IDEA-Net: Dynamic 3D Point Cloud Interpolation via Deep Embedding Alignment

Yiming Zeng[1] Yue Qian[1] Qijian Zhang[1] Junhui Hou[1] Yixuan YUAN[1] Ying He[2]
[1] City University of Hong Kong [2] Nanyang Technological University
ym.zeng@my.cityu.edu.hk, jh.hou@cityu.edu.hk

Abstract

This paper investigates the problem of temporally interpolating dynamic 3D point clouds with large non-rigid deformation. We formulate the problem as estimation of point-wise trajectories (i.e., smooth curves) and further reason that temporal irregularity and under-sampling are two major challenges. To tackle the challenges, we propose IDEA-Net, an end-to-end deep learning framework, which disentangles the problem under the assistance of the explicitly learned temporal consistency. Specifically, we propose a temporal consistency learning module to align two consecutive point cloud frames point-wisely, based on which we can employ linear interpolation to obtain coarse trajectories/in-between frames. To compensate the high-order nonlinear components of trajectories, we apply aligned feature embeddings that encode local geometry properties to regress point-wise increments, which are combined with the coarse estimations. We demonstrate the effectiveness of our method on various point cloud sequences and observe large improvement over state-of-the-art methods both quantitatively and visually. Our framework can bring benefits to 3D motion data acquisition. The source code is publicly available at https://github.com/ZENGYIMING-EAMON/IDEA-Net.git.

1. Introduction

Dynamic 3D point clouds, which are sequences of 3D point cloud frames sampled in the temporal domain for capturing the changes in geometric details or motion of scenes/objects, have been widely used in many application scenarios, such as autopilot [22], immersive communication [6], computer animation [27], and virtual/augmented reality [40]. Despite of rapid development in 3D sensing technology [41], it is still difficult and costly to acquire 3D point cloud sequences with high temporal resolution (HTR), which hinders to finely represent deformable 3D objects [36]. Instead of relying on hardware development, we consider computational methods to construct an HTR point cloud sequence from one with low temporal resolution (LTR), as illustrated in Fig. 1.

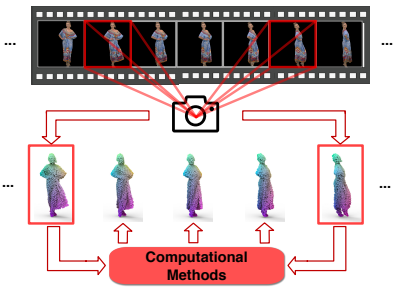


Figure 1. Illustration of the problem considered in this paper. One can adopt a low-cost 3D sensing device to sample the motion at a low frequency, leading to an LTR point cloud sequence, then apply the computational method to interpolate/estimate in-between point cloud frames to obtain an HTR one for finely representing the 3D motion of objects (or 3D shapes/objects deforming over time). We are interested in point cloud sequences with *massive non-rigid deformation*. Moreover, in real application scenario, the point cloud frames of a sequence are independently captured in the sensor space, thus lacking *point-wise temporal consistency*.

Although the considered problem shares similar properties with 2D video frame interpolation, both of which aim to interpolate/predict the in-between frames of any two consecutive frames of an LTR sequence, the essentially different data modality (i.e., illumination vs. geometry information) makes it non-trivial to extend existing 2D video frame interpolation methods [13,14,21] to 3D point clouds. Moreover, the unordered and irregular nature of 3D point cloud data in spatial and temporal domains poses great challenges.

Recently, several deep learning-based interpolation methods for 3D point cloud sequences have been proposed [12,19,29,31]. Nevertheless, for the flow-based PointINet [19], it is mainly applicable to shapes with nearly rigid transformation and cannot well generalize to those with large non-rigid deformation. For the auto-encoder-based methods like [29,31], which directly interpolate global features, since the global features are abstract and insufficient

[1]This work was supported by the HK RGC Grant CityU 11202320 and 11218121. Corresponding author: J. Hou.

(a) (b) (c)

Figure 2. Visual comparisons of (a) our IDEA-Net, (b) PointINet [19], and (c) the ground-truth on the *Swing* sequence.

to describe the details of motion changes, the interpolated frames tend to have similar shape appearances and lack temporal continuity, leading to stuck motion sequences. Besides, they are architecturally designed as separate learning stages, instead of fully end-to-end, which may suffer from severe information loss. Unlike existing works, we seek to build an interpretable interpolation framework with a clear geometric explanation. Moreover, in terms of application scenarios, we are interested in challenging dynamic point cloud data with large non-rigid deformation.

Technically, we formulate the problem as estimation of point-wise trajectories (i.e., smooth curves in 3D Euclidean space) and reason that the challenges are mainly posed by temporal irregularity and under-sampling, which motivates us to disentangle the problem, leading to a two-step learning process: *i)* coarse linear interpolation and *ii)* trajectory compensation. Based on the explicit formulation, we propose IDEA-Net, an end-to-end deep interpolation framework, which features a dual-branch structure and consists of three steps: *1)* extracting point-wise high-dimensional features, *2)* learning point-wise temporal consistency and deducing coarse trajectories/in-between frames via linear interpolation, and *3)* exploiting temporally regularized features to compensate the non-linear components of smooth trajectories. Experiments on both synthetic and real-scanned data demonstrate our IDEA-Net quantitatively and visually outperforms state-of-the-art methods to a large extent, as visualized in Fig. 2. We also conduct extensive ablation studies to validate the rationality of our design.

In summary, we make the following contributions:

- a new formulation for the problem of temporally interpolating dynamic 3D point cloud sequences;
- a symmetric and coarse-to-fine network for end-to-end reconstructing HTR point cloud sequences from LTR point cloud sequences with large non-rigid deformation.

