TransLO: A Window-Based Masked Point Transformer Framework for Large-Scale LiDAR Odometry

Jiuming Liu[1]*, Guangming Wang[1]*, Chaokang Jiang[2], Zhe Liu[3], Hesheng Wang[1]†

[1]Department of Automation, Key Laboratory of System Control and Information Processing of Ministry of Education, Key Laboratory of Marine Intelligent Equipment and System of Ministry of Education, Shanghai Jiao Tong University
[2]Engineering Research Center of Intelligent Control for Underground Space, Ministry of Education, School of Information and Control Engineering, Advanced Robotics Research Center, China University of Mining and Technology
[3]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China
{liujiuming,wangguangming,liuzhesjtu,wanghesheng}@sjtu.edu.cn
ts20060079a31@cumt.edu.cn

Abstract

Recently, transformer architecture has gained great success in the computer vision community, such as image classification, object detection, etc. Nonetheless, its application for 3D vision remains to be explored, given that point cloud is inherently sparse, irregular, and unordered. Furthermore, existing point transformer frameworks usually feed raw point cloud of $N \times 3$ dimension into transformers, which limits the point processing scale because of their quadratic computational costs to the input size N. In this paper, we rethink the structure of point transformer. Instead of directly applying transformer to points, our network (TransLO) can process tens of thousands of points simultaneously by projecting points onto a 2D surface and then feeding them into a local transformer with linear complexity. Specifically, it is mainly composed of two components: Window-based Masked transformer with Self Attention (WMSA) to capture long-range dependencies; Masked Cross-Frame Attention (MCFA) to associate two frames and predict pose estimation. To deal with the sparsity issue of point cloud, we propose a binary mask to remove invalid and dynamic points. To our knowledge, this is the first transformer-based LiDAR odometry network. The experiment results on the KITTI odometry dataset show that our average rotational and translational RMSE achieves 0.500 °/100m and 0.993 % respectively. The performance of our network surpasses all recent learning-based methods and even outperforms LOAM on most evaluation sequences. Codes will be released on https://github.com/IRMVLab/TransLO.

Introduction

LiDAR odometry plays a fundamental role in perceiving the environment for various applications, including Simultaneous Localization and Mapping (SLAM) system (Zhu et al. 2022), robot navigation (Wang et al. 2018), autonomous driving (Zheng and Zhu 2021), etc. As depicted in Fig. 1, the target of this task is to estimate pose transformation between two consecutive point cloud frames (Li et al. 2019).

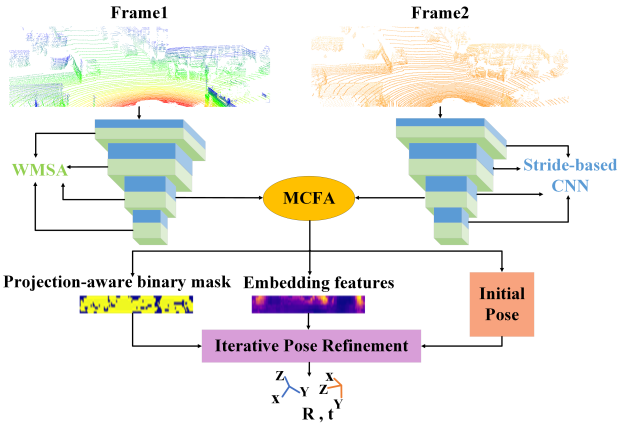


Figure 1: The pipeline of our LiDAR odometry. We design a hierarchical transformer-based odometry network. Pose between two consecutive point cloud frames is estimated by the cross-attention mechanism and iterative refinement.

Traditional methods treat LiDAR odometry as a scan registration problem where the Iterative Closest Point algorithm (ICP) (Besl and McKay 1992) is generally employed to refine the transformation. Some feature-based methods (Zheng et al. 2020; Zhou et al. 2022) are also proposed using point, line, or planar features. Since soft correspondences of point clouds can be learned by convolutional neural networks (CNNs) in an end-to-end manner, learning-based methods have attracted increasing attention in recent years. However, CNNs focus on the interrelation between local areas and enlarge the receptive field as the network deepens. This will create a problem: CNN can't capture relationships between patches that have similar features but locate over great distances. To overcome this limitation, we further explore transformer-based LiDAR odometry for long-range dependencies, based on our observation that global descriptors are helpful for correlating above similar patches.

We observe that some intrinsic qualities of transformer fit fairly well with the task: 1) Self-attention is naturally invariant to permutation of input. Point cloud is irregular

*These authors contributed equally.
†Corresponding Author.

and unordered, which means it also calls for a permutation-invariant set operator. 2) As mentioned above, CNNs only focus on local features while transformer architecture can capture global features. This characteristic is important since some dynamic objects and invalid points are not suitable for ego-motion estimation. These outliers can be filtered out by the global learning of feature maps. 3) Local transformer methods reduce computational costs to linearity complexity, which enables the network to process large-scale 3D data.

The aforementioned three aspects support our research intuition to develop a transformer backbone for LiDAR odometry. Motivated by the motion information estimation method from soft correspondences in optical flow (Wang, Ren, and Wang 2022) and scene flow tasks (Wang et al. 2022a,b, 2021a), we directly estimate pose from conditioned features after the cross attention. Overall, our contributions are as follows:

- For the large-scale odometry task, we propose a novel hierarchical transformer-based network. We rethink the structure of point transformer. Instead of original point clouds or their feature embeddings as input, we first project point clouds on a cylindrical surface to acquire a pseudo image. This procedure not only solves the irregularity problem of point clouds but also facilitates subsequent research since existing transformer methods for images can be extended to the 3D vision community.
- Window attention is applied (WMSA) in each feature extraction layer consuming linear computational costs. Furthermore, a cross-frame transformer module (MCFA) is designed to associate two consecutive frames. We also design a projection-aware binary mask in both self and cross attention modules above to represent whether each pixel-wise position is invalid in sparse feature maps.
- Extensive experiments carried out on the KITTI odometry dataset (Geiger et al. 2013) indicate our method outperforms all existing traditional and learning-based odometry methods. Our performance is even superior to LOAM with mapping on most sequences. To the best of our knowledge, this is the first time that transformer architecture is employed in the LiDAR odometry task.

