Artificial Intelligence - BLG521E

Project Report

Assoc. Prof. Dr. Sanem Sarıel

Group 15

Ahmet Talha Cetin[*] **- 511211106**

Arslan Artykov[†] **- 511201110**

[*]cetinah16@itu.edu.tr

[†]artykov17@itu.edu.tr

1 Introduction

Inertial Measurement Unit (IMU) is widely used in many applications, from smartphones to safety critical applications, like aircraft navigation. These sensors contain three-axis gyroscope and three-axis accelerometer, which measure three dimensional angular velocity and linear acceleration, respectively. Unlike other sensors, such as a camera, IMUs are not easily affected by environmental issues, such as occlusion or brightness change. Therefore, they are usually preferred as complementary sensor with other sensors. On the other hand, IMU measurements are usually noisy and biased, which restrict their stand alone usage.

There are a lot of methods to process the noisy IMU measurements. Some of these methods are data filtering, intrinsic model compensation, and probabilistic sensor fusion [5]. Although these methods were exhaustively studied in literature, they are still prone to errors due to the IMU modelling errors [5], [8]. The paper, which we implemented in the context of this project, proposes a DNN-based, particularly, Recurrent Neural Network (RNN) based IMU data filtering approach [1]. One of the significance of the proposed method is that, it regresses integrated motion terms instead of relative IMU poses. Finally, the authors of the paper assert that their method outperforms other IMU data filtering methods, such as traditional low-pass filter.

2 Related Works

One of the main reasons of widely usage of IMUs is their low cost. However, low-cost IMU sensors are prone to various error sources, such as mechanical, thermal and electronic. Traditional, non-statistical IMU filtering methods are reducing high frequency elements [8], auto regressing and moving average methods [13]. Besides noises, IMUs are also prone to biases, which are usually modeled by additive terms into IMU measurement equations [9]. Although it is enough to model noises and biases for expensive IMUs to get accurate measurements [5], one needs to consider other factors when it comes to low-cost sensors. Some of these factors are misalignment parameters, G-sensitivity matrix and thermal effects [10], [14], [15]. Considering all these factors in closed-form mathematical equations is infeasible and almost impossible due to inner complexities of low-cost IMUs. Therefore, modelling low-cost IMUs with data driven methods has turned out to be a popular trend in the literature. After the proven success of the Recurrent Neural Networks (RNN) in other fields, researchers have attempted to utilize RNNs in IMU pose integration. Traditional approaches take raw measurements as input and regress integrated poses with ground-truth ones, which leads to accurate predictions only under certain conditions [7], [10], [11], [12].

In spite of proper modelling, there will be some noise and bias in the measurements, which will lead to accumulated drift in integrated poses. Therefore, IMUs are used as either complementary sensors in navigation systems [2], [3], [5], [16] or motion dynamics of the used platform is included in order to reduce the drift [6], [7], [17]. On top of this, in some works, IMUs are used as complementary sensors along with motion constraints [4], [18].

In this paper, an LSTM-based IMU filtering method was proposed, which regresses integrated motion terms instead of relative pose measurements. According to the authors, cleaned IMU measurements can directly be used in inertial navigation systems [1].

3 IMU Models

3.1 IMU Measurement Equations

It is assumed that an IMU at frame I, moves with respect to a global frame G can be modelled as

$$\left[\boldsymbol{\omega}_m^T, \boldsymbol{a}_m^T\right]^T = g_\pi\left(\boldsymbol{\omega}_{rm}, \boldsymbol{a}_{rm}, \boldsymbol{\pi}\right)$$

where ω_{rm} and a_{rm} are raw gyroscope and accelerometer data, π is the model parameter vector, g_π is the pre-processing model. Preprocessing models filters noise and bias terms. In this project it is aimed to implement a pre-processing model g_π and train model parameters with recurrent neural network structure.

3.2 IMU Pose Equations

IMU state variables selected as,

$$\boldsymbol{x} = \left[{}_I^G\overline{\boldsymbol{q}}^\top, \boldsymbol{b}_g^\top, {}^G\boldsymbol{v}_I{}^\top, \boldsymbol{b}_a^\top, {}^G\boldsymbol{p}_I{}^\top\right]^\top \in \mathbb{R}^{16}$$

where ${}_I^G\overline{\boldsymbol{q}}^\top$ presents rotation from IMU frame to global frame in quaternion, and $\boldsymbol{p}_I{}^\top$ and ${}^G\boldsymbol{v}_I{}^\top$ stand for the IMU's position and velocity.
${}_I^G\overline{\boldsymbol{q}}\left(t_k\right), {}^Gp_I\left(t_k\right)$, and ${}^Gv_I\left(t_k\right)$ IMU's rotation, position and velocity at time k. IMU's kinematic equations are:

