

PointINet: Point Cloud Frame Interpolation Network

Fan Lu¹, Guang Chen^{1*}, Sanqing Qu¹, Zhijun Li², Yinlong Liu³, Alois Knoll³

¹ Tongji University, ² University of Science and Technology of China, ³ Technische Universität München
{lufan, guangchen, 2011444}@tongji.edu.cn, zjli@ieee.org, Yinlong.Liu@tum.de, knoll@in.tum.de

Abstract

LiDAR point cloud streams are usually sparse in time dimension, which is limited by hardware performance. Generally, the frame rates of mechanical LiDAR sensors are 10 to 20 Hz, which is much lower than other commonly used sensors like cameras. To overcome the temporal limitations of LiDAR sensors, a novel task named Point Cloud Frame Interpolation is studied in this paper. Given two consecutive point cloud frames, Point Cloud Frame Interpolation aims to generate intermediate frame(s) between them. To achieve that, we propose a novel framework, namely Point Cloud Frame Interpolation Network (PointINet). Based on the proposed method, the low frame rate point cloud streams can be upsampled to higher frame rates. We start by estimating bi-directional 3D scene flow between the two point clouds and then warp them to the given time step based on the 3D scene flow. To fuse the two warped frames and generate intermediate point cloud(s), we propose a novel learning-based points fusion module, which simultaneously takes two warped point clouds into consideration. We design both quantitative and qualitative experiments to evaluate the performance of the point cloud frame interpolation method and extensive experiments on two large scale outdoor LiDAR datasets demonstrate the effectiveness of the proposed PointINet. Our code is available at <https://github.com/ispc-lab/PointINet.git>.

Introduction

LiDAR is one of the most important sensors in numerous applications (e.g., autonomous vehicles and intelligent robots). However, the frame rates of typical mechanical LiDAR sensors (e.g., Velodyne HDL-64E, Hesai Pandar64, etc.) are greatly limited by hardware performance. Frame rates of LiDAR are generally 10 to 20 Hz, which contributes to temporal and spatial discontinuity of point cloud streams. Compared with the low frame rate of LiDAR, the frame rates of other commonly used sensors on intelligent vehicles and robots are typically much higher. For example, the frame rate of cameras and Inertial Measurement Unit (IMU) can achieve over 100 Hz. The large difference in frame rate can cause difficulty to synchronize LiDAR with other sensors. Upsampling low frame rate LiDAR point cloud streams to higher frame rates can be an efficient solution to that

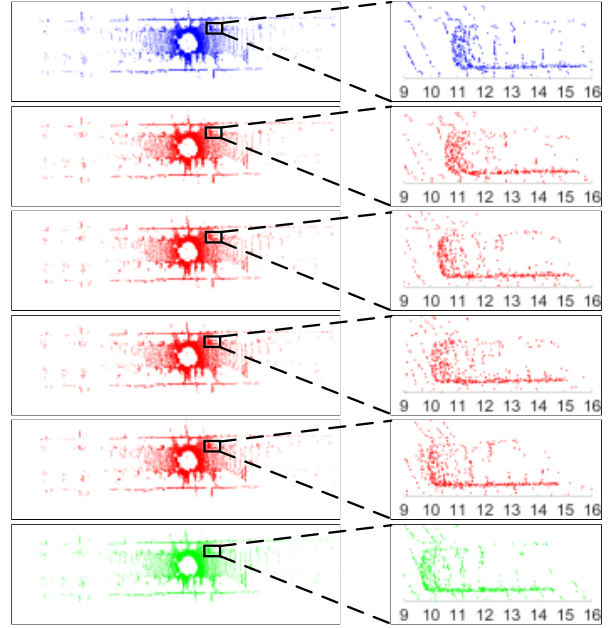


Figure 1: Illustration of Point Cloud Frame Interpolation. The blue and green point clouds are two input frames and the red point clouds are four interpolated frames. We zoom in an area to display the details for better visualization.

(Liu et al. 2020). Besides, higher frame rate may enhance the performance of several applications like object tracking (Kiani Galoogahi et al. 2017). It is worth noting that video frame interpolation is commonly utilized to generate high frame rate videos from low frame rate ones (Jiang et al. 2018) (e.g., from 30 Hz to 240 Hz). Compared to the success of video frame interpolation, frame interpolation of 3D point clouds has not been well explored. Therefore, it is needed to explore frame interpolation algorithms for 3D point clouds to overcome the temporal limitations of LiDAR sensors.

Based on the above considerations, a novel task named *Point Cloud Frame Interpolation* is studied in this paper. Given two consecutive point clouds, point cloud frame interpolation aims to predict intermediate point cloud frame according to the given time step to form spatially and tem-

*Guang Chen is the corresponding author.

porally coherent point cloud streams (see Fig. 1). Consequently, low frame rate LiDAR point cloud streams (10 to 20 Hz) can be upsampled to high frame rate ones (50 to 100 Hz) based on point cloud frame interpolation.

