Unsupervised Balanced Covariance Learning for Visual-Inertial Sensor Fusion

Youngji Kim¹, Sungho Yoon², Sujung Kim³, and Ayoung Kim¹*

Abstract—Incorporating multi-sensor, in filter-based as well as graph-based simultaneous localization and mapping (SLAM), relies on the uncertainties involved in each measurement. Proper covariance estimation is thus critical to balance confidence levels among sensors. Despite its importance, traditional covariance approximation mostly relied on first order derivative or fixed measurement covariance and therefore tended to be error-prone, and even heuristic. Recently, deep learning for uncertainty estimation yielded meaningful performance, but applied to a single sensor in a supervised manner. Unlike the traditional supervised manner, we introduce an unsupervised loss for uncertainty modeling, to learn uncertainty without needing ground truth covariance as a label. Most important, we overcome the limitation of learning a single sensor’s uncertainty by introducing a way of balancing uncertainty between different sensor modalities. In doing so, we alleviate the uncertainty balancing issue between sensors that has often been encountered in the multi-sensor SLAM application. Targeting covariance learning for visual odometry, particularly with regard to the integration of inertial sensors, the proposed uncertainty learning method was validated in a visual-inertial odometry application over the public datasets under artificial visual and inertial degradations to mimic harsh environment.

I. INTRODUCTION AND RELATED WORKS

The state of a robot is usually modeled as a Gaussian distribution with a mean and variance. A reliable state estimator focuses on the mean as well as the variance, but the importance of uncertainty is sometimes overlooked, as the performance of the estimator is measured only by the mean values. Variance is as important as mean values, as shown in the examples that utilize uncertainty for practical robotics applications.

Traditionally, a common approach to choosing reliability of visual measurements in SLAM was to use constant covariances [1], [2]. By ignoring measurement uncertainties, SLAM performance relied mostly on optimization. This method was prone to error when the measurements change dynamically due to adverse conditions such as dynamic objects and inconsistent lighting on camera. A more advanced information-theoretic approach was based on Cramer Rao Lower Bound (CRLB) [3], which is used in some visual odometry (VO) approaches such as [4]. In their work, CRLB is computed by inverting the Fisher information matrices derived from the photometric alignment and multiplying an empirical scale factor to make it a conservative lower bound. For vision-based motion estimation, first-order approximation in [5] has been widely applied with some efforts to improve the estimation of relative pose uncertainty. For example, [6] considered camera calibration uncertainty in addition to the propagated relative pose uncertainty.

Meanwhile, learning-based uncertainty estimation for sensor measurement has become a game-changer by allowing for the consideration of data uncertainty. CELLO [7] was a pioneering work that suggested the approach of learning sensor covariance through likelihood optimization. After CELLO, variants such as CELLO-EM [8] for learning without ground truth labels and CELLO-3D [9] for 3D point cloud registration have emerged.

Recently, deep learning has simplified the process of training and inferring covariances enormously. Whereas CELLO and its variants required domain knowledge to specify feature space, rendering the application of complex sensor measurements like a camera-based VO inevitably challenging, deep learning alleviated this problem by substituting hand-coded features with deep networks. Liu et al. [10] proposed a deep learning-based uncertainty estimation for VO, DICE. They designed a convolutional neural network that has the image as input and the parameters of the covariance as output, so that the uncertainty becomes a function of the raw measurement. Wang et al. [11] also proposed a method of estimating uncertainty with deep networks. Their focus was on end-to-end learning-based VO, but they expanded their learning-based VO into a probabilistic approach by incorporating uncertainty estimation. Learning-based inertial odometry has recently become popular in the literature [12], [13], [14]. Liu et al. [14] proposed a deep learning method for estimating the relative inertial odometry (IO) positional displacements and their uncertainties.

However, none of the learning-based approaches considered the sensor fusion perspective. It had not been proven that the uncertainty learned from a single sensor may be usable in a multi-sensor fusion application such as visual inertial odometry (VIO). One prominent approach in the literature is the learning-based feature selection proposed by [15]. Although this approach considers multiple sensors in an end-to-end manner, we instead seek to balance between sensors via proper covariance estimation rather than having to choose between two sensor modalities.

