NeuralPCI: Spatio-temporal Neural Field for 3D Point Cloud Multi-frame Non-linear Interpolation Supplementary Material

Zehan Zheng*, Danni Wu*, Ruisi Lu, Fan Lu, Guang Chen†, Changjun Jiang
Tongji University
{zhengzehan, woodannie, lrs910, lufan, guangchen, cjjiang}@tongji.edu.cn

A. Overview

In this document, we present more details and several extra results as well as visualization. In Appendix B, we introduce details of the datasets used in our work. Then we elaborate on the implementation details of our NeuralPCI and other baselines in Appendix C. And in Appendix D, we provide extra results in multiple aspects, such as the convergence, different numbers of input frames, explicit versus implicit frame interpolation, varying point cloud densities and ground point removal. Finally, we show more qualitative results in Appendix E.

B. Dataset Details

In this section, we introduce DHB dataset and the open-source autonomous driving datasets based Non-Linear Drive (NL-Drive) dataset. The train/val/test split of datasets facilitates the comparison of benchmarks. NeuralPCI optimizes at run-time, so it doesn't need the training data. We perform NeuralPCI as well as NSFP [4] on the test set directly to obtain the evaluation results. Other learning-based methods are pre-trained on the training set first and then compared on the same test set.

B.1. DHB Dataset

DHB dataset [7] consists of 14 point cloud sequences indicating dynamic human bodies, in which each point cloud frame is sampled to 1024 points. To align with baseline method [7], we adopt six sequence with 1,600 frames, (i.e., *Longdress, Loot, Redandblack, Soldier, Squat_2, Swing*) as the test dataset, and the remaining eight sequences with 1,600 frames as the train dataset.

B.2. NL-Drive Dataset

We construct NL-Drive dataset based on three public autonomous driving datasets, namely KITTI odometry dataset [3], Argoverse 2 sensor dataset [2], and Nuscenes dataset [1]. KITTI odometry dataset contains 22 LiDAR point cloud sequences in total, 11 sequences with ground truth (00∼10), and we use 00∼06 for training, 07∼08 for validation, and the others for test. Argoverse 2 sensor dataset is composed of 1,000 scenarios with 150 LiDAR sweeps per scenario on average, while Nuscenes dataset consists of 1,000 driving scenes with about 400 LiDAR frames for each scene. For both datasets, we utilize the top 700 scenes to train, 701∼850 scenes to validate and the remaining 150 scenes to test. Thus, we define the data source of the NL-Drive dataset as the mentioned splited datasets based on the training, validation and test ratio of 14:3:3. For Nuscenes dataset, we first downsample point clouds from 20Hz to 10Hz in order to acquire larger motion between input frames and align with the other two datasets. We select the point clouds at a given interval of frames from the 10Hz point cloud as input, and the remaining point clouds as the ground truth of interpolation. Particularly, the frequency of

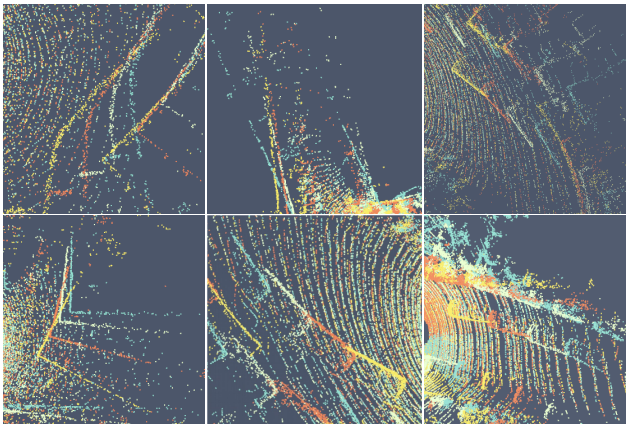


Figure S1. **Visualization (zoom-in view) for the point clouds of four consecutive input frames with equal time interval in the NL-Drive dataset.** The 1st to 4th frames sorted by chronological order are colored in blue, white, red and yellow, respectively. The motion of surrounding objects is nonlinear.

* Equal contribution. † Corresponding author.