Concretely, to achieve temporally interpolation of point cloud streams, we proposed a novel learning-based framework named PointINet (Point Cloud Frame Interpolation Network). The proposed PointINet consists of two main components: point cloud warping module and points fusion module. Two consecutive point clouds are firstly input into the point cloud warping module to warp the two point clouds to the given time step. To achieve that, we start by estimating the bi-directional 3D scene flow between two consecutive point clouds for motion estimation. 3D scene flow represents the motion field of points from one point cloud to the other one. Here we adopt a learning-based scene flow estimation network named FlowNet3D (Liu, Qi, and Guibas 2019) to predict the 3D scene flow. Then the two point clouds are warped to the given time step based on the linearly interpolated 3D scene flow. Thereafter, the key problem is how to fuse the two frames to form a new intermediate point cloud. 3D point clouds are unstructured and unordered (Qi et al. 2017a). Thus, there are no direct correspondences between points in two point clouds like pixels in two images. Consequently, it is non-trivial to perform fusion of the two point clouds. To address the problem, we propose a novel points fusion module. The points fusion module adaptively sample points from two warped point clouds and construct k -nearest-neighbor (k NN) cluster for each sampled point according to the time step to adjust the contributions of two point clouds. After that, the proposed attentive points fusion adopts an attention mechanism to aggregate points in each cluster to generate the intermediate point clouds. The overall architecture of the proposed PointINet is shown in Fig. 2.

To evaluate the proposed method, we design both qualitative and quantitative experiments. Besides, experiments on applications are also performed to evaluate the quality of the generated interpolated point clouds. Extensive experiments on two large scale outdoor LiDAR datasets demonstrate the effectiveness of the proposed PointINet.

To summarize, our main contributions are as follows:

- To overcome the temporal limitations of LiDAR sensors, a novel task Point Cloud Frame Interpolation is studied.
- A new learning-based framework named PointINet is presented to effectively generate intermediate frames between two consecutive point clouds.
- Both qualitative and quantitative experiments are conducted to verify the validity of the proposed method.

Related work

In this section we briefly review the literature relevant to point cloud frame interpolation. We start by describing common methods for video frame interpolation and then review 3D scene flow estimation methods for point clouds.

Video frame interpolation

Currently a large number of video frame interpolation methods are based on optical flow estimation (Liu et al. 2019;

Reda et al. 2019; Jiang et al. 2018; Xu et al. 2019; Liu et al. 2017). One of the most representative work of optical flow-based methods is Super SloMo (Jiang et al. 2018), which utilizes learning-based method to predict bi-directional optical flow to estimate the motion between consecutive frames. Then two input frames are further warped and fused with occlusion reasoning to generate the final intermediate frames. (Reda et al. 2019) utilizes cycle consistency to support unsupervised learning of video frame interpolation. (Xu et al. 2019) proposes a quadratic video interpolation method to exploit the acceleration information in videos. Another part of the methods for video frame interpolation are kernel-based (Niklaus, Mai, and Liu 2017a,b). (Niklaus, Mai, and Liu 2017a) estimates a kernel on each location and predict the output pixel locations by performing convolution on the patches. (Niklaus, Mai, and Liu 2017b) further improves the method by formulating frame interpolation as local separable convolution over input frames using pairs of 1D kernels. Recently, (Bao et al. 2019) combines kernel and optical flow-based methods. They utilize optical flow to predict rough locations of pixels and then refine the location using estimated kernels.

3D Scene flow estimation

3D scene flow of point clouds can be considered as a promotion of 2D optical flow in 3D scenes, which represents the 3D motion field of points. Compared with the high research interest in 2D optical flow estimation (Ilg et al. 2017; Dosovitskiy et al. 2015; Sun et al. 2018), there is relative little work on 3D scene flow estimation. FlowNet3D (Liu, Qi, and Guibas 2019) is a pioneering work of deep learning-based 3D scene flow estimation. (Liu, Qi, and Guibas 2019) proposes a flow embedding layer to model the motion of points in different point clouds. Following FlowNet3D, FlowNet3D++ (Wang et al. 2020) proposes geometric constraints to further improve the accuracy. HPLFlowNet (Gu et al. 2019) introduces Bilateral Convolutional Layers (BCL) in scene flow estimation. PointPWC-Net (Wu et al. 2019) proposes a novel cost volume and estimates the 3D scene flow in a coarse-to-fine manner. Recently, (Mittal, Okorn, and Held 2020) provides several unsupervised loss functions to support the generalization of pre-trained scene flow estimation models on more real datasets. In our implementation, we select FlowNet3D to perform 3D scene flow estimation between two point clouds due to the simplicity and effectiveness.

Point cloud frame interpolation

In this section, we first introduce the overall architecture of the proposed point cloud frame interpolation network (PointINet) and then explain the details of the two key components of PointINet, namely point cloud warping module and points fusion module.

Overall architecture

The overall architecture of PointINet is shown in Fig. 2. Given two consecutive point clouds $P_0 \in \mathbb{R}^{N \times 3}$ and $P_1 \in \mathbb{R}^{N \times 3}$ with a time step $t \in (0, 1)$, the goal of PointINet