2. Related Work

Deep 2D video frame interpolation aims to increase the frame rate of a video by generating the in-between frames. The existing methods can be generally classified into two categories: kernel-based and flow-based. The former [21, 24–26] generates the in-between frames by convolution over local patches directly. The latter [2, 13, 17, 18, 38] adopts the estimated flow to guide the warping process of the input frames. Different from 2D images/videos with the regular grid structure, 3D point clouds are characterized by both spatial and temporal irregularity, which impedes the direct extension of 2D video frame interpolation models.

Deep dynamic 3D point cloud processing. The key challenges of this task lie in the temporal irregularity and large deformation in the point cloud sequence. Existing techniques can be broadly classified into three types. (1) Voxelize a point cloud sequence into a 4D volumetric grid [5, 20, 23]. For example, FaF [20] uses 3D CNN to extract features. MinkowskiNet [5] analyzes the voxelized 4D tensor via a sparse 4D CNN. (2) Adopt some sequential modules to deal with temporal information. For example, Yang *et al.* [8] proposed PointRNN, PointGRU, and PointLSTM to model dynamic point clouds. (3) Directly perform sequence processing on raw points [9–11, 16, 19, 28, 32]. For example, [9, 10, 16, 32] perform feature aggregation by querying neighboring points in spatial and temporal domains. They have been applied to several tasks such as action recognition, pose estimation and segmentation. However, such aggregation is inaccurate, especially for point cloud sequences with large motions. To address this, PointINet [19] adopts a scene flow estimator to interpolate two point clouds. However, PointINet fails to interpolate shapes with large deformation, e.g., human shapes.

Deep 3D shape interpolation. Inspired by the deep learning-based methods for 3D point cloud processing [29, 30, 34, 37], a number of works adopting neural networks for 3D shape interpolation have been proposed [1, 7, 12, 19, 31, 39], which can be roughly divided into two categories. (1) Auto-Encoder (AE)-based methods. For example, [1, 12, 39] directly interpolate global features and feed the interpolated feature vector into the decoder to regress in-between point cloud frames. [31] further improves this kind of methods by introducing an edge AE trained with 3D meshes. However, interpolating global features without considering local region deformations may cause significant information loss. [7] adds normal information into the edgeConv [37] and calculates the geodesic matrices of remeshed 3D objects to explicitly constrain the learning of correspondence and shape interpolation for 3D meshes simultaneously. However, the required normals and topological information are unavailable in raw point clouds, making

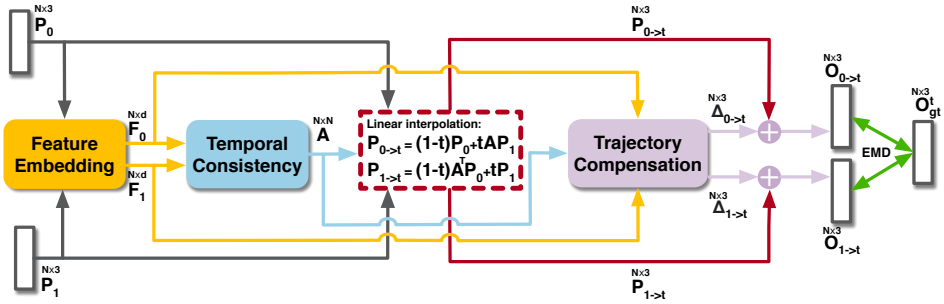


Figure 3. The flowchart of the proposed IDEA-Net for temporally interpolating any two consecutive frames of an LTR point cloud sequence in an end-to-end manner. Besides, the user can vary the parameter t in the range of $(0, 1)$ for interpolating frames continuously after training. We refer the readers to *Supplementary Material* for the detailed configuration of our network.

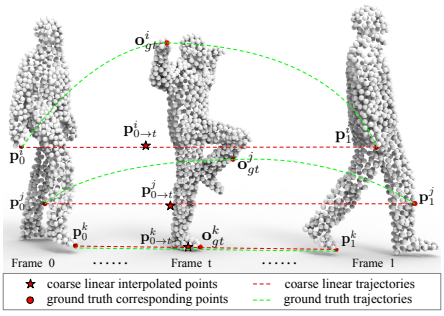


Figure 4. Illustration of the point-wise trajectories of typical points of a point cloud sequence, where the green and red dashed lines denote the ground-truth and linearly interpolated trajectories, respectively.

it non-trivial to extend the method to dynamic point cloud interpolation. Moreover, additional post-processing may be required to align the generated shapes or refine the correspondences [7, 12, 31]. (2) Flow-based methods. Similar to 2D image interpolation, [19] adopts the pre-trained flow estimation network for 3D point clouds, i.e., FlowNet3D [15], to generate the bi-directional 3D flow, which is then used to warp the input frames to generate in-between estimations. However, it cannot work well on data with large deformation due to the limitation of the adopted flow estimation.

3. Proposed Method

3.1. Problem Formulation

Without loss of generality, let $\mathbf{P}_0 \in \mathbb{R}^{N\times 3}$ and $\mathbf{P}_1 \in \mathbb{R}^{N\times 3}$ be any two consecutive frames of an LTR point cloud sequence each with N points[1], and $\mathbf{p}_0^i$ and $\mathbf{p}_1^j \in \mathbb{R}^{1\times 3}$ the i-th and j-th points of $\mathbf{P}_0$ and $\mathbf{P}_1$, respectively. Assume that each point of $\mathbf{P}_0$ could be aligned to that of $\mathbf{P}_1$, and let the matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ explicitly encode such point-wise temporal consistency, i.e., if $\mathbf{p}_0^i$ corresponds to $\mathbf{p}_1^j$, $a_{ij} = 1$; otherwise, $a_{ij} = 0$. Let $\mathbf{a}_i \in \mathbb{R}^{1\times N}$ denote the i-th row of $\mathbf{A}$. Note that $\mathbf{A}$ is *unknown*.

