ISSN 0005-1179, Automation and Remote Control, 2022, Vol. 83, No. 4, pp. 503–519. © Pleiades Publishing, Ltd., 2022.
Russian Text © The Author(s), 2022, published in Avtomatika i Telemekhanika, 2022, No. 4, pp. 5–26.

SURVEYS

3D Recognition: State of the Art and Trends

S. R. Orlova[1*] **and A. V. Lopata**[2**]

[1] *Peter the Great St. Petersburg Polytechnic University, St. Petersburg, 195251 Russia*

[2] *Central Research and Development Institute of Robotics and Technical Cybernetics, St. Petersburg, 194064 Russia*

*e-mail: *farhulina.sr@edu.spbstu.ru, **alopota@rtc.ru*

Received August 9, 2021; revised December 27, 2021; accepted December 30, 2021

Abstract—We consider the field of three-dimensional technical vision and in particular three-dimensional recognition. The problems of three-dimensional vision are singled out, and methods for obtaining and presenting three-dimensional data, as well as applications of three-dimensional vision, are reviewed. Deep learning methods in 3D recognition problems are surveyed. The main modern trends in this field are revealed. So far, quite a few neural network architectures, convolutional layers, sampling, pooling, and aggregation operations, and methods for representing and processing three-dimensional input data have been proposed. The field is under active development, with the greatest variety of methods being presented for point clouds.

Keywords: 3D recognition, deep learning, computer vision

DOI: 10.1134/S0005117922040014

1. INTRODUCTION

Three-dimensional technical or computer vision is becoming increasingly relevant: the technologies and equipment necessary for obtaining three-dimensional representations, their processing, and visualization are becoming cheaper and more accessible, and simultaneously, the interest in three-dimensional computer vision applications is growing. The emergence of virtual and augmented reality technologies, as well as the growing popularity of mobile robots and various intelligent service systems, further stimulate the development of technical vision.

The field of three-dimensional recognition is especially interesting, since it is aimed at solving the problem of understanding the surrounding world, which is one of the main problems in technical and computer vision. Despite the fact that great progress has been made in image recognition problems, three-dimensional models give a much more complete picture of the scene, the objects in it, and their relative position. It is intuitively clear that although the recognition of three-dimensional representations is a more difficult problem, it should allow obtaining better and more stable systems that act in space or analyze it and objects in it. Although it is obvious that the problem of understanding the world around the machine will not be solved in the near future, it is worth starting to form the foundation for future research by developing the field of three-dimensional recognition. In addition, there are many less global problems that modern science and technology are quite capable of solving.

2. 3D TECHNICAL VISION

Three-dimensional technical (computer) vision involves working with three-dimensional representations of existing or synthesized objects and scenes. There are several typical problems of three-dimensional technical vision:

1. *Getting 3D representations of existing objects and scenes*—3D scanning. There are many methods and equipment for this problem, and there are also neural network methods that generate three-dimensional representations.
2. *Processing, filtering, and analysis of three-dimensional representations* can have various goals: filtering noise and outliers, corrections (for example, interpolation or decimation), combining several partial representations into one, or converting to another type of representation (for example, obtaining a polygonal model from the point cloud).
3. *Mapping and localization, SLAM,* are popular in mobile robotics.
4. *Visualization of three-dimensional representations* can be both an auxiliary problem (result monitoring when working with three-dimensional representations) and the main problem (in augmented reality applications).
5. *Pattern recognition by 3D representations and semantic analysis* is a related field that involves various computer vision, machine learning, and deep learning methods. The problems in this group are aimed at obtaining information from three-dimensional representations of a higher level of abstraction: for example, recognizing a scene in a point cloud in the form of a living room with furniture and people. This is necessary to create systems that can understand the world around us and interact with it.

2.1. Representations of 3D Data

Three-dimensional data representing scenes and objects of the real world or synthesized by any method and constructed by a designer can be represented in various forms. Combining knowledge from areas such as vision and computer graphics, one can make a single list that includes all the most common forms of representation:

1. Point cloud.
2. Depth map, often considered as a 2.5-dimensional representation.
3. RGB-D image, also considered as a 2.5-dimensional representation.
4. Collection of images, strictly speaking, is not a three-dimensional representation. However, there are many methods for obtaining a point cloud from a collection of images (with or without a depth channel), including video storyboard.
5. 3D models:
   (a) Wireframe models.
   (b) Boundary (surface) models.
   (c) Solid models:
      (i) Constructive solid geometry, CSG.
      (ii) Voxel models.
   (d) Polygon mesh.
   (e) Parametric models.
   (f) Implicit surfaces.