We address the former example, uncertainty modeling for sensor measurements and its application to VIO. By adding layers for covariance estimation to the existing VO layers
provided by [16], we suggest an unsupervised loss to learn, simultaneously, the mean and the variance of VO. Differing from previous works such as [17], [18], which presented learning a fully unsupervised uncertainty in the observed scene, we further investigate VO uncertainties and break their combined uncertainty into individual components contained in the observed scene and the estimated VO pose. Moreover, as uncertainty estimation becomes more critical when fusing multiple sensors, we also provide a method of estimating balanced uncertainties from multiple sensors. We propose a method of learning covariance for IO and balancing it with VO covariance. To the best of our knowledge, this stands as the first report of unsupervised uncertainty learning and balancing applied to VIO. In summary, the contributions we make in this paper are as follows.

1) We suggest a method of balancing relative magnitudes of uncertainties from different sensors and apply the learned uncertainties to multi-sensor fusion. By doing so, we achieve uncertainty meaningful in both relative and absolute manner.
2) Focusing on uncertainty learning, we propose an unsupervised uncertainty learning scheme that does not require ground truth in both mean and covariance.
3) We validate that balanced uncertainty performs robustly, despite harsh data degradation in vision and inertial sensors.

II. UNSUPERVISED UNCERTAINTY LEARNING FOR VISUAL AND INERTIAL ODOMETRY

A. Unsupervised Uncertainty Learning

Between the two types of uncertainty, namely aleatoric (data) uncertainty and epistemic (model) uncertainty [19], our focus is on aleatoric uncertainty which tells us the error caused by noises from the data themselves. For instance, dynamic objects cause mismatch between consecutive image scenes and lead to VO error.

1) Indirect Supervised Uncertainty Learning: Because securing ground truth uncertainty is rarely feasible when only ground truth mean is available, aleatoric uncertainty is often learned indirectly from the error in the predictive mean value. If the ground truth mean is available, the loss for learning uncertainty is given below by summing up over $N$ pixels as

$$L_{\text{sup}} = \frac{1}{N} \sum_{i=1}^{N} \frac{||y_i - \hat{y}_i||^2_{\hat{\Sigma}_{y_i}} + \log |\hat{\Sigma}_{y_i}|}{\hat{\Sigma}_{y_i}}, \quad (1)$$

where $|| \cdot ||^2_{\hat{\Sigma}}$ denotes Mahalanobis distance, normalizing the error with variance as $||e||^2_{\hat{\Sigma}} = e^T \hat{\Sigma}^{-1} e$.

2) Fully Unsupervised Uncertainty Learning: Unfortunately, the ground truth mean may no longer be available, necessitating fully unsupervised training in terms of mean and uncertainty. Sharing a common philosophy with [8], we modified the loss (1) by switching the ground truth $y$ and its prediction $\hat{y}$ into the measurement $z = g(x)$ and its prediction $\hat{z} = h(x, \hat{y})$. The unsupervised loss becomes

$$L_{\text{unsup}} = \frac{1}{N} \sum_{i=1}^{N} ||z_i - \hat{z}_i||^2_{\hat{\Sigma}_{e_i}} + \log |\hat{\Sigma}_{e_i}|, \quad (2)$$

Here, $g$ converts input data $x$ into the measurement $z$, whereas $h$ converts input data $x$ and the network prediction $\hat{y}$ into the predicted measurement $\hat{z}$. Further, the covariance used in the (2) includes observation-related as well as prediction-related terms since the error is given as

$$e = z - \hat{z} = g(x) - h(x, \hat{y}), \quad (3)$$
and the final aleatoric uncertainty becomes
\[
\hat{\Sigma}_e = \frac{\partial g_{\Sigma}}{\partial x} \Sigma \frac{\partial g_{\Sigma}}{\partial x}^\top + \frac{\partial h_{\Sigma}}{\partial y} \Sigma \frac{\partial h_{\Sigma}}{\partial y}^\top + \frac{\partial h_{\Sigma}}{\partial y} \Sigma \frac{\partial h_{\Sigma}}{\partial y}^\top
\]
\[
\hat{\Sigma}_e = \hat{\Sigma}_{\text{obs}} + \frac{\partial h}{\partial y} \hat{\Sigma}_y \frac{\partial h}{\partial y}^\top.
\]

3) Training via Covariance Balancing: While training, elements in this equation should be obtained to compose the overall error uncertainty \(\hat{\Sigma}_e\). We must solve for two unknowns, \(\hat{\Sigma}_{\text{obs}}\) and \(\hat{\Sigma}_y\) from a single constraint yielding the undetermined problem. We tackle this issue by introducing uncertainty balancing, which is discussed in §III. The balancing loss not only balances the covariances learned from different sources, but also works as the third constraint that makes all of the covariances determined.

