Proceedings of the 2022 Integrated Communication, Navigation and Surveillance Conference (ICNS),
5-7 April 2022, Dulles, USA
DOI:10.1109/ICNS54818.2022.9771517

Experimental Evaluation of GNSS and IMU Fusion Using Gated Recurrent Unit

Shuoyuan Xu, Ivan Petrunin, and Antonios Tsourdos, Cranfield University, United Kingdom

▪ Abstract

In this paper, a data-driven Inertial navigation systems (INS) and Global Navigation Satellite System (GNSS) fusion algorithm based on the use of the Gated Recurrent Unit (GRU) is proposed. In this project, we trained the GRU neural network with Inertial Measurement Unit (IMU) raw data and GNSS Position, Velocity and Timing (PVT) solutions as input and the position difference between GNSS and ground truth as labels. Therefore, the trained model can estimate the rover's positions by subtracting the predicted GNSS error from GNSS positions given IMU raw measurements and GNSS PVT solutions. To evaluate the performance of GNSS/INS fusion algorithms in realistic scenarios, we developed an experimental platform. Our experimental platform consists of a moving test rig and an external validation system. The moving test rig consists of a rover equipped with an LPMS-CU2: 9-Axis Inertial Measurement Unit (IMU) and U-Blox ZED-F9P GNSS receiver. For validation purposes, we employ an onboard real-time kinematic positioning (RTK)-GNSS receiver. The test scenarios include both open-sky and challenging conditions near buildings, which is beneficial for devolving and testing urban navigation systems. After training with collected experimental data in multiple test scenarios, the proposed algorithm is able to improve GNSS positioning accuracy by more than 60% for the open-sky environment and 30% for the urban environment.

▪ Introduction

Global Navigation Satellite Systems (GNSS) is one of the most important position, navigation, and timing (PNT) sources in various applications, such as Urban Air Mobility, Unmanned Aircraft System (UAS) Traffic Management, and Air Traffic Management. However, GNSS performance can be degraded when operating alone on high-speed platforms, in an urban environment, or applications with high accuracy requirements [1]. Inertial navigation systems (INS) can perform well in a short time range yet accumulates past errors, leading to performance that can deteriorate in the long range. Fusing Inertial Navigation Systems (INS) data with GNSS is a common way of partially mitigating the weakness of GNSS [2], [3].

Common fusion algorithms for INS and GNSS integration can be categorised by the utilised GNSS data: Uncoupled integration, Loosely coupled integration, Tightly coupled integration and Ultra-tight/deep integration. For uncoupled integration, INS bridged the data when GNSS is not available. However uncoupled integration doesn't mitigate the error accumulation issues of INS, thus not suitable for high accuracy navigation [4]. For loosely coupled integration [5], the GNSS receiver calculated the navigation solution (position and velocity) from the GNSS signal and combined it with the INS output. Tight and ultra-tight coupling [6], [7] use the GNSS raw measurement (pseudorange, pseudorange rate, tracking phase data, etc) and fused them with IMU data to obtain fused Position, Velocity and Timing (PVT). Even though tight and ultra-tight can provide better accuracy than loosely coupled, they involve the intermediate data of GNSS receivers (i.e., the digital tracking loops), which are not always accessible [8] from commercial products. Therefore, loosely coupled, as one of the most common fusion schemes for INS and GNSS integration, is adopted in this paper.

There are two major categories of loosely coupled algorithms: rule-based and data-driven approaches. For rule-based methods, filters are mostly applied, including Kalman filter [], Extended Kalman filter [9], Unscented Kalman filter [10], and particle filter [11]. Filtering algorithms require the knowledge of all measurement models and noise to perform GNSS/INS fusion. However, for real applications, this assumption tends to cause trouble. For instance, when multipath occurs neither the GNSS observation model nor the GNSS noise is capable of effectively modelling the measurement to state conversion. Moreover, the sensor measurement model and the noise term may not be directly available or contains errors when certain conditions are not considered (e.g. communication delay or packet loss). Instead, various tests and approximations are required. Therefore, we aim to develop a data-driven approach for fusing GNSS and Inertial Measurement Unit (IMU) data with the capabilities of handling a more complex environment and less demanding on modelling of the sensors.

