DIDO: Deep Inertial Quadrotor Dynamical Odometry

Kunyi Zhang, Chenxing Jiang, *Member, IEEE*, Jinghang Li, Sheng Yang, Teng Ma, Chao Xu, *Senior Member, IEEE*, and Fei Gao

***Abstract*—In this work, we propose an interoceptive-only state estimation system for a quadrotor with deep neural network processing, where the quadrotor dynamics is considered as a perceptive supplement of the inertial kinematics. To improve the precision of multi-sensor fusion, we train cascaded networks on real-world quadrotor flight data to learn IMU kinematic properties, quadrotor dynamic characteristics, and motion states of the quadrotor along with their uncertainty information, respectively. This encoded information empowers us to address the issues of IMU bias stability, quadrotor dynamics, and multi-sensor calibration during sensor fusion. The above multi-source information is fused into a two-stage Extended Kalman Filter (EKF) framework for better estimation. Experiments have demonstrated the advantages of our proposed work over several conventional and learning-based methods.**

***Index Terms*—Localization, sensorimotor learning, visual-inertial SLAM.**

I. Introduction

AERIAL robots are popular autonomous vehicles prevalently used in entertainment and industrial applications. To locate themselves, exteroceptive sensors (e.g., LiDAR and cameras) play a prominent role in common scenarios. However, when encountering degenerate cases where features are either heavily repeated or insufficient, these robots in-turn rely on interoceptive sensors (e.g., IMU and rotor tachometer) to deduct their poses. Inaccurate dynamics models, unestimated bias can lead to rapid divergence of state propagation.

Recent approaches [1]–[6], for exteroceptive sensor disabled scenarios, choose to learn the correspondence between IMU sequences and the relative poses through a data-driven approach. Although these works show the potential of deep neural networks for estimation, they are typically designed for scenes of walking pedestrians or low-degree-of-freedom (DOF) ground vehicles, which is difficult to generalize to aerial robots.

Manuscript received 24 February 2022; accepted 17 June 2022. Date of publication 7 July 2022; date of current version 20 July 2022. This letter was recommended for publication by Associate Editor H. Andreasson and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0108104, in part by Alibaba Innovative Research (AIR) Program, and in part by the National natural Science Foundation of China under Grant 62003299. *(Corresponding author: Fei Gao.)*

Kunyi Zhang, Chenxing Jiang, Chao Xu, and Fei Gao are with the State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China, and also with the Huzhou Institute, Zhejiang University, Huzhou 313000, China (e-mail: kunyizhang@zju.edu.cn; 3180101480@zju.edu.cn; cxu@zju.edu.cn; fgaoaa@zju.edu.cn).

Jinghang Li is with the Huzhou Institute, Zhejiang University, Huzhou 313000, China (e-mail: jhanglee.ro@gmail.com).

Sheng Yang and Teng Ma are with the Alibaba DAMO Academy Autonomous Driving Lab, Hangzhou 311121, China (e-mail: shengyang93fs@gmail.com; damon.mt@alibaba-inc.com).

Digital Object Identifier 10.1109/LRA.2022.3189168

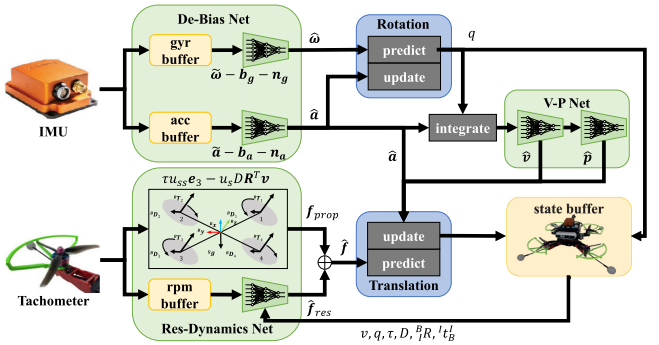

Fig. 1. The filter pipeline of proposed Deep Inertial quadrotor Dynamical Odometry (DIDO). The green parts are cascaded networks designed for learning kinematic properties, quadrotor dynamic characteristics, and motion states. The blue parts are EKF processes.

As a 4 DOF under-actuated system, there is a coupling of kinematic states between the velocity and tilt of the quadrotor, which are not fully exploited in the general state estimation system. The algorithm proposed in [7] can obtain the tilt and velocity of a quadrotor using only the tachometer and IMU data. However, its dynamics model only considers a quadratic thrust model and a first-order drag model, which the quadrotors do not strictly conform to. NeuroBEM [8] employs a neural network to compensate for the unmodeled parts such as airflow during the dynamics modeling, but it is not utilized for state estimation.

Therefore, this letter proposes a deep inertial-dynamical odometry framework, employing neural networks to learn both IMU and dynamics properties for better performance in quadrotor state estimation. The kinematic properties of IMU (accelerometer and gyroscope) are learned by two convolutional neural networks (CNN) for measurement de-biasing. The dynamic characteristics of quadrotors, with respect to the propeller rotation speed and states, are learned to compensate for mechanistic modeling by a CNN. Besides, the velocity and position of the quadrotor are observed in real-time by a recurrent neural network (RNN). Finally, the networks mentioned above are fused with IMU and tachometer measurements in a two stage EKF framework, as shown in Fig. 1. Specifically, in the stage of rotation, the de-biased angular velocity and acceleration are fused to get accurate rotation by gravity-alignment. In the translation stage, the modified dynamics is regard as process model, and the acceleration along with velocity and position from the network are considered as observation models, for accurate estimation of kinematic, dynamic states, and extrinsic parameters. The letter's contributions are enumerated below:

1) We present a series of networks to learn the IMU kinematics and quadrotor dynamics, demonstrating the capability

to provide precise observations with only raw IMU and tachometer data.
2) We propose a complete state estimation system that combines quadrotor dynamics and network observations within a two-stage EKF to jointly estimate kinematic, dynamic states, and extrinsic parameters.

II. Related Works

A. IMU Learning Based Estimation

IONet [1], RoNIN [2] and RIDI [3] are end-to-end learning methods for pose estimation. IONet [1] uses a long short-term memory network (LSTM) based network, whose inputs are accelerometer and gyroscope measurements within a time window in the world coordinate system. RoNIN [2] proposes three different neural network architectures to solve the inertial navigation problem: LSTM, temporal convolutional network (TCN), and residual network (ResNet). These models regress the velocity and displacement of the pedestrian in 2 dimensions. RIDI [3] design a two-stage system, which first regresses the velocity of pedestrians and then optimizes correction terms for IMU to match the velocity. The displacement of a pedestrian is finally obtained after integration.

AI-IMU [4], IDOL [5], TLIO [6] and CTIN [9] combines deep neural networks with filter framework to obtain better performance. AI-IMU [4] is specially designed for the purpose of vehicle's pose estimation, using a CNN to estimate the covariance of accelerometer and gyroscope measurements and applying them to an invariant extended Kalman filter (IEKF). In addition, the filter considers the vehicle sideslip constraints as well as the motion constraints in the vertical direction. IDOL [5] and TLIO [6] are both applied for pedestrian's pose estimation. IDOL [5] is divided into a rotation estimation module and a translation estimation module. In the rotation module, an LSTM network with gyroscope, accelerometer, and magnetometer data as inputs is trained to output the rotation, which is fused with the rotation obtained by integrating the gyroscope using the extended Kalman filter (EKF). In the position estimation module, the position is regressed using the LSTM network after converting the acceleration to the world coordinate system using the above rotation. TLIO [6] uses a ResNet network similar to RoNIN [2] to estimate pedestrian displacements without yaw, and a stochastic cloning extended Kalman filter (SCEKF) [10] framework to fuse the output values of the network with raw IMU measurements, for estimating position, rotation, and sensor bias in a tightly coupled manner. CTIN [9] uses ResNet and LSTM to obtain the local and global embeddings, and feeds these two embeddings into two multi-headed Transformers [11] to obtain the velocity and its covariance.