### B. Learning-based VO

1) Camera Measurement: The VONet consists of DepthNet and PoseNet as in Fig. 1. The inputs consist of stereo images \(x_{\text{Depth}} = \{I_{\text{left}}, I_{\text{right}}\}\) at time \(k\) for DepthNet and a set of consecutive image frames \(x_{\text{Pose}} = \{I_{k-1}, I_k\}\) for PoseNet. The measurements are the estimated left/right frames (i.e., image or depth) for DepthNet \((z_{\text{Depth}} = \{I_{\text{left}}, I_{\text{right}}, D_{\text{left}}, D_{\text{right}}\}\) and the estimated frame and relative pose for PoseNet \((z_{\text{Pose}} = \{I_k, D_k, p_k\}\)). From the overall VONet, we aim to learn a relative pose \(y = p^k = p^{k-1,k} = [\Delta r_{k-1,k}^\top, \Delta t_{k-1,k}^\top]^\top\).

When checking consistency in losses, we need a mapping function \(h\) that maps a \(source\) frame to the \(target\) frame via the predicted relative pose \(\hat{y}\) from the network. In DepthNet, the depth and image losses are evaluated by checking left-right consistency, having the \(target\) be left and \(source\) be right. For PoseNet, the loss was computed from the temporal consistency between consecutive image and depth frames, having the \(target\) be \(k\)th frame and \(source\) be \((k-1)\)th frame. Specifically, the measurement function \(h\) can be written as
\[
\hat{I} = h(I^s, D^s, p^{s,t}),
\]
where \(s\) and \(t\) indicate the \(source\) and \(target\) frames.

2) Networks: As shown in Fig. 1, we added new networks for uncertainty learning (gray boxes in VONet) to the vanilla UnDeepVO [16]: (i) fully connected layers for the pose uncertainty; and (ii) decoders for the observation-related uncertainty denoted as VarianceNets. The pose uncertainty network learns the pose estimation uncertainty \(\Sigma_p\), and the VarianceNet captures the pixel-wise data-driven uncertainty.

3) Image and Depth Losses: When training PoseNet and DepthNet, we use the sum of the image loss \(L_{\text{image}}\) and depth loss \(L_{\text{depth}}\), following a similar form to that used in (2). The loss is calculated by checking the consistency between \(target\) and \(source\) images as
\[
L_{\text{image}} = \frac{1}{M} \sum_{i=1}^{M} \left[ \frac{1}{N} \sum_{p=1}^{N} \frac{(I^t_i(p) - \hat{I}_i(p))^2}{\hat{\sigma}^2_{I_i(p)}} + \log \hat{\sigma}^2_{I_i(p)} \right],
\]
where \(I^t_i(p)\) indicates the intensity of the pixel \(p\) in the \(target\) image and \(\hat{\sigma}^2_{I_i(p)}\) is its corresponding uncertainty. For \(M\) images in the batch, the loss is summed up over \(N\) pixels. Depth loss \(L_{\text{depth}}\) is also given as (6), using depth image \(D\) instead of the intensity image \(I\). The proposed image and depth loss indicate that VO is learned in a self-supervised manner as in previous works such as [16], [20]. All losses for VONet are listed in Fig. 1.

4) Uncertainty in VO: Overall, we learn the combined uncertainty for each loss while the loss includes observation-related and prediction-related uncertainty as in (4). As we are interested in pose estimation, we focus on the combined uncertainty obtained from PoseNet. For instance, the uncertainty of the image loss is given as
\[
\hat{\sigma}^2_{I_{i}(p)} = \hat{\sigma}^2_{I_{i,\text{obs}}} + \left( \frac{\partial h_{\Sigma_p}}{\partial p} \right) \Sigma_p \left( \frac{\partial h_{\Sigma_p}}{\partial p} \right)^\top.
\]

![Fig. 2: Sample scenes and the estimated observation-related uncertainty. The images in the left column are source images and the images in the right column are target images overlaid with the uncertainty. The pixel is color-coded by variance, showing red color to indicate a larger variance. The sample scenes are classified into three categories based on the cause of inherent errors in the scene.](Image 323x476 to 433x509)

![Low High](Image 445x476 to 555x509)

(a) Unusual lightings

(b) Occlusions

(c) Moving objects
of measurement error with respect to the pose is numerically computed. The resulting observation-related uncertainty, learned from the temporal image consistency, is shown in Fig. 2. The data uncertainty successfully captured inherent errors in the scene, which were caused by unusual lighting, occlusions, and moving objects.