There are several data-driven approaches for performing sensor fusion of GNSS and INS sensors. Recurrent Neural Network (RNN) [12] was first applied to GNSS/INS integrations. The issue with RNN is that it has difficulties solving problems that require learning long-term dependencies due to the gradient of the loss function decaying. [13] applied RNN and its variations long-short term memory network (LSTM) and gated recurrent unit (GRU) for INS navigation when GNSS fails at open sky environment. The results showed that Both LSTM and GRU outperforms RNN with less computational efficiency. Among them, GRU provides higher computational efficiency [14]. As for the urban environment, [15] used LSTM to estimate

This research is funded by European Space Agency under NAVISP Element 2 program with grant number 4000134037/21/NL/MP/mk

the satellite visibility, thus enabling a more accurate PVT result. The discussed approaches are mostly evaluated through simulated data which lacks the real world capability test. It can be identified that from the above literature, RNN based machine learning method is capable of processing IMU data, providing reliable PVT given time serial measurement data. Furthermore, since phenomenons like multipath are hard to model mathematically, we need real world data to validate our algorithm.

Inspired by the literature mentioned above, this paper aims to develop a machine learning based fusion algorithm that is capable of providing accurate positioning results given IMU and GNSS data even within an urban environment. To develop a practical algorithm, an experimental platform is developed with the capability of real-time data collection and validation purposes. These two are also the major contributions of this paper. The rest of the paper is organised as follows. Section II presents our proposed algorithm. Section III gives the architecture of our experimental platform and experiment setups. In Section IV, the performance analysis of the proposed algorithm is done via real-world experiments. Finally, some discussions and conclusions are provided.

▪ GRU based IMU-GNSS Fusion

▪ *GRU*

As discussed before, a machine learning approach are required for solving the fusion problem. Considering the nature of the real-time navigation application, the machine learning model needs to be not complicated and trained quickly considering that the model needs to run on a vehicle onboard computer. Also, since navigation is time-dependent, the machine learning model must simultaneously consider the relation between current and past data. Therefore, GRU, as an efficient variant of RNN, is adopted as the choice for this paper since it meets the listed requirements. The architecture of GRU is shown in Figure 1.

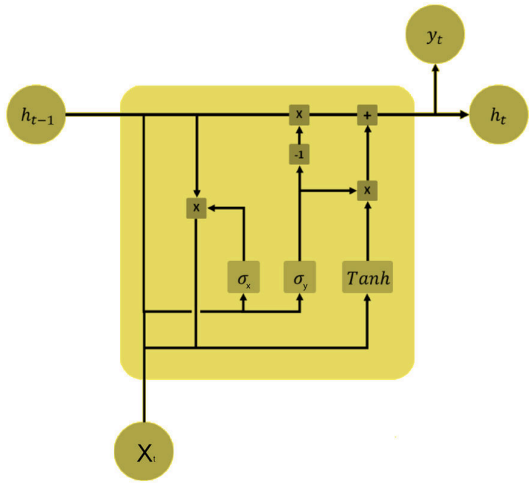


Figure 1: GRU architecture

The GRU model can be describe as the follow equations [13]:

$$\begin{aligned} z_t &= \sigma\left(W_z \cdot [h_{t-1}, X_t] + b_z\right) \\ r_t &= \sigma\left(W_r \cdot [h_{t-1}, X_t] + b_r\right) \\ \hat{h}_t &= \tanh\left(W \cdot [r_t \odot h_{t-1}, X_t] + b\right) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \\ y_t &= \sigma_y\left(W_y h_t + b_y\right) \end{aligned} \tag{1}$$