The most popular representations are the point cloud, the polygon mesh, and the voxel model. The depth map, the RGB-D image, and the multiview image collection are not strictly 3D representations but are extremely popular (RGB-D cameras are becoming more available) and are used in photogrammetrical methods for object or scene reconstruction.

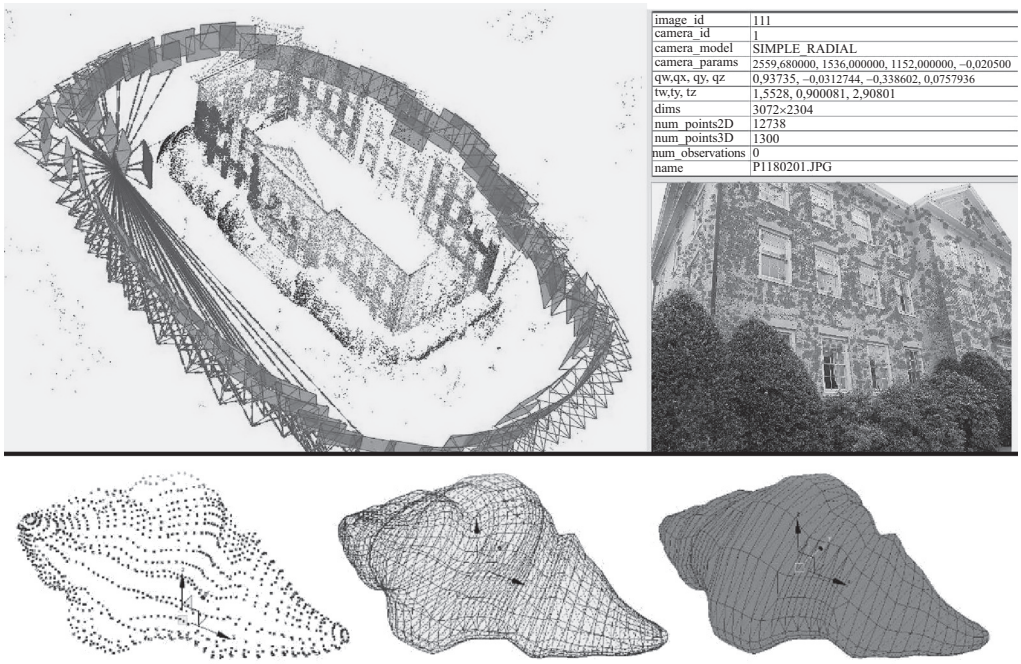

Fig. 1. Top left: point cloud constructed with COLMAP software [1] from 128 (RGB) images of one of the buildings of the University of North Carolina at Chapel Hill [2]. The calculated camera positions are shown. One position (shown in darker color) is selected. Top right: the image corresponding to the selected position and its parameters. The points (pixels) used in the construction of the cloud are marked. Bottom: a 3D model of a croissant based on a polygon mesh. The vertices are shown on the left, the edges are shown in the center, and the faces of the model are shown on the right. The image has been borrowed from [3].

A *point cloud* is a set of points in space that represents figures, objects, and surfaces. Each point is represented by three-dimensional coordinates; there can also be auxiliary attributes, for example, intensity or color. Point clouds can be obtained and stored; they permit quickly performing linear transformations in the matrix representation. Point clouds are also easy to match with each other and to map onto a plane. Most 3D scanning methods produce exactly a point cloud.

A *depth map* is a two-dimensional single-channel image containing information about the distance from the scene surfaces to the shooting point (area).

An *RGB-D image* is a two-dimensional four-channel image containing the depth map in one channel and the color information (RGB scene image) in the remaining three channels.

Voxel (VOlumetric piXEL) or *voxel grids* are based on the following approach: the entire three-dimensional space is represented in the form a uniform mesh or matrix whose cells contain voxels—three-dimensional volumetric pixels. The position of a voxel is denoted by its position in the matrix rather than the coordinates in space. A voxel may contain a set of attributes, for example, color or material. Voxel models are fairly frequently used in medicine in analysis problems.

