Deep Hierarchical Representation of Point Cloud Videos via Spatio-Temporal Decomposition

Hehe Fan, Xin Yu, Yi Yang, *Senior Member, IEEE*, and Mohan Kankanhalli, *Fellow, IEEE*

Abstract—In point cloud videos, point coordinates are irregular and unordered but point timestamps exhibit regularities and order. Grid-based networks for conventional video processing cannot be directly used to model raw point cloud videos. Therefore, in this work, we propose a point-based network that directly handles raw point cloud videos. First, to preserve the spatio-temporal local structure of point cloud videos, we design a point tube covering a local range along spatial and temporal dimensions. By progressively subsampling frames and points and enlarging the spatial radius as the point features are fed into higher-level layers, the point tube can capture video structure in a spatio-temporally hierarchical manner. Second, to reduce the impact of the spatial irregularity on temporal modeling, we decompose space and time when extracting point tube representations. Specifically, a spatial operation is employed to encode the local structure of each spatial region in a tube and a temporal operation is used to encode the dynamics of the spatial regions along the tube. Empirically, the proposed network shows strong performance on 3D action recognition, 4D semantic segmentation and scene flow estimation. Theoretically, we analyse the necessity to decompose space and time in point cloud video modeling and why the network outperforms existing methods.

Index Terms—Point cloud, spatio-temporal modeling, video analysis, action recognition, semantic segmentation, scene flow estimation

1 Introduction

Point cloud videos provide more flexibility for action recognition and decision making in poor visibility environments, and covers more precise geometry dynamics than the conventional videos. A point cloud video can be represented by a sequence of 3D point coordinate sets. Moreover, when RGB images are available, they are often used as additional features associated with 3D points to enhance the discriminativeness of point clouds. However, unlike conventional grid-based videos, point cloud videos are irregular and unordered in the spatial dimension and points do not emerge consistently over time. Therefore, as shown in Fig. 1, existing grid-based methods on conventional videos [1], [2], [3] are not suitable for directly modeling raw point cloud videos.

To capture the dynamics in point cloud videos, one solution is converting a point cloud video to a sequence of 3D voxels, and then applying grid-based 4D convolutions to the voxel sequence. However, directly performing convolutions on voxel sequences requires a large amount of computation and special engineering efforts, e.g., sparse convolution [4]. Furthermore, quantization errors are inevitable during voxelization, which may restrict applications that require precise measurement of scene geometry. Another solution is appending 1D temporal dimension to 3D points and treating point cloud videos as unordered 4D point sets [5], in which point tracking is usually employed to capture the dynamics in a region. Because points may flow in and out across frames, computing an accurate point trajectory is extremely difficult, especially for long videos. Moreover, simply concatenating coordinates and timestamps together neglects the temporal order and regularity, which may not properly exploit the temporal information and lead to inferior performance.

In this paper, we propose a novel point-based deep neural network, named PSTNet++, to directly encode raw point cloud videos without resorting to unstable point tracking. Inspired by 3D convolutional neural networks [1], [2], [3] on conventional video modeling, our PSTNet++ encodes point cloud videos in a spatio-temporally hierarchical manner. Instead of tracing points along an entire video, the proposed method models a video by capturing spatio-temporal local structure changes and thus alleviates the requirement for point tracking. Specifically, we introduce a point tube, which is formed by selecting an anchor point and propagating the anchor and its surrounding spatial region to a few nearest frames. By subsampling frames and points and enlarging spatial regions as the point features are fed into higher-level layers, our PSTNet++ can construct spatio-temporal hierarchy for point cloud video modeling.

To reduce the impact of the spatial irregularity of points on temporal modeling, we propose a point spatio-temporal operation (PSTOp) that decouples the spatio-temporal structure encoding. Specifically, PSTOp consists of (i) a spatial operation that encodes the spatial structure of 3D points

- *Hehe Fan and Mohan Kankanhalli are with the School of Computing, National University of Singapore, Singapore 119077. E-mail: hehe.fan@nus.edu.sg, mohan@comp.nus.edu.sg.*
- *Xin Yu is with the Center for Artificial Intelligence, University of Technology Sydney, Ultimo, NSW 2007, Australia. E-mail: xin.yu@uts.edu.au.*
- *Yi Yang is with the College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang 310058, China. E-mail: yangyics@zju.edu.cn.*

Manuscript received 14 April 2021; revised 5 August 2021; accepted 6 December 2021. Date of publication 14 December 2021; date of current version 3 November 2022.
*This work was supported in part by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme under Grant A18A2b0046.*
(Corresponding author: Hehe Fan.)
Recommended for acceptance by L. Torresani.
Digital Object Identifier no. 10.1109/TPAMI.2021.3135117

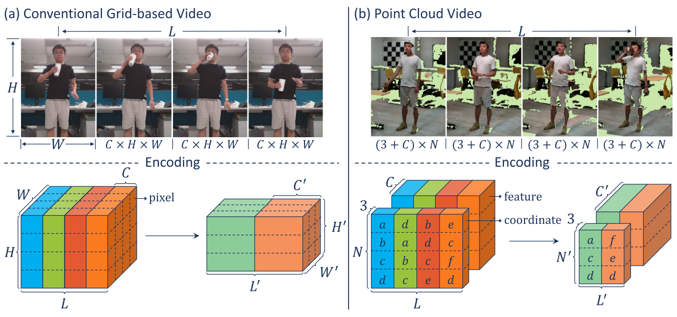


Fig. 1. Illustration of grid-based and point-based processing on videos. (a) For a grid-based video, each grid represents a feature of a pixel, where C, L, H and W denote the feature dimension, the number of frames, height and width, respectively. A 3D convolution encodes an input to an output of size $C' \times L' \times H' \times W'$. (b) A point cloud video consists of a coordinate part ($3 \times L \times N$) and a feature part ($C \times L \times N$), where N indicates the number of points in a frame. Our point spatio-temporal operation (PSTOp) encodes an input to an output composed of a coordinate tensor ($3 \times L' \times N'$) and a feature tensor ($C' \times L' \times N'$). Usually, $L' \leq L$ and $N' \leq N$ so that networks can model point cloud videos in a spatio-temporally hierarchical manner. Note that points in different frames are not consistent, and thus it is challenging to preserve the spatio-temporal structure.

and (ii) a temporal operation that encodes the temporal dynamics of point cloud videos. Moreover, different from the 4D point-based method [5], which defines point cloud video distance as the maximum frame-level distance (Hausdorff distance) among all the corresponding timesteps and neglects other timesteps, our PSTOp treats point cloud video distance as the sum of all the frame-level distances and thus properly reflects the global point cloud video difference.

To conduct point-wise segmentation tasks, we extend our PSTOp to a transposed version that interpolates temporal dynamics and spatial features. Experiments on 3D action recognition, 4D semantic segmentation and scene flow estimation demonstrate the effectiveness of PSTNet++ on point cloud video modeling. The contributions of this paper are fourfold:

- Benefiting from our point tubes, we build a spatio-temporally hierarchical network to preserve the local structure of raw point cloud videos.
- We propose a point spatio-temporal operation that decomposes spatial and temporal information in encoding the local structure of raw point cloud video.
- To decode point cloud videos for point-level prediction tasks, we propose a point spatio-temporal transposed operation that is able to interpolate the temporal dynamics and spatial features.
- Extensive experiments on seven datasets indicate that our method remarkably improves the accuracy of 3D action recognition, 4D semantic segmentation and scene flow estimation.

This paper is an extension of our previous work PSTNet [6]. For PSTNet, we did not provide a theorem to guarantee its effectiveness. In PSTNet++, we improve PSTNet and theoretically prove the effectiveness of PSTNet++ on raw point cloud video modeling. Experiments also demonstrate that PSTNet++ is superior to PSTNet. Moreover, compared to our previous work PSTNet, we include another two datasets for 3D action recognition, i.e., N-UCLA [7] and UWA3DII [8].

2 Related Work

Learning Representations on Grid-based Videos. Impressive progress has been made on generating compact and discriminative representations for RGB/RGBD videos due to the success of deep neural networks. For example, two-stream convolutional neural networks [9], [10] employ a spatial stream and an optical flow stream for video modeling. To capture the temporal dependencies of videos, recurrent neural networks [11], [12] and pooling techniques [13] are employed. In addition, by stacking multiple 2D frames into a 3D tensor, 3D convolutional neural networks [1], [2], [3], [14] are widely used to learn spatio-temporal representations for videos, and achieve promising performance. Besides, interpretable video or action reasoning methods [15], [16] are proposed by explicitly parsing changes in videos.

Static Point Cloud Processing. Static point cloud analysis has been widely investigated in many problems, such as classification, object part segmentation, scene semantic segmentation [17], [18], [19], [20], [21], [22], [23], reconstruction [24], [25] and object detection [26], [27]. Most recent works aim to directly manipulate point sets without transforming coordinates into regular voxel grids. Since a point cloud is essentially a set of unordered points and invariant to permutations of its points, static point cloud processing methods mainly focus on designing effective point-based spatial correlation operations that do not rely on point orders. However, those methods do not take the temporal dynamic information into account and may produce suboptimal results when processing point cloud videos.

Dynamic Point Cloud Modeling. Compared with static point cloud processing, dynamic point cloud modeling is a fairly new task but very important for intelligent systems to understand the dynamic world. Fast and Furious (FaF) [28] converts a point cloud frame into a bird's view voxel and then extracts features via 3D convolutions. MinkowskiNet [4] uses 4D Spatio-Temporal ConvNets on a 4D occupancy grid. Fan and Yang [29] proposed a series of point recurrent neural networks (PointRNNs) for moving point cloud prediction. MeteorNet [5] extends 3D points to 4D points by appending a temporal dimension and then employs the framework of PointNet++ [18] to process those 4D points. 3DV [30] first integrates 3D motion information into a regular compact voxel set and then applies PointNet++ to extract representations from the set for 3D action recognition via temporal rank pooling. P4Transformer [31] employs transformer to avoid point tracking for raw point cloud video modeling. Niemeyer *et al.* [32] learned a temporal and spatial vector field in which every point is assigned with a motion vector of space and time for 4D reconstruction. Prantl *et al.* [33] learned stable and temporal coherent feature spaces for point-based super-resolution. CaSPR [34] learns to encode spatio-temporal changes of object shapes from dynamic point clouds for reconstruction and camera pose estimation. Different from previous works, we propose a point-based network to model spatio-temporal information of raw point cloud videos via decomposing space and time hierarchically. Besides, scene flow estimation [35], [36] can be seen as computing the motion between two point cloud frames. Different from the scene flow estimation methods, our PSTNet++ aims to capture the long-term dependency of multiple frames.

