

DiffPCI: Large Motion Point Cloud frame Interpolation with Diffusion Model

Tianyu Zhang¹ Haobo Jiang² Jian Yang¹ Jin Xie^{3*}

¹ College of Computer Science, Nankai University ² Nanyang Technological University

³ School of Intelligence Science and Technology, Nanjing University

tianyu.zhang@mail.nankai.edu.cn haobo.jiang@ntu.edu.sg csjyang@nankai.edu.cn csjxie@nju.edu.cn

Abstract

Point cloud interpolation aims to recover intermediate frames for temporally smoothing a point cloud sequence. However, real-world challenges, such as uneven or large scene motions, cause existing methods to struggle with limited interpolation precision. To address this, we introduce DiffPCI, a novel diffusion interpolation model that formulates the frame interpolation task as a progressive denoising diffusion process. Training DiffPCI involves two key stages: a forward interpolation diffusion process and a reverse interpolation denoising process. In the forward process, the clean intermediate frame is progressively transformed into a noisy one through continuous Gaussian noise injection. The reverse process then focuses on training a denoiser to gradually refine this noisy frame back to the ground-truth frame. In particular, we derive a point cloud interpolation-specific variational lower bound as our optimization objective for denoiser training. Furthermore, to alleviate the interpolation error especially in highly dynamic scenes, we develop a novel full-scale, dual-branch denoiser that enables more comprehensive front-back frame information fusion for robust bi-directional interpolation. Extensive experiments demonstrate that DiffPCI significantly outperforms current state-of-the-art frame interpolation methods (e.g. 27% and 860% reduction in the Chamfer Distance and Earth Mover’s Distance on Nuscenes).

1. Introduction

Advancements in 3D sensor technology, particularly LiDAR, have facilitated the utilization of sequential point clouds in applications such as autonomous driving, environmental monitoring, and 3D modeling. However, the study of high-frequency sequential point clouds is still limited by the frequency constraint of existing LiDAR sensors. Point cloud frame interpolation (PCI) addresses this challenge by generating intermediate frames, provid-

ing richer environmental information and smoother target representations, with a variety of downstream applications such as autonomous driving and 3D reconstruction [5, 13–15, 26, 42, 48, 52].

In earlier works, intermediate point clouds were constructed as Pseudo-LiDAR [19, 20] using known camera intrinsics. Recent approaches [22, 45, 47, 54] focus on directly interpolating point clouds onto the LiDAR data by approximating correspondences between points in consecutive frames. However, methods that estimate correspondences using implicit models [12, 45, 54] suffer from poor interpretability and require complex model designs. By contrast, methods such as PointNet [22] and FastPCI [47], which explicitly estimate scene flow for frame interpolation, offer greater interpretability. Among these, PointNet [22] employs bidirectional pre-trained optical flow to estimate point mappings between frames, addressing the imbalance issue in frame predictions caused by unidirectional optical flow. However, in highly dynamic autonomous driving scenarios, significant and irregular object motion greatly increases the uncertainty of frame interpolation, resulting in limited performance and high-noise predictions, particularly in the cases involving substantial movement.

The geometric distortion or motion trajectory hopping of the interpolation point cloud is often generated by the above methods due to the local feature correlation fracture. The fundamental reason is that these methods treat interpolation as a deterministic optimization problem and try to predict the intermediate frame directly through a single feed-forward network or iterative optimization while ignoring the probability distribution characteristics and multi-modal possibilities implied in the motion process. The diffusion model can explicitly model the gradient path from noise distribution to target data distribution by decomposing the data generation process into a Markov chain with gradual de-noising, which provides a new perspective for solving large motion interpolation problems. Currently, the effectiveness of the diffusion model in 3D point cloud generation [21, 25, 44, 51, 55] and video interpolation [3, 11, 38] has been studied and verified, but its potential in point cloud

*Corresponding author.

time series interpolation has not been fully explored.

In this paper, we introduce diffusion model into our PCI task, and develop an effective diffusion model-driven PCI framework, DiffPCI, by formulating the interpolation task as a progressive denoising process. Specifically, training our DiffPCI contains two main operations: a forward interpolation diffusion process and a reverse interpolation denoising process. The forward process aims to convert the ground-truth intermediate point cloud frame step-by-step into a noisy one through progressive noise injection. The reverse process then focuses on learning a denoiser to progressively refine this noisy frame to the ground-truth one. Notably, to ensure effective denoiser training, we develop a PCI-specific variational lower bound for model learning. Furthermore, to alleviate the interpolation challenge caused by significant or uneven object motions, we develop an effective full-scale, dual-branch denoising network, comprehensively fusing the forward-branch and backward-branch denoising outputs for more reliable intermediate frame recovering. In particular, a stack of densely connected layers is also designed for full-scale motion awareness and information fusion, thereby producing more discriminative geometric representations for point cloud interpolation.

To summarize, our main contributions are as follows:

- To the best of our knowledge, we are the first to introduce diffusion models to the PCI field and propose a novel diffusion-based interpolation model that formulates intermediate frame estimation as a progressive denoising process.
- To handle challenging scenarios with non-uniform or large motions, we design a full-scale denoiser that integrates a dual-branch full-scale motion estimator with an adaptive selection module. This denoiser employs multiple densely connected layers for full-scale geometric information fusion, enhancing geometric representations for improved interpolation accuracy.
- DiffPCI outperforms all existing methods on diverse autonomous driving datasets. In particular, on Nuscenes with large motion scenes, the chamfer distance and Earth movement distance are reduced by 27% and 860%.

2. Related Work

2.1. Diffusion model for point cloud data

Diffusion models [7, 36] have gained significant attention in recent years. As demonstrated in [24], they can reconstruct the original natural image from any noisy version and its time exponent t by training a neural network. This approach has led to substantial advancements in image/3D generation [49] and video synthesis [4, 8, 43]. However, applying these models directly to point clouds is challenging due to the irregular distribution of points in three-dimensional space, as opposed to the structured grid format of images.

To address this, [25] first introduces a method that treats points like particles in a thermodynamic system interacting with a heat source, transforming the noise distribution into a desired shape distribution for static point cloud generation. PVD [55] developed a point voxel diffusion model for 3D shape generation and completion. Afterward, LION [44] further advanced the field by integrating feature-based and point-based latent diffusion models with VAE for generating point cloud shapes. While these methods are limited to generating static point cloud data, simulating dynamic point clouds is essential for point cloud frame interpolation (PCI). Subsequently, DifFlow3D [21] and DiffSF [51] are developed. DifFlow3D proposes a plug-and-play diffusion-based refinement module guided by strong conditions of coarse stream embeddings, geometric encoding, and cross-frame cost. DiffSF designs an end-to-end diffusion model for efficient scene flow estimation. Inspired by these two methods, we aim to develop an end-to-end diffusion model tailored for point cloud interpolation in non-uniform motion scenarios. Yet, it is impractical to interpolate the frame linearly using the above method. PCI aims to generate a complete point cloud at the intermediate moment, and it needs to solve the motion inference and geometric reconstruction problems at the same time. Although the motion modeling technique is shared with the scene flow estimation algorithm, the interpolation task is more complicated due to the coupled generative reconstruction and physical constraints.