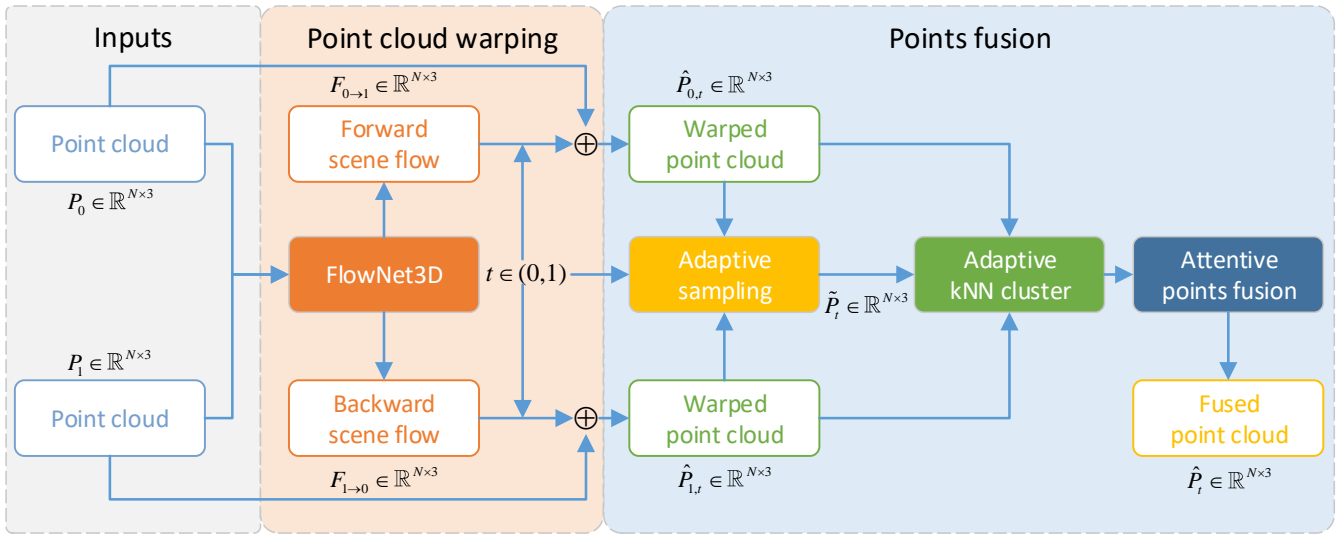


Figure 2: Overall architecture of the proposed PointINet. Given the input two consecutive point clouds, PointINet follows a pipeline consists of point cloud warping module and points fusion module.

is to predict the intermediate point cloud \hat{P}_t at time step t . PointINet consists of two key modules: point cloud warping module to warp the two input point clouds to the given time step t and points fusion module to fuse the two warped point clouds. We will describe the two modules in detail below.

Point cloud warping

Given two point clouds P_0 and P_1 , point cloud warping module aims to predict the position of each point of P_0 in $\hat{P}_{0,t}$, where $\hat{P}_{0,t}$ is the corresponding point cloud of P_0 at time step t (also predict $\hat{P}_{1,t}$ for P_1). The key point here is to estimate the motion of each point from P_0 to $\hat{P}_{0,t}$. We first predict the bi-directional 3D scene flow $F_{0 \rightarrow 1} \in \mathbb{R}^{N \times 3}$ and $F_{1 \rightarrow 0} \in \mathbb{R}^{N \times 3}$ between two point clouds P_0 and P_1 to estimate the motion of points. 3D scene flow is the 3D motion field of points, which can be regarded as a promotion of optical flow in 3D point clouds. Here we utilize an existing learning-based framework FlowNet3D (Liu, Qi, and Guibas 2019) to estimate the bi-directional 3D scene flow. Suppose that the motion of points between two consecutive frames of point clouds is linear, the scene flow $F_{0 \rightarrow t}$ and $F_{1 \rightarrow t}$ can be approximated by linearly interpolating $F_{0 \rightarrow 1}$ and $F_{1 \rightarrow 0}$, which can be represented as

$$\begin{aligned} F_{0 \rightarrow t} &= t \times F_{0 \rightarrow 1} \\ F_{1 \rightarrow t} &= (1 - t) \times F_{1 \rightarrow 0} \end{aligned} \quad (1)$$

Then P_0 and P_1 can be warped to the given time step t based on the interpolated 3D scene flow $F_{0 \rightarrow t}$ and $F_{1 \rightarrow t}$,

$$\begin{aligned} \hat{P}_{0,t} &= P_0 + F_{0 \rightarrow t} \\ \hat{P}_{1,t} &= P_1 + F_{1 \rightarrow t} \end{aligned} \quad (2)$$

Points fusion

The goal of the points fusion module is to fuse the two warped point clouds and generate intermediate point clouds.

The architecture of the points fusion module is displayed in the right column of Fig. 2. The input of this module is two warped point clouds $\hat{P}_{0,t} \in \mathbb{R}^{N \times 3}$ and $\hat{P}_{1,t} \in \mathbb{R}^{N \times 3}$ and the output is the fused intermediate point cloud $\hat{P}_t \in \mathbb{R}^{N \times 3}$. In video frame interpolation, the fusion step mostly concentrates on occlusion and missing regions prediction due to the structured 2D grid-based representation. However, the fusion of two point clouds is non-trivial because point clouds are unstructured and unordered. In the proposed PointINet, we start the fusion by adaptively sampling points from two warped point clouds based on time step t and then construct k -nearest-neighbor (k NN) clusters centered on the sampled points. After that, the attentive points fusion module adopts an attention mechanism to generate the final intermediate point clouds. The details of the key components of the points fusion module will be described below.

Adaptive sampling The first step of the points fusion module is to combine the two warped point clouds to a new point cloud. Intuitively, the contributions of the two point clouds to the intermediate point clouds are not always the same. For example, the intermediate frame \hat{P}_t at $t = 0.2$ should be more similar to the first frame P_0 than the second frame P_1 . Based on the above observation, we randomly sample N_0 and N_1 points from $\hat{P}_{0,t}$ and $\hat{P}_{1,t}$ to generate two sampled point clouds $\tilde{P}_{0,t} \in \mathbb{R}^{N_0 \times 3}$ and $\tilde{P}_{1,t} \in \mathbb{R}^{N_1 \times 3}$, respectively. Here, $N_0 = (1 - t) \times N$ and $N_1 = t \times N$. This operation enables the network to adaptively adjust the contributions of the two warped point clouds according to the target time step t . The point cloud close to time step t contributes more to the intermediate frame \hat{P}_t . After that, $\tilde{P}_{0,t}$ and $\tilde{P}_{1,t}$ are combined to a new point cloud $\tilde{P}_t \in \mathbb{R}^{N \times 3}$.