To construct the temporal hierarchy, our method employs a frame selection mechanism, which involves a temporal kernel size, a temporal striding size and a temporal padding size.

3 Proposed Method

In this section, we first present point tubes. Second, we introduce how the proposed point spatio-temporal operation (PSTOp) extracts features from point tubes. In order to address dense point prediction tasks, i.e., semantic segmentation, we develop a point spatio-temporal transposed operation (PSTTransOp). Finally, we incorporate our operations into deep spatio-temporally hierarchical networks, i.e., PSTNet++, to address different dynamic point cloud tasks.

Let $\boldsymbol{P}_t \in \mathbb{R}^{3\times N}$ and $\boldsymbol{F}_t \in \mathbb{R}^{C\times N}$ denote the point coordinates and features of the tth frame in a point cloud video, where N and C denote the number of points and feature channels, respectively. Given a point cloud video $\big([\boldsymbol{P}_1;\boldsymbol{F}_1],[\boldsymbol{P}_2;\boldsymbol{F}_2],\ldots,[\boldsymbol{P}_L;\boldsymbol{F}_L]\big)$, the proposed PSTOp will encode the video to embedded frames $\big([\boldsymbol{P}'_1;\boldsymbol{F}'_1],[\boldsymbol{P}'_2;\boldsymbol{F}'_2],\ldots,[\boldsymbol{P}'_{L'};\boldsymbol{F}'_{L'}]\big)$, where L and L' indicate the number of frames and $\boldsymbol{P}'_t \in \mathbb{R}^{3\times N'}$, and $\boldsymbol{F}'_t \in \mathbb{R}^{C'\times N'}$ represent the encoded coordinates and features (usually $N' < N$, $L' < L$ and $C' > C$).

3.1 Point Tube

The power of 3D convolutional neural networks (CNNs) in conventional video modeling comes from spatio-temporally hierarchical architectures. This motivates us to build spatio-temporal hierarchy in raw point cloud video modeling. Grid-based 3D convolutions can be easily performed on regular conventional videos by shifting along the length, height and width dimensions to capture the spatio-temporal local structure. However, it is challenging to preserve spatio-temporal local structure for point cloud video encoding, because point cloud videos are irregular and unordered in 3D space and points emerge inconsistently across different frames. To address this problem, we introduce a point tube. In contrast to pixel cubes, i.e., the receptive field of 3D convolutions, in which pixels are distributed regularly, point tubes are dynamically generated according to input videos so that dense areas have more tubes than sparse ones. Specifically, the point tube is constructed as follows:

Temporal Anchor Frame Selection. For an entire point cloud video, we need to select some anchor frames to generate our tubes. Temporal anchor frames in a point cloud video are automatically selected based on temporal radius (r_t), temporal stride (s_t) and temporal padding (p), as shown in the solid frames in Fig. 2. Moreover, we set $r_t \geq p$ to avoid selecting a padding frame as an anchor frame. Specifically, the frames with timestamp $\{r_t+1-p, r_t+1-p+s_t, r_t+1-p+2s_t, \ldots, L+p-r_t\}$ are selected as temporal anchor frames, and $L+2p-2r_t-1$ can be divided by s_t so that point tubes are correctly constructed with the same temporal size. The length L' of the encoded video is then expressed as $\frac{L+2p-2r_t}{s_t}+1$.

Spatial Anchor Point Sampling. Once a temporal anchor frame is selected, we need to choose spatial anchor points that can represent the distribution of all the points in the frame. Given a spatial subsampling rate s_s, this operation aims to subsample N points to $N' = \lfloor\frac{N}{s_s}\rfloor$ points. We employ the farthest point sampling (FPS) [18] to sample points in each anchor frame. Point tubes are generated based on the sampled anchor points.

Transferring Spatial Anchor Points. 3D convolutions can effectively capture local changes within a cube rather than tracking a specific pixel. Inspired by this idea, we propagate the positions of sampled anchor points to the neighboring frames without tracking, and they are regarded as the anchor points in those frames. Specifically, each anchor point is transferred to the r_t nearest frames. In general, the temporal radius r_t is much smaller, e.g., 1 or 2, than the input length, to effectively model local temporal dynamics. The original and transferred anchor points form the central axis of a tube.

Spatial Neighbor. This step aims to find the spatial neighbors of every anchor point in each frame for performing the spatial operation. A spatial radius r_s is used to search neighbors within the tube slice, where spatial local structure of points is depicted. Note that, padding is usually used in grid-based operation to align feature maps. However, point-based spatial operation is not conducted on grids and thus spatial padding is not employed.

By capturing the spatio-temporal structure in point tubes, our network is able to capture the dynamic changes in local areas. The temporal radius r_t and spatial radius r_s allow our point tube to capture spatio-temporal local structure. By progressively subsampling frames (according to s_t) and points (according to s_s) and enlarging spatial radius r_s as the layer goes deeper, our point tube can capture longer dependencies of temporal and spatial information of point cloud videos, as shown in Fig. 2.

3.2 Point Spatio-Temporal Operation (PSTOp)

3.2.1 Decomposing Space and Time in Point Cloud Video Modeling

To capture the structure in point tubes, we propose a point spatio-temporal operation (PSTOp). Because point cloud videos are spatially irregular and unordered but temporally ordered, our PSTOp decouples these two dimensions in order to reduce the impact of the spatial irregularity of points on temporal modeling. Moreover, the scales of spatial displacements and temporal differences in point cloud videos may not be compatible. Treating them uniformly is not conducive for network optimization. Besides, because space and time are orthogonal and independent of each other, this also motivates us to decompose spatial and temporal modeling. In this way, we not only make the spatio-temporal modeling easier but also significantly improve the ability to capture the temporal information.

Our PSTOp consists of a spatial operation and a temporal operation. The spatial operation is employed to capture the local structure of points within each spatial region of a point tube. The temporal operation is used to model the dynamics of the spatial regions along the point tube. In this fashion, we are able to capture the spatial layout and temporal dynamics of points in a point tube. Note that, the decomposition can also be expressed as applying temporal operation first to encode point trajectories and then spatial operation to trajectory features. However, doing so requires point tracking to achieve point trajectories. Because it is difficult to computing accurate point trajectories and tracking points

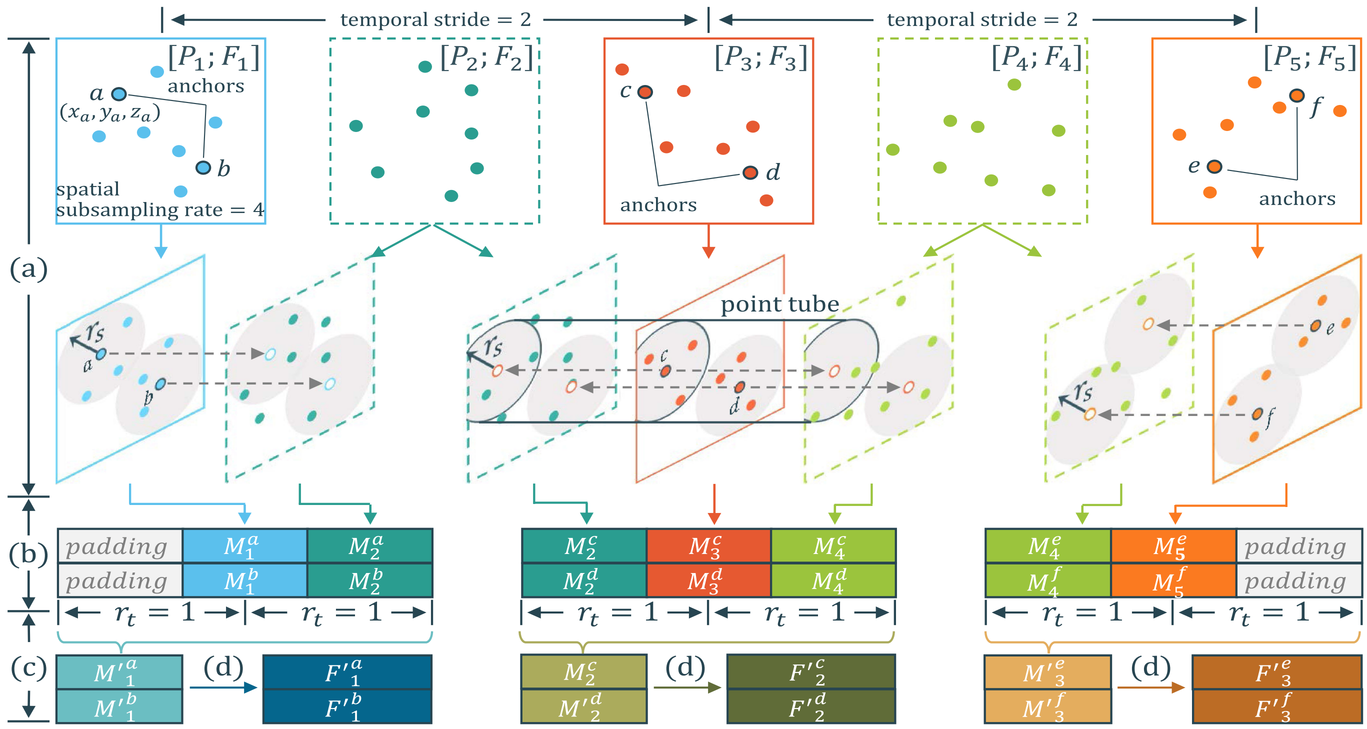


Fig. 2. Illustration of the proposed point spatio-temporal operation (PSTOp). The input contains $L = 5$ frames, with $N = 8$ points per frame. **(a)** Point tube construction. Based on the temporal radius $r_t = 1$, temporal stride $s_t = 2$, and temporal padding $p = 1$, the 1st, 3rd, 5th frames are selected as temporal anchor frames. According to a spatial subsampling rate $s_s = 4$, two spatial anchor points are sampled by FPS in each anchor frame. The sampled anchor points are then transferred to the $r_t = 1$ nearest neighboring frames. A point tube is constructed with a spatial radius r_s for the anchor points. **(b)** The spatial operation encodes the local structure around each anchor point. **(c)** The temporal operation first encodes the $2r_t + 1$ spatial features individually and then merge the features to a spatio-temporal feature. As a result, the original point cloud video of size $L \times N = 5 \times 8$ is encoded as a video of size $L' \times N' = 3 \times 2$.

usually relies on point colors [5] and may fail to handle colorless point clouds, we opt to model the spatial structure of irregular points first, and then capture the temporal information from the spatial regions.