Related Work

Traditional LiDAR Odometry

Iterative Closest Point (ICP) (Besl and McKay 1992) is the most common and popular method to align two point clouds, which calculates translation and rotation matrix by iterative updates until convergence. After that, two ICP variants (point-to-line ICP (Censi 2008) and point-to-plane ICP (Low 2004)) are proposed. As a renowned searching tree-based method, LOAM (Zhang and Singh 2014) pioneered a series of research. Edge and planar features are extracted respectively to estimate motion transformation between two point cloud frames. However, it doesn't discern ground and non-ground points which undermines accuracy and computational cost. Some subsequent variants (Shan and Englot 2018; Kuettel and Ferrari 2012) employ ground segmentation or ground-constrained for improving their performance.

Deep LiDAR Odometry

Although deep learning has gained great progress in visual odometry (Li et al. 2020; Yang et al. 2020), deep LiDAR odometry is still underdeveloped. Nicolai et al. (Nicolai et al. 2016) first investigate this field by projecting point clouds to a 2D plane and applying convolution layers to them. However, their experiment results are undesirable. DeepLO (Cho, Kim, and Kim 2019) proposes the first unsupervised LiDAR odometry framework. LO-Net (Li et al. 2019) presents a deep convolution network for real-time odometry which can implicitly exploit the sequential dependencies in the data. Recently, many works are devoted to learning-based LiDAR odometry estimation in both supervised (Li et al. 2021; Wang et al. 2022c) and unsupervised (Cho, Kim, and Kim 2020; Wang et al. 2020) approaches. Among these networks, PWCLO-Net (Wang et al. 2021b) designs a hierarchical embedding mask optimization pipeline to address 3D LiDAR odometry and achieves state-of-the-art performance.

Transformer in Computer Vision

Transformer (Vaswani et al. 2017) is proposed to tackle the machine translation problem and now serves as a mainstream backbone in NLP. Inspired by its scaling successes in NLP, more researchers attempt to extend it to various computer vision tasks, such as semantic segmentation (Xie et al. 2021; Liu et al. 2022; Zhang et al. 2022), object detection (Carion et al. 2020; Yin et al. 2021), image classification (Dosovitskiy et al. 2021) etc. Nevertheless, transformer for point cloud processing remains rarely studied, which are mostly designed for indoor or object-level tasks. Moreover, existing works (Pan et al. 2021; Zhao et al. 2021; Guo et al. 2021; Fischer et al. 2021) always feed raw points (for example, 1024 points) into transformer. It blocks large-scale applications due to their quadratic complexity.

To achieve the linear complexity (Ren et al. 2022a; Dong et al. 2022; Ren et al. 2022b), Swin Transformer (Liu et al. 2021) partitions an image into a series of non-overlapped windows and computes attention within each window. Its performance surpasses previous pipelines by a large margin. Inspired by Swin Transformer, we leverage window-based cross attention modules to associate two frames. Also, our work is the first to leverage spatial shift on the point cloud association task. Intuitively, shift operation can capture more global features, which is significant for learning dynamics globally and improving accuracy effectively.

Proposed Method

Overall Architecture

An overview of our proposed methods is demonstrated in Fig. 2. Given two consecutive frames of point clouds PC_1 and PC_2, odometry aims to regress the ego-motion between two frames. To be specific, we first project point clouds onto a cylindrical surface to get pseudo images. Then, several stride-based sampling layers coupled with Window-based Masked Self Attention (WMSA) are employed to encode feature embeddings. Additionally, we enlarge the receptive field through CNNs instead of patch merging methods in the vanilla Swin Transformer. To regress pose, a Masked Cross

