Proceedings of the 2022 IEEE
International Conference on Robotics and Biomimetics
December 5-9, 2022, Xishuangbanna, China

CertainOdom: Uncertainty Weighted Multi-task Learning Model for LiDAR Odometry Estimation

Leyuan Sun, Guanqun Ding, Yusuke Yoshiyasu and Fumio Kanehiro

***Abstract*—As a basic and indispensable module, LiDAR odometry estimation is widely used in robotics. In recent years, learning-based modeling approaches for odometry estimation have been validated to be feasible. However, it is necessary to consider security factors as the highest priorities when we apply the learning-based model to certain high-risk real-world scenarios, such as autonomous driving. The odometry uncertainty estimation provides more valuable information for downstream tasks, such as route planning and navigation. In this paper, we propose an end-to-end neural network (namely CertainOdom) to solve odometry and uncertainty estimation tasks by applying multi-task learning. Instead of using the manually-tuned hyper-parameters, we employ the learnable uncertainties to weigh the balance between the error of translation and orientation in the loss function. We evaluate the estimated trajectory and uncertainty on KITTI dataset. We also compare the robustness against the traditional geometry-based methods on our artificially degraded KITTI LiDAR dataset. Extensive experimental results show that our model with uncertainty weighted loss achieves competitive performance in LiDAR odometry estimation. We also explain our uncertainties qualitatively and quantitatively.**

I. Introduction

An accurate and robust robot odometry estimation is essential for various aspects of robot navigation, locomotion, and other domains such as Augmented Reality (AR), Virtual Reality (VR), and autonomous driving. Traditional Simultaneous Localization And Mapping (SLAM) techniques mostly adopt vision-based strategies, which are sensitive to environmental changes, such as illumination, reflections, moving objects, and texture-less backgrounds. Recently, the learning-based approaches combined with geometry-based SLAM [1] [2] and pure end-to-end neural network [3] have validated that the robustness of odometry estimation can be improved, especially in a dynamic environment. However, relying solely on odometry estimation for planning and navigation in real-world scenarios still carries a high risk, especially when encountering imperfect sensor data.

Leyuan Sun, Guanqun Ding, and Fumio Kanehiro are with Department of Intelligent and Mechanical Interaction Systems, Graduate School of Science and Technology, University of Tsukuba, Tsukuba, Ibaraki, Japan.

Leyuan Sun and Fumio Kanehiro are with CNRS-AIST Joint Robotics Laboratory (JRL), IRL, National Institute of Advanced Industrial Science and Technology (AIST). son.leyuansun, f-kanehiro@aist.go.jp

Guanqun Ding is with the Digital Architecture Research Center (DARC), National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan. guanqun.ding@aist.go.jp

Yusuke Yoshiyasu is with the Computer Vision Research Team, Artificial Intelligence Research Center (AIRC), National Institute of Advanced Industrial Science and Technology (AIST), Japan. yusuke-yoshiyasu@aist.go.jp

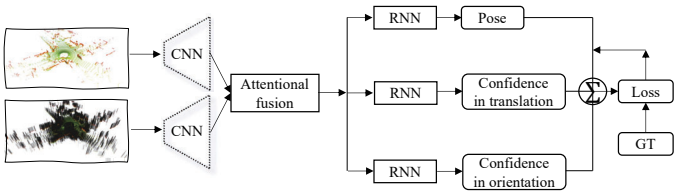

Fig. 1. Overview of proposed end-to-end LiDAR odometry estimation with uncertainty. The inputs of the proposed model are raw point cloud and its normal estimation.

In recent years, uncertainty estimation has been considered as an effective solution to reduce the risk and improve the feasibility of the system for real-world applications. Kendall *et al.* [4] define the aleatoric and epistemic uncertainty, which come from data and model separately. The aleatoric uncertainty is visualized for the first time in a semantic segmentation task. Yang *et al.* [5] use a self-supervised neural network to predict the uncertainty in photometric loss, and they utilize it in a traditional nonlinear optimization back-end for SLAM. Gasperini *et al.* [6] develop a Gaussian process-based uncertainty estimation in classification task for object detection, and they apply the Gaussian-based uncertainty in autonomous driving, which also benefits for system's downstream tasks.

The overview of the proposed model is shown in Fig. 1. Inspired by recent successful works [4]–[6], we propose a framework of multi-task learning network, which includes a LiDAR odometry estimation module and an uncertainty estimation module in translation and rotation. Different from the studies of [7] and [8] that use manual hyperparameters for loss weighting, we employ the predicted uncertainty to balance the errors of translation and orientation. The LiDAR feature extractor of the proposed model is modified from the models of [9] and [10]. To combine the features extracted from the raw point cloud and normal representation, we consider two fusion methods: direct addition and attentional fusion strategy [11]. Finally, each task of the model is regressed by individual Recurrent Neural Network (RNN) and fully connected layers.

We evaluate the trajectory and uncertainty on the raw KITTI [12] and our generated degraded KITTI LiDAR dataset through cropping and adding Gaussian noise to a raw point cloud. By evaluating on these two datasets, we can compare the robustness of proposed approach against a traditional geometry-based method LIO-SAM [13]. Regarding to the evaluation of uncertainty estimation, we not only demonstrate the predicted variance with input data, but also show the errors between ground truth and predicted pose in each axis, with 1σ