2.2. Point Cloud Frame Interpolation

The field of video frame interpolation has seen significant growth [1, 10, 16, 18, 23, 28–31, 34, 35, 41, 46] in the past decade. Many researchers apply diffusion models for this task [3, 11], demonstrating impressive performance. However, the point cloud frame interpolation (PCI) algorithm based on diffusion model has not been explored. Accurate estimation of inhomogeneous motion between adjacent point cloud frames remains a key objective in contemporary PCI methods. PointINet [22] utilizes a pre-trained scene flow estimator to estimate linear motion bidirectionally. However, this method does not account for nonuniform motion and lacks a specialized design for motion estimators. In contrast, IDEA-Net [45] employs a deep embedding alignment method to achieve dynamic 3D point cloud interpolation. Nonetheless, this one-to-one correspondence constrains effectiveness in large-scale outdoor datasets characterized by point clouds of complex motion. NeuroGauss4D-PCI [12] integrates 4D neural field and Gaussian deformation field, yielding superior interpolation performance in non-rigid deformation scenarios; however, it suffers from suboptimal real-time performance. NeuralPCI [54] addresses this limitation by constructing an implicit 4D neural field, leveraging its continuous modeling capabilities to es-

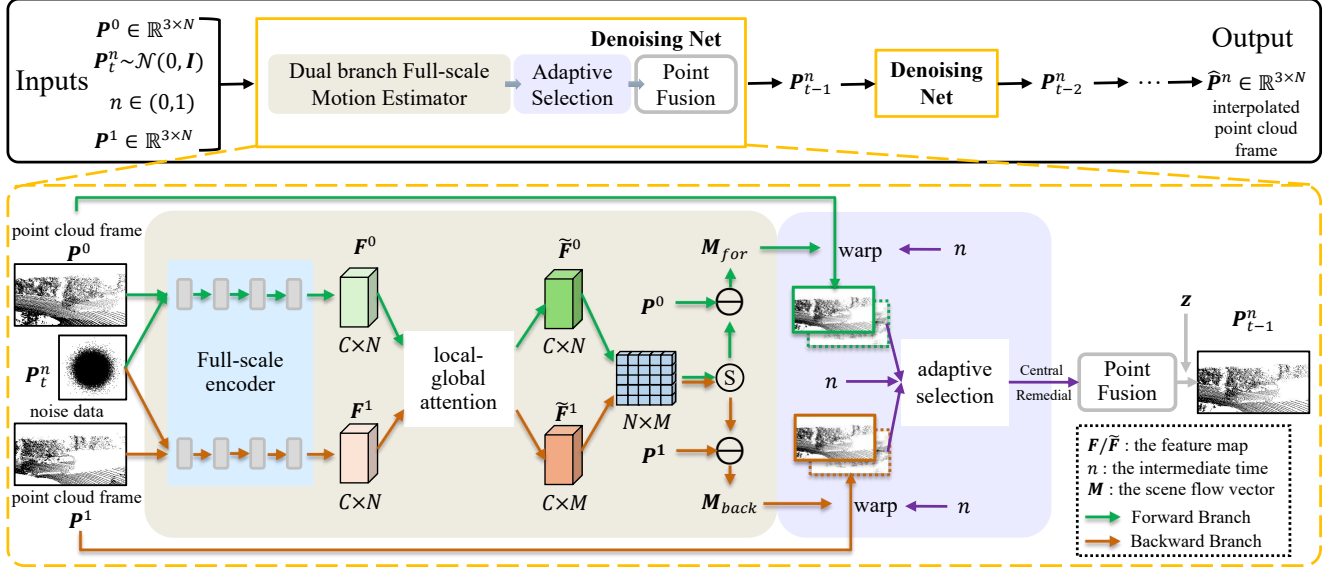


Figure 1. **Overview of Reverse Denoising pipeline.** Utilizing two adjacent frames along with an intermediate time as conditional inputs, $\mathbf{P}^0 \in \mathbb{R}^{3 \times N}$, $\mathbf{P}^1 \in \mathbb{R}^{3 \times N}$ and n , the noise data \mathbf{P}_t^n of standard Gaussian distribution is gradually converted through a dual-branch denoising network to yield the optimal interpolation result for the specified intermediate time. Specifically, the denoising network estimates bidirectional motion using a dual branch full-scale motion estimator. Then, four interpolated frames are obtained by warping the input frames. Adaptive selection module dynamically selects the central frame and remedial frame from them for point fusion module to output the fused interpolated frame.

estimate uneven motion between frames. FastPCI [47], on the other hand, employs structural guidance for motion estimation, achieving rapid and accurate PCI. Despite these advancements, both methods encounter challenges in producing satisfactory frame interpolation results in large motion scenes, such as those found in the Nuscenes dataset. In this work, we introduce a novel point cloud interpolation framework that incorporates a diffusion model, which iteratively refines interpolation results through a step-by-step denoising process. Our framework is highlighted by a dual-branch full-scale motion estimator and an adaptive selection module integrated into the reverse denoising network. It is meticulously designed for precise PCI in scenarios involving non-uniform motion, including large motion.

3. Methodology

This section introduces DiffPCI, a novel framework based on the diffusion model for large-motion point cloud frame interpolation. We first present the overview of DiffPCI in Section 3.1, including the forward diffusion and dual-branch reverse denoising process. In particular, in the denoising network of the reverse process, we customize the full-scale motion estimator and adaptive selection module for large motion scenarios in Section 3.2. We also propose several regularization losses to optimize diffusion training and will elaborate in Section 3.3.

3.1. Diffusion Interpolation Model

Given two consecutive frames with low temporal resolution $\mathbf{P}^0 \in \mathbb{R}^{3 \times N}$ and $\mathbf{P}^1 \in \mathbb{R}^{3 \times N}$ in the time step $n = 0$ and $n = 1$, as shown in Fig. 1, the objective of DiffPCI is to generate the point cloud frame $\mathbf{P}^n \in \mathbb{R}^{3 \times N}$ in arbitrary intermediate time step $n \in (0, 1)$. Inspired by the remarkable success achieved by diffusion model in the generative AI field, we also formulate the point cloud interpolation task as a denoising diffusion process, and develop an effective dual-branch diffusion interpolation model, DiffPCI, for quality intermediate frame prediction. Training our DiffPCI primarily contains the forward diffusion process and the dual-branch reverse denoising process. In the following, we elaborate on these two processes and please also refer to Algorithms 1 and 2 for detailed training and inference pipelines.

