Gyro-Net: IMU Gyroscopes Random Errors Compensation Method Based on Deep Learning

Yunqi Gao[1,4], Dianxi Shi[1], Ruihao Li[1], Zhe Liu[2], Wen Sun[3]

***Abstract*— To solve the problem of inaccurate orientation estimation after long-term operations of the Inertial Measurement Unit (IMU), we present a learning-based method (called Gyro-Net) to estimate and compensate for IMU gyroscope random errors. We firstly introduce a semi-dense network structure, which extracts different scale features by IFES Block (IMU Feature Extraction & Selection Block), and adopts skip-connections and transition layers to adjust the feature pipeline. In this way, we can reuse features between different blocks before and after feature extraction, selection and compression. Driven by a proposed absolute and relative loss, the network can be trained and achieve the reduction of cumulative estimated orientation errors. The experimental results in public datasets show that our method can effectively and accurately estimate the orientation from raw IMU data. Moreover, we apply the network output directly to Open-VINS, and the results show that Gyro-Net can improve the accuracy of pose estimation for Open-VINS, especially in scenarios where camera-based estimation often struggles (e.g., fast motion, drastic lighting, viewpoint changes and motion blur).**

I. INTRODUCTION

The orientation estimation plays an extremely important role in the state estimation for robotics. When estimating the robot's trajectory, a small error in heading can often result in an unbelievable gap in the postion estimation, which can result in the failure of the entire odometry. Methods for orientation estimation typically include: One is using the camera to estimate rotation by geometric constraints, but it has the disadvantage of being susceptible to external environmental interference and low frequency. The other is using gyroscopes to integrate directly, and it has the advantages of being immune to external visual information, low power consumption and high operating frequency. However, it is easily affected by temperature and exists random errors such as bias and random walk etc.. After a long period of operation, integration will lead to non-negligible cumulative errors of orientation, and the reliability of the data will be greatly reduced. A low-cost IMU was selected to achieve the heading estimation task, so that the robot can work under extreme conditions. Therefore, in order to improve the accuracy of orientation estimation, it is necessary to compensate for the random errors caused by the IMU gyroscopes.

In order to reduce the effect of various errors, Allen Variance (open source code[a,b], extended Kalib library [1]) is usually applied in calibration before the use. However, this calibration does not provide an accurate estimation of time-varying errors. Therefore, there are also some direct or indirect fusion by introducing complementary sensors with IMU, allowing the robotic system to fuse features from multiple sensors and to obtain more robust pose estimation [2]. For example, some methods use cameras (constituting visual-inertial odometry (VIO)/visual-inertial simultaneous localization and mapping (VI-SLAM) [3]–[5]), LiDAR, GNSS and wheel encoder, etc. to obtain more information and fuse them [6]–[8] for pose estimation; Meanwhile, IMU is also used to obtain intermediate quantities, combined with other sensors as input to the Extended Kalman Filter (EKF) for pose estimation [9]–[11].

With the rise of deep learning in robotics, odometers have been divided into visual inertial odometers [10], [12], [13] that incorporate visual information, and inertial odometers [9], [14], [15] that are composed using only IMUs. Some methods only focus on orientation estimation [16], [17]. Most works are trained by regressing the relative pose of IMU, while [18] is used to estimate the pose by regressing the observable IMU motion items. A related work [19] uses the transformer-based method for inertial positioning. However, the existing algorithms do not consider the intrinsic connection between accelerometer and gyroscope, and does not use the IMU information sufficiently. In this work, we propose a deep learning-based method to compensate gyroscope random errors, and this core include: Firstly, the network is improved by introducing group convolution and attention mechanisms, meanwhile the features are reused to obtain a more compact structure. Secondly, focusing on the global loss, a global loss function is designed to effectively reduce the cumulative error of IMU data. In summary, the contributions of this paper are as follows:

- We design a semi-dense network structure to achieve the effective use of IMU information. The proposed network uses IFES block to extract richer features and incorporates the attention mechanism for selecting

Manuscript received 1 July 2022; accepted 20 November 2022. Date of publication 19 December 2022; date of current version 3 February 2023. This letter was recommended for publication by Associate Editor M. Kaess and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported in part by Science and Technology Innovation 2030 Major Project under Grant 2020AAA0104802 and in part by the National Natural Science Foundation of China under Grants 91948303 and 61903377.

[1]Artificial Intelligence Research Center (AIRC), Defense Innovation Institute, Beijing, 100166, China

[2]College of Computer, National University of Defense Technology, Changsha, 410073, China

[3]Sino-French Institute, Renmin University of China, Beijing, 100872, China

[4]Tianjin Artificial Intelligence Innovation Center (TAIIC), Tianjin 300457, China.

Correspondence: dxshi@nudt.edu.cn

Digital Object Identifier 10.1109/LRA.2022.3230594

[a]https://github.com/gaowenliang/imu_utils

[b]https://github.com/rpng/kalibr_allan

information between the accelerometer and gyroscope to improve the accuracy performance. Moreover, the network also adopts skip-connections and transition layers to adjust the feature pipeline and reuse features between different blocks before and after feature extraction, selection, and compression.
- In order to reduce the impact of cumulative errors on orientation estimation, we design a global loss function that combines relative angle loss with absolute angle loss. This enables the network to effectively reduce both local errors and accumulation errors over a long period of time.

We evaluate our method in the public datasets EuRoC [20], TUM-VI [21], and TUM-VIE [22]. The results show that: Gyro-Net achieves some advantages over other existing IMU methods for orientation estimation. Specifically, the absolute orientation/yaw errors are reduced by 30%/23% in EuRoC and 20%/13% in TUM-VI, respectively. Moreover, open-loop navigation (i.e., OpenVINS) performs even more competitively with closed-loop VINS-Mono on both datasets – clearly beating it on the EuRoC dataset.