978-1-6654-8109-0/22/$31.00 © 2022 IEEE

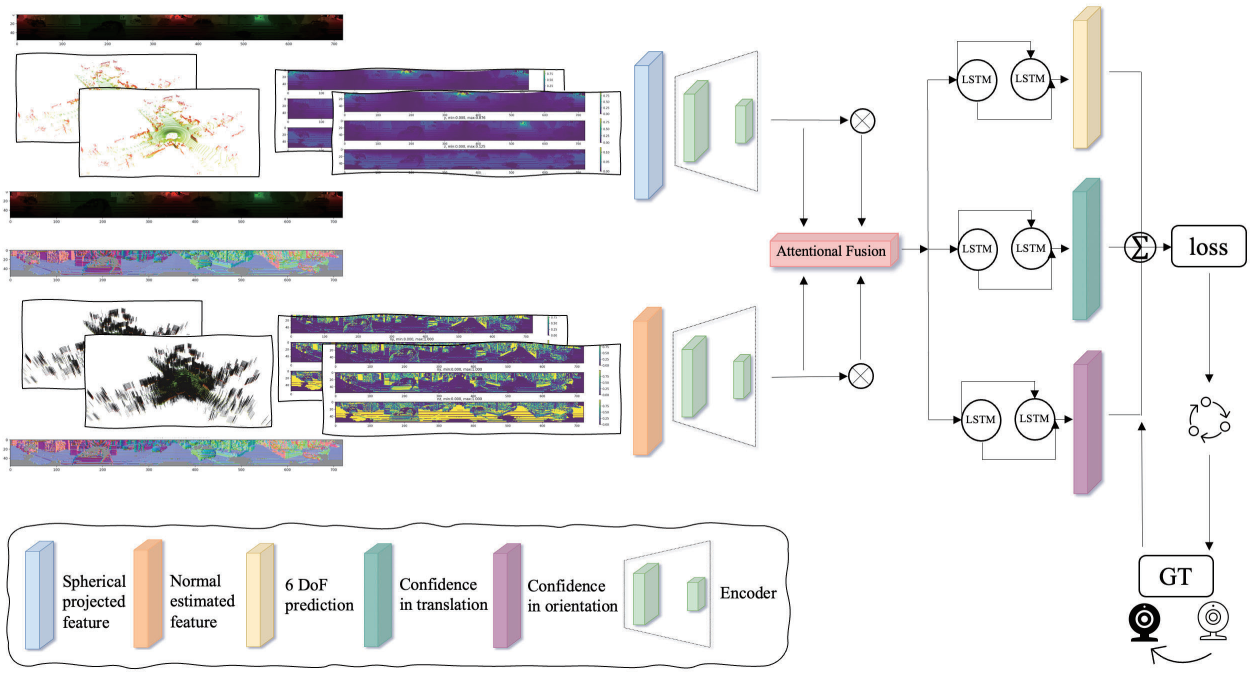

Fig. 2. Network architecture of proposed CertainOdom. It includes a spherical projection encoder, a normal projection encoder, and three multi-task LSTM-based decoders.

and 3σ bounds to check the correlations between input data, uncertainty and the error of output.

The contributions of this paper can be summarized as follows:

- We propose an end-to-end neural LiDAR odometry estimation network called CertainOdom. It not only can predict the LiDAR ego-motion, but also is able to calculate its uncertainty in translation and rotation.
- Instead of using manually tuning parameters, the learnable uncertainty of proposed model can be leveraged to weigh the mean squared errors between translation and rotation.
- We artificially generate a degraded LiDAR dataset based on KITTI [12] by randomly cropping and adding Gaussian noise to raw point cloud data. Based on this dataset, we evaluate the robustness by comparing traditional geometry-based models and proposed learning-based LiDAR odometry estimation method.
- We qualitatively and quantitatively demonstrate the predicted uncertainty estimation on raw KITTI [12] dataset and our degraded KITTI dataset.

II. Related work

A. Geometry-based Odometry Estimation

Iterative Closest Point (ICP) [14] is a basic point cloud registration method in LiDAR odometry estimation. LeGO-LOAM [15] adopts the segmentation module to feature extractor, thereby obtaining the planar and edge features for nonlinear optimization based on GTSAM[1] library. Recently, researchers consider fusing LiDAR features and IMU (Inertial Measurement Unit) information to improve the accuracy of odometry estimation, since the features from these two sensors can be regarded as complementary representations. LIO-SAM [13] is a well-known work that combines these two modalities in the field of odometry estimation. In order to test the robustness of LIO-SAM [13] model, we evaluate it on our degraded-KITTI dataset by using different imperfect data as input. In Section IV.B, we explain more details about the generation of degraded LiDAR point cloud.

In Fig. 3, we show the comparison results of LIO-SAM [13] model on original KITTI odometry [12] dataset and our degraded-KITTI dataset. We can observe that the trajectory of LIO-SAM [13] drifts a little on the scene of cropped point cloud data, but it can quickly be re-localized on the rest of original point cloud data. On the other hand, LIO-SAM [13] is more sensitive to the point cloud data with Gaussian noise, which drifts further and cannot be re-localized again, compared with the situation of cropped point clouds. We believe these drift and re-localization problems could be

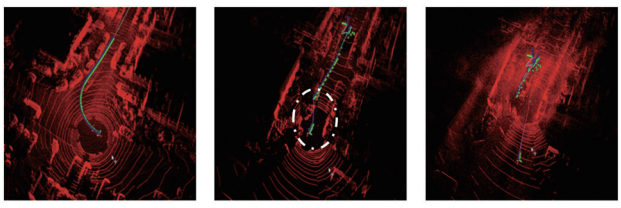

Fig. 3. LIO-SAM [13] under KITTI [12] dataset (2011-09-26-0035 sequence). From left to right: results on original data, crop data, and Gaussian noisy data.

[1]https://gtsam.org/

caused by the ICP characteristic, which is a nonlinear least squares optimization problem. Compared with the distortion of cropped point clouds, the degradation of adding Gaussian noise to raw point clouds is more challenging for LIO-SAM [13] to find the optimal value during the optimization.