Forward Interpolation Diffusion Process. With a specific intermediate timestamp n , the forward diffusion process aims to gradually convert the ground-truth intermediate frame \mathbf{P}^n into a noisy one, forming a forward Markov chain $\mathbf{P}^n = \mathbf{P}_0^n \rightarrow \mathbf{P}_1^n \dots \rightarrow \mathbf{P}_T^n$. Here, the starting frame \mathbf{P}_0^n is equal to the ground-truth frame \mathbf{P}^n , and the ending \mathbf{P}_T^n follows a standard Gaussian distribution: $\mathbf{P}_T^n \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Each random variable $\mathbf{P}_t^n \sim q(\mathbf{P}_t^n | \mathbf{P}_{t-1}^n) := \mathcal{N}(\mathbf{P}_t^n; \sqrt{1 - \beta_t} \mathbf{P}_{t-1}^n, \beta_t \mathbf{I})$ can be expressed in a closed form $\mathbf{P}_t^n \sim q(\mathbf{P}_t^n | \mathbf{P}^n)$ as follows:

$$\mathbf{P}_t^n = \sqrt{\bar{\alpha}_t} \mathbf{P}^n + \sqrt{1 - \bar{\alpha}_t} \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (1)$$

Algorithm 1 Training

Input dataset $\{\mathbf{P}_s^0, \mathbf{P}_s^1, \mathbf{P}_s^1\}_{s=1}^S$ of consecutive frame triplets, maximum diffusion step T , noise schedule $\{\beta_t\}_{t=1}^T$
repeat
 Sample $\mathbf{P}^0, \mathbf{P}^1, n, \mathbf{P}^n, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 Sample $t \sim \text{Uniform}(\{1, \dots, T\})$
 Add noise $\mathbf{P}_t^n = \sqrt{\bar{\alpha}_t} \mathbf{P}^n + \sqrt{1 - \bar{\alpha}_t} \epsilon$
 Interpolate $\hat{\mathbf{P}}^n = f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n)$
 Optimize loss $\mathcal{L}_{intp} = \text{loss}(\hat{\mathbf{P}}^n, \mathbf{P}^n)$
until converged

Algorithm 2 Inference

Input original frames $\mathbf{P}^0, \mathbf{P}^1$, maximum diffusion step T , noise schedule $\{\beta_t\}_{t=1}^T$
Sample $n, \mathbf{P}_T^n \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
 $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$
 else $\mathbf{z} = \mathbf{0}$
 Interpolate
 $\mathbf{P}_{t-1}^n = \tilde{\mu}_t(\mathbf{P}_t^n, f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n)) + \mathbf{z}$
end for
Return: $\hat{\mathbf{P}}^n$ as the interpolated frame

where $\bar{\alpha}_t = \prod_{s=0}^t \alpha_s = \prod_{s=0}^t (1 - \beta_s)$ denotes the diffusion coefficients, and β_s represents the noise coefficient determined by a cosine schedule [27].

Reverse Interpolation Denoising Process. Given a forward Markov chain $\mathbf{P}^n = \mathbf{P}_0^n \rightarrow \mathbf{P}_1^n \cdots \rightarrow \mathbf{P}_T^n$ produced by the forward diffusion process, the objective of the reverse denoising process is to learn a denoiser for progressively refining the noisy point cloud frame \mathbf{P}_T^n into the clean one, forming a reverse Markov chain $\mathbf{P}_T^n \rightarrow \cdots \rightarrow \mathbf{P}_1^n \rightarrow \mathbf{P}_0^n = \mathbf{P}^n$. Formally, our denoiser can be formulated as a conditional parameterized normal distribution as follows:

$$p_\theta(\mathbf{P}_{t-1}^n | \mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1) = \mathcal{N}(\mathbf{P}_{t-1}^n; \mu_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n), \tilde{\beta}_t \mathbf{I}), \quad (2)$$

where $\tilde{\beta}_t := \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t$. The frames $\mathbf{P}^0, \mathbf{P}^1$ and the timestamp n represent the conditions for guiding our reverse denoising. $\mu_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n)$ indicates the parameterized mean of the normal distribution. In order to effectively train our denoiser, we derive an interpolation-specific variational lower bound as the optimization objective:

$$\begin{aligned} \mathbb{E} [\log p_\theta(\mathbf{P}^n | \mathbf{P}^0, \mathbf{P}^1)] &\geq \mathbb{E} \left[\log \left(\frac{p_\theta(\mathbf{P}_{0:T}^n | \mathbf{P}^0, \mathbf{P}^1)}{q(\mathbf{P}_{1:T}^n | \mathbf{P}^n)} \right) \right] \\ &\approx -\mathbb{E} \left[\sum_2^T \text{D}_{KL}(q(\mathbf{P}_{t-1}^n | \mathbf{P}_t^n, \mathbf{P}^n) || p_\theta(\mathbf{P}_{t-1}^n | \mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1)) \right. \\ &\quad \left. - \log p_\theta(\mathbf{P}^n | \mathbf{P}_1^n) \right]. \end{aligned} \quad (3)$$

Please refer to Appendix A.1. for a detailed derivation. Then, we can supervise the parameter optimization of our denoiser by minimizing the Kullback-Leibler (KL) divergence in Eq. 3. In detail, as formulated in [7, 36], the posterior distribution can be further represented as

$$q(\mathbf{P}_{t-1}^n | \mathbf{P}_t^n, \mathbf{P}^n) := \mathcal{N} \left(\mathbf{P}_{t-1}^n; \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} \mathbf{P}^n + \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{P}_t^n, \tilde{\beta}_t \mathbf{I} \right). \quad (4)$$

This inspires us to reformulate the mean of the prior denoising distribution in Eq. 2 as $\mu_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n) :=$

$$\frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n) + \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{P}_t^n. \quad (5)$$

Consequently, minimizing the KL divergence is equivalent to optimizing a denoising network $f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n)$ to approximate the ground-truth intermediate frame \mathbf{P}^n . The training loss can be written as:

$$\mathcal{L}_{intp} = \text{loss}(\hat{\mathbf{P}}^n, \mathbf{P}^n) = \text{loss}(f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n), \mathbf{P}^n), \quad (6)$$

where we employ the Chamfer distance as the loss function (see Sec. 3.3). In the following, we will elaborate on the network architecture of our denoising network for robust intermediate frame refinement.