Benefiting from the decomposition of spatio-temporal modeling, we can employ existing methods [18], [19], [20], [21], [23] in static point cloud processing as the spatial operation. However, in order to encode long videos, the spatial operation should be both computation-efficient and memory-efficient. In this paper, we employ the simple yet effective PointNet++ [18] as the spatial operation. Moreover, as shown in Section 3.2.2, the use of PointNet++ can theoretically guarantee the effectiveness of our PSTOp. For the temporal operation, since point cloud videos are temporally ordered and regular, we propose to use a series of time-related multilayer perceptrons (MLPs) to distinguish different moments, instead of directly encoding timestamps like MeteorNet [5]. Our PSTOp is formulated as follows,

$$\boldsymbol{M}_t^{(x,y,z)} = \underset{\|(\delta_x,\delta_y,\delta_z)\|\le r_s}{\mathrm{MAX}}\, \mathcal{S}\text{−MLP}\Big(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}, \delta_x, \delta_y, \delta_z\Big),$$

$$\boldsymbol{F}_t'^{(x,y,z)} = \sum_{k=-r_t}^{r_t} \mathcal{T}\text{−MLP}_k\Big(\boldsymbol{M}_{t+k}^{(x,y,z)}\Big), \tag{1}$$

where $(x, y, z) \in \boldsymbol{P}_t$ and $(\delta_x, \delta_y, \delta_z)$ represents displacement. The $\boldsymbol{F}_t^{(x,y,z)} \in \mathbb{R}^C$ is the feature of point (x, y, z) at time t. The r_s and r_t are the spatial radius and temporal radius, respectively. The $\mathcal{S}$−MLP is the shared MLP of the spatial operation. The $\mathcal{T}\text{−MLP}_{-r_t}, \dots, \mathcal{T}\text{−MLP}_{r_t}$ are the time-related MLPs of the temporal operation. We illustrate our PSTOp in Fig. 2.

3.2.2 *Theoretical Analysis*

For simplicity, we do not consider spatial or temporal radius in this section. Let $\boldsymbol{S} = [\boldsymbol{P}; \boldsymbol{F}]$, $D = 3 + C$ and $\mathcal{X} = \{\boldsymbol{S} | \boldsymbol{S} \subseteq [0, 1]^D, |\boldsymbol{S}| = N, N \in \mathbb{Z}^+\}$ is the set of D-dimensional point clouds inside a D-dimensional unit cube. Suppose $\mathcal{X}_t$ denotes the set of point clouds at time $t \in \mathbb{Z}$ and $\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L$ is the set of point cloud videos of length L. Suppose $\mathsf{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2, \dots, \boldsymbol{S}_L) \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L$ is a specific video sampled from the set of point cloud videos.

I. Static Point Cloud (Individual Point Cloud Frame)

A point cloud is essentially a unordered point set. Therefore, the Hausdorff metric can be used to measure the distance of two individual point cloud frames [5], [17], [18]. Specifically, the Hausdorff distance returns the greatest of all the distances from a point in one cloud to the closest point in the other cloud, to measure how far two point cloud frames are from each other. Given two point clouds $\boldsymbol{S}$ and $\boldsymbol{S}'$, the Hausdorff distance $d_H(\cdot, \cdot)$ is defined as follows,

$$d_H(\boldsymbol{S}, \boldsymbol{S}') = \max\Big\{ \sup_{\boldsymbol{s}\in\boldsymbol{S}} \inf_{\boldsymbol{s}'\in\boldsymbol{S}'} d(\boldsymbol{s}, \boldsymbol{s}'), \sup_{\boldsymbol{s}'\in\boldsymbol{S}'} \inf_{\boldsymbol{s}\in\boldsymbol{S}} d(\boldsymbol{s}, \boldsymbol{s}') \Big\}, \tag{2}$$

where $d(\cdot, \cdot)$ is a distance metric for two points. Based on the Hausdorff distance, PointNet [17] theoretically proves its ability to model static point clouds (shown in Table 1).

II. Point Cloud Video

A point cloud video is a sequence of point clouds. To measure the distance between two point cloud videos, MeteorNet [5] defines the video-level distance as the maximum per-frame Hausdorff distance among all respective frame pairs,

TABLE 1
Comparison Among MinkowskiNet [4], PointNet [17]/PointNet++ [18], MeteorNet [5], PSTNet [6] and our PSTNet++

Voxel-based Method			
Method		Network	
MinkowskiNet		$\boldsymbol{F}_t'^{(x,y,z)} = \sum_{l=-\lfloor L/2\rfloor}^{\lfloor L/2\rfloor} \sum_{w=-\lfloor W/2\rfloor}^{\lfloor W/2\rfloor} \sum_{h=-\lfloor H/2\rfloor}^{\lfloor H/2\rfloor} \sum_{d=-\lfloor D/2\rfloor}^{\lfloor D/2\rfloor} \mathcal{W}_l^{(h,w,d)} \cdot \boldsymbol{F}_{t+l}^{(x+w,y+h,z+d)}$	
Point-based Method			
Method	Distance Metric	Theorem	Network
PointNet, PointNet++ *(static point)*	Hausdorff: $d_H(\cdot,\cdot)$	Suppose $f : \mathcal{X} \rightarrow \mathbb{R}$ is a continuous set function *w.r.t* Hausdorff distance $d_H(\cdot,\cdot)$. $\forall\epsilon > 0$, $\exists$ a continuous function h and a continuous γ such that for any $\boldsymbol{S} \in \mathcal{X}$, $\lvert f(\boldsymbol{S}) - \gamma\big(\underset{i\in\{1,\cdots,N\}}{\mathrm{MAX}}\{h(\boldsymbol{S}^i)\}\big)\rvert < \epsilon.$	$\boldsymbol{F}'^{(x,y,z)} = \underset{\lVert\delta_x,\delta_y,\delta_z)\rVert\leq r_s}{\mathrm{MAX}} \mathrm{MLP}\big(\boldsymbol{F}^{(x+\delta_x,y+\delta_y,z+\delta_z)}, \delta_x, \delta_y, \delta_z\big)$
MeteorNet	$d_S(\mathsf{S},\mathsf{S}') = \max_t \{d_H(\boldsymbol{S}_t, \boldsymbol{S}_t')\}$ where $t \in \{1,\cdots,L\}$	Suppose $f : \mathcal{X}_1 \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_L \rightarrow \mathbb{R}$ is a continuous function *w.r.t* $d_S(\cdot,\cdot)$. $\forall\epsilon > 0$, $\exists$ a continuous function h and a continuous γ such that for any $\mathsf{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_L) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_L$, $\lvert f(\mathsf{S}) - \gamma\big(\underset{t\in\{1,\cdots,L\},\ i\in\{1,\cdots,N\}}{\mathrm{MAX}}\{h(\boldsymbol{S}_t^i, t)\}\big)\rvert < \epsilon.$	$\boldsymbol{F}_t'^{(x,y,z)} = \underset{\lVert\delta_x,\delta_y,\delta_z)\rVert\leq r_s}{\mathrm{MAX}} \mathrm{MLP}\big(\boldsymbol{F}_{t+\delta_t}^{(x+\delta_x,y+\delta_y,z+\delta_z)}, \delta_x, \delta_y, \delta_z, \delta_t\big)$ where $\delta_t = l - t,\ l \in \{1,\cdots,L\}$
PSTNet	-	-	$\boldsymbol{M}_t^{(x,y,z)} = \sum_{\lVert\delta_x,\delta_y,\delta_z)\rVert\leq r_s} \mathcal{S}^{(\delta_x,\delta_y,\delta_z)} \cdot \big(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}\big)$ $\boldsymbol{F}_t'^{(x,y,z)} = \sum_{k=-r_t}^{r_t} \mathcal{T}_k \cdot \big(\boldsymbol{M}_t^{(x,y,z)}\big)$
PSTNet++	$d_S(\mathsf{S},\mathsf{S}') = \sum_{t=1}^{L} \lambda_t\big(d_H(\boldsymbol{S}_t, \boldsymbol{S}_t')\big)$ where λ_t is continuous and strictly increasing and $\lambda_t(0) = 0$	Suppose $f : \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L \rightarrow \mathbb{R}$ is a continuous function *w.r.t* $d_S(\cdot,\cdot)$. $\forall\epsilon > 0$, $\exists$ a continuous function h, a set of continuous functions $\eta_1, \cdots, \eta_L$ and a continuous function γ such that for any $\mathsf{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_L) \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L$, $\lvert f(\mathsf{S}) - \gamma\Big(\sum_{t=1}^{L} \eta_t\big(\underset{i\in\{1,\cdots,N\}}{\mathrm{MAX}}\{h(\boldsymbol{S}_t^i)\}\big)\Big)\rvert < \epsilon.$	$\boldsymbol{M}_t^{(x,y,z)} = \underset{\lVert\delta_x,\delta_y,\delta_z)\rVert\leq r_s}{\mathrm{MAX}} \mathcal{S}\text{-MLP}\big(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}, \delta_x, \delta_y, \delta_z\big)$ $\boldsymbol{F}_t'^{(x,y,z)} = \sum_{k=-r_t}^{r_t} \mathcal{T}\text{-MLP}_k\big(\boldsymbol{M}_t^{(x,y,z)}\big)$