C. Learning based IMU Preintegration

Forster et al. [21] established a noise propagation method for preintegration, but they did not deal with changing errors caused by mechanical malfunctions often affected by varying environment as suggested by [15]. The proposed self-supervised initial covariance learning also relies on the model [21]; however, this initially learned covariance will be elaborated through balancing against VONet to incorporate visual raw data in order to overcome the limitations of single sensor learning.

1) IMU Measurement: Using the preintegrated inertial measurement unit (IMU) model in [21], we define inputs of the network as IMU raw data, consisting of angular velocity $\omega$, linear acceleration $a$, and the time difference between the consecutive input $\Delta t$ as $x = [\omega, a, \Delta t]$. A set of IMU measurements is given as $z^{I} = \{r^T, v^T, t^T\}$, where each component represents the preintegrated rotation $r$, velocity $v$ and translation $t$ as

$$
\begin{align*}
    r &= \text{Log} \left( \prod_{k=1}^{n} \text{Exp}(w_k - b_o)\Delta t_k \right), \\
    v &= \sum_{k=1}^{n} R_{1k}(a_k - b_o)\Delta t_k, \text{ and} \\
    t &= \sum_{k=1}^{n} v_{1k}\Delta t_k + \frac{1}{2} R_{1k}(a_k - b_o)\Delta t^2.
\end{align*}
$$

Exp and Log indicate exponential and logarithm map, converting rotations in $\mathbb{R}^3$ to $\text{SO}(3)$ and vice versa. The $\text{SO}(3)$ rotation $R_{1k}$ in (9) comes from the rotation vector $r_{1k}$ as $R_{1k} = \text{Exp}(r_{1k})$. Further, $b_o$ and $b_o$ are biases for the angular velocity and linear acceleration, respectively. The network prediction directly corresponds to the measurement as $\tilde{y} = \tilde{z}^{I}$, so that the measurement function becomes identity.

2) Preintegration Losses: To train the IMU Preintegration (IP)Net, we defined the preintegration loss for the rotation $L_r$, velocity $L_v$, and translation $L_t$ as we did in the self-supervised loss in (2). The preintegration loss for the rotation is given as

$$
L_r = \frac{1}{N} \sum_{i=1}^{N} ||r_i - \hat{r}_i||^2_{\Sigma_r} + \log ||\Sigma_r||. 
$$

All losses $L_r$, $L_v$, $L_t$ for IPNet are presented in Fig. 1.

3) Networks and Uncertainty: Because IMU data comes in sequentially, we accounted for this characteristic by using Long Short Term Memory (LSTM). There are cascaded LSTMs to predict rotation, velocity and translation to reflect the hierarchical relationship among the measurements. The network outputs prediction of the $\hat{r}$, $\hat{v}$ and $\hat{t}$ and their covariances, $\Sigma_r$, $\Sigma_v$ and $\Sigma_t$. We assume $\Sigma_x = 0$ and this prediction uncertainty becomes the final IPNet uncertainty. The details of IPNet are given in Fig. 1 and Table. I.

III. UNCERTAINTY BALANCING IN SENSOR FUSION

A. Uncertainty Balancing for VO and IO Fusion

Next, we explain how the proposed uncertainty estimation and balancing methods are applied to the end-to-end learning-based visual and inertial fusion (Fig. 1). The role of this balancing step is two-fold: first, we balance uncertainty from VONet and IPNet by checking intersensor consistency; and second, we estimate IMU bias term based on consistency losses for acceleration and gravity via BiasNet.