II. RELATED WORK

A. Deep learning methods for processing IMU

VI-Net [12] uses a two-layer long short-term memory (LSTM) network to extract inertial data features and fuse the features directly to image features for pose estimation. Then, Chen et al. [23] proposes two fusion models based on different masking strategies in terms of fusion strategies between visual features and IMU features, allowing the network to automatically learn the most appropriate combination of features for the purpose of adaptive fusion and improving the robustness of trajectory estimation. IONet [14] first proposes a LSTM structure to output the relative displacements in 2D polar coordinates and accumulate them to obtain 2D position information. AbolDeepIO [15] respectively uses three LSTMs for feature extraction of accelerometer, gyroscope and sampling time. It concatenates features as input of the lower LSTM to regress position and rotation.

B. Deep learning methods for processing accelerometers

RIDI [24] assumes that the orientation of the Android device is known to rotate IMU data into a gravity-aligned frame, using SVR regression velocity to optimize the bias, but still using double integration from the corrected acceleration data to obtain the trajectory. On the basis of RIDI, RoNIN [25] uses three network structures based on ResNet, LSTM and TCN to regress velocity. TLIO [9] uses ResNet to extract feature and estimates the displacement and uncertainty. The network output is input to the EKF with IMU tightly coupled to estimate pose. RNIN-VIO [25] adds fully connected (FC) layers and LSTM to improve the network of TLIO and fuses visual information into the EKF to cope with the pose estimation in challenging scenes. Zhang et al. [18] obtains a more general model by regressing the observable IMU motion term instead of the relative poses. NILoc [19] presents the problem of inertial localization and provided a solution by a transformer-based network to estimate the position.

C. Deep learning methods for processing in gyroscopes

OriNet [16] uses four LSTMs to extract current moment gyroscope, previous moment gyroscope, sampling time and prior quaternion state features. It fuses them by slow fusion as input of LSTM and FC to estimate rotation and introduces genetic algorithms to correct the gyroscope bias. It also adds Gaussian noises to the training to increase the generalization ability of the network. Weber et al. [26] designs a network structure based on RNN+FC or CNN+FC to directly regress the attitude. Brossard et al. [17] uses only five dilated convolutions to correct for gyroscope noise and bias, and integrates the corrected gyroscope to directly calculate the attitude and competes with VIO methods. The network estimation results are also applied to Open-VINS, which improves Open-VINS heading estimation accuracy. In summary, many deep learning-based position estimation algorithms can achieve relatively good results in different motion scenarios, and as the scenarios become increasingly complex, accurate position estimation alone does not satisfy the existing task requirements. Therefore, we are more concerned with how to make full use of IMU information for more accurate heading estimation.

III. PRELIMINARIES

We define notations and frame definitions that we use in the paper. We consider $(.)^{bm}$ as measurements in body frame. $(.)^b_k$ denotes measurements in body frame at moment k. Finally, we denote $\hat{(\cdot)}$ the estimate of network. N is Sequence length.

According to [1] the measurement model of IMU, after taking into account white Gaussian noises, bias and scale factors, the gyroscope measurements is modeled as:

$$\omega_k^{bm} = S_g M_g \omega_k^b + S_{ga} a_k^b + \eta_g + b_g, \tag{1}$$

where, ω_k^b and a_k^b are ground truth of the angular velocity and acceleration in the body frame. S_g is scale factor. M_g is axis-misalignment. S_{ga} denotes the g-sensitivity representing the effect of the acceleration on the gyroscope measurements. η_g is zero-mean Gaussian noises. b_g is the bias. We set the initial values of $S_g M_g$ to identity matrix and trainable network parameters.

When performing orientation estimation, by integrating the IMU angular velocity measurements, Eq. 2 that uses quaternions to represent rotation can be shown as:

$$\mathbf{q}_{k+1} = \mathbf{q}_k \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\omega\delta t \end{bmatrix}, \tag{2}$$

where $\mathbf{q}_k$ represents the quaternion at moment k, ω is the average of k to k+1 moments.

IV. PROPOSED METHOD

A. Overview

The structure of gyroscope random error compensation system is shown in Fig. 1, which takes IMU data as input

and consists of three main parts: Data Preprocess, Network and Loss Function. Data Preprocess includes two steps: data augmentation and data padding. Firstly, referring to [16], we add Gaussian noises to IMU data for data augmentation to avoid overfitting during training, we add noise with a standard deviation of 1e-3 to the acceleration component and 8e-5 to the gyroscope component by the trial-and-error method. Secondly, because the longer the IMU works the larger the error, the data padding uses the value of the first sampling moment to learn the IMU features with small errors. The core of Gyro-Net is as follows: we extract multi-scale features on different details by using group dilated convolutions, and we use channel attention to compute weights and to reorganize features. It reuses the features by using skip-connections and transition layers to obtain a more compact and semi-dense network structure, which can estimate and compensate for gyroscope random errors to obtain Clean Gyro. Then, the quaternions from Clean Gyro integration and ground truth are used to construct relative and absolute loss functions, which can focus on local and globe errors.

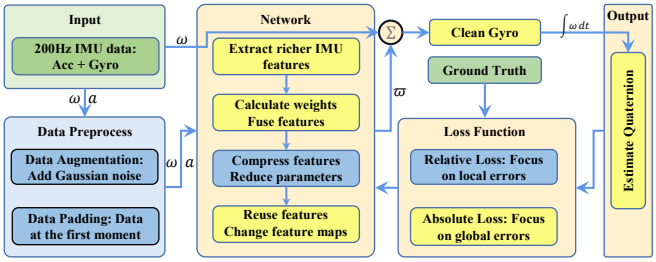


Fig. 1. Schematic illustration of our method. Acc: acceleration, gyro: gyroscope. It mainly consists of three parts: Data Preprocess, Neural Network (estimate and compensate for gyroscope random errors) and Loss Function (enable the network to focus on both the local relative error and the global cumulative error). Our method mainly focuses on the design of the network and loss function (yellow part).

