DEEP LIDAR ODOMETRY

Qing Li[1], Cheng Wang[1,2,*], Shaoyang Chen[1], Xin Li[3], Chenglu Wen[1], Ming Cheng[1], Jonathan Li[1,4]

[1]Fujian Key Laboratory of Sensing and Computing for Smart City and the School of Information Science and Engineering, Xiamen University, Xiamen 361005, China
[2]Fujian Collaborative Innovation Center for Big Data Applications in Governments, Fuzhou 350003, China
[3]Geometric and Visual Computing Group, Louisiana State University, LA 70808, USA
[4]GeoSTARS Lab, the Department of Geography and Environmental Management, University of Waterloo, Canada

Commission I, WG I/6

KEY WORDS: Lidar, Odometry, Motion Estimation, Neural Network, Mask, Point Cloud

ABSTRACT:

Most existing lidar odometry estimation strategies are formulated under a standard framework that includes feature selection, and pose estimation through feature matching. In this work, we present a novel pipeline called LO-Net for lidar odometry estimation from 3D lidar scanning data using deep convolutional networks. The network is trained in an end-to-end manner, it infers 6-DoF poses from the encoded sequential lidar data. Based on the new designed mask-weighted geometric constraint loss, the network automatically learns effective feature representation for the lidar odometry estimation problem, and implicitly exploits the sequential dependencies and dynamics. Experiments on benchmark datasets demonstrate that LO-Net has similar accuracy with the geometry-based approach.

1. INTRODUCTION

Among the applications in robotics and autonomous driving, motion estimation and map-building are among the most fundamental prerequisites. Most modern mobile platforms rely on lidars or cameras for 3D geometry perception. Many researchers are working to achieve real-time 6 degree-of-freedom simultaneous localization and mapping (SLAM) with camera-based and lidar-based approaches. Compared to cameras, lidars can provide a 360-degree view with one sensor, acquire more accurate distance information, and are not sensitive to lighting conditions. Although camera-based methods have advantages in loop-closure detection, their sensitivity to illumination and viewpoint change may make such capabilities unreliable. Laser-based mapping and localization methods have been extensively studied in the field of robotics. These methods can function even at night, and the high resolution of many 3D lidars allows for the capture of fine details of an environment at long ranges. Therefore, our work focuses on using 3D lidar to achieve real-time motion estimation and mapping. While lidars provide accurate 3D point cloud measurements, estimating the motion between two consecutive laser scans is a complicated task, due to the sparse and non-uniform nature of the point clouds, as well as the missing appearance information. Moreover, characteristic patterns, such as circular rings, produced by a moving scanner, can easily mislead local correspondence estimation algorithms.

Motion estimate of the mobile platforms can be used as a prior when aligning consecutive laser scans, allowing for fast and relatively accurate mapping. The variety of approaches that exists either focus on efficiency, for example when used for autonomous navigation, or on accuracy when building high-fidelity maps. Errors caused by pairwise scan registration in lidar odometry, lead to misalignments and degeneration in mapping. Registering laser scans to a map which is built by aggregating previous measurements, often minimizes accumulated error. In addition, graph-based optimization is also adopted to minimize accumulated errors (Kümmerle et al., 2011, Frese et al., 2005, Olson et al., 2006), but the optimization is computationally demanding for large maps. Developing an accurate and robust real-time lidar odometry estimation and mapping system is desirable.

In this work, we design a deep neural network architecture for lidar odometry estimation problems. We accumulate the motion specific features by incorporating pairwise scans, interpret the spatial relations of scans by applying normal consistency, and locate the effective area by fusing mask prediction. We are inspired by the recent CNNs-based camera localization and pose regression works (Zhou et al., 2017, Abhinav V., 2018, Kendall et al., 2017, Yang et al., 2018b) in the design of network structure, as well as the traditional lidar odometry methods (Zhang , Singh, 2014, Moosmann , Stiller, 2011, Deschaud, 2018) in the aspect of lidar mapping. In summary, the main contributions of our work are as follows: 1) We propose a novel scan-to-scan lidar odometry estimation network which simultaneously learns to estimate the normal and mask as an auxiliary task. 2) We incorporate the temporal spatial geometric consistency constraint into the network, which provides higher order interaction between consecutive scans and better regularizes the learning of odometry. We perform comprehensive evaluation on the two commonly used benchmark datasets *KITTI* (Geiger et al., 2013) and *Ford Campus Vision and Lidar Data Set* (Pandey et al., 2011). Experiment results manifest that our framework can effectively improve the accuracy and robustness of traditional geometry-based approach. To the best of our knowledge, our proposed method is the first neural network regression model that is comparable to traditional geometry feature-based techniques for 3D lidar odometry estimation.