The voxel-based MinkowskiNet employs a 4D convolution to capture the spatio-temporal structure of point cloud videos. The L, W, H and D denote length, width, height and depth of the size of the 4D convolution kernel $\mathcal{W}$. MeteorNet extends 3D points to 4D points (x, y, z, t) and models the spatial and temporal structure together. By contrast, we propose to decouple space and time in modeling to better exploit the temporal order and regularity. The continuous functions in theorems are implemented as MLPs in networks. In practice, the γ function in theorems is usually omitted in networks.

$$\text{MeteorNet}: d_S(\mathsf{S},\mathsf{S}') = \max_{t=1,\ldots,L}\{d_H(\boldsymbol{S}_t, \boldsymbol{S}_t')\}. \tag{3}$$

However, as point cloud videos are essentially frame sequences, this metric focuses on the greatest frame-level distance and neglects other frame-level distances. In this way, it cannot properly represent the global difference between two point cloud videos and may overestimate the difference at a certain timestep. To better model the distance of two point cloud videos, we propose to use the sum of per-frame Hausdorff distance as the distance metric,

$$\text{PSTNet++(w/otemporal)}: d_S(\mathsf{S},\mathsf{S}') = \sum_{t=1}^{L} d_H(\boldsymbol{S}_t, \boldsymbol{S}_t'). \tag{4}$$

In this way, the difference at each moment in a video is involved and considered. However, a problem is that Eq. (4) neglects the temporal structure because all moments in a video are treated equally. To encode temporal information, we add a time-related function λ_t for each moment t,

$$\text{PSTNet++(w/temporal)}: d_S(\mathsf{S},\mathsf{S}') = \sum_{t=1}^{L} \lambda_t\big(d_H(\boldsymbol{S}_t, \boldsymbol{S}_t')\big). \tag{5}$$

Each time-related function is continuous and strictly increasing. Moreover, to be consistent with distance, $\lambda_t(0)$ is 0. The metric Eq. (4) can be seen a special case of the general metric Eq. (5) when $\lambda_1, \lambda_2, \ldots, \lambda_L$ are identity functions.

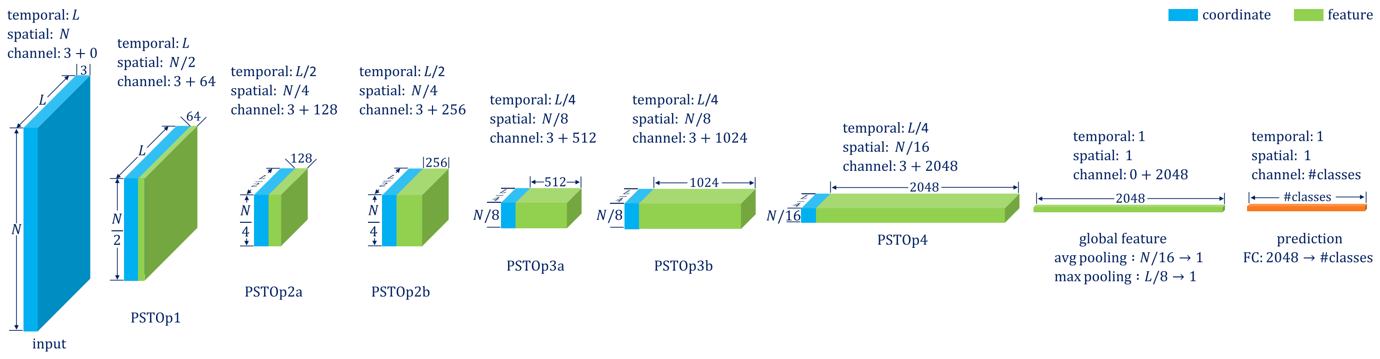
Based on our point cloud video distance metric Eq. (5), we provide the theoretical foundation for our PSTNet++ by showing the universal approximation ability of PSTOp operation to continuous functions on point cloud videos.

Theorem 1. *Suppose $f : \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L \rightarrow \mathbb{R}$ is a continuous function w.r.t the $d_S(\cdot,\cdot)$ in Eq. (5). $\forall\epsilon > 0$, $\exists$ a continuous function h, a set of continuous functions $\eta_1, \ldots, \eta_L$ and a continuous function γ such that for any $\mathsf{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_L) \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_L$,*

$$\left| f(\mathsf{S}) - \gamma\left(\sum_{t=1}^{L} \eta_t\big(\mathrm{MAX}\big(h(\boldsymbol{S}_t^1), \ldots, h(\boldsymbol{S}_t^N)\big)\big)\right)\right| < \epsilon,$$

where $\boldsymbol{S}_t^1, \ldots, \boldsymbol{S}_t^N$ are the elements of $\boldsymbol{S}_t$ extracted in a certain order and MAX is a vector max operator that takes N vectors as input and returns a new vector of the element-wise maximum.

The proof of this theorem is in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2021.3135117. The h function is corresponding to $\mathcal{S}-$MLP and the $\eta_1, \ldots, \eta_L$ functions, which are derived from the $\lambda_1, \ldots, \lambda_L$ functions, are corresponding to $\mathcal{T}-$MLPs in the PSTOp operation.



🟦 coordinate
🟩 feature
temporal: L spatial: N channel: $3+0$
input
temporal: L spatial: $N/2$ channel: $3+64$
PSTOp1
temporal: $L/2$ spatial: $N/4$ channel: $3+128$
PSTOp2a
temporal: $L/2$ spatial: $N/4$ channel: $3+256$
PSTOp2b
temporal: $L/4$ spatial: $N/8$ channel: $3+512$
PSTOp3a
temporal: $L/4$ spatial: $N/8$ channel: $3+1024$
PSTOp3b
temporal: $L/4$ spatial: $N/16$ channel: $3+2048$
PSTOp4
temporal: 1 spatial: 1 channel: $0+2048$
global feature
avg pooling : $N/16 \rightarrow 1$
max pooling : $L/8 \rightarrow 1$
temporal: 1 spatial: 1 channel: #classes
prediction
FC: $2048 \rightarrow$ #classes

Fig. 3. Spatio-temporally hierarchical PSTNet++ for 3D action recognition. The architecture uses six PSTOp layers to extract point cloud video features and one fully-connected (FC) layer for classification. Following [5], [6], [30], we only use point coordinates for 3D action recognition in this paper. Therefore, the number of input channels is 3.

III. Discussion

1) Theoretical explanation for decomposing space and time.

As shown in Table 1, the video distance metric of MeteorNet, i.e., Eq. (3), uses the max operation to encode the temporal difference, which is consistent with the spatial distance metric, i.e., Hausdorff distance. Therefore, MeteorNet can model the spatial and temporal information simultaneously. However, as the max operation may not properly represent global temporal distance and may lead to inaccuracy, we employ the sum operation to capture the temporal difference. Due to the inconsistency of basic operations between the spatial and temporal dimensions, we decompose space and time in modeling.

2) PointNet++ can be viewed as a special case of PSTNet++.

When there is only one frame in a video, Theorem 1 becomes $\left|f(S)-(\gamma\circ\eta_1)\big(\mathrm{MAX}\big(h(S^1),\ldots,h(S^N)\big)\big)\right|<\epsilon$, which is equal to the theorem of PointNet++ (shown in Table 1) by considering $\gamma\circ\eta_1$ as a single function. Therefore, PointNet++ can be seen as a special case of our PSTNet++.

3) Comparison with MinkowskiNet [4].

MinkowskiNet is a voxel-based method, which jointly captures the spatial and temporal structure of 4D tesseracts via a 4D convolution kernel. In contrast, our PSTNet++ is a point-based method, which separately models the spatial and temporal structure of point tubes via a spatial MLP and a set of temporal MLPs. Both MinkowskiNet and PSTNet++ aim to capture the structure of a spatio-temporal local area, which is the 4D tesseract for MinkowskiNet and point tube for PSTNet++. The different is that, as a voxel-based method, MinkowskiNet can learn a 4D convolution kernel from different discrete directions or displacements. In contrast, as point-based, PSTNet++ captures the spatial structure by directly encoding continuous directions or displacements. Moreover, MinkowskiNet treats space and time equally and models the spatial and temporal structure together, while our PSTNet++ decouples spatio-temporal modeling, as space and time are orthogonal and independent and the scales of space and time are not compatible.