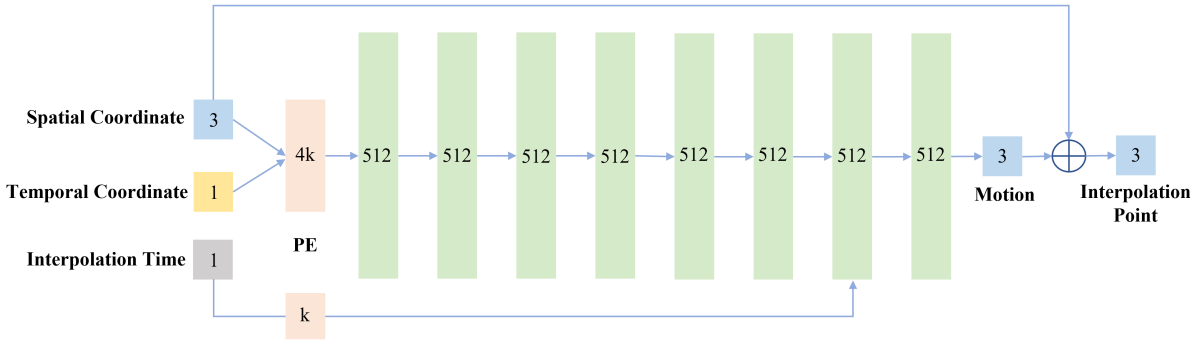


Figure S2. **Network architecture of our proposed NeuralPCI.**

input point clouds is 2.5Hz when there are three interpolation frames to predict between the middle two input frames.

Our NL-Drive dataset is intended to focus on large movements in as many autonomous driving scenarios as possible. Thus, we try to sort out hard samples that possess the largest relative pose transformation between frames while ensuring it is above the selection threshold from all scenes of the data source. These samples tend to contain nonlinear motions under the precondition of ego-vehicle large motions. The details for constructing NL-Drive dataset are as follows. We take the standard case in the main paper as an example, i.e., a sample contains 4 frames as multi-frame input and 3 frames between the middle two input frames to interpolate. First, we calculate the 6-DOF relative pose transformation between each two input frames. Then, we transform the relative ego-vehicle pose to the LiDAR sensor coordinate system, in which the rotation is uniformly expressed as the Euler angle and the translation as the translation vector, indicating the rotational angular velocity and translational velocity of the ego-vehicle between two frames to some extent. In this form, we can intuitively infer and compare the magnitude of movements from the value. Next, considering that the pitch and roll rotation in autonomous driving datasets is much slighter than the yaw rotation, we define the metric for the rotational motion as the yaw angle. The metric for translational motion is defined as the root-mean-square of the translation vector. We select out the top-k samples with the largest rotational or translational values from each scene and filter them with the threshold (5.0° for yaw, $2.5m$ for translation) that is utilized to balance the greater coverage of scenes and the need for large movements. Finally, we exhibit the local zoom-in view of typical samples from NL-Drive dataset in Fig. S1, from which the nonlinear motion of objects can be evidently observed.

C. Method Details

In this section, we elaborate on the network architecture of NeuralPCI and more details of other baselines.

Table S1. **Results with different layer widths for NeuralPCI.**

layer width	DHB ($\times 10^{-3}$)		NL-Drive	
	CD↓	EMD↓	CD↓	EMD↓
128	0.563	3.931	0.828	105.230
256	0.547	3.766	**0.768**	102.173
512	**0.541**	3.677	0.801	**97.029**
1024	0.542	**3.651**	0.770	97.099

Table S2. **Results with different layer depths for NeuralPCI.**

layer depth	DHB ($\times 10^{-3}$)		NL-Drive	
	CD↓	EMD↓	CD↓	EMD↓
4	0.543	3.755	0.810	107.301
6	**0.536**	3.693	0.759	102.150
8	0.541	**3.677**	0.801	**97.029**
10	0.546	3.737	**0.750**	97.213

C.1. Network Architecture of NeuralPCI

As shown in the Fig. S2, NeuralPCI consists of an 8-layer 512-unit MLP, using the LeakyReLU activation function for each layer *except* the last one. Taking one point of the input point cloud as an example, its 3-dimensional spatial coordinate and 1-dimensional temporal coordinate are concatenated and positional encoded (PE) to obtain a $4k$-dimensional input fed into the MLP. The positional encoding function $\Gamma(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}^{nk}$ is shown below:

$$\Gamma(\mathbf{x}) = (\mathbf{x}, \sin(\mathbf{x}), \cos(\mathbf{x}), \ldots, \sin(2^m\mathbf{x}), \cos(2^m\mathbf{x})) \quad (1)$$

where we set m to 0 for convenience, thus k is equal to 3.

Besides, 1-dimensional interpolation time is concatenated with the 512-dimensional features of the penultimate layer by a skip connection, which is used to regulate the final 3-dimensional motion output. This output is added element-wise to the spatial coordinate of the input point cloud to obtain the final interpolation point. All points in the point cloud are computed in parallel by MLP with *shared weights*, accumulating the gradients and back-propagating to update the parameters of NeuralPCI.