*Corresponding author – cwang@xmu.edu.cn

2. RELATED WORK

Classical registration methods used in pose estimation include Iterative Closest Point (ICP) (Pomerleau et al., 2013, Besl , McKay, 1992) and its variants (Besl , McKay, 1992, Pomerleau et al., 2013) and feature-based approaches (Rusu et al., 2009, Velas et al., 2016).

By finding correspondences at a point-wise level, ICP aligns two sets of points iteratively by minimizing distance between corresponding points until stopping criteria are satisfied. When the scans include large quantities of points, ICP may suffer from prohibitive computational cost, and points from the current lidar scanning data may miss their spatial counterparts in next scan due to sparsity of scan resolution. To this end, many variants of ICP have been proposed to improve its efficiency and accuracy (Rusinkiewicz , Levoy, 2001). Feature-based matching methods are attracting more attention, as they require less computational resources by extracting representative features from the data. These features should be suitable for effective matching and invariant of point-of-view. However, most existing feature-based methods do not take into account the factors in the environment that may inhibit the odometry estimation, such as dynamic objects. A low-drift and real-time lidar odometry and mapping (LOAM) method is proposed in (Zhang , Singh, 2014, Zhang , Singh, 2017), and has been considered as the state-of-the-art lidar motion estimation method. It extracts the line and surface features in lidar data and performs point feature to edge and plane scan-matching to find correspondences between scans. LOAM dose not consider the dynamic objects in the scene and achieves low-drift and real-time odometry estimation by having two modules running in parallel. The estimated motion of scan-to-scan registration is used to correct the distortion of laser scans and guarantee the real-time performance. The high accuracy odometry estimation is produced by registering onto a map to cancel the drift.

Recently, deep learning based methods have outperformed classical approaches in many computer vision tasks. Many Convolutional Neural Networks (CNNs) architectures and training models have been proposed. Despite their success in many 2D vision problems, the exploration of developing effective CNNs to process 3D geometric data, such as 6-DoF pose estimation, has been limited. More recently, the methods that using CNNs to regress the 6-DoF pose from RGB images have been explored (Wang et al., 2017, Zhou et al., 2017, Yang et al., 2018a, Yin , Shi, 2018). But these methods inevitably suffer from the inaccurate depth prediction and scale drift.

3. METHOD

3.1 Data Encoding and Normal Estimation

To convert the original sparse and irregular point clouds into a structured representation, we encode the lidar data into point cloud matrices by a cylindrical projection (Chen et al., 2017). Given a 3D point $p = (x, y, z)$ in the lidar coordinate system, the projection function can be formulated as

$$\begin{aligned} \alpha &= \arctan(y/x)/\Delta\alpha \\ \beta &= \arcsin(z/\sqrt{x^2+y^2+z^2})/\Delta\beta \end{aligned} \quad (1)$$

where α and β are the indexes which set the points' positions in the matrix. $\Delta\alpha$ and $\Delta\beta$ are the average angular resolution between consecutive beam emitters in the horizontal and vertical directions, respectively. The element at (α, β) is assigned intensity value and range value of the lidar point p. We keep the point closer to the lidar when multiple points are projected into a same position. After applying this projection on the lidar data, we get a matrix of size $H \times W \times C$, and C is the number of matrix channels.

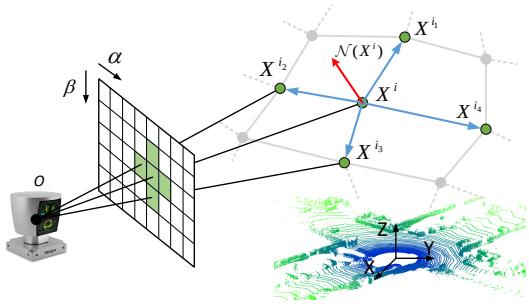


Figure 1. Illustration of data encoding and normal estimation.