$${}_I^G\overline{\boldsymbol{q}}\left(t_{k+1}\right) = {}_I^G\overline{\boldsymbol{q}}\left(t_k\right)\Delta\overline{\boldsymbol{q}}$$

$$
\begin{aligned}
{}^{G}\boldsymbol{v}_I\left(t_{k+1}\right) &= {}^{G}\boldsymbol{v}_I\left(t_k\right) + \int_{t_k}^{t_{k+1}} {}^{G}\boldsymbol{a}_I(\tau) d\tau \\
&= {}^{G}\boldsymbol{v}_I\left(t_k\right) + {}^{G}\boldsymbol{g}\Delta t + {}_{I}^{G}\boldsymbol{R}\left(t_k\right)_{I(t_k)}^{I(t_{k+1})}\boldsymbol{\beta}
\end{aligned}
$$

$$
\begin{aligned}
{}^{G}p_I\left(t_{k+1}\right) =& {}^{G}p_I\left(t_k\right) + {}^{G}v_I\left(t_k\right)\Delta t + \int_{t_k}^{t_{k+1}}\int_{t_k}^{\tau} {}^{G}\boldsymbol{a}_I(s) ds d\tau \\
=& {}^{G}p_I\left(t_k\right) + {}^{G}v_I\left(t_k\right)\Delta t + \frac{1}{2}{}^{G}g(\Delta t)^2 \\
&+ {}_{I}^{G}\boldsymbol{R}\left(t_k\right)_{I(t_k)}^{I(t_{k+1})}\gamma
\end{aligned}
$$

where $\Delta t = t_{k+1} - t_k$ and

$$
\begin{matrix} I\left(t_{k+1}\right) \\ I\left(t_k\right) \end{matrix} \boldsymbol{\beta} = \int_{t_k}^{t_{k+1}} I\left(t_k\right)\boldsymbol{R}\cdot{}^{I(\tau)I(\tau)}\boldsymbol{a} d\tau
$$

$$
\begin{matrix} I\left(t_{k+1}\right) \\ I\left(t_k\right) \end{matrix} \boldsymbol{\gamma} = \int_{t_k}^{t_{k+1}}\int_{t_k}^{\tau} I(s)^{I(t_k)}\boldsymbol{R}\cdot{}^{I(s)}\boldsymbol{a} ds d\tau
$$

It is also defined that $\Delta\overline{\boldsymbol{q}} = {}_{I(t_k)}^{I(t_{k+1})}\overline{\boldsymbol{q}}, \Delta\boldsymbol{\beta} = {}_{I(t_k)}^{I(t_{k+1})}\boldsymbol{\beta}, \Delta\boldsymbol{\gamma} = {}_{I(t_k)}^{I(t_{k+1})}\boldsymbol{\gamma}$. These Δ terms can be expressed only inertial linear and angular accelerations. These terms can be calculated during $t_i \leq t \leq t_j$ interval by following formulas.Instead of quaternions rotation matrices are used for expressing difference between orientations. Ground truth formulations for these learnable terms calculated as below.

$$
\Delta \mathrm{R}_{ij} \doteq \mathrm{R}_i^{\top}\mathrm{R}_j
$$

$$
\Delta\boldsymbol{\beta} \doteq \mathrm{R}_i^{\top}\left(\mathbf{v}_j - \mathbf{v}_i - \mathrm{g}\Delta t_{ij}\right)
$$

$$
\Delta\gamma \doteq \mathrm{R}_i^{\top}\left(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\sum_{k=i}^{j-1}\mathbf{g}\Delta t^2\right)
$$

4 Learning Method and Implementation Details

4.1 Network Architecture

The architecture of the proposed method is shown in 1 As it is seen on the figure, the network consists of two main blocks, namely, the DNN (Deep Neural Network) block and the integration block. The network accepts raw IMU data as input and propagates it through deep neural network, which consists of recurrent neural network and fully-connected layers stacked one after another. Bi-directional LSTM (Long-Short Term Memory) was proposed as the recurrent branch. At the end of the fully-connected layer, cleaned IMU data is obtained, which is directly sent into the integration part. This part does not contain any learnable parameters and it only integrates the cleaned inertial data. During inference one can take only the output of RNN part and utilize for a sensor fusion task.

