3D Vision with Transformers: A Survey

Jean Lahoud, Jiale Cao, Fahad Shahbaz Khan, Hisham Cholakkal, Rao Muhammad Anwer, Salman Khan, and Ming-Hsuan Yang

Abstract—The success of the transformer architecture in natural language processing has recently triggered attention in the computer vision field. The transformer has been used as a replacement for the widely used convolution operators, due to its ability to learn long-range dependencies. This replacement was proven to be successful in numerous tasks, in which several state-of-the-art methods rely on transformers for better learning. In computer vision, the 3D field has also witnessed an increase in employing the transformer for 3D convolution neural networks and multi-layer perceptron networks. Although a number of surveys have focused on transformers in vision in general, 3D vision requires special attention due to the difference in data representation and processing when compared to 2D vision. In this work, we present a systematic and thorough review of more than 100 transformers methods for different 3D vision tasks, including classification, segmentation, detection, completion, pose estimation, and others. We discuss transformer design in 3D vision, which allows it to process data with various 3D representations. For each application, we highlight key properties and contributions of proposed transformer-based methods. To assess the competitiveness of these methods, we compare their performance to common non-transformer methods on 12 3D benchmarks. We conclude the survey by discussing different open directions and challenges for transformers in 3D vision. In addition to the presented papers, we aim to frequently update the latest relevant papers along with their corresponding implementations at: https://github.com/lahoud/3d-vision-transformers.

Index Terms—3D vision, transformers, survey, point cloud, self-attention, RGB-D, voxels

1 Introduction

One fundamental problem in computer vision is to understand the scenes and objects in the three-dimension space. It allows a compact representation of relationships and provides the ability to navigate and manipulate in the real world. 3D vision plays an important role in various domains and includes applications in autonomous driving, robotics, remote sensing, medical treatment, augmented reality, design industry, among many others. There has been an increasing interest in the 3D field due to numerous reasons: (1) development of various 3D capture sensors, e.g., LiDAR and RGB-D sensors, (2) introduction of numerous large-scale 3D geometry datasets that were collected and labeled in 3D, and (3) advances in 3D deep learning methods.

Common approaches to 3D deep learning methods employ deep convolution neural networks (CNNs) and multi-layer perceptrons (MLPs). Nevertheless, transformer-based architectures that use the attention mechanism have shown a strong contender for such methods in various fields, such as natural language processing (NLP) and 2D image processing. While convolution operators have limited receptive fields and translation equivariance properties, the attention mechanism operates globally and can thus encode long-range dependencies, allowing attention-based methods to learn richer feature representations.

Witnessing the success of transformer-based architectures in the image domain, numerous 3D vision methods have recently adopted transformers in the model designs. These architectures have been proposed as a solution for most common 3D vision applications. In 3D, the transformer has replaced previous learning methods or supplemented them, benefiting from its ability to capture long-range information and learn task-specific inductive biases.

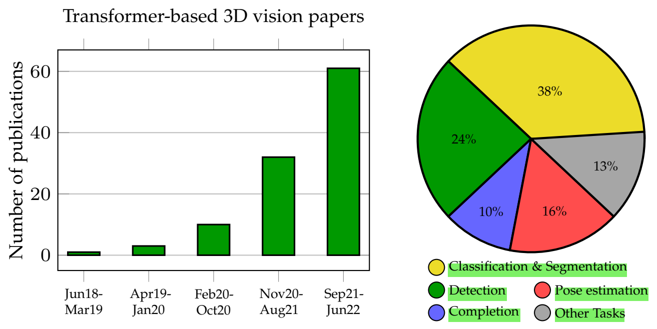


Fig. 1. (Left) A bar graph showing the number of transformer-based 3D vision papers in recent years. For papers available on arXiv, we use the first submission date; otherwise, the journal or conference publication date is used. A consistent growth reflects the increased attention to the transformer architecture in the recent literature. (Right) A pie chart showing the percentage of papers in this survey for each application.

Given the increasing interest of transformers in 3D vision (Fig. 1, left), a survey that gives an overview of the available methods is of great importance to give a holistic view of this emerging field. In this survey, we review methods that use transformers for 3D vision tasks, including classification, segmentation, detection, completion, pose estimation, and others (Fig. 1, right). We highlight transformer design choices in 3D vision, which allows it to process data with various 3D representations. For each application, we discuss key properties and contributions of proposed transformer-based methods. Finally, we compare their performance to alternative methods on the widely used 3D

- J. Lahoud, F. Khan, H. Cholakkal, R. Anwer, and S. Khan are with Mohamed bin Zayed University of Artificial Intelligence, UAE. E-mail: {firstname.lastname}@mbzuai.ac.ae
- J. Cao is with the School of Electrical and Information Engineering, Tianjin University, China. E-mail: connor@tju.edu.cn
- F. Khan is also with Linköping University, Sweden
- S. Khan is also with Australian National University, Australia
- M.-H. Yang is with University of California at Merced, Yonsei University, and Google. Email:mhyang@ucmerced.edu

datasets/benchmarks to assess the competitiveness of the transformer integration in this field.

We note that numerous surveys have studied deep learning methods in 3D vision. Among these surveys, many studies that have been published provide an overall review of methods that process 3D data [1], [2], [3], [4]. Other studies focus on specific 3D vision applications, such as segmentation [5], [6], [7], classification [8], or detection [9], [10]. Furthermore, some surveys examine 3D deep learning methods from a representation perspective [11], [12], and others limit their studies to a specific data input sensor [10], [13]. Given that most of the surveys were published prior to the recent success of the transformer architecture, attention to the transformer-based architectures is still missing.

With the plethora of recent vision methods that rely on the attention mechanism and the transformer architecture, many works have emerged that survey these methods. Some of these works consider transformers in vision in general [14], [15], [16], [17], [18], while others focus on a specific aspect, such as efficiency [19], or a specific application, such as video [20] or medical imaging [21]. Considering the differences between the 2D and 3D data representation and processing, special attention to transformers applied to 3D vision applications is essential. Thus, we focus on transformer architectures applied in the 3D vision field.

This survey includes methods that employ the transformer architecture with 3D inputs and/or outputs. 3D data can be obtained with numerous sensors, such as RGB-D sensors for indoors, LiDAR for outdoors, as well as specialized medical sensors. We include methods that either use point clouds as input or dense 3D grid. A dense 3D grid can also be obtained by taking images at different slices, which is common in medical imaging. In addition, representative methods that apply the transformer architectures to other input data, such as multi-view images or bird-eye view images, and generate output in 3D are also included.

2 Preliminaries

Significant advances have been recently made in the field of 3D computer vision. In this section, we first review different representations of 3D data, as well as numerous processing techniques that enable learning from such data. For the transformer model, we present its main component (attention), architecture, and valuable properties.

2.1 3D Representation

Images and videos have an inherent natural representation characterized by pixels on a standard grid. On the other hand, such organized grid structure does not exist for 3D geometry. In this section, we discuss widely-used representations of 3D data which allow employing different deep learning algorithms/techniques. Fig. 2 shows different 3D representations of the Stanford bunny.

Multi-view Representation. A 3D shape can be represented by a set of 2D images captured from different viewpoints. Compared to other representations in 3D, this representation is relatively efficient mainly due to having one less dimension, which yields a smaller data size. With this representation, one can exploit 2D learning methods for 3D analysis. Capturing such data is readily obtainable using 2D cameras, as opposed to 3D sensors that are more expensive and less common. Although multi-view representation targets easier 2D processing, one can also extract 3D information and process it in 3D. This is carried out through stereo vision in which the relative positions of objects in multiple views allow the extraction of 3D information using the triangulation of the camera rays.

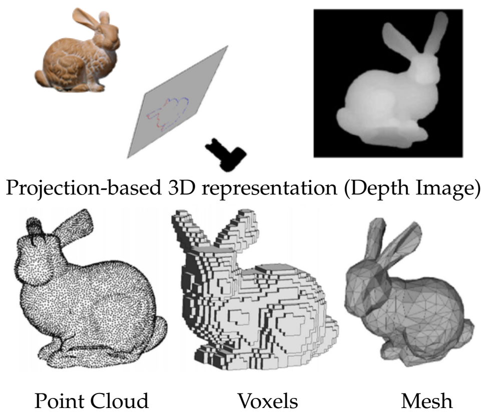
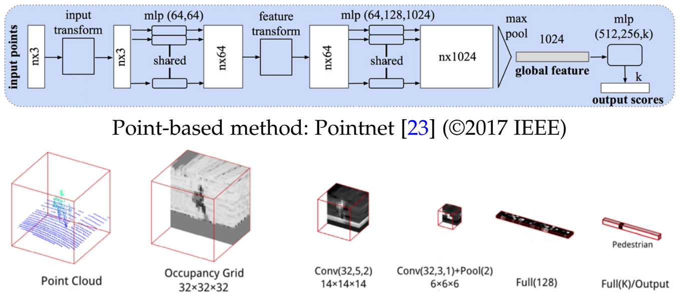


Fig. 2. Various 3D representations of the Stanford bunny [22]. The projection-based 3D representation can only visualize an object from one viewpoint. Other representations provide 3D information for all viewpoints.

Depth Images. A depth image provides the distance between the camera and the scene for each pixel (Fig. 2). Such data is commonly presented as RGB-D data, which is a structured representation that is constituted of a color image and a corresponding depth image. This data can be easily acquired using depth sensors such as the Kinect, among many others. Depth images can also be obtained from multi-view/stereo images, in which a disparity map is calculated for every pixel within an image. Since a depth image is captured from one viewpoint, it does not describe the whole object geometry - objects are seen from one side only. Nevertheless, since many 2D algorithms can be directly employed on such structured data, using this representation benefits from the great advances in 2D processing.

Point Cloud. A point cloud is a set of vertices in 3D space, represented by their coordinates along the x, y, and z axes. Such data can be acquired from 3D scanners, e.g., LiDARs or RGB-D sensors, from one or more viewpoints. Color information captured by RGB cameras can be optionally superimposed on the point cloud as additional information. Unlike images that are usually represented as matrices, a point cloud is an unordered set. As such, processing such data entails a permutation invariant method so that the output does not vary with different ordering of the same point cloud.

Voxels. A voxel representation provides information on a regular grid in 3D space. The voxel (volume element) is analogous to pixels (pictures/pix elements) on which information of 2D images is placed. The information provided at every voxel includes occupancy, color, or other features. A voxel representation can be obtained from a point cloud through the process of voxelization, which groups all features of 3D points within a voxel for later processing. The structured nature of 3D voxels allows processing such information similar to 2D methods, e.g., convolutions. In



Voxel-based method: Voxnet [24] (©2015 IEEE)

Fig. 3. Comparison between voxel-based and point-based methods. A point-based method directly processes the point cloud, whereas a voxel-based scheme handles the data on a regular grid via voxelization. Figures from [23] and [24].

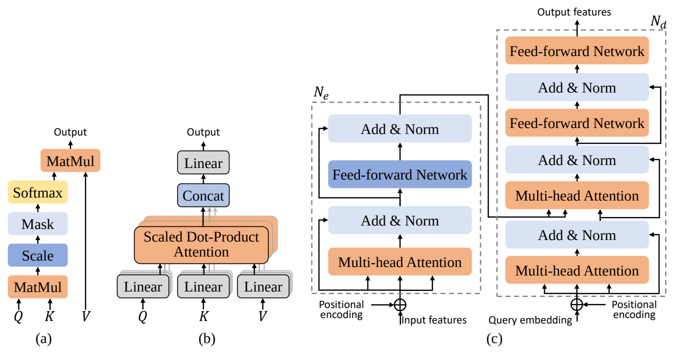


Fig. 4. Attention and transformer structures. (a) scale dot-product attention. (b) multi-head attention. (c) transformer. In both (a) and (b), there are three inputs: a query vector Q, a key vector K, and a value vector V, and a weighted output. In (c), the transformer consists of an encoder (left) and a decoder (right).

a 3D convolution, the kernel slides in three dimensions as opposed to two dimensions in 2D convolutions. On the other hand, a voxel representation is usually sparse as it contains a lot of empty volumes corresponding to space around objects. Additionally, since most capturing sensors collect information of object surfaces, object internals are also represented by empty volumes.

Meshes. A mesh is a collection of vertices, edges, and faces (polygons). The elementary component is the polygon, which is a planar shape defined by connecting a group of 3D vertices. Compared to the point cloud which only provides vertices locations, a mesh includes information of the object surface. The mesh is commonly used in computer graphics applications to represent 3D models. Nonetheless, processing the surface information directly using deep learning methods is not straightforward, and many techniques resort to sampling points from the surfaces in order to transform the mesh representation into a point cloud.

2.2 3D Processing

For 2D image understanding, the common representation on the regular grid is utilized. On the other hand, 3D data can be represented differently, and various methods have been proposed. Existing 3D deep learning approaches can be categorized as: (1) point-based, (2) voxel-based, and (3) projection-based.

Point-Based Deep Networks. Point-based methods directly process such data without transforming it into a regular set. Therefore, such methods extract feature information using permutation invariant techniques. PointNet [23] uses pointwise MLPs with the global max-pooling operator to extract features while being permutation invariant (Fig. 3). Nonetheless, PointNet does not capture local structures in the physical space around 3D points. As such, PointNet++ [25] is proposed to combine local features at multiple scales.

A few recent approaches use graph neural networks (GNNs) to process point clouds. The nodes on the graph correspond to 3D points and information is passed through edges connecting the nodes. Dynamic Graph CNN [26] exploits local geometric structures by constructing a local neighborhood graph using K-nearest neighbor (kNN).

Another approach to directly process 3D point clouds is to use continuous convolution operations. For example, SpiderCNN [27] uses a family of polynomial functions as kernels for convolution. The kernel weights at neighboring points are thus dependent on the distance to those points. KPConv [28] introduces a point convolution in which the kernel is represented as a set of points in the Euclidean space with kernel weights. On the other hand, PointConv [29] uses nonlinear functions of local 3D coordinates as convolution kernels with weight and density functions. The weight functions are learned with MLPs, whereas the density functions are learned by kernel density estimation.

Voxel-Based Deep Networks. Instead of processing unordered and irregular point cloud sets, numerous methods transform the 3D data into a regular grid by voxelization. In [24] 3D convolutions are applied to the dense voxel grid for object recognition. Nevertheless, when compared to 2D images, the added dimension leads to a significant increase in the size of the data to process, leading to constraints on object size or voxel resolution. Furthermore, this approach is not computationally efficient as it does not benefit from the sparse nature of the 3D data. Alternatively, other methods [30], [31] operate convolutions at occupied voxels only, which greatly reduces the computational requirements. This allows processing at a higher resolution, which is reflected in a higher accuracy compared to the dense approach. Other methods [32], [33] propose to learn the 3D representation at higher resolutions by partitioning the space into an octree hierarchy. In the octree structure, densely occupied regions are modeled with high accuracy whereas empty regions are represented by large cells.Fig. 3 shows a comparison between a voxel-based and a point-based method.

Projection-Based Deep Networks. Another way to transform the irregular point cloud sets into a regular one is through projection. Once 3D data is projected onto a plane, numerous 2D methods can be used for analysis. Existing approaches include projecting point cloud sets into multiple views [34], [35], [36], [37], onto a 2D plane for processing [38], or onto an estimated tangent plane and applying convolutions with continuous kernels [39].

2.3 Transformer

Self-Attention. Transformers [40] have been widely used in numerous language and vision tasks. In transformers, scaled dot-product attention is the key, which aims to capture the dependencies between different input elements. Fig. 4(a) shows a typical scaled dot-product attention module. The attention module takes a query vector Q, a key vector K, and a value vector V as inputs and generates a weighted sum of the values:

$$Z = f_{att}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{1}$$

where d_k represents feature dimension in the query and key that is used to scale the output of dot product operation.