Adaptive k NN cluster We input \tilde{P}_t into the adaptive k NN cluster module to generate k -nearest-neighbor clusters as input to the followed attentive points fusion module. For

each point in \hat{P}_t , we search for K nearest neighbors in two warped point clouds $\hat{P}_{0,t}$ and $\hat{P}_{1,t}$. Similar to adaptive sampling, the number of neighbors in $\hat{P}_{0,t}$ and $\hat{P}_{1,t}$ are also adaptively adjusted according to t to balance the contributions of two point clouds. Thus, we query K_0 neighbors in $\hat{P}_{0,t}$ and K_1 neighbors in $\hat{P}_{1,t}$, where $K_0 = (1-t) \times K$ and $K_1 = t \times K$. As a result, we obtain N clusters and each cluster consists of K neighbor points. Denoting the center point of a cluster as x^i and the neighbor points as $\{x_1^i, \dots, x_k^i, \dots, x_K^i\} \in \mathbb{R}^{K \times 3}$. Then each neighbor point is subtracted by the center point as $(x_k^i - x^i)$ to obtain the relative position of neighbor points in a cluster. Besides, the Euclidean distance between neighbor point and the center point $\|x_k^i - x^i\|_2$ is calculated as an additional channel of the cluster. Consequently, the final feature of a single cluster can be denoted as $F^i = \{f_1^i, \dots, f_k^i, \dots, f_K^i\} \in \mathbb{R}^{K \times 4}$.

Attentive points fusion Attention mechanism has been widely used in 3D point cloud learning (Yang et al. 2019; Wang et al. 2019; Wang, He, and Ma 2019). Here we adopt an attention mechanism to aggregate the feature of neighbor points to generate new points for the intermediate point clouds. The network architecture of the attentive points fusion module can be seen in Fig. 3. Inspired by PointNet (Qi et al. 2017a) and PointNet++ (Qi et al. 2017b), we input the feature F^i of a single cluster into a shared multi layer perceptron (Shared-MLP) to generate a feature map. Then the followed maxpool layer and a Softmax function are applied to predict one-dimensional attentive weights $W^i = \{w_1^i, \dots, w_k^i, \dots, w_K^i\} \in \mathbb{R}^{K \times 1}$ for all neighbor points in the cluster. After that, the new point \hat{x}^i can be represented as the weighted sum of the neighbor points,

$$\hat{x}^i = \sum_{k=1}^K x_k^i \cdot w_k^i, \quad i = 1, \dots, N \quad (3)$$

Finally, the generated intermediate point cloud \hat{P}_t can be represented as $\hat{P}_t = \{\hat{x}^1, \dots, \hat{x}^N\} \in \mathbb{R}^{N \times 3}$. Intuitively, the proposed attentive points fusion module can assign higher weights to points in the cluster that are more consistent with the target point cloud. After the points fusion module, each generated point in the new intermediate point cloud is aggregated from neighbor points in the two point clouds in its receptive field. Besides, the contributions of two point clouds are dynamically adjusted according to the time step t with the help of adaptive sampling and adaptive k NN cluster module. Consequently, the generated intermediate point cloud is an effective fusion of the two input point clouds.

Loss

Chamfer distance (Fan, Su, and Guibas 2017) is commonly used to measure the similarity of two point clouds. Here we utilize chamfer distance to supervise the training of the proposed PointNet. Given the generated intermediate point cloud $\hat{P}_t \in \mathbb{R}^{N \times 3}$ and the ground truth one $P_t \in \mathbb{R}^{N \times 3}$, the

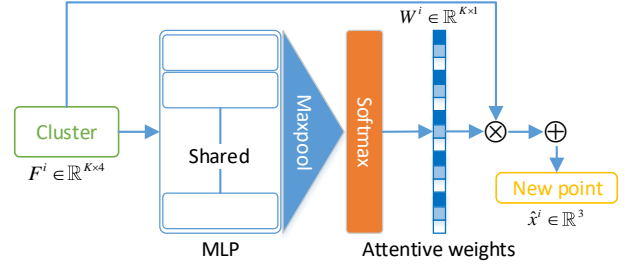


Figure 3: The network architecture of the proposed attentive points fusion module.

chamfer distance loss can be represented as

$$\mathcal{L} = \frac{1}{N} \sum_{\hat{x}^i \in \hat{P}_t} \min_{x^j \in P_t} \|\hat{x}^i - x^j\|_2 + \frac{1}{N} \sum_{x^j \in P_t} \min_{\hat{x}^i \in \hat{P}_t} \|\hat{x}^i - x^j\|_2 \quad (4)$$

where $\|\cdot\|_2$ represents the L_2 -norm.

Experiments

We perform both qualitative and quantitative experiments to demonstrate the performance of the proposed method. Besides, we also perform experiments on two applications (*i.e.*, keypoints detection and multi frame Iterative Closest Point (ICP)) to better evaluate the quality of the generated intermediate point clouds.