3.2. Full-Scale Motion-aware Dual-Branch Denoising Network

After formulating our diffusion interpolation model, we now introduce how to design our denoising model $f_\theta(\mathbf{P}_t^n, \mathbf{P}^0, \mathbf{P}^1, n)$. Following [22, 47], this denoising model leverages estimated scene motion (i.e., scene flow from frame \mathbf{P}^0 to \mathbf{P}^1) to explicitly infer the intermediate frame \mathbf{P}^n . To achieve robust 3D interpolation, we introduce two key architectural innovations: **(i)** A dual-branch full-scale motion estimator, sufficiently fusing diverse motion information to estimate both uneven and large scene motions; **(ii)** An adaptive selection module, which dynamically refines motion estimation by leveraging interpolation time.

3.2.1. Dual Branch Full-scale Motion Estimator

The motion estimator aims to estimate the scene flow from frame \mathbf{P}^0 to frame \mathbf{P}^1 . While the backbone of the motion estimator bears resemblance to DiffSF [51], there are two key distinctions tailored for the point cloud frame interpolation task. One is that the full-scale encoder is designed for full-scale geometric feature encoding to cope with the diverse motion patterns (e.g., uneven or large motions) in real-world scenes. The other is that we have devised a dual-branch motion estimation module for robust flow estimation, which tackles the inaccuracy issue of unidirectional

flow motion estimation methods such as DiffSF [51]. The detailed elaboration is provided below.

Full-scale Geometric Feature Encoding. In our full-scale feature extraction module, as illustrated in Fig. 2, we take DGCNN [39] as the basic backbone to extract point-wise geometric features, where a learnable graph is iteratively updated over the neighborhood to facilitate contextual geometric feature aggregation. The early-stage features in DGCNN capture local geometric details essential for estimating small motions, while the later-stage features encode global information, enabling the identification of larger dynamic changes across point cloud frames. While effective, this dynamically updated neighborhood graph may cause neighborhood shifts across layers, leading to feature misalignment and potential scene information loss (as early local features may be overlooked). To address this issue, we adopt a dense cascading strategy, termed full-scale encoding, aimed at preserving multi-scale geometric cues by progressively aggregating features across DGCNN encoding layers. This process can be formulated as below:

$$\mathbf{F}_l^0 = H_l([\mathbf{P}^0 + \mathbf{P}_t^n, \mathbf{F}_1^0, \dots, \mathbf{F}_{l-1}^0]), \quad (7)$$

$$\mathbf{F}_l^1 = H_l([\mathbf{P}^1 + \mathbf{P}_t^n, \mathbf{F}_1^1, \dots, \mathbf{F}_{l-1}^1]), \quad (8)$$

where \mathbf{F}_l^0 and \mathbf{F}_l^1 denote the geometric features for frames \mathbf{P}^0 and \mathbf{P}^1 in the l -th EdgeConv layer in DGCNN; $H_l(\cdot)$ is a non-linear transformation layer that combines EdgeConv, convolution, and max-pooling layers. During this process, each EdgeConv encoding layer integrates both dynamically updated EdgeConv features and outputs from previous layers, ensuring full-scale feature fusion for diverse motion estimation. We treat the feature outputs at the last iteration layer as the final full-scale geometric representations of frames \mathbf{P}^0 and \mathbf{P}^1 , which are denoted as $\mathbf{F}^0 \in \mathbb{R}^{C \times N}$ and $\mathbf{F}^1 \in \mathbb{R}^{C \times M}$, respectively.

Dual-branch Motion Estimation. With the extracted full-scale features \mathbf{F}^0 and \mathbf{F}^1 above, we then target developing an effective dual-branch motion estimator that combines both forward estimation and backward estimation for robust scene-flow prediction.

Specifically, in the forward estimation branch, we first introduce a local-global attention module that captures both local geometric information and global contextual relationships to further enhance the full-scale features achieved above by establishing long-range geometric dependencies. This can be formulated as:

$$\{\tilde{\mathbf{F}}^0, \tilde{\mathbf{F}}^1\} = CA(SA(LA(\mathbf{P}^0, \mathbf{P}^1, \mathbf{F}^0, \mathbf{F}^1))). \quad (9)$$

Here, $LA(\cdot)$ denotes a local Point Transformer [53]; $SA(\cdot)$ and $CA(\cdot)$ represent the self-attention and cross-attention mechanisms [33, 37], respectively. Please refer to the supplementary materials A.3. for more details. Then, inspired by [50], we compute a similarity matrix $\mathbf{S}^0 =$

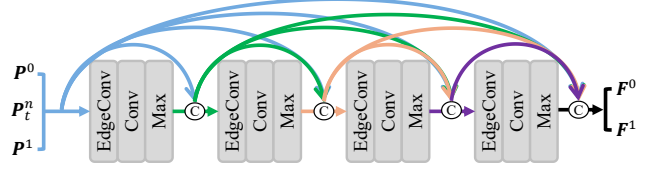


Figure 2. The illustration of Full-scale encoder. The layer that combines EdgeConv [39], Conv, and Max integrates information from local neighborhood points. EdgeConv enables each layer to dynamically update the local graph structure established by the k -nearest neighbor (k -NN). To address the issue of feature loss resulting from these dynamic updates, we stack four layers and employ dense cascading techniques [9] to enable feature reuse and extract full-scale fused features.

$\text{softmax}(\tilde{\mathbf{F}}^0 \cdot \tilde{\mathbf{F}}^1 / d) \in \mathbb{R}^{N \times M}$ between $\tilde{\mathbf{F}}^0$ and $\tilde{\mathbf{F}}^1$ for estimating the ‘soft’ correspondences of frame \mathbf{P}^0 by performing the weighted sum over frame \mathbf{P}^1 , that is $\tilde{\mathbf{P}}^1 = \mathbf{S}^0 \mathbf{P}^1 \in \mathbb{R}^{N \times 3}$. Consequently, the forward scene flow from frame \mathbf{P}^0 to frame \mathbf{P}^1 can be estimated by computing the coordinate offsets between their correspondences as follows:

$$\mathbf{M}_{for} = \mathbf{S}^0 \cdot \mathbf{P}^1 - \mathbf{P}^0 = \tilde{\mathbf{P}}^1 - \mathbf{P}^0 \in \mathbb{R}^{N \times 3}. \quad (10)$$

Analogously, the backward branch computes the soft correspondences for frame \mathbf{P}^1 , and also computes the correspondence offset as the scene motion from frame \mathbf{P}^1 to \mathbf{P}^0 :

$$\mathbf{M}_{back} = \mathbf{S}^1 \cdot \mathbf{P}^0 - \mathbf{P}^1 = \tilde{\mathbf{P}}^0 - \mathbf{P}^1 \in \mathbb{R}^{M \times 3}. \quad (11)$$