The scaled dot-product attention generates a single attention map to represent the relationship between the query and the key. To better represent the relationship, multi-head attention (Fig. 4(b)) aims to attend to information from different sub-spaces. Multi-head attention consists of h scaled dot-product attention modules and generates h different outputs:

$$Z_i = f_{att}(QW_i^q, KW_i^k, VW_i^v), i = 1, ..., h, \tag{2}$$

where f_{att} represents scaled dot-product attention in Eq. 1, W_i^q, W_i^k, W_i^v are learnable weight matrices for the query, key, and value. Finally, multi-head attention concatenates the h outputs and feds it to a linear layer to generate the final output.

Transformer Architecture Fig. 4(c) shows a typical transformer model based on an encoder-decoder structure. The encoder has N_e identical blocks, where each block consists of a multi-head self-attention sub-layer and a feed-forward network. The multi-head self-attention sub-layer captures the relationship between different input elements, while the feed-forward network converts the features for each input element with multi-layer perceptrons. In addition, there is a residual connection and a normalization operation after each sub-layer. The decoder has N_d identical blocks, where each block consists of a multi-head self-attention sub-layer, a multi-head cross-attention sub-layer, and a feed-forward network. The multi-head self-attention sub-layer captures the relationship between different decoder elements, while the multi-head cross-attention sub-layer performs attention on the outputs of the encoder by taking the outputs of the encoder as the key and value. The feed-forward network converts the features for each input element with multi-layer perceptrons. Similar to the encoder, there is a residual connection and a normalization operation after each sub-layer in the decoder.

Transformer Properties. The transformer properties are summarized as follows:

- Transformers generate the outputs according to the relationship between different elements. Namely, transformers can dynamically aggregate the inputs instead of learning static weights.
- Transformers are permutation invariant. A common representation of 3D information is the point cloud data, which is an unordered set of points. Therefore, a permutation invariant technique is required to ensure a consistent output for the same input (object).
- Transformers are capable of processing arbitrary-sized inputs. This is suitable for the 3D domain since the input data occurs in varying sizes.
- Transformers model long-range relationships. They are not bound to narrow receptive fields and are suitable for 3D vision tasks with scattered input.

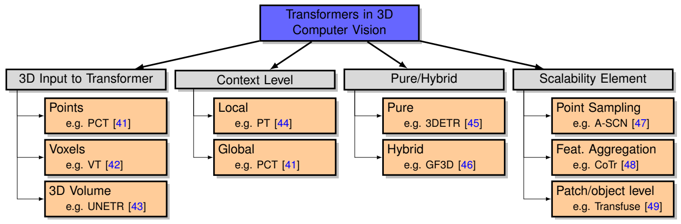


Fig. 5. Taxonomy of the transformer design in 3D Computer Vision. We group the methods into underlying approach differences pertaining to the input to the transformer, context level, its combination with other learning methods (pure/hybrid), and scalability element. We show the common choices in each group as well as example paper references.

These properties show that using transformers in 3D computer vision tasks has good prospects. As a result, transformer-based 3D vision tasks have attracted much attention in recent years.

3 Transformer Design in 3D Vision

The attention block captures long-range dependencies which facilitate learning context not fully exploited in convolution-based networks. These long-range dependencies can play an important role in scene understanding especially when the local information is ambiguous. Moreover, transformers can be applied to the sets, which is the natural representation of a point cloud. Unlike image representations, point clouds can occur in different lengths, sharing similarities to words in sentences. Given the success of transformers in NLP, one would hope that the transformer integration into the 3D domain would follow a similar trend. Additionally, transformers applied in 2D require adding position information to the feature information. In 3D, the position is available as coordinates of the points in the point cloud. The aforementioned properties of transformers have formed a ground for using the transformer architecture in the 3D domain. Nevertheless, there are numerous ways to integrate a transformer into a 3D application pipeline. Therefore, we discuss key characteristics of such integration in this section. We base our discussion on the taxonomy shown in Fig. 5.

3D Input to Transformer. Different 3D data representations can be processed using a transformer architecture. The choice of the data representation affects data size, data distribution, level of detail (granularity), and structure. Moreover, the data representation would allow transformer-based methods to be coupled with existing schemes for that specific representation.

The ability of a transformer to process unordered sets allows its application directly on point clouds. The input to the transformer in this case would be the point coordinates, as well as any additional features used with non-transformer-based architectures, such as color, normal, and height from the ground, among many others. Since points belong to a continuous domain, an efficient sampling technique is required prior to processing.

Large point clouds can be cropped to a certain physical size to help process fewer points while keeping a fine resolution to capture local geometric features. A fixed size in the Euclidean space does not lead to a fixed number of points. Therefore, models that process fixed-sized inputs require sampling from the point cloud. Sampling can be achieved through random sampling, farthest point sampling, kNN sampling, or sampling from meshes, to name a few. Such sampling affects the number of points representing a given object since it depends on the scene complexity. While most methods do sampling in the pre-processing stages, some methods sample during the training process.This can lead to a significant overhead in the training process.

An alternative to processing point clouds directly with a transformer is to convert the input to a regular grid. The equally spaced voxels allow similar representation of objects irrespective of the number of points in the point cloud. It also facilitates neighbor search, if required, since it can be searched with a hash table. On the other hand, a fine grid resolution is required to capture fine shape information, which leads to a cubic increase in the data to process. In addition, since 3D scenes are dominated by empty space, it is inefficient to process empty voxel grids. Processing occupied voxels would result in different sized inputs, and thus can be sampled similar to point clouds. Nonetheless, point density consistency and the easier search would still be present for the voxel representation.

Context Level. An efficient vision application should be able to capture fine local information as well as global context. In both cases, an increase in the computational requirement is encountered. Therefore, data is commonly processed on different scales to achieve both targets.

Transformers that process 3D information can be applied to a local neighborhood of points to capture local shape information. Similar to methods in the 2D domain, local pooling would allow processing on a different scale with a larger receptive field. The larger receptive field provides interaction between farther points for learning context.

Since applying a transformer on local information requires a multi-layer application, a transformer that consumes the whole 3D data is also feasible. This reduces the necessity for local neighborhood sampling since the whole point cloud is used at once. Nonetheless, the size of the input data is limited, and a good balance between point cloud coverage and density of points is needed.

Pure and Hybrid Transformer. Pure transformer architectures rely on attention layers to extract features and generate task specific output. In some cases, non-attention layers are utilized to encode the input or to supplement the attention layers. We consider an architecture as a pure transformer if it does not rely on previous non-attention architectures or backbones in the proposed pipeline.

Since transformers are capable of capturing global context, they can be used to extract richer features. One way to integrate transformers into deep learning architectures is the replace the feature extraction module with one that is attention/transformer based. Instead of fully relying on transformers to extract features, one can alternatively process local feature extraction using non-transformer-based methods, and then couple it with a transformer for global feature interaction. Transformers can also be complemented by non-transformer layers to extract richer information. The complementarity can be caused by different resolutions to which each method can be applied.

Scalability. 3D data incur more information when compared to 2D due to the additional third dimension. On the other hand, transformers are computationally expensive since they need to generate a large attention map, which has quadratic complexity with respect to the input size. Given the increased data size and transformer input size limit, it requires a sampling scheme to enable processing. Typical approaches decrease the input size to the transformer, and thus allowing scalability including: farthest point sampling for point clouds with kNN feature aggregation, voxel feature aggregation for low-resolution volumetric representation, patch feature embedding, feature downsampling using CNN-based methods, and object level self-attention.

4 Transformers in 3D Vision: Applications

The transformer architecture has been integrated into various 3D vision applications. In this section, we review methods based on the targeted 3D vision tasks, including object classification, object detection, segmentation, point cloud completion, pose estimation, and others.

4.1 Object Classification

We first give an overview of methods that employ the transformer within a defined local region is presented, and then discuss methods that apply the transformer on the global level. Table 1 shows an overview of these methods using based on the above-defined taxonomy.

Local Transformers. Point Transformer [44] applies self-attention in the local neighborhood of each data point. A point transformer block consists of the attention layer, linear projections, and residual connection (Fig. 6). Additionally, instead of using the 3D point coordinates as position encoding, an encoding function is used with linear layers and ReLU nonlinearity. To increase the receptive field of the proposed transformer architecture, transition down layers are introduced, as well as transition up to retrieve the original data size.

3DCTN [57] proposes to combine graph convolution layers with transformers. The former learns local features efficiently, whereas the latter is capable of learning global context. Taking the point cloud with normals as input, the network consists of two modules that downsample the point set, with each module having two blocks: the first block is a local feature aggregation module using a graph convolution, and the second block is a global feature learning module using a transformer consisting of offset-attention and vector attention. LFT-Net [60] proposes a local feature transformer network that uses self-attention to learn features of point clouds. It also introduces a Trans-pooling layer that aggregates local features to reduce the feature size.

Global Transformers. At the global scale, the attention module has been integrated into various parts of the network with different inputs and position embeddings. Attentional ShapeContextNet [47] is one of the early adoptions of self-attention for point cloud recognition. To learn shape context, the self-attention module is used to select contextual regions, aggregate and transform features. This is carried out by replacing hand-designed bin partitioning and pooling with a weighted sum aggregation function with input learned by self-attention applied on all the data. In [56], Adaptive Wavelet Transformer first performs multi-resolution analysis within the neural networks to generate visual representation decomposition using the lifting

TABLE 1
Overview of transformer-based methods for classification. Important attributes for the transformer integration are shown here, which include the input, sampling element that enables transformer processing, architecture (pure or hybrid), and context level on which the transformer operates. A highlight of the main contributions is also included. All the methods also perform object part segmentation except the ones with an asterisk (*)

Method	Input	Scalability Element	Architecture	Context	Highlight
Point Transformer [44]	points	farthest point sampling	pure	local	applies self-attention in a local neighborhood, Transition down and up to increase receptive field
Point Transformer [50]	points	local (ball query)/ global (FPS)	with own sortnet	local / global	extract ordered local feature sets from different subspaces (SortNet), global and local-global attention
Attentional ShapeContextNet [47]	points	random sampling	pure	global	replace hand-designed bin partitioning and pooling by a weighted sum aggregation function with input learned by self-attention
Yang *et al.* [51]	points	random sampling	pure	global	absolute and relative position embedding as input to attention module, group attention similar to depthwise separable convolutions [52] and channel shuffle [53].
PCT [41]	points	farthest point sampling	pure	global	offset-attention calculates the element-wise difference between the self-attention and the input features
PVT [54]	voxels, points	local (hash table), global (all)	pure	local / global	combines voxel-based and point-based transformer models to extract feature information
TransPCNet* [55]	points	kNN aggregation	hybrid	global	feature embedding module and attention module to learn features to detect defects in sewer represented by 3D point clouds
Adaptive Wavelet Transformer [56]	points from graph	k nearest neighbor	hybrid	global	perform multi-scale analysis to generate visual representation decomposition using the lifting scheme approach
3DCTN* [57]	points	query ball	hybrid	local	combines graph convolution layers(local feature aggregation) with transformers (global feature learning)
DTNet [58]	points	farthest point sampling	pure	global	Dual Point Cloud Transformer module to capture long-range position and channel correlations
CpT [59]	points	k nearest neighbor	pure	local / global	uses a dynamic point cloud graph to create a point embedding that is fed into the transformer layer
LFT-Net [60]	points	k nearest neighbor	pure	local	local feature transformer with local position encoding, self-attention pooling function for feature aggregation
Point-BERT [61]	points	FPS and point patches	hybrid	global	point tokenization which converts a point cloud into discrete point tokens, and masked point modeling for pre-training
Liu *et al.* [62]	points	FPS, kNN	pure	local / global	radius-based feature abstraction for better feature extraction, group-in-group relation-based transformer architecture
Pang *et al.* [63]	points	FPS, kNN (or FPS and point patches)	pure	global	Transformer-based autoencoder with asymmetric design and shifting mask tokens operation for pre-training
PAT [64]	voxels	voxelization	hybrid	local / global	patch attention module (PAT) and a multi-scale attention module (MST)
3CROSSNet [65]	points	farthest point sampling	hybrid	local / global	Point-wise Feature Pyramid, Cross-Level Cross-Attention, and CrossScale Cross-Attention
3DMedPT [66]	points	farthest point sampling	hybrid	global	local context augmentation, relative positional embedding, and local context aggregation at query
Wu *et al.** [67]	points &others	soft k-means clustering	pure	global	centroid attention: summarizing self-attention feature mapping to a smaller number of outputs
MLMSPT [68]	points	farthest point sampling	pure	local / global	point pyramid transformer followed by multi-level and multi-scale transformer

scheme technique. The generated approximation and detail components capture geometric information that is of interest for downstream tasks. A transformer is then used to pay different attention to features from approximation and detail components and to fuse them with the original input shape features. TransPCNet [55] aggregates features using a feature embedding module, feeds them into separable convolution layers with a kernel size of 1, and then uses an attention module to learn features to detect defects in sewers represented by 3D point clouds.

Other methods propose variations of the attention module. Yang *et al.* [51] develop Point Attention Transformers (PATs) by applying an attention module to a point cloud represented by both absolute and relative position embeddings. The attention module uses a multi-head attention design with group attention, which is similar to depthwise separable convolution [52], in addition to channel shuffle [53]. Point Cloud Transformer (PCT) [41] applies offset-attention to the input point embedding. The offset-attention layer calculates the element-wise difference between the self-attention features and the input features (Fig. 6). It also uses a neighbor embedding by sampling and grouping neighboring points for better local feature representation. DTNet [58] aggregates point-wise and channel-wise multi-head self-attention models to learn contextual dependencies from the position and channel.

Some methods focus on pre-training the transformer by masking parts of the input. Point-BERT [61] first partitions the input point cloud into point patches, inspired by Vision Transformers [69], and uses a mini-Pointnet [23] to generate a sequence of point embeddings. The point embeddings are then used as an input to a transformer encoder, which is pre-trained by masking some point embeddings with a mask token, similar to [70]. The tokens are obtained using a pre-learned point Tokenizer that converts the point embeddings into discrete point tokens. Similarly, Pang *et al.* [63] divide the input point cloud into patches and randomly mask them during pre-training. A transformer-based autoencoder is used to retrieve the masked point patches by learning high-level latent information from unmasked point patches.

Local and Global Transformers. Numerous methods have been proposed to use transformer architecture for learning both local and global information. For that, the transformer has been deployed at various stages to process different information. Engel *et al.* [50] employ local-global attention to capture local and global geometric relations and shape information. The input features to the attention module are local features of an ordered subset, which is learned via a permutation invariant network module. In [59], CpT uses a dynamic point cloud graph to create a point embedding that is fed into the transformer layer. The transformer layer is comprised of sample-wise attention that dynamically processes local point set neighbors as well as inter-point attention. On the other hand, Liu *et al.* [62] group points using farthest distance sampling with K nearest neighbors. It then uses group abstraction and radius-based feature abstraction to obtain group features. A transformer is then used with groups as well as across all point groups. 3DMedPT [66] embeds local point cloud context by downsampling the points and then grouping local features similar to DGCNN [26]. It proposes the use of relative positional embeddings and local response aggregation at query.

Point-Voxel Transformer (PVT) [54] combines voxel-based and point-based Transformer models to extract fea-

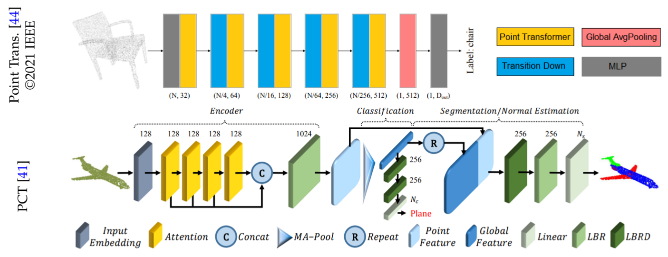


Fig. 6. Example of local (Point Transformer) and global (PCT) methods. The local method uses multiple transformer layers at different scales to achieve a larger receptive field. Global methods apply multiple attention layers for richer feature representation. Figure from [44] and [41].

ture information. The voxel-based model captures local features efficiently because of regular data locality. It uses a local attention module, whose computational complexity is linear with respect to the input voxel size. The point-based model captures global features and also remedies the information loss during voxelization. It uses relative attention which is a self-attention variant that considers pairwise relationships or distance between input points.