B. A semi-dense Network Structure

According to Eq. 1, The gyroscope random error compensation is modeled as:

$$\hat{\omega}_k^b = (S_g M_g)^{\wedge} \omega_k^{bm} + \hat{\varpi} = \hat{W}\omega_k^{bm} + \left(S_{ga} a_k^b + \eta_g + b_g\right)^{\wedge}.$$

The random error estimation $\hat{\varpi}$ is calculated by the neural network as follows:

$$\hat{\varpi} = f\left(u_{k-N}^{bm}, \ldots, u_k^{bm}\right), \tag{3}$$

where $f(.)$ is the function defined by the neural network. Inspired by ResNet [27] and DenceNet [28], we propose a semi-dense network (Fig. 2(a)), which consists IFES Block, IMU Feature Compress Block (IFC Block) skip-connections and transition layers. In the orientation estimation, both gyroscope and acceleration data are used, but they have different weights to the task. Inspired by SKNet [29], we design the IFES block (shown in Fig. 2(b)) to extract richer IMU features by using two different receptive fields convolutions, and dynamically adjust the receptive fields for feature selection. Specifically, the six input channels of the first IFES Block correspond to the acceleration and gyroscope components in the xyz direction respectively. These channels are assigned weights by FC, which can dynamicly restructure features in different scenes. After the network, the number of channels is changed from 6 to 128. Finally the weights are assigned to the three channels by 1*1 convolution representing the gyroscopic value of the reduced random error on the three components of xyz. Overall, the block first uses two convolution kernels with different receptive fields to obtain different scale features. Then global average pooling is performed to embed the feature maps into the channels, and two FC layers and softmax is used to obtain the importance of each channel for heading estimation, i.e., Eq. 4. Finally, the weights on the channels are multiplied with the feature maps obtained by group convolution. Therefore, the final output of the network is Eq. 5.

$$W_{att} = \sigma\left(W_{fc1+fc2+avg}\left(W_{c1}(\mathrm{F}) + W_{c2}(\mathrm{F})\right)\right), \tag{4}$$

$$Output = W_{att} * (\mathrm{F}_1 + \mathrm{F}_2). \tag{5}$$

The parameters of the IFES block are shown in Table I, where FC-setting means the feature scale transformation in the FC layer. For example, 6-32-6 means that the first FC layer with input of 6 and output of 32, and the second layer with input of 32 and output of 6. The role of the first FC layer is as follows: Firstly, when the number of channels is less than 32, we expand the number of channels. It can effectively prevent information loss. Secondly, when the number of channels is greater than 32, the channels are compressed, and it can reduce the network parameters.The second FC layer is for channel importance prediction; The parameters are conv in Table I, 7-9 means that the sizes of the two convolutions in the IFES Block are 7 and 9, respectively.

We rename the convolution layer mentioned in [17] as IFC block, which is used to connect two IFES blocks for feature compression. We reuse the features using skip-connections and transition layers to prevent feature information loss when features are transferred between different blocks. We connect the output of IFC Block to the next non-adjacent IFC Block by skip-connections and to the second non-adjacent IFES Block by transition layers, forming a semi-dense network structure. The transition layer (Fig. 2(c)) can reduce the scale of the feature map and reduce the number of parameters. The ReLU [30] is used in the last transition layer which enables the network to handle the sharp corrections caused by the violent motion in the IMU signal. The network parameters corresponding to the transition layers are shown in Table I. In this way, Gyro-Net can strengthen the transfer of features and adjust the feature pipeline to make more effective use of feature information, which improves the compactness of the model. The semi-dense structure allows the network to be back-propagated with some layers receiving additional supervision signals through shorter connections, achieving an implicit deep supervision. In this way, this improves the feature flow between different blocks, and the errors are easily transferred to the earlier layers.

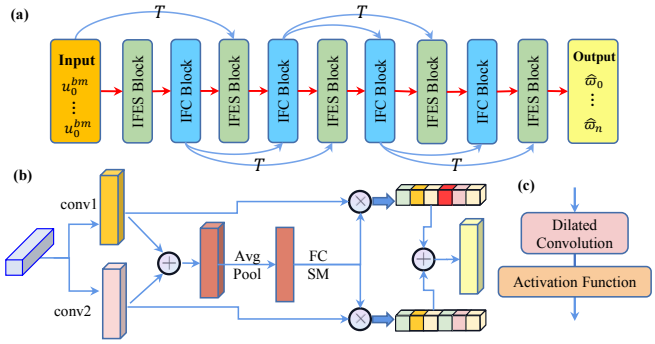


Fig. 2. Gyro-Net implementation details. (a) shows the Gyro-Net network structure, where T denotes transition layers, IFC Block is used to compress the feature scale. (b) is the structure of the IFES block for IMU feature extraction and selection. (c) is the structure of the transition layer to change the size of the feature map.

TABLE I: Gyro-Net parameters. Conv: The 1D kernel size of each block. Dia: The dilation of conv. FC-Setting: The parameters of FC layers in IFES block. In-Out: Number of input and output channels. AcFun: Activation Function.

Layer	IFES Block			Transition layer			
	Conv	Dia	FC-Setting	Conv	Dia	In-Out	AcFun
1	7-9	1	6-32-6	1	1	6-16	GELU
2	7-9	2	16-32-16	7	4	16-32	GELU
3	7-9	4	32-32-32	7	16	32-64	GELU
4	7-9	6	64-32-64	7	64	64-128	ReLU
5	7-9	1	128-32-128	-	-	-	-

C. Loss Function

In order to enable the network to focus on both local and global accuracy, we use relative and absolute loss functions.