Models based on polygon meshes, similar to boundary ones, consist of vertices, edges, and faces.

However, unlike boundary models, the faces of a polygon mesh are always flat, i.e., are polygons. At the moment, models based on a triangular polygon mesh are the most popular form of 3D data representation in the field of computer graphics and in interactive visualization problems.

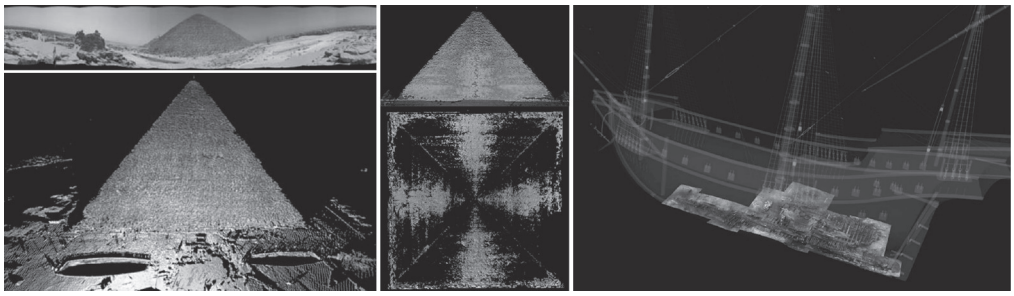

Fig. 2. Left: a scan of the east side of the Cheops Pyramid; a 2D panorama is shown above [4]. Center: a detected anomaly; the eastern side of the pyramid with color (and brightness) coding of the deviation of the slope surface from the plane is shown at the top, and the bottom shows the location of the anomalies in the top view [4]. Right: the position of the 3D model of the excavated part of the bottom of a 16th-century ship with respect to the whole ship model [5].

2.2. 3D Recognition Applications

At the moment, with the increasing availability of sensor, computer, and software tools capable of performing three-dimensional scanning and reconstruction, there is a wide field of possible applications of three-dimensional vision and in particular recognition.

In medicine, deep learning methods are used to segment 3D images and detect and evaluate the position, size, and shape of any elements—organs, tumors, etc. Here a voxel representation is often used, since many scanning methods involve obtaining a set of images when the scanner moves in one direction with a given step. Applications are possible to help assess the posture, stoop of the patient, and the nature of their motions.

In geomatics, three-dimensional recognition can be useful in the problems of marking the terrain and searching for any objects in arrays of three-dimensional data. The generation of three-dimensional objects can be in demand in the development and design of products.

Augmented reality applications often use 2D images; this is due to the need to use such applications on mobile devices with low computing resources and memory. However, in some problems it may be necessary to use three-dimensional representations, for example, augmented reality applications for medicine, providing high-quality visualization of the areas to be studied as well as segmentation and detection of any elements and allowing one to interact with a three-dimensional model. Such technologies can be used in production.

In the field of autonomous driving, deep learning methods are applied in scene analysis problems such as object detection and scene segmentation.

In the field of mobile and service robots and service systems, there can be many various applications of three-dimensional recognition, including all kinds of problems: scene segmentation, segmentation of parts of objects, detection and localization of objects and key points, classification, pose estimation, etc. In this area, both the problems and the environment can be very variable, and interaction with a person is also often assumed, so that the option of using a three-dimensional representation of a scene or objects can be most beneficial, because it provides most complete information about the scene.

3. PROBLEMS AND METHODS OF THREE-DIMENSIONAL RECOGNITION

The main problem of computer vision is the problem of understanding the world around the machine. A person, even when looking at an image of a real scene rather than the scene itself,

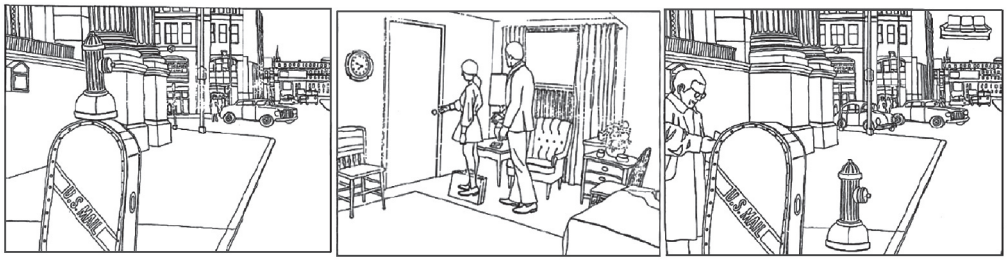

Fig. 3. Examples of Biederman constraint violations. Left: a fire hydrant violates position constraints. Center: violation of the relative position (briefcase and legs). Right: triple violation—the sofa violates the constraints of support, probability of occurrence, and size. The images have been borrowed from [7].