On the other hand, some methods focus on applying attention on multiple scales. Patchformer [64] constructs a voxel-based architecture that integrates a patch attention module (PAT) and a multi-scale attention module (MST). The PAT module applies weighted summation over a small set of bases to capture the global shape, and thus achieves linear complexity to the input size whereas the MST module applies attention to features of different scales. In [65], 3CROSSNet first extracts multi-scale features using a point-wise feature pyramid module. Cross-attention is then applied across levels to learn inter-level and intra-level dependencies. Another cross-attention module is applied across scales to better represent between scales and within scales interactions. MLMSPT [68] proposes a point pyramid transformer that captures features from multiple levels and scales. A multi-level transformer and a multi-scale transformer are then used to capture contextual information from different levels and scales.

Wu *et al.* [67] introduce centroid attention, in which self-attention maps information in the inputs into a smaller output. During training, a soft K-means clustering objective function is optimized. The centroid attention then transforms the input sequence into the set of centroids.

4.2 3D Object Detection

Numerous attention-based methods have been proposed for 3D object detection. Table 2 shows an overview of these methods and the adopted transformers. Most of these methods are applied to one domain: indoors or outdoors. Limiting applications to one domain is due to the varied modality of data collected indoors compared to outdoors, where RGB-D sensors are the common indoor 3D sensors and LiDAR is common outdoors. This leads to different dataset distributions, densities, and ranges. Nevertheless, methods from one domain can be applied to another domain having the same representation, but often require substantial adaptation to achieve competitive results.

One of the methods that have been applied indoors and outdoors is Pointformer [71]. It uses a transformer-based feature learning block with three parts: a local transformer to capture fine information in the local region, a local-global transformer to integrate the learned local features with the global information, and a global transformer to capture the global context. The local transformer aggregates features from the local regions into a subsampled set of points, thus reducing the computational requirements. Other methods have been only applied to either indoor or outdoor datasets.

Indoor Object Detection MLCVNet [83] builds on top of [84] and uses self-attention modules to aggregate contextual information at multiple levels, namely patch, object, and global scene levels. At the patch level, the attention module is used to generate better voting to object centroid points. The object level attention module captures contextual information among proposals, and the global level attention module uses patch and cluster information to learn global scene context. 3DETR [45] proposes an end-to-end transformer consisting of two modules: a transformer encoder applied directly on the point cloud for extracting feature information, and a transformer decoder to predict 3D bounding boxes (see Fig. 7). The decoder transformer layers use non-parametric query embeddings from seed points for better 3D detection.

Liu *et al.* [46] use a transformer module to extract and refine object representations from object candidates with features learned using PointNet++ [25]. The transformer module consists of multiple multi-head self-attention and multi-head cross-attention and operates on a subset of the points, sampled using the k-closest point sampling technique. ARM3D [80] uses an attention-based module to extract fine-grained relations among proposal features learned using non-transformer-based architectures. The objectness score is used to choose the proposals, and each proposal is matched with other proposals to learn relation contexts.

In [92], BrT uses a transformer to allow interactive learning between images and point clouds. It adopts conditional object queries of aligned points and image patches and adds point-to-patch projection for better learning.

Outdoor Object Detection. For outdoor environments, the transformer architecture has been utilized to process data from different sources and in different representations. Numerous transformer models with voxel representations have been proposed. Among these methods is the Voxel Transformer [42], which processes an input voxel grid through a series of sparse and submanifold voxel modules. It performs multi-head self-attention on non-empty voxels through local attention and dilated attention. A voxel query mechanism is used to accelerate searching for non-empty voxels, benefitting from having the data on a regular grid. Fan *et al.* [77] show that downsampling the feature maps in 3D similar to the 2D domain would lead to a loss of information, and propose a single stride transformer to maintain the same resolution throughout the whole network. It also uses a voxelized input, but the transformer operates with sparse regional attention to reduce the computational requirement for the transformer modules. Fast Point Transformer [78] is developed to speed up local self-attention networks. Since local self-attention usually requires finding the k-nearest points, it is usually a bottleneck. The proposed self-attention module learns on a point cloud with voxel hashing architecture, which allows fast neighborhood selection and is cou-

TABLE 2
Overview of 3D object detection methods using the transformer architecture. A highlight of the main design attributes and contributions is shown here. These methods use a variety of input representations, employ multiple sampling strategies for scalability, use a pure or hybrid transformer integration, and apply the transformer locally or globally.

Method	Input	Scalability Element	Architecture	Context	Highlight
Pointformer [71]	points	Linformer for scalability	pure	global/ local	a feature learning block with a local, local-global, and global transformer
Voxel Transformer [42]	voxels	voxel discretization	pure	local/ dilated	multi-head self-attention on non-empty voxels through local attention and dilated attention
Sheng *et al.* [72]	points	proposal to point attention	hybrid	global	uses raw points and proposals as input into a channel-wise transformer with a proposal-to-point encoding module and a channel-wise decoding module
Liu *et al.* [46]	points	k-closest points sampling	hybrid	global	stacked multi-head self-attention and multi-head cross-attention to extract and refine object representations for object candidates
DETR3D [73]	features	object queries	hybrid	-	multi-head attention to refine object queries by incorporating object interactions, similar to DETR [74].
3DETR [45]	points	pointnet++ aggregation	almost pure	global	A transformer encoder is applied directly on the point cloud for extracting feature information, and a transformer decoder to predict 3D bounding boxes
SA-Det3D [75]	points, voxels, pillars	attend to salient regions	hybrid	global	augment multiple convolution-based methods with full self-attention or deformable self-attention
M3DETR [76]	points, voxels	applied on output	hybrid	global	combines raw points, voxels, and bird-eye view representations under a unified transformer-based architecture
Fan *et al.* [77]	voxels	regional grouping	pure	local	transformer operates with sparse regional attention on voxelized input
Fast Point Transformer [78]	voxels	(centroid aware voxelization)	pure	local	speed-up local self-attention networks with voxel hashing architecture and centroid-aware voxelization and devoxelization
Voxel Set Transformer [79]	voxels	voxelization	hybrid	local	A voxel-based set attention module with two cross-attentions. It applies self-attention to token clusters with varying sizes and processes them with a linear complexity.
ARM3D [80]	points	FPS	hybrid	global	attention module to learn relation features for proposals
Yuan *et al.* [81]	points	only temporal channel	hybrid	temporal	temporal encoder and spatial decoder with a multi-head attention mechanism to aggregate information from the adjacent video frames
Dao *et al.* [82]	voxels	voxel discretization	hybrid	global	vector attention that learns different weights for the point feature channels
MLCVNet [83]	points	pointnet++ aggregation	hybrid	global	builds on top of [84] and uses self-attention modules to learn contextual information at the patch, object, and global scene levels
Yin *et al.* [85]	pillars	discretized pillar nodes	hybrid	local	spatial transformer attention and temporal transformer attention on point pillars
SCANet [86]	BEV	after encoder	hybrid	global	attention after VGG encoder for point cloud BEV and RGB
MonoDETR [87]	images	feature downsampling	hybrid	global	predicts and encodes depth to use it as input into a depth-aware decoder
Transfusion [88]	BEV, images	feature map from conv backbone	hybrid	global	convolution backbone to extract feature maps, transformer decoder to fuse LiDAR based queries with image features
CAT-Det [89]	points, images	ball query	hybrid	local/ global	combines a Pointformer applied on point cloud with Imageformer applied on RGB images
PETR [90]	images	conv-based encoder	hybrid	global	fuses 3D features from multi-view images with 2D features
BoxeR [91]	BEV, images	conv encoder for 2D, PointPillar for 3D	hybrid	local	applies attention to a sampled grid within a box in 2D and 3D
BrT [92]	points, images	pointnet++ aggregation	pure	global	bridging point tokens (from point cloud) and patch tokens from images, point-to-patch projection
VISTA [93]	BEV, RV	voxelization, projection to BEV and RV	hybrid	global	replace MLP in attention with convolutions, apply on BEV and RV
PDV [94]	voxels	voxel discretization	hybrid	global	voxel centroid localization, density-aware RoI grid pooling, grid point Self-Attention

pled with a centroid-aware voxelization and devoxelization to embed continuous 3D coordinates. Recently, Voxel Set Transformer [79] presents a global approach to model long-range dependencies in a point cloud. It introduces a voxel-based set attention (VSA) module, which consists of two cross-attentions as a replacement for the self-attention, and can process inputs of varying sizes in parallel with linear complexity. PDV [94] uses 3D sparse convolutions to extract feature information from a voxelized 3D scene, followed by a region proposal network head to generate bounding boxes. Voxel features are then pooled and used as an input to a self-attention module to refine the bounding boxes.

Other methods choose to apply transformers with a point cloud representation. In [72], Sheng *et al.* supplement a two-stage 3D detector with self-attention modules. This method first generates proposals via a 3D voxel-based region proposal network and then uses the raw points and the proposals as input into a channel-wise transformer in order to enrich the proposals with global context information. The channel-wise transformer consists of a proposal-to-point encoding module and a channel-wise decoding module that transforms the encoded features into final object proposals comprising confidence prediction and box regression. On the other hand, PLNL-3DSSD [95] uses local and non-local attention with set abstraction modules to model relationships between objects.

Moreover, numerous methods apply the transformer to multi-view images or BEV, benefitting from the advancement of transformer application to images. Transfusion [88] uses convolution backbones to extract LiDAR BEV feature map along with an image feature map. A transformer-based decoder takes object queries as input and outputs initial bounding box predictions using the LiDAR information. Next, a spatially modulated cross attention mechanism then performs fusion between camera image features and LiDAR object queries. SCANet [86] extracts features from RGB images and point cloud bird-eye-view using two VGG-16 encoders. A spatial-channel attention module is then employed to extract multi-scale and global context features to recalibrate the features. BoxeR [91] introduces box attention, which learns attention weights for points sampled on a grid within boxes. In 2D, it uses convolution encoder features for proposals as input and generates object queries. The object queries are then decoded into bounding boxes using instance attention. It learns attention weights that are invariant to rotation, so another transformer is used on the bird-eye's view to generate 3D bounding boxes. Recently, MonoDETR [87] modifies DETR to generate 3D bounding boxes from monocular images. The modification includes adding depth features to the input of the transformer. The

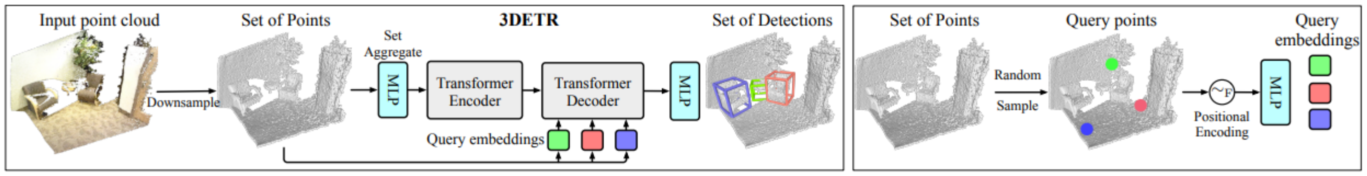


Fig. 7. 3DTER: Pure transformer-based architecture for indoor 3D object detection. It consists of a transformer encoder applied directly on the point cloud for extracting feature information, and a transformer decoder to predict 3D bounding boxes. The decoder transformer is similar to DETR [74] with adaptation to 3D detection through non-parametric query embeddings and Fourier positional embeddings. Figure from [45] (©2021 IEEE).

depth features are generated using a depth predictor and a depth encoder. The transformer consists of a depth-aware decoder with self-attention as well as visual and depth cross-attention. VISTA [93] proposes to replace the linear projections in the regular attention module with convolutional operators. It applies the proposed attention to the projection of the features of a voxelized 3D scene into two views, bird-eye-view and range view.

For point cloud videos, Yuan *et al.* [81] present a Temporal-Channel Encoder and a Spatial Decoder for 3D LiDAR-based video object detection. The Temporal-Channel encoder is used to learn relations between different frames utilizing multi-head attention mechanism. The spatial decoder also utilizes a multi-head attention mechanism to aggregate relevant information of the adjacent video frames.

Since different representations might provide complementary information, some works have used the transformer with multiple representations. SA-Det3D [75] proposes to augment multiple convolution-based methods operating on points, voxels, and pillars with self-attention modules. It introduces two variants for self-attention: a full self-attention module, which is a pairwise self-attention mechanism, and a deformable self-attention module that learns deformations over randomly sampled locations to cover the most representative and informative parts. On the other hand, M3DETR [76] aggregates information from raw points, voxels, and bird-eye view, under a unified transformer-based architecture. The transformers enable interactions among multi-representation, multi-scale, multi-location feature attention. Dao *et al.* [82] propose to use vector attention to refine voxel-based Region Proposal Networks. Compared to the multi-head attention, vector attention learns different weights for the different point feature channels, therefore it enables to capture richer information into the Regions of Interest and pooled points. CAT-Det [89] combines a Pointformer applied on a point cloud with Imageformer applied on RGB images. The two modalities are then complemented with cross-modal feature interaction and multi-modal feature aggregation using a transformer.

On the other hand, Yin *et al.* [85] use spatial features to encode information on a point cloud discretized by pillars. The spatial features are extracted from a given point cloud using graph-based operations and 2D CNN. Features from consecutive frames are then passed to a spatio-temporal transformer module that is constituted of spatial transformer attention as well as Temporal transformer attention.

Other methods use the transformer to fuse or refine information generated by non-transformer methods. DETR3D [73] uses multi-view RGB images to detect objects in 3D. It uses non-transformer-based 2D feature extraction as well as 3D box prediction. It only uses multi-head attention to refine object queries by incorporating object interactions, similar to DETR [74]. PETR [90] first extracts features from multi-view images using a 2D backbone network (ResNet). It then uses the camera frustum space to generate a 3D mesh grid and coordinates in 3D space. 2D image features and 3D coordinates are then fused using an MLP-based encoder to generate 3D position-aware features. A transformer decoder then updates object queries based on their interaction with the 3D position-aware features.

4.3 3D Segmentation

3D segmentation aims at segmenting the 3D data constituting elements based on given semantic categories. It needs to overcome various challenges, such as class imbalance, size variation, and shape variation. In this section, we categorize methods based on the input data domain. We first review methods that take as input a 3D representation of a single object and then segment its parts. Next, we survey approaches that segment complete scenes with multiple objects. We then present methods that provide point cloud segmentation from videos. In addition, we review methods that segment 3D medical images. An overview of the transformer design for these methods is shown in Table 3.

Object Part Segmentation. To perform 3D semantic segmentation on a given point, a label for every point is required. Numerous methods that learn point-wise features for point cloud classification can be employed for semantic segmentation. This procedure is common in part-segmentation methods with the relatively smaller point clouds. Numerous transformer models have been developed for point cloud classification and semantic segmentation including: Attentional shapecontextnet [47], Yang *et al.* [51], Point2Sequence [113], Point Transformer [44], Point Cloud Transformer (PCT) [41], Point-Voxel Transformer (PVT) [54], Adaptive wavelet Transformer [56], and Dual Transformer [58].