Relative loss: Referring to [17] to design the local loss. To be able to match ground truth, the estimated $\hat{\omega}_k^b$ is integrated to calculate the increment of the quaternion over a period of sampling time as follows:

$$\delta \mathbf{q}_{k,k+j} = \mathbf{q}_k^{-1} \mathbf{q}_{k+j} = \prod_{k=i}^{i+j-1} \Omega\left(\omega_k \delta t\right), \tag{6}$$

the number of each integration is determined by parameter j, and the loss function is defined as follows:

$$L_j = \sum_{\text{floor }(N/j)} H\left(\Omega^{-1}\left(\delta \mathbf{q}_{k,k+j}^{-1}\left(\delta \mathbf{q}_{k,k+j}\right)^{\wedge}\right)\right), \tag{7}$$

where, $\Omega(.)$ represents the angle conversion to quaternions. Ω^{-1} denotes the conversion from quaternions to angles. $H(.)$ is the Huber function, the Huber cost threshold is set to 0.005. Therefore, the relative loss is defined as follows:

$$L_{rl} = L_{16} + 0.5 * L_{32}. \tag{8}$$

Absolute loss: In order to avoid the impact of the cummulative errors on the accuracy caused by the long runnning of the system, the loss function shown in Fig. 3 is designed, where the square contains 64 sampling moments, the blue line represents the quaternion increment during the square time period.

Firstly, the whole sequence is cut into several time periods in 64 sampling moments, and at the initial time, the ground truth ω_k^b and the estimated value $\hat{\omega}_k^b$ are integrated to obtain the initial quaternion increments $\delta q_{0,64}$ and $\delta \hat{q_{0,64}}$, respectively. The specific mathematical expression is:

$$\begin{cases} L_{al1}^m = \left(\delta \mathbf{q}_{0,64}^{-1}\right)^{\wedge}\left(\delta \mathbf{q}_{64*(\ m-1),64*\ m}\right)^{\wedge} \\ L_{al2}^m = \delta \mathbf{q}_{0,64}^{-1} \delta \mathbf{q}_{64*(\ m-1),64*\ m} \end{cases}, \tag{9}$$

where, $m \in [1, \mathrm{M}], \mathrm{M} = \text{ floor } (\mathrm{N}/64)$. Then the two increments are made in the loss:

$$L_{al} = \sum_{m=1}^{M} H\left(\Omega^{-1}\left(\left(L_{al1}^m\right)^{-1} L_{al2}^m\right)\right). \tag{10}$$

The final loss function is defined as follows, where α_1 and α_2 are set to 1 by default:

$$L_{all} = \alpha_1 L_{rl} + \alpha_2 L_{al}, \tag{11}$$

it enables the loss function to eliminate the errors at different scales, which can not only focus on the local errors of IMU data, but also effectively reduce the accumulation of global errors over a long period of time.

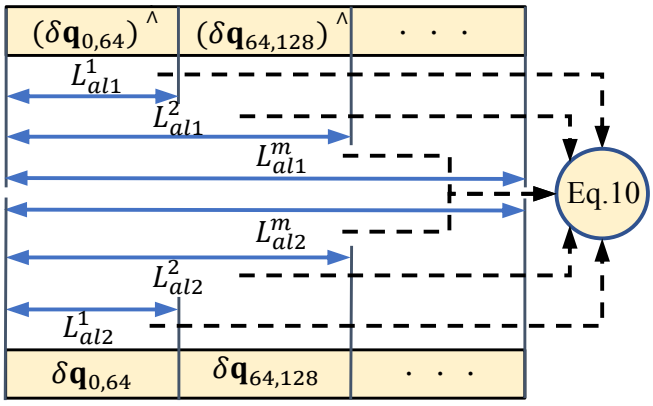


Fig. 3. Absolute loss diagram: The blue line indicates the rotation increment within the range. The black dashed line indicates the incremental input of the estimated and ground truth into Eq. 10. Eq. 10 stands for Equation 10 in Sec.IV.C of this paper.

V. EXPERIMENTS EVALUATION

In this section, we perform a series of experiments to verify the accuracy of our proposed Gyro-Net. The IMU-based inertial method and the visual inertial method using IMU+camera are compared and finally applied to Open-VINS. The results demonstrate the positive effect of our method on position and rotation estimation.

A. Method Implementation

Our experiments are conducted on the widely used EuRoC, TUM-VI and TUM-VIE datasets. To fairly compare each metric, the same training and test sets are used for learning-based method on both datasets. EuRoC consists of images and 200 Hz IMU data (uncalibrated) acquired by MAV in two scenes, containing a total of 11 sequences. The training set is MH_01_easy, MH_03_medium, MH_05_difficult, V1_02_medium, V2_01_easy and V2_03_difficult; the test set is MH_02_easy, MH_04_difficult, V2_02_medium, V1

_03_difficult and V1_01_easy. The TUM-VI dataset consists of images and 200Hz IMU data (properly calibrated) acquired by handheld devices in different scenes. Since the handheld devices have occlusion to the motion capture system for a period of time, each device has 0.2s of time unavailable and needs to be removed. The training set is room1, room3 and room5 and the test set is room2, room4 and room6. The training duration for each of the two datasets is the first 50s of the sequences, and the remainder is used as validation. Compared to the TUM-VI dataset, TUM-VIE adds two event cameras. Because each sequence has a shorter acquisition time, the first 17s of training dataset are only used for training. The training set is mocap-6dof, mocap-desk2 and mocap-shake2; the test set is mocap-desk and mocap-shake.

Our method is based on PyTorch 1.10.1, with the ADAM optimizer [22] and the learning rate is set to 0.01. We train for 1800 epochs. Eq. 11 uses default values in the EuRoC and TUM-VIE dataset. Since the absolute loss of our design needs to consider the before-and-after relationship of the sequences, the parameters in Eq. 11 are set as $\alpha_1 = 1$, $\alpha_2 = 0.2$ for the TUM-VI dataset with missing ground truth when performing training. All experiments uses a GTX 3090 device, each training sequences takes about 33s, and the video memory consumption is within 6G.