B. Learning-based Odometry Estimation

PoseNet [7] is a famous work to estimate the camera ego-motion by utilizing two consecutive frames. In this model, the authors adopt MSE (Mean Squared Error) as their loss to optimize their model parameters. However, they leverage handcrafted hyper-parameters as weights to balance the translation MSE loss and orientation MSE loss [7], which could not be adaptive for different scenarios of point clouds. DeepVO [8] introduces a Recurrent Neural Network (RNN) based model for pose regression. In DeepVO [8], it is also suffered from the problem of human-designed hyper-parameter for the loss weighting. Usually, the hyper-parameters need to be tuned manually with extensive validation experiments, which are time-consuming and sensitive to different datasets. There are also several supervised [10] and unsupervised [16] [17] methods for LiDAR odometry estimation, but few works consider applying to pose uncertainty estimation in this field.

C. Uncertainty Estimation

Kendall *et al.* [4] explain the definition of aleatoric and epistemic uncertainty. The study of [18] uses the method of epistemic uncertainty estimation, which adds the dropout layer before each weight layer for both the training stage and inference stage. For aleatoric uncertainty, the method [19] develops an unsupervised uncertainty estimation framework based on the work of [4]. By using an unsupervised photometric loss, the model [19] obtains the pixel uncertainty of camera sensor data, which utilizes the pre-integration of IMU measurement as IMU's ground truth. However, it cannot guarantee accuracy, especially when IMU data is imperfect [19].

MDN-VO [20] applies the Mixture Density Network (MDN) to the camera ego-motion task as a regressor. Different from the traditional deep neural network to predict a value for odometry estimation, MDN predicts a distribution including means, standard deviations, and mixture coefficients. This method [20] claims that the mean value could be used as the pose prediction, and the standard deviation could be considered as the uncertainty. However, MDN-VO [20] does not clearly assess the relationship between uncertainty and input data, especially when the model encounters challenging environments.

III. Methodology

A. LiDAR Feature Extractor

First of all, we convert each point $p_i = (x, y, z)$ of LiDAR point cloud to 2D (u, v) through a spherical projection method $\mathbb{R}^3 \Rightarrow \mathbb{R}^2$ proposed in the study of [21], which can be calculated by Eq. 1:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}[1 - arctan(y, x)\pi^{-1}]\omega \\ [1 - (arcsin(zr^{-1}) + f_{up})f^{-1}]h \end{pmatrix} \quad (1)$$

where (h, w) are the height and width of the output range image; $r = \|p_i\|_2$ denotes the range of point p_i. The vertical field-of-view of sensor is represented as $f = f_{up} + f_{down}$. Similar to the solution in DeepLO [10], if more than one 3D point is projected onto the same pixel coordinate, we select the nearest pixel value.

The normal information is also an important feature in the LiDAR point cloud, which is useful for the registration of point clouds. In this work, we adopt the method proposed in [22] to obtain the normal estimation. This method [22] leverages the weighted cross products among the four-point neighbours for normal calculation, which can be represented as follows:

$$n_i = \frac{\sum_{j=1}^{4} n_{i_j}^{'}}{\left\| \sum_{j=1}^{4} n_{i_j}^{'} \right\|}, n_i^{'} = \sum_{j=1}^{4} b_{i,i_j} b_{i,i_{j+1}} (d_{i,i_j} \times d_{i,i_{j+1}}) \quad (2)$$

where n_i is the normal of point p_i; d_{i,i_j} indicates the distance between p_i and p_j, and it is weighted by b_{i,i_j}. More details of Eq. 2 are explained in the study of [22].

In Fig. 2, the spherical projections (u, v) of two consecutive LiDAR point clouds (L_{i-1}, L_i) are shown on the top and bellow of raw point cloud. We also visualize three channels of LiDAR scan $L_i = [x, y, z]$. We show the same structure of normal maps with $Ln_i = [n_x, n_y, n_z]$ in Fig. 2. Then, they are embedded by f_i, which is a CNN based encoder to obtain extracted features (S_i, N_i):

$$\begin{aligned} S_i &= f_i([L_{i-1}, L_i]) \\ N_i &= f_i([Ln_{i-1}, Ln_i]) \end{aligned} \quad (3)$$

B. Attentional Feature Fusion

After obtaining the spherical and normal projections, we feed them into CNN for extracting features based on PointSeg [9]. By following the ATVIO [23] model that fuses two sensors' features, the output features can be combined in three different ways: simply concatenate, point-wise summation [10], and self-attention fusion. Different from self-attention [24] [25], Dai *et al.* [11] develop an advanced iterative attentional feature fusion (iAFF) method. iAFF [11] has the advantages of fusing different scales and inconsistent features, and it is applicable for most complex scenarios. Instead of fusing different sensors directly, we aggregate the features from spherical projection S and normal map N by the following formula:

$$S \uplus N = M(S + N) \otimes S + (1 - M(S + N)) \otimes N \quad (4)$$

where M is the multi-scale channel attention module (MS-CAM) proposed in the study of [11].

C. Multi-task Regression

1) RNN-based Temporal Modelling: RNN has good potential to learn the knowledge of temporal information from sequential data [8] [20] [23] [26]. We adopt Long Short Term Memory (LSTM) as the RNN structure, which can be represented by:

$$r_t, h_t = LSTM(f_t, h_{t-1}) \quad (5)$$

where f_t denotes the output of the attentional feature fusion $S \uplus N$, h_t and r_t are the hidden states and outputs of LSTM.

2) Loss Function: In the supervised learning-based 6D pose regression task, PoseNet [7] predicts a pose vector $\mathbf{p}$ that consists of translation x and rotation (represented by quaternion q), which can be written as:

$$\mathbf{p} = [x, q] \tag{6}$$

To regress the 6D pose, PoseNet [7] uses the following equation as loss function:

$$loss = \|x - \hat{x}\|_2 + \beta \left\| \frac{q}{\|q\|} - \hat{q} \right\|_2 \tag{7}$$

where the $\hat{\mathbf{p}} = [\hat{x}, \hat{q}]$ represents the ground truth; β is a manually tuned hyperparameter to balance the mean squared error between translation and rotation. This loss function is also employed in DeepVO [8], ATVIO [23], VINet [26] and SelectFusion [27].