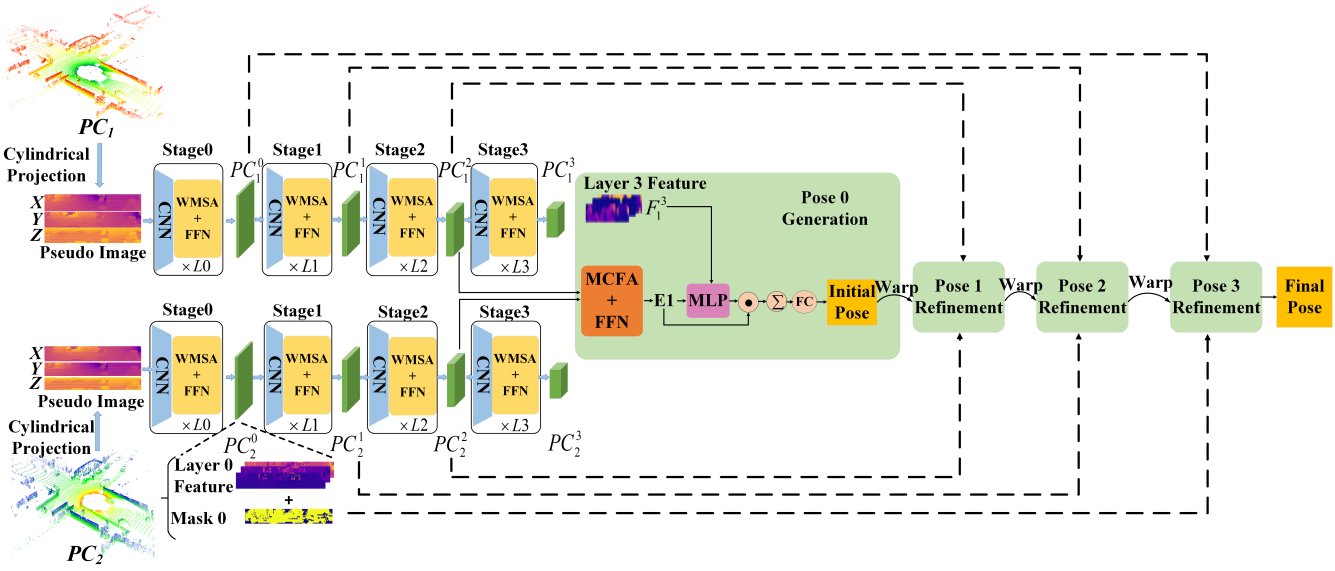


Figure 2: TransLO. Feature extraction of our TransLO is composed of both CNNs and transformer with Window-based Masked Self Attention (WMSA). FFN is the abbreviation of a feed-forward network. Masked Cross Frame Attention (MCFA) block is employed to associate two frames and output initial embeddings in layer 2. Initial embeddings and features of Layer 3 are input to MLP and Fully Connected (FC) layers for pose generation. This pose will warp features from upper layers through upsampling. MCFA and MLP layers are then employed again to refine the pose iteratively.

Frame Attention (MCFA) module is designed. Even though point clouds have an ordered representation after projection, their sparsity issue is not intuitively solved. Thus, a projection-aware mask indicating which coordinate has invalid points is added to both WMSA and WCFA. Finally, the pose warping operation in (Wang et al. 2021b) is introduced for iterative pose refinement.

Cylindrical Projection and Feature Extraction

Cylindrical Projection: To obtain the corresponding 2D structured grid-like coordinates, every raw 3D point cloud is projected as the following formulas:

$$u = arctan2(y/x)/\Delta\theta, \tag{1}$$

$$v = arcsin(z/\sqrt{x^2+y^2+z^2})/\Delta\phi, \tag{2}$$

in which x, y, z represent raw 3D coordinates of point clouds and $\Delta\theta, \Delta\phi$ are horizontal and vertical resolutions of LiDAR sensor. The output u, v are 2D corresponding coordinates. Previous works usually fill 2D coordinate positions with depth or intensity values and then apply 2D CNNs. To make the best use of raw geometric information, we directly fill raw 3D coordinates into their corresponding projected 2D positions instead. This procedure outputs a tensor with the size of H×W×3, where the last dimension means the original XYZ coordinates of point clouds.

Efficient Stride-based Sampling and Grouping: Classic 3D points learning methods (Qi et al. 2017a,b) apply Farthest Point Sampling (FPS) and KNN to sample and group features, which is time-consuming. We propose a stride-based sampling strategy as introduced in Fig. 4. A series of fixed-stride indexes are first generated as 2D kernel centers, and M neighbor points for each kernel are grouped and gathered within a certain radius. This design is more effective than random sampling because we can compute each kernel operation in parallel on CUDA. The formula is as follows:

$$f_i^G = \underset{m=1,2,\cdots,M}{MAX}(MLP((x_i^m - x_i) \oplus f_i^m \oplus f_i)), \tag{3}$$

where x_i is the $i\text{-}th$ sampled point (kernel center) and x_i^m represents its $m-th$ neighbor point. f_i and f_i^m are their features. f_i^G means the output feature. Operator $\oplus$ means concatenation of vectors. The feature extraction process above is adopted in every sampling layer coupled with the WMSA block, which will be illustrated in the next section.

Window-based Masked Self Attention (WMSA)

Validity Argument of Shifted Window: Inspired by Swin Transformer (Liu et al. 2021), our attention blocks retain window partition and shift operations, but also possess different characteristics. First of all, our LiDAR covers 360° in the horizontal direction but has a limited field vertically. This leads to strip-like feature maps where the width is rather larger than the height. Thus, we need to reason whether window partition and shift still work in advance.

The visualization result is displayed in Fig. 3, we set the window size as 4 and shift size as 2. The scale of the input feature map is 4×16. Upper figure in Fig. 3 shows how cyclic shift works toward the top-left direction. Then, shifted feature map is divided into four windows. There are several sub-windows in each partitioned window that are originally far apart before shift, eg. B and C in window 3. In this case, the masking process is used for preventing cross-sub-window attention computation in each window. As in Fig. 3,

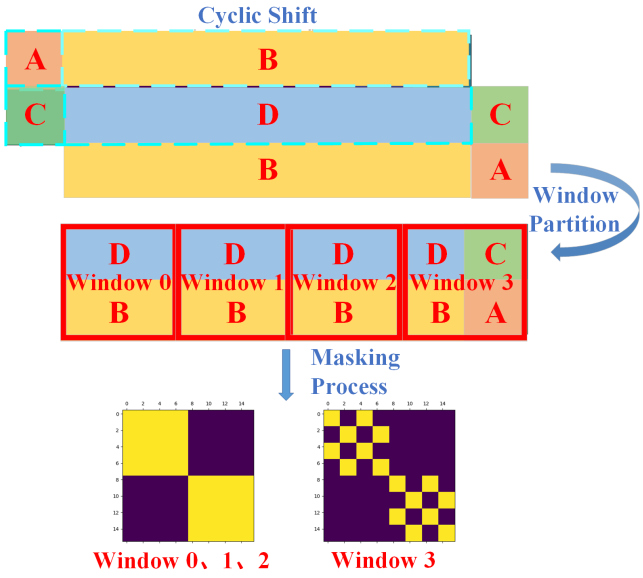


Figure 3: Cyclic shift and window partition. Upper figure shows how cyclic shift works and blue dotted lines indicate unshifted feature maps. Then, shifted feature map is divided into four 4×4 windows. The masking process is utilized to prevent calculating attention among different sub-windows.

we visualize how masking process works, where tensors of each window are flattened and attention weights will be only calculated within the same sub-window (colored yellow).