To further investigate the influence of other network structure hyperparameters on the experimental results, we explore the MLP with different layer widths and depths. The results are presented in Tables S1 and S2, which indicates that the hyperparameters of network structure have a relatively minor impact on the interpolation results. Taking both accuracy and efficiency into account, we choose the structure with the parametric number of 1.847M as depicted in Fig. S2.

C.2. Optimization Details

The computational cost of EMD increases with the number of point clouds, and down-sampling with EMD loss leads to worse results. Therefore, we only use it on DHB dataset (1024 points). In contrast, we utilize smoothness loss on NL-Drive dataset, which adopts k-nearest neighbor (k = 9) to further regulate local rigid motion for autonomous driving scenarios. Empirically, we set the weights α, β, γ in the total loss to 1, 50, 0 for DHB Dataset and 1, 0, 1 for NL-Drive Dataset during optimization.

C.3. Other Baselines

We adopt the linear interpolation results of outstanding scene flow methods NSFP [4] and PV-RAFT [6] as baselines, with the consideration of explicit interpolation methods. We calculate both forward scene flow $\boldsymbol{f}_{0\to1}$ and backward scene flow $\boldsymbol{f}_{1\to0}$ from the pair-frame inputs, and interpolate linearly to acquire the scene flow between the reference frames and the interpolation frame, which is used to warp the input frame. Then, as described in Eqs. (2) and (3), we can obtain the intermediate frame based on the forward or backward scene flow.

$$\hat{P}_{fwd} = P_0 + t \times \boldsymbol{f}_{0\to1} \tag{2}$$

$$\hat{P}_{bwd} = P_1 + (1-t) \times \boldsymbol{f}_{1\to0} \tag{3}$$

where $t \in (0, 1)$ means the time step of the intermediate frame, and P_0 and P_1 indicate the reference frames before and after the in-between frame.

D. Supplementary Experiments

In this section, we conduct further supplementary experiments to verify the effectiveness of our method, covering aspects of the convergence, different numbers of input frames, explicit versus implicit frame interpolation, varying point cloud densities and ground point removal.

D.1. Convergence and Efficiency

Since NeuralPCI is optimized at runtime, there is a trade-off between its accuracy and time consumption. We plot the convergence curve of NeuralPCI on DHB dataset in Fig. S3,

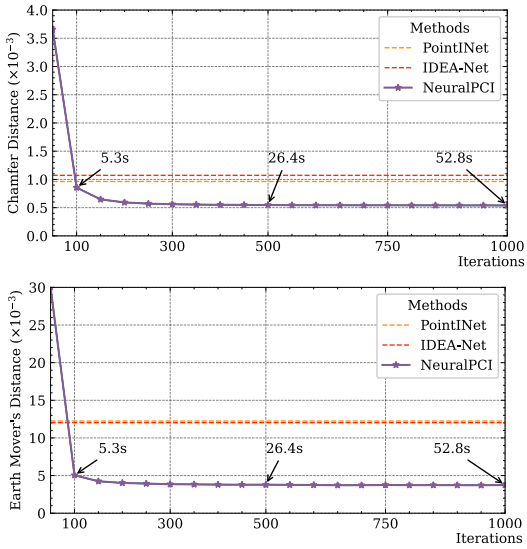


Figure S3. **Convergence curve of NeuralPCI.** As references, the performance of PointINet [5] and IDEA-Net [7] is denoted by dashed lines. Compared to them, NeuralPCI only needs less than 100 iterations to achieve better CD and EMD results.

in which each data point represents the average result of the overall dataset after corresponding iterations. It is evident that NeuralPCI has an excellent convergence, as the error decreases significantly in the first 100 iterations and exceeds previous SOTA methods. When the number of iterations grows, the error gradually reduces, and the convergence is almost complete after 500 iterations. And the entire optimization of 1000 iterations is finished in less than one minute. Therefore, the number of iterations can be determined according to the specific usage scenario. In the case of high timeliness requirements, satisfactory results can be obtained within only 5 seconds, while in other off-board applications, the number of iterations can be appropriately raised to further improve the accuracy.