3.2.2. Adaptive Selection and Point Fusion

Based on the estimated forward and backward motions \mathbf{M}_{for} and \mathbf{M}_{back} above, we then employ them to warp the preceding and subsequent frames $\mathbf{P}^0, \mathbf{P}^1$ to time n , thereby interpolating four point cloud frames at time n as follows:

$$\begin{aligned} \mathbf{P}_{for}^{0 \rightarrow n} &= \mathbf{P}^0 + n\mathbf{M}_{for}, \\ \mathbf{P}_{for}^{1 \rightarrow n} &= \mathbf{P}^1 + (1-n)\hat{\mathbf{M}}_{for}, \\ \mathbf{P}_{back}^{0 \rightarrow n} &= \mathbf{P}^0 + n\hat{\mathbf{M}}_{back}, \\ \mathbf{P}_{back}^{1 \rightarrow n} &= \mathbf{P}^1 + (1-n)\mathbf{M}_{back}. \end{aligned} \quad (12)$$

Here, $\hat{\mathbf{M}}_{back}$ denotes the correspondence-aligned backward motion, obtained by reordering \mathbf{M}_{back} according to feature nearest neighbor correspondences between \mathbf{P}^0 and \mathbf{P}^1 , while $\hat{\mathbf{M}}_{for}$ is similar. Considering that each interpolated frame inevitably includes interpolation errors, we employ Point Fusion to comprehensively integrate these interpolations, ensuring error balance and robust interpolation. Since the point fusion module employs a fusion method based on k -NN clustering, as performed in [22], we first select a center frame and a remedial frame from the aforementioned

four point cloud frames, taking into account the interpolation time n . Specifically, the contributions of forward and backward interpolated frames to the intermediate frame are not always equal. When the interpolation time n exceeds $\frac{1}{2}$, $\mathbf{P}_{back}^{1 \rightarrow n}$ is designated as the central frame, while $\mathbf{P}_{for}^{1 \rightarrow n}$ serves as the corresponding remedial frame. Conversely, $\mathbf{P}_{for}^{0 \rightarrow n}$ and $\mathbf{P}_{back}^{0 \rightarrow n}$ are associated with the latter situation. After applying k -NN clustering, the resulting point features are fed into an MLP followed by a softmax layer to compute a weighted sum, ultimately yielding the predicted intermediate frame. Please refer to Appendix A.4. for more details.

3.3. Training objective

The objective of the reverse denoising process in our training comprises two key components:

$$\mathcal{L}_{intp} = \text{loss}(\hat{\mathbf{P}}^n, \mathbf{P}^n) = \text{loss}_{intp} + \text{loss}_1 + \text{loss}_2, \quad (13)$$

One is the interpolation loss, calculating the Chamfer Distance (CD) between the predicted point cloud frame $\hat{\mathbf{P}}^n$ and the ground truth \mathbf{P}^n at time n as follows:

$$\begin{aligned} \text{loss}_{intp} &= CD(\hat{\mathbf{P}}^n, \mathbf{P}^n) \\ &= \frac{1}{N} \sum_{\hat{x}^i \in \hat{\mathbf{P}}^n} \min_{x^j \in \mathbf{P}^n} \|\hat{x}^i - x^j\|_2 + \\ &\quad \frac{1}{N} \sum_{x^j \in \mathbf{P}^n} \min_{\hat{x}^i \in \hat{\mathbf{P}}^n} \|\hat{x}^i - x^j\|_2, \end{aligned} \quad (14)$$

The other is that, to effectively train our proposed dual-branch full-scale motion estimator and significantly enhance the performance of our algorithm, we introduce two dual-branch regularization terms loss_1 , loss_2 as self-supervised signals:

$$\begin{aligned} \text{loss}_1 &= CD(\mathbf{P}_{for}^{0 \rightarrow n}, \mathbf{P}^n) + \gamma * CD(\mathbf{P}_{for}^{1 \rightarrow n}, \mathbf{P}^n), \\ \text{loss}_2 &= CD(\mathbf{P}_{back}^{1 \rightarrow n}, \mathbf{P}^n) + \gamma * CD(\mathbf{P}_{for}^{0 \rightarrow n}, \mathbf{P}^n). \end{aligned} \quad (15)$$

Here, we set $\gamma = 0.1$. Our empirical findings indicate that configuring γ within (0.1-0.01) can yield comparable performance.

4. Experiments

4.1. Datasets

The proposed DiffPCI is evaluated on three large outdoor LiDAR datasets: KITTI odometry [6], Argoverse 2 sensor [40], and Nuscenes [2]. We adopt the same training and testing set splits as NeuralPCI [54] and FastPCI [47]. In particular, the frequency of NuScenes data is significantly decreased from 20Hz to 10Hz, which greatly increases the distance between point clouds (i.e. large motion) for 3D point cloud frame interpolation. Please refer to Appendix B. for more details about datasets. The input temporal resolution is 2.5Hz. During inference, three intermediate frames are interpolated for every two input frames.

4.2. Evaluation metrics

Following [22, 47, 54], we employ two general metrics, CD and Earth Mover’s Distance (EMD), as quantitative metrics to evaluate the similarity and consistency between the generated point clouds and the ground truth ones.

4.3. Implementation details

All experiments of three training sets using the same parameters are conducted on a single NVIDIA GeForce RTX 3090 GPU. We employ PyTorch [32] to implement our framework. We use the Adam optimizer [17] with a weight decay of 10^{-4} . Our initial learning rate is 10^{-4} . We adjust the learning rate using the PyTorch OneCycleLR scheduler. The number of forward diffusion steps T is set to 20. For the reverse denoising process, the number of steps during training is set to 20, while that in inference is set to 3. DiffPCI is trained for 600k iterations with a batch size of 2. We adopt the methodology outlined in [51] by randomly sampling 4096 points from the point cloud frames of the training set for training purposes and 8192 points from the test set for inference.

4.4. Comparison with state-of-the-art methods

To comprehensively demonstrate the superiority of our method, we show comparison results with the previous SOTA methods, namely PointNet [22], NeuralPCI [54] and FastPCI [47]. In addition, to illustrate the advantages of DiffPCI in the context of the diffusion model, we provide a thorough comparison with our baseline scene flow methods, GMSF [50] and DiffSF [51]. The interpolation results for the latter are derived through direct linear interpolation of the scene flow. The results present in this study utilize the same dataset splitting and preprocessing techniques as those described in [47].