Binary Mask and Window-based Point Transformer: Our proposed point transformer is built by replacing the standard self-attention module with a window-based masked self-attention (WMSA) block as described in Fig. 4. Due to the inherent sparsity of point clouds, our projected feature maps are filled with many invalid pixels. Therefore, a projection-aware binary mask is designed as:

$$mask = \begin{cases} 0, & X = 0, Y = 0, Z = 0 \\ 1, & otherwise\ . \end{cases} \tag{4}$$

Note that this binary mask is different from the aforementioned masking mechanism for cutting the cross-subwindow connection. This mask is introduced due to the sparsity characteristic of point clouds and pixel-corresponding to projected feature maps. Though it is simple, further studies carried out in the experiment section prove its significance. Following vanilla transformer architecture, one complete transformer stage in Fig. 2 can be described as:

$$\hat{F}^l = WMSA(LN(F^{l-1})) + F^{l-1}; \tag{5}$$

$$F^l = MLP(LN(\hat{F}^l)) + \hat{F}^l; \tag{6}$$

$$\hat{F}^{l+1} = sWMSA(LN(F^l)) + F^l; \tag{7}$$

$$F^{l+1} = MLP(LN(\hat{F}^{l+1})) + \hat{F}^{l+1}; \tag{8}$$

where WMSA (sWMSA) represents (shifted) Window-based Masked Self Attention. F^{l-1} and F^{l+1} are input and output features for stage l. MLP denotes the feed-forward network (FFN) with GELU non-linearity. Also,

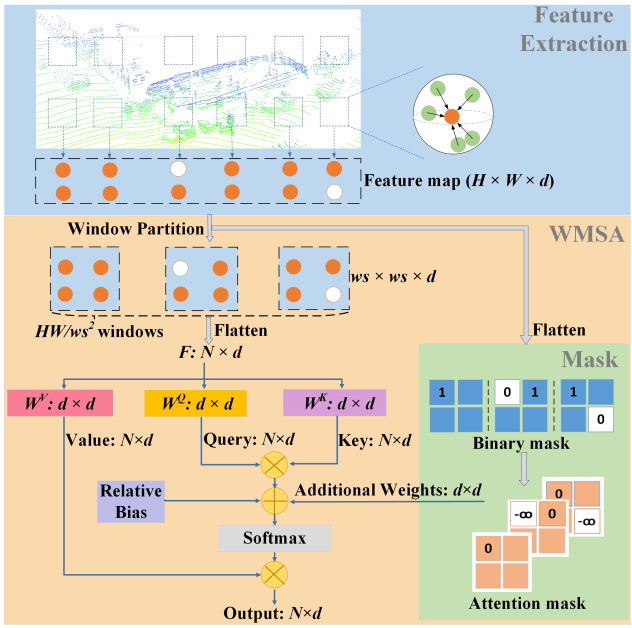


Figure 4: Window-based masked self attention (WMSA). Our attention block modifies the classic multi-head self-attention with shifted windows and a projection-aware binary mask. ws means window size. After partitioning, tensors within each window ($ws \times ws$) will be flattened to N.

LayerNorm (LN) and residual connections are applied respectively before and after each module. For feature $F^l = \{f_k | f_k \in \mathbb{R}^d\}_{k=1}^N$, WMSA (sWMSA) can be calculated as:

$$WMSA(F^l) = (Head_1 \oplus \cdots \oplus Head_H)W^O, \tag{9}$$

$$\begin{aligned} Head_h &= Attn(F^l \times W_h^Q, F^l \times W_h^K, F^l \times W_h^V) \\ &= Attn(Q^h, K^h, V^h) \\ &= softmax(\frac{Q^h K^h}{\sqrt{d_{head}}} + attn\_mask + Bias)V^h, \end{aligned} \tag{10}$$

where $Head_1 \cdots Head_H$ represent the output of H heads in Multi-head self-attention. $W_h^Q \in \mathbb{R}^{d \times d_{head}}$, $W_h^K \in \mathbb{R}^{d \times d_{head}}$, $W_h^V \in \mathbb{R}^{d \times d_{head}}$, $W^O \in \mathbb{R}^{Hd_{head} \times d}$ are learned projected functions. $Bias$ is the relative position encoding operation. And $attn\_mask$ is the attention mask generated by the binary mask as illustrated in Fig. 4:

$$attn\_mask = \begin{cases} 1 \times e^{-10}, & where\ mask = 0 \\ 0, & otherwise\ . \end{cases} \tag{11}$$

Masked Cross Frame Attention (MCFA)

The masked cross-frame transformer is built for estimating ego-motion in both the initial pose generation and pose refinement module. Similar to WMSA, shifted window and binary masks are also used in our Masked Cross Frame Attention (MCFA) as shown in Algorithm 1.

Take the first frame for example, F_1^s and M_1^s are first fed into a WMSA module, which enables points to interact with other points in the same frame. Q_1^s, K_1^s, and V_1^s in WMSA are all from the same input frame through three independent learnable projection matrices just like the formula (10). Then, output features of WMSA F_1^I will attend to F_2^I for ego-motion estimation. Specifically, F_1^I is linearly projected as $query$ (Q_1^I), F_2^I is linearly projected as key (K_2^I) and $value$ (V_2^I), and we calculate attention weights by entering them into the same (s)WMSA block. This step enables the network to learn relative position transformation between two frames. Finally, we reverse window partition and shift and output transformed correlated features ${F_1}^{L_s}$, ${F_2}^{L_s}$ for generating the initial motion embedding $E1 = \{e_i | e_i \in \mathbb{R}^C\}_{i=1}^N$ as illustrated in Fig. 2. Then, a weighting parameter is calculated as:

$$W = softmax(MLP(E1 \oplus F_1^3)), \tag{12}$$

where $W = \{w_i | w_i \in \mathbb{R}^C\}_{i=1}^N$. The initial pose is then generated by:

$$q^3 = \frac{FC(\sum_{i=1}^{n} e_i \odot w_i)}{|FC(\sum_{i=1}^{n} e_i \odot w_i)|}, \tag{13}$$

$$t^3 = FC(\sum_{i=1}^{n} e_i \odot w_i), \tag{14}$$

where FC denotes the fully connected layer. The quaternion $q_3 \in \mathbb{R}^4$ and translation vector $t_3 \in \mathbb{R}^3$ are refined from coarse to fine as in (Wang et al. 2021b).