Complete Scenes Segmentation. Fast Point Transformer [78] proposes to speed up local self-attention networks as each such module usually requires finding the k-nearest points, which is computationally expensive. The proposed self-attention module learns on a point cloud with voxel hashing architecture. The voxel hashing allows fast neighborhood selection and is coupled with a centroid-aware voxelization and devoxelization to preserve the embedding of continuous coordinates. On the other hand, Stratified Transformer [96] first aggregates local structure information using a point embedding module [28]. It then uses a transformer-based hierarchical structure with several downsampling layers to obtain multi-level features, which are then upsampled layer by layer similar to U-Net [114]. The proposed transformer uses a stratified strategy for key sampling to increase the receptive field and aggregate long-range contexts (see Fig. 8).

TABLE 3
Overview of 3D segmentation methods using the transformer architecture. We divide 3D segmentation methods into three categories: (1) methods that perform 3D semantic segmentation on complete scenes (rather than single object part segmentation), (2) panoptic segmentation, and (3) medical imaging segmentation. 3D medical images are represented by a dense regular grid, in contrast to data collected by LiDARs and RGB-D sensors which is sparse.

Method	Input	Scalability Element	Architecture	Context	Highlight
Complete scenes segmentation:					
Fast Point Transformer [78]	voxels	centroid aware voxelization	pure	local	speed-up local self-attention networks with voxel hashing architecture and centroid-aware voxelization and devoxelization
Stratified Transformer [96]	points	local aggregation with KPConv	hybrid	local	transformer-based hierarchical structure with a stratified strategy for keys sampling
Segment-Fusion [97]	points	group points to 3D segments	hybrid	global	graph segmentation to segment features, segment fusion using attention encoder block
P4Transformer [98]	point cloud video	temporal radius & stride, spatial radius & subsampling	hybrid	global	extracts features of sampled local spatio-temporal using 4D convolution, then use feature vector as input to transformer
Wei *et al.* [99]	point cloud video	FPS then set abstraction	hybrid	global	set abstraction & convolution layers to obtain patch features which are input to a transformer
Panoptic:					
Xu *et al.* [100]	points, voxels	voxelization	hybrid	global	sparse cross-scale attention network that aggregates sparse features at multiple scales and global voxel-encoded attention
Medical imaging segmentation					
UNETR [43]	3D MRI images	patches	hybrid	global	patch embedding as input to the transformer, skip connection between encoder and decoder
D-Former [101]	3D medical images	patches	hybrid	local / global	Dilated Transformer that applies self-attention alternately in local and global scopes
CoTr [48]	3D medical images	from CNN encoder	hybrid	global	CNN encoder, multi-scale deformable self-attention, CNN decoder
T-AutoML [102]	3D CT scans	encodings of architectures, augmentation, hyperparameters	hybrid	-	Transformer to adapt to dynamic length to search for the best network architecture
Transfuse [49]	3D medical images	patches	hybrid	global	fusion between CNN features and transformer features at multiple scales
Karimi *et al.* [103]	3D medical images	patches	pure	global	3D patches as input to pure transformer
SpecTr [104]	3D medical images	from CNN encoder	hybrid	global	depth-wise convolution, spectral normalization, and transformers as encoder
TransBTS [105]	3D MRI images	from CNN encoder	hybrid	global	transformer at 3D UNet bottleneck
Segtran [106]	3D medical images	FPN, from CNN encoder	hybrid	global	CNN with FPN as input to Squeeze-and-Expansion Transformer
nnFormer [107]	3D medical images	from CNN embedding	hybrid	local / global	local and global self-attention on CNN embedding with skip self-attention
BiTr-UNet [108]	3D medical images	from CNN encoder	hybrid	global	Transformer at 3D UNet bottleneck, CBAM for 3D CNN
AFTer-UNet [109]	3D volume	from CNN encoder	hybrid	global	axial fusion transformer to fuse inter-slice and intra-slice information
Peiris *et al.* [110]	3D volume	patches	pure	local / global	encoder with local/global attention, decoder with parallel window-based self/cross attention
Swin UNETR [111]	3D medical images	patches	hybrid	global	1D sequence embedding as input to a Swin Transformer [112]

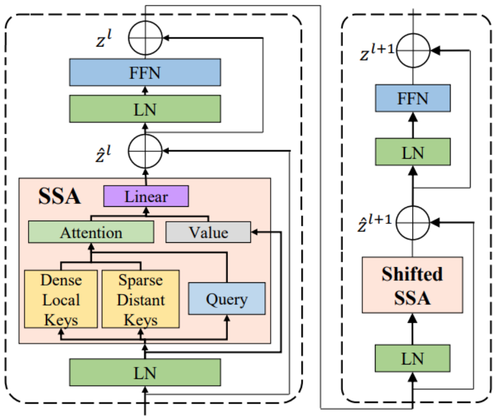


Fig. 8. Stratified Transformer for 3D point cloud segmentation: A transformer implementation applied locally with two properties to increase the receptive field: (1) A hierarchical structure is used to extract multi-level features, and (2) a stratified self-attention (SSA) block samples distant points as keys in a sparse way. Figure from [96].

Segment-Fusion [97] employs graph segmentation methods to group points and their respective features into segments with segment-wise features. These features are then fused using a stack of attention encoder blocks, in which the attention matrix is also multiplied with the adjacency matrix to account for the connections between segments. The output of the attention-based is then grouped into object instances using the connected component algorithm.

Transformers have also been applied to panoptic segmentation on point clouds. Xu *et al.* [100] first generate point-wise features and sparse voxel features for a given point cloud. Voxel features are then aggregated using a cross-scale attention module, which allows capturing long-range relationship of object context and increases regression accuracy for the over-segmented large objects.

Point Cloud Video Segmentation. P4Transformer [98] applies a transformer to point cloud videos for 3D action recognition and 4D semantic segmentation. It first samples and constructs local spatio-temporal areas, and uses a 4D convolution to encode them into a feature vector that can be processed by a transformer. In [99], Wei *et al.* first extract features using set abstraction layers from [25] and use a resolution embedding module to retain geometric information in the extracted features. It then applies a convolution on the features of neighboring frames to group them into patches. The patches are used as input to a spatio-temporal transformer to capture context information for the tasks of 3D action recognition and 4D semantic segmentation.

3D Medical Images Segmentation. UNETR [43] divides the input 3D volume into a sequence of uniform non-overlapping patches and projects them into an embedding space using a linear layer. A transformer is then applied to learn sequence representations of the input volume (encoder) and capture the global multi-scale information. In [48], CoTr uses a CNN-encoder to extract multi-scale feature maps from an input 3D medical image. The feature maps are embedded with positional encoding and processed using a deformable self-attention transformer. The features are then upsampled to the original resolution using a CNN decoder.

In [102], T-AutoML introduces an automated search algorithm for finding the best network architecture, hyperparameters, and augmentation methods for lesion segmentation in 3D CT images. It uses the transformer model for its ability to operate on varying embedding lengths. On the other hand, D-Former [101] proposes local and global attention-based modules to increase the scopes of information interactions without increasing the number of patches. A Dilated Transformer applies self-attention for pair-wise patch relations captured in the local and global scopes. It also applies dynamic position encoding to embed relative and absolute position information.

Recently, Transfuse [49] presents a fusion module to fuse information from two branches: a CNN branch which encodes features from local to global and the other one is a transformer branch which starts with global self-attention and then recovers local information. In [103] Karimi *et al.* first divide the input 3D image block into 3D patches. A one-dimensional embedding is then computed for each patch and passed through an attention-based encoder to predict segmentation of the center patch. SpecTr [104] takes as input a sequence of spectral images and then alternately processes them with depth-wise convolution, spectral normalization, and transformers with sparsity constraint in the encoder. It then uses a decoder with skip connections similar to 3D U-Net [115]. On the other hand, TransBTS [105] uses a 3D CNN to generate feature maps that capture spatial and depth information, and then utilizes a transformer encoder to model the long-distance global contextual dependency. The transformer output is then alternately upsampled, stacked, and convoluted to produce the segmentation labels. Segtran [106] uses CNN layers to extract features, which are used as input to Squeeze-and-Expansion transformer layers to learn the global context. A Feature Pyramid Network is applied before the transformer to increase the spatial resolution, and after the transformer to upsample to the original resolution.

Zhou *et al.* combine convolution and self-attention operations by applying local and global volume-based self-attention operations in nnFormer [107]. It also proposes to use skip attention which is analogous to skip connections in UNet-like architectures. BiTr-UNet [108] applies a transformer block at the bottleneck of a 3D UNet architecture i.e., after the 3D CNN encoder and before the upsampling layers. It integrates CBAM [116] into the convolution layers by expanding it for 3D CNN. AFTer-UNet [109] encodes neighboring slice groups using a CNN encoder and then applies an axial fusion transformer. The axial fusion transformer fuses inter-slice and intra-slice information which is then passed into a CNN decoder for segmentation. Peiris *et al.* [110] propose an encoder block design with local and global self-attention layers. It uses a decoder with window-based self and cross attention, where both attention mechanisms use one shared query projection. In addition, it proposes a convex combination approach in the decoder along with Fourier positional encoding. Swin UNETR [111] projects multi-modal input data into a 1D sequence of embedding and uses it as input to an encoder composed of hierarchical Swin Transformer [112]. The Swin Transformer uses shifted windows to compute self-attention at multiple resolutions and has skip connections to a FCNN decoder.

4.4 3D Point Cloud Completion

Point cloud completion aims to generate a complete point cloud from a partial point cloud of an object. Table 4 gives a brief summary of some 3D point cloud completion methods using the transformer structure. Fig. 9 shows the architecture of PoinTr [117]. PoinTr [117] treats point cloud completion as a set-to-set translation task and adopts a geometry-aware transformer to predict the missing point cloud. Compared to the vanilla transformer, the geometry-aware transformer contains two branches, where one branch adopts self-attention to extract semantic features and another branch adopts kNN model to extract geometric features. The output features of two branches are fused to generate the output feature of geometry-aware transformer. Wang *et al.* [118] integrate the proposed down-sampling and up-sampling operations into the transformer encoder-decoder structure [117] to perform point cloud completion. Instead of max pooling, the down-sampling operation adopts neighbor pooling to select features with the highest activations. In [119], Wang *et al.* introduce a geometric details perception module and a self-feature augment module to capture both local and global information and avoid local k-nearest neighbor operation.

We note that some methods only use a transformer encoder or decoder for 3D point cloud completion. Xiang *et al.* [120] propose a skip-transformer module in the decoder to capture context for sparse point deconvolution. The skip-transformer aims to use an attention mechanism to learn spatial context from the previous decoding layer during point generation stage. Recently, Su *et al.* [121] employ a transformer encoder to extract context information from coarse point cloud generation. After that, a point cloud upsampling is used to generate a fine point cloud. In [122], Lin *et al.* utilize a transformer encoder to learn semantic affinity information, which is followed by a morphing-atlas

TABLE 4
State-of-the-art 3D point cloud completion methods using transformers. These methods use a variety of input representations, employ a pure or hybrid architecture, and apply the transformer locally or globally.

Method	Input	Architecture	Context	Highlight
PoinTr [117]	points	hybrid	global/ local	a set-to-set translation and geometry-aware transformer for point cloud completion
Wang *et al.* [118]	points	hybrid	global/ local	neighboring pooling integrated with transformer encoder-decoder
PointAttN [119]	points	hybrid	global/ local	a geometric details perception module and a self-feature augment module
SnowflakeNet [120]	points	hybrid	local	a skip-transformer module to capture shape context
Su *et al.* [121]	points	hybrid	global/ local	attention for feature aggregation and upsampling
PCTMA-Net [122]	points	hybrid	global	a transformer encoder to learn semantic affinity information
AutoSDF [123]	voxel	hybrid	global	transformer-based autoregressive modeling in low-dimension latent space
ShapeFomer [124]	voxel	hybrid	global	a transformer to predict a distribution of object completions
PDR [125]	point	hybrid	local	aggregating local features by the attention operation
MFM-Net [126]	point	hybrid	global	a self-attention layer for global refinement

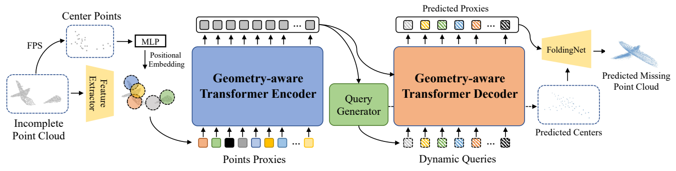


Fig. 9. PoinTr [117]: Geometry-aware transformer for point cloud completion. PoinTr first extracts the deep features of center points, second adopts the geometry-aware transformer encoder-decoder to predict the proxies of missing points, and third employs FoldingNet to complete the point cloud. The figure is from [117] (©2021 IEEE).

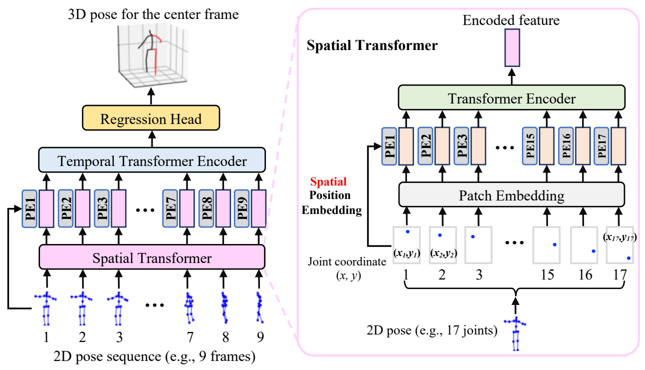


Fig. 10. PoseFormer [127]: Transformer-based approach for 3D human pose estimation in videos. PoseFormer takes the 2D pose sequence of multiple frames, generated by an off-the-shelf 2D pose detector, as the input. After that, PoseFormer employs a spatial-temporal transformer to exploit the local and global information of different poses. Finally, a regression head is used to predict the 3D pose of the center frames. The figure is from [127] (©2021 IEEE).

point generation decoder. Some methods consider reducing the computational costs when applying a transformer for point cloud completion. Mittal *et al.* [123] propose to first map the high-dimension 3D shape to low-dimension latent space and second perform transformer-based autoregressive modeling. On the other hand, Yan *et al.* [124] use a novel vector quantized deep implicit functions to sparsely encode 3D shape and employ a transformer module to predict conditional distribution of location and content.

Some methods adopt the attention module in the transformer to help feature extraction. For example, Lyu *et al.* [125] adopt the attention operation, instead of the pooling operation, to aggregate local features. Cao *et al.* [126] add a self-attention layer in global refinement module that is used to enhance the details and avoid distortion.

4.5 3D Pose Estimation

3D pose estimation aims to estimate 3D joint locations of objects from images or videos. Table 5 summarizes the related 3D pose estimation methods using the transformer structure. A few methods that fuse spatio-temporal information for video based pose estimation have been developed. Zheng *et al.* [127] propose a spatial-temporal transformer encoder module, PoseFormer, to model local relations within frame and global relations across frames. Fig. 10 shows the architecture of PoseFormer. Hassanin *et al.* [128] show that PoseFormer has the issue of poor locality and propose two novel cross interaction modules to integrate locality and inter interaction. Specifically, a cross-joint interaction module is used to encode local part information within frame, and a cross-frame interaction module is to encode information of the joints across frames. On the other hand, Li *et al.* [129] adopt a vanilla transformer to exploit long-range information and design a strided transformer to progressively aggregate long-range information of different frames into a single 3D representation. In [130], Shan *et al.* introduce a self-supervised pre-training method with transformer for 3D human pose estimation. Li *et al.* [132] propose to learn spatio-temporal representations of multiple pose hypotheses with the proposed multi-hypothesis transformer. In multi-hypothesis transformer, there are three modules: multi-hypothesis generation (MHG), self-hypothesis refinement (SHR), and cross-module interaction (CHI). MHG aims to explore spatial information within frame, and SHR and CHI aim to explore temporal information accross frames. Zhao *et al.* [133] develop a graph-oriented transformer to model the relation between different joints, which integrates graph convolution and attention together. These methods above employ transformer to predict single 3D pose estimation of the central frame in the video (called seq2frame). Different from these approaches, Zhang *et al.* [131] present MixSTE for 3D pose sequence estimation that peforms 3D pose estimation for all frames in the video (called seq2seq). MixSTE takes each 2D joint as a token and performs spatio-temporal attention in an alternating style.