Datasets

We evaluate the proposed method on two large scale outdoor LiDAR datasets, namely KITTI odometry dataset (Geiger, Lenz, and Urtasun 2012) and nuScenes dataset (Caesar et al. 2020). KITTI odometry dataset provides 11 sequences with ground truth (00-10) and we use sequence 00 to train the network, 01 to validate and the others to evaluate. NuScenes dataset consists of 850 training scenes and we use the first 100 scenes for training and the remaining 750 scenes for evaluation. Due to the lack of high frame rate LiDAR sensors, we simply downsample the 10 Hz point clouds in KITTI odometry dataset to 2 Hz and the 20 Hz point clouds in nuScenes dataset to 4 Hz for training and the quantitative experiments. Consequently, there are 4 intermediate point clouds between two consecutive frames in the downsampled point cloud streams.

Implementation details

We start by training FlowNet3D on Flythings3D dataset (Mayer et al. 2016) and then refine the network on KITTI scene flow dataset (Menze and Geiger 2015). We directly use the data pre-processed by (Liu, Qi, and Guibas 2019) to train FlowNet3D. Then we further refine the pre-trained FlowNet3D on KITTI odometry dataset and nuScenes dataset, respectively. During this procedure, the current frame with a randomly selected frame within N_s frames before and after it are used as a training pair. Then the first frame is warped to the second frame with the predicted

scene flow and the chamfer distance (see Eq. 4) between the warped point cloud and the second point cloud is adopted as the loss function to supervise the refinement of FlowNet3D. After that, the weight of FlowNet3D is fixed when training the followed points fusion module. During the training of the points fusion module, two consecutive frames and a randomly sampled frame from the 4 intermediate point clouds with the corresponding time step are utilized as a training sample. We randomly downsample the point clouds to 16384 points during training and the number of neighbor points K is set to 32 in our implementation. The channels of the layers of Shared-MLP in attentive points fusion module are set to [64, 64, 128]. All of the network is implemented using PyTorch (Paszke et al. 2019) and Adam is used as the optimizer. Besides, the points fusion module is only trained on KITTI odometry dataset and we simply generalize the trained model to nuScenes dataset for evaluation.

Qualitative experiments

The goal of the proposed PointINet is to generate high frame rate LiDAR streams from low frame rate ones. However, there are no existing high frame rate LiDAR sensors. Thus, we train the FlowNet3D with $N_s = 1$ to provide proper scene flow estimation for closer point clouds and then directly apply the points fusion module trained on the downsampled point cloud streams on 10 Hz point cloud streams of KITTI odometry dataset to generate high frame rate point cloud streams. Here we provide a qualitative visualization in Fig. 4, where the number of points here is set to 32768. The 10 Hz point cloud streams are upsampled to 40 Hz and the time step of intermediate frames are set to 0.25, 0.50 and 0.75. According to Fig. 4, the proposed PointINet well estimates the motion of points between two clouds and the fusion algorithm can preserve the details of the point cloud. In addition to that, we also provide several demo videos in the supplementary materials to compare high frame rate point cloud streams with low frame rate point cloud streams. According to the demo videos, the high frame rate point cloud streams are obviously temporally and spatially smoother than low frame rate ones.

Quantitative experiments

Evaluation metrics We evaluate the similarity and consistency between the generated point clouds and the ground truth ones on the downsampled point cloud streams using two evaluation metrics: Chamfer distance (CD) and Earth mover’s distance (EMD). CD is previously described in Eq. 4. EMD is also a commonly used metric to compare two point clouds (Weng et al. 2020), which is implemented by solving a linear assignment problem. Given two point clouds $\hat{P}_t \in \mathbb{R}^{N \times 3}$ and $P_t \in \mathbb{R}^{N \times 3}$, EMD can be calculated as

$$EMD = \min_{\phi: \hat{P}_t \rightarrow P_t} \frac{1}{N} \sum_{\hat{x} \in \hat{P}_t} \|\hat{x} - \phi(\hat{x})\|_2 \quad (5)$$

where $\phi: \hat{P}_t \rightarrow P_t$ is a bijection.

Baselines To demonstrate the performance of the proposed PointINet, we define 3 baselines to make comparison with our method: (1) *Identity*. We simply duplicate the

Metric	Identity	Align-ICP	Scene flow	Ours
CD↓	1.398	0.752	0.687	0.457
EMD↓	68.93	83.79	57.13	39.46

Table 1: Results of quantitative evaluation of PointINet and other baselines on KITTI odometry dataset.

Metric	Identity	Align-ICP	Scene flow	Ours
CD↓	0.617	0.555	0.511	0.487
EMD↓	54.24	51.12	50.97	47.98

Table 2: Results of quantitative evaluation of PointINet and other baselines on nuScenes dataset.

first point cloud frame as the intermediate point clouds. (2) *Align-ICP*. We firstly estimate the rigid transformation between the two consecutive frame of point clouds using Iterative Closest Point (ICP) algorithm and then linearly interpolate that to obtain the transformation between the first frame and intermediate frame. Thereafter, the first point cloud is transformed to the intermediate frame based on the transformation. (3) *Scene flow*. We estimate the 3D scene flow between the consecutive two frames using FlowNet3D and calculate the scene flow from the first frame to the intermediate frame by linear interpolation. Then the intermediate point clouds are obtained by transform the first point cloud according to the 3D scene flow. All of the point clouds are downsampled to 16384 points by randomly sampling in quantitative experiments.

Results The CD and EMD of the proposed PointINet and other baselines on KITTI odometry dataset and nuScenes dataset are shown in Table 1 and Table 2, respectively. According to the results, the performance of our method significantly outperforms other baselines. For example, the chamfer distance of the proposed PointINet is about 1/3, 3/5 and 2/3 of *Identity*, *Align-ICP* and *Scene flow* on KITTI odometry dataset, respectively. It is worth noting that our method is superior to *Scene flow* by an obvious margin, which also reflects the effectiveness of the points fusion module. Noting that we only train the points fusion module on KITTI odometry dataset and the results on nuScenes dataset also demonstrate the generalization ability of the network.

