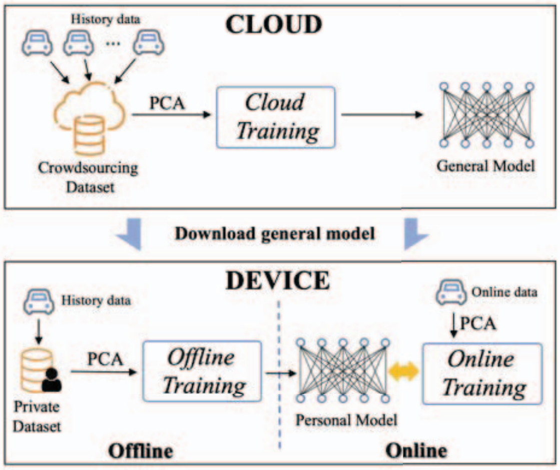


Fig. 9. Mechanism of customized training.

Compared Alternatives. We compare our performance with EKF, LSTM, and GRU solutions for vehicle tracking. The EKF approach is currently applied by many ride-hailing platforms to track vehicles without GPS. The LSTM and GRU are implemented by replacing the TCN block with LSTM (*torch.nn.LSTM*) and GRU (*torch.nn.GRU*). Their special network parameters: *hidden_size=256* and *hidden_layer=3*.

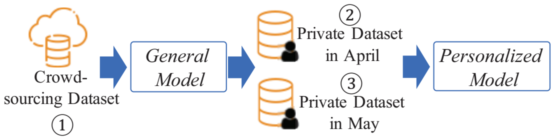


Fig. 10. Multi-stage dataset for model tranining.

TABLE V
MAE ON LOCATION INFERENCE (M)

Models	30-second tracking		60-second tracking	
	Beijing	Shenzhen	Beijing	Shenzhen
EKF	74.77	72.54	165.53	163.95
GRU	33.16	29.00	94.33	84.68
LSTM	25.36	23.23	74.51	69.52
TCN	**23.64**	**21.17**	**71.19**	**66.23**

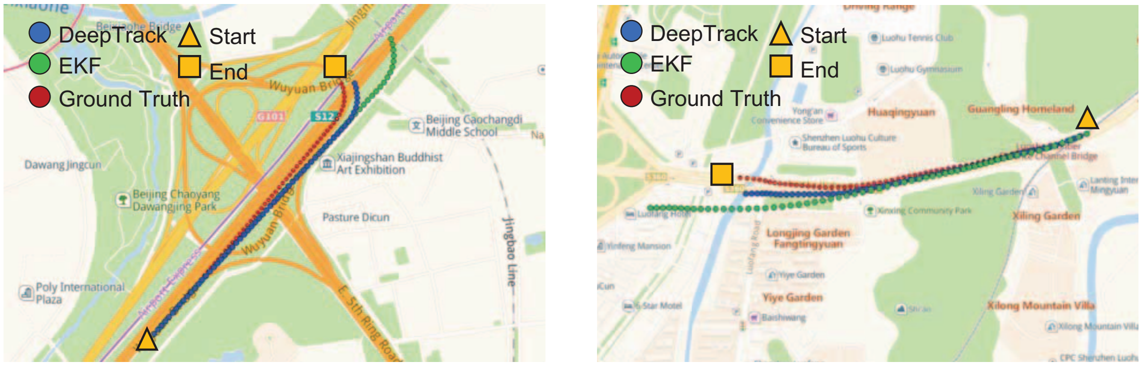
B. Inertial Sequence Learning

Table V depicts vehicle's location errors during 30-second and 60-second intervals. We use the MAE (Mean Absolute Error) calculated by ground truth and our predictions as the loss metric. Compared with EKF, GRU and LSTM, we observe that TCN achieves the least MAE values, both in short and long terms. Besides, our TCN-based inference model only has 1/3 parameters of GRU and LSTM, which is more suitable for running on smartphones. Figure 11 demonstrates the TCN's effectiveness in an overpass in Beijing and a tunnel in Shenzhen for real time tracking without GPS.

C. Accuracy of Personalized Model

We first evaluate the accuracy of the personal model during offline training. Then we leverage online data collected during vehicle driving and evaluate the accuracy improved by online training.

1) Effectiveness of Pose Estimation and Offline Training: We comprehensively compare the model performance between offline training model and other alternatives, based on two guidelines: (1) transforming coordinate with PCA-based pose estimation; (2) training personal model with offline private



(a) An overpass in Beijing

(b) A tunnel in Shenzhen

Fig. 11. Examples of real time tracking, with our prototype and ground truth.