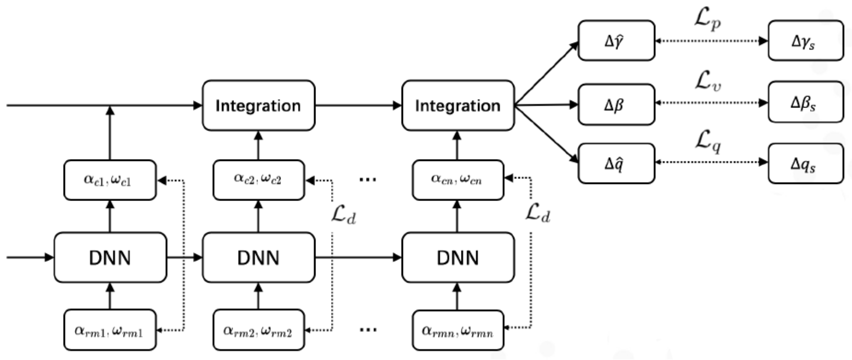

Figure 1: The network architecture [1]

4.2 Loss Functions

The ideal way of training this kind of models is training with perfect ground-truth inertial data, which are angular velocity and linear acceleration. Unfortunately, it is almost impossible to get such data. Therefore, the authors proposed to calculate the integration terms from ground-truth pose and use them in the loss function, which is formulated as follows,

$$\begin{aligned}\mathcal{L}_q &= \left|\log\left(\Delta\boldsymbol{q}_s \otimes \Delta\hat{\boldsymbol{q}}^{-1} \otimes \boldsymbol{q}_b^{-1}\right)\right|_h \\ \mathcal{L}_v &= \left|\Delta\boldsymbol{\beta}_s - \Delta\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_b\right|_h \\ \mathcal{L}_p &= \left|\Delta\boldsymbol{\gamma}_s - \Delta\hat{\boldsymbol{\gamma}} - \hat{\boldsymbol{\gamma}}_b\right|_h\end{aligned}$$

where variables with hat operator indicates the DNN output and the ones with s subscript show the ground-truth values. At the loss functions, Δq is quaternion integration term, $\Delta\beta$ and $\Delta\gamma$ are learnable integration terms. Also, $\otimes$ indicates the quaternion product [1]. $\hat{\boldsymbol{\gamma}}_b, \hat{\boldsymbol{\beta}}_b, \hat{\boldsymbol{q}}_b$ are the biases that coming from added noise to the IMU signals.
In addition to these losses, regularization losses are added to the framework as proposed. Regularization loss given as

$$\mathcal{L}_d = \max\left(|\boldsymbol{u}_m - \hat{\boldsymbol{u}}| - \lambda, 0\right)$$

Hyperparameters	Value
# of Hidden Layers	1
# of Neurons at Hidden Layer	128
# of Epochs	50
Learning Rate	10^{-3}
Training Sequence Length(s)	0.01
Batch Size	32
Regularization Param. (λ)	100
Accelerometer Noise	$\mathcal{N}(\mu = 0, \sigma = 0.005)$
Gyro Noise	$\mathcal{N}(\mu = 0, \sigma = 0.001)$

Table 1: Hyperparameters

where $|\boldsymbol{u}_m - \hat{\boldsymbol{u}}|$ is the vector norm of difference between raw measurent $\boldsymbol{u}_m$ and LSTM outputs $\hat{\boldsymbol{u}}$. λ is the control variable that controls the magnitude difference between raw measurements and LSTM outputs.

4.3 Training

The paper is implemented with PyTorch framework. The authors suggest to use ADAM optimizer with 0.0001 learning rate. The batch size was chosen as 32 in the paper [1]. In the paper framework trained for 700 epochs. However, in our setup there are less datapoints, therefore, model converges much faster. Models are trained for 50 epochs. Hyperparameters for the training can be seen in the Table 1.

The proposed method was tested on three platforms: EuRoC dataset, KAIST dataset and Ground robots. In our implementation, we trained and tested our models on EuRoC and Blackbird datasets, which has comparably noisier inertial measurements.
Extra noises are added to the IMU measurements
For EuRoC dataset, MH_01,MH_03, MH_05, V1_02, V2_01 ,V2_03 sequences are used for train dataset. For the validation MH_02, MH_04, V1_03, V2_02, V1_01 sequences are used as proposed by the paper references.[19]

Sequence	Position Error RMSE (m)	Velocity Error RMSE (m/s)	Angle Error RMSE (rad)
MH_02	0.0991	0.07268	0.04086
MH_04	0.0995	0.07881	0.04067
V1_03	0.0991	0.08559	0.05908
V2_02	0.0992	0.09283	0.05534
V1_01	0.0991	0.14640	0.04269

