A Lightweight Learning-based Visual-Inertial Odometry

Yingfu Xu and Guido C. H. E. de Croon *
Delft University of Technology

http://www.imavs.org/

ABSTRACT

In this paper, we propose a learning-based lightweight visual-inertial odometry (VIO) based on an uncertainty-aware pose network and an extended Kalman filter (EKF). The pose network serving as the VIO vision front-end predicts the relative motion of the camera between consecutive image frames and estimates the prediction uncertainty. The training of the pose network can be conducted without requiring ground-truth labels. The distributions of visual measurements are fused with inertial measurements by an EKF that is the VIO back-end. Evaluations show that the proposed VIO fails to outperform a state-of-the-art feature-point-based VIO solution in accuracy. But it has high time efficiency, translational motion estimation with metric scale, estimation of gravity direction, and generalization to new environments. So, unlike most works on learning-based visual ego-motion estimation in the literature, the proposed VIO can be directly deployed on an MAV. The comparative studies of supervision signals and forms of translational motion prediction provide insights that can contribute to future research.

1 INTRODUCTION

Ego-motion estimation of micro air vehicles (MAVs) is essential for autonomous flight and has been a major research topic in the robotics domain. Monocular cameras are often used for ego-motion estimation of lightweight MAVs in environments without stable GPS signals, because of their small size, lightweight, low energy consumption, and rich environmental information captured. After years of research, ego-motion estimation approaches that use vision in conjunction with an inertial measurement unit (IMU), *i.e.*, visual-inertial odometry (VIO), are widely applied to MAVs. Many mainstream VIO approaches, *e.g.* multi-state constraint Kalman filter (MSCKF) [1], use visual feature points as the vision processing front-end. Assuming the scene is stationary, information about the camera motion can be obtained by tracking the two-dimensional (2-d) pixel locations of the same feature point in multiple temporally consecutive images. Therefore, extracting a sufficient number of feature points and then tracking or matching them accurately across frames is the key to accurate ego-motion estimation. However, because of the dependence on appearance consistency and image gradient, visual feature points are susceptible to varying illumination and image blur.

As an alternative to feature points, researchers have been exploring deep learning for visual ego-motion estimation. Learning-based approaches train one or multiple artificial neural network(s) (ANN) to predict the relative pose [2, 3] between temporally consecutive images or visual correspondences that encode ego-motion, *e.g.*, optical flow [4, 5] and planar homography transformation [6]. A pose network can be trained by the ground-truth relative pose in supervised learning. But the acquisition of accurate camera position and orientation in the real world often requires external sensors such as motion capture systems. So the scenes of captured images for training are restricted to the range of the stationary motion capture sensors. Thus it limits the size of the training dataset and deteriorates the generalization capacity of the pose network. In contrast, self-supervised learning is an attractive approach for relieving the need for ground truth. The first self-supervised learning scheme of a pose network was proposed in [2], where a monocular depth network is trained simultaneously. It is sometimes referred to as self-supervised structure-from-motion (SfM) in the literature. According to the predicted dense depth and relative camera pose, a virtual view can be synthesized by reprojecting the world points into the image frame. The photometric error between the synthesized view and the actually captured image is the main supervision signal. This scheme has been further developed by many follow-up works and is adopted in this paper. A more detailed description is provided in Section 2.

It has been observed that ANNs show better robustness towards unfavorable conditions such as illumination change, motion blur, and dynamic scenes [3, 5]. There are learning-based approaches [4, 5] achieving better accuracy on the EuRoC MAV datasets [7] than a traditional approach [8]. But general challenges exist in learning-based approaches. Firstly, many works train and test on the same dataset without considering the network's generalization capability. Secondly, achieving high accuracy requires a significant amount of computational power, making such approaches unsuitable for computationally constrained MAVs. Thirdly, many works cannot be directly deployed on MAVs due to various limitations. For example, translational motion lacks scale in [5]. The scale is ambiguous in the case of self-supervised learning with monocular videos [2, 9, 10]. The scale is also unknown in the case of self-supervised VIO that processes IMU measurements with an ANN [11]. Because the loss function

*Email addresses: yingfu.xu.94@gmail.com, g.c.h.e.decroon@tudelft.nl

http://www.imavs.org/

has no constraint on the scale, the metric-scale accelerometer measurements can be scaled arbitrarily by the ANN. Another problem in using an IMU network is that, when the supervision signal only regularizes the relative pose [11], the gravity information encoded in the accelerometer measurements is not exploited. So the gravity direction remains unknown, though it is observable for a monocular visual-inertial system [12].

This paper aims to address the above-mentioned challenges by introducing a learning-based VIO. It can be directly deployed on an MAV due to its notable characteristics, such as high efficiency, translational motion estimation with metric scale, estimating the direction of gravity, and generalizing to new environments. The vision front-end is a pose network that infers the relative rotation and translation of the camera from an image pair. It is capable of estimating the prediction uncertainty, as is useful for the EKF-based VIO back-end. In addition, the network's training can be conducted with or without ground-truth labels. In the latter case, self-supervised learning schemes for pose and depth [9, 10] are adopted to first train "teacher" networks with high prediction accuracy. The pose predictions of the teacher networks serve as the learning target of the lightweight "student" pose network that is the VIO vision front-end. The effects of three elements of the pose network design are evaluated and analyzed. They are the supervision signal (with *vs.* without ground truth), the scale of translational motion prediction (with metric scale *vs.* direction only), and with *vs.* without fine-tuning on the target dataset.