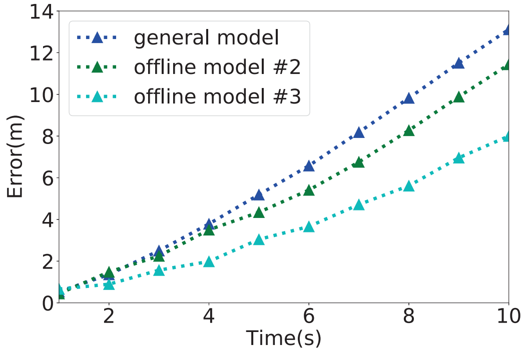


Fig. 12. Offline comparison.

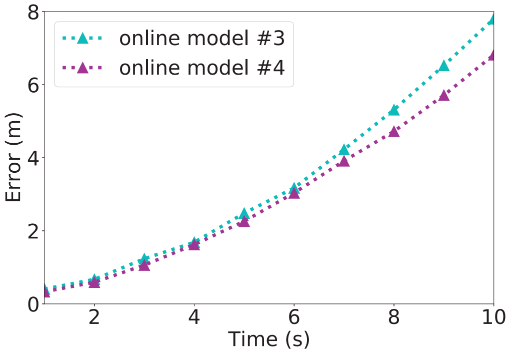


Fig. 13. Online comparison.

data in April or not. Finally, we have 4 alternatives for comparison, as shown in Table IV.

Figure 12 denotes the substantial improvement contributed by offline training. When we only leverage the offline data to retrain the general model (offline model #2), it can also reduce localization results of 10s inference from general's 13.114m into 11.437m (↓ 12.79%). If we combine the PCA's result and personalized retrain (offline model #3), it shows the reduction of errors into 8.0m (↓ 38.99%). This experimental result demonstrates that the combination of leveraging PCA-based transformation and offline private data training is the wise solution.

2) Effectiveness of Online Training: Offline training obtains a respectable precision offline model (*offline model #3*), i.e.,training and testing the model using the historical data in April as input. On contrary, online training updates the model by leveraging the latest trajectory of such driver as input data. Figure 13 shows the improvement. Compared to the inference result of offline training, the online training (*online model #4*) improves localization error again from offline model's 7.802m to 6.818m (↓ 12.61%). Although our inference model has been well personalized by offline training with historical data, it still have a improvement by the online data. This result inspires that we can continuously integrate the latest inertial data as training sets and update our inference model to reduce the localization error.

VI. Related work

A. Sequential Deep Learning

RNN is a classic way to model the sequence data. It is widely used in natural language processing [11], speech recognition [12], machine translation [13], stream data processing [14], [15], and traffic prediction [16]. LSTM [17] is a special RNN. It controls the transmission state through gates, which improves the effect on the long-term memory. GRU [18], a variant of LSTM, combines the forget gate and the input gate into a single update gate, and its model is simpler than the LSTM. Recently, CNN has achieved breakthroughs in learning sequence, with better performance than RNN in learning long-term sequences. As an unique CNN, the temporal convolutional network (TCN [19]) realizes the causal transmission of data by means of causal convolutions and expands the receptive field by dilated convolutions. Compared with RNN, it is more flexible and smaller for mobile devices.

B. Localization in GPS-blocked Environment

At present, location tracking is usually based on GPS. However, when driving into long tunnels and indoor scenes, the GPS signal is in a low intensity and can not provide accurate locations. The positioning errors will accumulate to extreme values. Ultrasonic wave [20], WiFi [21], Bluetooth [22] technologies are mainly used for indoor localization. However, these solutions highly depend on dedicated deployments in the environment. Users also need to carry special devices for sensing the environment, which constraints the wide deployment of indoor location based services.

C. Inertial Tracking

Inertial dead reckoning has been widely used for indoor tracking, e.g., step counting. Mapcraft [23] used floor plan to reduce the error of stride length and heading direction. Hilsen Beck *et al.* leveraged WiFi fingerprints [24] to improve the location accuracy. However, these studies aimed to track pedestrians instead of vehicles. Gao *et al.* proposed VeTrack [25] to track vehicles at low speeds in GPS-blocked parking lots. It required a large number of landmarks (such as bumps and turns) to calibrate the location, which is not appropriate for large-scale tracking. Since the phone's coordinate system is not always aligned with the vehicle, the inertial readings of smartphone cannot be directly used to infer vehicle's location. Building a posture relationship between smartphone and vehicle is necessary. Y. Awang [26] leveraged the embedded sensors to predict the smartphone's posture in car. R. Gao [25] compared of the shadow tracking method and 3D tracking method.