Evaluation on Nuscenes. Rows 3-8 in Table 1 display the quantitative results for Nuscenes. DiffPCI achieves the best performance on all frames and in terms of all metrics, and most metrics are leading by a wide margin. Our self-supervised algorithm achieves 0.71 and 25.5 reductions over PointNet on average indicator CD and EMD for the same dual branch type, respectively. The qualitative experiment presents the frame interpolation visualization results of the proposed DiffPCI and other SOTA methods on the Nuscenes dataset, as shown in line (a) of Fig. 3. Our DiffPCI yields the best visual quality, the yellow box and its enlarged content clearly indicate that, compared with other methods, our approach yields human bodies with more distinct outlines.

Evaluation on Argoverse 2 sensor. The quantitative comparison results are shown in rows 9-14 of Table 1, where DiffPCI achieves the best performance on average metrics. Our method is slightly worse than the FastPCI [47] method on the two metrics in Frame-2, but exceeds the FastPCI

Table 1. **Quantitative comparison with the state-of-the-art on Nuscenes, Argoverse 2 sensor, and KITTI odometry.** The best result and the second best are **boldfaced** and underlined respectively. *Frame-1*, *Frame-2* and *Frame-3* refer to the three uniformly intermediate frames to be interpolated between the two input frames. Average denotes the average result on these three frames.

Dataset	Methods	Type	Frame-1		Frame-2		Frame-3		Average	
			CD	EMD	CD	EMD	CD	EMD	CD↓	EMD↓
Nuscenes	GMSF [50]	forward branch	1.12	178.75	1.17	227.07	1.60	286.11	1.30	230.64
		backward branch	3.41	294.34	2.41	233.88	1.62	171.79	2.48	233.34
	DiffSF [51]	forward branch	<u>0.96</u>	165.85	<u>0.90</u>	<u>190.39</u>	1.23	232.67	<u>1.03</u>	196.30
	PointNet [22]	dual branch	1.48	183.03	1.67	202.10	1.50	186.51	1.55	190.54
	NeuralPCI [54]	neural field	1.00	163.10	1.37	205.24	1.06	168.98	1.15	179.11
	FastPCI [47]	single branch	1.02	<u>162.78</u>	1.28	205.75	<u>1.03</u>	<u>152.39</u>	1.11	<u>173.64</u>
	DiffPCI(our)	dual branch	0.86	157.85	0.89	185.18	0.78	152.08	0.84	165.04
Argoverse 2 sensor	GMSF [50]	forward branch	0.69	64.12	0.89	77.62	1.21	92.41	0.93	78.05
		backward branch	1.82	98.37	1.33	81.34	0.91	63.96	1.35	81.22
	DiffSF [51]	forward branch	0.68	57.86	0.88	67.82	1.19	79.81	0.91	68.50
	PointNet [22]	dual branch	0.83	57.89	1.25	67.73	1.06	62.97	1.05	62.86
	NeuralPCI [54]	neural field	0.68	55.03	0.88	65.93	<u>0.69</u>	55.30	0.75	58.75
	FastPCI [47]	single branch	<u>0.68</u>	<u>54.99</u>	0.77	64.29	0.74	<u>54.59</u>	<u>0.73</u>	<u>57.95</u>
	DiffPCI(our)	dual branch	0.60	52.43	<u>0.79</u>	<u>64.45</u>	0.60	53.48	0.67	56.79
KITTI odometry	GMSF [50]	forward branch	0.55	65.76	0.61	77.46	0.86	94.21	0.67	79.14
		backward branch	1.56	102.60	1.04	83.16	0.71	64.25	1.10	83.34
	DiffSF [51]	forward branch	<u>0.49</u>	61.36	0.56	71.26	0.82	86.22	0.63	72.95
	PointNet [22]	dual branch	0.72	55.25	0.82	77.87	0.83	73.74	0.79	69.19
	NeuralPCI [54]	neural field	0.64	52.61	0.85	62.73	0.64	52.15	0.71	55.83
	FastPCI [47]	single branch	0.54	51.23	0.61	<u>59.84</u>	<u>0.58</u>	<u>50.27</u>	<u>0.58</u>	<u>53.78</u>
	DiffPCI(our)	dual branch	0.47	<u>51.46</u>	<u>0.59</u>	59.47	0.45	50.09	0.50	53.67

method by 1.16 and 0.06 on the average EMD and CD metric. Fig. 3 (b) shows the qualitative experimental results of interpolated frames on the Argoverse 2 sensor dataset. According to the blue circle, the results of the road and the boundary of our method are closer to the ground truth.

Evaluation on KITTI odometry. Rows 15-20 of Table 1 show the results on the KITTI odometry dataset, where DiffPCI achieves the best results on overall metrics. While DiffPCI performs marginally inferior to the FastPCI method [47] in terms of EMD metrics during Frame-1, it surpasses FastPCI on average CD and EMD metrics. The line (c) of Fig. 3 shows the visual comparisons among different methods on the KITTI odometry dataset. In the highlighted red close-up, DiffPCI demonstrates superior edge definition and minimal noise for the vehicle. Additionally, as depicted in the green circle, the walls generated by our method display significantly enhanced clarity.

4.5. Ablation study

The contributions of the essential components in DiffPCI, specifically the dual branch architecture, the full-scale encoder, the adaptive selection module, and the regularization loss, are presented in Table 2.

Dual-branch architecture aims to address the issue of imbalanced information reception between front and back frames, and fully learn the static and dynamic information

of the historical and future frames. The temporal arrangement of the dual branch adheres to the cyclic consistency observed in the point cloud sequence. We show the frame interpolation results of diffuse PCI with only one branch (forward branch) on the three data sets in rows 3, 9, and 15 of Table 2. The absence of the dual-branch architecture causes significant performance degradation.

Full-scale encoder is the most critical component for encoding local and global information from before and after frames. The extraction of global geometry information is improved through the multi-layer stacking and dense connection of the local information extraction layer. To verify the effectiveness of our proposed full-scale encoder, we simply replace the encoder with DGCNN, and the results are shown in rows 4, 10, and 16 of Table 2. It can be observed that the frame insertion performance after replacing DGCNN gets the second worst result on all three data sets.

Adaptive selection module is designed to fully leverage the motion obtained by the dual branches, thereby augmenting the acquisition of global information. We perform a comparative experiment with the proposed DiffPCI by removing two remediation frames. As shown in table 2, the CD metric is similar to DiffPCI, whereas the EMD is poorer. This indicates that the two remedial frames within the adaptive selection module enhance the reconstruction quality of global distribution to a certain degree.