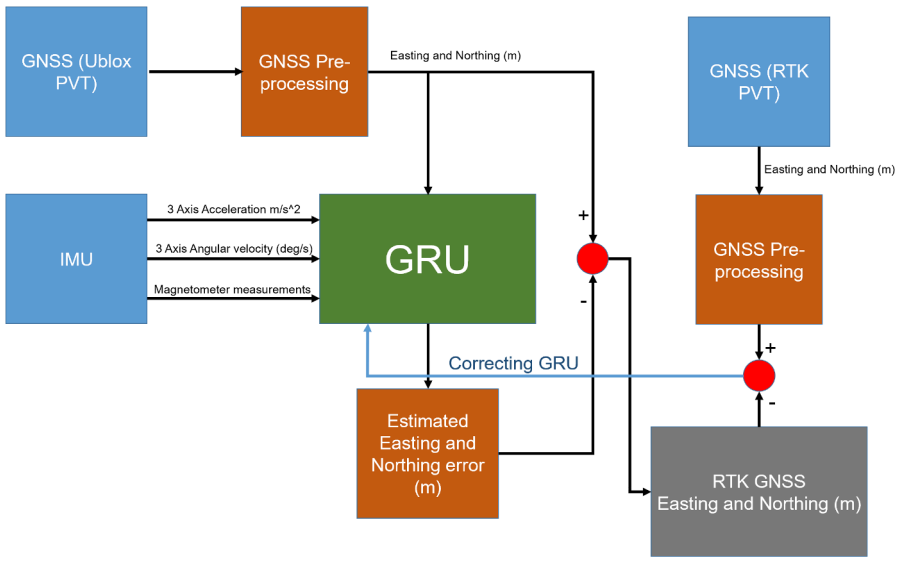
where W_z, W_r, W are the parameter matrices, b_z, b_r, b are the bias terms, h_t is the hidden layer vector, y_t is the output vector, X_t is the input vector, and σ_h, σ_y are the activation functions (Tanh). GRU takes the information from the previous time step h_{t-1} and multiply it with a weight matrix. The same is done for the new input X_t and is then combined with the previous state to create the new hidden state h_t. Because of the way GRU is structured, it allows for information in the past to be linked with the information at the current time step. Unlike other neural networks that links one set of inputs to one set outputs, RNN has "memory" to learn the relation between time steps. However, RNN has the limitation of not able to learn the long-term relationship since it only considers the gradient of the data between steps. GRU, as a recent variant of RNN, is capable of mitigating the RNN's issue of learning the long-term relationship by memorising long-term with three gates. GRU uses only one 'update' gate to decide how much information will be kept and what information will be updated and added into the states, which makes GRU one of the most efficient model among other RNN variants.

▪ *System Architecture*

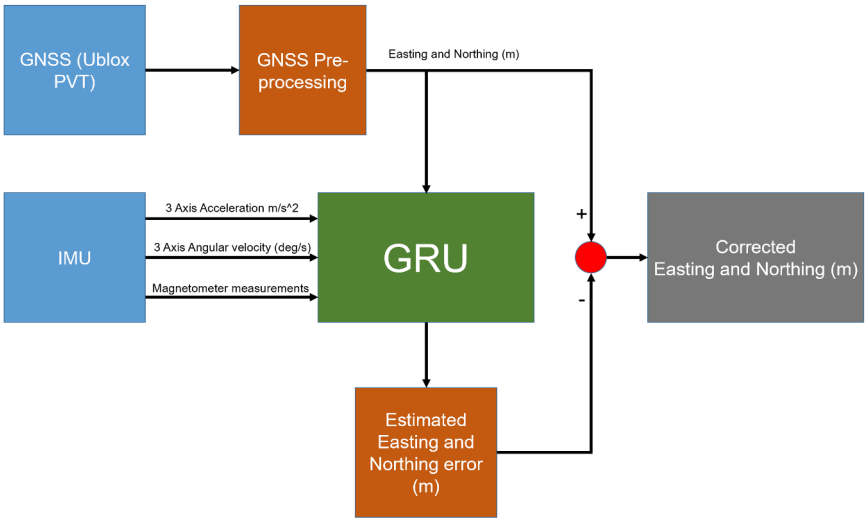
In this project, we used machine learning (ML) based fusion algorithms to improve the PVT accuracy of the GNSS receiver given IMU data. IMU can perform well in a short time range; however, since IMU accumulates past errors, the performance can deteriorate in the long range. On the other hand, GNSS can provide time-dependent navigation solutions while maintaining homogeneous accuracy. However, the data rate of GNSS receivers may not fulfil the requirement for some applications, and navigation solutions may not be available in challenging environments. Combining the advantages of INS and GNSS can allow systems to produce high data rate solutions with interpolation and bridge the GNSS outage.