TABLE 5
Summary of 3D pose estimation related approaches using the transformer. Some methods focus on videos or multi-frames, some methods focus on multi-view frames, some methods take single depth or RGB image as input, and some methods are for 6D pose estimation.

Method	Input	Arch.	Context	Highlight
Videos or multi-frames				
PoseFormer [127]	sequence	pure	global/ local	a spatial-temporal transformer encoder module
Crossformer [128]	sequence	pure	global/ local	a cross-joint interaction module and a cross-frame interaction module
STE [129]	sequence	pure	global/ local	vanilla and strided transformers for global and local information aggregation
P-STMO [130]	sequence	pure	global/ local	self-supervised pre-training technique with transformer
MixSTE [131]	sequence	pure	global/ local	spatio-temporal attention in an alternating style
MHFormer [132]	sequence	pure	global/ local	spatio-temporal representations of multiple pose hypotheses
GraFormer [133]	sequence	hybrid	global/ local	a novel model by combining graph convolution and transformer
Multi-view frames				
Epipolar [134]	multi-view	hybrid	local	enhance the feature of source view from and source view
RayTran [135]	multi-view	hybrid	local	a ray-traced transformer to progressively exchange information
Singe depth or RGB image				
Hand-transformer [136]	point	hybrid	global	a non-autoregressive hand transformer to model the relationships of input points
Cheng *et al.* [137]	point	hybrid	global	an attention based confidence network for multi-view feature fusion
METRO [138]	image	hybrid	global	a progressive dimensionality reduction transformer to predict 3D coordinates
HOT-Net [139]	image	hybrid	global	a hand-object transformer based on non-autoregressive transformer
HandsFormer [140]	image	hybrid	global	self-attention between the keypoints to associate them
PI-Net [141]	image	hybrid	global	a self-attention module to improve the embeddings of each person
Ugrinovic *et al.* [142]	image	hybrid	global	encode global information of multiple persons with set transformer
6D pose estimation				
6D-ViT [143]	image	pure	global	a two-branch transformer encoder-decoder structure
Dang *et al.* [144]	image	hybrid	global	a DCP-based architecture with transformer for feature extraction
Goodwin *et al.* [145]	image	pure	global	zero-shot category-level pose estimation

Instead of replying on single views, a few methods use multiple views for 3D pose estimation. He *et al.* [134] propose an epipolar transformer [146] to enhance the feature of reference view with the feature of the corresponding point in source view. Tyszkiewicz *et al.* [135] exploit features of multiple frames and empty volumetric feature as inputs, and employ a ray-traced transformer to progressively exchange information for 3D representation.

A few approaches estimate 3D pose from depth or RGB images. Huang *et al.* [136] introduce a non-autoregressive hand transformer (NARHT) to avoid sequential inference in transformer and achieve a fast inference speed. Inspired by non-autoregressive transformer [147], NARHT uses a structured-reference extractor to predict reference pose and model the relationships of input points and reference pose fin parallel. Cheng *et al.* [137] propose an attention based confidence network to predict the confidence of each virtual view and select the important views for depth-based hand pose estimation. In [138], Lin *et al.* present a progressive dimensionality reduction transformer to predict 3D coordinates of each joint for RGB-based human pose and mesh reconstruction. These methods above focus on single object pose estimation. Huang *et al.* [139] develop a hand-object transformer network to leverage the joint correlations between hand and object for hand-object pose estimation. In addition, some multi-object pose estimation methods are proposed. Hampali *et al.* [140] first extract a set of keypoints and second took their appearance and spatial encodings as the input to transformer for 3D hand and object pose estimation. In [141], Guo *et al.* employ a self-attention module to improve the embeddings of each person by integrating the embeddings of other persons. Recently, Ugrinovic *et al.* [142] exploit the set transformer [148] to encode global information of multiple persons for improved multi-person pose estimation.

Aside from 3D pose estimation, a few models have been developed to describe 6D pose information. Zou and Huang [143] design a two-branch transformer encoder-decoder structure to respectively extract features from image and point cloud, and perform 6D pose estimation based on the aggregated features of two branches. Dang *et al.* [144] employ a learn-based point cloud registration for 6D pose estimation, where a transformer is used to combine the features from two point inputs in point cloud registration architecture DCP [149]. Goodwin *et al.* [145] develop a novel zero-shot category-level 6D pose estimation task and employ a self-supervised transformer for feature extraction.

4.6 Other Tasks

3D Object Tracking. 3D Point Cloud object tracking aims to localize the object in 3D space given a template point cloud. Recently, a few 3D tracking methods based on transformers have been developed. Cui *et al.* [150] use a transformer to exploit local and global information within a point cloud and across different point clouds for 3D tracking prediction. Similarly, Zhou *et al.* [151] employ a self-attention module to capture long-range dependencies and a cross-attention module for a coarse matching.

3D Motion Prediction. 3D motion prediction aims to predict the future pose based on a history of past motion. Mao *et al.* [152] propose a motion attention module to capture the motion relation of the long-term history of motion. Similar to 3D pose estimation, some approaches [153], [154], [155] use transformers to capture spatial and temporal long-range dependency between joints. Gonzalez *et al.* take pose sequences [156] as the input and use non-autoregressive transformer [147] for motion prediction. Zheng *et al.* [157] propose an ego-centric motion prediction dataset and develop a cross-modal transformer module for this task.

3D Reconstruction. Wang *et al.* [158] focus on multi-view 3D object reconstruction and proposed a 3D volume transformer for feature extraction and fusion. Zanfir *et al.* [159] reconstruct human 3D shape from a monocular image with the THUNDR model. THUNDR predicts and regularizes an intermediate 3d marker representation with CNN feature extraction and transformer refinement. Mahmud and Frahm [160] develop single-view and multi-view object reconstruction methods, termed as VPFusion, with transformer-based feature fusion. VPFusion adopts transformer to perform cross-view feature fusion.

TABLE 6
Common datasets for 3D vision tasks. These datasets cover a range of 3D applications: classification, segmentation, detection, completion, and pose estimation. They are collected using different sensors for indoor scenes, outdoor scenes, and objects.

Dataset	Task	Size	Cls	Note
ModelNet40 [163]	classification	12,311 models	40	CAD models
ShapeNet [164]	segmentation	16,880 models	16	50 different parts
S3DIS [165]	segmentation	271 rooms	13	six areas
ScanObjectNN [166]	classification	2902 point clouds	15	with background/occlusions
SUN RGB-D [167]	detection	10,335 frames	37	Over 64,000 3D BB
ScanNet [168]	segm., det.	1513 scenes	40	100 hidden test scenes
KITTI [169]	detection	14,999 frames	3	80,256 labeled objects
NuScenes [170]	detection	1k scenes	23	40K key frames, 8 attributes
Completion3D [171]	completion	30,958 models	8	same size of 2048 × 3
PCN [172]	completion	30,974 models	8	less than 2048 points
Human3.6M [173]	pose estimation	3.6M video frames	17	Indoor with 11 actors
MPI-INF-3DHP [174]	pose estimation	14 camera views	8	indoor/outdoor with 8 actors

Point Cloud Registration. DCP [149] uses self-attention and conditional attention to approximate combinatorial matching between two points clouds. In [161], Fu *et al.* introduce a point cloud registration technique based on graph matching. It utilizes the transformer to generate edges for the graph construction. REGTR [162] employs the transformer for point cloud registration by predicting the probability of each point to lie in the overlapping region of two scans.

5 Benchmark Performance

To understand the effect of incorporating the transformer architecture into 3D vision pipelines, it is essential to compare against previous methods. In this section, we present quantitative comparisons of transformer-based architecture to state-of-the-art non-transformer methods on benchmark datasets. We start by providing details on the datasets and evaluation metrics, and then show the quantitative results for different tasks.

5.1 Datasets

Datasets play two important roles in the advancement of computer vision tasks. First, annotations allow training deep learning models to solve challenging problems. Second, datasets provide a basis for quantitative comparisons to measure the effectiveness of a proposed method. Numerous datasets have been developed for 3D vision tasks. For classification of 3D objects, ModelNet40 [163] is widely used, as well as ScanObjectNN [166]. For 3D segmentation, ShapeNet [164] provides part segmentation annotation, and S3DIS [165] and ScanNet [168] provide indoor scenes segmentation. For indoor 3D object detection, SUN RGB-D [167] provides oriented bounding boxes annotations for scenes from a single RGB-D frame, and ScanNet [168] can also be utilized by transforming instance segmentation labels into axis aligned bounding box annotations. For 3D

TABLE 7
Shape classification results on ModelNet40 benchmark (P: points, N:normals, *: pre-trained).

Method	Repr.	# points	Acc (%)
Non-Transformer methods:			
PointNet [23]	P	1k	89.2
PointNet++ [25]	P + N	5k	91.9
PointCNN [175]	P	1k	92.5
KPConv [28]	P	6.8k	92.9
DGCNN [26]	P	1k	92.9
RS-CNN [176]	P	1k	93.6
Transformer-based methods:			
A-ShapeContextNet [47]	P	1k	90.0
Yang *et al.* [51]	P	1k	91.7
3DETR [45]	P		91.9
Point Transformer [50]	P	1k	92.8
MLMSPT [68]	P	1k	92.9
DTNet [58]	P	1k	92.9
Wu *et al.* [67]	P	1k	93.2
LFT-Net [60]	P + N	2k	93.2
PCT [41]	P	1k	93.2
3DCTN [57]	P + N	1k	93.3
3DMedPT [66]	P	1k	93.4
3CROSSNet [65]	P	1k	93.5
PAT [64]	P + N	1k	93.6
Point Transformer [44]	P + N	1k	93.7
Point-BERT [61]	P	8k	93.8*
Point-MAE [63]	P	1k	93.8*
Adaptive Wavelet [56]	P	1k	93.9
CpT [59]	P	1k	93.9
Liu *et al.* [62]	P	1k	93.9
PVT [54]	P + N	1k	94.0

TABLE 8
Object classification results on ScanObjectNN dataset [166]. We report classification accuracy on the three main variants: OBJ-BG, OBJ-ONLY, and PB-T50-RS

Method	OBJ-BG	OBJ-ONLY	PB-T50-RS
Non-Transformer methods:			
PointNet++ [25]	82.3	84.3	77.9
DGCNN [26]	82.8	86.2	78.1
Transformer-based methods:			
Point-BERT [61]	87.43	88.12	83.07
Pang *et al.* [63]	90.02	88.29	85.18

object detection in outdoor scenes, KITTI [169] and nuScenes [170] are the most commonly used. For 3D point cloud completion, Completion3D [171] and PCN [172] are two of the most widely used datasets. For 3D pose estimation, Human3.6M [173] and MPI-INF-3DHP [174] are among the most popular datasets. The main attributes of these datasets are shown in Table 6.

5.2 Object Classification

We show benchmark performance of transformer-based 3D object classification methods on ModelNet40 dataset [163] in Table 7. This dataset is comprised of 9843 training CAD models and 2468 testing CAD models, labeled with 40 classes. We compare the performance of the transformer-based methods to state-of-the-art non-transformer methods. Given a CAD model, object classification methods uniformly sample a fixed number of points, and in some cases, append normal information to the input. In addition, some methods choose to pre-train the classification network on another dataset in a self-supervised way [61], [63]. The quantitative evaluations on ModelNet40 show competitive results for transformer-based architectures. The accuracy of multiple transformer methods has exceeded that of state-of-the-art non-transformer methods.

The classification accuracy for transformer-based methods as well as state-of-the-art non-transformer methods on the ScanObjectNN dataset [166] is shown in Table 8. This dataset represents a real-world scenario with point clouds containing background and occlusion. We show results with pre-training on ShapeNet dataset [164] and finetuning on ScanObjectNN. The transformer-based methods show improved performance over the non-transformer methods.

TABLE 9
Object Part Segmentation on ShapeNet [164] and scene semantic segmentation results on S3DIS [165]. We compare transformer-based methods to state-of-the-art non-transformer methods. For part segmentation, we show instance mean IoU, whereas for scene segmentation, we report mean accuracy (mAcc) and mean IoU.

	ShapeNet	S3DIS	
Method	ins. mIoU	mAcc	mIoU
Non-Transformer methods:			
ShapeNet [164]	81.4	-	-
PointNet [23]	83.7	49.0	41.1
PointNet++ [25]	85.1	-	51.5
DGCNN [26]	85.2	-	56.1
MinkowskiNet [31]	–	71.7	65.4
KPConv [28]	86.4	72.8	67.1
Transformer-based methods:			
Point Transformer [44]	86.6	76.5	70.4
Point Transformer [50]	85.9	-	-
A-ShapeContext [47]	84.6	-	52.7
Yang *et al.* [51]	-	70.8	60.1
PCT [41]	86.4	67.7	61.3
PVT [54]	86.6	-	68.2
Wavelet Transformer [56]	86.6	-	-
DTNet [58]	85.6	-	-
CpT [59]	86.6	72.6	62.3
LFT-Net [60]	86.2	76.2	65.2
Point-BERT [61]	85.6	-	-
Liu *et al.* [62]	86.6	-	-
Point-MAE [63]	86.1	-	-
PAT [64]	86.5	-	68.1
3CROSSNet [65]	85.3	-	-
3DMedPT [66]	84.3	-	-
MLMSPT [68]	86.4	-	62.9
Segment-Fusion [97]	-	-	65.3
Fast Point Transformer [78]	-	77.9	70.3
Stratified Transformer [96]	86.6	78.1	72.0

5.3 3D Segmentation

We show benchmark performance of various transformer-based methods for 3D segmentation in two domains: (1) object part segmentation and (2) complete scenes segmentation. Although the task is similar in these two domains, the required data size to properly represent the shape information is usually different.

For 3D object part segmentation, we evaluate on the widely used ShapeNet dataset [164]. It contains 14,007 training examples of point cloud objects and 2874 testing examples. Objects are of 16 different categories, and each category is labeled with 2 to 6 parts, with 50 parts in total. For evaluation, the intersection-over-union (IoU) of each category is computed as the average IoU of all the objects in that category. Additionally, the instance mIoU represents the average IoU across all object instances.

Quantitative evaluation on the ShapeNet dataset is shown in Table 9. Almost all transformer-based methods

TABLE 10
3D object detection comparison on SUN RGB-D [167] and ScanNet [168]. We show mean AP (@IoU=0.25) for transformer-based methods and compare them to state-of-the-art non-transformer methods.

Method	SUN RGB-D mAP25	ScanNet mAP25
Non-Transformer methods:		
VoteNet [84]	59.1	62.9
H3DNet [177]	60.1	67.2
Transformer-based methods:		
Pointformer [71]	61.1	64.1
Liu *et al.* [46]	63.0	69.1
3DETR [45]	59.1	65.0
Fast Point Tr. [78]	-	59.1
ARM3D [80]	60.1	65.9
MLCVNet [83]	59.8	64.5
BrT [92]	65.4	71.3

surpass state-of-the-art point-based schemes. When compared to the classification task which only requires a holistic understanding of an object for category labeling, part segmentation requires local shape understanding for proper segmentation. This shows that the increased attention in the transformer architecture to global context supports local shape understanding.