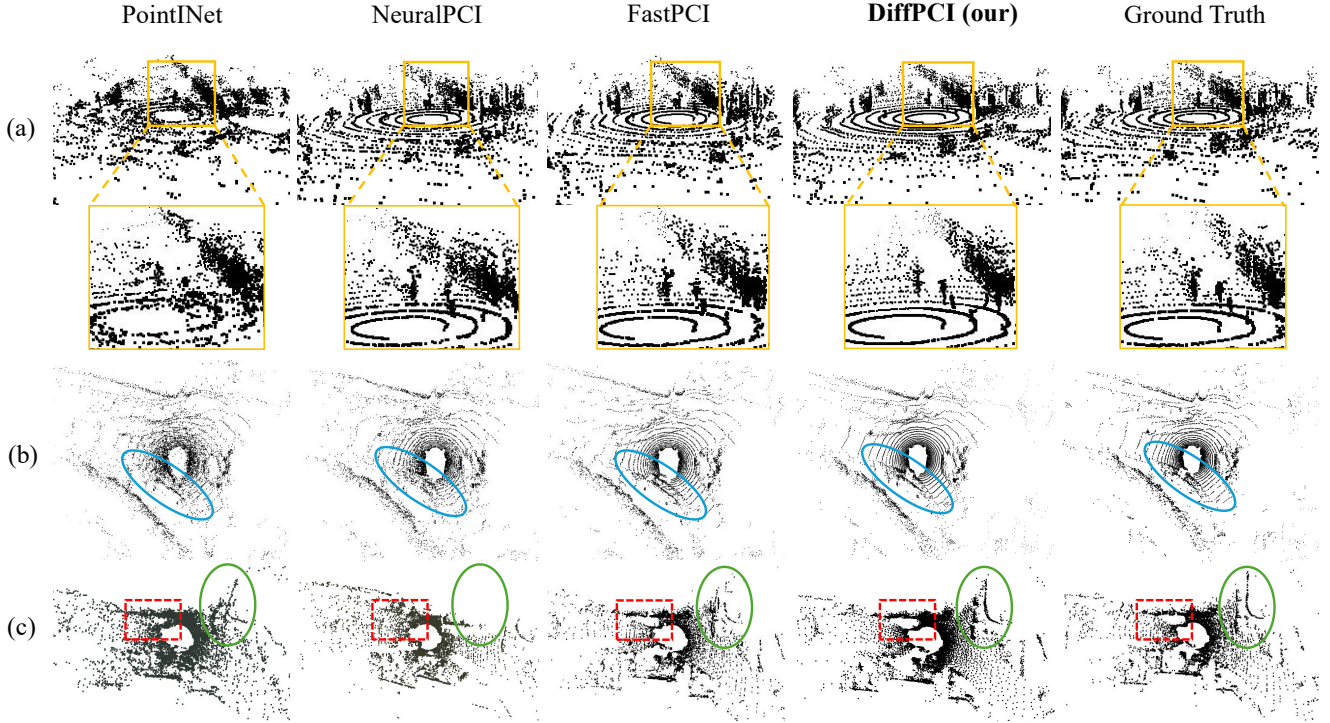


Figure 3. **Qualitative comparisons with the state-of-the-art on Nuscenes, Argoverse 2 sensor, and KITTI odometry dataset.** Rows (a)-(c) represent the results on three datasets, respectively. Each column represents a different method. Our DiffPCI (4th column) yields the best qualitative results compared to the state-of-the-art PointNet (1st column), NeuralPCI (2nd column) and FastPCI (3rd column). The enlarged content within the yellow box represents three persons, the blue circle highlights a protruding obstacle along the curve, the red box denotes a vehicle, and the green circle indicates a roadblock reflection. Our DiffSF interpolation results are notably closer to the ground truth, providing clearer outlines of individuals and vehicles and accurately interpolating roadblocks encountered during driving.

Table 2. **Ablation studies on all three datasets.** We show that our dual branch structure, the full-scale encoder, the adaptive selection module, and the regularization term improve the performance of all datasets. The best results are in **bold**.

Dataset	Models	Metrics	
		CD↓	EMD↓
Nuscenes	single branch	1.12	178.15
	encoder → DGCNN	0.90	168.55
	w/o adaptive selection	0.86	171.20
	w/o loss ₁	0.87	167.27
	w/o loss ₂	0.85	165.98
	DiffPCI(ours)	0.84	165.04
Argoverse 2 sensor	single branch	0.83	61.47
	encoder → DGCNN	0.78	64.59
	w/o adaptive selection	0.68	62.08
	w/o loss ₁	0.69	58.43
	w/o loss ₂	0.68	57.87
	DiffPCI(ours)	0.67	56.79
KITTI odometry	single branch	0.58	57.24
	encoder → DGCNN	0.57	62.17
	w/o adaptive selection	0.53	59.51
	w/o loss ₁	0.52	55.35
	w/o loss ₂	0.52	54.46
	DiffPCI(ours)	0.50	53.67

Regularization losses are closely related to the effect of dual branch full-scale motion estimation. We leverage these two losses to provide a self-supervised signal for estimating the dual branch motion. The results without those are shown in Table 2 of lines 6-7, 11-12, and 17-18. With these two regularization losses, the final frame interpolation performance of DiffPCI is further enhanced.

5. Conclusion

In this work, we introduced a novel point cloud interpolation framework that incorporates a diffusion model, which iteratively refines interpolation results through a step-by-step denoising process. Our framework is highlighted by a dual-branch full-scale motion estimator and an adaptive selection module integrated into the reverse denoising network. It is meticulously designed for precise point cloud frame interpolation in scenarios involving non-uniform motion, including large motion. Extensive experiments show that DiffPCI performs state-of-the-art results on three autonomous driving datasets. Despite its advancements, further point cloud interpolation research on the robustness of occluded scenarios and the generalization of unseen scenarios is needed in the future.

Acknowledgements

This work was supported by the National Key R&D Program of China No. 2024YFC3015801 and National Science Fund of China under Grant Nos. 62361166670, U24A20330 and 62276144.