B. Uncertainty Balancing via Intersensor Consistency

The first role is to balance uncertainty relatively from intersensor consistency. During this consistency check, the observation in camera $C$ is transformed into the same formation of the observation in IMU $I$, via a sensor modality transfer function $g_{C \rightarrow I}(\cdot)$ as

$$
\hat{z}^{C} = g_{C \rightarrow I}(\hat{z}^{I}).
$$

Specifically, the transfer function computes the transformed observation $\hat{z}^{C} = \{r^{C \rightarrow I}, v^{C \rightarrow I}, t^{C \rightarrow I}\}$ (8)-(10) using relative pose estimated from VONet, $\hat{z}^{C} = \{p^{k \rightarrow I}, \hat{p}^{k \rightarrow I}\}$.

An overview of the balancing process is illustrated in Fig. 1. Two pre-trained networks, VONet (II-B) and IPNet (II-C) are trained together with BiasNet to balance VO and IMU uncertainties. The balancing loss is given as

$$
L_{\text{balancing}} = L_{\text{image}} + L_{r} + L_{v} + L_{t} + L_{C \rightarrow I, x} + L_{C \rightarrow I, v} + L_{C \rightarrow I, t},
$$

whereas

$$
\begin{align*}
    L_{C \rightarrow I, x} &= \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left\| g_{C \rightarrow I}(\Delta \hat{r}_{ik}^{C}) - \hat{r}_{ik}^{I} \right\|^{2}_{\Sigma_{r_{ik}}^{C \rightarrow I}} + \left\| \Delta \hat{r}_{ik}^{C} \right\|^{2}_{\Sigma_{r_{ik}}^{C \rightarrow I}}, \\
    L_{C \rightarrow I, v} &= \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left\| g_{C \rightarrow I}(\Delta \hat{v}_{ik}^{C}) - \hat{v}_{ik}^{I} \right\|^{2}_{\Sigma_{v_{ik}}^{C \rightarrow I}}, \\
    L_{C \rightarrow I, t} &= \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left\| g_{C \rightarrow I}(\Delta \hat{t}_{ik}^{C}) - \hat{t}_{ik}^{I} \right\|^{2}_{\Sigma_{t_{ik}}^{C \rightarrow I}}.
\end{align*}
$$

The first four terms $L_{\text{image}}$, $L_{r}$, $L_{v}$ and $L_{t}$ are losses from the camera and IMU provided in (6) and (11). These losses prevent the balancing loss from being completely divergent and keep the learning process stable even when both sensors are unreliable at the same time.

<table>
<thead>
<tr>
<th>TABLE I: IPNet and BiasNet. The size of input is n.</th>
</tr>
</thead>
<tbody>
<tr>
<td>network</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>IPNet</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>BiasNet</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
intersensor consistency losses computed from the preintegrated rotation, velocity and translation measurements. In the intersensor consistency loss, the measurement from the camera is transferred into the preintegrated measurement from the IMU using the sensor modality transfer function \( g_{C\rightarrow I} \). The detailed derivation of each term is given in the Appendix.

C. BiasNet training

Secondly, we introduce BiasNet to estimate the slowly varying biases in angular velocity and linear acceleration. Because the IPNet was pre-trained in a self-supervised manner without ground truth labels, the trained network only learned consistency missing an absolute reference. Specifically, the bias term could not be learned when we train IPNet alone in a self-supervised manner. In this training phase, bias is learned using balancing loss introduced in the previous subsection. There are two bias losses for estimating the bias in angular velocity \( b_a \) and in linear acceleration \( b_a \).

\[
L_{b_a} = L_{image} + L_r + L_{C\rightarrow I,r} \tag{15}
\]

\[
L_{b_a} = L_{image} + L_v + L_a + L_{C\rightarrow I,v} + L_{C\rightarrow I,t} \tag{16}
\]

The differences in camera-IMU measurements, \( r^C_{i,k} - r^I_{i,k} \) and \( t^C_{i,k} - t^I_{i,k} \), are fed back to BiasNet as input, which returns the increment in biases, \( \Delta b_a \) and \( \Delta b_a \), as output. In this way, the network learns how much increment in biases is needed given the history of difference in measurements from the camera and IMU.

D. Training Details

The proposed architecture was implemented using Tensorflow and trained with NVIDIA GeForce GTX 1080 Ti. Our VONet was implemented based on [16] and IPNet and BiasNet were implemented as described in Table I. We used the same training parameters for all of the training steps that we took. We used an Adam optimizer [22] with \( \beta_1 = 0.9 \), \( \beta_2 = 0.999 \), and an initial learning rate of \( 10^{-4} \). The learning rate was gradually decreased by a half and a quarter of the initial value at 3/5 and 4/5 of the total iterations.