B. Quadrotor Dynamics and State Estimation

A quadrotor is usually powered by four rotating propellers, and the relationship between rotors' speed and dynamical states can be obtained by modeling the quadrotor's mechanics. Traditionally, a simple quadratic model is widely adopted, where the thrust and axial torque generated by the rotor are proportional to the square of their rotation speed with a constant factor [12]. The momentum theorem does not take into account either the motion of the rotor in air or the interactions between these rotors and the body. The blade-element-momentum (BEM) theory [13]–[15] combines blade-element theory and momentum theory to alleviate the difficulty of calculating the induced velocity and accurately capture the aerodynamic force and torque acting on single rotor in a wide range of operating conditions. But it also does not take into account any interactions between different propellers and body.

NeuroBEM [8] uses a neural network to fit the residuals between the real quadrotor dynamics and the calculated values from the BEM theory. It [8] takes the rotor speed and kinematic states as the network's inputs to allow the quadrotor to perform highly maneuverable trajectories. In [7], the authors estimate the tilt and velocity of the quadrotor in the body frame by applying only the tachometer and gyroscope as inputs and taking the accelerometer as observation. VIMO [16] and VID-Fusion [17] also consider the constraints of quadrotor dynamics in visual inertial fusion to improve the robustness and accuracy of pose estimation.

III. Kinematic and Dynamic Network Design

A. Preliminaries

1) IMU Model: IMU measurements include the gyroscope $\widetilde{\boldsymbol{\omega}}$ and non-gravitational acceleration $\widetilde{\boldsymbol{a}}$, which are measured in the IMU frame (the $\mathcal{I}$ frame with IMU sensor as the center, Front-Left-Up as the order of three axes) and given by:

$$\begin{aligned} {}^{\mathcal{I}}\widetilde{\boldsymbol{\omega}} &= {}^{\mathcal{I}}\boldsymbol{\omega} + {}^{\mathcal{I}}\boldsymbol{b}_{\boldsymbol{\omega}} + \boldsymbol{n}_{\boldsymbol{\omega}}, \\ {}^{\mathcal{I}}\widetilde{\boldsymbol{a}} &= {}^{\mathcal{I}}\boldsymbol{a} + {}^{\mathcal{I}}\boldsymbol{b}_{\boldsymbol{a}} + {}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}^{\top}{}^{\mathcal{G}}\mathbf{g} + \boldsymbol{n}_{\boldsymbol{a}}, \end{aligned} \tag{1}$$

where ${}^{\mathcal{I}}\boldsymbol{\omega}$ and ${}^{\mathcal{I}}\boldsymbol{a}$ are the true angular velocity and acceleration, ${}^{\mathcal{G}}\mathbf{g} = [0, 0, 9.8]$ (unit: m/s^2) is the gravity vector in the gravity-aligned frame (the $\mathcal{G}$ frame with z-axis pointing down vertically), ${}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}$ is the rotation matrix from the $\mathcal{I}$ frame to the $\mathcal{G}$ frame, $\boldsymbol{n}_{\boldsymbol{\omega}}$ and $\boldsymbol{n}_{\boldsymbol{a}}$ are the additive Gaussian white noise in gyroscope and acceleration measurements, $\boldsymbol{b}_{\boldsymbol{\omega}}$ and $\boldsymbol{b}_{\boldsymbol{a}}$ are the bias of IMU modeled as random walk:

$$\begin{aligned} \boldsymbol{n}_{\boldsymbol{\omega}} &\sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{\omega}}), & \dot{\boldsymbol{b}}_{\boldsymbol{\omega}} &\sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{b}_{\boldsymbol{\omega}}}), \\ \boldsymbol{n}_{\boldsymbol{a}} &\sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{a}}), & \dot{\boldsymbol{b}}_{\boldsymbol{a}} &\sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{b}_{\boldsymbol{a}}}). \end{aligned} \tag{2}$$

2) Quadrotor Dynamics: Because the propagation of kinematic states is driven by multiple propulsion units in a quadrotor system, we model the Newtonian dynamics according to [18]. The total driving force of a quadrotor in the body frame (the $\mathcal{B}$ frame with center of mass as the center, the same order as the $\mathcal{I}$ frame) is the sum of the thrust ${}^{\mathcal{B}}\boldsymbol{F}_t$ and drag force ${}^{\mathcal{B}}\boldsymbol{F}_d$ generated by each propulsion unit as follow:

$${}^{\mathcal{B}}\boldsymbol{F} = \sum_{i=1}^{4} \left({}^{\mathcal{B}}\boldsymbol{F}_{t_i} - {}^{\mathcal{B}}\boldsymbol{F}_{d_i} \right) = \sum_{i=1}^{4} \left(\tau u_i^2 \boldsymbol{e}_3 - u_i D {}^{\mathcal{B}}\boldsymbol{v}_i \right), \tag{3}$$

where τ is the thrust coefficient for the propellers, $D = diag(d_x, d_y, d_z)$ is the matrix of effective linear drag coefficients, $\boldsymbol{e}_3 = [0, 0, 1]^{\top}$ is the z axis in any frame, and u_i and ${}^{\mathcal{B}}\boldsymbol{v}_i$ are the rotation speed and velocity of the i-th rotor, respectively. Actually, the velocity of each rotor is ${}^{\mathcal{B}}\boldsymbol{v}_i = {}^{\mathcal{B}}\boldsymbol{v} + {}^{\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{r}_i^{\mathcal{B}}$, where ${}^{\mathcal{B}}\boldsymbol{v}$ and ${}^{\mathcal{B}}\boldsymbol{\omega}$ are the linear and angular velocity of the quadrotor's center of mass (COM), ${}^{\mathcal{B}}\boldsymbol{r}_i^{\mathcal{B}}$ is the position of the i-th rotor relative to the COM. To simplify the calculation,

we ignore the velocity discrepancy of different rotors and express it as ${}^{\mathcal{B}}\boldsymbol{v}_i \approx {}^{\mathcal{B}}\boldsymbol{v}$. The input notations are abbreviated as $U_{ss} = \sum_{i=1}^{4} u_i^2$ and $U_s = \sum_{i=1}^{4} u_i$. Therefore, we can obtain the Newtonian equation in the $\mathcal{G}$ frame:

$$m\frac{d}{dt}\left({}^{\mathcal{G}}\boldsymbol{v}\right) = {}^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}\left(\tau U_{ss}\boldsymbol{e}_3 - U_s D^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}}\right) - m^{\mathcal{G}}\mathbf{g}. \tag{4}$$

Note that, we divide the rotor speed by 10,000 to ensure the numerical stability.

B. Network Design

1) De-Bias Net: As an interoceptive sensor, IMU often requires several integrations to obtain the kinematic states, but the system would drift or diverge because of the failure to accurately estimate the accelerometer bias ${}^{\mathcal{I}}\boldsymbol{b}_a$ and gyroscope bias ${}^{\mathcal{I}}\boldsymbol{b}_\omega$. So, we design a *De-Bias Net* to learn the kinematic characteristics of IMU for de-biasing. The two *De-Bias Net*s for accelerometer and gyroscope have the same network architecture, a 1D version of ResNet [19] with only one residual block to boost the inference process. Fully-connected layers are extended to output.

The respective input features of both *De-Bias Net*s are historical raw accelerometer measurements ${}^{\mathcal{I}}\widetilde{\boldsymbol{a}}$ and gyroscope measurements ${}^{\mathcal{I}}\widetilde{\boldsymbol{\omega}}$, each output is ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{a}}$ and ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$ at every moment, separately.