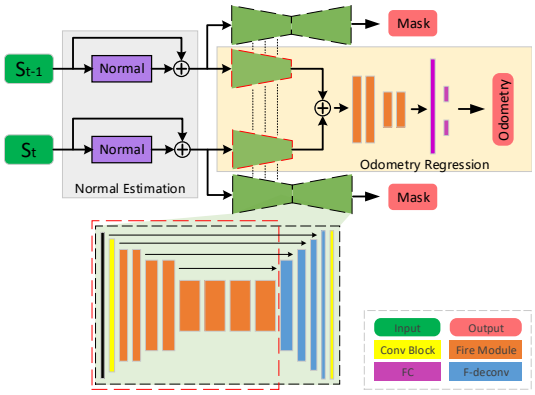


Figure 2. Network architecture of feature extraction layers (red dashed line) and mask prediction layers (black dashed line) . Our network takes two consecutive lidar scans as input and infers the relative 6-DoF pose.

As shown in Figure 1, given a 3D point X^i and its k neighbors X^{ij}, $j = 1, 2, \ldots, k$ on the grid, we simplify the normal estimation by computing the weighted cross products over X^i's four neighbors. Then we smooth normal vectors using a moving average filter (Moosmann, 2013). This can be formulated as

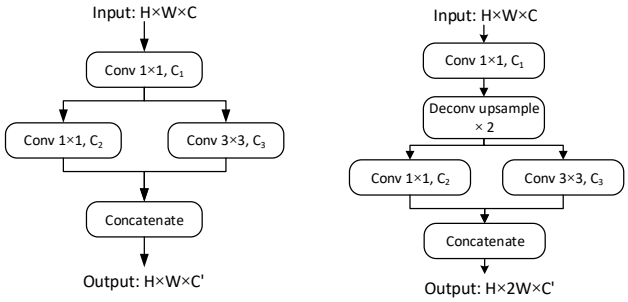
$$\mathcal{N}(X^i) = \sum_{X^{ik}, X^{ij} \in \mathcal{P}} (w_{ik}(X^{ik} - X^i) \times w_{ij}(X^{ij} - X^i)) \quad (2)$$

where $(X^{ik} - X^i)$ is a 3D vector, w_{ik} is the weight of X^{ik} with respect to X^i. We set $w_{ik} = \exp(-0.2|r(X^{ik}) - r(X^i)|)$ to put more weight on points which have similar range value r with X^i, and less weight otherwise. $\mathcal{P}$ is the set of neighboring points of X^i, such as $\{X^{i_1}, X^{i_2}, X^{i_3}, X^{i_4}\}$ in Figure 1.

3.2 Network Structure

As shown in Figure 2, our LO-Net takes two consecutive scans $(S_{t-1}; S_t)$ as input and jointly estimates the 6-DoF relative pose between the scans, point-wise normal vector, and a mask of moving objects for each scan.

Lidar point clouds are considered as the 3D model of the scene, and often contain dynamic objects such as cars and



(a) fireConv (b) fireDeonv

Figure 3. The structure of fireConv and fireDeconv from SqueezeNet.

pedestrians in the road environment. These factors may inhibit the learning pipeline of odometry regression. Based on the encoder-decoder architecture (see Figure 2), we deploy a mask prediction network (Zhou et al., 2017, Yang et al., 2017) to learn the compensation for dynamic objects, and improve the effectiveness of the learned features and the robustness of the network.

In order to reduce the number of model parameters and computation cost, and make it is possible to run the network on a low performance platform, such as mobile robot or backpack system. We replace most of convolutional layers of the network with fireConv, and deconvolutional layers with fireDeconv. The fireConv and fireDeconv module of SqueezeNet (Iandola et al., 2016) can construct a light-weight network that can achieve similar performance as AlexNet (Krizhevsky et al., 2012). Structures of the two modules are shown in Figure 3. The first 1×1 convolution compresses the input tensor. The second 1×1 convolution and the 3×3 convolution let the network to learn more feature representations from different kernel sizes.

Since the width of intermediate features is much larger than its height, we only down-sample the width by maxpooling during feature extraction. The reweighing layer (Wang et al., 2018) is adopted to learn a more robust feature representation. For mask prediction, the fireDeconv is used to up-sample the feature maps and get the original scale resolution point-wise mask prediction. To infer the 6-DoF relative pose between the two input scans, we concatenate output features from the enlargement layer (Wang et al., 2018) of mask prediction layers. The last two fully-connected layers output the translation x and rotation quaternion q, respectively.