The architecture of the proposed fusion algorithm is shown in Figure 2. The proposed algorithm is based on the idea of using a GRU neural network to estimate the GNSS PVT error with IMU and GNSS measurements as input. Our algorithms contains two modes: training and testing. Both the training and testing data set are generated from real experimental data using our developed test rig.

The output from IMU contains the raw measurement of 3 directional acceleration, angular rate, and magnetometer reading. As for the GNSS, it contains latitude, longitude, and height. It is known that GNSS is not capable of



(a) Training



(b) Testing

Figure 2: Fusion algorithm for training and testing

providing accurate height measurement since all visible satellites are on the same side of the earth as the receiver. Therefore, we are not using height as a data source. It is worth noting that we didn't include any processed GNSS data from the receiver such as velocity and angular rate to avoid introducing further uncertainties from the receiver.

The GRU model estimates the GNSS PVT error given IMU and GNSS measurements as input. By subtracting the estimated GNSS PVT error given from the GNSS PVT solution, the fused PVT can be obtained. For training, we use the IMU raw measurement and GNSS PVT solutions as input and ground truth trajectory to train the model. The collection of ground truth in this paper will be further discussed in the next section. The parameters required to conduct the training of the proposed deep RNN are: maximum number of epochs is 200, number of GRU units is 500, steps per epoch set to 100, dropout set to 0.4, sigmoid chosen as the activation function, and Adam as the optimiser. The training process is to minimise the difference between the estimated GNSS error and the real GNSS error computed by subtracting the GNSS result from the ground truth position of the receiver. To measure the progress of the training, estimated positioning errors are compared to the ground truth to compute the mean squared error. Once low enough mean squared error is achieved, the system is well trained and can be applied in real applications.

▪ Experimental Setup

To investigate the performance of our proposed algorithm in real applications, we developed a test rig and carefully designed a series of experiments around it. In this section, the proposed experimental platform and corresponding setups are presented. Then general architecture

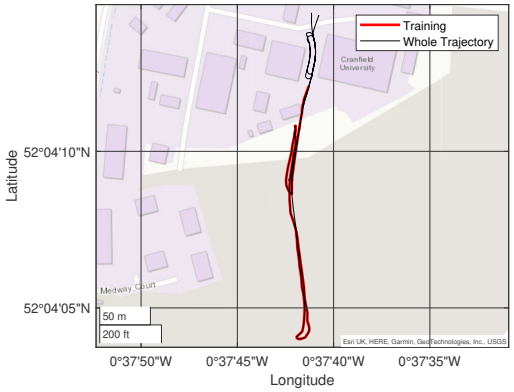
and each component of the test rig are presented. Finally, the experiment setup and trajectory design are discussed.

▪ *System Architecture*

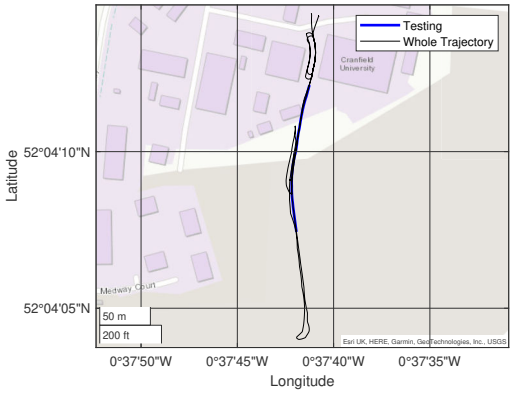
The flowchart of the planned hardware configuration for the mobile test is presented in Figure 5. All components are installed on the rover. Here, the same receiver with a network-based RTK service is applied as the validation source since it is able to achieve centimeter level accuracy in real time which enables point to point validation. The antenna is connected to a signal splitter to ensure that the signal feed into both the validation receiver and experimental receiver are identical. An IMU module is also connected to the computational unit, a ThinkCentre M90n-1 (I5-8265u, 8G RAM, 256G storage) mini desktop for providing IMU input. After the fusion is done, the fused result and corresponding validation trajectory can be obtained.