VII. Conclusion

In this paper, we conduct extensive experiments on crowdsourced traffic datasets to explore the key factors on vehicular inertial tracking. We summarize three technical challenges and propose a novel vehicle real time tracking solution. It consists of PCA-based coordinate transformation algorithm, TCN-based inertial sequence learning model, and customized re-training mechanism. The experimental results show that our approach can obviously improve the accuracy and robustness of vehicle tracking without GPS.

Acknowledgments

This work is supported in part by Beijing NSF L192004, NSFC 62072029, DiDi Research Collaboration Plan, and CCF-Tencent Open Fund.

References

[1] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings of IEEE INFOCOM*, vol. 2, 2000, pp. 775–784.

[2] X. Wang, L. Gao, S. Mao, and S. Pandey, "Csi-based fingerprinting for indoor localization: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, 2016.

[3] F. Zafari, I. Papapanagiotou, and K. Christidis, "Microlocation for internet-of-things-equipped smart buildings," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 96–112, 2015.

[4] iBeacon, https://developer.apple.com/ibeacon/ [Online; accessed July 27, 2021.

[5] P. Baronti, P. Pillai, V. W. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards," *Computer communications*, vol. 30, no. 7, pp. 1655–1695, 2007.

[6] S. Holm, "Hybrid ultrasound-rfid indoor positioning: Combining the best of both worlds," in *Proceedings of IEEE International Conference on RFID*, 2009, pp. 155–162.

[7] Decawave, "Real time location: An introduction," http://www.decawave.com/sites/default/files/resources/aps003_dw1000_rtls_introduction.pdf. [Online; accessed July 27, 2021.

[8] Y. Gao, Z. Yao, X. Cui, and M. Lu, "Analysing the orbit influence on multipath fading in global navigation satellite systems," *IET Radar, Sonar & Navigation*, vol. 8, no. 1, pp. 65–70, 2014.

[9] J. Collins and R. Langley, "Mitigating tropospheric propagation delay errors in precise airborne gps navigation," in *Proceedings of Position, Location and Navigation Symposium (PLANS)*, 1996, pp. 582–589.

[10] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep., 2007.

[11] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of ICML*, 2008, pp. 160–167.

[12] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of ICML*, vol. 32, 2014, pp. 1764–1772.

[13] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," *CoRR*, vol. abs/1511.01432, 2015. [Online]. Available: http://arxiv.org/abs/1511.01432

[14] T. Li, J. Tang, and J. Xu, "A predictive scheduling framework for fast and distributed stream data processing," in *Proceedings of IEEE International Conference on Big Data (Big Data)*, 2015, pp. 333–338.

[15] T. Li, Z. Xu, J. Tang, and Y. Wang, "Model-free control for distributed stream data processing using deep reinforcement learning," *Proceedings of the VLDB Endowment*, vol. 11, no. 6, pp. 705–718, 2018.

[16] R. Gao, X. Guo, F. Sun, L. Dai, J. Zhu, C. Hu, and H. Li, "Aggressive driving saves more time? multi-task learning for customized travel time estimation," in *Proceedings of IJCAI*, 2019, pp. 1689–1696.

[17] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.

[18] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014.

[19] S. B. J. Z. Kolter and V. Koltun, "An empirical evaluation of generic convolutional and recurrent network for sequence modeling," *CoRR*, vol. abs/1803.01271, 2018.

[20] F. Sato, Y. Motomura, C. Premachandra, and K. Kato, "Absolute positioning control of indoor flying robot using ultrasonic waves and verification system," in *Proceedings of International Conference on Control, Automation and Systems (ICCAS)*, 2016.

[21] S. Zhang, W. Wang, and T. Jiang, "Wi-fi-inertial indoor pose estimation for microaerial vehicles," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 5, pp. 4331–4340, 2021.

[22] S. Tomazic and I. Skrjanc, "An Automated Indoor Localization System for Online Bluetooth Signal Strength Modeling Using Visual-Inertial SLAM," *SENSORS*, vol. 21, no. 8, APR 2021.

[23] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Lightweight Map Matching for Indoor Localisation using Conditional Random Fields," in *Proceedings of IPSN*, 2014, pp. 131–142.

[24] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, and E. Steinbach, "Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning," in *Proceedings of UbiComp*, 2014, p. 147–158.

[25] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, and G. Luo, "Smartphone-based real time vehicle tracking in indoor parking structures," *IEEE Transactions on Mobile Computing*, vol. 16, no. 7, pp. 2023–2036, 2017.

[26] Y. Wang, J. Yang, H. Liu, Y. Chen, M. Gruteser, and R. P. Martin, "Sensing vehicle dynamics for determining driver phone use," in *Proceeding of ACM MobiSys*, 2013, pp. 41–54.