B. Compared Methods

For the inertial method using only IMU, the following methods are compared:

- RawIMU: uncalibrated IMU data, i.e. direct sensor readings;
- OriNet [16]: network based on LSTM, with training and test sets consistent with ours;
- Denoise IMU [17]: network using convolutions to correct gyroscope noise, and directly calculating attitude;
- Gyro-Net: Our proposed network, using attention mechanism to achieve accurate selection of features between the same blocks, and achieving feature reuse between different blocks by skip-connections.

For the VIO system, the following methods are compared:

- VINS-Mono [4]: the VIO system of monocular + IMU based on sliding windows, performing well on the EuRoC dataset;
- VINS-Mono (loop closure): On the basis of VINS-Mono, loop closure detection is added to improve the accuracy of pose estimation;
- Open-VINS: A versatile filter-based vision-inertial estimator that supports multiple modes of sensor selection. We set it to the mode of stere+IMU;
- Open-VINS (prop): Open-VINS system applying our IMU gyroscope processing method.

C. Experimental Results

In order to quantitatively verify the effectiveness of the system, Absolute Orientation Error (AOE), Relative Orientation Error (ROE) and Absolute Trajectory Error (ATE) metrics introduced in [31] are adopted. Table II and Fig. 4 show the results of our method and other comparative methods in terms of AOE and ROE. It can be found that our proposed Gyro-Net outperforms the existing algorithms on the test data. Applying our results directly to Open-VINS achieves better accuracy in orientation estimation than the original VIO method.

From Fig. 4, we can clearly see that the ROE of Gyro-Net is smaller than that of Denoise-IMU for each sub-trajectory, and is comparable to that of Open-VINS and VINS-Mono. Also Open-VINS (prop) is the most stable among the VIO systems compared, with minimal error variation.

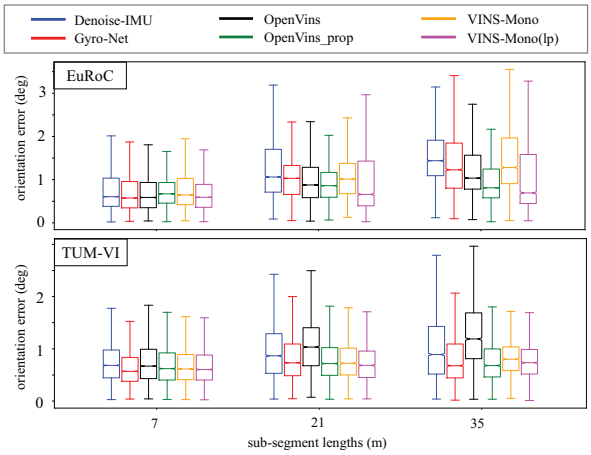


Fig. 4. ROE in terms of 3D orientation on the test sequences. Gyro-Net errors are small and do not fluctuate much when the trajectory becomes longer.

1) Gyro-Net performance for orientation estimation in inertial methods: Since the roll and pitch are observable in the VIO system, only the absolute error in the yaw direction is calculated when comparing with the VIO algorithm in Table II. Meanwhile, we design Table III to confirm that our proposed method be effective in all three directions of estimation, we compare the best results Denoise-IMU with Gyro-Net estimates in three directions of Euler angles. Result shows that our method achieves accurate estimation not only in yaw, but also in roll and pitch directions with effective random error estimation, which improves the overall orientation estimation results and outperforms baseline in general.

To illustrate our advantages more clearly, we choose some challenging scenes for testing by counting the motion characteristics of the test sequences such as MH_04_difficult with fast motion in dark environment and V1_03_difficult with obvious light perspective shift and motion blur.We observe the estimation effect of MH_04_difficult in the three Euler angular directions over the whole time axis, see Fig. 5. Also, to describe the transformation of the curve more precisely, the roll estimates of the MH_04_difficult sequence is changed to between [0,180]. It can be found that our method (orange line), is closer to the ground truth and possesses a smaller estimation error. Our method is able to perform orientation estimation with small cumulative errors, especially at some inflection points where it can be clearly seen that our method

TABLE II: Absolute orientation/yaw error on test sequences. Units are deg. The best results of the VIO system are highlighted in bold with bule. The best results of the inertial method are highlighted in bold with black.

	VINS-Mono	VINS-Mono (lp)	Open-VINS	Open-VINS (prop)	Raw IMU	OriNet	DenoiseIMU	Gyro-Net
MH_02_easy	1.34/1.32	**0.57/0.50**	1.11/1.05	0.93/0.85	146/130	5.75/**0.51**	**1.39**/0.85	2.07/0.9
MH_04_difficult	1.44/1.40	1.06/1.00	1.60/1.16	**0.98/0.83**	130/77.9	8.85/7.27	1.40/0.25	**0.63/0.14**
V1_01_easy	0.97/0.90	0.57/0.44	0.80/0.67	**0.57/0.36**	71.3/71.2	6.36/2.09	1.13/**0.49**	**0.66/0.49**
V1_03_difficult	4.72/4.68	4.06/4.00	2.32/2.27	**1.49/1.12**	119/84.9	14.7/11.5	2.70/0.96	**1.04/0.46**
V2_02_medium	2.58/2.41	1.83/1.61	1.85/1.61	**1.37/0.85**	117/86	11.7/6.03	3.85/2.25	**3.09/1.74**
Mean	2.21/2.14	1.62/1.52	1.55/1.37	**1.07/0.80**	125/89.0	9.46/5.48	2.10/0.96	**1.49/0.74**
Room2	**0.60/0.45**	0.69/0.50	2.47/2.36	1.56/1.23	118/88.1	-/-	1.31/1.18	**1.15/1.02**
Room4	0.76/0.63	0.66/0.51	0.97/0.88	**0.65/0.35**	74.1/48.2	-/-	1.48/0.85	**1.06/0.83**
Room6	0.58/0.38	**0.54**/0.33	0.63/0.51	0.62/**0.30**	94.0/76.1	-/-	1.04/0.57	**0.85/0.28**
Mean	0.66/0.49	**0.63/0.45**	1.33/1.25	0.94/0.62	95.7/70.8	-/-	1.28/0.82	**1.02/0.71**

TABLE III: Denoise-IMU and Gyro-Net estimates in three directions of roll/pitch/yaw on the test sequences.