Loss Function

Our network outputs the poses from four layers and adopts a multi-scale supervised loss. $\mathcal{L}^l = \mathcal{L}_t^l + \lambda \mathcal{L}_r^l$ denotes the training loss of the layer l in our network. Translation error $\mathcal{L}_t^l$ and rotation error $\mathcal{L}_r^l$ can be calculated as:

$$\mathcal{L}_t^l = \|t_{gt}^l - t^l\|_2, \tag{15}$$

$$\mathcal{L}_r^l = \|q_{gt}^l - \frac{q^l}{\|q^l\|}\|_2, \tag{16}$$

where q^l, t^l and q_{gt}^l, t_{gt}^l are the predicted and ground truth poses respectively.

Experiments

Dataset and Implement Details

We evaluate our network performance on the KITTI odometry dataset which is widely used in pose estimation and point cloud registration tasks. We set the number of input points as 150000 and the initial feature map after cylindrical projection as 64 (H)×1792 (W). Note that this feature map size should be divisible by window size 4. All training and evaluation experiments are conducted on a single NVIDIA Titan RTX GPU with PyTorch 1.10.1. The Adam optimizer is adopted with β_1 = 0.9, β_2 = 0.999. The initial learning rate is 0.001 and exponentially decays every 200000 steps until 0.00001. The batch size is set as 8.

Algorithm 1: MCFA

Input: Uncorrelated point cloud features F_1^s , F_2^s and their corresponding masks M_1^s, M_2^s of two consecutive frames in stage s.
Parameter: Number of transformer blocks in stage s: L_s.
Output: Correlated point cloud features ${F_1}^{L_s}, {F_2}^{L_s}$.

Let $i = 0$. (Parameter i is used for indicating whether the cyclic shift is employed in WMSA.)
while $i <= L_s$ **do**
  Points attend to other points in the same frame.
  $F_1^I = WMSA(LN(Q_1^s, K_1^s, V_1^s), M_1^s) + F_1^s$.
  $F_2^I = WMSA(LN(Q_2^s, K_2^s, V_2^s), M_2^s) + F_2^s$.
  if $i\%2 == 0$ **then**
    Points attend to the other frame points (without shift).
    $F_1^i = WMSA(LN(Q_1^I, K_2^I, V_2^I), M_1^s) + F_1^I$.
    $F_2^i = WMSA(LN(Q_2^I, K_1^I, V_1^I), M_2^s) + F_2^I$.
  else
    Points attend to the other frame points (with shift).
    $F_1^i = sWMSA(LN(Q_1^I, K_2^I, V_2^I), M_1^s) + F_1^I$.
    $F_2^i = sWMSA(LN(Q_2^I, K_1^I, V_1^I), M_2^s) + F_2^I$.
  end if
  $i += 1$.
end while
return ${F_1}^{L_s}, {F_2}^{L_s}$

Comparison with The State-of-the-Art

We compare our network with the state-of-the-art on KITTI dataset, including both classic methods and learning-based ones. Since existing training and testing sequence settings are inconsistent in different methods, we test and evaluate our framework accordingly for a fair comparison.

00-06 as training sequences and 07-10 as testing sequences. For classic odometry, we compare our performance with CLS (Velas, Spanel, and Herout 2016), GICP (Segal, Haehnel, and Thrun 2009), LOAM (Zhang and Singh 2017; Shan and Englot 2018), SUMA (Behley and Stachniss 2018), and (Vizzo et al. 2021). As demonstrated in Table 1, quantitative results illustrate the rotation RMSE (°/100m) and translation error (%) of our network are extremely smaller than theirs. Compared with full LOAM which dominates the KITTI odometry benchmark for years, our approach outperforms it on most sequences. For learning-based odometry SelfVoxeLO (Xu et al. 2020), LO-Net (Li et al. 2019), SfMLearner (Zhou et al. 2017), and RSLO (Xu et al. 2022), we observe that our odometry accuracy surpasses all of theirs on most sequences. Even though our methods don't design an extra mask network, our performance is still superior to LO-Net (Li et al. 2019). Compared with Xu et al. (Xu et al. 2022), our model performance outperforms theirs without the need for a motion voting mechanism and two-stage estimation. For a fair comparison, we only assess the odometry part of each network excluding mapping optimization.

00-08 as training sequences and 09/10 as testing sequences. We further compare our networks with three learning-based methods, namely ConvLSTM (Zou et al.