For scene semantic segmentation, we use the Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) [165]. It is collected using Matterport scanners and contains 3D point clouds of 3 indoor areas with 271 rooms from 3 different buildings. For evaluation, some methods perform 6-fold validation, whereas others use Area 5 as a test set and other areas to train the model. Since Area 5 is different from other areas, the latter approach is more common as it better shows the method's ability to generalize to a different scene. Quantitative comparison on the S3DIS dataset is shown in Table 9. The quantitative evaluation shows that the early adoptions of the attention mechanism [47] for 3D segmentation do not show significant performance gain over non-attention based methods. Recently, Stratified Transformer [96] achieves significant improvement and is currently state-of-the-art on the S3DIS dataset for 3D semantic segmentation. The performance gain can be attributed to the efficient transformer implementation, which applies the transformer in the local context to capture fine shape information, and uses a hierarchical structure and sparse distant sampling to capture global context using a large receptive field.

5.4 3D Object Detection

For indoor 3D object detection, the SUN RGB-D dataset [167] provides 3D bounding box annotations to scenes from a single RGB-D frame. It comprises 10,335 RGB-D frames annotated with amodal and oriented bounding boxes with 37 object classes. Nevertheless, a standard evaluation protocol is to use only 10 common classes for training and evaluation. The training set is composed of 5285 frames whereas the testing set has 5050 frames.

Quantitative evaluations of transformer models against state-of-the-art VoteNet [84] and H3DNet [177] methods on the SUN RGB-D are shown in Table 10. While the least performing transformer model is 3DETR, it is a straightforward transformer encoder and decoder without exploiting recent techniques in 3D object detection. Nevertheless, 3DETR performs as well as the VoteNet method, which has been adopted by other approaches [83], [177], [178]. The Bridged Transformer (BrT) [92] achieves the best performance on the SUN RGB-D dataset. It benefits from the availability of RGB images and the applicability of transformers to both images and point clouds in order to bridge the learning process between the two domains.

TABLE 11
Performance evaluations with state-of-the-art methods on the KITTI 3D object test set [169]. The results are reported by the mAP with 40 recall points, 0.7 IoU threshold for car, and 0.5 for others. (L: LiDAR, C: Color)

		Car (IoU=0.7)	Pedestrian (IoU=0.5)	Cyclist (IOU=0.5)
Method	Input	Easy/Mod./Hard	Easy/Mod./Hard	Easy/Mod./Hard
Non-Transformer methods:				
PointRCNN [179]	L	87.0 / 75.6 / 70.7	48.0 / 39.4 / 36.0	75.0 / 58.8 / 52.5
PV-RCNN [180]	L	90.3 / 81.4 / 76.8	52.2 / 43.3 / 40.3	78.6 / 63.7 / 57.7
Monoflex [181]	C	19.9 / 13.9 / 12.1	-	-
Transformer-based methods:				
Pointformer [71]	L	87.1 / 77.1 / 69.3	50.7 / 42.4 / 39.6	75.0 / 59.8 / 54.0
Voxel Trans. [42]	L	89.9 / 82.1 / 79.1	-	-
Sheng *et al.* [72]	L	87.8 / 81.8 / 77.2	-	-
SA-Det3D [75]	L	88.3 / 81.5 / 77.0	-	82.2 / 68.5 / 61.3
M3DETR [76]	L	90.3 / 81.7 / 77.0	45.7 / 39.9 / 37.7	83.8 / 66.7 / 59.0
Voxel Set Tr. [79]	L	88.5 / 82.1 / 77.5	-	-
Dao *et al.* [82]	L	87.1 / 80.3 / 76.1	-	78.5 / 64.6 / 57.8
SCANet [86]	L+C	76.1 / 66.3 / 58.7	-	-
MonoDETR [87]	C	25.0 / 16.5 / 13.6	-	-
CAT-Det [89]	L+C	89.9 / 81.3 / 76.7	54.3 / 45.4 / 41.9	83.7 / 68.8 / 61.5
PDV [94]	L	90.4 / 81.9 / 77.4	-	83.0 / 67.8 / 60.5

The ScanNet [168] dataset consists of scenes represented by 3D meshes based on reconstruction of multiple RGB-D frames. The original ScanNet dataset does not include 3D bounding boxes annotations, and the hidden test set in the benchmark does not include evaluation for 3D object detection. Thus, the common approach is to generate axis-aligned 3D bounding boxes are enclose the provided 3D instance segmentation masks. The training set contains 1201 scans, and the validation set of 312 scans is used for evaluation.

Performance evaluations on the ScanNet dataset are shown in Table 10. The Bridged Transformer (BrT) [92] achieves the highest mAP @IoU=0.5. Along with the 3D scans, it uses the 25,000 frames that were used to reconstruct ScanNet scenes. Images are required to extract patches that are used as input to a transformer bridging the information between images and point clouds. For methods that do not use the additional 2D images as input, Liu *et al.* [46] achieve the best performance through a group-free transformer-based method. This method uses PointNet++ to extract local point features, and the transformer generates the bounding boxes in a global fashion.

For outdoor 3D object detection, the KITTI dataset [169] is one of the most widely used datasets for autonomous driving and provides 3D object detection annotations. It contains 7481 training samples and 7518 testing samples and uses standard average precision (AP) on easy, moderate, and hard difficulties. The performance evaluations on KITTI benchmark are shown in Table 11. We also show the input used in each method, which can either be LiDAR only, RGB only, or both. Existing methods that rely on RGB alone under-perform the ones that make use of the LiDAR information. Since the task is to localize objects in the 3D scene, RGB alone lacks necessary 3D information required for accurately placing the bounding boxes. Nevertheless, the transformer-based architecture MonoDETR [87]

TABLE 12
Performance evaluations of transformer-based methods and state-of-the-art non-transformer methods on nuScenes 3D object detection test set [170]. We report the mAP for detection of 10 classes.

Method	Modality	mAP
Non-Transformer methods:		
PointPillars [38]	LiDAR	30.5
3DSSD [182]	LiDAR	42.7
CBGS [183]	LiDAR	52.8
FCOS3D [184]	images	35.8
Transformer-based methods:		
Yin *et al.* [85]	LiDAR	45.4
SA-Det3D [75]	LiDAR	47.0
Pointformer [71]	LiDAR	53.6
Dao *et al.* [82]	LiDAR	47.0
VISTA [93]	LiDAR	63.0
Transfusion [88]	LiDAR + images	68.9
PETR [90]	images	43.4
DETR3D [73]	images	41.2
Yuan *et al.* [81]	images	50.5

TABLE 13
Performance evaluations of point cloud completion on Completion3D [171] and PCN [172]. L2 Chamfer distance is used as metric on Completion3D, while L1 Chamfer distance is used as metric on PCN.

Method	Completion3D: Avg. ↓	PCN: Avg. ↓
Non-Transformer methods:		
PCN [172]	18.22	9.64
PMP-Net [185]	9.23	8.66
Transformer-based methods:		
PoinTr [117]	-	8.38
Snowflakenet [120]	7.60	7.21
PointAttN [119]	6.63	6.86
Wang *et al.* [118]	6.64	7.96

outperforms Monoflex [181]. For LiDAR input, PDV [94] achieves the best performance on the “easy” car category. It makes use of a self-attention module to capture long-range dependencies of grid points, where features are computed using 3D sparse convolutions. For the moderate and hard difficulties, the Voxel Transformer [42] and the Voxel Set Transformer [79] achieve the best performance, benefiting from the integration of the attention mechanism into the feature extraction module.

Another widely used outdoor dataset for 3D object detection is nuScenes [170], which contains 1000 scenes that include heavy traffic and challenging driving situations. We show detection results for various methods in Table 12. We compare transformer models using depth from LiDAR and/or images to state-of-the-art non-transformer methods. The transformer methods show improved performance for both the LiDAR-based and image-based 3D object detection. The best performance is achieved with the Transfusion method [88] in which a transformer fuses information from both the depth and color modalities. The results show that the transformer is capable of learning the complementary information from both modalities.

5.5 3D Point Cloud Completion

As discussed earlier, Completion3D [171] and PCN [172] are usually used for 3D point completion. Completion3D dataset has 30,958 point cloud models in 8 categories. The partial and complete point clouds have the same size of 2,048 $\times$ 3. The partial 3D point clouds are back-projection of depth images from 3D space, and L2 Chamfer distance is used for performance evaluation on Completion3D [171]. Table 13 (middle column) shows the point completion performance of some non-transformer and transformer-based methods on Completion3D dataset [171]. PCN [172] and PMP-Net [185] are non-transformer methods whereas PoinTr [117], Snowflakenet [120], PointAttN [119], and Wang *et al.* [118] are transformer-based methods. Clearly, methods based on transformers perform better for this task using Completion3D. Among these transformer-based methods, PointAttN [119] and Wang *et al.* [118] have the smallest L2 Chamfer distance (*i.e.,* 6.63 and 6.64), where they employ a transformer to extract global and local information for improved performance.

PCN [172] dataset has 30,974 point cloud models in 8 categories. The partial point clouds have less than 20,48 points, while the complete point clouds have 16,384 points. In the following experiments, L1 Chamfer distance is used for performance evaluation on PCN dataset [172]. Table 13 (right column) shows the point completion performance on PCN [171]. Among these transformer-based methods, Snowflakenet [120] and PointAttN [119] have the smallest L1 Chamfer distance (*i.e.,* 7.21 and 6.86).

TABLE 14
3D pose estimation performance comparison on Human3.6M [173] under Protocols 1&2 where 2D pose detection is used as input.

Method	Protocol 1: Avg. ↓	Protocol 2: Avg. ↓
Non-Transformer methods:		
UGCN [186]	45.6	35.5
Chen *et al.* [187]	44.6	35.6
Transformer-based methods:		
METRO [138]	54.0	36.7
PoseFormer [127]	44.3	34.6
CrossFormer [128]	43.7	34.3
STE [129]	43.7	35.2
P-STMO [130]	44.1	34.4
MHFormer [132]	43.0	-
MixSTE [131]	42.4	33.9

TABLE 15
3D pose estimation performance comparison on MPI-INF-3DHP [174].

Method	PCK ↑	AUC ↑	MPJPE ↓
Non-Transformer methods:			
Chen *et al.* [187]	87.9	54.0	78.8
Transformer-based methods:			
PoseFormer [127]	88.6	56.4	77.1
CrossFormer [128]	89.1	57.5	76.3
P-STMO [130]	97.9	75.8	32.2
MHFormer [132]	93.8	63.3	58.0
MixSTE [131]	94.2	63.8	57.9

5.6 3D Pose estimation

Human3.6M [173] and MPI-INF-3DHP [174] are two widely used 3D pose estimation datasets for 3D pose estimation. The Human3.6M dataset [173] contains 3.6 million video frames recorded in 4 different views of the indoor scene. There are 17 action categories performed by 11 different actors. Two metrics (MPJPE and P-MPJPE) are used for performance evaluation. The mean per joint position error (MPJPE) metric, denoted as Protocol 1, is calculated as the averaged Euclidean distance in millimeters between the predicted and ground-truth joints. The P-MPJPE metric, denoted as Protocol 2, is the MPJPE after the post-processing of rigid alignment. Table 14 shows the 3D pose estimation performance of some non-transformer and transformer-based methods on Completion3D [171]. Compared to non-transformer schemes, most transformer-based methods usually have better performance under both Protocol 1 and Protocol 2. Among these transformer-based methods, MixSTE [131] has the best performance with an average MPJPE of 42.4 and 33.9 under Protocol 1 and Protocol 2. MixSTE consists of a temporal transformer and a spatial transformer, where the two transformers are performed in an alternating style. The temporal transformer is to learn the temporal relation of each joint, while the spatial transformer is to build the spatial correlation of different joints.

The MPI-INF-3DHP dataset [174] consists of 1.3 million frames recorded in both indoor and outdoor scenes. There are 8 action categories performed by 8 different actors. Table 15 shows the results of some methods. Three metrics (MPJPE, PCK, and AUC) are used for performance evaluation. The PCK metric is a 3D extension of the percentage of correct keypoints with a threshold of 150mm, while AUC is calculated with a range of PCK thresholds. Among these transformer-based methods, P-STMO [130] has the best performance, with PCK of 97.9, AUC of 75.8, and MPJPE of 32.2. P-STMO employs the self-supervised pre-training transformer for pose estimation.

6 Discussion and Conclusion

Integrating the transformer into the pipeline of 3D applications has been shown to be effective in numerous areas. Given the competitive performance on multiple datasets, the transformer was demonstrated to be an adequate replacement for convolution and multi-layer perceptron operations, thanks to its ability to learn long-range dependencies. Nonetheless, a generic transformer backbone for 3D processing is still missing. Unlike image processing with transformers which has focal methods that many other methods rely on [69], [112], most transformer-based 3D methods use different transformer designs and integration. It is of great interest to develop a general-purpose transformer method that processes point clouds and learns rich features on the local and global scale. The transformer is required to learn fine shape information, and at the same time operate on a scene global scale to make use of the scene context.

Additionally, most transformer-based 3D methods subsample 3D data to a fixed size input. A fixed input size is the norm in images considering the given number of pixels, but 3D input size usually varies. Therefore, a proper sampling strategy that (1) preserves the potential to learn fine information, (2) maintains similar object structure irrespective of the scene size, and (3) generates a feasible data size for transformer processing, is essential. Although NLP processes inputs of varying sizes, words in sentences are all tokenized, and input information is not lost. On the other hand, 3D input sampling leaves out potentially important information and is dependent on the scene size. A sampling strategy worth exploring is to use a data-driven approach. For example, a small sub-sample of the input can be used as seeds for a flexible and data-driven sampling. This is similar to previous input aggregation methods, but sampling would be adapted to input information, and original information would always be accessible.

The choice of the position embedding is also essential. The position information in the 3D field not only orders the information sequences but is also the main feature for understanding shape information. While most 3D vision methods use the 3D position as part of the input to the transformer, this information might require prior encoding and might not be explicitly available after multiple feature extraction layers. Therefore, a proper position embedding would benefit the learning task. The choice of the position embedding should help models learn translation invariance and retain context information from other scene elements.

Existing transformer models rely on data augmentation in the training process. Although numerous transformer methods use off-the-shelf 3D augmentation methods, several 3D augmentation techniques have shown significant improvement over common non-transformer methods [188], [189]. A similar trend could also benefit transformer based methods, with the utilization of 3D-specific augmentations that are well suited for training transformers.

Compared to NLP or 2D image processing, 3D vision rely less on pre-training. This is also observed in transformer-based methods, where pre-training is seldom used. Large-scale fully-supervised or self-supervised pre-training has helped improve performance as well as the robustness of 2D models. Similar pre-training models would also benefit the 3D domain. This can either be done with learning on a fully labeled large-scale dataset, akin to ImageNet in 2D, or with self-supervised learning.