The network parameters are shown in Table 1 and 2. All convolutional and deconvolutional layers are followed by Rectified Linear Unit (ReLU) (Glorot et al., 2011) except for the last output layers where nonlinear activation function is applied. We experimented with batch normalization performed on convolutional layers, data normalization and rescaling, however they did not yield significant performance gains, rather in some cases they negatively affected the odometry accuracy.

3.3 Loss Function

We use $\mathcal{L}_x(S_{t-1}; S_t)$ and $\mathcal{L}_q(S_{t-1}; S_t)$ to demonstrate how to learn the relative translational and rotational components, respectively.

$$\begin{aligned} \mathcal{L}_x(S_{t-1}; S_t) &= \|x_t - \hat{x}_t\|_2 \\ \mathcal{L}_q(S_{t-1}; S_t) &= \left\| q_t - \frac{\hat{q}_t}{\|\hat{q}_t\|} \right\|_2 \end{aligned} \tag{3}$$

Table 1. Network parameters of mask prediction layers. For fireConv and fireDeconv layers, the filter size is represented as $C_1 - C_2 - C_3$ as shown in Figure 3.

Layer	Filter size	Kernel size / Stride
conv 1	64	3 / 1×2
maxpooling		3 / 1×2
fireConv 1	16-64-64	
fireConv 2	16-64-64	
maxpooling + reweighing 1		3 / 1×2
fireConv 3	32-128-128	
fireConv 4	32-128-128	
maxpooling + reweighing 2		3 / 1×2
fireConv 5	48-192-192	
fireConv 6	48-192-192	
fireConv 7	64-256-256	
fireConv 8	64-256-256	
enlargement + reweighing 3		
fireDeconv 1	64-128-128	
fireDeconv 2	64-64-64	
fireDeconv 3	16-32-32	
fireDeconv 4	16-32-32	
dropout (0.5)		
conv 2	2	3 / 1×1

Table 2. Network parameters of odometry regression layers.

Layer	Filter size	Kernel size / Stride
fireConv 1	64-256-256	
fireConv 2	64-256-256	
maxpooling + reweighing 1		3 / 2×2
fireConv 3	80-384-384	
fireConv 4	80-384-384	
maxpooling		3 / 2×2
fc 1	512	
dropout (0.5)		
fc 2	3	
fc 3	4	

where x_t and q_t are the ground truth relative translational and rotational components, $\hat{x}_t$ and $\hat{q}_t$ denote their predicted counterparts. Due to the difference in scale and units between the translational and rotational pose components, previous works (Kendall et al., 2015, Wang et al., 2017) gave a weight regularizer λ to the rotational loss to jointly learn the 6-DoF pose. However, the hyper-parameter λ need to be tuned when using new data from different scene. To avoid this problem, we use two learnable parameters s_x and s_q to balance the scale between the translational and rotational components in the loss term (Kendall et al., 2017).

$$\begin{aligned} \mathcal{L}_o = \mathcal{L}_x(S_{t-1}; S_t) \exp(-s_x) + s_x \\ + \mathcal{L}_q(S_{t-1}; S_t) \exp(-s_q) + s_q \end{aligned} \tag{4}$$

We use the initial values of $s_x = 0.0$ and $s_q = -2.5$ for all scenes during the training.

Let $X_{t-1}^{\alpha\beta}$ and $X_t^{\alpha\beta}$ be the spatial corresponding point elements of the consecutive data matrices S_{t-1} and S_t, respectively. We can derive $\hat{X}_t^{\alpha\beta}$ from $X_{t-1}^{\alpha\beta}$ through

$$\hat{X}_t^{\alpha\beta} = P T_t P^{-1} X_{t-1}^{\alpha\beta} \tag{5}$$

where T_t is the relative rigid pose transformation between the consecutive scans. P denotes the projection process and P^{-1} is its inverse operation. Therefore, $\hat{X}_t^{\alpha\beta}$ and $X_t^{\alpha\beta}$ are a pair of matching elements, and we can measure the similarity between corresponding elements to verify the correctness of pose transformation. In this work, we compare the normal $\mathcal{N}(x)$ as it reflects smooth surface layouts and clear edge structures of the road environment. Thus, the constraint of pose

transformation can be formulated as minimizing

$$\mathcal{L}_n = \sum_{\alpha\beta} \|\mathcal{N}(\hat{X}_t^{\alpha\beta}) - \mathcal{N}(X_t^{\alpha\beta})\|_1 \cdot e^{|\nabla r(\hat{X}_t^{\alpha\beta})|} \quad (6)$$

where $\nabla r(\hat{X}_t^{\alpha\beta})$ is a local range smooth measurement, ∇ is the differential operator with α and β. The item $e^{|\cdot|}$ allows the loss function to focus more on sharply changing areas in the scene.