D.2. Pair-frame or Multi-frame

In order to eliminate the unfairness of comparing with the existing pair-frame point cloud interpolation methods, a more comprehensive experiment is conducted. Firstly, using PointINet [5] as an example, we input every two frames of all four input point cloud frames to obtain intermediate frame prediction and fuse the interpolation results by random sampling fusion or nearest neighbor (NN) fusion. Random sampling selects points with equal probability from each predicted point cloud result. And NN fusion means to find the nearest point from another point cloud and average the spatial coordinates. Based on this, the pair-frame point cloud interpolation method can also fully utilize the information of all four input point clouds. From the results in Table S3, it can be seen that since the final predicted

Table S3. **Quantitative results for PointINet [5] with different pair-frame inputs and multi-frame fusion methods.** We denote the input frames as frames 1∼4. Different pair frames are used separately as input to predict the same interpolation frame, and the predicted results A∼D are finally fused.

Type		Input Frames	DHB ($\times 10^{-3}$)		NL-Drive	
			CD↓	EMD↓	CD↓	EMD↓
pair-frame	A	frame 2, 3	**0.97**	**12.23**	**1.06**	**101.12**
	B	frame 1, 3	1.33	12.81	1.87	125.90
	C	frame 2, 4	1.33	12.93	1.72	129.30
	D	frame 1, 4	3.49	21.20	4.72	227.91
multi-frame	random fusion	B+C	1.33	15.64	1.52	112.07
		A+B+C	1.19	15.85	1.25	105.19
		A+B+C+D	1.46	17.47	1.62	118.44
	NN fusion	B+C	1.25	12.95	1.66	127.68
		A+B+C	1.05	12.34	1.17	105.00
		A+B+C+D	1.17	12.54	1.29	108.91

point cloud is located between the second and third input frames, the interpolation results using these two frames as input have the highest accuracy, while those using the first and third frames and using the second and fourth frames have the suboptimal accuracy, and those using the first and fourth frames have the worst accuracy. Secondly, we fuse the aforementioned two or more prediction results by random sampling fusion or nearest neighbor fusion. However, it can be noted that simple point cloud fusion is difficult to achieve higher accuracy.

Finally, we evaluate the results of all methods using both two-frame input and multi-frame input on DHB dataset as well as NL-Drive dataset. As shown in Table S4, simply migrating the existing pair-frame interpolation method to multi-frame input does not give better results, while our method can still achieve decent interpolation accuracy even when only using the middle two frames as input. Most importantly, NeuralPCI provides a better way to integrate the spatio-temporal information of multi-frame input point clouds, achieving 10.0% CD reduction and 12.4% EMD reduction on DHB dataset and 4.8% CD reduction and 2.0% EMD reduction on NL-Drive dataset compared to pair-frame input, respectively.

D.3. More Input Frames

Beyond the standard four-frame input, our proposed NeuralPCI can also be flexibly extended to more point cloud input frames. Thus, as shown in Table S5, we present the results of NeuralPCI on two datasets with more frames of point clouds as input. Nevertheless, since the predicted interpolation point cloud is always between the two point clouds directly in the middle, the multi-frame inputs that are too far away from it are dramatically less relevant in terms of motion and do not contribute better information to assist interpolation. Besides, the limited modeling capability of MLP causes performance degradation. As a result, the four-frame input is the most appropriate.

Table S4. **Quantitative results for pair-frame or multi-frame point clouds as input.** The *4-frame* results are based on the NN-fusion of A+B+C inputs described in Table S3.

Input	Methods	DHB ($\times 10^{-3}$)		NL-Drive	
		CD↓	EMD↓	CD↓	EMD↓
2-frame	NSFP [4]	1.22	7.81	1.75	132.13
	PV-RAFT [6]	0.92	6.14	1.64	140.42
	PointINet [5]	0.96	12.25	1.06	101.12
	IDEA-Net [7]	1.02	12.03	-	-
	Neural PCI	0.60	4.20	0.84	98.99
4-frame	NSFP [4]	1.58	8.25	2.30	149.03
	PV-RAFT [6]	1.10	6.63	1.64	144.56
	PointINet [5]	1.05	12.34	1.17	105.00
	IDEA-Net [7]	1.07	12.17	-	-
	Neural PCI	**0.54**	**3.68**	**0.80**	**97.03**

Table S5. **Quantitative results with different numbers of input frames for NeuralPCI.** Among them, *2 frames* input indicates the pair frame setting and *4 frames* input is the standard setting in our main paper.

Input Frames	DHB ($\times 10^{-3}$)		NL-Drive	
	CD↓	EMD↓	CD↓	EMD↓
2 frames	0.60	4.20	0.84	98.99
4 frames	**0.54**	**3.68**	**0.80**	**97.03**
6 frames	0.55	3.74	0.86	98.82
8 frames	0.57	3.87	0.96	104.44

Table S6. **Quantitative results for explicit and implicit interpolation.** *Ex* indicates explicit interpolation, and *Im* indicates implicit interpolation.