In DeepLO [10], Lo-Net [28] and MS-Transformer [29], a more advanced loss function is introduced to learn adaptive weights with *torch.nn.Parameter*, as shown in Eq.8:

$$loss = L_x exp(-s_x) + s_x + L_q exp(-s_q) + s_q \tag{8}$$

where L_x and L_q are the error of translation and rotation, s_x and s_q are learned parameters. In Eq.8, the learnable parameters s_x and s_q are fixed after training. And no matter how the input changes, they did not contain the meaning of uncertainty. However, we employ multiple decoders to regress the uncertainty estimation instead of using a fixed learnable parameter, and the uncertainty regression is also a task in proposed model.

In this paper, we mainly focus on the aleatoric uncertainty to improve the robustness of learning-based odometry estimation in real-world scenarios, which indicates the error caused by noises from the data itself. As a supervised method to learn the aleatoric uncertainty [4], the proposed model leverages the following loss function to optimize the learnable parameters:

$$loss = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\sigma(x_i)^2} \|y_i - f(x_i)\|^2 + \frac{1}{2} log\sigma(x_i)^2 \tag{9}$$

where σ is the estimated uncertainty for the input x; y_i and $f(x_i)$ denote the ground truth and pose prediction of input x_i, respectively.

Considering Eq. 7 and Eq. 9, we would like to propose a 6D pose estimation model with uncertainty estimation. As shown in Fig. 4, we assume the sliding-window size is 4, there are 4 LiDAR frames $[i-1, i+2]$. By using 6D pose regressor, we can obtain the predicted relative pose $[f2f_i, f2f_{i+1}, f2f_{i+2}]$ of frame-to-frame $(f2f)$ in $\mathbf{se(3)}$ [30] (as the orange nodes in Fig. 4), which can be described by following formula:

$$\mathbf{se(3)} = \left\{ \xi \in \mathbb{R}^6 = \begin{bmatrix} \rho \\ \phi \end{bmatrix}, \rho \in \mathbb{R}^3, \phi \in so(3) \right\} \tag{10}$$

In order to make full use of frames in one sliding window, we propose to predict the transformations between frames for

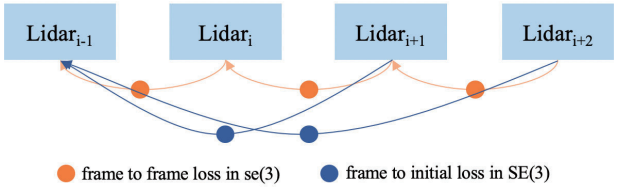

Fig. 4. Frame to frame $(f2f)$ and to initial $(f2g)$ loss in sliding window.

the initial frame $(f2g)$ in the window $[f2g_{i+1}, f2g_{i+2}]$, which are represented as blue nodes in Fig. 4. Inspired by VINet [26] that uses both $se(3)$ and $SE(3)$ in pose regression, we apply $SE(3)$ for the frame to initial loss term, which can be formulated as follows:

$$\begin{aligned} \mathbf{SE(3)} = \mathbf{T} \in \mathbb{R}^{4\times4} &= \left\{ \begin{pmatrix} R & t \\ 0^T & 0 \end{pmatrix} \mid R \in SO(3), t \in \mathbb{R}^{3\times3} \right\}, \\ \hat{\xi} \in \mathbb{R}^{4\times4} &= \begin{bmatrix} \hat{\phi} & \rho \\ 0^T & 0 \end{bmatrix} \end{aligned} \tag{11}$$

where T is transformation; $\hat{\xi}$ denotes a vector to anti-symmetric matrix. In specific rotation representation, we choose quaternion for the full concatenated pose in $SE(3)$ by following the study of [26]. The prediction in f2f and f2g are represented as $y_i^f = (\rho_i^f, \phi_i^f)$ and $y_i^g = (t_i^g, q_i^g)$, where q_i^g denotes quaternion. The corresponding ground truths are written as $\hat{y}_i^f$ and $\hat{y}_i^g$. Also, we formulate the loss function $L(\theta, \sigma_x, \sigma_y, \sigma_z, \sigma_r)$ as a function of our network, where θ is the weights of network and $\sigma(x, y, z, r)$ denotes the uncertainties in x, y, z translation and rotation. Specifically, it can be calculated by:

$$\begin{aligned} L_x(\theta, \sigma_x) = \textstyle\sum_i \frac{1}{2} exp(-s_x^i)([\left\| \rho_{ix}^f - \hat{\rho}_{ix}^f \right\|^2, \\ \left\| t_{ix}^g - \hat{t}_{ix}^g \right\|^2]) + \frac{1}{2} s_x^i \end{aligned} \tag{12}$$

$$\begin{aligned} L_r(\theta, \sigma_r) = \textstyle\sum_i \frac{1}{2} exp(-s_r^i)([\left\| \phi_i^f - \hat{\phi}_i^f \right\|^2, \\ \left\| \frac{q_i^g}{\|q_i^g\|} - \hat{q}_{ix}^g \right\|^2]) + \frac{1}{2} s_r^i \end{aligned} \tag{13}$$

where $\rho_i^f \in \mathbb{R}^3 = (\rho_{ix}^f, \rho_{iy}^f, \rho_{iz}^f)$ includes translation in x, y and z axis. By following the study of [4], we use log variance $s_i = log\sigma_i^2$. By using the learnable uncertainties, we could weigh between translation and orientation error, instead of using well-designed weighting hyperparameters, which is similar to Kendall *et al.* [31] uses uncertainty to weigh semantic, instance segmentation and depth estimation tasks. Finally, the total loss can be computed as:

$$\begin{aligned} L(\theta, \sigma_x, \sigma_y, \sigma_z, \sigma_r) = \sum_i (L_x(\theta, \sigma_x) + L_y(\theta, \sigma_y) + \\ L_z(\theta, \sigma_z) + L_r(\theta, \sigma_r)) \end{aligned} \tag{14}$$