For VONet, we trained the depth network for 60 epochs with a batch size of 16. Then, we trained the depth and pose networks alternately using the same number of epochs and the same batch size. When training the depth and pose networks, we resized the input image into a resolution of 256 × 512 and generated four additional levels of image pyramids. We augmented the input data by randomly selecting half of the images and shifting their gamma, brightness, and color values. For IPNet, we also trained the network sequentially. We trained the network for the preintegrated rotation for 30 epochs and then trained the network for the preintegrated velocity and translation for 30 epochs.

Last, we conducted the balancing process by training the uncertainty estimation parts of VO and IP networks and BiasNet for 30 epochs. In each iteration, the networks for uncertainty and bias were alternately trained. First, we fixed all the other layers including the BiasNet and trained the uncertainty networks using the balancing loss given in (13). Then, we fixed the uncertainty networks and trained the BiasNet using (16).

IV. EXPERIMENT

We provide an evaluation of the proposed uncertainty estimation and its application to VO using publicly available driving datasets, the KITTI odometry dataset [23] and KAIST urban dataset [24].

A. Datasets and Synthetic Degradation

In the KITTI dataset, we used sequence 00–08 as a training set and 09–10 as a testing set. In the KAIST urban dataset, we used Urban28 as a training set and parts of Urban38 as a testing set. We first trained the network using the KITTI dataset and then trained again with the KAIST urban datasets.

To see the effects of various sensor degradations on the uncertainty estimation, we added synthetic errors to the original data. For realistic visual degradation, we created synthetic images affected by changes in lightings and varying weather conditions. Rain and night images were generated by using [25] and [26]. To produce a saturated image, each image pixel was multiplied by a constant and clipped to the maximum pixel value. Fog images were created by using the atmospheric degradation model from [27].

For IMU degradation, we focused on modeling unexpected errors. We modeled missing data, which can be caused by an unstable packet drop from the bus, by randomly removing one percent of the entire sequence. In addition, we created random walk errors to model noises caused by unexpected mechanical malfunctions. We randomly selected an interval of IMU sequences and infused random walk error within it. For all three of the test datasets, we set 2,000 as the length of the interval with random errors.

B. Evaluation of Uncertainty

1) Qualitative Evaluation: Fig. 3 shows the results of the learned uncertainty on the KITTI 10 and KAIST urban 38 datasets. The uncertainty captures error fluctuations as depicted in the one sigma bounds. For example, the thumbnail images represent situations where large error and uncertainty are measured. The uncertainty also reflects the relative magnitude of errors in each axis. Since the driving data has large errors in the traveling direction (x-axis), we see larger uncertainty in the x-axis than in other axes.

We also presents qualitative performance on the various sensor degradations in Fig. 4 (a)–(b). For visual degradation, we list the results in order of increasing error. The shaded area, which represents a trace of translational and rotational covariances, keeps increasing as more errors are involved. When sporadic IMU degradation exists, the uncertainty captures those errors and indicates drastic fluctuations.

2) Quantitative Evaluation: For quantitative evaluation, we measured average log-likelihood over the KITTI and KAIST urban test sequences. Average log-likelihood is negative of the supervised loss given as (1). Please note that we used the ground truth pose provided by the dataset
Fig. 3: The graphs show translational and rotational errors in each axis and their 1σ bounds depicted with shaded regions. Thumbnail images on the top illustrate situations in which large uncertainty occurs. Large translation errors in y axis occur at curved roads (A). Large z-axis uncertainties within highly dynamic environments with moving cars are shown in (B). Uneven road, such as a speed bump, causes large uncertainty in pitch motion (C).