Table 2: EuRoC Validation Results

5 Results

5.1 Tests on EuRoC Dataset

Training and validation is completed around 60 minutes for EuRoC Dataset. Train and validation losses can be seen at the Figure 2

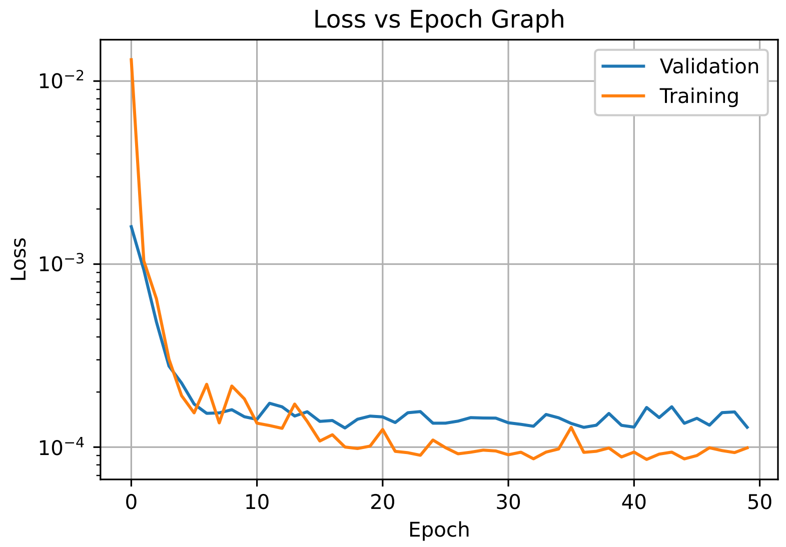

Figure 2: EuRoC Dataset Train-Validation Losses

For each validation sequence RMSE values can be seen at the Table 2.

5.2 Tests on Blackbird Dataset

Training and validation is completed around 80 minutes for Blackbird Dataset. Train and validation losses can be seen at the Figure 3

Sequence	Position Error RMSE (m)	Velocity Error RMSE (m/s)	Angle Error RMSE (rad)
clover_1p0	0.2350	0.2235	0.31119
dice_2p0	0.19580	0.2713	0.31154
oval_1p0_O	0.19759	0.24000	0.31289

Table 3: Blackbird Validation Results

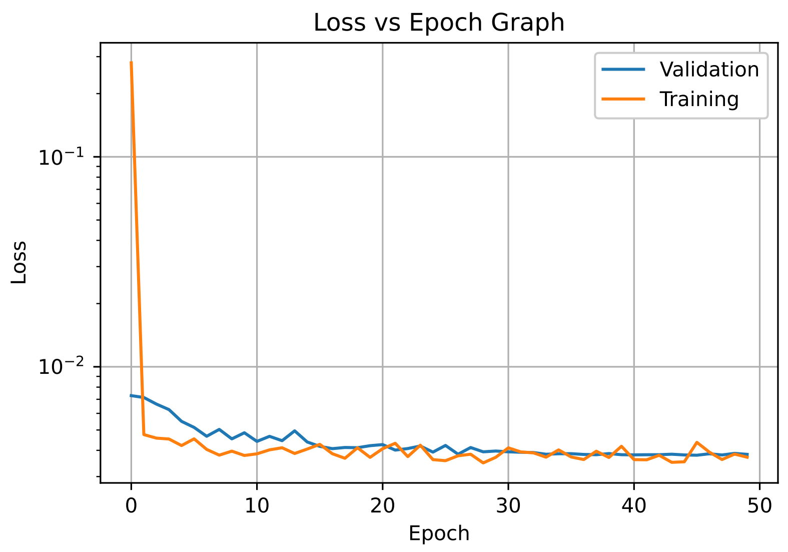

Figure 3: Blackbird Dataset Train-Validation Losses

For each validation sequence RMSE values can be seen at the Table 3.

5.3 Comparison with Original Paper

In the paper, method compared with 5 different existing methods. Two of which contain sensor fusion with visual sensors and three of them only based on IMU measurements. Root Means Squared Error (RMSE) between ground truth and prediction was used as comparison metric.