	Method	07†		08†		09†		10†		Mean on 07-10	
		t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
Classic	ICP-po2po	5.17	3.35	10.04	4.93	6.93	2.89	8.91	4.74	7.359	3.407
	ICP-po2pl	1.55	1.42	4.42	2.14	3.95	1.71	6.13	2.60	4.735	1.932
	GICP	0.64	0.45	1.58	0.75	1.97	0.77	1.31	0.62	1.921	0.733
	CLS	1.04	0.73	2.14	1.05	1.95	0.92	3.46	1.28	2.148	0.995
	LOAM w/o mapping	10.87	6.76	12.72	5.77	8.10	4.30	12.67	8.79	10.820	5.426
	LOAM with mapping	0.69	0.50	**1.18**	**0.44**	1.20	0.48	1.51	**0.57**	1.278	0.504
	LeGO-LOAM	1.12	0.81	1.99	0.94	1.97	0.98	2.21	0.92	2.49	1.00
	SuMa	1.75	1.17	2.53	0.96	1.92	0.78	1.81	0.97	2.93	0.92
	Vizzo et al.	0.72	0.55	1.44	0.61	1.51	0.66	1.38	0.84	1.55	0.74
DL-based	SfMLearner	21.3	6.65	21.9	2.91	18.8	3.21	14.3	3.30	19.1	4.02
	LO-Net	1.70	0.89	2.12	0.77	1.37	0.58	1.80	0.93	1.330	0.688
	SelfVoxeLO	2.51	1.15	2.65	1.00	2.86	1.17	3.22	1.26	2.81	1.15
	RSLO	2.37	1.15	2.14	0.92	2.61	1.05	2.33	0.94	2.36	1.02
	Ours	**0.55**	**0.43**	1.29	0.50	**0.95**	**0.46**	**1.18**	0.61	**0.993**	**0.500**

Table 1: Comparison with the state-of-the-art. t_{rel}, r_{rel} indicate the average translation RMSE (%) and rotation RMSE (°/100m) respectively on all subsequences in the length of $100, 200, ..., 800m$. '†' means the testing sequences. 'NG' means results are not given. The best result for each sequence is bold, and the second best is underlined.

Method	09		10		Mean	
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
ConvLSTM	3.49	1.00	5.81	1.80	4.650	1.400
Nubert et al.	1.54	0.68	1.78	**0.69**	1.660	0.685
Self-VLO	2.58	1.13	2.67	1.28	2.620	1.210
Ours	**1.01**	**0.47**	**1.41**	0.71	**1.210**	**0.590**

Table 2: The LiDAR odometry results on sequences 09 and 10 of KITTI odometry dataset.

Method	07		08		Mean	
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
LodoNet	1.86	1.64	2.04	0.97	1.950	1.305
Ours	**0.53**	**0.22**	**1.40**	**0.62**	**0.965**	**0.420**

Table 3: The LiDAR odometry results on sequences 07 and 08 of KITTI odometry dataset.

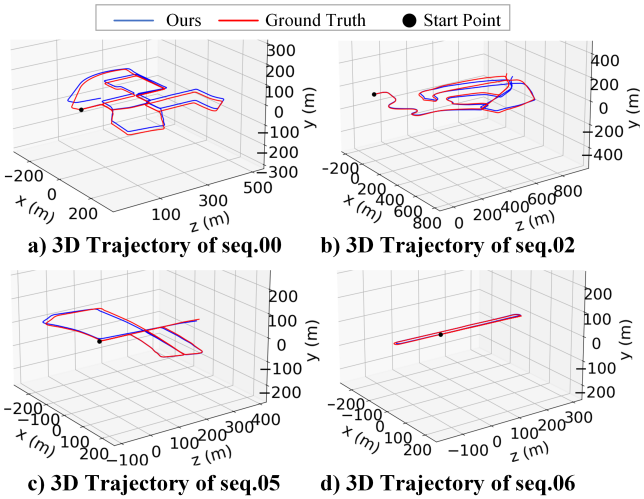


Figure 5: Trajectory of TransLO and ground truth. We visualize four 3D trajectory samples on KITTI dataset.

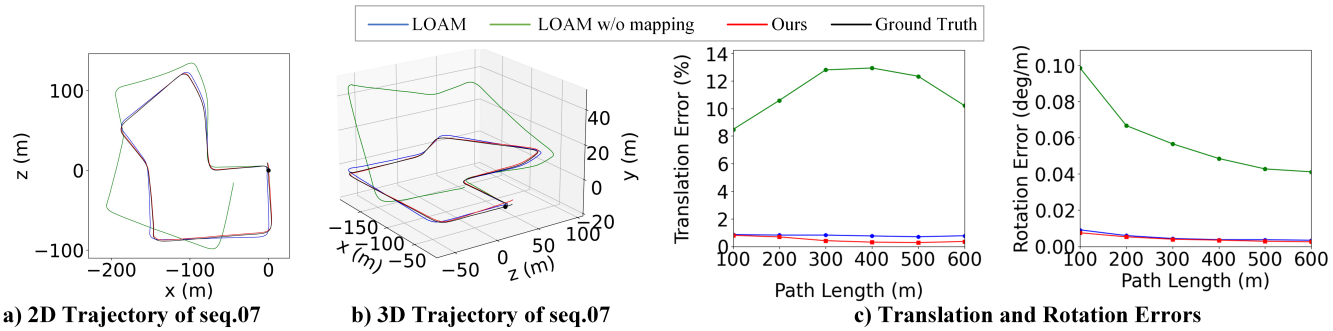
2020), (Nubert, Khattak, and Hutter 2021), and SelfVLO (Li et al. 2021). As illustrated in Table 2, both our translation and rotation errors are smaller than theirs even with mapping. SelfVLO is self-supervised visual-LiDAR odometry with flip consistency. Although our method utilizes no visual information, we still achieve better performance.

00-06/09/10 as training sequences and 07/08 as testing sequences. LodoNet (Zheng et al. 2020) is a network with 2d keypoints matching which also transfers the LiDAR frames to 2D space. As shown in Table 3, our accuracy is superior to LodoNet on both 07 and 08 sequences. Our translation error is almost 1 (%) smaller than theirs. Also, we get 0.885 (°/100m) improvement with respect to rotation error.

Ablation Study

We conduct ablation studies to assess the effectiveness of three main components: projection-aware masks, Window-based Masked Self Attention (WMSA), and Masked Cross Frame Attention (MCFA). We first compare our model with the baseline (Wang et al. 2021b) in Table 4 (a), our average rotation errors are slightly larger (0.01°/100m). However, translation errors are reduced by 10% of theirs. Moreover, our full network runtime has only a linear increase, which indicates the high efficiency of our TransLO.