2 TEACHER NETWORKS

2.1 Improved Self-Supervised SfM

The mission of the teacher networks is to perform self-supervised learning and provide accurate relative pose predictions that serve as the learning targets in the training of a lightweight pose network. We take Monodepth2 [9] and its open-sourced code as the base of the teacher networks and adopt two additional methods to achieve higher pose prediction accuracy. The first is a loss term L_{geo} that constrains the depth predictions, proposed by [10]. It is based on the geometry consistency of the 3-d positions of the world points. The second is to conduct the pose network inference multiple times by iterative view synthesis that relies on the depth map prediction, proposed by [13]. Instead of predicting the total transformation by a single inference, the pose network incrementally refines the relative pose prediction by inferring from input images that are more and more similar.

The self-supervised learning loss function of the teacher networks is shown in Eq. (1). V denotes the set of all pixels. L_{photo} is referred to as the reprojection-based loss in some works and in this paper. L_{photo} combines the $L1$ loss of pixel-wise photometric error and the structured similarity index measure (SSIM) loss, with α = 0.85. N is the total number of iterations of pose network inference. The geometry consistency loss L_{geo} is computed only once using the final pose prediction after all iterations. Hyperparameters λ_{geo} and λ_i are scalar weights of loss terms. $\mathcal{M}_{\text{auto}}(\cdot)$ stands for applying the auto-masking strategies of Monodepth2 to the pixels. The smoothness loss for depth prediction is the edge-aware smoothness implemented by the open-sourced code of Monodepth2. We omit its expression in Eq. (1). An interested reader can refer to [9, 10, 13] for details of the loss function.

$$\begin{aligned} L &= \lambda_{\text{geo}} \cdot L_{\text{geo}} + \sum_{i=1}^{N} \lambda_i \cdot L_{\text{photo},i} \\ L_{\text{photo},i} &= \frac{1}{|V|} \sum_{k \in V} \mathcal{M}_{\text{auto}}(\frac{\alpha}{2}(1 - \text{SSIM}(I_t, \widetilde{I}_{s,i})_k) \\ &\quad + (1-\alpha) \cdot |I_{t,k} - \widetilde{I}_{s,i,k}|) \end{aligned} \tag{1}$$

Instead of predicting depth maps at four scales as Monodepth2 does, we predict a single depth map at the full resolution. It results in higher accuracy in both depth and pose predictions. To better learn how to handle fast motion, our implementation uses a bigger temporal step in loading image snippets. There is 25% possibility to load I_{i-2}, I_i, and I_{i+2} and 75% possibility to load the neighboring images I_{i-1}, I_i, and I_{i+1}.

2.2 Datasets and Network Training

TartanVO [5] and Droid-SLAM [4] are two of a few works that achieve good performance of cross-dataset generalization. Both of them use TartanAir [14] dataset for training. We take most of the images (296,899) for training and a small number (1,522) for in-domain testing. Testing images are randomly sampled from all the sequences.

A pose network infers relative pose from raw images, so camera intrinsics affect the prediction. Such a pose network requires the input image to always have the specific image resolution and camera intrinsics. Therefore, in cross-dataset testing and real-world deployment, the images need to be preprocessed to be the same as training images in terms of resolution and camera intrinsics. In this paper, input images to a pose network are required to be grey-scale and have a resolution of 192 $\times$ 352 and intrinsics $f_x = 176$, $f_y = 176$, $c_x = 176$, $c_y = 96$. This setting fits both the training set (TartanAir) and the testing set (EuRoC MAV dataset [7]). Images of TartanAir lose pixels near the top and bottom edges after being transformed to this setting. But if we transform the images of EuRoC to have the same resolution and camera intrinsics as TartanAir, the characteristics of the lens and optic sensor of the camera used in collecting the EuRoC dataset lead the images to have black curving edges. This phenomenon reveals the lack of cross-camera generalization of pose networks. A solution based on preprocessing the input optical flow map is proposed in [14], while the pose network in this paper uses raw images as input. So the cross-camera generalization is not solved.

http://www.imavs.org/

The networks in this section are trained on the TartanAir training set for 100 epochs. We utilized the 1 cycle learning rate policy [15] with the maximum learning rate of 2.5e-4 in self-supervised learning. The architectures of the pose and depth networks are based on ResNet-18, the same as Monodepth2. The number of iterations $N = 3$. The weights of the loss for each iteration are $\lambda_0 = 0.25$, $\lambda_1 = 0.5$, and $\lambda_2 = 1.0$. The weight for geometry consistency $\lambda_{\text{geo}} = 0.2$. The batch size is 32.

We compare the pose prediction accuracy of the iterative teacher networks trained by the loss function (Eq. (1)) with the original Monodepth2 in Table 1. The accuracy metric is the average of the norms of error vectors, $e = \frac{1}{N}\sum ||\boldsymbol{v}_{\text{GT}} - \boldsymbol{v}_{\text{Pred.}}||_2$. The rotation is expressed by axis-angle and implemented in the open-sourced code of Monodepth2. The predicted translation and the ground-truth translation are normalized to 3-d unit vectors before calculating the error because the scale of the translation prediction is ambiguous. When the translation predictions are used as the learning targets of the uncertainty-aware pose network, they are also normalized.