We define two Mean Square Error (MSE) loss functions $\mathcal{L}_{\text{MSE},\boldsymbol{a}}$ and $\mathcal{L}_{\text{MSE},\boldsymbol{\omega}}$ for accelerometer and gyroscope, respectively, on the following integrated increments:

$$\begin{aligned}
\mathcal{L}_{\text{MSE},\boldsymbol{a}} &= \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{v}_{i,i+n} - \widehat{\boldsymbol{v}}_{i,i+n}\|_2^2,\\
\mathcal{L}_{\text{MSE},\boldsymbol{\omega}} &= \frac{1}{N}\sum_{i=1}^{N}\|\log\left(\left(\widehat{\boldsymbol{q}}_{i,i+n}\right)^* \otimes \boldsymbol{q}_{i,i+n}\right)\|_2^2,
\end{aligned} \tag{5}$$

where

$$\begin{aligned}
\widehat{\boldsymbol{v}}_{i,i+n} &= \int_i^{i+n} {}^{\mathcal{G}}_{\mathcal{I}_t}\mathbf{R}({}^{\mathcal{I}_t}\widetilde{\boldsymbol{a}} - {}^{\mathcal{I}_t}\widehat{\boldsymbol{b}}_{\boldsymbol{a}})dt,\\
\boldsymbol{v}_{i,i+n} &= {}^{\mathcal{G}}\boldsymbol{v}_{i+n} - {}^{\mathcal{G}}\boldsymbol{v}_i,\\
\widehat{\boldsymbol{q}}_{i,i+n} &= \int_i^{i+n} \frac{1}{2}{}^{\mathcal{I}_i}_{\mathcal{I}_t}\boldsymbol{q} \otimes ({}^{\mathcal{I}_t}\widetilde{\boldsymbol{\omega}} + {}^{\mathcal{I}_t}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}})dt,\\
\boldsymbol{q}_{i,i+n} &= ({}^{\mathcal{G}}_{\mathcal{I}_i}\boldsymbol{q})^* \otimes {}^{\mathcal{G}}_{\mathcal{I}_{i+n}}\boldsymbol{q}.
\end{aligned} \tag{6}$$

Note that, the subscript i is the abbreviation for time t_i, $\widehat{\boldsymbol{v}}_{i,i+n}$ and $\widehat{\boldsymbol{q}}_{i,i+n}$ are the intermediate variables calculated from the estimated values ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{a}}$ and ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$ of the *De-Bias Net* by the forward Euler method actually, $\boldsymbol{v}_{i,i+n}$ and $\boldsymbol{q}_{i,i+n}$ are the ground truth velocity increment and relative rotation of each sliding window, n is the window size, N is the batch size, and $\otimes$ means the quaternion multiplication. In training, ${}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}$ is the ground truth rotation.

2) Res-Dynamics Net: Actually, the dynamics model of quadratic thrust and approximately linear drag in Eq. (3) can not accurately model quadrotors because of complicated effects related to airflow and so on. Therefore, like NeuroBEM [8], we adopt the same ResNet [19] as *De-Bias Net*s to capture the unmodeled part of the quadrotor dynamics.

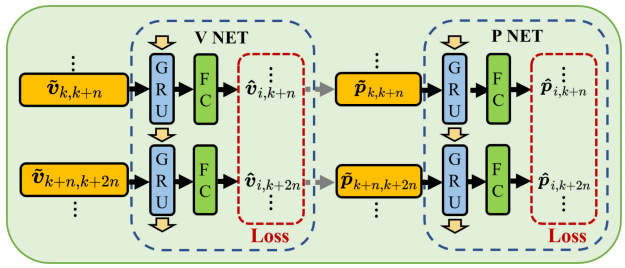

Fig. 2. The network structure for *V-P Net*.

We regard a sliding window of angular and linear velocity in the $\mathcal{B}$ and rotor speed data as input features to train the *Res-Dynamics Net* in a supervised fashion, whose history window size is set to 20.

In order to fit the unmodeled force and fuse it into the EKF framework reasonably, a MSE loss function is replaced by a Negative log-likelihood (NLL) loss [20] after the MSE loss stabilizes and converges:

$$\begin{aligned}
\mathcal{L}_{\text{MSE},\boldsymbol{f}} &= \frac{1}{N}\sum_{i=1}^{N}\|{}^{\mathcal{B}}\boldsymbol{a}^{\mathcal{G}}_{\mathcal{B}i} - {}^{\mathcal{B}}\widehat{\boldsymbol{a}}^{\mathcal{G}}_{\mathcal{B}i}\|_2^2,\\
\mathcal{L}_{\text{NLL},\boldsymbol{f}} &= \frac{1}{2N}\sum_{i=1}^{N}\log\det(\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{f}_i}) + \frac{1}{2}\mathcal{L}_{\text{MSE},\boldsymbol{f}},
\end{aligned} \tag{7}$$

where

$$\begin{aligned}
{}^{\mathcal{B}}\widehat{\boldsymbol{a}}^{\mathcal{G}}_{\mathcal{B}} &= \frac{1}{m}(\tau U_{ss}\boldsymbol{e}_3 - U_s D^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}} + \widehat{\boldsymbol{f}}_{res}) - {}^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top\mathcal{G}}\mathbf{g},\\
{}^{\mathcal{B}}\boldsymbol{a}^{\mathcal{G}}_{\mathcal{B}} &= {}^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top\mathcal{G}}\boldsymbol{a}^{\mathcal{G}}_{\mathcal{I}} + {}^{\mathcal{I}}_{\mathcal{B}}\mathbf{R}^{\top}({}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} \times ({}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} \times {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}) + {}^{\mathcal{I}}\widehat{\boldsymbol{\alpha}} \times {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}).
\end{aligned} \tag{8}$$

Note that, $\widehat{\boldsymbol{f}}_{res}$ and $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{f}}$ are the unmodeled force and its corresponding covariance, ${}^{\mathcal{B}}\widehat{\boldsymbol{a}}^{\mathcal{G}}_{\mathcal{B}}$ is the intermediate variable calculated from $\widehat{\boldsymbol{f}}_{res}$, ${}^{\mathcal{G}}\boldsymbol{a}^{\mathcal{G}}_{\mathcal{I}}$ is the ground truth of acceleration, ${}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} = {}^{\mathcal{I}}\widetilde{\boldsymbol{\omega}} - {}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$, ${}^{\mathcal{I}}\widehat{\boldsymbol{\alpha}} = \frac{d}{dt}{}^{\mathcal{I}}\widehat{\boldsymbol{\omega}}$ and ${}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}$ is the extrinsic translation between the $\mathcal{I}$ and $\mathcal{B}$ frame. In practice, we low-pass filter the $\widehat{\boldsymbol{\omega}}, \widehat{\boldsymbol{\alpha}}$ to reduce noise.

3) V-P Net: Even though *De-Bias Net*s help to reduce the bias of IMU measurements, the remaining tiny offset of acceleration would not avoid the cumulative error in the integration process. Therefore, we design a cascaded GRU network to learn velocity and position approximation that has less error accumulation over a long period of time. As every dimension of velocity and position are independent to each other, we adopt three cascaded GRU networks for the X, Y, and Z axes separately. The cascaded architecture has two parts: *V Net* and *P Net*, these two parts have the same structure as shown in Fig. 2.

Firstly, we integrate the de-biased acceleration as follow:

$$\widehat{\boldsymbol{v}}_{k,k+m} = \int_k^{k+m} {}^{\mathcal{G}}_{\mathcal{I}_t}\mathbf{R}({}^{\mathcal{I}_t}\widetilde{\boldsymbol{a}} - {}^{\mathcal{I}_t}\widehat{\boldsymbol{b}}_{\boldsymbol{a}})dt, \tag{9}$$

where the size of an integration window is m without overlap among each window. Each axis of $\widehat{\boldsymbol{v}}_{k,k+m}$ and integration time dt are input features of *V Net*. The hidden state is passed to fully-connected layers to regress the uniaxial velocity relative to the start of a sequence.