Methods	Type	DHB ($\times 10^{-3}$)		NL-Drive	
		CD↓	EMD↓	CD↓	EMD↓
PointINet [5]	linear	0.96	12.25	1.06	101.12
Ours (Ex)	linear	0.57	3.99	0.80	112.90
	quadratic	0.56	3.81	0.84	112.83
	cubic	0.60	3.90	0.89	113.24
Ours (Im)	neural field	**0.54**	**3.68**	**0.80**	**97.03**

D.4. Explicit or Implicit Interpolation

NeuralPCI establishes the motion relationship through an implicit 4D spatio-temporal neural field and derives the corresponding output by varying the interpolation time input. On the contrary, we can also use an explicit approach to model the equations of nonlinear motion and generate the interpolation point clouds at intermediate moments. That is, one of the input point clouds is fed into NeuralPCI as the reference to predict the other three input point clouds, and

Table S7. **Quantitative results after removal of ground points on NL-Drive dataset.**

Methods	Frame - 1		Frame - 2		Frame - 3		Average	
	CD	EMD	CD	EMD	CD	EMD	CD↓	EMD↓
PV-RAFT [6]	1.90	150.55	2.87	217.33	2.27	253.80	2.34	207.23
NSFP [4]	1.24	137.03	2.26	198.57	3.37	256.44	2.29	197.35
PointINet [5]	1.28	138.29	1.72	**154.32**	1.35	133.32	1.45	141.98
NeuralPCI	**0.92**	**127.31**	**1.16**	167.34	**0.91**	**125.99**	**1.00**	**140.21**

thus the order of the corresponding points in all the obtained point clouds is aligned with the reference frame. Finally, according to the spatial position of each point at four moments, we employ linear (Eq. (7)), quadratic (Eq. (8)) and cubic (Eq. (9)) equations to describe its motion and calculate the position at the intermediate moment. The equations are as follows:

$$\boldsymbol{v}_0 = P_1 - \hat{P}_0, \quad \boldsymbol{v}_1 = \hat{P}_2 - P_1, \quad \boldsymbol{v}_2 = \hat{P}_3 - \hat{P}_2 \tag{4}$$

$$\boldsymbol{a}_0 = \boldsymbol{v}_1 - \boldsymbol{v}_0, \quad \boldsymbol{a}_1 = \mathbf{v}_2 - \boldsymbol{v}_1 \tag{5}$$

$$\boldsymbol{b} = \boldsymbol{a}_1 - \boldsymbol{a}_0 \tag{6}$$

$$\hat{P}_t = P_1 + \frac{(\boldsymbol{v}_0 + \boldsymbol{v}_1)}{2} t \tag{7}$$

$$\hat{P}_t = P_1 + \frac{(\boldsymbol{v}_0 + \boldsymbol{v}_1)}{2} t + \frac{\boldsymbol{a}_0}{2} t^2 \tag{8}$$

$$\hat{P}_t = P_1 + \frac{(\boldsymbol{v}_0 + \boldsymbol{v}_1 + \boldsymbol{v}_2)}{3} t + \frac{(\boldsymbol{a}_0 + \boldsymbol{a}_1)}{4} t^2 + \frac{\boldsymbol{b}}{6} t^3 \tag{9}$$

where $\hat{P}_0, \hat{P}_2, \hat{P}_3$ are the predicted point clouds of NeuralPCI based on the reference point cloud P_1, and the points in these four frames located at different time steps posses one-to-one correspondences. Let $t \in (0, 1)$, we calculate the interpolation point cloud $\hat{P}_t$ between P_1 and $\hat{P}_2$.

As shown in the Table S6, explicit modeling of motion can also yield nice interpolation results, but one single equation can not cover well all the complex motions of the real world, whereas the implicit output of NeuralPCI benefits from the higher order fitting properties of MLP and enables more accurate nonlinear motion estimation for each sample.

D.5. Point Cloud Density

In the comparison experiments of the main paper, each sample of DHB dataset has 1024 points, and the input point cloud of the NL-Drive dataset is sampled uniformly to 8192 points. To further validate the performance of each method under point clouds with different densities, we design a series of experiments with input point clouds of the point number gradients, i.e., 1024, 2048, 4096, 8192, and 16384 points, on NL-Drive dataset for a fair comparison. The results are shown in Fig. S4, and our proposed NeuralPCI is robust for both sparse and dense point clouds and achieves the state-of-the-art performance.