	Denoise-IMU	Gyro-Net
MH_02_easy	**0.58** / 0.94 / **0.85**	1.62 / **0.91** / 0.90
MH_04_difficult	1.32 / 0.41 / 0.25	**0.51** / **0.34** / **0.14**
V1_01_easy	0.90 / 0.47 / **0.49**	**0.37** / **0.23** / **0.49**
V1_03_difficult	2.32 / 0.99 / 0.96	**0.67** / **0.64** / **0.46**
V2_02_medium	2.23 / **2.18** / 2.25	**1.27** / 2.22 / **1.74**
Mean	1.47 / 1.00 / 0.96	**0.89** / **0.87** / **0.74**
Room2	0.43 / 0.42 / 1.18	**0.37** / **0.39** / **1.02**
Room4	1.09 / **0.44** / 0.85	**0.30** / 0.59 / **0.83**
Room6	**0.51** / 0.69 / 0.5	0.57 / **0.55** / **0.29**
Mean	0.68 / 0.52 / 0.84	**0.40** / **0.50** / **0.74**

TABLE IV: Comparison of RMSE in relative orientation (rad) over ten IMU frames on the EuRoC.

	DenoiseIMU	TLIO	IDP	Gyro-Net
MH_02_easy	0.0060	0.0045	**0.0025**	0.0060
MH_04_difficult	0.0061	0.0058	0.0037	**0.0032**
V1_01_easy	0.0168	0.0181	0.0162	**0.0052**
V1_03_difficult	0.0051	0.0099	0.0067	**0.0048**
V2_02_medium	0.0118	0.0083	**0.0068**	0.0100
Mean	0.0092	0.0093	0.0072	**0.0058**

outperforms [17]. In the case of MH_04_difficult sequences moving at high speed (between 0.73min-0.78min at 2.00m/s-2.86m/s; between 1.17min-1.26min at 1.5m/s-2.66m/s), our method achieves better local relative accuracy and global accuracy in heading estimation.

We select V1_03_difficult with the largest reduction of AOE metrics in the test sequence and draw the error transformation curve of the whole sequence, see Fig. 6. It can be found that our method always maintains a very low error, especially at the cusps. Between 0.9s-1.4s, when the velocity and viewpoint of V1_03_difficult change drastically, our method is also able to perform heading estimation with low errors.

In order to investigate the relative orientation accuracy within short time windows. In this test, the RMSE for the relative orientation over 10 IMU readings are computed, and we also add IDP [18] into the competing algorithms. The results are reported in Table. IV.

In the comparative experiment, we compares Gyro-Net and Denoise-IMU. Fig. 7 shows the distribution of the Euler angular errors across the entire test set. It denotes that our methods performs better than Denoise-IMU on orientation metrics. It can be found that our method always maintains a very low error. To demonstrate the generality of our method,

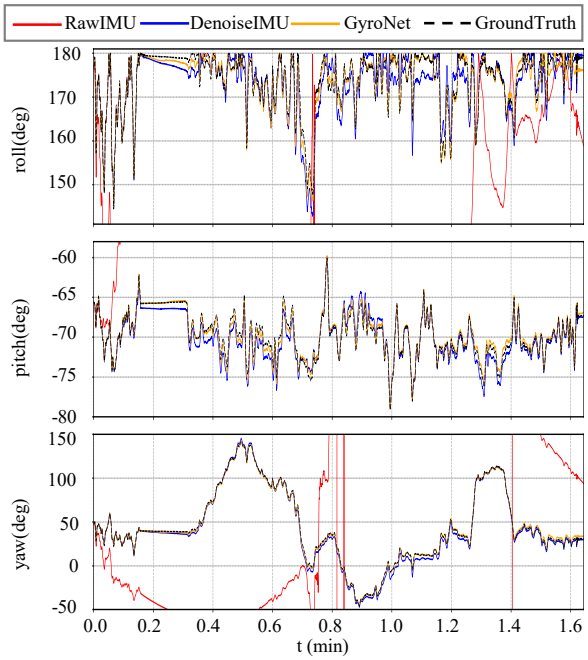


Fig. 5. Orientation estimates on the test sequences MH_04_difficult

TABLE V: RMSE of absolute orientation error (degree) / RMSE of relative orientation error (degree) over ten IMU frames on the TUM-VIE datasets.

	Denoise-IMU	Gyro-Net	Note
mocap-desk	7.32/0.4695	**5.52/0.0834**	loop motion
mocap-shake	5.23/0.2483	**2.09/0.0103**	Violent shaking
Mean	6.28/0.3589	**3.81/0.0469**	-

we also tests on the TUM-VIE dataset. Table V shows the AOE and RMSE in relative orientation (rad) over ten IMU frames for each sequence in the TUM-VIE dataset. For those test sequences, the results show that Gyro-Net can achieve higher accuracy in both long and short time periods, and can achieve better results in shark sequences. The main reason is that Gyro-Net uses a more flexible feature combination approach to the learning of random error features and pays better attention to the local relative and global errors through loss functions. The deep-supervision learning method implemented in semi-dense networks is able to learn discriminative features,thus it can cope with more complex situations.