Sequence	Traditional Methods	Original Paper	Our Implementation
MH_02	0.185	0.150	0.0991
MH_04	0.151	0.138	0.0995
V1_03	0.267	0.147	0.0991
V2_02	0.111	0.104	0.0992
V1_01	0.081	0.066	0.0991

Table 4: Positional RMSE (m) Over The Entire Trajectory

Sequence	Traditional Methods	Original Paper	Our Implementation
MH_02	0.0093	0.0025	0.04086
MH_04	0.0103	0.0037	0.04067
V1_03	0.0168	0.0067	0.05908
V2_02	0.0130	0.0068	0.05534
V1_01	0.0234	0.0162	0.04269

Table 5: Angle RMSE (m) Over The Entire Trajectory

At the Table 4 position error comparisons can be seen.
At the Table 5 angle error comparisons can be seen.

Note: All the developments are made in Google Colab. Jupyter notebooks are given in the zip file. However, if results are wanted to be duplicated Google Account and password that we created for this project is given below. All the data and codes are given there.

6 Bibliography

[**1**] Zhang, M., Zhang, M., Chen, Y., Li, M. (2021). IMU data processing for inertial aided navigation: A recurrent neural network based approach. arXiv preprint arXiv:2103.14286.

[**2**] Li, M., Mourikis, A. I. (2013). High-precision, consistent EKF-based visual-inertial odometry. The International Journal of Robotics Research, 32(6), 690-711.

[**3**] Zhang, M., Zuo, X., Chen, Y., Liu, Y., Li, M. (2021). Pose Estimation for Ground Robots: On Manifold Representation, Integration, Reparameterization, and Optimization. IEEE Transactions on Robotics.

[**4**] Nisar, B., Foehn, P., Falanga, D., Scaramuzza, D. (2019). Vimo: Simultaneous visual inertial model-based odometry and force estimation. IEEE Robotics and Automation Letters, 4(3), 2785-2792.

[**5**] Farrell, J. A. (2016). Aided Navigation: GPS with High Rate Sensors Errata list.

[**6**] Abdulrahim, K., Moore, T., Hide, C., Hill, C. (2014). Understanding the performance of zero velocity updates in MEMS-based pedestrian navigation. International Journal of Advancements in Technology, 5(2).

[**7**] Wagstaff, B., Kelly, J. (2018, September). LSTM-based zero-velocity detection for robust inertial navigation. In 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN) (pp. 1-8). IEEE.

[**8**] Kang, C. H., Kim, S. Y., Park, C. G. (2011). Improvement of a low cost MEMS inertial-GPS integrated system using wavelet denoising techniques. International Journal of Aeronautical and Space Sciences, 12(4), 371-378.

[**9**] N. Trawny and S. I. Roumeliotis, "Indirect kalman filter for 3d pose estimation," University of Minnesota, Dept. of Comp. Sci. Eng., Tech. Rep, vol. 2, 2005.

[**10**] Li, M., Yu, H., Zheng, X., Mourikis, A. I. (2014, May). High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation. In 2014 IEEE International Conference on Robotics and Automation (ICRA) (pp. 409-416). IEEE.

[11] Chen, C., Lu, X., Markham, A., Trigoni, N. (2018, April). Ionet: Learning to cure the curse of drift in inertial odometry. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[12] Yan, H., Herath, S., Furukawa, Y. (2019). Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, and new methods. arXiv preprint arXiv:1905.12853.

[13] Quinchia, A. G., Falco, G., Falletti, E., Dovis, F., Ferrer, C. (2013). A comparison between different error modeling of MEMS applied to GPS/INS integrated systems. Sensors, 13(8), 9549-9588.

[14] Yang, Y., Geneva, P., Zuo, X., Huang, G. (2020). Online imu intrinsic calibration: Is it necessary?. Proc. of Robotics: Science and Systems (RSS), Corvallis, Or.

[15] Niu, X., Li, Y., Zhang, H., Wang, Q., Ban, Y. (2013). Fast thermal calibration of low-grade inertial sensors and inertial measurement units. Sensors, 13(9), 12192-12217.

[16] Geneva, P., Eckenhoff, K., Yang, Y., Huang, G. (2018, October). LIPS: Lidar-inertial 3d plane slam. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 123-130). IEEE.

[17] Ahmed, A., Roumeliotis, S. (2018, May). A visual-inertial approach to human gait estimation. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 4614-4621). IEEE.

[18] Kottas, D. G., Wu, K. J., Roumeliotis, S. I. (2013, November). Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 3172-3179). IEEE.

[19] Silva do Monte Lima, J. P., Uchiyama, H., Taniguchi, R. I. (2019). End-to-end learning framework for imu-based 6-dof odometry. Sensors, 19(17), 3777.