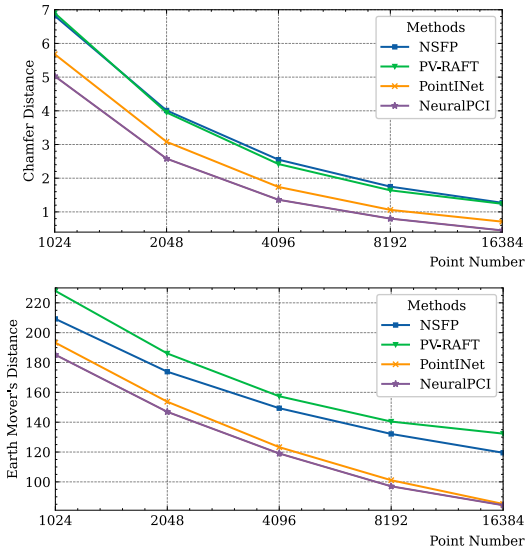


Figure S4. **Different densities of input point cloud on NL-Drive dataset.** NeuralPCI works and achieves the optimum results with point clouds of diverse densities.

D.6. Removal of Ground Points

Ground points cover a large portion of the point clouds in outdoor autonomous driving scenarios, which remain relatively stationary with respect to the ego vehicle and contain little particular movement information. While it makes sense to recover ground points in the point cloud interpolation, the presence of these static points also affects the demonstration of the advantages of our method for nonlinear motion estimation. Therefore, we remove the ground points from the NL-Drive dataset and conduct the same comparison experiments as shown in the Table S7. The final conclusion remains consistent with the main paper, which shows our method also outperforms previous SOTA methods on dynamic objects after eliminating the influence of static ground points.

E. Qualitative Results

We provide more qualitative results of our method and baseline methods on DHB and NL-Drive datasets as Figs. S5 and S6. It can be seen that the intermediate point

cloud frame predicted by NeuralPCI is closest to the ground truth among all the methods on both datasets. On DHB dataset, both the PointINet [5] and IDEA-Net [7] show diverse degrees of outliers (*e.g*. blurry legs and bent guns), especially at the edge of the object, where the motion tends to be larger than the center. On NL-Drive dataset, it is hard for PointINet to predict shape-discernible points, *e.g*. cars in the surroundings, while our method produces fewer distortions and artifacts.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1

[2] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019. 1

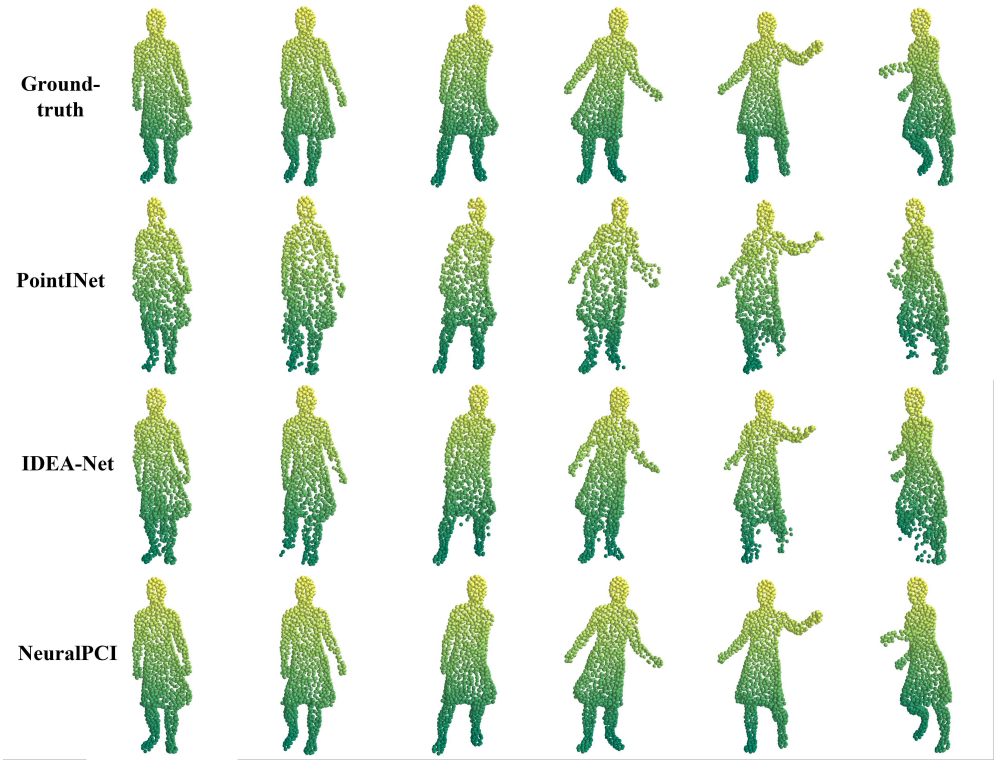
[3] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 1