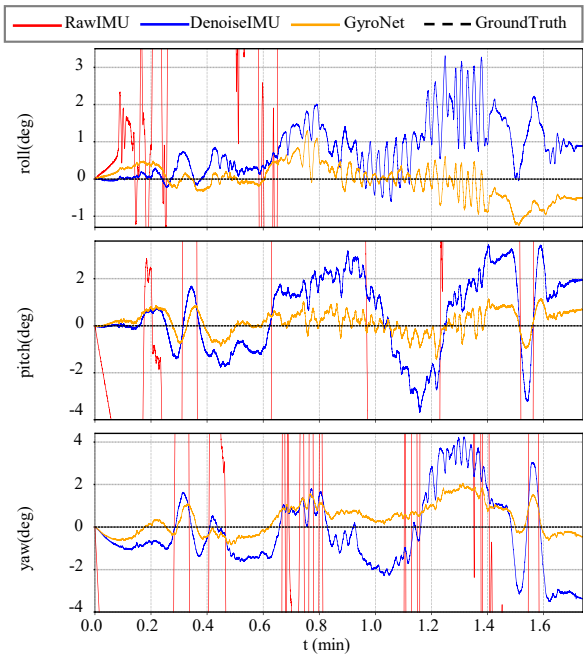


Fig. 6. Orientation errors of the sequences V1_03_difficult.

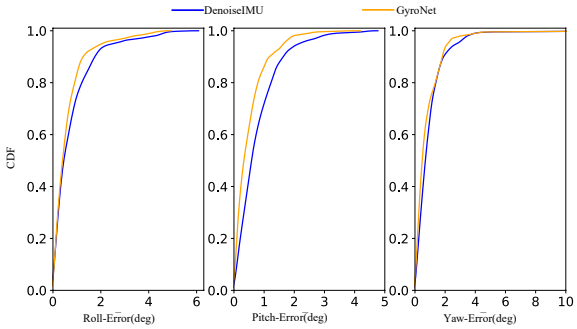


Fig. 7. Cumulative density function of Gyro-Net orientation estimation error and Denoise-IMU orientation estimation error on the entire test set. A steeper curve corresponds to a better network performance.

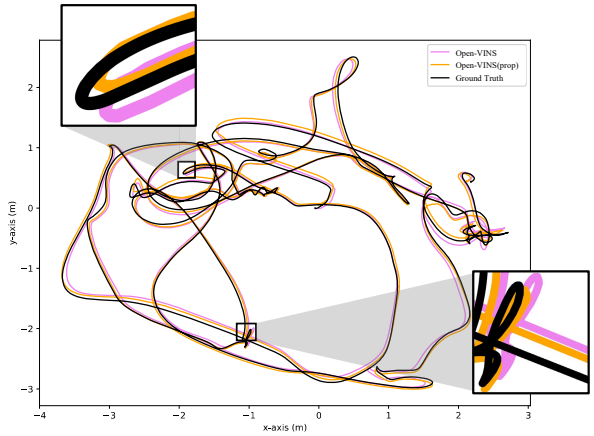
2) Open-VINS (prop) performance for displacement estimation in the VIO methods: To verify the effectiveness of gyroscope random error compensation, we subject all test sequences to trajectory error (ATE) calculations on public datasets as shown in Table VI. It can be seen that Open-VINS (prop) has higher accuracy than Open-VINS on most of the sequences, indicating that accurate navigation estimation has a positive effect on displacement estimation. Table VI shows that the improved sequences are generally those in which visual information is limited, such as fast motion, motion blur, or scenes with drastic lighting changes. This indicates that the method of gyroscopes random error compensation can compensate for the camera's shortcomings in position estimation.

To demonstrate the results more intuitively, we also give the estimated trajectories in the XY plane by Open-VINS (prop) on V2_02_medium and MH_04_difficult sequence. These two sequences are the most challenging sequences in the EuRoC MAV dataset because of the aggressive motion, texture-less area and dark scene. Although all systems have

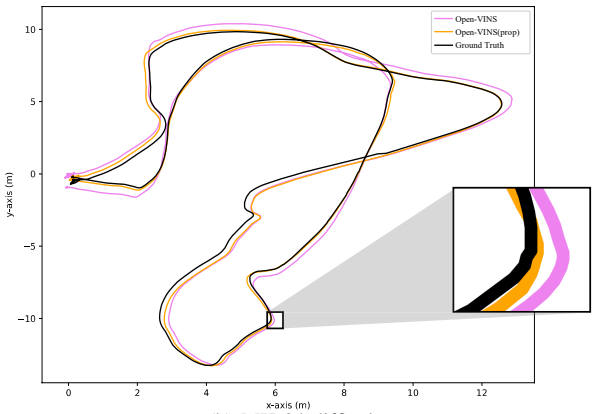
TABLE VI: ATE of Open-VINS (prop) on test sequences.

	VINS-Mono (lp)	Open-VINS	Open-VINS (prop)	Note
MH_02_easy	**0.114**	0.125	0.136	Good texture,Bright
MH_04_difficult	0.193	0.441	**0.191**	fast,Dark
V1_01_easy	0.055	0.074	**0.053**	slow,Bright
V1_03_difficult	0.151	0.079	**0.076**	fast,motion blur
V2_02_medium	0.125	0.079	**0.062**	fast,bright
Mean	0.128	0.160	**0.103**	-
Room2	0.063	0.086	**0.061**	bright,gental viewpoint
Room4	**0.043**	0.053	0.050	fierce viewpoint
Room6	0.053	0.039	**0.031**	slow,gental viewpoint
Mean	0.053	0.059	**0.047**	-

similar trajectories, as shown in Fig. 8, the trajectory estimated by Open-VINS (prop) is the closest to the ground truth, especially in the perspective change and accelerating scenarios.



(a) V2_02_medium



(b) MH_04_difficult

Fig. 8. Trajectory estimation results on EuRoC sequences.

3) Ablation Experiments: To prove the effectiveness of our propsoed improvements, we designed two sets of ablation experiments to be trained on EuRoC which include Gyro-Net (Net) with only network structure modified and Gyro-Net (AL) only loss function changed. We use AOE and RMSE in relative orientation (rad) over ten IMU frames to evaluate the performance. The results in Table VII show that both improvements in the network and the loss function help us to improve the baseline.