Applications

In order to better evaluate the quality of the generated intermediate point clouds and the similarity with original point clouds, we apply two applications on the interpolated point cloud streams and the original ones, namely Keypoints detection and Multi frame ICP. We firstly respectively downsample the 10 Hz point clouds in KITTI odometry dataset and 20 Hz point clouds in nuScenes dataset to 5 Hz and 10 Hz and then interpolate them to the original frame rates as the interpolated point cloud streams. The results of the two applications on the two different point cloud streams are compared to verify the validity of the proposed PointINet.

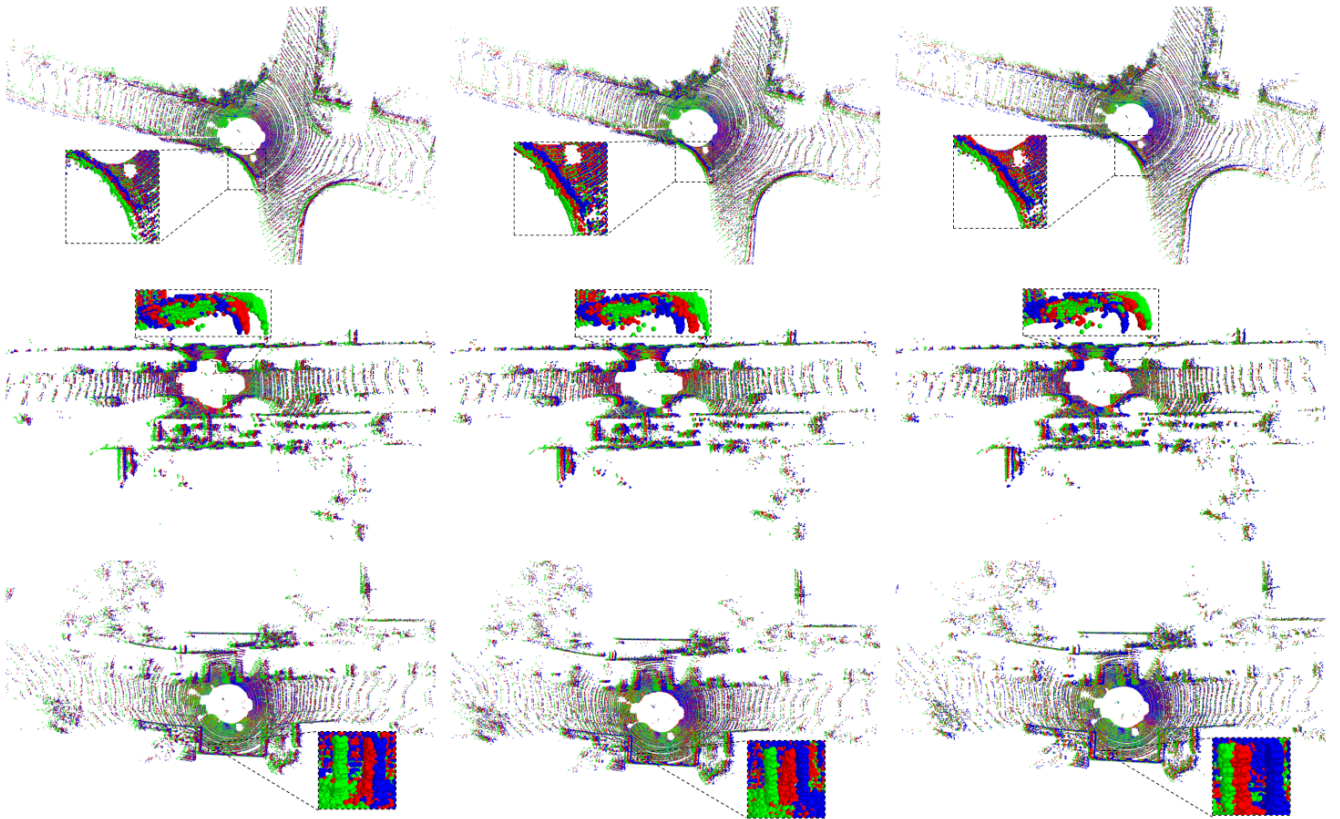


Figure 4: Qualitative results of the proposed PointINet. From top to bottom rows are the interpolation results of 3 pairs of consecutive frames. The time step t of columns from left to right are 0.25, 0.50 and 0.75, respectively. Blue, green and red point clouds represent first frames, second frames and the predicted intermediate frames, respectively. Besides, we zoom in an area of the point cloud and then rotate it to a proper perspective to better visualize the details of the interpolated point cloud.