Figure 3: Test rig illustration

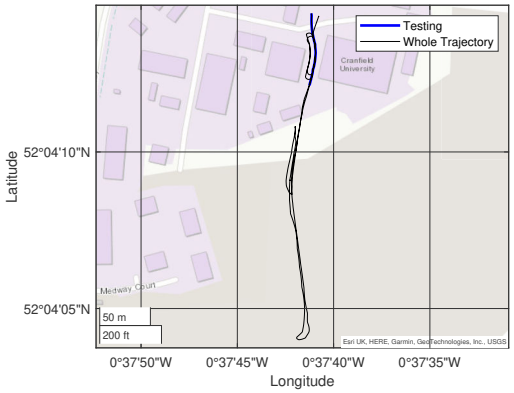
The selected IMU is powered and accessed through USB. Furthermore, the sensor fusion algorithm is set to be integrated with the GNSS receiver data, processed on the ThinkCentre computer. Detailed explanations on the fusion module will be presented in later sections. The control centre of the rover will be a Raspberry Pi 4B to receive commands, communicate navigation information, and command the microcontroller to control the actuators. The additional sensor module for fusion application is selected as the LPMS-CU2 from Omni-instrument. LPMS-CU2 is a 9-Axis inertial measurement unit (IMU) with USB connectivity. The integrated component of LPMS-CU2 includes a 3-axes accelerometer, a 3-axes gyroscope, a 3-axes magnetometer, and a barometer. The data output can be either raw data from each sensor or quaternion with an update rate of 400 Hz. Considering the vehicle's maximum speed, the overall sensitivity and update rate of the LPMS-CU2 is more than enough for this project.



(a) Training Section



(b) Testing with open sky data



(c) Testing with urban data

Figure 4: Trajectory for training and testing

The LeoRover platform is selected for its development flexibility and high payload in this project. The LeoRover has a dimension of 447x433x249mm and a payload capacity of 5kg, suitable for loading our devices. The total weight of the GNSS Receiver devices is estimated to be around 3kg - with batteries, GNSS Receiver, and a mini-PC. The surface dimension of the LeoRover is only going to be enough for stacking up all the devices. The largest among all devices of GNSS Receiver is the battery, with a 220x150x20 mm. The average run time of LeoRover is estimated to be 1 hour with added weight using its 11.1

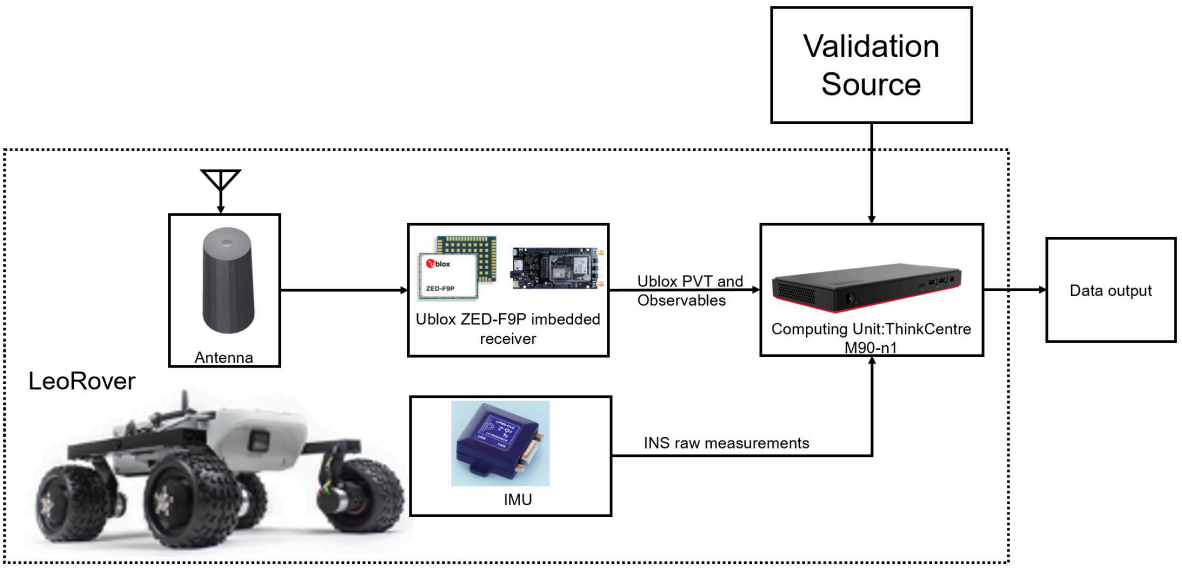


Figure 5: Architecture of the testing rig

V, 5000 mAh Li-Ion battery.