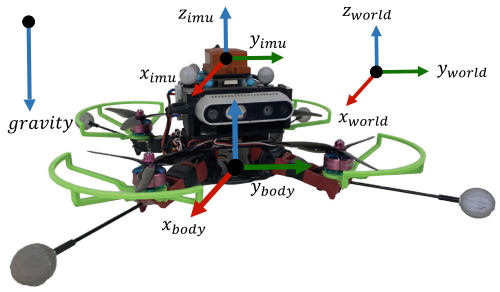

Fig. 3. The quadrotor platform, which assembles an IMU and 4 tachometers, is built for data acquisition.

Then, we use results of the *V Net* to calculate the displacement over the integration window as shown in (10), which is the input feature of *P Net* to regress the uniaxial position relative to the start of a sequence.

$$\widehat{\boldsymbol{p}}_{k,k+m} = \widehat{\boldsymbol{v}}_{k+m}\boldsymbol{t}_{k,k+m} - \iint_{k}^{k+m} {}^{\mathcal{G}}_{\mathcal{I}_t}\mathbf{R}({}^{\mathcal{I}_t}\widetilde{\boldsymbol{a}} - {}^{\mathcal{I}_t}\widehat{\boldsymbol{b}}_{\boldsymbol{a}})dt^2, \tag{10}$$

where $\widehat{\boldsymbol{v}}_{k+m} = \widehat{\boldsymbol{v}}_{i,k+m} + \widehat{\boldsymbol{v}}_i$, $\widehat{\boldsymbol{v}}_i$ is the initial velocity of the sequence, $\widehat{\boldsymbol{v}}_{i,k+m}$ is the result of the *V Net* in the $(k+m)$-th step and $\boldsymbol{t}_{k,k+m}$ is the duration of the integration window.

Similarly, we use MSE and NLL loss over the whole sequence as follows:

$$\mathcal{L}_{\text{MSE},\boldsymbol{v},\boldsymbol{p}} = \frac{1}{NM}\sum_{i=0}^{N}\sum_{j=0}^{M}\left(\|\boldsymbol{v}_{i,j} - \widehat{\boldsymbol{v}}_{i,j}\|_2^2 + \|\boldsymbol{p}_{i,j} - \widehat{\boldsymbol{p}}_{i,j}\|_2^2\right),$$

$$\mathcal{L}_{\text{NLL},\boldsymbol{v},\boldsymbol{p}} = \frac{1}{2NM}\sum_{i=1}^{N}\sum_{j=0}^{M}(\log\det(\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{v}_{i,j}}) + \log\det(\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{p}_{i,j}})) + \frac{1}{2}\mathcal{L}_{\text{MSE},\boldsymbol{v},\boldsymbol{p}}, \tag{11}$$

where $\boldsymbol{v}_{i,j} = {}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{I}_j} - {}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{I}_i}$, $\boldsymbol{p}_{i,j} = {}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{I}_j} - {}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{I}_i}$, M is the sequence length, N is batch size, $\widehat{\boldsymbol{v}}_{i,j}$ and $\widehat{\boldsymbol{p}}_{i,j}$ are the j-th predicted uniaxial velocity and position relative to the start of sequence i, $\boldsymbol{v}_{i,j}$ and $\boldsymbol{p}_{i,j}$ are the ground truth, $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{v}_{i,j}}$ and $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{p}_{i,j}}$ are the covariance of uniaxial velocity and position.

C. Dataset Preparation

There are only the Blackbird dataset [21] and the VID dataset [22] in the flying robot community that contain quadrotor dynamics. The former trajectories are too repetitive to generalize well for networks, while the latter suffers from insufficient data for training. Therefore, to satisfy the training requirements, we equip a new quadrotor as a data acquisition platform and record adequate data. The quadrotor assembles 4 tachometers and an additional IMU[1] as shown in Fig. 3, and the ground truth is provided by a VICON[2] system.

To ensure the diversity of trajectories, we record a total of 274 yaw-constant and yaw-forward trajectories (total time: 325 min, total distance: 19.92 km), which include 247 Random, 15 Circle and 12 Figure-8 trajectories. Where Random trajectories are sampled and optimized [23] to be executed by the quadrotor as a smooth trajectory.

To provide high-frequency and accurate supervised data for network training, we first fit the full state of kinematics at the IMU frequency with B-splines [24] in the gravity-aligned coordinate frame, and then identify the dynamics model of the quadrotor by offline optimization.

D. Training Details

1) Covariance Training: Except for *De-Bias Net*, the mean and covariance outputs of the corresponding networks are available for the remaining three networks. Every covariance $\widehat{\boldsymbol{\Sigma}}$ is a 3×3 matrix with 6 degrees of freedom, but a diagonal form is assumed in this letter and parameterized by 3 coefficients written as $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{\xi}} = \text{diag}(e^{2\widehat{\boldsymbol{\xi}}_x}, e^{2\widehat{\boldsymbol{\xi}}_y}, e^{2\widehat{\boldsymbol{\xi}}_z})$. Since it is difficult to converge the direct training covariance by $\mathcal{L}_{\text{NLL}}$, we use $\mathcal{L}_{\text{MSE}}$ in the first 20 epoch and then replace by $\mathcal{L}_{\text{NLL}}$.

2) V-P Net: When training *V-P Net*, we use different lengths of sequences. In experiment, the hidden sizes of GRU are [2, 64, 128, 256], integration window size m is 20.

3) Data Separation and Optimizer: We select 200 Random trajectories for training and the remaining as the validation and test sets. Adam optimizer is chose to minimize the loss function, and the initial learning rate is 10^{-4}. The length of validation and test set is from 58.5 (m) to 107.1 (m).

IV. Inertial Dynamical Fusion Pipeline

A. Rotation Stage

We separate the rotation and translation estimation in a two-stage EKF framework.

1) State: In the first rotation stage, there is only the rotation of the $\mathcal{I}$ in the $\mathcal{G}$ frame taken as filter's state:

$$\boldsymbol{x} = {}^{\mathcal{G}}_{\mathcal{I}}\boldsymbol{q}. \tag{12}$$

2) Process Model: The rotational equation is given:

$$\dot{\boldsymbol{x}} = \frac{1}{2}{}^{\mathcal{G}}_{\mathcal{I}}\boldsymbol{q} \otimes {}^{\mathcal{I}}\boldsymbol{\omega} = \frac{1}{2}{}^{\mathcal{G}}_{\mathcal{I}}\boldsymbol{q} \otimes ({}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} + \boldsymbol{n}_{\boldsymbol{\omega}}), \tag{13}$$

where ${}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} = {}^{\mathcal{I}}\widetilde{\boldsymbol{\omega}} - {}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$, $\boldsymbol{n}_{\boldsymbol{\omega}} \sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{\omega}})$, and ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$ is the output of the gyroscope *De-Bias Net*.

3) Measurement Model: The attitude controllers of most flying robots rely on the complementary filters [25], [26] to obtain attitude observations. Similarly, we consider the gravity alignment constraint to obtain the tilt observation:

$${}^{\mathcal{I}}\widehat{\boldsymbol{a}} \approx {}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}^{\top}{}^{\mathcal{G}}\mathbf{g} + \boldsymbol{n}_{\boldsymbol{a}}. \tag{14}$$

Similarly, ${}^{\mathcal{I}}\widehat{\boldsymbol{a}} = {}^{\mathcal{I}}\widetilde{\boldsymbol{a}} - {}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{a}}$, $\boldsymbol{n}_{\boldsymbol{a}} \sim \mathcal{N}(0, \boldsymbol{\Sigma}^2_{\boldsymbol{a}})$, and ${}^{\mathcal{I}}\widehat{\boldsymbol{b}}_{\boldsymbol{a}}$ is the output of the accelerator *De-Bias Net*.