The predicted mask $\mathcal{M}(X_t^{\alpha\beta}) \in [0, 1]$ indicates the area where geometric consistency can be modeled or not, and implicitly ensures the reliability of the features learned in the pose regression network. Therefore, the geometric consistency error as formulated in Equation (6) is weighted by

$$\mathcal{L}_n = \sum_{\alpha\beta} \mathcal{M}(X_t^{\alpha\beta}) \|\mathcal{N}(\hat{X}_t^{\alpha\beta}) - \mathcal{N}(X_t^{\alpha\beta})\|_1 \cdot e^{|\nabla r(\hat{X}_t^{\alpha\beta})|} . \quad (7)$$

There is no ground truth label or supervision to train the mask prediction. To avoid the network minimize the loss by setting all values of the predicted mask to be 0, we add a cross-entropy loss as a regularization term

$$\mathcal{L}_r = -\sum_{\alpha\beta} \log P(\mathcal{M}(X_t^{\alpha\beta}) = 1). \quad (8)$$

In summary, our final objective function to minimize for odometry regression is

$$\mathcal{L} = \mathcal{L}_o + \lambda_n \mathcal{L}_n + \lambda_r \mathcal{L}_r \quad (9)$$

where λ_n and λ_r are the weighting factors for geometric consistency loss and mask regularization, respectively.

4. EXPERIMENTS

Implementation details. The point cloud data we use is captured by the Velodyne HDL-64 3D lidar sensor. During encoding the data matrix, we set $H = 64$ and $W = 1800$ by considering the sparseness of point clouds. The width of input data matrix are resized to 1792 by cropping both ends of the matrix. The whole framework is implemented with the popular Tensorflow library (Abadi et al., 2016). During the training, the mask prediction network is pre-trained using KITTI 3D object detection dataset, and all layers are trained simultaneously. The loss weights of Equation (9) are set to be $\lambda_n = 0.15$ and $\lambda_r = 0.05$, and the batch size is 8. We choose the Adam (Kingma , Ba, 2014) solver with default parameters for optimization. The network is trained on an NVIDIA 1080 Ti GPU.

Baselines. We compare our approach with several existing lidar odometry estimation methods: ICP-point2point (ICP-po2po), ICP-point2plane (ICP-po2pl), GICP (Segal et al., 2009), CLS (Velas et al., 2016) and Velas et al. (Velas et al., 2018). The first two ICP methods are implemented using the Point Cloud Library (Rusu , Cousins, 2011). As far as we know, (Velas et al., 2018) is the only deep learning based lidar odometry method that has comparable results. Loop closure detection is not implemented for all methods since we aim to test the limits of accurate odometry estimation.

We firstly conduct the training and testing experiments on the KITTI dataset. Then, based on the model trained only on the KITTI dataset, we directly test the model on the Ford dataset. We use the KITTI odometry evaluation metrics (Geiger et al., 2012) to quantitatively analyze the accuracy of odometry estimation. Table 3 shows the evaluation results of the methods on KITTI and Ford datasets. Although there are differences between the two datasets, such as different lidar calibration parameters and different systems for obtaining ground truth, our approach still achieves the best average performance among evaluated methods. Some trajectories produced by different methods are shown in Figure 4. Figure 5 shows the average evaluation errors on KITTI Seq. 00-10.

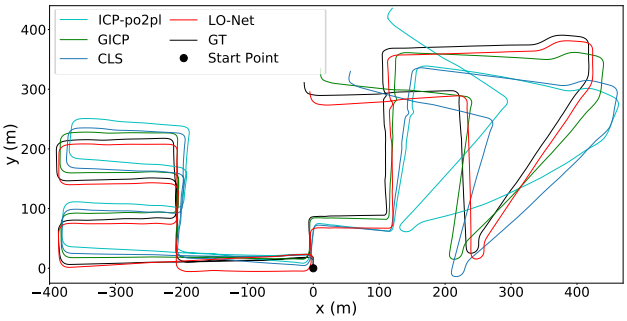


Figure 4. Trajectory plots of KITTI Seq. 08 with ground truth. The results of ICP-po2po are not shown as its large scale drift.