References

[1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *TPAMI*, vol. 43, no. 12, pp. 4338–4364, 2020. 2

[2] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3d data: A survey," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–38, 2017. 2

[3] S. A. Bello, S. Yu, C. Wang, J. M. Adam, and J. Li, "Deep learning on 3d point clouds," *Remote. Sens.*, vol. 12, no. 11, p. 1729, 2020. 2

[4] W. Liu, J. Sun, W. Li, T. Hu, and P. Wang, "Deep learning on point clouds and its application: A survey," *Sensors*, vol. 19, no. 19, p. 4188, 2019. 2

[5] Y. He, H. Yu, X. Liu, Z. Yang, W. Sun, Y. Wang, Q. Fu, Y. Zou, and A. Mian, "Deep learning based 3d segmentation: A survey," *arXiv preprint arXiv:2103.05423*, 2021. 2

[6] B. Gao, Y. Pan, C. Li, S. Geng, and H. Zhao, "Are we hungry for 3d lidar data for semantic segmentation? a survey of datasets and methods," *T-ITS*, vol. 23, no. 7, pp. 6063–6083, 2022. 2

[7] Y. Xie, J. Tian, and X. X. Zhu, "Linking points with labels in 3d: A review of point cloud semantic segmentation," *IEEE Geoscience and Remote Sensing Magazine*, vol. 8, no. 4, pp. 38–59, 2020. 2

[8] D. Griffiths and J. Boehm, "A review on deep learning techniques for 3d sensed data classification," *Remote. Sens.*, vol. 11, no. 12, p. 1499, 2019. 2

[9] D. Fernandes, A. Silva, R. Névoa, C. Simões, D. Gonzalez, M. Guevara, P. Novais, J. Monteiro, and P. Melo-Pinto, "Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy," *Information Fusion*, vol. 68, pp. 161–191, 2021. 2

[10] Y. Wu, Y. Wang, S. Zhang, and H. Ogai, "Deep 3d object detection networks using lidar data: A review," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1152–1171, 2020. 2

[11] Y.-P. Xiao, Y.-K. Lai, F.-L. Zhang, C. Li, and L. Gao, "A survey on deep geometry learning: From a representation perspective," *Computational Visual Media*, vol. 6, no. 2, pp. 113–133, 2020. 2

[12] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, and B. Ottersten, "A survey on deep learning advances on different 3d data representations," *arXiv preprint arXiv:1808.01462*, 2018. 2

[13] Y. Li, L. Ma, Z. Zhong, F. Liu, M. A. Chapman, D. Cao, and J. Li, "Deep learning for lidar point clouds in autonomous driving: A review," *TNNLS*, vol. 32, no. 8, pp. 3412–3432, 2020. 2

[14] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, 2021. 2

[15] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, "A survey of visual transformers," *arXiv preprint arXiv:2111.06091*, 2021. 2

[16] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on vision transformer," *TPAMI*, 2022. 2

[17] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *arXiv preprint arXiv:2106.04554*, 2021. 2

[18] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu, "Attention mechanisms in computer vision: A survey," *Computational Visual Media*, pp. 1–38, 2022. 2

[19] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *ACM Comput. Surv.*, 2022. 2

[20] F. Shamshad, S. Khan, S. W. Zamir, M. H. Khan, M. Hayat, F. S. Khan, and H. Fu, "Transformers in medical imaging: A survey," *arXiv preprint arXiv:2201.09873*, 2022. 2

[21] J. Selva, A. S. Johansen, S. Escalera, K. Nasrollahi, T. B. Moeslund, and A. Clapés, "Video transformers: A survey," *arXiv preprint arXiv:2201.05991*, 2022. 2

[22] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *SIGGRAPH*, 1994, pp. 311–318. 2

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017, pp. 652–660. 3, 6, 14

[24] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *IROS*, 2015, pp. 922–928. 3

[25] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NeurIPS*, 2017, pp. 5099–5108. 3, 7, 10, 14

[26] 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 TOG*, vol. 38, no. 5, pp. 1–12, 2019. 3, 6, 14

[27] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *ECCV*, 2018, pp. 87–102. 3

[28] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *CVPR*, 2019, pp. 6411–6420. 3, 9, 14

[29] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," in *CVPR*, 2019, pp. 9621–9630. 3

[30] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *CVPR*, 2018, pp. 9224–9232. 3

[31] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *CVPR*, 2019, pp. 3075–3084. 3, 14

[32] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-cnn: Octree-based convolutional neural networks for 3d shape analysis," *ACM TOG*, vol. 36, no. 4, pp. 1–11, 2017. 3

[33] G. Riegler, A. Osman Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *CVPR*, 2017, pp. 3577–3586. 3

[34] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *ICCV*, 2015, pp. 945–953. 3

[35] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *CVPR*, 2017, pp. 1907–1915. 3

[36] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3d shape segmentation with projective convolutional networks," in *CVPR*, 2017, pp. 3779–3788. 3

[37] A. Kanezaki, Y. Matsushita, and Y. Nishida, "Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *CVPR*, 2018, pp. 5010–5019. 3

[38] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *CVPR*, 2019, pp. 12 697–12 705. 3, 16

[39] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, "Tangent convolutions for dense prediction in 3d," in *CVPR*, 2018, pp. 3887–3896. 3

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008. 3

[41] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021. 4, 6, 7, 9, 14

[42] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, and C. Xu, "Voxel transformer for 3d object detection," in *CVPR*, 2021, pp. 3164–3173. 4, 7, 8, 15, 16

[43] A. Hatamizadeh, Y. Tang, V. Nath, D. Yang, A. Myronenko, B. Landman, H. R. Roth, and D. Xu, "Unetr: Transformers for 3d medical image segmentation," in *WACV*, 2022, pp. 574–584. 4, 10

[44] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *CVPR*, 2021, pp. 16 259–16 268. 4, 5, 6, 7, 9, 14

[45] I. Misra, R. Girdhar, and A. Joulin, "An end-to-end transformer model for 3d object detection," in *CVPR*, 2021, pp. 2906–2917. 4, 7, 8, 9, 14, 15

[46] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong, "Group-free 3d object detection via transformers," in *CVPR*, 2021, pp. 2949–2958. 4, 7, 8, 15

[47] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional shapecontextnet for point cloud recognition," in *CVPR*, 2018, pp. 4606–4615. 4, 5, 6, 9, 14, 15

[48] Y. Xie, J. Zhang, C. Shen, and Y. Xia, "Cotr: Efficiently bridging cnn and transformer for 3d medical image segmentation," in *MICCAI*, 2021, pp. 171–180. 4, 10

[49] Y. Zhang, H. Liu, and Q. Hu, "Transfuse: Fusing transformers and cnns for medical image segmentation," in *MICCAI*, 2021, pp. 14–24. 4, 10, 11

[50] N. Engel, V. Belagiannis, and K. Dietmayer, "Point transformer," *IEEE Access*, vol. 9, pp. 134 826–134 840, 2021. 6, 14

[51] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, "Modeling point clouds with self-attention and gumbel subset sampling," in *CVPR*, 2019, pp. 3323–3332. 6, 9, 14

[52] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *CVPR*, 2017, pp. 1251–1258. 6

[53] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *CVPR*, 2018, pp. 6848–6856. 6

[54] C. Zhang, H. Wan, S. Liu, X. Shen, and Z. Wu, "Pvt: Point-voxel transformer for 3d deep learning," *arXiv preprint arXiv:2108.06076*, 2021. 6, 9, 14

[55] Y. Zhou, A. Ji, and L. Zhang, "Sewer defect detection from 3d point clouds using a transformer-based deep learning model," *Automation in Construction*, vol. 136, p. 104163, 2022. 6

[56] H. Huang and Y. Fang, "Adaptive wavelet transformer network for 3d shape representation learning," in *ICLR*, 2021. 5, 6, 9, 14

[57] D. Lu, Q. Xie, L. Xu, and J. Li, "3dctn: 3d convolution-transformer network for point cloud classification," *arXiv preprint arXiv:2203.00828*, 2022. 5, 6, 14

[58] X.-F. Han, Y.-F. Jin, H.-X. Cheng, and G.-Q. Xiao, "Dual transformer for point cloud analysis," *arXiv preprint arXiv:2104.13044*, 2021. 6, 9, 14

[59] C. Kaul, J. Mitton, H. Dai, and R. Murray-Smith, "Cpt: Convolutional point transformer for 3d point cloud processing," *arXiv preprint arXiv:2111.10866*, 2021. 6, 14

[60] Y. Gao, X. Liu, J. Li, Z. Fang, X. Jiang, and K. M. S. Huq, "Lft-net: Local feature transformer network for point clouds analysis," *T-ITS*, 2022. 5, 6, 14

[61] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-bert: Pre-training 3d point cloud transformers with masked point modeling," *arXiv preprint arXiv:2111.14819*, 2021. 6, 14

[62] S. Liu, K. Fu, M. Wang, and Z. Song, "Group-in-group relation-based transformer for 3d point cloud learning," *Remote. Sens.*, vol. 14, no. 7, p. 1563, 2022. 6, 14

[63] Y. Pang, W. Wang, F. E. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked autoencoders for point cloud self-supervised learning," *arXiv preprint arXiv:2203.06604*, 2022. 6, 14

[64] Z. Cheng, H. Wan, X. Shen, and Z. Wu, "Patchformer: A versatile 3d transformer based on patch attention," *arXiv preprint arXiv:2111.00207*, 2021. 6, 7, 14

[65] X.-F. Han, Z.-Y. He, J. Chen, and G.-Q. Xiao, "3crossnet: Cross-level cross-scale cross-attention network for point cloud representation," *RA-L*, vol. 7, no. 2, pp. 3718–3725, 2022. 6, 7, 14

[66] J. Yu, C. Zhang, H. Wang, D. Zhang, Y. Song, T. Xiang, D. Liu, and W. Cai, "3d medical point transformer: Introducing convolution to attention networks for medical point cloud analysis," *arXiv preprint arXiv:2112.04863*, 2021. 6, 14

[67] L. Wu, X. Liu, and Q. Liu, "Centroid transformers: Learning to abstract with attention," *arXiv preprint arXiv:2102.08606*, 2021. 6, 7, 14

[68] X.-F. Han, Y.-J. Kuang, and G.-Q. Xiao, "Point cloud learning with transformer," *arXiv preprint arXiv:2104.13636*, 2021. 6, 7, 14

[69] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021. 6, 17

[70] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. 6

[71] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," in *CVPR*, 2021, pp. 7463–7472. 7, 8, 15, 16

[72] H. Sheng, S. Cai, Y. Liu, B. Deng, J. Huang, X.-S. Hua, and M.-J. Zhao, "Improving 3d object detection with channel-wise transformer," in *CVPR*, 2021, pp. 2743–2752. 8, 15

[73] Y. Wang, V. C. Guizilini, T. Zhang, Y. Wang, H. Zhao, and J. Solomon, "Detr3d: 3d object detection from multi-view images via 3d-to-2d queries," in *CoRL*, 2022, pp. 180–191. 8, 9, 16

[74] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020, pp. 213–229. 8, 9

[75] P. Bhattacharyya, C. Huang, and K. Czarnecki, "Sa-det3d: Self-attention based context-aware 3d object detection," in *CVPR*, 2021, pp. 3022–3031. 8, 9, 15, 16

[76] T. Guan, J. Wang, S. Lan, R. Chandra, Z. Wu, L. Davis, and D. Manocha, "M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers," in *WACV*, 2022, pp. 772–782. 8, 9, 15

[77] L. Fan, Z. Pang, T. Zhang, Y.-X. Wang, H. Zhao, F. Wang, N. Wang, and Z. Zhang, "Embracing single stride 3d object detector with sparse transformer," *arXiv preprint arXiv:2112.06375*, 2021. 7, 8

[78] C. Park, Y. Jeong, M. Cho, and J. Park, "Fast point transformer," *arXiv preprint arXiv:2112.04702*, 2021. 7, 8, 9, 10, 14, 15

[79] C. He, R. Li, S. Li, and L. Zhang, "Voxel set transformer: A set-to-set approach to 3d object detection from point clouds," *arXiv preprint arXiv:2203.10314*, 2022. 8, 15, 16

[80] Y. Lan, Y. Duan, C. Liu, C. Zhu, Y. Xiong, H. Huang, and K. Xu, "Arm3d: Attention-based relation module for indoor 3d object detection," *Computational Visual Media*, pp. 1–20, 2022. 7, 8, 15

[81] Z. Yuan, X. Song, L. Bai, Z. Wang, and W. Ouyang, "Temporal-channel transformer for 3d lidar-based video object detection for autonomous driving," *TCSVT*, vol. 32, no. 4, pp. 2068–2078, 2022. 8, 9, 16

[82] M.-Q. Dao, E. Héry, and V. Frémont, "Attention-based proposals refinement for 3d object detection," *arXiv preprint arXiv:2201.07070*, 2022. 8, 9, 15, 16

[83] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang, "Mlcvnet: Multi-level context votenet for 3d object detection," in *CVPR*, 2020, pp. 10 447–10 456. 7, 8, 15

[84] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *CVPR*, 2019, pp. 9277–9286. 7, 8, 15

[85] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention," in *CVPR*, 2020, pp. 11 495–11 504. 8, 9, 16

[86] H. Lu, X. Chen, G. Zhang, Q. Zhou, Y. Ma, and Y. Zhao, "Scanet: Spatial-channel attention network for 3d object detection," in *ICASSP*, 2019, pp. 1992–1996. 8, 15

[87] R. Zhang, H. Qiu, T. Wang, X. Xu, Z. Guo, Y. Qiao, P. Gao, and H. Li, "Monodetr: Depth-aware transformer for monocular 3d object detection," *arXiv preprint arXiv:2203.13310*, 2022. 8, 15

[88] X. Bai, Z. Hu, X. Zhu, Q. Huang, Y. Chen, H. Fu, and C.-L. Tai, "Transfusion: Robust lidar-camera fusion for 3d object detection with transformers," *arXiv preprint arXiv:2203.11496*, 2022. 8, 16

[89] Y. Zhang, J. Chen, and D. Huang, "Cat-det: Contrastively augmented transformer for multi-modal 3d object detection," *arXiv preprint arXiv:2204.00325*, 2022. 8, 9, 15

[90] Y. Liu, T. Wang, X. Zhang, and J. Sun, "Petr: Position embedding transformation for multi-view 3d object detection," *arXiv preprint arXiv:2203.05625*, 2022. 8, 9, 16

[91] D.-K. Nguyen, J. Ju, O. Booji, M. R. Oswald, and C. G. Snoek, "Boxer: Box-attention for 2d and 3d transformers," *arXiv preprint arXiv:2111.13087*, 2021. 8

[92] Y. Wang, T. Ye, L. Cao, W. Huang, F. Sun, F. He, and D. Tao, "Bridged transformer for vision and point cloud 3d object detection," in *CVPR*, 2022, pp. 12 114–12 123. 7, 8, 15

[93] S. Deng, Z. Liang, L. Sun, and K. Jia, "Vista: Boosting 3d object detection via dual cross-view spatial attention," in *CVPR*, 2022, pp. 8448–8457. 8, 9, 16

[94] J. S. Hu, T. Kuai, and S. L. Waslander, "Point density-aware voxels for lidar 3d object detection," in *CVPR*, 2022, pp. 8469–8478. 8, 15, 16

[95] H. Liu, H. Ma, Y. Chen, X. Li, and T. Hu, "Plnl-3dssd: Part-aware 3d single stage detector using local and non-local attention," in *ICIP*, 2021, pp. 3148–3152. 8

[96] X. Lai, J. Liu, L. Jiang, L. Wang, H. Zhao, S. Liu, X. Qi, and J. Jia, "Stratified transformer for 3d point cloud segmentation," in *arXiv preprint arXiv:2203.14508*, 2022. 9, 10, 14, 15

[97] A. Thyagharajan, B. Ummenhofer, P. Laddha, O. J. Omer, and S. Subramoney, "Segment-fusion: Hierarchical context fusion for robust 3d semantic segmentation," in *CVPR*, 2022, pp. 1236–1245. 10, 14

[98] H. Fan, Y. Yang, and M. Kankanhalli, "Point 4d transformer networks for spatio-temporal modeling in point cloud videos," in *CVPR*, 2021, pp. 14 204–14 213. 10

[99] Y. Wei, H. Liu, T. Xie, Q. Ke, and Y. Guo, "Spatial-temporal transformer for 3d point cloud sequences," in *WACV*, 2022, pp. 1171–1180. 10

[100] S. Xu, R. Wan, M. Ye, X. Zou, and T. Cao, "Sparse cross-scale attention network for efficient lidar panoptic segmentation," *arXiv preprint arXiv:2201.05972*, 2022. 10