B. Translation Stage

1) State: The state of the second stage is defined as:

$$\boldsymbol{x} = ({}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{B}}, {}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}}, \tau, \boldsymbol{d}, {}^{\mathcal{I}}_{\mathcal{B}}\boldsymbol{q}, {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}), \tag{15}$$

where ${}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{B}}$ and ${}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}}$ are respectively the velocity and position of the quadrotor body $\mathcal{B}$ frame expressed in the $\mathcal{G}$ frame, $\boldsymbol{d}$ is the drag vector of (d_x, d_y, d_z), and $({}^{\mathcal{I}}_{\mathcal{B}}\boldsymbol{q}, {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}})$ is the extrinsic parameter between the $\mathcal{B}$ and the $\mathcal{I}$ frame.

[1]https://www.xsens.com/mti-300
[2]https://www.vicon.com/

TABLE I
THE TABLE LISTS INITIAL VALUES AND COVARIANCES SETTING FOR ALL POSE ESTIMATION EXPERIMENTS

	${}^{\mathcal{G}}_{\mathcal{I}}\boldsymbol{q}$	${}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{B}}$	${}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}}$	τ	$\boldsymbol{d}$	${}^{\mathcal{I}}_{\mathcal{B}}\boldsymbol{q}$	${}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}$
value	${}^{\mathcal{G}}_{\mathcal{I}_0}\boldsymbol{q}$	${}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{I}_0}$	${}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{I}_0}$	1.1	$[0,0,0]$	$[1,0,0,0]$	$[0,0,0]$
covariance	$1\mathrm{e}{-8}\times\mathbf{I}_3$	$1\mathrm{e}{-4}\times\mathbf{I}_3$	$1\mathrm{e}{-6}\times\mathbf{I}_3$	$1\mathrm{e}{-4}$	$5\mathrm{e}{-4}\times\mathbf{I}_3$	$5\mathrm{e}{-5}\times\mathbf{I}_3$	$5\mathrm{e}{-4}\times\mathbf{I}_3$

2) Process Model: We regard the quadrotor dynamics as the input, and express the complete process model as follows:

$$
\begin{aligned}
{}^{\mathcal{G}}\dot{\boldsymbol{p}}^{\mathcal{G}}_{\mathcal{B}} &= {}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}},\\
{}^{\mathcal{G}}\dot{\boldsymbol{v}}^{\mathcal{G}}_{\mathcal{B}} &= \frac{1}{m}{}^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}(\tau U_{ss}\boldsymbol{e}_3 - U_s D^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top}{}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}} + \widehat{\boldsymbol{f}}_{res} + \boldsymbol{n}_{\boldsymbol{f}}) - {}^{\mathcal{G}}\mathbf{g},\\
\dot{k}_t &= 0, \quad \dot{\boldsymbol{d}} = \mathbf{0}, \quad {}^{\mathcal{I}}_{\mathcal{B}}\dot{\boldsymbol{q}} = \mathbf{0}, \quad {}^{\mathcal{I}}\dot{\boldsymbol{t}}^{\mathcal{I}}_{\mathcal{B}} = \mathbf{0},
\end{aligned}
\tag{16}
$$

where ${}^{\mathcal{G}}_{\mathcal{B}}\mathbf{R} = {}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}\,{}^{\mathcal{I}}_{\mathcal{B}}\mathbf{R}$, and $\widehat{\boldsymbol{f}}_{res}$ and $\widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{f}}(\boldsymbol{n}_{\boldsymbol{f}} \sim \mathcal{N}(0, \widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{f}}))$ are the *Res-Dynamics Net* outputs.

3) Measurement Model: Firstly, since accelerometer measurements are not used before, we take the inconsistency of the $\mathcal{I}$ and $\mathcal{B}$ frame into account and express the dynamics measurement equation as follow:

$$
\begin{aligned}
{}^{\mathcal{I}}\widehat{\boldsymbol{a}} = \frac{1}{m}{}^{\mathcal{I}}_{\mathcal{B}}\mathbf{R}\left(\tau U_{ss}\boldsymbol{e}_3 - U_s D^{\mathcal{G}}_{\mathcal{B}}\mathbf{R}^{\top}{}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}} + \widehat{\boldsymbol{f}}_{res}\right)\\
- {}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} \times ({}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} \times {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}) - {}^{\mathcal{I}}\widehat{\boldsymbol{\alpha}} \times {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}} + \boldsymbol{n}_a.
\end{aligned}
\tag{17}
$$

What's more, our aforementioned *V-P Net* can be used as observers of velocity and displacement:

$$
\begin{aligned}
{}^{\mathcal{G}}\widehat{\boldsymbol{v}}^{\mathcal{G}}_{\mathcal{I}} &= {}^{\mathcal{G}}\boldsymbol{v}^{\mathcal{G}}_{\mathcal{B}} - {}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}\left({}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} \times {}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}}\right) + \boldsymbol{n}_v,\\
{}^{\mathcal{G}}\widehat{\boldsymbol{p}}^{\mathcal{G}}_{\mathcal{I}} &= {}^{\mathcal{G}}\boldsymbol{p}^{\mathcal{G}}_{\mathcal{B}} - {}^{\mathcal{G}}_{\mathcal{I}}\mathbf{R}\,{}^{\mathcal{I}}\boldsymbol{t}^{\mathcal{I}}_{\mathcal{B}} + \boldsymbol{n}_p.
\end{aligned}
\tag{18}
$$

Specially, *V-P Net* outputs ${}^{\mathcal{G}}\widehat{\boldsymbol{v}}^{\mathcal{G}}_{\mathcal{I}}$ and ${}^{\mathcal{G}}\widehat{\boldsymbol{p}}^{\mathcal{G}}_{\mathcal{I}}$ as well as their covariances $\widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{v}}(\boldsymbol{n}_{\boldsymbol{v}} \sim \mathcal{N}(0, \widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{v}}))$ and $\widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{p}}(\boldsymbol{n}_{\boldsymbol{p}} \sim \mathcal{N}(0, \widehat{\boldsymbol{\Sigma}}^2_{\boldsymbol{p}}))$. More details about the EKF process and observability proof can be found in the supplementary material [27].

V. EXPERIMENTS

A. Metrics Definition

In order to assess the performance of our system, we define the metrics including the Absolute Translation Error (**ATE**), the Absolute Rotation Error (**ARE**), the Relative Translation Error (**RTE**), the Relative Rotation Error (**RRE**), and the Translation Drift (**TD**), the Rotation Drift (**RD**) as follows:

- **ATE** (m):= $\sqrt{\frac{1}{l}\sum_{i=1}^{l}\|{}^{\mathcal{G}}\boldsymbol{p}_i - {}^{\mathcal{G}}\widehat{\boldsymbol{p}}_i\|_2^2}$,
- **ARE** (m):= $\sqrt{\frac{1}{l}\sum_{i=1}^{l}\|\boldsymbol{d}({}^{\mathcal{G}}_{\mathcal{I}_i}\boldsymbol{q}, {}^{\mathcal{G}}_{\mathcal{I}_i}\widehat{\boldsymbol{q}})\|_2^2}$,
- **RTE** (%):= $\sqrt{\frac{1}{l}\sum_{i=1}^{l}\|{}^{\mathcal{G}}\boldsymbol{p}_{i,i+\Delta t} - {}^{\mathcal{G}}\widehat{\boldsymbol{p}}_{i,i+\Delta t}\|_2^2}$,
- **RRE** (rad):= $\sqrt{\frac{1}{l}\sum_{i=1}^{l}\|\boldsymbol{d}({}^{\mathcal{I}_{i+\Delta t}}_{\mathcal{I}_i}\boldsymbol{q}, {}^{\mathcal{I}_{i+\Delta t}}_{\mathcal{I}_i}\widehat{\boldsymbol{q}})\|_2^2}$,
- **TD** (rad):= $\|{}^{\mathcal{G}}\boldsymbol{p}_l - {}^{\mathcal{G}}\widehat{\boldsymbol{p}}_l\|_2^2/(\text{trajectory-length})$,
- **RD** (rad/ min) := $\|\boldsymbol{d}({}^{\mathcal{G}}_{\mathcal{I}_l}\boldsymbol{q}, {}^{\mathcal{G}}_{\mathcal{I}_l}\widehat{\boldsymbol{q}})\|_2^2/(\text{sequence-duration})$,

where the rotation distance is $\boldsymbol{d}(\boldsymbol{q}, \widehat{\boldsymbol{q}}) = log(conj(\widehat{\boldsymbol{q}}) \otimes \boldsymbol{q})$, l is the length of each dataset and Δt is set to $1/20s$.