Obtaining an arbitrary point cloud frame between $\mathbf{P}_0$ and $\mathbf{P}_1$ is equivalent to estimating the trajectory within each pair of points $\left\{\mathbf{p}_0^i,\ \mathbf{a}_i\mathbf{P}_1\right\}_{i=1}^{N}$. Generally, the trajectory of each point is a smooth curve in 3D Euclidean space; moreover, the fluctuation of the curves corresponding different points varies due to the articulated structure, non-rigid deformation, and other factors, as illustrated in Fig. 4. However, directly estimating such a curve only with two endpoints may have high uncertainty. Thus, we disentangle this challenging problem and formulate it as a two-step coarse-to-fine process. Specifically, we first uniformly approximate all point-wise trajectories by linear curve fitting, and accordingly the coarse in-between frame at time $\forall t \in (0, 1)$ denoted as $\mathbf{P}_{0\to t} \in \mathbb{R}^{N\times 3}$ can be interpolated as

$$\mathbf{P}_{0\to t} = (1-t)\mathbf{P}_0 + t\mathbf{A}\mathbf{P}_1. \tag{1}$$

Although such a simple linear interpolation process is inaccurate, it is able to provide rational initialization to reduce ambiguity to some extent. Then, to further compensate the high-order nonlinear components of trajectories missed in Eq. (1) and correct the errors resulted from the inaccurate estimation of $\mathbf{A}$, we introduce a trajectory compensation process. Particularly, we can point-wisely map the input point clouds to a high-dimensional feature space by a typical nonlinear mapping function $\phi(\cdot) : \mathbb{R}^3 \to \mathbb{R}^d$ and then estimate a function $f(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ to fuse the aligned features, which are finally transformed back to the point cloud space by another nonlinear function $\psi(\cdot) : \mathbb{R}^d \to \mathbb{R}^3$ to obtain the increments $\mathbf{\Delta}_{0\to t} \in \mathbb{R}^{N\times 3}$ for trajectory compensation:

$$\mathbf{\Delta}_{0\to t} = \psi\left(f(\phi(\mathbf{P}_0), \phi(\mathbf{P}_1), \mathbf{A}, t)\right). \tag{2}$$

We expect that the high-order nonlinear components of the trajectories could be learned from the cues provided by the feature representation that can embed both local and global shape information of $\mathbf{P}_0$ and $\mathbf{P}_1$, as well as the contrast between the feature representations of them. The predicted in-between frame of an HTR sequence is finally obtained as

$$\mathbf{O}_{0\to t} = \mathbf{P}_{0\to t} + \mathbf{\Delta}_{0\to t}. \tag{3}$$

Moreover, because the transpose of the alignment matrix $\mathbf{A}^\mathsf{T} \in \mathbb{R}^{N\times N}$ also depicts the temporal consistency information from $\mathbf{P}_1$ to $\mathbf{P}_0$, the previous formulation can be

[1]Note that the points contained in a point cloud frame are randomly stacked to form a matrix.

equivalently written as

$$\mathbf{P}_{1\to t} = (1-t)\mathbf{A}^{\mathsf{T}}\mathbf{P}_0 + t\mathbf{P}_1, \tag{4}$$

$$\mathbf{\Delta}_{1\to t} = \psi\left(f(\phi(\mathbf{P}_0), \phi(\mathbf{P}_1), \mathbf{A}^{\mathsf{T}}, 1-t)\right), \tag{5}$$

$$\mathbf{O}_{1\to t} = \mathbf{P}_{1\to t} + \mathbf{\Delta}_{1\to t}. \tag{6}$$

Ideally, the two point clouds represented by $\mathbf{O}_{0\to t}$ and $\mathbf{O}_{1\to t}$ are the same. From Eqs. (1)-(6), it can be seen that the problems of dynamic point cloud interpolation mainly rely on the learning of the point-wise temporal consistency and the realization of the trajectory compensation process.

3.2. Overview of our Framework

Based on the above formulation, we propose an end-to-end deep learning-based framework dubbed IDEA-Net, a *dual-branch* network, which mimics the two equivalent coarse-to-fine processes. As shown in Fig. 3, our IDEA-Net comprises of three modules: feature representation, learning point-wise temporal consistency, and trajectory compensation. Specifically, the feature representation module first embeds 3D coordinates into a high-dimensional feature space by exploring both the local and global geometry of a point cloud, leading to the point-wise high-dimensional features. Taking the features as input, the temporal consistency module then predicts a relaxed matrix $\mathbf{A}$ with the alignment effect, which naturally induces the coarse interpolations via Eqs. (1) and (4). Finally, the trajectory compensation module regresses the aligned and interpolated high-dimensional features with the learned $\mathbf{A}$ to generate nonlinear increments in a residual learning manner for compensating the high-order components of trajectories. We empirically pick $\mathbf{O}_{0\to t}$ if $t < 0.5$ and $\mathbf{O}_{1\to t}$, otherwise as the final interpolated frame. In what follows, we will detail each module.

3.3. Hierarchical Feature Representation

We employ DGCNN [37] as our backbone to map 3D coordinates of input point clouds into a high-dimensional feature space, in which both local and global semantics are progressively embedded through the dynamic graph construction mechanism. Specifically, this module is composed of four layers of EdgeConv, which dynamically selects neighbours to aggregate local information to obtain point-wise features. Besides, a global feature that is formed by the adaptive max and average pooling for all point-wise features is concatenated to each local feature to get the final point-wise features, denoted as $\mathbf{F}_0 \in \mathbb{R}^{N\times d}$ and $\mathbf{F}_1 \in \mathbb{R}^{N\times d}$ for $\mathbf{P}_0$ and $\mathbf{P}_1$, respectively. Denote by $\mathbf{f}_0^i$ and $\mathbf{f}_1^j \in \mathbb{R}^{1\times d}$ the i-th and j-th rows of $\mathbf{F}_0$ and $\mathbf{F}_1$, which encode the high-dimensional features of the i-th and j-th points of $\mathbf{P}_0$ and $\mathbf{P}_1$, respectively. We refer the readers to [37] for more details of DGCNN.