3.3 Point Spatio-Temporal Transposed Operation (PSTTransOp)

After PSTOp, the original point cloud video is both spatially and temporally subsampled. However, for point-level prediction tasks, we need to provide point features for all the original points. Thus, we develop a point spatio-temporal transposed operation (PSTTransOp). Suppose $\big([\boldsymbol{P}'_1;\boldsymbol{F}'_1],[\boldsymbol{P}'_2;\boldsymbol{F}'_2],\ldots,[\boldsymbol{P}'_{L'};\boldsymbol{F}'_{L'}]\big)$ is the encoded video of the original one $\big([\boldsymbol{P}_1;\boldsymbol{F}_1],[\boldsymbol{P}_2;\boldsymbol{F}_2],\ldots,[\boldsymbol{P}_L;\boldsymbol{F}_L]\big)$. PSTTransOp propagates features $(\boldsymbol{F}'_1,\boldsymbol{F}'_2,\ldots,\boldsymbol{F}'_{L'})$ to the original point coordinates $(\boldsymbol{P}_1,\boldsymbol{P}_2,\ldots,\boldsymbol{P}_L)$, thus outputting new features $(\boldsymbol{F}''_1,\boldsymbol{F}''_2,\ldots,\boldsymbol{F}''_L)$, where $\boldsymbol{F}''_t\in\mathbb{R}^{C''\times N}$ and C'' denotes the new feature dimension. To this end, PSTTransOp first recovers the temporal length by a temporal transposed operation, and then increases the number of points and assigns temporal features to original points via feature propagation [18],

$$\begin{aligned}
\boldsymbol{M}''^{(x,y,z)}_{t+k} &= \mathcal{T}\text{--MLP}'_k\big(\boldsymbol{F}'^{(x,y,z)}_t\big),\\
\boldsymbol{F}''^{(x,y,z)}_{t+k} &= \mathcal{S}\text{--MLP}'\left(\frac{\sum_{\|(\delta_x,\delta_y,\delta_z)\|\le r} w(\delta_x,\delta_y,\delta_z)\boldsymbol{M}''^{(x+\delta_x,y+\delta_y,z+\delta_z)}_{t+k}}{\sum_{\|(\delta_x,\delta_y,\delta_z)\|\le r} w(\delta_x,\delta_y,\delta_z)}\right),
\end{aligned} \tag{6}$$

where $k\in[-r_t,r_t]$ and $w(\delta_x,\delta_y,\delta_z)=\frac{1}{\|(\delta_x,\delta_y,\delta_z)\|^2}$. An illustration of PSTTransOp operation is shown in Appendix B, available in the online supplemental material.

3.4 PSTNet++ Architectures

PSTNet++ for 3D Action Recognition. As shown in Fig. 3, we employ six PSTOp layers and a fully-connected (FC) layer for 3D action recognition. In the 1st, 2nd, 4th, 6th layers, the spatial subsampling rate is set to 2 to halve the spatial resolution. The spatial radius progressively increases to grow spatial receptive fields when the network goes deeper. In the 2nd and 4th layers, the temporal stride is set to 2 to halve the temporal resolution. In the 2-4 layers, the temporal radius is set to 1 to capture temporal correlation. Note that, in the 1st PSTOp layer, the temporal radius is set to 0, in which only the spatial structure is modeled. After the PSTOp layers, average and max poolings are used for spatial pooling and temporal pooling, respectively. Finally, the FC layer maps the global feature to action predictions.

PSTNet++ for 4D Semantic Segmentation. We use four PSTOp layers and four PSTTransOp layers for 4D semantic segmentation. The spatial subsampling rate is set to 4 to reduce the spatial resolution in the 1-3 PSTOp layers and 2 in the fourth PSTOp layer. The spatial radius of PSTOp layers progressively increases to grow spatial receptive fields when the network goes deeper. In the 3rd PSTOp and 2nd PSTTransOp layers, the temporal radius is set to 1 to capture temporal correlation. Skip

TABLE 2
Architecture Specs of PSTNet++ for 3D Action Recognition and 4D Semantic Segmentation. The r_o denotes the initial spatial radius

PSTNet++ for 3D Action Recognition

Layer	$\mathcal{S}-$MLP			$\mathcal{T}-$MLP			
	r_s	s_s	MLP	r_t	s_t	p	MLP
PSTOp1	r_o	2	[45]	0	1	$[0, 0]$	[64]
PSTOp2a	$2r_o$	2	[96]	1	2	$[1, 0]$	[128]
PSTOp2b	$2r_o$	1	[192]	1	1	$[1, 1]$	[256]
PSTOp3a	$4r_o$	2	[384]	1	2	$[1, 0]$	[512]
PSTOp3b	$4r_o$	1	[768]	1	1	$[1, 1]$	[1024]
PSTOp4	$4r_o$	2	[1536]	0	1	$[0, 0]$	[2048]
FC	$2048 \rightarrow$ # action classes						

PSTNet++ for 4D Semantic Segmentation

Encoding	$\mathcal{S}-$MLP			$\mathcal{T}-$MLP			
	r_s	s_s	MLP	r_t	s_t	p	MLP
PSTOp1	r_o	4	[32, 32]	0	1	$[0, 0]$	[128]
PSTOp2	$2r_o$	4	[64, 64]	0	1	$[0, 0]$	[256]
PSTOp3	$4r_o$	4	[128, 128]	1	1	$[0, 0]$	[512]
PSTOp4	$8r_o$	2	[256, 256]	0	1	$[0, 0]$	[1024]

Decoding	$\mathcal{T}-$MLP$'$				$\mathcal{S}-$MLP$'$
	r_t	s_t	p	MLP	MLP
PSTTransOp4	0	1	$[0, 0]$	[256]	[256]
PSTTransOp3	1	1	$[0, 0]$	[256]	[256]
PSTTransOp2	0	1	$[0, 0]$	[256]	[128]
PSTTransOp1	0	1	$[0, 0]$	[128]	[128]
1D Convolution	$L \times N \times 128 \rightarrow L \times N \times$ # semantic classes				

The temporal padding p is split as $[p_1, p_2]$, where p_1 and p_2 denote the beginning padding and the ending padding, respectively. As 3D action recognition usually uses many frames, to improve computational efficiency and reduce memory usage, we only use only layer in each MLP of our 3D action recognition architecture.

connections are added between the PSTOp layers and PSTTransOp layers. After the last PSTTransOp layer, a 1D convolution layer is appended for semantic predictions. Our PSTNet++ for 4D semantic segmentation is illustrated in Appendix C, available in the online supplemental material.

In addition, batch normalization and ReLU activation are inserted between layers. The specs of our PSTNet++ architectures for 3D action recognition and 4D semantic segmentation are described in Table 2.

4 Experiments

In this section, we conduct a video-level classification, i.e., 3D action recognition, and a point-level classification, i.e., 4D semantic segmentation, to demonstrate the effectiveness and superiority of PSTNet++. We also apply our PSTNet++ to scene flow estimation to explicitly show its ability for motion modeling.

To build a unified network and train the network with mini-batch, the number of spatial neighbors needs to be fixed. We follow existing point-based works [6], [18], [19], [20] to randomly sample a fixed number of spatial neighbors. For 3D action recognition, the number is set to 9. For 4D semantic segmentation, we follow [5], [6] to set the number to 32. If the actual number of neighbors of a point is less than the set number, we randomly repeat some neighbors.

We train our models for 35 epochs with the SGD optimizer. Learning rate is set to 0.01, and decays with a rate of 0.1 at the 10th epoch and the 20th epoch, respectively.

4.1 3D Action Recognition

To show the effectiveness in video-level classification, we apply our PSTNet++ to 3D action recognition. Following [5], [6], [30], we sample 2,048 points for each frame. Point cloud videos are split into multiple clips (with a fixed number of frames) as inputs. For training, video-level labels are used as clip-level labels. For evaluation, the mean of the clip-level predicted probabilities is used as the video-level prediction. Point colors are not used.

4.1.1 MSR-Action3D

The MSR-Action3D [40] dataset consists of 567 Kinect v1 depth videos, with 20 action categories and 23K frames in total. We use the same training/test split as previous works [5], [6], [39]. Following [5], [6], batch size is set to 16. We set the initial spatial radius r_o to 0.7. We conduct experiments with 10 times and report the mean.

Accuracy comparison with the state-of-the-art. We compare our PSTNet++ with skeleton-based, depth-based and point-based 3D action recognition methods on this dataset. The accuracy comparisons are reported in Table 3. The proposed PSTNet++ significantly outperforms all the state-of-the-art methods, demonstrating the superiority of our PSTOp on feature extraction. For example, when using 24 frames, our PSTNet++ outperforms MeteorNet by 4.18%.

Comparison with skeleton-based and depth-based methods. Skeleton-based methods ignore the object and scene information during action recognition. In contrast, our point-based method is able to take the advantage of all the information in a video and thus enhance action reasoning. Depth-based methods project width and height but treat depth as pixel value. As width, height and depth are treated in different ways, depth-based methods may increase the challenge for neural networks to understand 3D data. Moreover, it is difficult to capture precise 3D motion from 2D depth frames. In contrast, our point-based method treats width, height, and depth equally, and thus properly models 3D data and captures 3D motion.

Memory usage and computational efficiency. To investigate the running time, we conduct experiments using 1 Nvidia Quadro RTX 6000 GPU. As shown in Table 4, our PSTNet+ + uses fewer parameters and is more efficient than MeteorNet. This is because, on one hand, MeteorNet employs three-layer MPLs while the MLPs in our PSTNet++ only contain one layer. One the other hand, when applying to 3D action recognition, to better capture temporal structure, besides each spatio-temporal neighbor's feature and 4D displacement, MeteorNet also feeds the anchor's feature into the input of its MLPs. By contrast, benefiting from decomposing space and time, the $\mathcal{S}-$MLPs in our PSTNet++ only

TABLE 3
Action Recognition Accuracy (%) on the MSR-Action3D [40] Dataset

Method	Input	# Frames	Accuracy
Vieira *et al.* [37]	depth	20	78.20
Kläser *et al.* [38]	depth	18	81.43
Actionlet [39]	skeleton	all	88.21
PointNet++ [18]	point	1	61.61
MeteorNet [5]	point	4	78.11
		8	81.14
		12	86.53
		16	88.21
		24	88.50
PSTNet [6]	point	4	81.14
		8	83.50
		12	87.88
		16	89.90
		24	91.20
PSTNet++ (ours)	point	4	81.53
		8	83.50
		12	88.15
		16	90.24
		24	**92.68**

TABLE 4
Comparison of Memory Usage and Computational Efficiency on 3D Action Recognition

Method	# Parameters (M)	Running time (ms)	
		16 frame	24 frame
MeteorNet [5]	17.6	54.46	80.11
PSTNet [6]	**8.44**	31.92	43.88
PSTNet++ (ours)	**8.44**	**31.76**	**43.19**

need to encode each spatial neighbor's feature and 3D displacement, and thus save parameters and computation.