Network	TartanAir Rot.	TartanAir Trans.	EuRoC Rot.	EuRoC Trans.
ours	**1.99e-03**	**8.77e-02**	3.09e-03	2.39e-01
ours-FT	-	-	**1.73e-03**	**1.62e-01**
Monodepth2	2.21e-03	1.23e-01	4.08e-03	2.89e-01
Monodepth2-FT	-	-	3.18e-03	2.48e-01

Table 1: Average pose prediction errors on TartanAir testing set and EuRoC testing set. "FT" indicates that the networks are fine-tuned on the EuRoC training set based on the parameters trained on the TartanAir training set. **Bold** represents the best.

As shown in Table 1, we perform an in-domain test on the TartanAir testing set and an out-of-domain test on a part of the EuRoC MAV dataset [7]. There are 11 sequences in the EuRoC dataset. The EuRoC training set is made of all sequences except for *V103*, *V203*, and *MH05*. These three sequences are used for testing. From the first and third rows of Table 1, we can see that our networks have smaller average errors in pose predictions on both tests.

We finetune the networks trained on the TartanAir dataset on the training set of EuRoC and test them on the three testing sequences, as shown in the second and fourth rows of Table 1. Although TartanAir is a large-size dataset with a wide distribution, we still observe that fine-tuning brings obvious improvement in pose prediction accuracy.

3 VIO BASED ON POSE NETWORK AND EKF

In the previous section, we introduce the methodology to train the self-supervised teacher network. As already explained in the introduction, the teacher networks are not suitable for the ego-motion estimation of an MAV. Specifically, the pose prediction of the teacher networks relies on one forward pass of the depth network and multiple forward passes of the pose network, which require considerable computational power. In addition, the translational predictions have an ambiguous scale and the gravity direction is not estimated.

In this section, we introduce a computationally efficient VIO that can estimate metric-scale translational motion and the gravity direction. Its vision front-end is a pose network. It performs a single forward pass for a pose prediction. The back-end is a simple EKF. We train several pose networks using two supervision signals and compare the resulting VIO accuracy. They are, respectively, the ground-truth poses and the teacher networks' pose predictions. When using the pose predictions of teacher networks for training, the front-end pose network is a student network, and the whole learning pipeline does not require ground-truth labels of the pose. Different from the teacher networks, the front-end pose network not only predicts the relative pose but also estimates the uncertainty of its prediction. So it is called an uncertainty-aware pose network.

3.1 Uncertainty-Aware Pose Network

The architecture of the pose network is based on the pose network used by Monodepth2. It has an encoder based on ResNet-18. We modify the architecture of the decoder part to enable it for uncertainty estimation. The output tensor of the last convolutional layer is the input to two fully connected (FC) subnetworks. One FC network predicts the mean value, and the other predicts the *aleatoric* uncertainty that reflects the noise inherent in the network input. It is referred to as predictive uncertainty in [6] and this paper. The two FC networks have the same architecture of two FC layers. There is a 5% dropout for the input of each FC layer, to estimate the *epistemic* uncertainty using MC-Dropout [16]. *Epistemic* uncertainty captures the ignorance about the ideal network that maps noiseless input to the desired output. It is called empirical uncertainty in [6] and this paper. The total uncertainty is the sum of predictive uncertainty and empirical uncertainty. The uncertainty estimation of the front-end uncertainty-aware pose network follows the same methodologies as in [6]. An interested reader can read [6] for more details. We introduce the most important steps in the following.

Treating the observed pose as a sample from a Gaussian distribution, the loss function for learning predictive uncertainty is the negative log-likelihood (NLL) loss, as shown in Eq. (2). This loss function is used for the same purpose in [17].

$$L_{\text{NLL}} = \sum_{n=1}^{6} \frac{1}{2\sigma_{n,\text{pred.}}^2} \|T_n - \mu_n\|^2 + \frac{1}{2}\log(\sigma_{n,\text{pred.}}^2) \quad (2)$$

μ_n is the prediction of the mean value of the relative pose vector. $\sigma_{n,\text{pred.}}^2$ is the variance prediction corresponding to the predictive uncertainty. n indexes over the six dimensions of

http://www.imavs.org/

relative pose (3-d rotation and 3-d translation). T_n denotes the learning target of the mean prediction μ_n. T_n can be the ground-truth relative pose or the pose prediction of the trained teacher networks. The pose network can be trained to predict translational motion with the metric scale. It can also be trained to predict normalized translation expressed by a unit vector that indicates the motion direction without scale. We implement both of them and compare them in Subsection 3.3. When the pose network is trained to predict translational direction, its prediction of the mean translation is normalized. The translational part of T_n is normalized accordingly.