3.4. Learning Point-Wise Temporal Consistency

As aforementioned, in reality, each frame of a point cloud sequence is captured individually in the camera space, resulting in temporal irregularity. Hence, we propose a temporal consistency module to explicitly align the pair of input point clouds in point-wise, i.e., learning the matrix $\mathbf{A}$. However, the matrix $\mathbf{A}$ is ideally a binary permutation matrix, making it impossible to directly optimize it in a deep learning framework. To overcome the challenge, we optimize a relaxed alignment matrix, i.e., $a_{ij} \geq 0$ and $\mathbf{a}_i\mathbf{1}^{\mathsf{T}} = 1$ with $\mathbf{1} \in \mathbb{R}^{1\times N}$ being the vector whose all entries are one. Note that this module is end-to-end optimized in the IDEA-Net framework without additional supervision.

Intuitively, a pair of aligned points should have similar semantic features. Motivated by this observation, we employ the distance between features to estimate $\mathbf{A}$. Specifically, we first compute $\widetilde{\mathbf{A}} \in \mathbb{R}^{N\times N}$ whose (i,j)-th entry $\widetilde{a}_{ij}$ is

$$\widetilde{a}_{ij} = 1/\|\mathbf{f}_0^i - \mathbf{f}_1^j\|_2, \tag{7}$$

where $\|\cdot\|_2$ returns the ℓ_2-norm of a vector. To further encourage $\mathbf{A}$ to mimic a binary matrix, we normalize the elements in each row

$$\widehat{a}_{ij} = (\widetilde{a}_{ij} - \mu_i)/\sigma_i, \tag{8}$$

where μ_i and σ_i are the mean and standard deviation of the i-th row of $\widetilde{\mathbf{A}}$. Finally, we apply a softmax operator on $\widehat{\mathbf{A}}$ row-wisely to fulfill the relaxed constraint, generating

$$a_{ij} = e^{\widehat{a}_{ij}} / \sum_{j=1}^{N} e^{\widehat{a}_{ij}}. \tag{9}$$

Remark. Due to the non-differentiable characteristic of strictly binary matrices, our learned matrix $\mathbf{A}$ under such a relaxation process is no longer expected to exactly indicate point-wise temporal consistency relationships. In fact, as revealed in previous studies [9], since points may flow in and out across frames, there may not exist a "ground truth" point-wise consistency in most cases. Hence, we can interpret that $\mathbf{A}$ is functionally generalized to achieve coarse matching at both point and feature levels, and further drives the subsequent refinement module. Besides, the errors caused by the inaccurate estimation of $\mathbf{A}$ may be fixed in the subsequent refinement step to some extent. See Section 4.4 for the detailed ablation study on this module.

3.5. Trajectory Compensation

With the learned $\mathbf{A}$ in Section 3.4, we can naturally obtain coarse interpolations via Eqs. (1) and (4). To simultaneously compensate the nonlinear components of trajectories and fix the interpolation errors caused by the inaccurate $\mathbf{A}$, we introduce the trajectory compensation module.

Specifically, as the feature embedding process in Section 3.3 can capture both the local and global geometric structures of the inputs, we employ it to realize the mapping

function $\phi(\cdot)$ in Eq. (2), i.e.

$$\mathbf{F}_0 = \phi(\mathbf{P}_0), \text{ and } \mathbf{F}_1 = \phi(\mathbf{P}_1). \tag{10}$$

Considering that the non-linear trajectories could be deemed linear after being projected into the high-dimensional feature space via the data-driven manner, we simply implement the fusion function $f(\cdot)$ in Eq. (2) with a linear function, i.e.,

$$\begin{aligned} \mathbf{F}_{0\to t} &= (1-t)\mathbf{F}_0 + t\mathbf{A}\mathbf{F}_1, \\ \mathbf{F}_{1\to t} &= (1-t)\mathbf{A}^{\mathsf{T}}\mathbf{F}_0 + t\mathbf{F}_1. \end{aligned} \tag{11}$$

Finally, we adopt a shared multi-layer perceptron (MLP) to implement the mapping $\psi(\cdot)$ for regressing the increments, i.e.,

$$\mathbf{\Delta}_{0\to t} = \psi(\mathbf{F}_{0\to t}), \text{ and } \mathbf{\Delta}_{1\to t} = \psi(\mathbf{F}_{1\to t}). \tag{12}$$

Remark. Directly regressing an in-between point cloud frame via Eqs. (2) and (5) seems to be a feasible solution. However, such an approach cannot yield satisfactory results (see the results in Section 4.4) due to the curse of dimensionality. This also validates the necessity and rationality of the coarse linear interpolation module in Fig. 3.

3.6. Loss Function

We train IDEA-Net end-to-end by simultaneously minimizing the earth mover's distance (EMD) [33] between the reconstructed point cloud from each branch and the ground-truth one $\mathbf{O}_{gt}^t$:

$$\mathcal{L} = \frac{1}{2}\left(\mathcal{L}_{EMD}\left(\mathbf{O}_{0\to t}, \mathbf{O}_{gt}^t\right) + \mathcal{L}_{EMD}\left(\mathbf{O}_{1\to t}, \mathbf{O}_{gt}^t\right)\right)$$

where $\mathcal{L}_{EMD}(\cdot,\cdot)$ computes the EMD between two point clouds.