When using one-layer MPLs for 3D action recognition, our temporal operation is identical to the temporal convolution of PSTNet. To highlight the difference in the spatial modeling, we reformulate their spatial operations as follows,

$$\begin{aligned}\text{PSTNet}: &\sum \mathcal{S}^{(\delta_x,\delta_y,\delta_z)} \cdot \left(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}\right)\\ &=\sum \left(\boldsymbol{W}_1 \cdot (\delta_x,\delta_y,\delta_z)^T \cdot \mathbf{1} \odot \boldsymbol{W}_2\right) \cdot \boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}\\ &=\sum \left(\boldsymbol{W}_1 \cdot (\delta_x,\delta_y,\delta_z)^T\right) \odot \left(\boldsymbol{W}_2 \cdot \boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}\right)\\ \text{PSTNet++}:&\text{MAX }\ \mathcal{S}\text{–MLP}\left(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)},\delta_x,\delta_y,\delta_z\right)\\ &= \text{MAX }\boldsymbol{W} \cdot \left(\boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)},\delta_x,\delta_y,\delta_z\right)\\ &= \text{MAX}\left(\boldsymbol{W}_1 \cdot (\delta_x,\delta_y,\delta_z)^T\right) + \left(\boldsymbol{W}_2 \cdot \boldsymbol{F}_t^{(x+\delta_x,y+\delta_y,z+\delta_z)}\right)\end{aligned}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{C_m\times 3}$, $\boldsymbol{W}_2 \in \mathbb{R}^{C_m\times C}$, $\boldsymbol{W} = [\boldsymbol{W}_1, \boldsymbol{W}_2]$, $\mathbf{1} = (1,\ldots,1) \in \mathbb{R}^{1\times C}$ is for broadcasting, $\cdot$ is matrix multiplication, $\odot$ is element-wise product, and C_m is the dimension of the intermediate feature. As shown in the above equations, the spatial operation of our PSTNet++ uses the same number of parameters as the spatial convolution of PSTNet. The difference is that PSTNet is based on sum pooling and element-wise product while PSTNet++ is based max pooling and addition. Because addition requires less computation than element-wise product, the proposed PSTNet++ is slightly faster than PSTNet.

TABLE 5
Influence of Spatio-Temporal Hierarchy (Hier) and Spatio-Temporal Decomposition (Decom) on Point Cloud Video Modeling

Method	$\mathcal{S}$ hier	$\mathcal{T}$ hier	Decom	Accuracy (%)		
				16-frame	24-frame	↑
MeteorNet [5]	✓			88.21	88.50	0.29
PSTNet++	✓	✓		89.20	90.59	1.39
	✓		✓	88.85	89.90	1.05
	✓	✓	✓	90.24	92.68	2.44

Note that, in order not to decompose space and time in PSTNet++, we use a similar operation to MeteorNet, which treats a 3D point cloud video as a 4D point set so that pint coordinates and timestamps can be modeled together. The MeteorNet method can be seen as a baseline in which only spatial hierarchy is constructed. Experiments are conducted on MSR-Action3D. The "↑" indicates the improvement of accuracy from 16 frames to 24 frames.

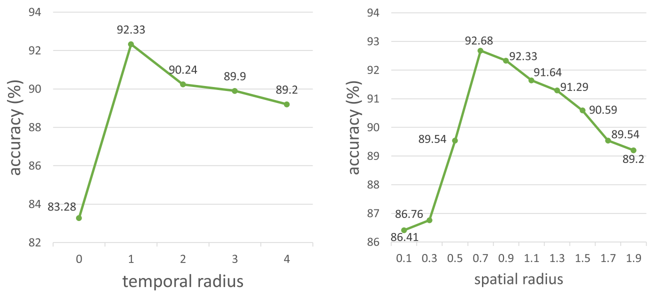


Fig. 4. Influence of temporal radius and initial spatial radius on MSR-Action3D with 24 frames.

What does PSTOp learn? To investigate what PSTOp learns, we visualize the output of each layer of our PSTNet++ for 3D action recognition in Fig. 5. Because we use ReLU as the activation function, all outputs are greater than zero and large outputs indicate high activation. To visualize the outputs, we squeeze point feature vectors to scalars via l_1 norm. As expected, PSTOp outputs higher activation on moving areas. This demonstrates that our PSTOp captures the most informative clues in action reasoning.

4.1.2 N-UCLA and UWA3DII

The N-UCLA [7] dataset contains 1475 action videos, with 10 action categories. These videos are captured using Microsoft Kinect v1 from 3 different viewpoints. The cross-view evaluation setting is used for test. The UWA3DII [8] dataset contains 1075 videos, with 30 categories. The video samples are captured using Microsoft Kinect v1. Batch size is set to 16. The initial spatial radius r_o to 0.5.

The experimental results are listed in Table 6. The proposed PSTNet++ outperforms all the state-of-the-art methods, demonstrating the effectiveness of our method. For example, with the UWA3DII dataset, our PSTNet++ outperforms 3DV-PointNet++ by 3.0%.

4.1.3 NTU RGB+D 60 and NTU RGB+D 120

The NTU RGB+D 60 [44] is the second largest dataset for 3D action recognition. It consists of 56K videos, with 60 action categories and 4M frames in total. The videos are captured using Kinect v2, with 3 cameras and 40 subjects (performers).

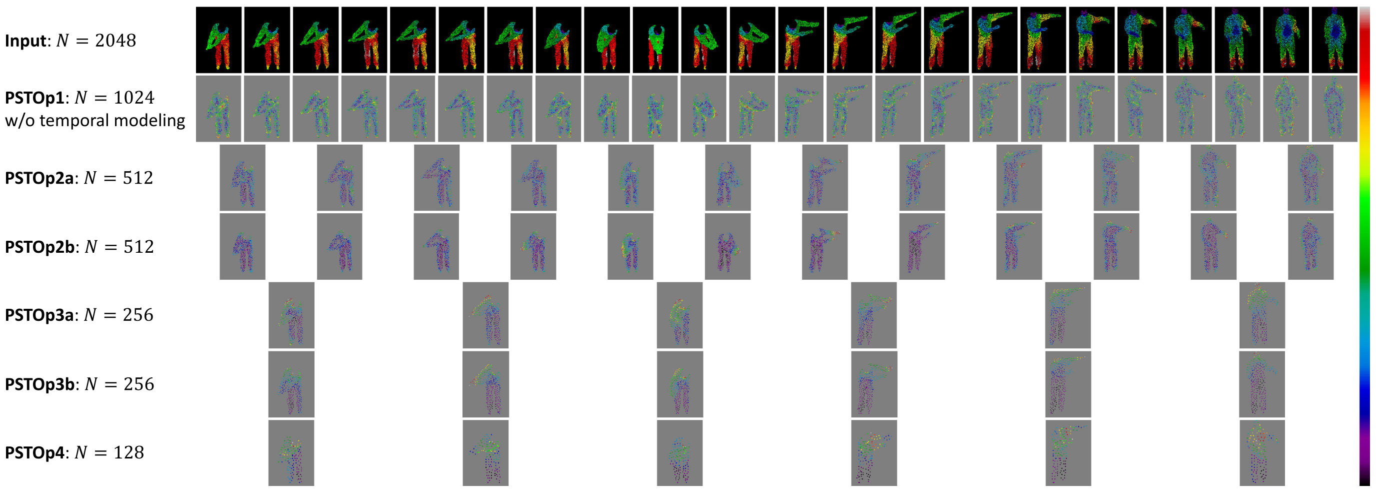


Fig. 5. Visualization of the output of each PSTOp layer in PSTNet++. Top: input point cloud video, where color encodes depth. Bottom: output of PSTOp, where brighter color indicates higher activation. For the input point cloud video, color encodes depth. For the outputs, brighter color indicates higher activation. Input videos consist of 24 frames. Due to the spatial subsampling s_s and the temporal stride s_t, points and frames progressively decrease along the network layers. Interestingly, PSTOp outputs high activation to salient motion, which supports our intuition that PSTOp effectively captures dynamics of point cloud videos. Note that, PSTOp1 does not capture temporal correlation because its temporal radius $r_t = 0$. In this case, PSTOp1 focuses on the appearance and therefore outputs high activation to the performer contour. Best viewed in color.

TABLE 6
Action Recognition Accuracy (%) on N-UCLA [7] and UWA3DII [8]

Method	Input	N-UCLA	UWA3DII
HON4D [41]	depth	39.9	28.9
SNV [42]	depth	42.8	29.9
AOG [7]	depth	53.6	26.7
HOPC [8]	depth	80.0	52.2
MVDI [43]	depth	84.2	68.1
3DV-PointNet++ [30]	voxel + point	95.3	73.2
PSTNet [6]	point	95.1	75.6
PSTNet++	point	**95.5**	**76.2**

TABLE 7
Influence of Temporal Modeling on PSTNet++ in 3D Action Recognition

Method	Point Cloud Video Distance	Acc (%)
w/o temporal	$d_S(\mathsf{S}, \mathsf{S}') = \sum_{t=1}^{L} d_H(S_t, S_t')$	87.6
w/ temporal	$d_S(\mathsf{S}, \mathsf{S}') = \sum_{t=1}^{L} \lambda_t(d_H(S_t, S_t'))$	**91.4**

Experiments are conducted on NTU RGB+D 60 with the cross-subject evaluation metric.

The dataset defines two types of evaluation, i.e., cross-subject and cross-view. The cross-subject evaluation splits the 40 performers into training and test groups. Each group consists of 20 performers. The cross-view evaluation uses all the samples from camera 1 for testing and samples from cameras 2 and 3 for training. The NTU RGB+D 120 [45] dataset, the largest dataset for 3D action recognition, is an extension of NTU RGB+D 60. It consists of 114K videos, with 120 action categories and 8M frames in total. The videos are captured with 106 performers and 32 collection setups (locations and backgrounds). Besides cross-subject evaluation, the dataset defines a new evaluation setting, i.e., cross-setup, where 16 setups are used for training, and the others are used for testing.