MC-Dropout [16] requires multiple forward passes to sample from the distributions of network parameters. m indexes over the forward passes. We set the number of MC-Dropout sampling M=8. After each forward pass, a mean prediction $\mu_{n,m}$ is obtained, as well as a variance prediction of predictive uncertainty $\sigma^2_{\text{pred.},n,m}$. As Eq. (3) shows, the variance of empirical uncertainty $\sigma^2_{\text{emp.},n}$ is calculated empirically from the multiple mean predictions $\mu_{n,m}$. The total variance σ^2_n is the sum of the empirical variance $\sigma^2_{\text{emp.},n}$ and the average of predictive variances $\sigma^2_{\text{Avg., pred.},n}$.

$$\begin{aligned}
\sigma^2_n &= \sigma^2_{\text{Avg., pred.},n} + \sigma^2_{\text{emp.},n} \\
\sigma^2_{\text{Avg., pred.},n} &= \frac{1}{M}\sum_{m=1}^{M} \sigma^2_{\text{pred.},n,m} \\
\sigma^2_{\text{emp.},n} &= \frac{1}{M}\sum_{m=1}^{M} (\mu_{n,m} - \mu_n)^2, \quad \mu_n = \frac{1}{M}\sum_{m=1}^{M} \mu_{n,m}
\end{aligned} \tag{3}$$

3.2 EKF-based Back-end

The VIO back-end is an EKF. It is a simplified variant of the EKF-based back-end of the robocentric VIO [18]. The robocentric VIO keeps a window of the previous camera poses in the state vector. Our simplified variant only estimates the relative pose between the current IMU frame and the local frame of reference. The EKF state vector is defined as

$$\begin{aligned}
\boldsymbol{x} := [&{}^{R_k}\boldsymbol{p}_G, \quad {}^{R_k}_{G}\boldsymbol{q}, \quad \boldsymbol{g}_{R_k}, \\
&{}^{I_t}\boldsymbol{p}_{R_k}, \quad {}^{I_t}_{R_k}\boldsymbol{q}, \quad \boldsymbol{v}_{I_t}, \quad \boldsymbol{b}_a, \quad \boldsymbol{b}_g].
\end{aligned} \tag{4}$$

I_t is the current IMU frame at time t. When a new image has been captured, the new reference frame R is set to be the same as the I_t at the time. R_k is the current reference frame. It is the kth reference frame since the VIO is initialized. G stands for the global frame. It is the first reference frame R_0, *i.e.*, the IMU frame when the first image is captured after the initialization of the VIO. ${}^{R_k}\boldsymbol{p}_G$ is a translation vector pointing from the origin of G to the origin of R_k, expressed in R_k. It is about the global position of R_k. ${}^{I_t}\boldsymbol{p}_{R_k}$ is a translation vector pointing from the origin of R_k to the origin of I_t, expressed in I_t. It is about the local position of I_t relative to R_k. ${}^{R_k}_{G}\boldsymbol{q}$ is the Hamilton quaternion reflecting the relative rotation between G and R_k. ${}^{I_t}_{R_k}\boldsymbol{q}$ reflects the relative rotation between R_k and I_t. $\boldsymbol{g}_{R_k}$ indicates the gravity vector expressed in R_k. $\boldsymbol{v}_{I_t}$ is the translational velocity of the IMU expressed in I_t. $\boldsymbol{b}_a$ and $\boldsymbol{b}_g$ are respectively the additive bias on accelerometer and gyroscope.

As shown in Eq. (5), IMU measurements are modeled as the sum of the desired actual value ($\hat{\boldsymbol{a}}$ and $\hat{\boldsymbol{\omega}}$), additive bias, and white Gaussian noise ($\boldsymbol{w}_a$ and $\boldsymbol{w}_g$).

$$\boldsymbol{a}_m = \hat{\boldsymbol{a}} + \boldsymbol{b}_a + \boldsymbol{w}_a, \quad \boldsymbol{\omega}_m = \hat{\boldsymbol{\omega}} + \boldsymbol{b}_g + \boldsymbol{w}_g \tag{5}$$

$$\begin{aligned}
{}^{I_t}\dot{\boldsymbol{p}}_{R_k} &= -[\hat{\boldsymbol{\omega}}]_\times \cdot {}^{I_t}\boldsymbol{p}_{R_k} + \boldsymbol{v}_{I_t} + \boldsymbol{w}_{\boldsymbol{p}}, \\
\dot{\boldsymbol{v}}_{I_t} &= -[\hat{\boldsymbol{\omega}}]_\times \cdot \boldsymbol{v}_{I_t} + \hat{\boldsymbol{a}} + \boldsymbol{R}({}^{I_t}_{R_k}\boldsymbol{q})^T \cdot \boldsymbol{g}_{R_k}, \\
{}^{I_t}_{R_k}\dot{\boldsymbol{q}} &= \frac{1}{2}{}^{I_t}_{R_k}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \hat{\boldsymbol{\omega}} \end{bmatrix}, \\
\dot{\boldsymbol{b}}_a &= \boldsymbol{w}_{\boldsymbol{b}_a}, \quad \dot{\boldsymbol{b}}_g = \boldsymbol{w}_{\boldsymbol{b}_g}.
\end{aligned} \tag{6}$$

Eq. (6) shows the IMU-driven state dynamics ($\dot{\boldsymbol{x}}$). $[\hat{\boldsymbol{\omega}}]_\times$ represents the skew-symmetric matrix associated with $\hat{\boldsymbol{\omega}}$. $\boldsymbol{w}_{\boldsymbol{p}}$ is the process noise in position integration. $\boldsymbol{R}({}^{I_t}_{R_k}\boldsymbol{q})$ is a transformation function from ${}^{I_t}_{R_k}\boldsymbol{q}$ to *SO3* rotation matrix that maps a vector expressed in I_t to its expression in R_k. $\otimes$ denotes quaternion product. We utilize the techniques introduced in [19] for quaternion-related calculation.