Remark. Compared with the single-branch design, the proposed dual-branch design with shared network parameters reconstructs in-between frames from two directions, which is equivalent to regularizing $\mathbf{A}$ during training. Accordingly, the trained model can generate a more feasible $\mathbf{A}$ and like-wise better reconstruction quality during inference. Section 4.4 demonstrates the superiority of such a dual-branch design via extensive evaluation.

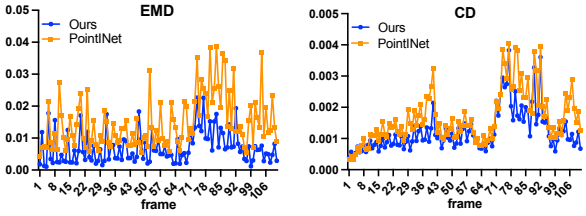
4. Experiments

4.1. Experiment Settings

Datasets. We first constructed a dataset named Dynamic Human Bodies dataset (DHB), containing 10 point cloud sequences from the MITAMA dataset [35] and 4 from the 8IVFB dataset[2]. The sequences in DHB record 3D human motions with large and non-rigid deformation in real world. Besides, we adopted the commonly-used DFAUST[3] dataset [3], a synthetic 3D human motion dataset used in Intrinsic Point Cloud Interpolation (PCI) [31] and PointNet AE [29], to fairly compare with them. For DHB, we used eight sequences to form the training set, and the remaining six sequences as the test set. For DFAUST, following [31] we used eleven action sequences to build the training dataset and three sequences for testing. We downsampled the acquired sequences in the temporal domain to generate input LTR point cloud sequences, i.e., we uniformly selected one frame every k_{train} frames during the training phase and k_{test} frames during the testing phase. See *Supplementary Material* for more details of the datasets.

[2]The MITAMA and 8IVFB datasets contain dynamic 3D meshes and real scanned point clouds, respectively. We uniformly sampled from each frame 1024 points.

[3]The DFAUST contains dynamic 3D meshes. Following the same setting as [31], we uniformly sampled from each frame 1000 points.



(a) (b)

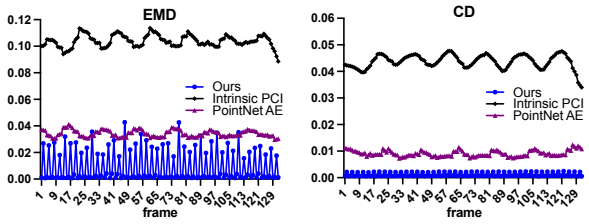
Figure 5. Comparisons of the frame-by-frame quality of the reconstructed in-between frames on *Swing* of the DHB dataset.

Compared methods. We compared the most recent work named PointINet [19], which is flow-based and designed for temporal interpolation of point cloud sequences collected by LiDAR. For a fair comparison, we retrained its official code with the same DHB dataset as ours. We also compared our method with two state-of-the-art AE-based methods, i.e., Intrinsic PCI [31] and PointNet AE [29]. We adopted their pre-trained models on the DFAUST dataset and trained our method with the same data as [31]. It is also worth noting that Intrinsic PCI [31] requires the edges of a template mesh as extra input during training. Besides, we followed the setting of [31], which adopts the ICP [4] as the post-processing, to align the interpolated frames from Intrinsic PCI [31] and PointNet AE [29] with the input frames. Note that our method does not need a template mesh and any post-processing.

Evaluation metrics. To quantitatively evaluate the interpolation quality, we provided both the average and the frame-by-frame Chamfer distance (CD) and EMD between the interpolated frames and the ground-truth ones of a sequence. Besides, we conducted subjective evaluation to compare different methods comprehensively. See Section 4.3 for details.

4.2. Experimental Results

Results on the DHB dataset. Table 1 shows the quantitative comparison with PointINet [19], where we set both k_{train} and k_{test} to 3 to generate LTR sequences for training and testing. As Table 1 shows, our method outperforms PointINet to a large extent under EMD metrics. The reason is that it is difficult for PointINet to explicitly predict accu-



(a) (b)

Figure 6. Comparisons of the frame-by-frame quality of the reconstructed in-between frames on *shake_arms* of the DFAUST dataset.

Table 1. Quantitative ($\times 10^{-3}$) comparison over the DHB dataset.

Methods	Swing		Longdress	
	EMD	CD	EMD	CD
PointINet [19]	15.03	1.70	10.09	0.95
Ours	**7.07**	**1.24**	**5.92**	**0.88**

Table 2. Quantitative ($\times 10^{-3}$) comparison over the DFAUST dataset.

Methods	shake_arms		shake_hips		shake_shoulders	
	EMD	CD	EMD	CD	EMD	CD
Intrinsic PCI [31]	103.85	43.74	52.93	22.04	64.70	28.55
PointNet AE [29]	34.23	9.09	40.85	12.35	29.18	11.99
Ours	**9.31**	**1.20**	**6.85**	**0.91**	**9.19**	**0.95**

rate flows on sequences with large deformation, while our method is free of this operation. Besides, as shown in Fig. 5, our method achieves lower EMD and CD for *most* frames and smaller fluctuations of frame-by-frame EMD and CD. Fig. 2 and Fig. 7 show the visual comparison, where it can be seen that our method can generate frames that are closer to ground-truth ones, whereas PointINet [19] tends to generate outliers and non-uniformly distributed points.

Results on the DFAUST dataset. Table 2 lists the quantitative comparison with Intrinsic PCI [31] and PointNet AE [29], where we set both k_{train} and k_{test} to 3 to generate LTR sequences for training and testing. From Table 2, it can be seen that our method significantly outperforms Intrinsic and PointNet AE. The reason is that Intrinsic and PointNet AE adopt separate learning stages to vaguely interpolate global features for generating in-between frames, resulting in severe loss of spatial information, while our method is end-to-end and geometrically meaningful. From Fig. 6, it can be observed that our approach achieves much lower EMD and CD over almost all frames than Intrinsic and PointNet AE. Fig. 8 provides visual demonstration of our method, from which we can see that these two AE-based methods fail to interpolate correct poses. Moreover, they cannot faithfully represent the original shapes.