represents it in three-dimensional form and evaluates the position of objects relative to each other, the space available for actions, etc. At the same time, a number of studies show that people perceive a three-dimensional scene in terms of the relative location of surfaces rather than in the form of an absolute depth map [6]. It is assumed that a person builds one or more three-dimensional models of the scene and uses them for various tasks: distance estimation, object search, and route building. Understanding the context also plays a big role. Irving Biederman, a neuroscience and vision scientist, formulated five constraints that a well-organized scene must follow: support, relative positioning, probability of occurrence, position, and size. Experiments have shown that people, on average, take longer to search for an object in a scene image if it violates at least one of the Biederman constraints [7]. Thus, people recognize objects in a familiar context faster.

Our life experience, all our existing knowledge about the world, helps in recognition. We know that grass grows from soil and is found in cracks between tiles but cannot grow from a solid cement floor. We understand what a reflection in mirrors or other surfaces and paintings on the walls are. We know that usually small children do not go out without adult accompaniment.

It is not yet clear how to transfer all this experience to a machine. However, at this stage in the development of computer vision, the idea to use the most complete, i.e., three-dimensional, visual representation of the scene seems appropriate. In addition to solving some applied problems that are relevant now or in the near future, the development of three-dimensional recognition methods may be a necessary step towards solving the problem of understanding the surrounding world by a machine.

The most popular 3D recognition problems currently include:

- Highlighting key points and assessing the position and angle; finding a correspondence between three-dimensional objects.
- Classification of objects and forms; feature extraction.
- Generation of three-dimensional objects and shapes (including reconstruction of sites).
- Recognition and generation of faces and bodies of people and animals.
- Object detection;
- Segmentation of objects or component parts of objects.

However, when considering deep learning methods, it is difficult to make a clear division into tasks. In most cases, the key element for solving any problem is a feature-extractor neural network, which is the basis of the model and to which various modules and branches are added that use the features obtained to solve target problems—classification, segmentation, etc. Therefore, it is more expedient to consider the area of three-dimensional recognition without being tied to tasks.

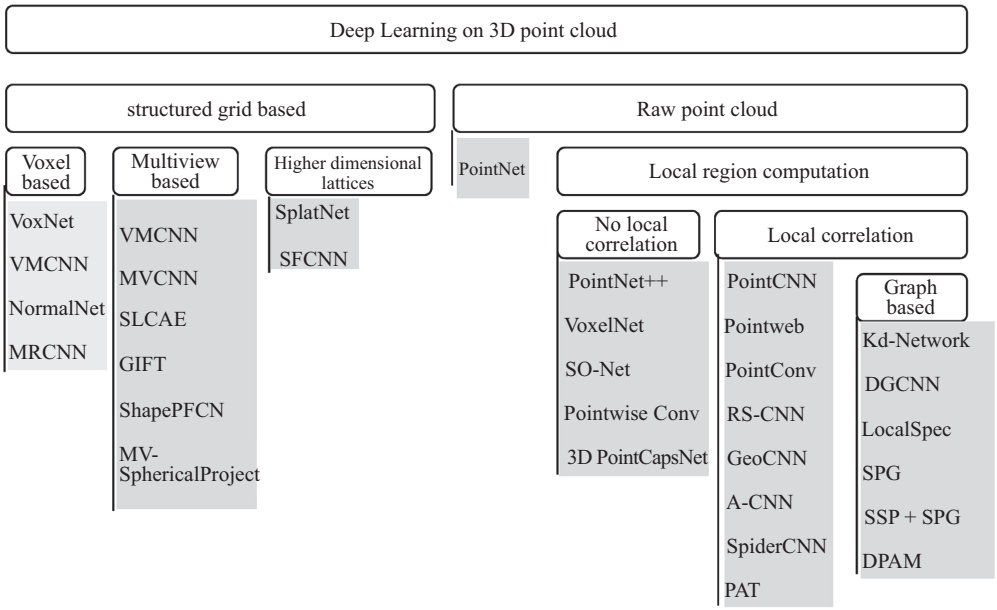

Fig. 4. In the publication [8], methods are first divided into groups according to the type of data (3D representations) they are intended for, and then the authors distinguish between narrower groups of methods based on the approaches they use.