▪ *Trajectory Design*

In this project, we plan to drive the developed rover platform on an existing smart road of Cranfield University, the Multi-User Environment for Autonomous Vehicle Innovation (MUEAVI). MUEAVI is a purpose-built experimental facility for the rapid development of on- and off-highway, ground and airborne autonomous solutions. This includes vehicles, infrastructure, data, logistics, environment, sensors and their implementation and management. We choose MUEAVI as our experimental environment for it contains both an urban section and an open-sky section - and is instrumented with the sensors that can be utilised for future performance validation independently of GNSS.

As shown in Figure 6 (a-1), the region on the north region of MUEAVI is between two buildings and is considered an urban canyon. The positioning output of the UBlox receiver in this region shows significant deviates to the real trajectory, which indicates that multipath happens within this section of the road. Apart from covering both the scenarios, we need to understand whether entering or exiting an urban canyon would have impacts on the PVT result of the receiver and our fusion algorithm. Furthermore, inertial sensors are well-known to have good performance when going straight lines with relatively slow speed, yet shows poorly performance with turns or inconsistent speed. We don't want to exclude such uncertainties in our system even though the selected rover can only provide relatively low speed. Taking these factors into consideration, we want not only the rover's trajectory to cover the selected section of MUEAVI multiple times, but we also want it to have enough turns and repeating trajectories. With all these considered, we planned to have two full circles in the urban region (Figure 6 (a-1) to (a-3)), two cycles with the open section (Figure 6 (a-4) to (a-6)), then all the way back to the entrance (Figure 6 (a-7) to (a-8)).

▪ Results and Discussion

In this paper, we use the starting 80% of the open sky section of the trajectory as the training data, the later trajectory within open sky or urban environment as the testing data. Detailed sections for training and testing are plotted as Figure 4.

▪ *Open Sky Results*

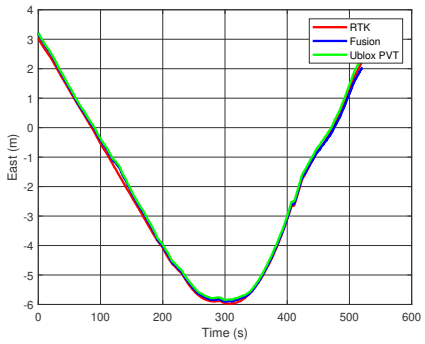
In Figure 7, the fusion result under opens sky environment is compared with the RTK GNSS generated trajectory in the North(N) and East(E) directions. The fused results are obtained by taking the U-blox PVT measurement and subtracting the estimated GNSS error. Comparing the fused result to the U-blox PVT and RTK result Figure 7-(c), visually it can be seen that the proposed fusion algorithm is capable of reducing the GNSS PVT errors. By comparing the root-mean-squared error (RSME) of the fused result to U-blox PVT, it can be identified that our fusion algorithm is capable of reducing the average error by more than 75%, the RSME reduces from 0.98m to 0.23m. From the trajectory comparison Figure 7-(a and b) and RMSE difference comparison Figure 7-(c), the majority of the errors generated by the U-blox PVT are mitigated by the GRU architecture. However, there are still some sections that the GRU struggles to provide reliable improvements, for instance, trajectory near the $400s$ time step when approaching a building. This may indicate that more training data is required to improve the performance of the GRU, which can be potentially solved by training the GRU on multiple routes. In Figure 7-(b), a comparison between the RMSE of the GRU and U-blox PVT is made. From this graph, it can be identified that the

Figure 6: Illustration of the designed trajectory

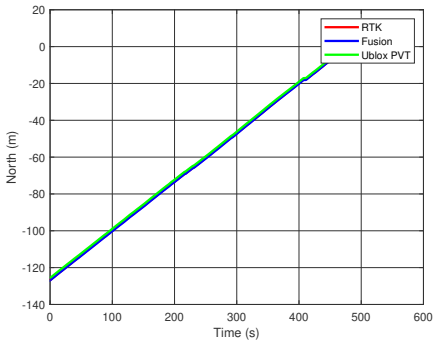
proposed method reduces the position error evenly instead of reducing drastically on very few specific points.