Evaluation of the flexibility. To demonstrate the flexibility of our method, we trained a single network with data that were generated by setting k_{train} to 3 and then evaluated the network with data that were generated by setting various $k_{\text{test}} \in \{3, 5, 7, 9, 11\}$. We also trained PointINet [19] with

Table 3. Quantitative comparisons under different training strategies. (1)-(3): the methods were trained with data that were generated with $k_{\text{train}} = 3$; (4)-(5): the methods were trained with the mixed data training strategy introduced in Section 4.2; Except for (3) where the flow estimation module of PointINet was pre-trained and fixed, the other networks were end-to-end trained from scratch. The testing data were generated by setting $k_{\text{test}} = 3$.

	Swing		Longdress	
	EMD	CD	EMD	CD
(1) Ours	7.07	1.24	5.92	**0.88**
(2) PointINet [19]	15.03	1.70	10.09	0.95
(3) PointINet [19] (pretrained flow)	15.38	1.72	10.63	0.96
(4) Ours (mixed)	**6.74**	**1.21**	**5.84**	0.89
(5) PointINet [19] (mixed)	81.43	14.19	84.49	9.80

the same setting for comparison. As shown in Fig. 9, it can be seen that with the value of k_{test} increasing, the interpolation problem becomes more challenging, and thus the reconstruction errors of both our method and PointINet gradually increase. However, our method always outperforms PointINet to a large extent, and especially the advantage is more obvious for a relatively larger k_{test}, demonstrating its stronger ability.

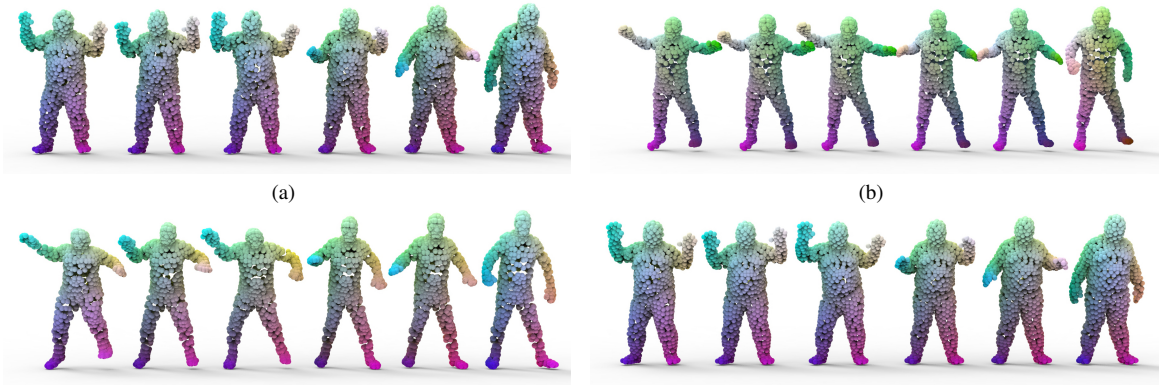
Evaluation on the mixed data training mechanism. In the previous experiments, we utilized all the ground-truth in-between frames for supervision. In this experiment, we set $k_{\text{train}} = 4$ and $k_{\text{test}} \in \{1, 3, 5, 7\}$ to generate training and testing data, respectively. During training, in each iteration we randomly selected only one of the in-between frames to-be-interpolated to optimize. Denote this training strategy as *mixed data training*. Such a training manner can speed up the training process and improve the robustness of the network, owing to the diversity of data in different iterations. As shown in Fig. 10, the performance of our method improves under this strategy, whereas PointINet [19] suffers from serious performance degradation (see Table 3). This observation also demonstrates the advantage of our design.

4.3. Subjective Evaluation

To conduct the subjective evaluation, we displayed the interpolated sequences by all methods and the corresponding ground-truth ones to 15 volunteers and asked them to vote for the method whose results they considered are closest to the ground-truth sequences. As shown in Fig. 11, our IDEA-Net gets the highest number of votes on all the testing sequences, especially compared with the PointINet in Fig. 11a. Besides, the evaluation results in Fig. 11b indicate Intrinsic and PointNet-AE also obtain good subjective evaluations since these two methods can generate shapes with uniformly distributed points and few outliers. However, as illustrated in Fig. 8, these global feature-based methods fail to generate the correct poses and cannot preserve the faithful shapes. We also refer the readers to the Github page for video demos.

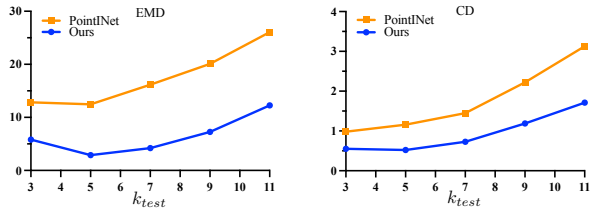
(a) (b) (c)

Figure 7. Visual comparison of the interpolated in-between frames by (a) Ours and (b) PointINet [19] on *Longdress* of the DHB dataset, and (c) the corresponding ground-truth frames. *Points of the interpolated frames by PointINet are non-uniformly distributed.*



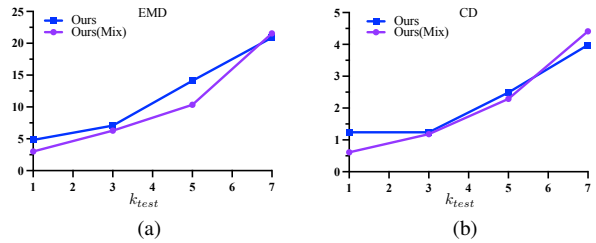
(a) (b) (c) (d)