Batch size is set to 16. We set the frame sampling rate and the initial spatial radius r_o to 2 and 0.1, respectively.

As indicated in Table 8, PSTNet++ outperforms other approaches. Particularly, as indicated by the cross-setup evaluation on NTU RGB+D 120, PSTNet++ outperforms the second best 3DV-PointNet++ [30] by 6.2%. Moreover, compared to PointNet++ [18], which only exploit appearance information, our PSTNet++ effectively captures temporal correlation and thus significantly improves recognition accuracy. For example, the cross-subject evaluation results on NTU RGB+D 60 shows PSTNet++ outperforms PointNet++ by a large margin of 11.3%.

4.2 4D Semantic Segmentation

To demonstrate that our method can be generalized to point-level dense prediction tasks, we apply PSTNet++ to 4D semantic segmentation. Following the works [4], [5], [6], we conduct experiments on point cloud videos with length of 3 frames.

Synthia 4D [4] uses the Synthia dataset [50] to create 3D videos. The Synthia 4D dataset includes 6 sequences of driving scenarios, where both objects and cameras are moving. Each sequence consists of 4 stereo RGB-D images taken from the top of a moving car. Following [5], we reconstruct 3D point cloud sequences from RGB and depth videos, and use the same training/validation/test split, with 19,888/815/1,886 frames, respectively. Following [5], batch size is set to 12. We set the initial spatial radius r_o to 0.9.

We compare PSTNet++ with the state-of-the-art voxel-based and point-based methods on this dataset. Note that, although semantic segmentation can be achieved from a single frame, exploring temporal consistency would facilitate exploring the structure of scenes, thus improving segmentation accuracy and robustness to noise. As seen in Table 10, our PSTNet++ outperforms the state-of-the-art. The Synthia 4D dataset contains 12 categories. Our PSTNet++ achieves four best accuracies among them. Moreover, our method

TABLE 8
Action Recognition Accuracy (%) on the NTU RGB+D 60 [44] and NTU RGB+D 120 [45] Datasets

Method	Input	NTU RGB+D 60		NTU RGB+D 120	
		Subject	View	Subject	Setup
Body Pose Evolution Map [46]	skeleton	-	-	64.6	66.9
2s-AGCN [47]	skeleton	88.5	95.1	-	-
DGNN [48]	skeleton	89.9	96.1	-	-
Wang *et al.* [49]	depth	87.1	84.2	-	-
MVDI [43]	depth	84.6	87.3	-	-
NTU RGB+D 120 Baseline [45]	depth	-	-	48.7	40.1
PointNet++ (appearance) [18]	point	80.1	85.1	72.1	79.4
3DV (motion) [30]	voxel	84.5	95.4	76.9	92.5
3DV-PointNet++ [30]	voxel + point	88.8	96.3	82.4	93.5
PSTNet [6]	point	90.5	96.5	87.0	**93.8**
PSTNet++ (ours)	point	**91.4**	**96.7**	**88.6**	**93.8**

saves 0.11M, relative 6% of parameters compared to MeteorNet [5]. We visualize a few segmentation examples in Fig. 6.

4.3 Scene Flow Estimation

Following the previous PSTNet [6] work, we also apply our PSTNet++ to scene flow estimation. We follow the setting proposed by MeteorNet [5], i.e., given a point cloud sequence, estimating a flow vector for every point in the last frame. However, because our point tube is constructed according to the middle frame, which is not applicable to the last-frame scene flow estimation, we follow [6] to adapt temporal anchor frame selection and spatial anchor point transferring. Specifically, we select the last frame in each tube as the anchor frame. Then, after spatial sampling, each anchor point is transferred to its previous nearest l frames. Following [5], [6], we first train our model on a FlyingThings3D dataset according to the synthetic method in [35], and then fine-tune the model on a KITTI scene flow dataset [5]. Point tracking is not used. As shown in Table 9, our PSTNet++ achieves the highest accuracy on scene flow estimation.

4.4 Ablation Study

4.4.1 Spatio-Temporally Hierarchical Modeling

Because points are not consistent and even flow in and out of a spatial region, embedding points in a spatially local area along an entire video handicaps capturing accurate local dynamics of point clouds. To address this problem, we construct temporal hierarchy in point cloud modeling. As shown in Table 5, even though without the decomposition of spatio-temporal encoding, our PSTNet++ still outperforms MeteorNet. Moreover, without temporally hierarchical modeling, MeteorNet only achieves a slight improvement of 0.29% from 16 frames to 24 frames. By contrast, PSTNet++ achieves an improvement of 1.39%, showing the effectiveness of temporally hierarchical modeling when encoding long videos.

4.4.2 Decomposition of Spatio-Temporal Encoding

To investigate our PSTNet++ without decomposing space and time, we treats a 3D point cloud video as a 4D point set so that our method can model pint coordinates and time-stamps together. This makes our method similar to MeteorNet, which is based on the max-based video distance metric Eq. (3). As show in Table. 5, even though without temporal hierarchy, decomposing space and time helps our PSTNet++ to outperform MeteorNet. Moreover, from 16 frames to 24 frames, our PSTNet++ achieves an improvement of 1.05%, while MeteorNet only achieves a slight improvement of 0.29%. The supports our motivation that, by decomposing space and time, we can leverage our sum-based video distance metric Eq. (5), which is able to reflect the global difference of two point cloud videos. However, the max-based video distance metric used in MeteorNet tends to focus on a specific moment and miss the information in other frames and thus cannot adequately capture the temporal structure of long videos.

4.4.3 Temporal Modeling

In this paper, we propose to use the sum of per-frame Hausdorff distance as the point cloud video distance metric, as shown in Eq. (4). In this way, the difference at each moment in a video is involved and considered. Moreover, we further add the temporal encoding, i.e., $\{\lambda_t\}$, into Eq. (4) to capture the temporal structure, resulting in Eq. (5). In this section, we investigate the influence of the temporal encoding. As shown in Table 7, exploiting the temporal encoding effectively improves action recognition.

4.4.4 Clip Length

Usually, information is not equally distributed in videos along time. Short point cloud clips may miss key frames and thus confuse models as noise. Therefore, as shown in Table 3, increasing clip length (i.e., the number of frames) benefits models for action recognition.

4.4.5 Temporal Radius

The temporal radius r_t controls the temporal dynamics modeling of point cloud videos. Fig. 4 shows the accuracy on MSR-Action3D with different r_t.

TABLE 9
Scene Flow Estimation Accuracy on the KITTI Scene Flow Dataset [5]

Method	Input	# Frames	End-Point-Error
FlowNet3D [35]	points	2	0.287
MeteorNet [5]	points	3	0.282
PSTNet [6]	points	3	0.278
PSTNet++ (ours)	points	3	**0.276**

TABLE 10
Semantic Segmentation Result (mIoU %) Details on the Synthia 4D Dataset [4]

Method	Input	#Frms	#Params	Bldn	Road	Sdwlk	Fence	Vegittn	Pole	Car	T. Sign	Pedstrn	Bicycl	Lane	T. Light	Average
3D MinkNet14 [4]	voxel	1	19.31 M	89.39	97.68	69.43	86.52	98.11	97.26	93.50	79.45	92.27	0.00	44.61	66.69	76.24
4D MinkNet14 [4]	voxel	3	23.72 M	90.13	98.26	73.47	87.19	99.10	97.50	94.01	79.04	**92.62**	0.00	50.01	68.14	77.46
PointNet++ [18]	point	1	0.88 M	96.88	97.72	86.20	92.75	97.12	97.09	90.85	66.87	78.64	0.00	72.93	75.17	79.35
MeteorNet [5]	point	3	1.78 M	**98.10**	97.72	88.65	94.00	97.98	**97.65**	93.83	**84.07**	80.90	0.00	71.14	**77.60**	81.80
PSTNet [6]	point	3	1.67 M	96.91	**98.33**	90.83	95.00	96.96	97.61	95.15	77.45	85.68	0.00	**75.71**	77.28	82.24
PSTNet++ (ours)	point	3	1.67 M	97.62	98.23	**90.88**	**96.01**	**99.34**	97.30	**97.82**	77.46	87.61	0.00	75.10	73.70	**82.60**

#Frms: number of frames. #Params: number of parameters.

When r_t is set to 0, temporal correlation is not captured. However, PSTNet++ can still observe 24 frames and the pooling operation allows PSTNet++ to capture the pose information of an entire clip. Moreover, some actions (e.g., "golf swing") can be easily recognized by a certain pose, and thus PSTNet++ with $r_0 = 1$ can still achieve satisfactory accuracy.

When r_t is greater than 1, PSTNet++ models temporal dynamics and therefore improves accuracy on actions that rely on motion or trajectory reasoning (e.g., "draw x", "draw tick" and "draw circle").

When r_t is greater than 3, the accuracy decreases. This mainly depends on motion in sequences. Because most actions in MSR-Action3D are fast (e.g., "high arm wave"), using smaller temporal kernel size facilitates capturing fast motion, and long-range temporal dependencies will be captured in high-level layers. Since we aim to present generic point-based operations, we do not tune the kernel size for each action but use the same size.