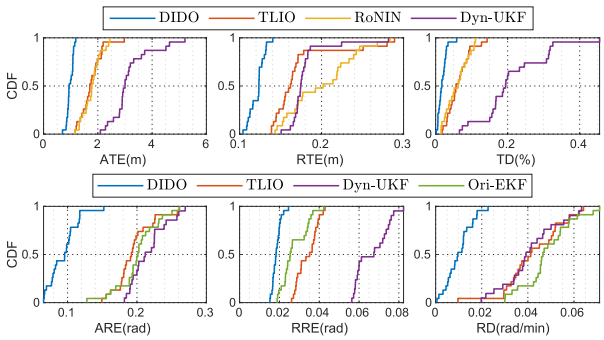

Fig. 4. Pose comparison between DIDO and other methods. Each subplot shows the cumulative density function (CDF) on the entire test set. The closer the graph is to the left, the better the performance.

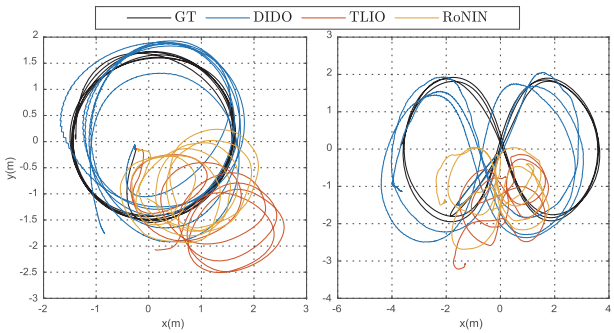

Fig. 5. Top view diagrams of on two test trajectories.

B. Pose Comparison

To better demonstrate our system, we select several conventional and learning-based algorithms to compare the estimation accuracy of 6D pose in the test set. Note that we set the initial values and covariances in Table I. The proposed DIDO and TLIO [6] are the complete systems whose results are the filter output. We train and test the former RoNIN-ResNet [2] by providing the ground truth rotation, so there is no rotation error shown. There are two conventional algorithms, Dyn-UKF [7] and Ori-EKF [28]. Dyn-UKF [7], integrates the dynamics as a forward process and updates the tilt and velocity in $\mathcal{B}$ frame by the raw accelerometer measurements, while all states are converted to the $\mathcal{G}$ frame by estimated rotation. Ori-EKF [28] only obtains the rotation using the raw IMU measurements.

Fig. 4 shows the error distributions among the entire test set, which distinctly shows the proposed DIDO outperforms other methods. Dyn-UKF [7] based only on the momentum theory does not work well, while DIDO compensates for the biased dynamics model resulting in a better performance. Fig. 5 presents three learning-based methods for position estimation on unseen simple trajectories. Fig. 10 depicts the performance of different methods for each axis. TLIO [6] and RoNIN-ResNet [2] both

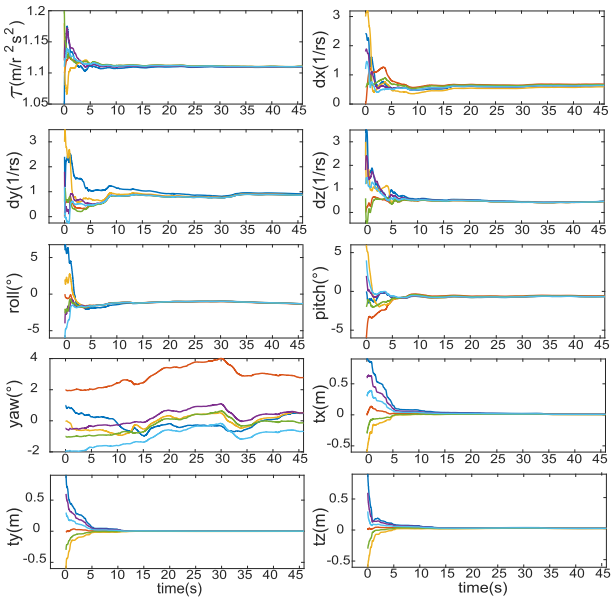

Fig. 6. Parameters convergence for 6 runs.

have unavoidable cumulative drift, while attributing to the *V-P Net*, DIDO has stable position estimation. Besides, *De-Bias Net* decelerates the severe yaw drift, which effectively helps the position estimation.

C. Parameters Estimation

To show the effectiveness and robustness of the parameter identification performance of the proposed system, we conduct an experiment by perturbing the initial values of the estimated parameters. Fig. 6 presents the convergence of the parameters in six independent runs, in which the majority of the parameters can converge quickly, except for the extrinsic parameter rotation expressed in terms of Euler angles (XYZ in Tait–Bryan angles). This is due to the fact that the quadrotor in our dataset does not fly fast enough, so the yaw angle of the extrinsic rotation appears to be weakly observable. The other degradation cases are: when ${}^{\mathcal{I}}\widehat{\boldsymbol{\omega}} = \mathbf{0}_{3\times1}$, ${}^{\mathcal{I}}\boldsymbol{t}_{\mathcal{B}}^{\mathcal{I}}$ is unobservable, and when ${}^{\mathcal{G}}\boldsymbol{v}_{\mathcal{B}}^{\mathcal{G}} = [0; 0; *]$, D and ${}_{\mathcal{B}}^{\mathcal{I}}\mathbf{R}$ are unobservable. The detailed proof process is provided in the supplementary material [27].

VI. Ablation Study

To be able to better illustrate the effectiveness of the proposed system, we verify the behavior of each network module through ablation experiments.

A. De-Bias Net

We compare the rotation metrics for the gyroscope *De-Bias Net* and translation metrics for the accelerometer one, respectively. As shown in the Fig. 7, the debiased gyroscope significantly outperforms method without debiasing, and updating by the debiased accelerometer results in better absolute rotation but with a slight loss of relative rotation. The debiased accelerometer also contributes to translation because debiasing facilitates the estimation of velocity increment (the input of *V-P Net*) and thus improves the accuracy of the *V-P Net* observation. Note

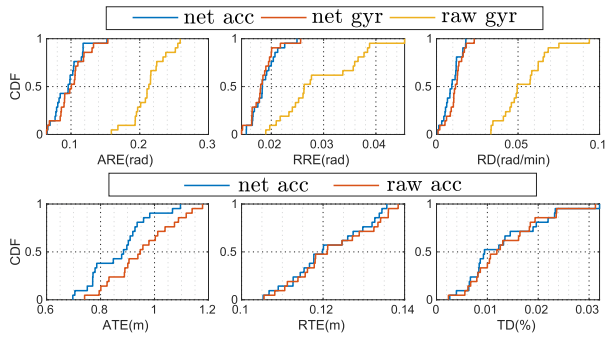

Fig. 7. De-Bias Net ablation study. The first row shows the rotational comparison between the three methods *net acc* (debiased gyroscope $\widehat{\boldsymbol{\omega}} = \widetilde{\boldsymbol{\omega}} - \widehat{\boldsymbol{b}}_{\boldsymbol{\omega}}$ integrating and debiased accelerometer $\widehat{\boldsymbol{a}} = \widetilde{\boldsymbol{a}} - \widehat{\boldsymbol{b}}_{\boldsymbol{a}}$ updating), *net gyr* ($\widehat{\boldsymbol{\omega}}$ integrating) and *raw gyr* ($\widetilde{\boldsymbol{\omega}}$ integrating). The second row shows the translational comparison between *net acc* (with accelerometer *De-Bias Net*) and *raw acc* (without accelerometer *De-Bias Net*).