▪ *Urban Results*

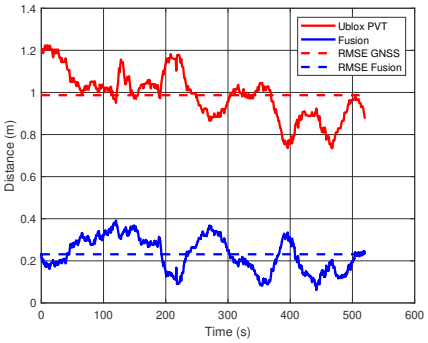
In Figure 8, the comparison of fused, RTK, and U-blox PVT of the urban trajectory are compared in the North(N) and East(E) directions. Compared to open sky data, the fused result of urban result improves the GNSS accuracy to 30% (Figure 8-(c)), from $1.3m$ to $0.96m$. By comparing the North and East plot, it can be identified that our proposed GRU model addresses mostly the bias of UBlox PVT. From East estimation, the fusion result mostly performs a bias removal while keeping the overall shape of the trajectory unchanged. This indicates that the proposed fusion algorithm is not mitigating GNSS errors evenly. The issue with the urban performance comes mainly with the fact that the multipath effect is not well represented in the proposed model. Improving the performance of our model in urban environment will be the main focus of our future study.



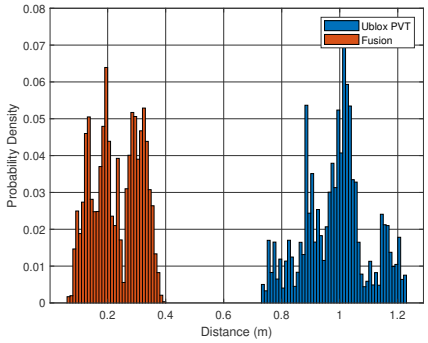
(a) East component of the rover position



(b) North component of the rover position

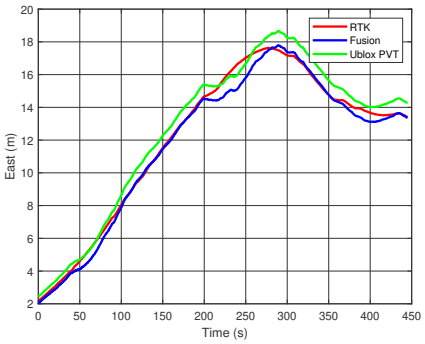


(c) Mean Squared distance

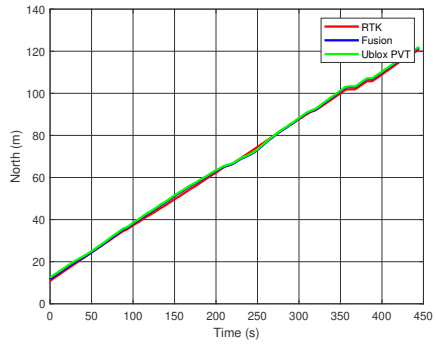


(d) Histogram of distance comparing to RTK

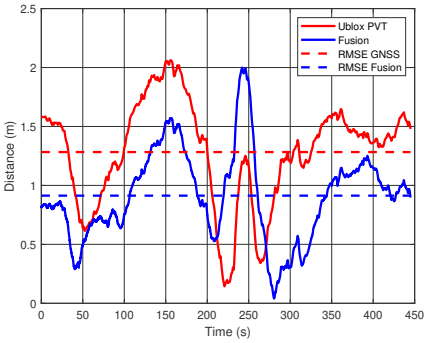
Figure 7: Fusion result on open sky data



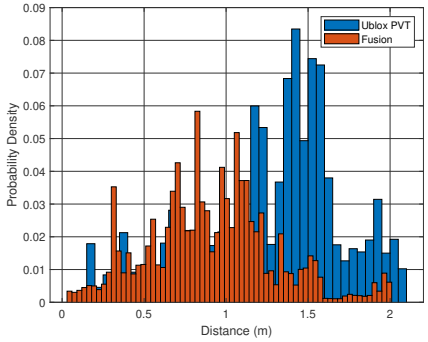
(a) East component of the rover position