Importance of WMSA: Compared with CNNs, transformer focuses more on global features and this characteristic is significant to large-scale localization and navigation.



a) 2D Trajectory of seq.07 **b) 3D Trajectory of seq.07** **c) Translation and Rotation Errors**

Figure 6: Estimated trajectory of LOAM and ours on KITTI 07 sequence with ground truth. Our trajectory is even more accurate than LOAM with mapping.

Method	07		08		09		10		Mean		Runtime($/ms$)
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	
Baseline (Wang et al. 2021b) **(a)**	0.60	0.44	1.26	0.55	**0.79**	**0.35**	1.69	0.62	1.085	**0.490**	66.4
Ours w/o mask and MCFA **(b)**	1.12	0.91	1.78	0.81	1.63	0.74	1.83	0.93	1.590	0.848	**57.8**
Ours w/o MCFA **(c)**	0.80	0.54	**1.25**	0.58	1.11	0.54	1.47	0.78	1.158	0.610	58.0
Ours w/o WMSA **(d)**	1.12	0.91	1.78	0.81	1.63	0.74	1.83	0.93	1.590	0.848	71.1
Ours **(e)**	**0.55**	**0.43**	1.29	**0.50**	0.95	0.46	**1.18**	**0.61**	**0.993**	0.500	84.8

Table 4: Ablation study for our proposed model. Notations: The best performance of each sequence is bold.

Overall, the importance of WMSA can be seen when comparing results in Table 4 (d) and (e).

Effect of the binary Mask: As mentioned before, a binary mask is generated simultaneously with the projected feature map. It can filter outliers that are harmful to odometry accuracy. Comparing Table 4 (b) and (c), the network accuracy descends when removing this mask.

Significance of MCFA: This module is used to establish soft correspondences of two frames. By replacing it with the cost-volume-only methods in our baseline, both translation and rotation errors are larger as in Table 4 (c).

Visualization

We visualize the trajectory of our network and the benefits of transformer-based methods in this section.

Trajectory Visualization. We further visualize 3D trajectory based on our estimated pose in Fig. 5. The figure shows our odometry can track the trajectory of the ground truth fairly well. Also, we conduct experiments to compare trajectory and estimation errors between LOAM and ours in Fig. 6. Although we do not have the mapping procedure, our odometry accuracy surpasses LOAM with mapping optimization as demonstrated in Fig. 6.

Why Transformer? Attention weights are also visualized as illustrated in Fig. 7. Compared with CNNs, our transformer can capture global learning of surroundings. For example, trees are allocated similar attention weights even with great distances. This characteristic is significant for large-scale 3D scene understanding as global features can be gathered without the distance constraint in CNN methods.

a) Similar patches. **b) Attention weights.**

Figure 7: Visualization of attention weights. Trees both near and far are allocated similar attention weights.

Conclusion

In this paper, we propose a novel end-to-end window-based masked point transformer network for large-scale LiDAR odometry. We combine CNNs and transformers to extract more global feature embeddings, which facilitate the outliers' rejection. We evaluate our framework on the KITTI odometry dataset. Experiment results show that our method achieves state-of-the-art performance.

Acknowledgements

This work was supported in part by the Natural Science Foundation of China under Grant 62073222, Grant U21A20480, and Grant U1913204; in part by the Science and Technology Commission of Shanghai Municipality under Grant 21511101900; and in part by the Open Research Projects of Zhejiang Laboratory under Grant 2022NB0AB01. The authors gratefully appreciate the contribution of Xinrui Wu from Shanghai Jiao Tong University.

References

Behley, J.; and Stachniss, C. 2018. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Robotics: Science and Systems*, volume 2018, 59.

Besl, P. J.; and McKay, N. D. 1992. Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, 586–606. Spie.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

Censi, A. 2008. An ICP variant using a point-to-line metric. In *2008 IEEE International Conference on Robotics and Automation*, 19–25. Ieee.

Cho, Y.; Kim, G.; and Kim, A. 2019. DeepLO: Geometry-Aware Deep LiDAR Odometry. arXiv:1902.10562.

Cho, Y.; Kim, G.; and Kim, A. 2020. Unsupervised geometry-aware deep lidar odometry. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2145–2152. IEEE.

Dong, X.; Bao, J.; Chen, D.; Zhang, W.; Yu, N.; Yuan, L.; Chen, D.; and Guo, B. 2022. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12124–12134.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Fischer, K.; Simon, M.; Olsner, F.; Milz, S.; Gross, H.-M.; and Mader, P. 2021. Stickypillars: Robust and efficient feature matching on point clouds using graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 313–323.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11): 1231–1237.

Guo, M.-H.; Cai, J.-X.; Liu, Z.-N.; Mu, T.-J.; Martin, R. R.; and Hu, S.-M. 2021. Pct: Point cloud transformer. *Computational Visual Media*, 7(2): 187–199.

Kuettel, D.; and Ferrari, V. 2012. Figure-ground segmentation by transferring window masks. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 558–565. IEEE.

Li, B.; Hu, M.; Wang, S.; Wang, L.; and Gong, X. 2021. Self-supervised visual-LiDAR odometry with flip consistency. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3844–3852.

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

Li, S.; Wang, X.; Cao, Y.; Xue, F.; Yan, Z.; and Zha, H. 2020. Self-supervised deep visual odometry with online adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6339–6348.

Liu, D.; Shetty, S.; Hinz, T.; Fisher, M.; Zhang, R.; Park, T.; and Kalogerakis, E. 2022. ASSET: autoregressive semantic scene editing with transformers at high resolutions. *ACM Transactions on Graphics (TOG)*, 41(4): 1–12.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10012–10022.

Low, K.-L. 2004. Linear least-squares optimization for point-to-plane icp surface registration. *Chapel Hill, University of North Carolina*, 4(10): 1–3.