Keypoints detection We perform 3D keypoints detection in the two point cloud streams and evaluate the repeatability of the detected keypoints. We choose 3 handcrafted 3D keypoints, namely SIFT-3D (Flint, Dick, and Van Den Hengel 2007), Harris-3D (Sipiran and Bustos 2011) and ISS (Zhong 2009). All of the keypoints are extracted using the implementation in PCL (Rusu and Cousins 2011). A keypoint in a point cloud is considered repeatable if its distance to the nearest keypoint in the other point cloud (after rigid transformation based on the ground truth pose) is within a threshold δ_r (δ_r is set to 0.5 m here) and the repeatability is the ratio of repeatable keypoints. We calculate the average repeatability of keypoints in current point cloud with keypoints in 5 frames before and after it and the number of keypoints is set to 256. Due to the lack of per-frame ground truth pose in nuScenes dataset, the keypoints detection experiments are only performed on KITTI odometry dataset and the results are shown in Table 3. According to the results, the repeatability of the interpolated point cloud streams is only slightly reduced compared with the original point cloud streams. For example, the repeatability of Harris-3D of interpolated point clouds is only 0.017 lower than that of original point clouds. The results reflects the high consistency of the generated intermediate point clouds with the ground truth point clouds

Keypoints	Harris-3D	SIFT-3D	ISS
Original	0.155	0.174	0.163
Interpolated	0.138	0.151	0.133

Table 3: The repeatability of 3 different keypoints of original and interpolated point clouds on KITTI odometry dataset.

from the side.

Multi frame ICP We perform iterative closest point (ICP) algorithm on N_m consecutive frames to estimate the rigid transformation between the first and last frames. N_m is set to 10 on KITTI odometry dataset. For nuScenes dataset, the ground truth pose is only provided for keyframes (about 2 Hz). Thus, N_m is set to be the same as the number of frames between two keyframes on nuScenes dataset. We utilize the implementation in PCL to perform ICP algorithm. The one-by-one transformation are accumulated to obtain the transformation between the first and last frames. Relative translation error (RTE) and relative rotation error (RRE) are calculated to evaluate the error of the estimated transformation of multi frame ICP. The results on KITTI odometry dataset and nuScenes dataset are displayed in Table 4 and Table 5,

Metric	Original	Interpolated	Difference
RTE (m)	4.31	4.57	0.26
RRE (deg)	2.70	2.95	0.25

Table 4: The performance of multi frame ICP of original and interpolated point cloud streams on KITTI odometry dataset.

Metric	Original	Interpolated	Difference
RTE (m)	1.65	1.72	0.07
RRE (deg)	0.91	0.92	0.01

Table 5: The performance of multi frame ICP of original and interpolated point cloud streams on nuScenes dataset.

respectively. We also calculate the difference between the errors of original and interpolated point cloud streams and display the results in right column of Table 4 and Table 5 for better comparison. According to the results, the RTE and RRE of the multi-frame ICP algorithm on the interpolated point cloud streams are very close to that on original point cloud streams. For example, The RTE on nuScenes dataset of the two results differs by only 0.07 m according to Table 5. The close performance indicates the similarity between the generated intermediate point clouds with the ground truth ones.

According to the experiments on the two applications, the performance of the algorithm on the interpolated point clouds is slightly inferior to the original point cloud streams due to the possible error of the proposed interpolation method. Nonetheless, the close performance on the two applications proves the high similarity and consistency of the generated point clouds with the original ones.

Efficiency

The efficiency of the proposed PointINet is evaluated on a PC with NVIDIA Geforce RTX 2060 and the average runtime to generate one intermediate frame for point clouds contain 16384, 32768 and 65536 points are displayed in Table 6. According to the results, most of the runtime is used to warp the point cloud and the proposed points fusion module requires relatively little time for computation. However, the computation time for points fusion module increases with the number of points due to the per-point computation for fusion. Overall, the proposed PointINet can efficiently generate intermediate frames.

Ablation study

We perform several ablation studies to analyze the effect of different components of the proposed PointINet (*e.g.*, adaptive sampling, adaptive k NN cluster and attentive points fusion) to the final results. The experimental setting is consistent with the quantitative experiments and we also use chamfer distance (CD) and earth mover’s distance (EMD) to evaluate the performance. All of the ablation studies are performed on KITTI odometry dataset.

Number of points	16384	32768	65536
Point cloud warping	167.3	291.1	529.3
Points fusion	36.4	81.3	196.6
PointINet	203.7	372.4	725.9

Table 6: The runtime (ms) of PointINet and its components for different number of points.

Methods	CD↓	EMD↓
full PointINet	0.457	39.46
w/o adaptive sampling	0.580	48.00
w/o adaptive k NN cluster	0.534	41.66
w/o attentive points fusion	0.555	40.67

Table 7: The quantitative evaluation results of ablation studies on KITTI odometry dataset.

Adaptive sampling We replace the adaptive sampling strategy by simply randomly sampling half of the points in two warped point clouds to form a new point cloud as the input to the adaptive k NN cluster module. The results are shown in the second row of Table 7. Based on the results, the CD and EMD increase by 0.123 and 8.54 without adaptive sampling, which demonstrates that the adaptive sampling strategy significantly improves the performance.

Adaptive k NN cluster We query $K/2$ neighbor points from the two warped point clouds fixedly rather than query points based on time step t . According to the results displayed in third row of Table 7, the CD and EMD without adaptive k NN cluster increase from 0.457 to 0.534 and 39.46 to 41.66, respectively. The results prove the effectiveness of the adaptive k NN cluster module.

Attentive points fusion To demonstrate the effect of the attentive points fusion module, we directly use the point cloud \tilde{P}_t from adaptive sampling as the intermediate point cloud and display the results in the bottom row of Table 7. According to the results, the attentive points fusion module obviously enhances the final performance.

Conclusions

In this paper, a novel task named *Point Cloud Frame Interpolation* is studied and a learning-based framework PointINet is designed for this task. Given two consecutive point clouds, the task aims to predict temporally and spatially consistent intermediate frames between them. Consequently, low frame rate point cloud streams can be upsampled to high frame rates using the proposed method. To achieve that, we utilize an existing scene flow estimation network for motion estimation and then warp the two point clouds to the given time step. Then a novel learning-based points fusion module is presented to efficiently fuse the two point clouds. We design both qualitative and quantitative experiments for this task. Extensive experiments on KITTI odometry dataset and nuScenes dataset demonstrate the performance and effectiveness of the proposed PointINet.