Figure 8. Visual comparison of the interpolated in-between frames by (a) Ours, (b) Intrinsic PCI [31], and (c) PointNet AE [29] on the sequence of the DFAUST dataset, and (d) the corresponding ground-truth frames. *The overall shapes of the interpolated frames by Intrinsic and PointNet AE obviously deviate from ground-truth ones.*



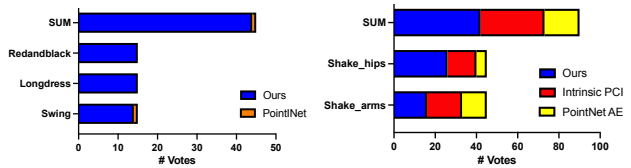
(a) (b)

Figure 9. Flexibility verification on *Squat_2* of the DHB dataset.



(a) (b)

Figure 10. Quantitative evaluation of our method trained with the mixed data training mechanism. The testing sequence is *Swing* of the DHB dataset.



(a) (b)

Figure 11. Subjective evaluation on the interpolated sequences by different methods on (a) DHB dataset and (b) DFAUST dataset. "SUM" refers to the sum of votes on all testing sequences for each dataset.

4.4. Ablation Study

To comprehensively and deeply understand the effects of the core modules and architecture design of IDEA-Net, we conducted the following ablation studies. For each case, we retrained the modified network with the same training stragety as the full network and tested it on the same dataset.

(a)-(b) Point-wise temporal consistency. We replaced this module with two types of matrices, i.e., a random permutation matrix and a matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ with $d_{ij} = 1/\|\mathbf{p}_0^i - \mathbf{p}_1^j\|_2$. As shown in Figs. 12a and 12b, without this module, the resulting in-between frames show more wrong

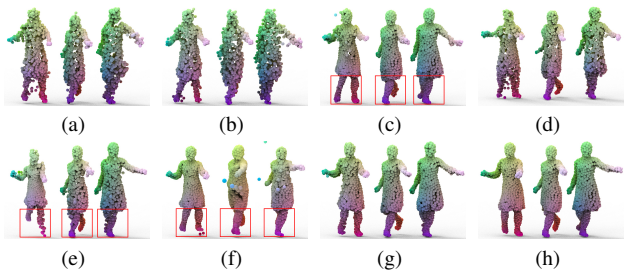


Figure 12. Visual comparisons of the results in ablation studies. (a)-(f) the interpolated frames by our method under the six ablation settings introduced in Section 4.4, and (g) the proposed full model; (h) the ground-truth in-between frames. We highlighted the regions with serious shape distortion in red boxes.

Table 4. Quantitative results ($\times 10^{-3}$) of the ablation studies on the DHB dataset. (a)-(f) correspond to the six settings in Sec. 4.4.

Settings	Swing		Longdress	
	EMD	CD	EMD	CD
Full model	**6.74**	**1.21**	**5.84**	**0.89**
(a)	23.46	3.11	17.82	2.22
(b)	25.02	3.26	21.50	2.26
(c)	10.36	2.38	6.33	0.97
(d)	18.28	2.85	24.32	2.57
(e)	11.00	1.78	6.13	1.40
(f)	25.47	9.05	29.85	5.51

points, and the values of both EMD and CD increase significantly, demonstrating the effectiveness of this module. We also refer the readers to *Supplementary Material* for the visual illustration of learned $\mathbf{A}$.

(c) Linear interpolation. We omitted the linear interpolation step and directly added $\mathbf{P}_0$ and $\mathbf{P}_1$ to the increments $\mathbf{\Delta}_{0\to t}$ and $\mathbf{\Delta}_{1\to t}$, respectively. We report the results in Table 4 (c) and Fig. 12c, showing that the loss gets worse and the generated shapes become less realistic than those of the full model, e.g., the shapes of the hands and legs.

(d) Directly regress in-between frames from features. Without employing the linear interpolation to generate coarse estimation, we directly regressed in-between point clouds from the interpolated feature derived by Eq. (12). Table 4 (d) and Fig. 12d provide the quantitative and visual results, respectively, where it can be seen that the CD and EMD values increase more than twice, and the resulting point clouds have considerable artifacts.

(e) Trajectory compensation. In Table 4 (e) and Fig. 12e, we reported the results of IDEA-Net without the compensation module, where it can be seen that without this module, the values of both EMD and CD increase and the resulting shapes are partially collapsed.

(f) Dual-branch design. To conduct comparisons, we trained a single-branch network, i.e., the branch predicting $\mathbf{O}_{0\to t}$ in IDEA-Net. As shown in Table 4 (f) and Fig. 12f, the single-branch model produces much larger EMD and CD than the dual-branch model (i.e., the full model). Besides, its visual results are noisy and the shapes (e.g., the legs) deviate from the ground-truth ones, validating the significant advantages of the symmetric dual-branch design with shared modules in terms of the reconstruction quality. Moreover, we showed the training (solid lines) and testing (dashed lines) losses for the dual- and single-branch models in Fig. 13. As can be seen, the dual-branch model can stabilize the training process. Also, it potentially avoids overfitting and improves the generalization ability of the model, i.e., although the training losses of these two designs converge to comparable values, the testing error of the dual-branch design is much smaller than that of the single-branch design.

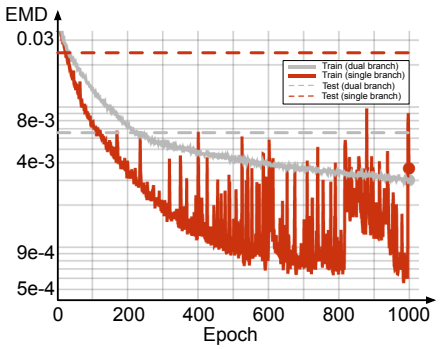


Figure 13. Comparison of the training and testing losses of the dual- and single-branch designs.