The filter states in Eq. (6) are propagated by the IMU measurements until a new image I_{k+1} is captured. The pair of I_{k+1} and the previous image I_k are the inputs of the uncertainty-aware pose network that is introduced in Subsection 3.1. The network outputs $\boldsymbol{\mu_t}$, $\boldsymbol{\sigma^2_t}$, $\boldsymbol{\mu_\theta}$, and $\boldsymbol{\sigma^2_\theta}$. They are the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}$ of translation $\boldsymbol{t}$ and rotation $\boldsymbol{\theta}$, respectively.

$$\begin{aligned}
\boldsymbol{z}_{\boldsymbol{\mu_t}} &= R^C_I \cdot (\boldsymbol{t}_{IC} + {}^{I_t}\boldsymbol{p}_{R_k} - \boldsymbol{R}({}^{I_t}_{R_k}\boldsymbol{q})^T \cdot \boldsymbol{t}_{IC}) + \boldsymbol{w}_{\boldsymbol{t}}, \\
\boldsymbol{z}_{\boldsymbol{\mu_\theta}} &= R^C_I \cdot \boldsymbol{\theta}({}^{I_t}_{R_k}\boldsymbol{q}) + \boldsymbol{w}_{\boldsymbol{\theta}}.
\end{aligned} \tag{7}$$

The measurement equations are shown in Eq. (7). R^C_I is the rotation matrix from the IMU frame to the camera frame. $\boldsymbol{t}_{IC}$ is the translation vector points from the IMU to the camera, expressed in the IMU frame. $\boldsymbol{\theta}(\cdot)$ converts a quaternion to an axis-angle expression. Elements of $\boldsymbol{\sigma^2_t}$ and $\boldsymbol{\sigma^2_\theta}$ are used as the diagonal elements of the measurement noise matrix. After the measurement update, the *a posterior* relative pose estimation ${}^{I_t}\hat{\boldsymbol{p}}_{R_k}$ and ${}^{I_t}_{R_k}\hat{\boldsymbol{q}}$ is composed to the global pose, as shown in Eq. (8).

$$\begin{aligned}
{}^{R_{k+1}}_{G}\boldsymbol{q} &= {}^{I_t}_{R_k}\hat{\boldsymbol{q}} \otimes {}^{R_k}_{G}\boldsymbol{q}, \\
{}^{R_{k+1}}\boldsymbol{p}_G &= \boldsymbol{R}({}^{I_t}_{R_k}\hat{\boldsymbol{q}})^T \cdot {}^{R_k}\boldsymbol{p}_G + {}^{I_t}\hat{\boldsymbol{p}}_{R_k}
\end{aligned} \tag{8}$$

The current IMU frame I_t becomes the new reference frame R_{k+1}. The expression of the gravity vector $\boldsymbol{g}_{R_{k+1}}$ in R_{k+1} is calculated as $\boldsymbol{g}_{R_{k+1}} = \boldsymbol{R}({}^{I_t}_{R_k}\hat{\boldsymbol{q}})^T \cdot \hat{\boldsymbol{g}}_{R_k}$. ${}^{I_t}\boldsymbol{p}_{R_{k+1}}$ and ${}^{I_t}_{R_{k+1}}\boldsymbol{q}$ are set to a zero vector and a unit quaternion whose vector part is a zero vector, respectively. Their corresponding elements in the covariance matrix are set to zeros too. Readers who

Table 2: Accuracy of the proposed method (top group, rows 1 to 6) compared with the baseline method (bottom group, rows 7 and 8). *V101* to *MH05* shown in the eleven columns from the right are the names of flight sequences of the EuRoC MAV dataset. The data below the sequence names show the root-mean-square errors (RMSE) of the absolute translation errors (ATEs) of the estimated trajectories. **Bold** represents the overall best and underline represents the best of the proposed method.

ID	FT[1]	GT[2]	MS[3]	V101	V102	V103	V201	V202	V203	MH01	MH02	MH03	MH04	MH05
①				4.24	2.34	3.04	4.05	4.97	5.88	6.42	3.92	4.79	14.1	4.04
②	✓			-	-	2.23	-	-	4.33	-	-	-	-	4.25
③		✓		3.48	2.51	2.74	4.07	5.49	7.73	5.60	3.93	6.18	17.7	19.8
④	✓	✓		-	-	2.42	-	-	7.99	-	-	-	-	8.16
⑤		✓	✓	1.23	2.86	3.74	1.86	4.37	4.27	7.24	5.06	**4.26**	3.24	3.68
⑥	✓	✓	✓	-	-	1.55	-	-	4.36	-	-	-	-	3.04
⑦	MSCKF (51 pts)			0.16	0.13	0.12	245	0.13	**0.16**	38.0	5.20	130	2.35	**1.00**
⑧	MSCKF (199 pts)			**0.09**	**0.09**	**0.11**	**0.12**	**0.10**	0.20	**0.34**	**0.24**	58.5	**0.65**	1.54

[1] Fine-tuning (FT) the pose network on the training sequences of the EuRoC dataset.
[2] Using ground-truth (GT) labels as the learning targets in the NLL loss. If not using GT, the learning targets are the predictions of the self-supervised teacher networks.
[3] The pose network learns to predict translational motion with the metric scale (MS). Otherwise, the prediction is the normalized translation, a unit vector.

http://www.imavs.org/

are interested in the filter design can refer to [18] for more details.