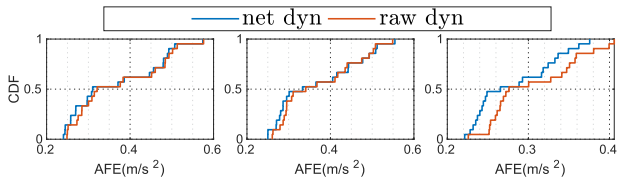

Fig. 8. Res-Dynamics Net ablation study. The metrics in three directions are used to show the force estimation of *net dyn* and *raw dyn* (with and without *Res-Dynamics Net*).

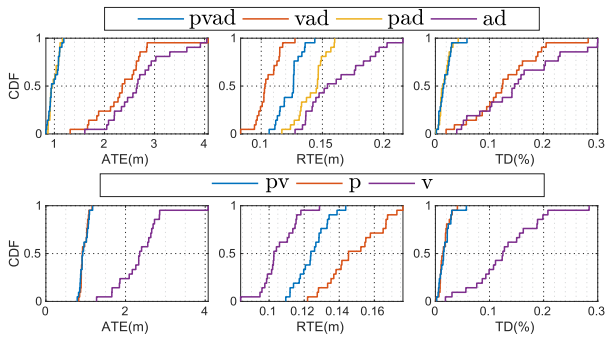

Fig. 9. V-P Net ablation study. Translational estimation is compared under six different combinations of *p* (position update), *v* (velocity update), *a* (acceleration update) and *d* (using the *Res-Dynamics Net*).

that, the system is updated by (17) and (18) when verifying the accelerometer *De-Bias Net*.

B. Res-Dynamics Net

The *Res-Dynamics Net* depends on dynamical parameters and other motion states which need to be estimated online in the proposed system, so we validate the tri-axial forces by comparing the complete EKF process with and without the *Res-Dynamics Net*. Similarly, we also define a metric to measure the precision of force estimation:

- **AFE**:= $\sqrt{\frac{1}{l}\sum_{i=1}^{l}\|\boldsymbol{f}_i - \widehat{\boldsymbol{f}}_i\|_2^2}$.

As shown in the Fig. 8, *Res-Dynamics Net* allows for more accurate dynamics modeling, especially in the z-axis.

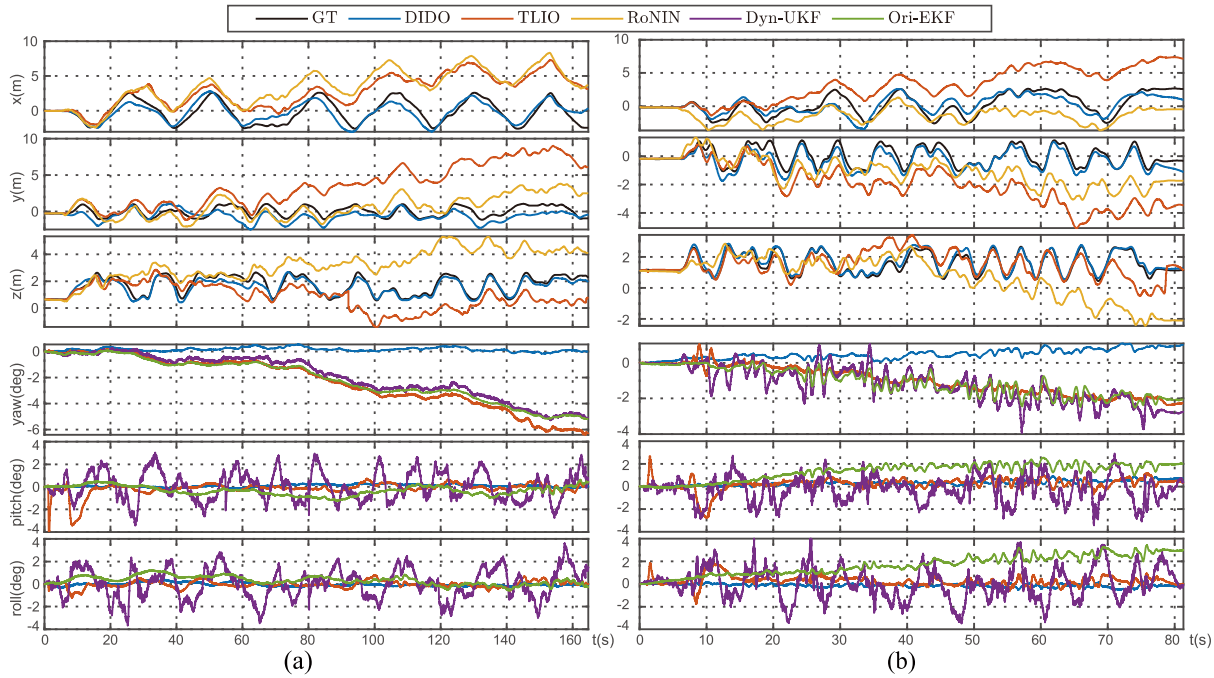

Fig. 10. Demonstration of different methods for pose estimation. There are two Random trajectories corresponding to different max uniaxial velocities (a:0.5 m/s, b:2.5 m/s) in the test set. The ground truth positions are plotted as black lines in the first row while ignoring the Dyn-UKF [7] due to its divergence. The rotation errors are drawn in the second row.

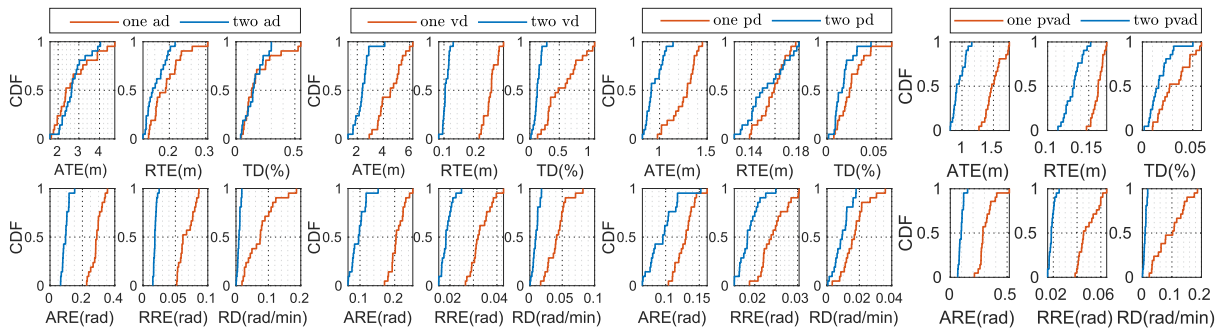

Fig. 11. Pose Comparison of EKF. One-stage and two-stage EKF are abbreviated as *one* and *two*, respectively. *p*, *v*, *a* and *d* hold the same meanings as Fig. 9. For example, *one vd* refers to a one-stage EKF updated by velocity in the case of the *Res-Dynamics Net*.