The reason why there is only one scalar value to represent the uncertainty in rotation instead of angles in roll, pitch and yaw is that, the representations of Euler angles used in [8]

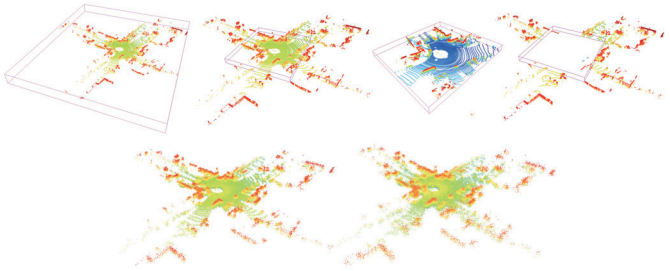

Fig. 5. Top row: Cropped LiDAR point cloud. From left to right: initialized axis-aligned bounding box, randomly reduce the size of bounding box, the point cloud inside of bounding box, and crop the point cloud are inside of the bounding box. Maximum cropped points are 50% number of total points. Bottom row: The point cloud with Gaussian noise (Std. deviation on left and right are 0.1 and 0.5).

[23] and angle axis have singularity problem compared with quaternion. Therefore, we do not use them as the representation of orientation in the loss function suggested by [32].

IV. Experiments

A. Implementation Details

The architecture is implemented with PyTorch and trained on an NVIDIA Quadro GV100 (VRAM:32GB) GPU. We train the proposed model on sequences 00-08 of KITTI dataset with totally 250 epochs, and test our model on sequences 09-10. The inference speed on this GPU is about 25fps, which could meet the real-time requirement. The network is trained with a batch size of 8 and window size of 4. We use Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with an initial learning rate $lr = 1e^{-4}$ for the training. To avoid over-fitting, the dropout rate is set as 0.1 after convolutional layers of the feature extractor. We use evo [33] package as the trajectory plot visualization tool. The KITTI odometry evaluation tool[2] is used for quantitative results. The details of network architecture are listed in Table.II.

B. Degraded KITTI

In the real world, degraded LiDAR data are usually holes from highly reflective areas and noise from sunlight [34]. Considering these two situations, we artificially generate the degraded point cloud by randomly cropping the raw KITTI LiDAR data using a bounding box, as shown in Fig. 5. The initial bounding box is generated from Axis-Aligned-Bounding-Box (AABB), then we randomly reduce its size to crop the point clouds. To imitate the noise that comes from sunlight, we add the Gaussian noise (standard deviation ranges from 0.1 to 1) to raw data, as shown in Fig. 5.

C. Trajectory Result

In Fig. 6, we show the trajectory comparison results of ablation study on two different feature fusion methods, w/ and w/o uncertainty weighted loss. The baseline means the framework applies the summation fusion, instead of using the attentional feature fusion and the loss function of Eq. 7, the default value of β of the baseline is the same with DeepVO [8]. From the trajectory results, we can observe that both attentional mechanisms and uncertainty loss can significantly improve the performance.

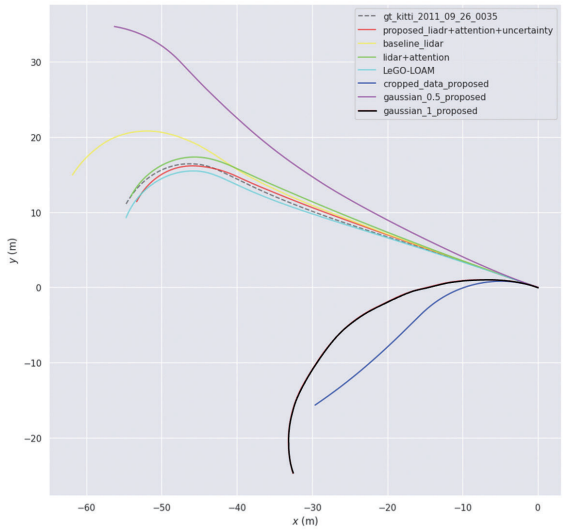

Fig. 6. Ablation test (trajectory 2nd-4th), Robustness test on degraded KITTI with the proposed method (trajectory 6th-8th).

The last three trajectories are the models used for testing the robustness of proposed on degraded KITTI dataset. By comparing with the geometry-based method (results are shown in Fig. 3), the proposed learning-based method is more sensitive to the cropped dataset. One of the possible reasons is that the cropped data directly loses information compared with the noise-added data, which makes it difficult for CNN to extract representative features.

We also plot the result of geometry-based LeGO-LOAM [15] LiDAR odometry method in Fig. 6. Besides, the quantitative results are reported in Table. I. We evaluate geometry-based (ORB-SLAM [35]) and learning-based visual odometry (DeepVO[3]) in Fig. 7 and 8. For a fair comparison, ORB-SLAM [35] is tested under monocular without closure loop detection module. To obtain the correct scale, we use the Umeyama alignment in the evo [33] package. We observe that the proposed method could obtain competitive performance. However, in comparison with Lo-Net [28], the rotation performance could be improved by fusing IMU in the future.

D. Uncertainty Results

We evaluate the uncertainty in real road scenarios, as shown in Fig. 9. In this sequence, the car accelerates in the x direction, then decelerates to start the turn, and finally turns right. During acceleration, we find that the uncertainty in the x-direction is high, which also affects the stability of the z-axis. When the speed slows down (start from around 40^{th}), the uncertainty of y-axis increases. During turns (starting from around 75^{th}),

[2]https://github.com/LeoQLi/KITTI_odometry_evaluation_tool

[3]https://github.com/ChiWeiHsiao/DeepVO-pytorch

TABLE I
ODOMETRY RESULT ON KITTI DATASET.