References

- [1] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *CVPR*, 2019. 2
- [2] 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 *CVPR*, page 11621–11631, 2020. 6
- [3] Duolikun Danier, Fan Zhang, and David Bull. Ldmvfi: Video frame interpolation with latent diffusion models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(2):1472–1480, 2024. 1, 2
- [4] Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7346–7356, 2023. 2
- [5] Daniel Garrido, Rui Rodrigues, A. Augusto Sousa, Joao Jacob, and Daniel Castro Silva. Point cloud interaction and manipulation in virtual reality. In *2021 5th International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, page 15–20, 2021. 1
- [6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, page 3354–3361, 2012. 6
- [7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020. 2, 4
- [8] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, pages 8633–8646. Curran Associates, Inc., 2022. 2
- [9] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5
- [10] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, 2022. 2
- [11] Siddhant Jain, Daniel Watson, Eric Tabellion, Aleksander Holynski, Ben Poole, and Janne Kontkanen. Video interpolation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7341–7351, 2024. 1, 2
- [12] Chaokang Jiang, Dalong Du, Jiuming Liu, Siting Zhu, Zhenqiang Liu, Zhuang Ma, Zhujin Liang, and Jie Zhou. Neurogauss4d-pci: 4d neural fields and gaussian deformation fields for point cloud interpolation. *arXiv preprint arXiv:2405.14241*, 2024. 1, 2
- [13] Haobo Jiang, Zheng Dang, Shuo Gu, Jin Xie, Mathieu Salzmann, and Jian Yang. Center-based decoupled point-cloud registration for 6d object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3427–3437, 2023. 1
- [14] Haobo Jiang, Zheng Dang, Zhen Wei, Jin Xie, Jian Yang, and Mathieu Salzmann. Robust outlier rejection for 3d registration with variational bayes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1148–1157, 2023.
- [15] Haobo Jiang, Mathieu Salzmann, Zheng Dang, Jin Xie, and Jian Yang. Se (3) diffusion model-based point cloud registration for robust 6d object pose estimation. *Advances in Neural Information Processing Systems*, 36:21285–21297, 2023. 1
- [16] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. Flavr: Flow-agnostic video representations for fast frame interpolation. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, page 2070–2081, 2021. 2
- [17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6
- [18] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. Ifrnet: Intermediate feature refine network for efficient frame interpolation. In *CVPR*, 2022. 2
- [19] Haojie Liu, Kang Liao, Chunyu Lin, Yao Zhao, and Meiqin Liu. Plin: A network for pseudo-lidar point cloud interpolation. *Sensors*, 20(6):1573, 2020. 1
- [20] Haojie Liu, Kang Liao, Chunyu Lin, Yao Zhao, and Yulan Guo. Pseudo-lidar point cloud interpolation based on 3d motion representation and spatial supervision. *IEEE Transactions on Intelligent Transportation Systems*, 23(7): 6379–6389, 2021. 1
- [21] Jiuming Liu, Guangming Wang, Weicai Ye, Chaokang Jiang, Jinru Han, Zhe Liu, Guofeng Zhang, Dalong Du, and Hesheng Wang. Diffflow3d: Toward robust uncertainty-aware scene flow estimation with iterative diffusion-based refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15109–15119, 2024. 1, 2
- [22] Fan Lu, Guang Chen, Sanqing Qu, Zhijun Li, Yinlong Liu, and Alois Knoll. Pointinet: Point cloud frame interpolation network. In *AAAI*, page 2251–2259, 2021. 1, 2, 4, 5, 6, 7
- [23] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *CVPR*, 2022. 2
- [24] Calvin Luo. Understanding diffusion models: A unified perspective, 2022. 2
- [25] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2837–2845, 2021. 1, 2
- [26] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion

- forecasting with a single convolutional net. In *CVPR*, page 3569–3577, 2018. 1
- [27] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 4
- [28] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *CVPR*, 2018. 2
- [29] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *CVPR*, page 670–679, 2017.
- [30] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, page 261–270, 2017.
- [31] Junheum Park, Keunsoo Ko, Chul Lee, and Chang-Su Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. In *ECCV*, 2020. 2
- [32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32, 2019. 6
- [33] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 5
- [34] Zhihao Shi, Xiangyu Xu, Xiaohong Liu, Jun Chen, and Ming-Hsuan Yang. Video frame interpolation transformer. In *CVPR*, page 17482–17491, 2022. 2
- [35] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. Xvfi: extreme video frame interpolation. In *ICCV*, 2021. 2
- [36] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2256–2265, Lille, France, 2015. PMLR. 2, 4
- [37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 5
- [38] Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation. In *(NeurIPS) Advances in Neural Information Processing Systems*, 2022. 1
- [39] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019. 5
- [40] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021. 6
- [41] Guangyang Wu, Xin Tao, Changlin Li, Wenyi Wang, Xiaohong Liu, and Qingqing Zheng. Perception-oriented video frame interpolation via asymmetric blending. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2753–2762, 2024. 2
- [42] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 1
- [43] Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video probabilistic diffusion models in projected latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18456–18466, 2023. 2
- [44] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Advances in Neural Information Processing Systems*, pages 10021–10039. Curran Associates, Inc., 2022. 1, 2
- [45] 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 *CVPR*, 2022. 1, 2
- [46] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *CVPR*, page 5682–5692, 2023. 2
- [47] Tianyu Zhang, Guocheng Qian, Jin Xie, and Yang Jian. Fast-pci: Motion-structure guided fast point cloud frame interpolation. In *ECCV*, 2024. 1, 3, 4, 6, 7
- [48] Xuying Zhang, Yutong Liu, Yanguang Li, Renrui Zhang, Yufei Liu, Kai Wang, Wanli Ouyang, Zhiwei Xiong, Peng Gao, Qibin Hou, and Ming-Ming Cheng. Tar3d: Creating high-quality 3d assets via next-part prediction. *arXiv preprint arXiv:2412.16919*, 2024. 1
- [49] Xuying Zhang, Yupeng Zhou, Kai Wang, Yikai Wang, Zhen Li, Shaohui Jiao, Daquan Zhou, Qibin Hou, and Ming-Ming Cheng. Ar-1-to-3: Single image to consistent 3d object generation via next-view prediction. *arXiv preprint arXiv:2503.12929*, 2025. 2
- [50] Yushan Zhang, Johan Edstedt, Bastian Wandt, Per-Erik Forssen, Maria Magnusson, and Michael Felsberg. Gmsf: Global matching scene flow. In *Advances in Neural Information Processing Systems*, pages 64415–64427. Curran Associates, Inc., 2023. 5, 6, 7
- [51] Yushan Zhang, Bastian Wandt, Maria Magnusson, and Michael Felsberg. Diffsf: Diffusion models for scene flow estimation. In *NeurIPS*, 2024. 1, 2, 4, 5, 6, 7
- [52] Zihao Zhang, Lei Hu, Xiaoming Deng, and Shihong Xia. Sequential 3d human pose estimation using adaptive point cloud sampling strategy. In *IJCAI*, page 1330–1337, 2021. 1
- [53] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16259–16268, 2021. 5
- [54] Zehan Zheng, Danni Wu, Ruisi Lu, Fan Lu, Guang Chen, and Changjun Jiang. Neuralpci: Spatio-temporal neural field for 3d point cloud multi-frame non-linear interpolation. In *CVPR*, 2023. 1, 2, 6, 7

- [55] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5826–5835, 2021. [1](#), [2](#)