Fig. 4: Sample images with visual degradation are shown and sporadic IMU degradation is marked with red in the timeline. The graphs indicate translational and rotational errors and the shaded area illustrates their covariance represented by trace. (a) All degradation cases caused error and covariance to increase. The range of the error enlarged and continuous error increase is found in rain in URBAN38. (B) Degradation in IMU measurement mainly induced an increase in rotational uncertainty. The zoomed views show that spikes in uncertainty increase when degradation in IMU occurs. Tables show evaluation results of the VO uncertainty (top) and IO uncertainty (bottom). Larger numbers indicate better performance.
for evaluation. To generate IO ground truth, in particular, we computed pseudo-preintegration using the ground truth pose. It represents how the estimated uncertainty captures error magnitudes on average and larger values indicate better performance. For VO uncertainty evaluation, our method was compared with the supervised learning based uncertainty estimation, DICE [10]. For IO uncertainty evaluation, we compared the result with inertial odometry with uncertainty estimation, TLIO [14]. Note that TLIO [14] was evaluated using relative positional displacements and their uncertainties unlike ours evaluated based on preintegrated states.

The first table in Fig. 4 shows the average log-likelihood of the VO uncertainty estimation. DICE is advantageous in this metric as the supervised loss is negative of the average log-likelihood itself, but our method shows comparable numbers although the network was trained in an unsupervised way without referring to ground truth errors. We note that, as more errors are involved, it becomes less likely that the uncertainty reflects the real error. However, our results are slightly better in saturation and rain images and much better in night images where DICE completely fails to capture uncertainty. We also report the correlation between error and variance on the yaw angles. Higher correlation value indicates that the estimated variance successfully captures the error fluctuations and ours show high correlation values on average except for the challenging night images.

The evaluation of IO uncertainty is given in the second table in Fig. 4. Our performance is better than TLIO [14] and stays constant even when degradation exists. As the conventional method [21] gives nearly constant covariances for all the measurements regardless of error, their average log-likelihood shows poor results.

3) Discussion on Uncertainty Balancing: The tables in Fig. 4 also include evaluations before and after balancing. The overall performance improved, especially after balancing in VO uncertainty. We found that the VO uncertainty was affected more from the balancing because the ambiguity problem between observation-related and prediction-related terms in (4) is settled by adding a balancing constraint. More importantly, during the balancing, different uncertainties converge to a similar evaluation result, which makes the balanced uncertainty more useful in the sensor fusion.

C. Evaluation of Odometry

Last, we evaluate the estimated covariances within the odometry application. By giving an example of VIO, we show how the visual and inertial measurements are complementarily combined within a pose-graph SLAM framework by using properly estimated covariances via

$$X^* = \arg\min_X \sum_i \|p_i - \tilde{p}_i\|^2_{\Sigma_{p_i}} + \sum_i \|z_i - \tilde{z}_i\|^2_{\Sigma_{z_i}},$$

where the optimal state $X^*$ represents a set of absolute poses and velocities. $p_i$ and $\Sigma_{p_i}$ are relative camera pose and its covariance obtained from VO, and $z_i$ and $\Sigma_{z_i}$ indicate the IMU measurement (8)–(10) and its covariance from IO. To test VIO in the SLAM framework, we used GTSAM 4.0 optimization toolbox [29]. We applied the learned VO to the relative pose factor. For the VO and IO factors, we used the learned uncertainties from the network.

VIO results on visual and inertial degradations are given in Fig. 5 and Table II. Compared to one of the traditional VIO methods, such as VINS-Mono [28], our results show robust performance overall. For instance, when severe data degradation exists (night scenes on visual degradation and
TABLE II: Evaluation of VIO on Visual and Inertial Degradations

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( t_{rel} )</td>
<td>( r_{rel} )</td>
</tr>
<tr>
<td><strong>K</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>default</td>
<td>7.32</td>
<td>1.16</td>
</tr>
<tr>
<td>I sat</td>
<td>9.04</td>
<td>0.87</td>
</tr>
<tr>
<td>T rain</td>
<td>14.40</td>
<td>2.16</td>
</tr>
<tr>
<td>T fog</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0 missing</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9 random</td>
<td>23.24</td>
<td>7.52</td>
</tr>
<tr>
<td><strong>K</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>default</td>
<td>15.81</td>
<td>1.51</td>
</tr>
<tr>
<td>I sat</td>
<td>31.45</td>
<td>4.09</td>
</tr>
<tr>
<td>T rain</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>T fog</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>I night</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0 missing</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9 random</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>U</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rain a</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>night n</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>missing</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>random b</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0 random</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