[4] Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural scene flow prior. *Advances in Neural Information Processing Systems*, 34:7838–7851, 2021. 1, 3, 4, 5

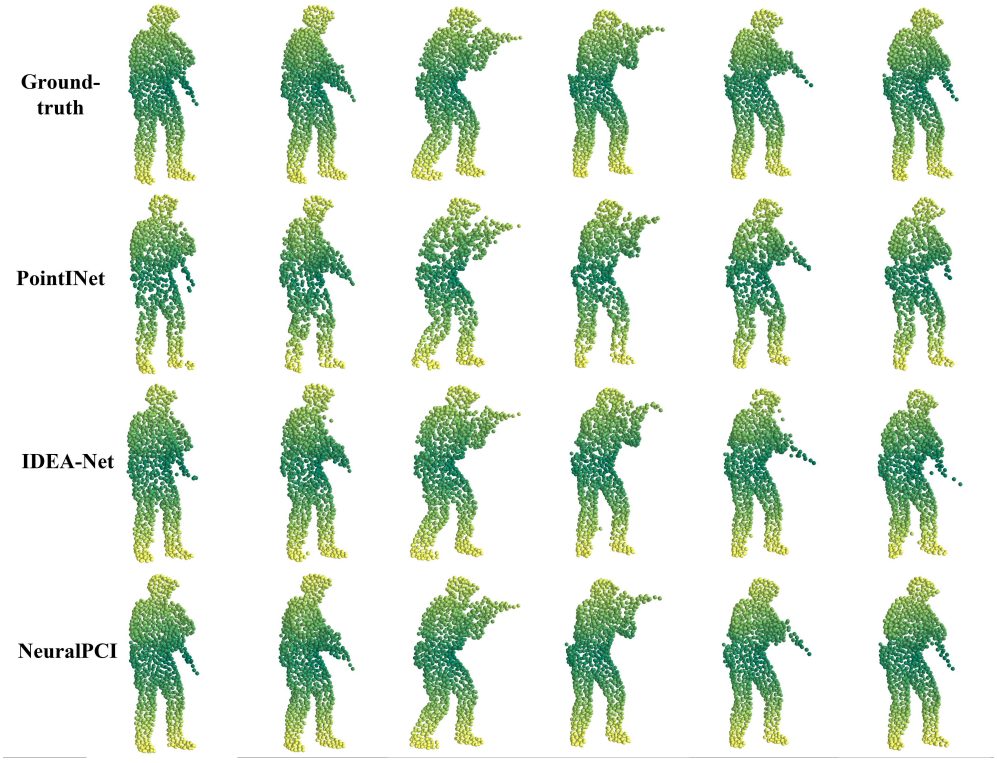
[5] Fan Lu, Guang Chen, Sanqing Qu, Zhijun Li, Yinlong Liu, and Alois Knoll. Pointinet: Point cloud frame interpolation network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2251–2259, 2021. 3, 4, 5, 6

[6] Yi Wei, Ziyi Wang, Yongming Rao, Jiwen Lu, and Jie Zhou. Pv-raft: point-voxel correlation fields for scene flow estimation of point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6954–6963, 2021. 3, 4, 5

[7] Yiming Zeng, Yue Qian, Qijian Zhang, Junhui Hou, Yixuan Yuan, and Ying He. Idea-net: Dynamic 3d point cloud interpolation via deep embedding alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6338–6347, 2022. 1, 3, 4, 6