Sequence	ORB-SLAM [3] [36]		DeepVO [8]		DeepLO [16]		LeGO-LOAM [37]		Lo-Net [37]		CertainOdom	
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
09	15.30	**0.26**	/	/	13.35	4.45	0.98	1.97	0.58	1.37	**0.53 (↑8.6%)**	1.25
10	3.68	**0.48**	8.11	8.83	5.83	3.53	0.92	2.21	0.93	1.80	**0.82 (↑10.9%)**	1.06
00	25.29	7.37	/	/	/	/	1.05	2.17	**0.91**	**0.92**	1.09	2.58
01	/	/	/	/	/	/	1.02	13.4	**0.47**	1.36	0.52	**1.26 (↑7.4%)**
02	/	/	/	/	/	/	1.01	2.17	0.71	1.52	**0.62 (↑12.7%)**	**1.12 (↑26.3%)**
03	21.07	18.36	8.49	6.89	/	/	1.18	2.34	0.66	**1.03**	**0.63 (↑4.5%)**	1.12
04	4.46	5.60	7.19	6.97	/	/	1.01	1.27	0.65	**0.51**	**0.58 (↑10.8%)**	0.72
05	26.01	10.62	2.62	3.61	/	/	0.74	1.28	0.69	**1.04**	**0.53 (↑23.2%)**	1.29
06	17.47	17.17	5.42	5.82	/	/	0.63	1.06	**0.50**	0.71	0.53	**0.62 (↑12.7%)**
07	24.53	10.83	3.91	4.60	/	/	**0.81**	1.12	0.89	1.70	0.96	1.77
08	32.40	12.13	/	/	/	/	0.94	1.99	**0.77**	2.12	1.23	**1.52 (↑23.6%)**

• t_{rel}: average sequence translational RMSE (%) on the length of 100m, 200m, ..., 800m.
• r_{rel}: average sequence rotational RMSE (deg/100m) on the length of 100m, 200m, ..., 800m.
• The performance results of existed methods are collected from the cited literature. ↑ is relative to the second-best.

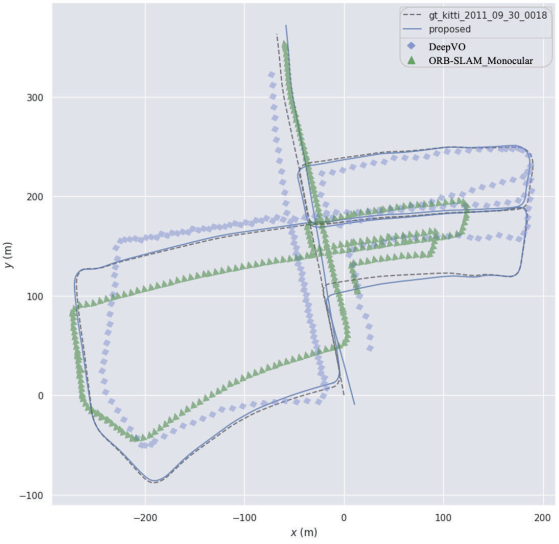

Fig. 7. Results on Sequence 05 of KITTI dataset [12].

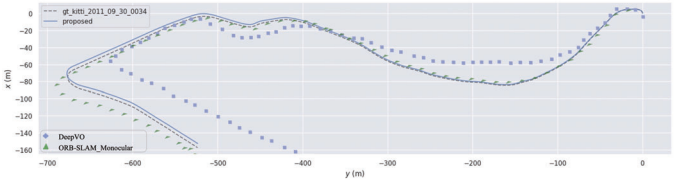

Fig. 8. Results on Sequence 10 (testing set) of KITTI dataset [12].

the uncertainties in the y axis and orientation are increased. When a car encounters a dynamic object such as a person in (d), it could be reflected in the z direction. The uncertainty is highest in the main driving direction x-axis.

In Fig. 10, we plot the ground truth, prediction pose of relative LiDAR frames and uncertainty estimation σ. By following the evaluation method of uncertainty in MDN-VO [20] and [19], we draw *prediction* $\pm$ 1σ and *prediction* $\pm$ 3σ bounded area in each axis and orientation, as depicted in the 1^{st} and 3^{rd} columns. To better understand the relationship

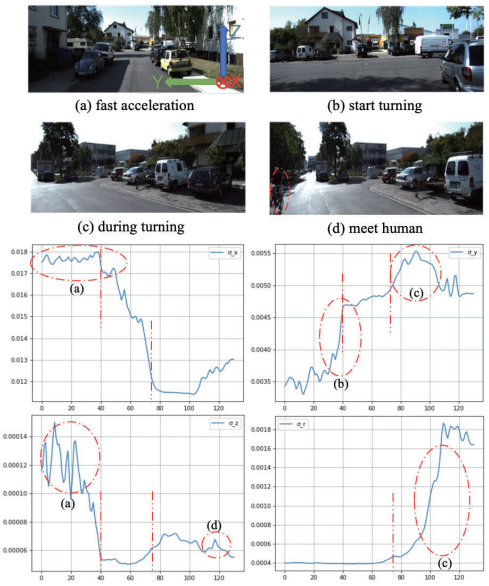

Fig. 9. Uncertainty evaluation with real road scenarios. Results are uncertainty in the x/y/z axis and orientation.

between prediction error and uncertainty, we also plot them in the 2^{nd} and 4^{th} columns of Fig. 10. As can be seen from Fig. 10, most prediction errors are bounded in the $\pm 1\sigma$ area. In addition, the uncertainty also reflects the fluctuation of error in the most of cases.

We also evaluate the uncertainty on degraded KITTI (Gaussian noise w/ 0.5 deviation) dataset, we find that both the uncertainty and error are increased, as depicted in Fig. 11. Based on all these results, we validate that the proposed model could learn the representations of uncertainty estimation for odometry prediction.