\( t_{rel} \): average translational Root Mean Square Error (RMSE) drift (%) on length of 100m – 800m. \( r_{rel} \): average rotational RMSE drift (°/100m) on length of 100m – 800m. VINS-Mono works well with moderate data degradation but the trajectory completely diverges when the errors get bigger. DICE shows solid performance which is slightly worse than ours. In urban scenes, performance of VINS-Mono is less affected by fog because nearby features from the road lane are robustly extracted regardless of the fog as seen in the thumbnail images. Ours includes ablation studies of VIO performance using VO and IO uncertainties before and after balancing, missing data on IMU degradation, VIO trajectory even diverges while ours does not. For the comparison with DICE [10], which gives VO covariances only, we kept other conditions such as VO and IO mean values and IO covariances but switched the VO covariance with theirs. DICE shows comparable results but our method outperforms in most of the sequences.

V. CONCLUSION

We proposed a general approach to unsupervised uncertainty estimation using deep networks. To overcome the limitation of uncertainty learning from a single sensor, we introduced a method of balancing uncertainties between different sensor modalities. We applied the uncertainty estimation and balancing methods to VIO. In our experiment, we validated the usefulness of learned uncertainty when the data degradation occurs in multiple sensors.

APPENDIX

1) Rotation consistency loss: The rotation observed by camera-based VO, \( \Delta \hat{R}_k^C \), is transferred to \( \Delta \hat{T}_k^C \) as

\[
\Delta \hat{T}_k^C = g_{C \rightarrow Z}(\Delta \hat{R}_k^C) = \log(A_1^R \hat{R}_k^C \exp(A_1^R \hat{D}_k^C)),
\]

where \( A_1^R \hat{R}_k^C \) is the rotation between the camera and IMU. Uncertainty for the rotational consistency \( \Sigma_{\Delta \hat{T}_k^C \rightarrow \Delta \hat{T}_k^C} \) is given as

\[
\Sigma_{\Delta \hat{T}_k^C \rightarrow \Delta \hat{T}_k^C} = J_{\hat{T}}^{-1}(\log(A_1^R \hat{R}_k^C)) \Sigma_{\Delta \hat{R}_k^C} J_{\hat{T}}^{-\top}(\log(A_1^R \hat{R}_k^C)) + \Sigma_{\Delta \hat{T}_k^C},
\]

where \( J_{\hat{T}}(\cdot) \) indicates the right Jacobian of the exponential map [21].

2) Velocity consistency loss: Similarly, the transferred velocity \( \Delta \hat{v}_k^C \) is approximated from the relative camera poses \( \hat{p}_{k-1}^C \) and \( \hat{p}_k^C \) as

\[
\hat{v}_k^C = g_{C \rightarrow Z}(\hat{p}_{k-1}^C, \hat{p}_k^C) = \frac{\hat{R}_C \Delta \hat{v}_k^C - \exp(\Delta \hat{v}_k^C) ^\top \Delta \hat{v}_k^C}{\Delta t},
\]

where \( \Delta t \) is the time difference between image frames, \( I_{k-1} \) and \( I_k \) with covariance of the velocity difference as

\[
\Sigma_{\Delta \hat{v}_k^C \rightarrow \Delta \hat{v}_k^C} = \frac{\hat{R}_C \Sigma \Delta \hat{v}_{k-1}^C \hat{R}_C^\top}{\Delta t^2} + \Sigma_{\Delta \hat{v}_k^C}.
\]

3) Translation consistency loss: The transferred translation \( \Delta \hat{t}_k^C \) is

\[
\Delta \hat{t}_k^C = g_{C \rightarrow Z}(\hat{p}_{k-1}^C, \hat{p}_k^C) = \Delta \hat{t}_k^C(\hat{R}_C \Delta \hat{v}_k^C - \exp(\Delta \hat{v}_k^C) ^\top \Delta \hat{v}_k^C),
\]

which is the same as the transferred velocity (21) except for the denominator \( \Delta t \). Covariance of the translation difference \( \Delta \hat{t}_k^C \rightarrow \Delta \hat{t}_k^C \) becomes

\[
\Sigma_{\Delta \hat{t}_k^C \rightarrow \Delta \hat{t}_k^C} = \Delta \hat{R}_C \Sigma \Delta \hat{v}_{k-1}^C \hat{R}_C + \Sigma_{\Delta \hat{t}_k^C}.
\]
REFERENCES