Nicolai, A.; Skeele, R.; Eriksen, C.; and Hollinger, G. A. 2016. Deep learning for laser based odometry estimation. In *RSS workshop Limits and Potentials of Deep Learning in Robotics*, volume 184, 1.

Nubert, J.; Khattak, S.; and Hutter, M. 2021. Self-supervised learning of lidar odometry for robotic applications. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 9601–9607. IEEE.

Pan, X.; Xia, Z.; Song, S.; Li, L. E.; and Huang, G. 2021. 3d object detection with pointformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7463–7472.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ren, P.; Li, C.; Wang, G.; Xiao, Y.; Du, Q.; Liang, X.; and Chang, X. 2022a. Beyond Fixation: Dynamic Window Visual Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11987–11997.

Ren, S.; Zhou, D.; He, S.; Feng, J.; and Wang, X. 2022b. Shunted Self-Attention via Multi-Scale Token Aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10853–10862.

Segal, A.; Haehnel, D.; and Thrun, S. 2009. Generalized-icp. In *Robotics: science and systems*, volume 2, 435. Seattle, WA.

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

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017.

Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Velas, M.; Spanel, M.; and Herout, A. 2016. Collar line segments for fast odometry estimation from velodyne point clouds. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 4486–4495. IEEE.

Vizzo, I.; Chen, X.; Chebrolu, N.; Behley, J.; and Stachniss, C. 2021. Poisson surface reconstruction for LiDAR odometry and mapping. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 5624–5630. IEEE.

Wang, G.; Hu, Y.; Wu, X.; and Wang, H. 2022a. Residual 3-D Scene Flow Learning With Context-Aware Feature Extraction. *IEEE Transactions on Instrumentation and Measurement*, 71: 1–9.

Wang, G.; Jiang, C.; Shen, Z.; Miao, Y.; and Wang, H. 2022b. SFGAN: Unsupervised Generative Adversarial Learning of 3D Scene Flow from the 3D Scene Self. *Advanced Intelligent Systems*, 4(4): 2100197.

Wang, G.; Ren, S.; and Wang, H. 2022. Unsupervised learning of optical flow with non-occlusion from geometry. *IEEE Transactions on Intelligent Transportation Systems*.

Wang, G.; Tian, X.; Ding, R.; and Wang, H. 2021a. Unsupervised Learning of 3D Scene Flow from Monocular Camera. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 4325–4331. IEEE.

Wang, G.; Wu, X.; Jiang, S.; Liu, Z.; and Wang, H. 2022c. Efficient 3d deep lidar odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, G.; Wu, X.; Liu, Z.; and Wang, H. 2021b. Pwclonet: Deep lidar odometry in 3d point clouds using hierarchical embedding mask optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 15910–15919.

Wang, G.; Zhang, C.; Wang, H.; Wang, J.; Wang, Y.; and Wang, X. 2020. Unsupervised learning of depth, optical flow and pose with occlusion from 3d geometry. *IEEE Transactions on Intelligent Transportation Systems*, 23(1): 308–320.

Wang, H.; Zheng, D.; Wang, J.; Chen, W.; and Yuan, J. 2018. Ego-motion estimation of a quadrotor based on nonlinear observer. *IEEE/ASME Transactions on Mechatronics*, 23(3): 1138–1147.

Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J. M.; and Luo, P. 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34: 12077–12090.

Xu, Y.; Huang, Z.; Lin, K.-Y.; Zhu, X.; Shi, J.; Bao, H.; Zhang, G.; and Li, H. 2020. SelfVoxeLO: Self-supervised LiDAR Odometry with Voxel-based Deep Neural Networks. arXiv:2010.09343.

Xu, Y.; Lin, J.; Shi, J.; Zhang, G.; Wang, X.; and Li, H. 2022. Robust self-supervised lidar odometry via representative structure discovery and 3d inherent error modeling. *IEEE Robotics and Automation Letters*, 7(2): 1651–1658.

Yang, N.; Stumberg, L. v.; Wang, R.; and Cremers, D. 2020. 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*, 1281–1292.

Yin, J.; Shen, J.; Gao, X.; Crandall, D.; and Yang, R. 2021. Graph neural network and spatiotemporal transformer attention for 3D video object detection from point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, J.; and Singh, S. 2014. LOAM: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2, 1–9. Berkeley, CA.

Zhang, J.; and Singh, S. 2017. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, 41(2): 401–416.

Zhang, W.; Huang, Z.; Luo, G.; Chen, T.; Wang, X.; Liu, W.; Yu, G.; and Shen, C. 2022. TopFormer: Token Pyramid Transformer for Mobile Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12083–12093.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16259–16268.

Zheng, C.; Lyu, Y.; Li, M.; and Zhang, Z. 2020. Lodonet: A deep neural network with 2d keypoint matching for 3d lidar odometry estimation. In *Proceedings of the 28th ACM International Conference on Multimedia*, 2391–2399.

Zheng, X.; and Zhu, J. 2021. Efficient LiDAR odometry for autonomous driving. *IEEE Robotics and Automation Letters*, 6(4): 8458–8465.

Zhou, L.; Huang, G.; Mao, Y.; Yu, J.; Wang, S.; and Kaess, M. 2022. PLC-LiSLAM: LiDAR SLAM with Planes, Lines and Cylinders. *IEEE Robotics and Automation Letters*.

Zhou, T.; Brown, M.; Snavely, N.; and Lowe, D. G. 2017. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1851–1858.

Zhu, Z.; Peng, S.; Larsson, V.; Xu, W.; Bao, H.; Cui, Z.; Oswald, M. R.; and Pollefeys, M. 2022. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12786–12796.

Zou, Y.; Ji, P.; Tran, Q.-H.; Huang, J.-B.; and Chandraker, M. 2020. Learning monocular visual odometry via self-supervised long-term modeling. In *European Conference on Computer Vision*, 710–727. Springer.