V. CONCLUSION AND FUTURE WORK

This paper proposes a multi-task learning framework CertainOdom, which leverages uncertainty weighted loss for LiDAR odometry estimation. Different from existing models using manually tuned parameters, the proposed method adopts

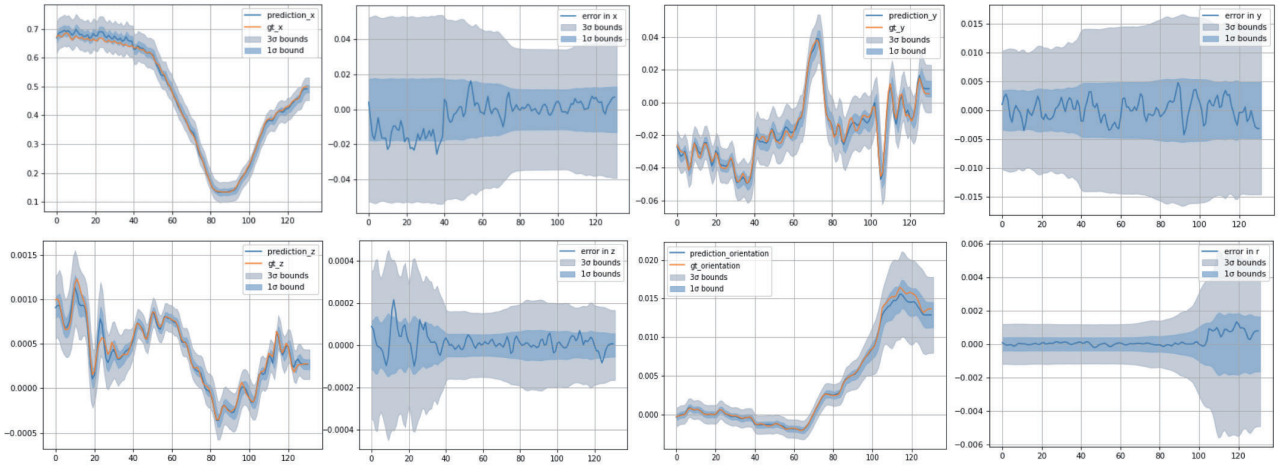

Fig. 10. Uncertainty estimation with prediction, ground truth and error in each axis and orientation. From left top to right bottom: x, y, z axis and orientation.

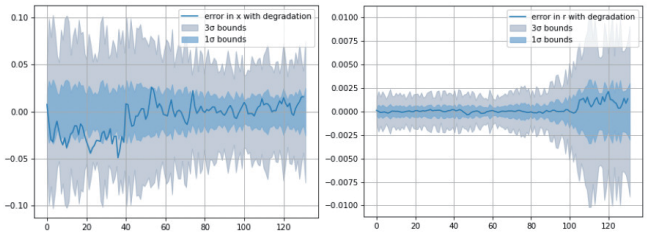

Fig. 11. Uncertainty estimation with the degraded dataset. From left to right: x-axis, orientation.

the uncertainties regressed from multi-decoders to balance the error in translation and rotation. The odometry results are evaluated on raw KITTI dataset and degraded KITTI dataset, we observe the different sensitivities between geometry-based and learning-based solution. The proposed method can not only obtain competitive odometry performance, but also explain uncertainty qualitatively and quantitatively. However, to the best of our knowledge, the interpretation of uncertainty could not be applied to all situations, some noise values are still difficult to interpret, which also exists in related works [19] [20]. One of the possible reasons is that the uncertainty estimation cannot deal with out-of-distribution data [38] [39], thus, it is necessary to improve the robustness of uncertainty estimation in the future work.

ACKNOWLEDGMENT

This research was partially supported by a research project grant from the JST-SPRING, the grant number is JPMJSP2124.

REFERENCES

[1] L. Sun, F. Kanehiro, I. Kumagai, and Y. Yoshiyasu, "Multi-purpose slam framework for dynamic environment," in *2020 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2020, pp. 519–524.

[2] L. Sun, R. P. Singh, and F. Kanehiro, "Visual slam framework based on segmentation with the improvement of loop closure detection in dynamic environments," *Journal of Robotics and Mechatronics*, vol. 33, no. 6, pp. 1385–1397, 2021.

[3] J.-W. Bian, H. Zhan, N. Wang, Z. Li, L. Zhang, C. Shen, M.-M. Cheng, and I. Reid, "Unsupervised scale-consistent depth learning from video," *International Journal of Computer Vision (IJCV)*, 2021.

[4] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[5] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1281–1292.

[6] S. Gasperini, J. Haug, M.-A. N. Mahani, A. Marcos-Ramiro, N. Navab, B. Busam, and F. Tombari, "Certainnet: Sampling-free uncertainty estimation for object detection," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 698–705, 2022.

[7] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2938–2946.

[8] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2043–2050.

[9] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, "Pointseg: Real-time semantic segmentation based on 3d lidar point cloud," *ArXiv*, vol. abs/1807.06288, 2018.

[10] Y. Cho, G. Kim, and A. Kim, "Deeplo: Geometry-aware deep lidar odometry," *arXiv preprint arXiv:1902.10562*, 2019.

[11] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3560–3569.

[12] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[13] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2020, pp. 5135–5142.

[14] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[15] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.

[16] Y. Cho, G. Kim, and A. Kim, "Unsupervised geometry-aware deep lidar odometry," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2145–2152.

[17] J. Nubert, S. Khattak, and M. Hutter, "Self-supervised learning of lidar odometry for robotic applications," in *2021 IEEE International*

TABLE II
NETWORK ARCHITECTURE.