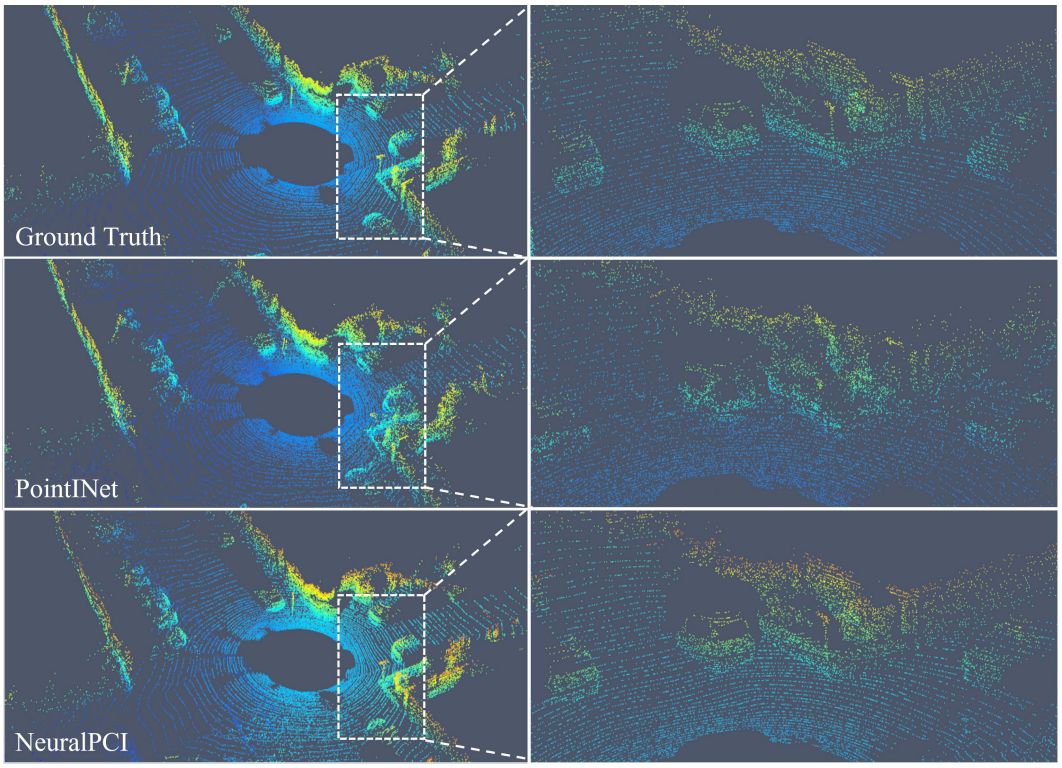


(a)

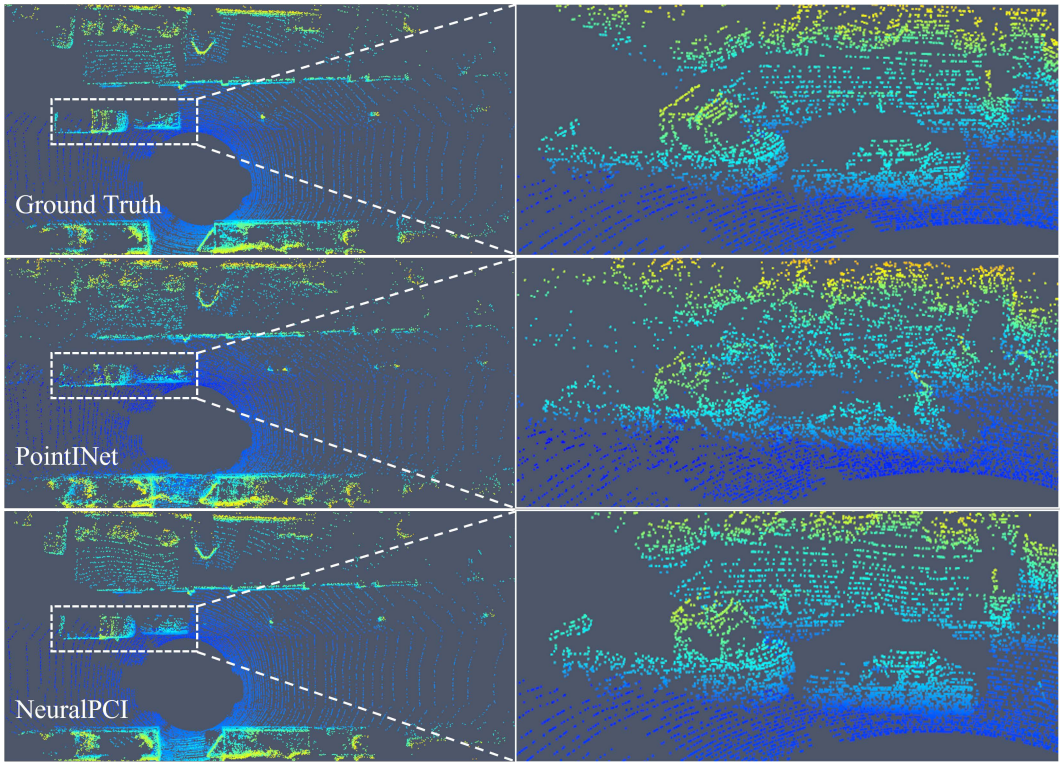


(b)

Figure S5. **Qualitative comparison on the test sequence (a) *Swing* and (b) *Soldier* of DHB dataset.**



(a)



(b)

Figure S6. **Qualitative comparison on NL-Drive dataset.**