(b) North component of the rover position



(c) Absolute distance



(d) Histogram of distance comparing to RTK

Figure 8: Fusion result on urban data

▪ Conclusion

In this paper, we developed a machine learning based IMU-GNSS fusion algorithm and an experimental platform to validate the proposed algorithm. The proposed fusion algorithm is based on the GRU neural network with IMU raw measurement and GNSS PVT as input, and GNSS PVT error as output. Therefore, the trained model can estimate the rover's positions by subtracting the predicted GNSS error from GNSS PVT solutions. The developed experimental platform is capable of providing synchronised data collection of sensor data, valid reference data, and corresponding time stamps. Moreover, the developed platform has versatile extendabilities which enables it to be applied to other types of trials with insignificant modifications. From the experimental result, our proposed algorithm is capable of providing state-of-the-art fusion results in an open-sky environment with an average of less than $0.3m$ average RMSE. As for the urban canyon, our model is able to reduce the average RMSE from $1.3m$ to $0.96m$. Future studies will focus on deriving methods on improving the GNSS/INS fusion performance in urban environment.

References

[1] A. Angrisano, et al., Gnss/ins integration methods, Dottorato di ricerca (PhD) in Scienze Geodetiche e Topografiche Thesis, Universita'degli Studi di Napoli PARTHENOPE, Naple 21 (2010).

[2] M. G. Petovello, Real-time integration of a tactical-grade IMU and GPS for high-accuracy positioning and navigation, Citeseer, 2003.

[3] A. Solimeno, Low-cost ins/gps data fusion with extended kalman filter for airborne applications, Masters of Science, Universidade Technica de Lisboa (2007).

[4] C. Jekeli, Inertial navigation systems with geodetic applications, de Gruyter, 2012.

[5] P. Aggarwal, MEMS-based integrated navigation, Artech House, 2010.

[6] W. Jiang, D. Liu, B. Cai, C. Rizos, J. Wang, W. Shangguan, A fault-tolerant tightly coupled gnss/ins/ovs integration vehicle navigation system based on an fdp algorithm, IEEE Transactions on Vehicular Technology 68 (7) (2019) 6365–6378.

[7] A. Noureldin, T. B. Karamat, J. Georgy, Fundamentals of inertial navigation, satellite-based positioning and their integration (2013).

[8] G. Falco, M. Pini, G. Marucco, Loose and tight gnss/ins integrations: Comparison of performance assessed in real urban scenarios, Sensors 17 (2) (2017) 255.

[9] W. Wen, Y. C. Kan, L.-T. Hsu, Performance comparison of gnss/ins integrations based on ekf and factor graph optimization, in: Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019), 2019, pp. 3019–3032.

[10] B. Gao, G. Hu, Y. Zhong, X. Zhu, Cubature kalman filter with both adaptability and robustness for tightly-coupled gnss/ins integration, IEEE Sensors Journal 21 (13) (2021) 14997–15011.

[11] O. Vouch, A. Minetto, G. Falco, F. Dovis, On the adaptivity of unscented particle filter for gnss/ins tightly-integrated navigation unit in urban environment, IEEE Access 9 (2021) 144157–144170.

[12] A. Noureldin, A. El-Shafie, M. Bayoumi, Gps/ins integration utilizing dynamic neural networks for vehicular navigation, Information fusion 12 (1) (2011) 48–57.

[13] P. Geragersian, I. Petrunin, W. Guo, R. Grech, An ins/gnss fusion architecture in gnss denied environment using gated recurrent unit, in: AIAA SCITECH 2022 Forum, 2022, p. 1759.

[14] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv:1412.3555 (2014).

[15] G. Zhang, P. Xu, H. Xu, L.-T. Hsu, Prediction on the urban gnss measurement uncertainty based on deep learning networks with long short-term memory, IEEE Sensors Journal 21 (18) (2021) 20563–20577.

2022-05-12

Experimental evaluation of GNSS and IMU fusion using gated recurrent unit

Xu, Shuoyuan

IEEE

Xu S, Petrunin I, Tsourdos A. (2022) Experimental evaluation of GNSS and IMU fusion using gated recurrent unit. In: 2022 Integrated Communication, Navigation and Surveillance Conference (ICNS), 5-7 April 2022, Dulles, USA

https://doi.org/10.1109/ICNS54818.2022.9771517