[101] Y. Wu, K. Liao, J. Chen, D. Z. Chen, J. Wang, H. Gao, and J. Wu, "D-former: A u-shaped dilated transformer for 3d medical image segmentation," *arXiv preprint arXiv:2201.00462*, 2022. 10, 11

[102] D. Yang, A. Myronenko, X. Wang, Z. Xu, H. R. Roth, and D. Xu, "T-automl: Automated machine learning for lesion segmentation using transformers in 3d medical imaging," in *CVPR*, 2021, pp. 3962–3974. 10

[103] D. Karimi, S. D. Vasylechko, and A. Gholipour, "Convolution-free medical image segmentation using transformers," in *MICCAI*, 2021, pp. 78–88. 10, 11

[104] B. Yun, Y. Wang, J. Chen, H. Wang, W. Shen, and Q. Li, "Spectr: Spectral transformer for hyperspectral pathology image segmentation," *arXiv preprint arXiv:2103.03604*, 2021. 10, 11

[105] W. Wang, C. Chen, M. Ding, H. Yu, S. Zha, and J. Li, "Transbts: Multimodal brain tumor segmentation using transformer," in *MICCAI*, 2021, pp. 109–119. 10, 11

[106] S. Li, X. Sui, X. Luo, X. Xu, Y. Liu, and R. Goh, "Medical image segmentation using squeeze-and-expansion transformers," *arXiv preprint arXiv:2105.09511*, 2021. 10, 11

[107] H.-Y. Zhou, J. Guo, Y. Zhang, L. Yu, L. Wang, and Y. Yu, "nnformer: Interleaved transformer for volumetric segmentation," *arXiv preprint arXiv:2109.03201*, 2021. 10, 11

[108] Q. Jia and H. Shu, "Bitr-unet: a cnn-transformer combined network for mri brain tumor segmentation," *arXiv preprint arXiv:2109.12271*, 2021. 10, 11

[109] X. Yan, H. Tang, S. Sun, H. Ma, D. Kong, and X. Xie, "After-unet: Axial fusion transformer unet for medical image segmentation," in *WACV*, 2022, pp. 3971–3981. 10, 11

[110] H. Peiris, M. Hayat, Z. Chen, G. Egan, and M. Harandi, "A volumetric transformer for accurate 3d tumor segmentation," *arXiv preprint arXiv:2111.13300*, 2021. 10, 11

[111] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. Roth, and D. Xu, "Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images," *arXiv preprint arXiv:2201.01266*, 2022. 10, 11

[112] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *CVPR*, 2021, pp. 10 012–10 022. 10, 11, 17

[113] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network," in *AAAI*, 2019, pp. 8778–8785. 9

[114] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015, pp. 234–241. 9

[115] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *MICCAI*, 2016, pp. 424–432. 11

[116] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *ECCV*, 2018, pp. 3–19. 11

[117] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "Pointr: Diverse point cloud completion with geometry-aware transformers," in *ICCV*, 2021, pp. 12 498–12 507. 11, 16

[118] Y. Wang, D. J. Tan, N. Navab, and F. Tombari, "Learning local displacements for point cloud completion," in *CVPR*, 2022, pp. 1568–1577. 11, 16

[119] J. Wang, Y. Cui, D. Guo, J. Li, Q. Liu, and C. Shen, "Pointattn: You only need attention for point cloud completion," *arXiv preprint arXiv:2203.08485*, 2021. 11, 16

[120] P. Xiang, X. Wen, Y.-S. Liu, Y.-P. Cao, P. Wan, W. Zheng, and Z. Han, "Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer," in *ICCV*, 2021, pp. 5499–5501. 11, 16

[121] Z. Su, H. Huang, C. Ma, H. Huang, and R. Hu, "Point cloud completion on structured feature map with feedback network," *Computational Visual Media*, 2022. 11

[122] J. Lin, M. Rickert, A. Perzylo, and A. Knoll, "Pctma-net: Point cloud transformer with mborphing atlas-based point generation network for dense point cloud completion," in *ICRA*, 2021, pp. 5657–5663. 11

[123] P. Mittal, Y.-C. Cheng, M. Singh, and S. Tulsiani, "Autosdf: Shape priors for 3d completion, reconstruction and generation," in *CVPR*, 2022, pp. 306–315. 11, 12

[124] X. Yan, L. Lin, N. J. Mitra, D. Lischinski, D. Cohen-Or, and H. Huang, "Shapeformer: Transformer-based shape completion via sparse representation," in *CVPR*, 2022, pp. 6239–6249. 11, 12

[125] Z. Lyu, Z. Kong, X. Xu, L. Pan, and D. Lin, "A conditional point diffusion-refinement paradigm for 3d point cloud completion," in *ICLR*, 2022. 11, 12

[126] Z. Cao, W. Zhang, X. Wen, Z. Dong, Y. shen Liu, and B. Yang, "Mfm-net: Unpaired shape completion network with multi-stage feature matching," *arXiv preprint arXiv:2111.11976*, 2021. 11, 12

[127] C. Zheng, S. Zhu, M. Mendieta, T. Yang, C. Chen, and Z. Ding, "3d human pose estimation with spatial and temporal transformers," in *ICCV*, 2021, pp. 11 656–11 665. 12, 16

[128] M. Hassanin, A. Khamiss, M. Bennamoun, F. Boussaid, and I. Radwan, "Crossformer: Cross spatio-temporal transformer for 3d human pose estimation," *arXiv preprint arXiv:2203.13387*, 2022. 12, 16

[129] W. Li, H. Liu, R. Ding, M. Liu, P. Wang, and W. Yang, "Exploiting temporal contexts with strided transformer for 3d human pose estimation," *TMM*, 2022. 12, 16

[130] W. Shan, Z. Liu, X. Zhang, S. Wang, S. Ma, and W. Gao, "P-stmo: Pre-trained spatial temporal many-to-one model for 3d human pose estimation," *arXiv preprint arXiv:2203.07628*, 2022. 12, 16, 17

[131] J. Zhang, Z. Tu, J. Yang, Y. Chen, and J. Yuan, "Mixste: Seq2seq mixed spatio-temporal encoder for 3d human pose estimation in video," in *CVPR*, 2022, pp. 13 232–13 242. 12, 16, 17

[132] W. Li, H. Liu, H. Tang, P. Wang, and L. V. Gool, "Mhformer: Multi-hypothesis transformer for 3d human pose estimation," in *CVPR*, 2022, pp. 13 147–13 156. 12, 16

[133] W. Zhao, W. Wang, and Y. Tian, "Graformer: Graph convolution transformer for 3d pose estimation," in *CVPR*, 2022, pp. 20 438–20 447. 12

[134] Y. He, R. Yan, K. Fragkiadaki, and S.-I. Yu, "Epipolar transformer for multi-view human pose estimation," in *CVPR Workshops*, 2020, pp. 4466–4471. 12

[135] M. J. Tyszkiewicz, K.-K. Maninis, S. Popov, and V. Ferrari, "Raytran: 3d pose estimation and shape reconstruction of multiple objects from videos with ray-traced transformers," *arXiv preprint arXiv:2203.13296*, 2022. 12

[136] L. Huang, J. Tan, J. Liu, and J. Yuan, "Hand-transformer: Non-autoregressive structured modeling for 3d hand pose estimation," in *ECCV*, 2020, pp. 17–33. 12

[137] J. Cheng, Y. Wan, D. Zuo, C. Ma, J. Gu, P. Tan, H. Wang, X. Deng, and Y. Zhang, "Efficient virtual view selection for 3d hand pose estimation," in *AAAI*, 2022, pp. 419–426. 12, 13

[138] K. Lin, L. Wang, and Z. Liu, "End-to-end human pose and mesh reconstruction with transformers," in *CVPR*, 2021, pp. 1954–1963. 12, 13, 16

[139] L. Huang, J. Tan, J. Meng, J. Liu, and J. Yuan, "Hot-net: Non-autoregressive transformer for 3d hand-object pose estimation," in *ACM MM*, 2020, pp. 3136–3145. 12, 13

[140] S. Hampali, S. D. Sarkar, M. Rad, and V. Lepetit, "Keypoint transformer: Solving joint identification in challenging hands and object interactions for accurate 3d pose estimation," *arXiv preprint arXiv:2104.14639*, 2021. 12, 13

[141] W. Guo, E. Corona, F. Moreno-Noguer, and X. Alameda-Pineda, "Pi-net: Pose interacting network for multi-person monocular 3d pose estimation," in *WACV*, 2021, pp. 2796–2806. 12, 13

[142] N. Ugrinovic, A. Ruiz, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Permutation-invariant relational network for multi-person 3d pose estimation," *arXiv preprint arXiv:2204.04913*, 2022. 12, 13

[143] L. Zou, Z. Huang, N. Gu, and G. Wang, "6d-vit: Category-level 6d object pose estimation via transformer-based instance representation learning," *arXiv preprint arXiv:2110.04792*, 2021. 12, 13

[144] Z. Dang, L. Wang, Y. Guo, and M. Salzmann, "Learning-based point cloud registration for 6d object pose estimation in the real world," *arXiv preprint arXiv:2203.15309*, 2022. 12, 13

[145] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Zero-shot category-level object pose estimation," *arXiv preprint arXiv:2204.03635*, 2022. 12, 13

[146] Y. He, R. Yan, K. Fragkiadaki, and S.-I. Yu, "Epipolar transformers," in *CVPR*, 2020, pp. 7779–7788. 12

[147] J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher, "Non-autoregressive neural machine translation," in *ICLR*, 2018. 13

[148] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," *arXiv preprint arXiv:1810.00825*, 2019. 13

[149] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *CVPR*, 2019, pp. 3523–3532. 13

[150] Y. Cui, Z. Fang, J. Shan, Z. Gu, and S. Zhou, "3d object tracking with transformer," in *BMVC*, 2021. 13

[151] C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu, "Pttr: Relational 3d point cloud object tracking with transformer," in *CVPR*, 2022, pp. 8531–8540. 13

[152] W. Mao, M. Liu, and M. Salzmann, "History repeats itself: Human motion prediction via motion attention," in *ECCV*, 2020, pp. 474–489. 13

[153] Y. Cai, L. Huang, Y. Wang, T.-J. Cham, J. Cai, J. Yuan, J. Liu, X. Yang, Y. Zhu, X. Shen, D. Liu, J. Liu, and N. M. Thalmann, "Learning progressive joint propagation for human motion prediction," in *ECCV*, 2020, pp. 226–242. 13

[154] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges, "A spatio-temporal transformer for 3d human motion prediction," in *3DV*, 2021, pp. 565–574. 13

[155] O. Medjaouri and K. Desai, "Hr-stan: High-resolution spatio-temporal attention network for 3d human motion prediction," in *CVPR Workshops*, 2022, pp. 2540–2549. 13

[156] A. Martinez-Gonzalez, M. Villamizar, and J.-M. Odobez, "Pose transformers (potr): Human motion prediction with non-autoregressive transformers," in *ICCV Workshops*, 2021, pp. 11 656–11 665. 13

[157] Y. Zheng, Y. Yang, K. Mo, J. Li, T. Yu, Y. Liu, K. Liu, and L. J. Guibas, "Gimo: Gaze-informed human motion prediction in context," *arXiv preprint arXiv:2204.09443*, 2022. 13

[158] D. Wang, X. Cui, X. Chen, Z. Zou, T. Shi, S. Salcudean, Z. J. Wang, and R. Ward, "Multi-view 3d reconstruction with transformer," *arXiv preprint arXiv:2103.12957*, 2021. 13

[159] M. Zanfir, A. Zanfir, E. G. Bazavan, W. T. Freeman, R. Sukthankar, and C. Sminchisescu, "Thundr: Transformer-based 3d human reconstruction with markers," *arXiv preprint arXiv:2106.09336*, 2021. 13

[160] J. Mahmud and J.-M. Frahm, "Vpfusion: Joint 3d volume and pixel-aligned feature fusion for single and multi-view 3d reconstruction," *arXiv preprint arXiv:2203.07553*, 2022. 13

[161] K. Fu, S. Liu, X. Luo, and M. Wang, "Robust point cloud registration framework based on deep graph matching," in *CVPR*, 2021, pp. 8893–8902. 13

[162] Z. J. Yew and G. H. Lee, "Regtr: End-to-end point cloud correspondences with transformers," in *CVPR*, 2022, pp. 6677–6686. 13

[163] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015, pp. 1912–1920. 13, 14

[164] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015. 13, 14

[165] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese, "Joint 2d-3d-semantic data for indoor scene understanding," *arXiv preprint arXiv:1702.01105*, 2017. 13, 14, 15

[166] M. A. Uy, Q.-H. Pham, B.-S. Hua, D. T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *ICCV*, 2019, pp. 1588–1597. 13, 14

[167] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *CVPR*, 2015, pp. 567–576. 13, 15

[168] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017, pp. 5828–5839. 13, 15

[169] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012, pp. 3354–3361. 13, 14, 15

[170] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020, pp. 11 621–11 631. 13, 14, 16

[171] L. P. Tchapmi, V. Kosaraju, S. H. Rezatofighi, I. Reid, and S. Savarese, "Topnet: Structural point cloud decoder," in *CVPR*, 2019, pp. 383–392. 13, 14, 16, 17

[172] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," in *3DV*, 2018, pp. 728–737. 13, 14, 16

[173] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *TPAMI*, vol. 36, no. 7, pp. 1325–1339, 2014. 13, 14, 16

[174] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt, "Monocular 3d human pose estimation in the wild using improved cnn supervision," in *3DV*, 2017, pp. 506–516. 13, 14, 16, 17

[175] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *NeurIPS*, 2018, pp. 820–830. 14

[176] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *CVPR*, 2019, pp. 8895–8904. 14

[177] Z. Zhang, B. Sun, H. Yang, and Q. Huang, "H3dnet: 3d object detection using hybrid geometric primitives," in *ECCV*, 2020, pp. 311–329. 15

[178] Y. Chen, H. Ma, X. Li, and X. Luo, "S-votenet: Deep hough voting with spherical proposal for 3d object detection," in *ICPR*, 2021, pp. 5161–5167. 15

[179] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *CVPR*, 2019, pp. 770–779. 15

[180] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *CVPR*, 2020, pp. 10 529–10 538. 15

[181] Y. Zhang, J. Lu, and J. Zhou, "Objects are different: Flexible monocular 3d object detection," in *CVPR*, 2021, pp. 3289–3298. 15, 16

[182] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3dssd: Point-based 3d single stage object detector," in *CVPR*, 2020, pp. 11 040–11 048. 16

[183] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3d object detection," *arXiv preprint arXiv:1908.09492*, 2019. 16

[184] T. Wang, X. Zhu, J. Pang, and D. Lin, "Fcos3d: Fully convolutional one-stage monocular 3d object detection," in *CVPR*, 2021, pp. 913–922. 16

[185] X. Wen, P. Xiang, Z. Han, Y.-P. Cao, P. Wan, W. Zheng, and Y.-S. Liu, "Pmp-net: Point cloud completion by learning multi-step point moving paths," in *CVPR*, 2021, pp. 7443–7452. 16

[186] J. Wang, S. Yan, Y. Xiong, and D. Lin, "Motion guided 3d pose estimation from videos," in *ECCV*, 2020, pp. 764–780. 16

[187] T. Chen, C. Fang, X. Shen, Y. Zhu, Z. Chen, and J. Luo, "Anatomy-aware 3d human pose estimation with bone-based pose decomposition," *TCSVT*, vol. 32, no. 1, pp. 198–209, 2021. 16

[188] A. Nekrasov, J. Schult, O. Litany, B. Leibe, and F. Engelmann, "Mix3d: Out-of-context data augmentation for 3d scenes," in *3DV*, 2021, pp. 116–125. 17

[189] G. Qian, Y. Li, H. Peng, J. Mai, H. A. A. K. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," *arXiv preprint arXiv:2206.04670*, 2022. 17