Ethics statement

The proposed point cloud frame interpolation method may have positive effects on the development of autonomous driving and intelligent robots, which can reduce the workload of human drivers and workers and also the incidence of traffic accidents. However, this development may also bring unemployment of human drivers and workers. Besides, the proposed method may have potential military applications like military unmanned aerial vehicles, which can threaten the safety of humans. We should explore more applications which can improve the quality of human life rather than harmful ones.

Acknowledgments

This work is funded by National Natural Science Foundation of China (No. 61906138), the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539 (Human Brain Project SGA3), and the Shanghai AI Innovation Development Program 2018.

References

- Bao, W.; Lai, W.-S.; Zhang, X.; Gao, Z.; and Yang, M.-H. 2019. Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE transactions on pattern analysis and machine intelligence*.
- Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11621–11631.
- Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; Van Der Smagt, P.; Cremers, D.; and Brox, T. 2015. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, 2758–2766.
- Fan, H.; Su, H.; and Guibas, L. J. 2017. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 605–613.
- Flint, A.; Dick, A.; and Van Den Hengel, A. 2007. Thrift: Local 3d structure recognition. In *9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications (DICTA 2007)*, 182–188. IEEE.
- Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 3354–3361. IEEE.
- Gu, X.; Wang, Y.; Wu, C.; Lee, Y. J.; and Wang, P. 2019. HplflowNet: Hierarchical permutohedral lattice flowNet for scene flow estimation on large-scale point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3254–3263.
- Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2462–2470.
- Jiang, H.; Sun, D.; Jampani, V.; Yang, M.-H.; Learned-Miller, E.; and Kautz, J. 2018. Super sloMo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9000–9008.
- Kiani Galoogahi, H.; Fagg, A.; Huang, C.; Ramanan, D.; and Lucey, S. 2017. Need for speed: A benchmark for higher frame rate object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, 1125–1134.
- Liu, H.; Liao, K.; Lin, C.; Zhao, Y.; and Guo, Y. 2020. Pseudo-LiDAR Point Cloud Interpolation Based on 3D Motion Representation and Spatial Supervision. *arXiv preprint arXiv:2006.11481*.
- Liu, X.; Qi, C. R.; and Guibas, L. J. 2019. FlowNet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 529–537.
- Liu, Y.-L.; Liao, Y.-T.; Lin, Y.-Y.; and Chuang, Y.-Y. 2019. Deep video frame interpolation using cyclic frame generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 8794–8802.
- Liu, Z.; Yeh, R. A.; Tang, X.; Liu, Y.; and Agarwala, A. 2017. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, 4463–4471.
- Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; and Brox, T. 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4040–4048.
- Menze, M.; and Geiger, A. 2015. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3061–3070.
- Mittal, H.; Okorn, B.; and Held, D. 2020. Just go with the flow: Self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11177–11185.
- Niklaus, S.; Mai, L.; and Liu, F. 2017a. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 670–679.
- Niklaus, S.; Mai, L.; and Liu, F. 2017b. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, 261–270.
- Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, 8026–8037.

- 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 *Advances in neural information processing systems*, 5099–5108.
- Reda, F. A.; Sun, D.; Dundar, A.; Shoeybi, M.; Liu, G.; Shih, K. J.; Tao, A.; Kautz, J.; and Catanzaro, B. 2019. Unsupervised video interpolation using cycle consistency. In *Proceedings of the IEEE International Conference on Computer Vision*, 892–900.
- Rusu, R. B.; and Cousins, S. 2011. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, 1–4. IEEE.
- Sipiran, I.; and Bustos, B. 2011. Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes. *The Visual Computer* 27(11): 963.
- Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J. 2018. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8934–8943.
- Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Shan, J. 2019. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 10296–10305.
- Wang, X.; He, J.; and Ma, L. 2019. Exploiting Local and Global Structure for Point Cloud Semantic Segmentation with Contextual Point Representations. In *Advances in Neural Information Processing Systems*, 4571–4581.
- Wang, Z.; Li, S.; Howard-Jenkins, H.; Prisacariu, V.; and Chen, M. 2020. FlowNet3D++: Geometric losses for deep scene flow estimation. In *The IEEE Winter Conference on Applications of Computer Vision*, 91–98.
- Weng, X.; Wang, J.; Levine, S.; Kitani, K.; and Rhinehart, N. 2020. Inverting the Pose Forecasting Pipeline with SPF2: Sequential Pointcloud Forecasting for Sequential Pose Forecasting. *CoRL*.
- Wu, W.; Wang, Z.; Li, Z.; Liu, W.; and Fuxin, L. 2019. PointPWC-Net: A Coarse-to-Fine Network for Supervised and Self-Supervised Scene Flow Estimation on 3D Point Clouds. *arXiv preprint arXiv:1911.12408*.
- Xu, X.; Siyao, L.; Sun, W.; Yin, Q.; and Yang, M.-H. 2019. Quadratic video interpolation. In *Advances in Neural Information Processing Systems*, 1647–1656.
- Yang, J.; Zhang, Q.; Ni, B.; Li, L.; Liu, J.; Zhou, M.; and Tian, Q. 2019. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3323–3332.
- Zhong, Y. 2009. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 689–696. IEEE.