VI. CONCLUSION

In this paper, we propose a method for the compensation of IMU gyroscope random error. Gyro-Net uses IFES block

TABLE VII: RMSE of absolute orientation error (degree) / RMSE of relative orientation error (degree) over ten IMU frames for Ablation Experiments.

	Denoise-IMU	Gyro-Net (Net)	Gyro-Net (AL)	Gyro-Net
MH_02_easy	**1.39/0.0060**	3.52/0.0146	1.95/0.0164	2.07/**0.0060**
MH_04_difficult	1.40/0.0061	1.08/0.0038	0.99/0.0035	**0.63/0.0032**
V1_01_easy	1.13/0.0168	0.73/0.0050	0.69/**0.0032**	**0.66**/0.0052
V1_03_difficult	2.70/0.0051	1.30/0.0047	1.39/**0.0021**	**1.04**/0.0048
V2_02_medium	3.85/0.0130	**2.16/0.0075**	3.54/**0.0075**	3.09/0.010
Mean	2.10/0.0092	1.76/0.0071	1.71/0.0065	**1.49/0.0058**

to extract and select IMU data features. The semi-dense network structure can adjust the feature flow and reuse features to achieve the deep-supervision and full utilization of features, and it uses relative and absolute loss for training to achieve the purpose of focusing on local accuracy and global accuracy. The experimental evaluations show that our methods achieve good performance in orientation estimation. The core of the algorithm lies in the well-designed network structure and the full utilization of IMU features. The simple use of the output angular velocity can also lead to the effective enhancement of the VIO method including, challenging situations such as motion blur, dramatic light changes, etc. In the future, we would like to design a more effective method to combine the network output and VIO system to achieve better performance.

REFERENCES

[1] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending Kalibr: Calibrating the Extrinsics of Multiple IMUs and of Individual Axes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4304–4311.

[2] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[3] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A Research Platform for Visual-Inertial Estimation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4666–4672.

[4] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[6] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast Direct Lidar-Inertial Odometry," *IEEE Transactions on Robotics*, 2022.

[7] S. Cao, X. Lu, and S. Shen, "GVINS: Tightly Coupled GNSS–Visual–Inertial Fusion for Smooth and Consistent State Estimation," *IEEE Transactions on Robotics*, 2022.

[8] F. Mahlknecht, D. Gehrig, J. Nash, F. M. Rockenbauer, B. Morrell, J. Delaune, and D. Scaramuzza, "Exploring Event Camera-based Odometry for Planetary Robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8651–8658, 2022.

[9] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, V. Kumar, and J. Engel, "TLIO: Tight Learned Inertial Odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.

[10] D. Chen, N. Wang, R. Xu, W. Xie, H. Bao, and G. Zhang, "RNIN-VIO: Robust Neural Inertial Navigation Aided Visual-Inertial Odometry in Challenging Scenes," in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2021, pp. 275–283.

[11] M. Brossard, A. Barrau, and S. Bonnabel, "RINS-W: Robust Inertial Navigation System on Wheels," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2068–2075.

[12] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017, pp. 3995–4001.

[13] Y. Almalioglu, M. Turan, M. R. U. Saputra, P. P. de Gusmão, A. Markham, and N. Trigoni, "SelfVIO: Self-Supervised Deep Monocular Visual-Inertial Odometry and Depth Estimation," *Neural Networks*, vol. 150, pp. 119–136, 2022.

[14] C. Chen, X. Lu, A. Markham, and N. Trigoni, "IONet: Learning to Cure the Curse of Drift in Inertial Odometry," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018, pp. 6468–6476.

[15] M. A. Esfahani, H. Wang, K. Wu, and S. Yuan, "AbolDeepIO: A Novel Deep Inertial Odometry Network for Autonomous Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 1941–1950, 2019.

[16] M. A. Esfahani, H. Wang, K. Wu, and S. Yuan, "OriNet: Robust 3-D Orientation Estimation with a Single Particular IMU," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 399–406, 2019.

[17] M. Brossard, S. Bonnabel, and A. Barrau, "Denoising IMU Gyroscopes with Deep Learning for Open-Loop Attitude Estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4796–4803, 2020.

[18] M. Zhang, M. Zhang, Y. Chen, and M. Li, "IMU Data Processing for Inertial Aided Navigation: A Recurrent Neural Network Based Approach," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3992–3998.

[19] S. Herath, D. Caruso, C. Liu, Y. Chen, and Y. Furukawa, "Neural Inertial Localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6604–6613.

[20] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC Micro Aerial Vehicle Datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[21] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1680–1687.

[22] S. Klenk, J. Chui, N. Demmel, and D. Cremers, "TUM-VIE: The TUM Stereo Visual-Inertial Event Dataset," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8601–8608.

[23] C. Chen, S. Rosa, Y. Miao, C. X. Lu, W. Wu, A. Markham, and N. Trigoni, "Selective Sensor Fusion for Neural Visual-Inertial Odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 542–10 551.

[24] H. Yan, Q. Shan, and Y. Furukawa, "RIDI: Robust IMU Double Integration," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 621–636.

[25] S. Herath, H. Yan, and Y. Furukawa, "RoNIN: Robust Neural Inertial Navigation in the Wild: Benchmark, Evaluations, & New Methods," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3146–3152.

[26] D. Weber, C. Gühmann, and T. Seel, "Neural Networks Versus Conventional Filters for Inertial-Sensor-Based Attitude Estimation," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. IEEE, 2020, pp. 1–8.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[29] X. Li, W. Wang, X. Hu, and J. Yang, "Selective Kernel Networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 510–519.

[30] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for Activation Functions," 2018. [Online]. Available: https://openreview.net/forum?id=SkBYYyZRZ

[31] Z. Zhang and D. Scaramuzza, "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.