For the EKF, when $\boldsymbol{\mu_t}$ is the normalized translation, the propagated camera translation is normalized accordingly. The measurement equation of translational motion is then

$$\boldsymbol{z}_{\boldsymbol{\mu_t}} = R_I^C \cdot \frac{\boldsymbol{t}_{IC} + {}^{I_t}\boldsymbol{p}_{R_k} - \boldsymbol{R}({}^{I_t}_{R_k}\boldsymbol{q})^T \cdot \boldsymbol{t}_{IC}}{\|\boldsymbol{t}_{IC} + {}^{I_t}\boldsymbol{p}_{R_k} - \boldsymbol{R}({}^{I_t}_{R_k}\boldsymbol{q})^T \cdot \boldsymbol{t}_{IC}\|} + \boldsymbol{w_t} \quad (9)$$

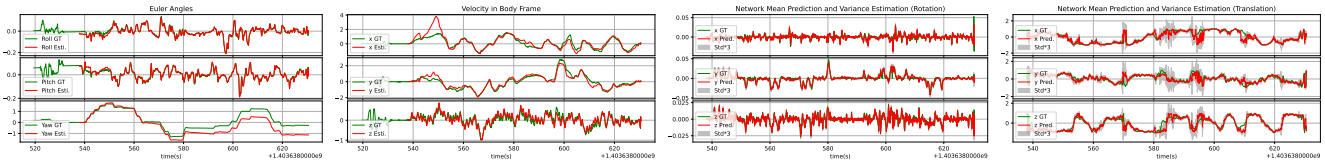
3.3 Evaluation

In the previous part of this section, we have successively introduced the vision front-end which is the uncertainty-aware pose network and the EKF-based robocentric back-end. The VIO system consisting of them is referred to as PoseNet-VIO in this paper. In this subsection, PoseNet-VIO is evaluated on the EuRoC MAV dataset [7]. The PoseNet-VIO variants in Table 2 have the same parameters in the EKF-based back-end. Their vision front-ends, the pose networks, are different. They are trained by the NLL loss (Eq. (2)) with different learning targets. The learning targets of ① and ② are the pose predictions of the self-supervised teacher networks. The remaining four networks learn from the ground truth. The translation prediction of ① to ④ in Table 2 are normalized to a unit vector that indicates the direction of translational motion. In contrast, ⑤ and ⑥ predict translational motion with the metric scale that is learned from the ground-truth labels of the relative poses. Since we use monocular videos to train the teacher networks, their prediction of translational motion has an ambiguous scale. Therefore, the pose networks trained by the teacher networks can only have normalized translational motion predictions. ②, ④, and ⑥ are fine-tuned on the EuRoC training datasets based on ①, ③, and ⑤, respectively. The teacher networks of ② are fine-tuned on the EuRoC training set based on the parameters of the teacher networks of ①. PoseNet-VIO with a fine-tuned network is evaluated on the EuRoC testing sequences. All the front-end pose networks are trained on the TartanAir dataset for 100 epochs. Fine-tuning on the EuRoC training set lasts for 50 epochs.

The root-mean-square error (RMSE) of absolute translation errors (ATE) is the metric of VIO accuracy. The alignment of the estimated trajectory and the ground-truth trajectory has 4-DoF (yaw and 3-d translation). The calculation is conducted by [20]. PoseNet-VIO is compared with a state-of-the-art (SOTA) VIO MSCKF [1] implemented by [21]. The C++ code of PoseNet-VIO is implemented based on the open-sourced code of [21]. Here we only compare with one SOTA VIO since the gap in accuracy is big, as shown in Table 2. The comparison of MSCKF with other VIO solutions can be found in [22].

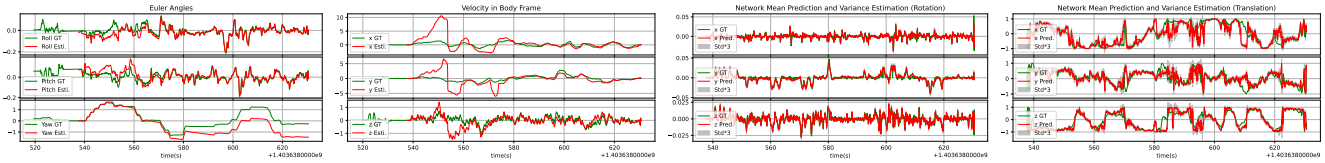
At the beginning of the five EuRoC sequences collected in the machine hall, the MAV was moved by a human operator and then stayed stationary for a while before taking off. For a PoseNet-VIO whose front-end pose network has normalized translation predictions, the lack of translational motion leads to drift. So, when testing PoseNet-VIO variants ① to ④, the starting time points of the *Machine Hall* (*MH*) sequences are about one second before the takeoff. For MSCKF, if there was a significant drift in the beginning, we did the same.