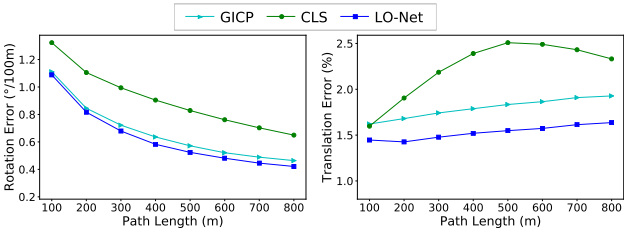


Figure 5. Evaluation results on the KITTI Seq. 00-10. We show the average errors of translation and rotation with respect to path length intervals. LO-Net achieves the best performance among all evaluated methods. The results of ICP-po2po and ICP-po2pl are not shown as its large error.

5. CONCLUSIONS

We presented LO-Net, a novel learning pipeline for lidar odometry estimation. We introduced the weighted geometric consistency constraint for regressing 6-DOF poses that consistent with the motion model. We evaluated our approach on two public datasets and showed the significant performance improvement over prior works. In our future work, we plan to incorporate recurrent units into our network to utilize the long-term temporal features, and investigate in more detail the geometry feature representation learned by the network to enable the whole framework to work in an end-to-end manner without costly collected ground truth data.

ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China (No. U1605254, 61728206), and the National Science Foundation of USA under Grants EAR-1760582.

Table 3. Odometry results on KITTI and Ford datasets. Our network is trained on KITTI sequences and then tested on the two datasets.

Seq.	ICP-po2po		ICP-po2pl		GICP (Segal et al., 2009)		CLS (Velas et al., 2016)		Velas et al. (Velas et al., 2018)[1]		LO-Net	
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
00†	6.88	2.99	3.80	1.73	**1.29**	**0.64**	2.11	0.95	3.02	NA	1.47	0.72
01†	11.21	2.58	13.53	2.58	4.39	0.91	4.22	1.05	4.44	NA	**1.36**	**0.47**
02†	8.21	3.39	9.00	2.74	2.53	0.77	2.29	0.86	3.42	NA	**1.52**	**0.71**
03†	11.07	5.05	2.72	1.63	1.68	1.08	1.63	1.09	4.94	NA	**1.03**	**0.66**
04†	6.64	4.02	2.96	2.58	3.76	1.07	1.59	0.71	1.77	NA	**0.51**	**0.65**
05†	3.97	1.93	2.29	1.08	**1.02**	**0.54**	1.98	0.92	2.35	NA	1.04	0.69
06†	1.95	1.59	1.77	1.00	0.92	**0.46**	0.92	**0.46**	1.88	NA	**0.71**	0.50
07*	5.17	3.35	1.55	1.42	**0.64**	**0.45**	1.04	0.73	1.77	NA	1.70	0.89
08*	10.04	4.93	4.42	2.14	**1.58**	**0.75**	2.14	1.05	2.89	NA	2.12	0.77
09*	6.93	2.89	3.95	1.71	1.97	0.77	1.95	0.92	4.94	NA	**1.37**	**0.58**
10*	8.91	4.74	6.13	2.60	**1.31**	**0.62**	3.46	1.28	3.27	NA	1.80	0.93
mean†	7.13	3.08	5.15	1.91	2.23	0.78	2.11	0.86	3.12	NA	**1.09**	**0.63**
mean*	7.76	3.98	4.01	1.97	**1.38**	**0.65**	2.15	1.00	3.22	NA	1.75	0.79
Ford-1	8.20	2.64	3.35	1.65	3.07	1.17	10.54	3.90	NA	NA	**2.27**	**0.62**
Ford-2	16.23	2.84	5.68	1.96	5.11	1.47	14.78	4.60	NA	NA	**2.18**	**0.59**

[1]: The results on KITTI dataset are taken from (Velas et al., 2018), and the results on Ford dataset are not available.
†: The sequences of KITTI dataset that are used to train LO-Net.
*: The sequences of KITTI dataset that are not used to train LO-Net.
t_{rel}: Average translational RMSE (%) on length of 100m-800m.
r_{rel}: Average rotational RMSE (°/100m) on length of 100m-800m.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al., 2016. Tensorflow: a system for large-scale machine learning. *OSDI*, 16, 265–283.