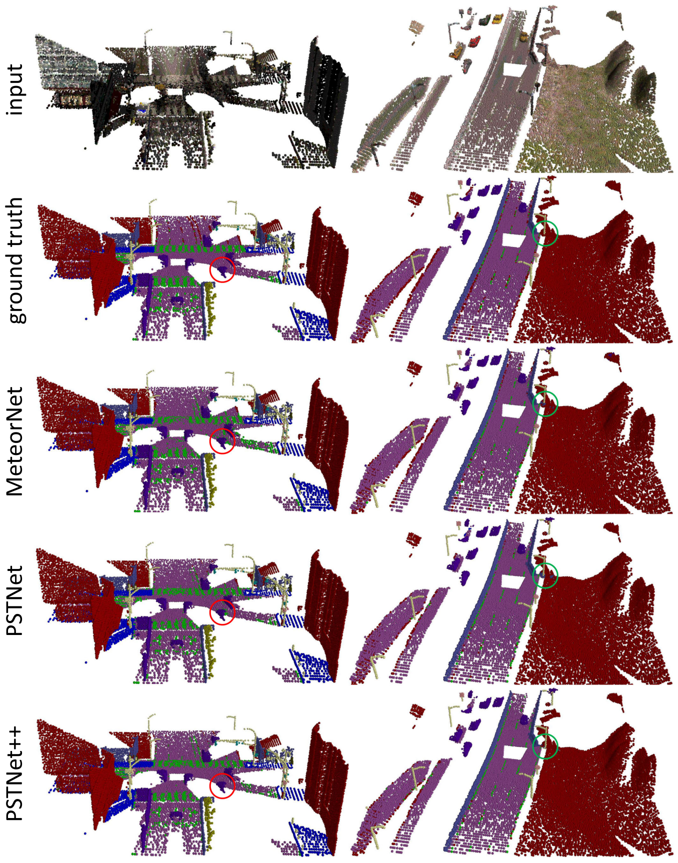


Fig. 6. Visualization of semantic segmentation examples from Synthia 4D. All of the methods achieve satisfactory results. However, our PSTNet++ performs better than MeteorNet and PSTNet on some small areas.

4.4.6 Spatial Radius

The spatial radius controls the range of the spatial structure to be modeled, and thus is import for extracting the spatially-local correlation. As shown in Fig. 4, using too small spatial radius cannot capture sufficient structure information while using large spatial radius will decrease the discriminativeness of local structure for temporal modeling.

5 Conclusion

In this paper, we propose an effective 3D point spatio-temporal operation (PSTOp) to learn informative representations from point cloud videos. Considering 3D points are spatially irregular and temporally ordered, we decouple the feature extraction in two successive steps, i.e., a spatial operation and a temporal operation, with the help of our newly introduced point tubes. Then, we develop a point spatio-temporal transposed operation (PSTTransOp) for point-level dense prediction tasks. We also demonstrate that by incorporating the proposed operations into deep network, dubbed PSTNet++, our method is competent to address various point-based tasks. Extensive experiments demonstrate that our PSTNet++ significantly improves the recognition and segmentation performance by effectively modeling point cloud videos. Moreover, we theoretically prove that our PSTNet++ is able to effectively model point cloud videos.

References

[1] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.
[2] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.
[3] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet?," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6546–6555.
[4] C. B. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convNets: Minkowski convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3075–3084.
[5] X. Liu, M. Yan, and J. Bohg, "MeteorNet: Deep learning on dynamic 3D point cloud sequences," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9245–9254.
[6] H. Fan, X. Yu, Y. Ding, Y. Yang, and M. Kankanhalli, "PSTNet: Point spatio-temporal convolution on point cloud sequences," in *Proc. Int. Conf. Learn. Representations*, 2021.

[7] J. Wang, X. Nie, Y. Xia, Y. Wu, and S. Zhu, "Cross-view action modeling, learning, and recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2649–2656.

[8] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. S. Mian, "Histogram of oriented principal components for cross-view action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 12, pp. 2430–2443, Dec. 2016.

[9] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., 2014, pp. 568–576.

[10] L. Wang *et al.*, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 20–36.

[11] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4694–4702.

[12] H. Fan, Z. Xu, L. Zhu, C. Yan, J. Ge, and Y. Yang, "Watching a small portion could be as good as watching all: Towards efficient video classification," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 705–711.

[13] H. Fan, X. Chang, D. Cheng, Y. Yang, D. Xu, and A. G. Hauptmann, "Complex event detection by identifying reliable shots from untrimmed videos," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 736–744.

[14] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6450–6459.

[15] T. Zhuo, Z. Cheng, P. Zhang, Y. Wong, and M. S. Kankanhalli, "Explainable video action reasoning via prior knowledge and state transitions," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 521–529.

[16] H. Fan, T. Zhuo, X. Yu, Y. Yang, and M. Kankanhalli, "Understanding atomic hand-object interaction with human intention," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Feb. 11, 2021. doi: 10.1109/TCSVT.2021.3058688.

[17] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.

[18] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. 31st Int. Conf. Neural Process. Syst.*, 2017, pp. 5099–5108.

[19] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 828–838.

[20] W. Wu, Z. Qi, and F. Li, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9621–9630.

[21] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 6410–6419.

[22] P. Ren *et al.*, "A comprehensive survey of neural architecture search: Challenges and solutions," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 76:1–76:34, 2021.

[23] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 146:1–146:12, 2019.

[24] A. Dai, A. X. Chang, M. Savva, M. Halber, T. A. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2017, pp. 2432–2443.

[25] L. Yu, X. Li, C. Fu, D. Cohen-Or , and P. Heng, "PU-Net: Point cloud upsampling network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2018, pp. 2790–2799.

[26] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, 2017, pp. 6526–6534.

[27] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9276–9285.

[28] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3569–3577.

[29] H. Fan and Y. Yang, "PointRNN: Point recurrent neural network for moving point cloud processing," 2019, *arXiv:1910.08287v2*.

[30] Y. Wang *et al.*, "3DV: 3D dynamic voxel for action recognition in depth video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 508–517.

[31] H. Fan, Y. Yang, and M. S. Kankanhalli, "Point 4D transformer networks for spatio-temporal modeling in point cloud videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14204–14213.

[32] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger, "Occupancy flow: 4D reconstruction by learning particle dynamics," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 5378–5388.

[33] L. Prantl, N. Chentanez, S. Jeschke, and N. Thuerey, "Tranquil clouds: Neural networks for learning temporally coherent features in point clouds," in *Proc. Int. Conf. Learn. Representations*, 2020.

[34] D. Rempe, T. Birdal, Y. Zhao, Z. Gojcic, S. Sridhar, and L. J. Guibas, "CaSPR: Learning canonical spatiotemporal point cloud representations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020.

[35] X. Liu, C. R. Qi, and L. J. Guibas, "FlowNet3D: Learning scene flow in 3D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 529–537.

[36] X. Gu, Y. Wang, C. Wu, Y. J. Lee, and P. Wang, "HPLFlowNet: Hierarchical permutohedral lattice flowNet for scene flow estimation on large-scale point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3254–3263.

[37] A. W. Vieira, E. R. Nascimento, G. L. Oliveira, Z. Liu, and M. F. M. Campos, "STOP: Space-time occupancy patterns for 3D action recognition from depth map sequences," in *Prog. Pattern Recognit., Image Anal., Comput. Vis. Appl. - 17th Iberoamerican Cong.*, vol. 7441, 2012, pp. 252–259.

[38] A. Kläser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *Proc. Brit. Mach. Vis. Conf.*, M. Everingham, C. J. Needham, and R. Fraile, Eds., 2008, pp. 1–10.

[39] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1290–1297.

[40] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2010, pp. 9–14.

[41] O. Oreifej and Z. Liu, "HON4D: histogram of oriented 4D normals for activity recognition from depth sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 716–723.

[42] X. Yang and Y. Tian, "Super normal vector for activity recognition using depth sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 804–811.

[43] Y. Xiao, J. Chen, Y. Wang, Z. Cao, J. T. Zhou, and X. Bai, "Action recognition for depth video using multi-view dynamic images," *Inf. Sci.*, vol. 480, pp. 287–304, 2019.

[44] A. Shahroudy, J. Liu, T. Ng, and G. Wang, "NTU RGB+D: A large scale dataset for 3D human activity analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1010–1019.

[45] J. Liu, A. Shahroudy, M. Perez, G. Wang, L. Duan, and A. C. Kot, "NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2684–2701, Oct. 2020.

[46] M. Liu and J. Yuan, "Recognizing human actions as the evolution of pose estimation maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1159–1168.

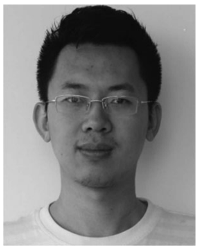
[47] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12 026–12 035.

[48] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with directed graph neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7912–7921.

[49] P. Wang, W. Li, Z. Gao, C. Tang, and P. O. Ogunbona, "Depth pooling based large-scale 3-D action recognition with convolutional neural networks," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1051–1061, May 2018.

[50] G. Ros, L. Sellart, J. Materzynska, D. Vázquez, and A. M. López, "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3234–3243.

Hehe Fan received the PhD degree in faculty of engineering and information technology from the University of Technology Sydney, Sydney, NSW, Australia, in 2020. He is currently a research fellow with the School of Computing, National University of Singapore, Singapore. His research interests include deep learning, computer vision and multimedia, with an emphasis on video analysis, dynamic point cloud, unsupervised learning, and vision-language applications.



Xin Yu received the BS degree in electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2009 and the dual PhD degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2015 and from the College of Engineering and Computer Science, Australian National University, Canberra, ACT, Australia, in 2019. He is currently a lecturer with the University of Technology Sydney. His research interests include computer vision and image processing.

Yi Yang (Senior Member, IEEE) received the PhD degree in computer science from Zhejiang University, Hangzhou, China, in 2010. He was a postdoctoral researcher with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA. His research interests include machine learning and its applications to multimedia content analysis and computer vision, including multimedia indexing and retrieval, surveillance video analysis, and video content understanding.

Mohan Kankanhalli (Fellow, IEEE) received the BTech degree from IIT Kharagpur and the MS and PhD degrees from the Rensselaer Polytechnic Institute. He is currently the Provost's chair professor with the Department of Computer Science, National University of Singapore (NUS). He is the director of N-CRiPT and also the dean with the School of Computing, NUS. His research interests include multimedia computing, multimedia security and privacy, image or video processing, and social media analysis. He is active with the Multimedia Research Community and on the editorial boards of several journals.