A design target of PoseNet-VIO is high efficiency. Thus it has non-iterative network inference and a basic EKF. The time consumption is constant. But since MSCKF uses feature points, the number of points affects the time consumption. The time consumptions of PoseNet-VIO and MSCKF are measured during the tests on *MH05* sequence of EuRoC. For PoseNet-VIO, the average total time cost is 7.811 ms, of



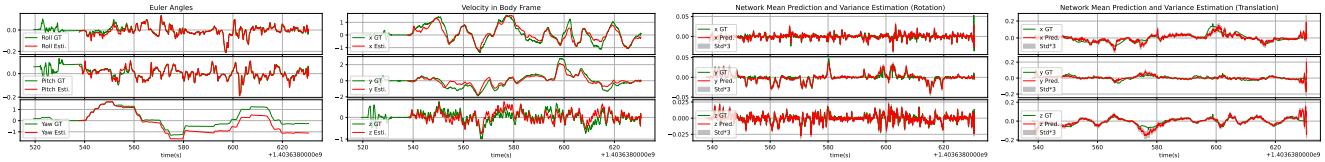
(a) VIO Euler Angles Estimation (rad). (b) VIO Velocity Estimation Expressed in MAV Body Frame (m/s). (c) Pose Network Rotation (Axis–Angle) Predictions (rad). (d) Pose Network Normalized Translation Predictions.

Figure 1: Results of PoseNet-VIO variant ①, tested on *MH05*. The two subplots on the left show the EKF *a posterior* estimations (VIO outputs) of the MAV attitude and translational velocity. The attitude is relative to the global frame whose z-axis is opposite to the gravity direction. The two subplots on the right show the outputs (mean prediction and uncertainty estimation of the relative pose) of the front-end pose network.



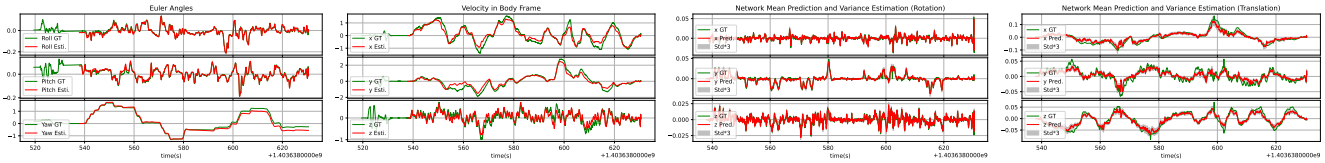
(a) VIO Euler Angles Estimation (rad). (b) VIO Velocity Estimation Expressed in MAV Body Frame (m/s). (c) Pose Network Rotation (Axis–Angle) Predictions (rad). (d) Pose Network Normalized Translation Predictions.

Figure 2: Results of PoseNet-VIO variant ③, tested on *MH05*.



(a) VIO Euler Angles Estimation (rad). (b) VIO Velocity Estimation Expressed in MAV Body Frame (m/s). (c) Pose Network Rotation (Axis–Angle) Predictions (rad). (d) Pose Network Translation Predictions (m).

Figure 3: Results of PoseNet-VIO variant ⑤, tested on *MH05*.



(a) VIO Euler Angles Estimation (rad). (b) VIO Velocity Estimation Expressed in MAV Body Frame (m/s). (c) Pose Network Rotation (Axis–Angle) Predictions (rad). (d) Pose Network Translation Predictions (m).

Figure 4: Results of PoseNet-VIO variant ⑥, tested on *MH05*.

http://www.imavs.org/

http://www.imavs.org/

which 6.036 ms is for network inference. For the default setting of MSCKF (⑧ in Table 2), the average total time cost is 17.415 ms, with 7.583 ms for processing feature points. The average number of points is around 199. We modified the number of feature points to track to a smaller number than the default to make the average of the total time consumption of processing a frame close to the time consumption of PoseNet-VIO. When the average number of maintained points in an image frame is around 51 (⑦ in Table 2), the average total time cost is 7.933 ms, with 3.821 ms for feature point tracking and detection.

In the sequences *V201*, *MH01*, and *MH03*, MSCKF ⑦ drifts after very slow motion, which leads to big trajectory errors. But in general, as shown in Table 2, PoseNet-VIO is far less accurate than MSCKF. Comparing the variants of PoseNet-VIO, a noticeable phenomenon is that networks predicting translational motion with metric scale lead to better VIO accuracy than networks predicting normalized translation. Comparing the ATEs of ③ and ⑤, we can see that ⑤ has advantages in seven out of the eleven sequences. This is further evidenced by Fig. 2 (b) and Fig. 3 (b), where the estimated velocity of ⑤ is more accurate. Without utilizing the knowledge of objects' sizes in the scene, the metric scale is unobservable for a monocular video. The metric-scale translation prediction is based on the knowledge of the training set. It can be problematic when generalizing to a different dataset. But Fig. 3 (d) shows that the errors of translational motion predictions are acceptable. The generalization is fine. We attribute it to the large number and wide distribution of the training samples. Subplot (b) for VIO velocity estimations of Fig. 2 shows that using normalized translation (③) leads to big drifts in the beginning before the estimations converge after around 35 seconds. For ①, it takes around 10 seconds for the velocity estimation to converge, as shown in subplot (b) of Fig. 1. While using metric-scale translation does not have this problem, as shown in the two subplots (b) of Fig. 3 and Fig. 4. In summary, given the current design, using normalized translation leads to worse VIO accuracy in cross-dataset evaluation.