C. V-P Net

To verify the effectiveness of the *V-P Net*, we design two sets of experiments. As shown in the Fig. 9, *P Net* and *V Net* enhance the estimation accuracy of position and velocity, respectively. Furthermore, using *V-P Net* results in better RTE with guaranteed ATE in translation estimation.

D. One-Stage or Two-Stage

To demonstrate the effect of these three measurement models on the state estimation system, we design four sets of experiments. As shown in the Fig. 11, rotation is prone to be incorrectly updated by each noisy observation in the one-stage EKF, which further affects the estimation of velocity and position. This is due to the fact that rotation is very sensitive as an explicit or implicit input to the networks. Once the rotation can not be updated correctly during the EKF process, it will make the outputs of networks worse and thus impair the state estimation system.

VII. Discussions

A. Dynamics or IMU Based

In this sensor fusion system, there are two ways to obtain acceleration, one is the accelerometer measurements, and the other is the quadrotor dynamics. However, if the accelerometer measurements are used as the process model, the constraint between tilt and velocity does not hold. As a consequence, the dynamics-based methods may be superior to the IMU-based one in the quadrotor estimation system.

B. Covariance

Covariances are of great importance because they can be used to describe the data distribution and also taken as weights for multi-sensor fusion. The neural network covariances can dynamically depict the uncertainty of the network output mean values, which reduce the inaccuracy led by the static covariance

setting. However, in the EKF process, the covariances may propagate inaccurately due to the linear approximation of the process and measurement models. Therefore, we do not directly use the neural network covariances but scale the covariances in a similar way to TLIO to obtain a better estimation.

VIII. Conclusion and Future Work

A. Conclusion

In this work, we propose an inertial and quadrotor dynamical odometry system introducing deep neural networks in a two-stage tightly-coupled EKF framework. To the best of our knowledge, this is the first estimation framework that combines IMU, quadrotor dynamics and deep learning methods to simultaneously estimate kinematic, and dynamic states, as well as extrinsic parameters. To take full advantage of the two interoceptive sensors, we design deep neural networks to regress the IMU bias, the dynamics compensation, motion states of the quadrotor, as well as their covariances.

Experimental results have demonstrated that the proposed method outperforms the other conventional and learning-based methods in pose estimation. It is accurate and invulnerable to consider *De-Bias Net* and gravity alignment constraints for rotation estimation. And it is efficient and versatile to take into account both quadrotor dynamics and deep neural networks for translation estimation. The code and data will be open-sourced to the community.[3]

B. Future Work

Even though the proposed method makes use of deep neural networks to improve the accuracy of estimation, it still does not guarantee a perfect globally consistent yaw angle and position under long and highly maneuverable flight. The plausible solution is to provide an absolute observation, such as GPS or magnetic compass. Besides, some of the proposed modules can be supplied in GPS-free environments with exteroceptive sensors to improve the robustness of estimation. These additional sensors can be fused with the proposed method by means of optimization or filtering.

References

[1] C. Chen, X. Lu, A. Markham, and N. Trigoni, "IONet: Learning to cure the curse of drift in inertial odometry," in *Proc. Thirty-Second AAAI Conf. Artif. Intell.*, 2018.

[2] S. Herath, H. Yan, and Y. Furukawa, "RoNIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, new methods," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pp. 3146–3152, 2020.

[3] H. Yan, Q. Shan, and Y. Furukawa, "RIDI: Robust IMU double integration," in *Proc. Euro. Conf. Comput. vision (ECCV)*, 2018, pp. 621–636.

[4] M. Brossard, A. Barrau, and S. Bonnabel, "AI-IMU dead-reckoning," *IEEE Trans. Intell. Veh.*, vol. 5, no. 4 pp. 585–595, Dec. 2020.

[5] S. Sun, D. Melamed, and K. Kitani, "IDOL: Inertial deep orientation-estimation and localization," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 6128–6137.

[6] W. Liu *et al.*, "TLIO: Tight learned inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5653–5660, Oct. 2020. [Online]. Available: http://dx.doi.org/10.1109/LRA.2020.3007421

[7] J. Svacha, G. Loianno, and V. Kumar, "Inertial yaw-independent velocity and attitude estimation for high-speed quadrotor flight," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1109–1116, Apr. 2019.

[8] L. Bauersfeld, E. Kaufmann, P. Foehn, S. Sun, and D. Scaramuzza, "NeuroBEM: Hybrid aerodynamic quadrotor model," *RSS: Robot., Sci., Syst.*, to be published, doi: 10.15607/RSS.2021.XVII.042.

[9] B. Rao, E. Kazemi, Y. Ding, D. M. Shila, F. M. Tucker, and L. Wang, "CTIN: Robust contextual transformer network for inertial navigation," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 5413–5421.

[10] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2002, pp. 1788–1795.

[11] A. Vaswani *et al.*, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 5998–6008, 2017.

[12] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robot. Automat. Mag.*, vol. 19, no. 3, pp. 20–32, 2012.

[13] W. Khan and M. Nahon, "Toward an accurate physics-based UAV thruster model," *IEEE/ASME Trans. Mechatron.*, vol. 18, no. 4, pp. 1269–1279, 2013.

[14] R. Gill and R. D'Andrea, "Propeller thrust and drag in forward flight," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, 2017, pp. 73–79.

[15] R. Gill and R. D'Andrea, "Computationally efficient force and moment models for propellers in UAV forward flight applications," *Drones*, vol. 3, no. 4, 2019, Art. no. 77.

[16] B. Nisar, P. Foehn, D. Falanga, and D. Scaramuzza, "VIMO: Simultaneous visual inertial model-based odometry and force estimation," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2785–2792, Jul. 2019.

[17] Z. Ding, T. Yang, K. Zhang, C. Xu, and F. Gao, "VID-Fusion: Robust visual-inertial-dynamics odometry for accurate external force estimation," *IEEE Int. Conf. Robot. Automat.* 2021, pp. 14469–14475.

[18] J. Svacha, J. Paulos, G. Loianno, and V. Kumar, "IMU-based inertia estimation for a quadrotor using newton-euler dynamics," *IEEE Robot. Automat. Lett.*, vol. 5, no. 3, pp. 3861–3867, Jul. 2020.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[20] 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 *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*. IEEE, 2021, pp. 275–283.

[21] A. Antonini, W. Guerra, V. Murali, T. Sayre-McCord, and S. Karaman, "The blackbird UAV dataset," *Int. J. Robot. Res.*, vol. 39, no. 10-11, pp. 1346–1364, 2020.

[22] K. Zhang *et al.*, "The visual-inertial-dynamical multirotor dataset," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 7635–7641, doi: 10.1109/ICRA46639.2022.9811956.

[23] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 2520–2525.

[24] P. Geneva and G. Huang, "vicon2gt: Derivations and analysis," University of Delaware, Tech. Rep. RPNG-2020-VICON2GT, 2020. [Online]. Available: http://udel.edu/ghuang/papers/tr_vicon2gt.pdf

[25] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Trans. Autom. Control*, vol. 53, no. 5, pp. 1203–1218, Jun. 2008.

[26] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, "Estimation of IMU and MARG orientation using a gradient descent algorithm," in *Proc. IEEE Int. Conf. Rehabil. Robot.*, 2011, pp. 1–7.

[27] K. Zhang *et al.*. (2022) Supplementary material: Deep inertial quadrotor dynamical odometry. [Online]. Available: https://github.com/zhangkunyi/DIDO/blob/main/DIDO_SupMat.pdf

[28] S. Sabatelli, M. Galgani, L. Fanucci, and A. Rocchi, "A double-stage Kalman filter for orientation tracking with an integrated processor in 9-d IMU," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 3, pp. 590–598, Mar. 2013.

[3] https://zhangkunyi.github.io/DIDO/