Abhinav V., Noha R., Wolfram B., 2018. Deep auxiliary learning for visual localization and odometry. *Proceedings Of The IEEE International Conference On Robotics And Automation (ICRA)*.

Besl, P.J., McKay, N.D., 1992. A Method for Rgistration of 3-D Shapes. *IEEE Transaction on Pattern Analisys and Machine Intelligence*, 14, 239–256.

Chen, X., Ma, H., Wan, J., Li, B., Xia, T., 2017. Multi-view 3d object detection network for autonomous driving. *IEEE CVPR*, 1number 2, 3.

Deschaud, J., 2018. IMLS-SLAM: scan-to-model matching based on 3D data. *arXiv preprint arXiv:1802.08633*.

Frese, U., Larsson, P., Duckett, T., 2005. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21, 196–207.

Geiger, A., Lenz, P., Stiller, C., Urtasun, R., 2013. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32, 1231–1237.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, IEEE, 3354–3361.

Glorot, X., Bordes, A., Bengio, Y., 2011. Deep sparse rectifier neural networks. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 315–323.

Iandola, F. N, Han, S., Moskewicz, M. W, Ashraf, K., Dally, W. J, Keutzer, K., 2016. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*.

Kendall, A., Cipolla, R. et al., 2017. Geometric loss functions for camera pose regression with deep learning. *Proc. CVPR*, 3, 8.

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

Kingma, D. P, Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., Hinton, G. E, 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097–1105.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W., 2011. g 2 o: A general framework for graph optimization. *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, 3607–3613.

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

Moosmann, F., Stiller, C., 2011. Velodyne slam. *Intelligent Vehicles Symposium (IV), 2011 IEEE*, IEEE, 393–398.

Olson, E., Leonard, J., Teller, S., 2006. Fast iterative alignment of pose graphs with poor initial estimates. *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, IEEE, 2262–2269.

Pandey, G., McBride, J. R, Eustice, R. M, 2011. Ford campus vision and lidar data set. *The International Journal of Robotics Research*, 30, 1543–1552.

Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S., 2013. Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34, 133–148.

Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the icp algorithm. *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, IEEE, 145–152.

Rusu, R. B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (fpfh) for 3d registration. *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, Citeseer, 3212–3217.

Rusu, R. B., Cousins, S., 2011. 3d is here: Point cloud library (pcl). *Robotics and automation (ICRA), 2011 IEEE International Conference on*, IEEE, 1–4.

Segal, A., Haehnel, D., Thrun, S., 2009. Generalized-icp. *Robotics: science and systems*, 2number 4, 435.

Velas, M., Spanel, M., Herout, A., 2016. Collar line segments for fast odometry estimation from velodyne point clouds. *ICRA*, 4486–4495.

Velas, M., Spanel, M., Hradis, M., Herout, A., 2018. Cnn for imu assisted odometry estimation using velodyne lidar. *Autonomous Robot Systems and Competitions (ICARSC), 2018 IEEE International Conference on*, IEEE, 71–77.

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

Wang, Y., Shi, T., Yun, P., Tai, L., Liu, M., 2018. PointSeg: Real-Time Semantic Segmentation Based on 3D LiDAR Point Cloud. *arXiv preprint arXiv:1807.06288*.

Yang, N., Wang, R., Stueckler, J., Cremers, D., 2018a. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. *European Conference on Computer Vision (ECCV)*. accepted as oral presentation, to appear, arXiv 1807.02570.

Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., 2018b. Lego: Learning edge with geometry all at once by watching videos. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 225–234.

Yang, Z., Wang, P., Xu, W., Zhao, L., Nevatia, R., 2017. Unsupervised Learning of Geometry with Edge-aware Depth-Normal Consistency. *arXiv preprint arXiv:1711.03665*.

Yin, Z., Shi, J., 2018. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2.

Zhang, J., Singh, S., 2014. Loam: Lidar odometry and mapping in real-time. *Robotics: Science and Systems*, 2, 9.

Zhang, J., Singh, S., 2017. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, 41, 401–416.

Zhou, T., Brown, M., Snavely, N., Lowe, D. G, 2017. Unsupervised learning of depth and ego-motion from video. *CVPR*, 2number 6, 7.