Another phenomenon worth noticing is the effect of fine-tuning. Fine-tuning brings better accuracy for all the variants, indicating that although we have a big-scale and widely distributed training set, the generalization capacity can still be improved. When the translation predictions have the metric scale, we can see better accuracy in all subplots of Fig. 4 than Fig. 3. Especially for the pose network's predictions of translational motion shown in subplot (d), the predictions are more accurate after fine-tuning. The possible reason is that the network further learns the metric scale of the objects captured in the EuRoC training set.

PoseNet-VIO variant ① interests us most because the training does not use ground-truth labels or in-domain data. Given the not-excellent but acceptable accuracy in Euler angle estimation and velocity estimation after the convergence, as shown in Fig. 1, we think this PoseNet-VIO variant can act as an attitude and velocity estimator and be used for short-term navigation. From Table 2, Fig. 1, and Fig. 2, we notice that using the ground-truth poses in the training of the front-end pose network (③) does not improve the VIO accuracy over using the predictions of the self-supervised teacher networks (①). In eight out of eleven sequences, PoseNet-VIO variant ① has better accuracy. The accuracy of network predictions of rotation has no clear difference, as shown in the two subplots (c) of Fig. 1 and Fig. 2. But for the translation prediction shown in the two subplots (d), ① performs better than ③, especially in terms of uncertainty estimation. The three-time standard deviations of ① better reflect the errors of the mean predictions. This phenomenon indicates that self-supervised teacher networks are powerful replacements for ground-truth labels.

As shown in the subplot (c) of Fig. 1 to 4, the network predictions of relative rotation are well aligned with the ground truth. The translation predictions are relatively worse and noisier. Around 585, 595, and 610 seconds, translation predictions are obviously inaccurate, as shown in subplot (d) of Fig. 1. From subplot (b), we can see that the velocity is slow during those time slots. Slow velocity is a possible trigger of inaccurate translation prediction. More research is required to enable the network that predicts translation direction to better cope with slow motion. About potential ways of improving the PoseNet-VIO, one way is a VIO back-end that utilizes network predictions of multiple time steps instead of only the newest network prediction (current design). Another way is to learn the metric scale from the self-supervised teacher networks. It requires the teacher networks to be trained on stereo videos.

4 CONCLUSIONS

In this paper, we proposed the PoseNet-VIO based on an uncertainty-aware pose network and an EKF. The accuracy of PoseNet-VIO is worse than mainstream VIO solutions, but it may act as an efficient attitude and velocity estimator for short-term navigation. Based on the cross-dataset evaluation and the comparison between different types of supervision, we have the following findings. a) The simulation-to-reality generalization capacity of the pose network is generally satisfactory, also for the metric-scale translation prediction. b) Training an uncertainty-aware pose network using the predictions of self-supervised teacher networks leads to rivaling VIO accuracy to using ground-truth pose. c) Metric-scale translational motion predictions produce better VIO accuracy than normalized translational direction predictions. These findings can be valuable to future research on learning-based visual ego-motion estimation.

REFERENCES

[1] Mingyang Li and Anastasios I Mourikis. High-precision, consistent ekf-based visual-inertial odome-

http://www.imavs.org/

try. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[2] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1851–1858, 2017.

[3] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks. *The International Journal of Robotics Research*, 37(4-5):513–542, 2018.

[4] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

[5] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Tartanvo: A generalizable learning-based vo. In *Conference on Robot Learning*, pages 1761–1772. PMLR, 2021.

[6] Yingfu Xu and Guido CHE de Croon. Cuahn-vio: Content-and-uncertainty-aware homography network for visual-inertial odometry. *arXiv preprint arXiv:2208.13935*, 2022.

[7] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163, 2016.

[8] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[9] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3828–3838, 2019.

[10] Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. *Advances in neural information processing systems*, 32, 2019.

[11] Yasin Almalioglu, Mehmet Turan, Muhamad Risqi U Saputra, Pedro PB de Gusmão, Andrew Markham, and Niki Trigoni. Selfvio: Self-supervised deep monocular visual–inertial odometry and depth estimation. *Neural Networks*, 150:119–136, 2022.

[12] Guoquan Huang. Visual-inertial navigation: A concise review. In *2019 international conference on robotics and automation (ICRA)*, pages 9572–9582. IEEE, 2019.

[13] Mehrdad Hosseinzadeh, Ramin Fahimi, Yang Wang, et al. Unsupervised learning of camera pose with compositional re-estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 11–20, 2020.

[14] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020.

[15] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

[16] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

[18] Zheng Huai and Guoquan Huang. Robocentric visual-inertial odometry. *The International Journal of Robotics Research*, 41(7):667–689, 2022.

[19] Joan Sola. Quaternion kinematics for the error-state kalman filter. *arXiv preprint arXiv:1711.02508*, 2017.

[20] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251. IEEE, 2018.

[21] Patrick Geneva, Kevin Eckenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. Openvins: A research platform for visual-inertial estimation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4666–4672. IEEE, 2020.

[22] Jeffrey Delmerico and Davide Scaramuzza. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 2502–2509. IEEE, 2018.