Encoder for point cloud and normal estimation projection	
[input]	Batch size(B) * Window size(S) -1 * 2 * 3 * 60 * 720
[layer1]	Sequential(Conv2d(6, 64, kernel = (3,5), stride = (1,2), padding = (1,2)), BatchNorm2d(64, 0.1), ReLU(inplace = True))
[layer2]	MaxPool2d(kernel = 3, stride = (1,2), padding = 1)
[layer3]	Sequential(FireLayer [40](64, 16, 64, 64, bn = True, 0.1), FireLayer(128, 16, 64, 64, bn = True, 0.1), Sequeeze and excitation layer form SEnet [41](128,2), MaxPool2d(kernel = 3, stride = (1,2), padding = 1))
[layer4]	Sequential(FireLayer(128, 32, 128, 128, bn = True, 0.1), FireLayer(256, 32, 128, 128, bn = True, 0.1), Sequeeze and excitation layer form SEnet(256,2), MaxPool2d(kernel = 3, stride = (1,2), padding = 1))
[layer5]	Sequential(FireLayer(256, 48, 192, 192, bn = True, 0.1), FireLayer(384, 48, 192, 192, bn = True, 0.1), FireLayer(384, 64, 256, 256, bn = True, 0.1), FireLayer(512, 64, 256, 256, bn = True, 0.1), Sequeeze and excitation layer form SEnet(512,2), MaxPool2d(kernel = 3, stride = (2,2), padding = 1))
[layer6]	Sequential(FireLayer(512, 64, 256, 256, bn = True, 0.1), FireLayer(512, 64, 256, 256, bn = True, 0.1), Sequeeze and excitation layer form SEnet(512,2), MaxPool2d(kernel = 3, stride = (2,2), padding = 1))
[layer7]	Sequential(FireLayer(512, 80, 384, 384, bn = True, 0.1), FireLayer(768, 80, 384, 384, bn = True, 0.1), Drouoput = 0.1)
[output]	After concatenation $\rightarrow$ 2 * B * W - 1 * 512
Attentional feature fusion	
[input]	x[0] $\rightarrow$ B * W - 1 * 512 and x[1] $\rightarrow$ B * W - 1 * 512

[Steps]	y = x[0] + x[1];
	x_1 = local_att [11](y);
	x_2 = global_att [11](y);
	x_1_2 = x_1 + x_2;
	weight = sigmoid(x_1_2);
	x_ = x[0] * weight + x[1] * (1 - weight);
	x_1' = local_att2(x_);
	x_2' = global_att(x_);
	x_1_2' = x_1' + x_2';
	weight' = sigmoid(x_1_2');
	x[0] = x[0] * weight';
	x[1] = x[1] * (1 - weight');
	output = x[0] + x[1]

[output]	B * W - 1 * 512
Decoders for 6D pose regression and uncertainty	
[input]	B * W - 1 * 512
[layer1]	B-LSTM, 2-layers, hidden size = 1024, Dropout = 0.1
[layer2]	Dropout = 0.2
[layer3]	Linear(1024,3) for translation
[layer3']	Linear(1024,4) for orientation
[layer3"]	Linear(1024,3) for uncertainty of translation
[layer3"']	Linear(1024,1) for uncertainty of orientation
[output]	B * W - 1 * 3/ 4/ 3/ 1

Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 9601–9607.

[18] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *2016 IEEE international conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4762–4769.

[19] Y. Kim, S. Yoon, S. Kim, and A. Kim, "Unsupervised balanced covariance learning for visual-inertial sensor fusion," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 819–826, 2021.

[20] N. Kaygusuz, O. Mendez, and R. Bowden, "Mdn-vo: Estimating visual odometry with confidence," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3528–3533.

[21] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and Accurate LiDAR Semantic Segmentation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[22] F. Moosmann, *Interlacing self-localization, moving object tracking and mapping for 3d range sensors*. KIT Scientific Publishing, 2013, vol. 24.

[23] L. Liu, G. Li, and T. H. Li, "Atvio: Attention guided visual-inertial odometry," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 4125–4129.

[24] W. Wang, B. Wang, P. Zhao, C. Chen, R. Clark, B. Yang, A. Markham, and N. Trigoni, "Pointloc: Deep pose regressor for lidar point cloud localization," *IEEE Sensors Journal*, vol. 22, no. 1, pp. 959–968, 2021.

[25] R. Gao, X. Xiao, W. Xing, C. Li, and L. Liu, "Unsupervised learning of monocular depth and ego-motion in outdoor/indoor environments," *IEEE Internet of Things Journal*, 2022.

[26] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[27] C. Chen, S. Rosa, C. X. Lu, B. Wang, N. Trigoni, and A. Markham, "Learning selective sensor fusion for state estimation," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[28] Q. Li, S. Chen, C. Wang, X. Li, C. Wen, M. Cheng, and J. Li, "Lo-net: Deep real-time lidar odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8473–8482.

[29] Y. Shavit, R. Ferens, and Y. Keller, "Learning multi-scene absolute pose regression with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2733–2742.

[30] L. Han, Y. Lin, G. Du, and S. Lian, "Deepvio: Self-supervised deep learning of monocular visual inertial odometry using 3d geometric constraints," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6906–6913.

[31] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7482–7491.

[32] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6555–6564.

[33] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[34] A. Swatantran, H. Tang, T. Barrett, P. DeCola, and R. Dubayah, "Rapid, high-resolution forest structure and terrain mapping over large areas using single photon lidar," *Scientific reports*, vol. 6, no. 1, pp. 1–12, 2016.

[35] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[36] J. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid, "Unsupervised scale-consistent depth and ego-motion learning from monocular video," *Advances in neural information processing systems*, vol. 32, 2019.

[37] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin2: Ultra light lidar-based slam using geometric approximation applied with kl-divergence," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 619–11 625.

[38] J. van Amersfoort, L. Smith, A. Jesson, O. Key, and Y. Gal, "On feature collapse and deep kernel learning for single forward pass uncertainty," *arXiv preprint arXiv:2102.11409*, 2021.

[39] Y. Liu, M. Pagliardini, T. Chavdarova, and S. U. Stich, "The peril of popular deep learning uncertainty estimation methods," *arXiv preprint arXiv:2112.05000*, 2021.

[40] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1887–1893.

[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.