5. Conclusion and Discussion

We proposed IDEA-Net, an end-to-end framework for temporally interpolating dynamic 3D point cloud sequences with large non-rigid deformation. We formulated the problem as estimation of point-wise trajectories which can be approximated in a coarse-to-fine manner with the aid of the explicitly learned temporal consistency. By extensive experiments and ablation studies, we demonstrated the significant advantages of the proposed IDEA-Net over state-of-the-art methods in both quantitatively and visually. We believe our framework can provide novel insights for the acquisition and processing of dynamic point cloud sequences.

Despite this paper reveals essential discoveries on the problem of dynamic 3D point cloud interpolation, we can expect to further improve performance by enriching specific technical roadmap from different aspects. First, we can increase the receptive field of the time domain by feeding a longer frame group instead of a pair. Second, we can consider introducing sequential modeling framework to jointly learn temporal correlation across multiple frames. Third, considering the fact that points move at different speed and acceleration, we can design higher-order trajectory estimation schemes to empower the whole model. Finally, it is highly desired to construct a quantitative metric for reliably evaluating the quality of interpolated point cloud sequences.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 2

[2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019. 2

[3] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5

[4] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 5

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 2

[6] Alexander Clemm, Maria Torres Vega, Hemanth Kumar Ravuri, Tim Wauters, and Filip De Turck. Toward truly immersive holographic-type communication: Challenges and solutions. *IEEE Communications Magazine*, 58(1):93–99, 2020. 1

[7] Marvin Eisenberger, David Novotny, Gael Kerchenbaum, Patrick Labatut, Natalia Neverova, Daniel Cremers, and Andrea Vedaldi. NeuroMorph: Unsupervised Shape Interpolation and Correspondence in One Go. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7473–7483, 2021. 2, 3

[8] Hehe Fan and Yi Yang. Pointrnn: Point recurrent neural network for moving point cloud processing. *arXiv preprint arXiv:1910.08287*, 2019. 2

[9] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14204–14213, 2021. 2, 4

[10] Hehe Fan, Xin Yu, Yuhang Ding, Yi Yang, and Mohan Kankanhalli. Pstnet: Point spatio-temporal convolution on point cloud sequences. In *International Conference on Learning Representations*, 2021. 2

[11] Hehe Fan, Xin Yu, Yi Yang, and Mohan Kankanhalli. Deep Hierarchical Representation of Point Cloud Videos via Spatio-Temporal Decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. 2

[12] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 1, 2, 3

[13] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018. 1, 2

[14] Tarun Kalluri, Deepak Pathak, Manmohan Chandraker, and Du Tran. Flavr: Flow-agnostic video representations for fast frame interpolation. *arXiv preprint arXiv:2012.08512*, 2021. 1

[15] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019. 3

[16] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9246–9255, 2019. 2

[17] Yu-Lun Liu, Yi-Tung Liao, Yen-Yu Lin, and Yung-Yu Chuang. Deep video frame interpolation using cyclic frame generation. In *Proceedings of the 33rd Conference on Artificial Intelligence (AAAI)*, 2019. 2

[18] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4463–4471, 2017. 2

[19] Fan Lu, Guang Chen, Sanqing Qu, Zhijun Li, Yinlong Liu, and Alois Knoll. Pointinet: Point cloud frame interpolation network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 1, 2, 3, 5, 6, 7

[20] 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 *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018. 2

[21] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 498–507, 2018. 1, 2

[22] Sajjad Mozaffari, Omar Y. Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–15, 2020. 1

[23] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5378–5388. IEEE, 2019. 2

[24] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1710, 2018. 2

[25] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 670–679, 2017. 2

[26] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings*

of the IEEE International Conference on Computer Vision, pages 261–270, 2017. 2

[27] Rick Parent. *Computer animation: algorithms and techniques*. Newnes, 2012. 1

[28] Lukas Prantl, Nuttapong Chentanez, Stefan Jeschke, and Nils Thuerey. Tranquil Clouds: Neural Networks for Learning Temporally Coherent Features in Point Clouds. In *International Conference on Learning Representations*, 2019. 2

[29] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, pages 652–660, 2017. 1, 2, 5, 6, 7

[30] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proc. NeurIPS*, pages 5105–5114, 2017. 2

[31] Marie-Julie Rakotosaona and Maks Ovsjanikov. Intrinsic point cloud interpolation via dual latent space navigation. In *European Conference on Computer Vision*, pages 655–672. Springer, 2020. 1, 2, 3, 5, 6, 7

[32] Davis Rempe, Tolga Birdal, Yongheng Zhao, Zan Gojcic, Srinath Sridhar, and Leonidas J. Guibas. Caspr: Learning canonical spatiotemporal point cloud representations. In *NeurIPS*, 2020. 2

[33] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision*, pages 59–66. IEEE, 1998. 5

[34] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019. 2

[35] Daniel Vlasic, Ilya Baran, W. Matusik, and Jovan Popovic. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH 2008*, 2008. 5

[36] Jian Wang. *High resolution 2D imaging and 3D scanning with line sensors*. PhD thesis, Carnegie Mellon University, 2018. 1

[37] 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 TOG*, 38(5):1–12, 2019. 2, 4

[38] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 2

[39] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 2

[40] André Zenner, Akhmajon Makhsadov, Sören Klingner, David Liebemann, and Antonio Krüger. Immersive process model exploration in virtual reality. *IEEE transactions on visualization and computer graphics*, 26(5):2104–2114, 2020. 1

[41] Longyu Zhang, Haiwei Dong, and Abdulmotaleb El Saddik. From 3d sensing to printing: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 12:1–23, 10 2015. 1