3.1. Deep Learning Methods in 3D Recognition Problems

The publication [8] covers deep learning methods for point cloud recognition from 2015 to 2020. The authors identified several approaches and provided examples of popular algorithms (Fig. 4).

The authors distinguish three properties inherent in point clouds: irregularity (uneven distribution of point densities in the volume of the cloud), being unstructured (the absence of any grid on which the points lie), and disorder (points are stored in a list, and the order of their location in this list is irrelevant). A few years ago, these properties did not allow using deep learning methods and especially convolutional neural networks designed for ordered, regular, and structured data with point clouds, and so in early research point clouds were converted to regular and structured representations. The following approaches are popular: voxelization, a set of views from different angles, and a higher-dimensional grid.

Voxelization involves converting a point cloud into a voxel grid of a fixed size. Here one can build a neural network using 3D convolutional filters. A large number of publications [9–14] use this approach and perform voxelization at the preprocessing stage. 3D ShapeNets [14] forms a voxel grid of $30 \times 30 \times 30$ voxels; a voxel has a value of 1 if it is part of the object and 0 if it is outside the object. Thus, the object can be represented by a binary tensor. However, the problem with voxel representations is sparseness—when an object occupies only a small part of the voxel grid, most of the calculations are performed on empty areas, which is extremely inefficient. More recent publications [15, 16] suggest a solution of this problem by using octrees, a tree data structure often used to partition 3D space, in which each cube voxel can contain exactly eight smaller voxel children. Thus, areas requiring more detailed processing can be placed in smaller voxel cells.

The use of a set of images obtained from different angles instead of a point cloud has a number of advantages. This approach allows one to use all the techniques used to recognize two-dimensional images, and the images themselves do not have artifacts (the voxel grid resolution is usually quite

coarse, which leads to a distortion of the original object shape). The first publication in this direction is MultiviewCNN [17], which suggested obtaining a feature description for each image of an object using a convolutional network, then aggregating all feature descriptions with a pooling operation, and applying another convolutional neural network to obtain the final feature vector for classification. A number of papers continued the direction in [11, 18–23], using all the achievements of deep learning in image processing, but at the moment the direction is losing relevance owing to the development of methods that can work with three-dimensional representations directly. Nevertheless, the paper [24] was presented in 2020, proposing a segmentation algorithm for polygon meshes based on a multiview approach for the semantic segmentation problem. It is noteworthy that the authors propose to use not original frames obtained by photographing an object or scene from different angles but synthetic images with an increased field of view and renderings obtained from the three-dimensional scene constructed. The authors claim that their algorithm, taking approximately 12 synthetic images as input, achieves segmentation accuracy higher than other existing multiview methods when using more than a thousand images. In this case, the virtual angles used in obtaining such images may not be possible, for example, because of the walls when hiding these walls.

The idea behind the approach with a higher-dimensional grid is to transform the point cloud to a permutation-resistant structured representation such as a permutation polyhedron. The neural network SPLATNet (Sparse Lattice Networks for Point Cloud Processing) [25] takes a point cloud as input and predicts the class for each point, i.e., performs semantic segmentation. The network is constructed on the basis of BCLs (bilateral convolutional layers). The BLC principle is to transform the input features corresponding to points to a permutation polyhedron and then perform a convolution operation with a kernel that is also a permutation polyhedron. The SFCNN method [26] adaptively projects the input point cloud onto a discretized sphere, after which it applies a series of convolution operations to extract local and global features, which are then aggregated to obtain the final feature representation. The network can be used for classification and segmentation.

The first neural network PointNet [27] for point cloud recognition was presented in 2017. It works with point clouds without any preprocessing. The authors identify the following three most important approaches implemented in the PointNet architecture:

1. A symmetric function for an unordered input that provides invariance to permutations in the input point cloud. The function is implemented as a multilayer perceptron with a subsampling layer by the maximum value (max pooling).
2. Aggregation of local and global information. In the segmentation branch, the authors combine the output vector of global features with a set of feature vectors for each cloud point and obtain an output membership vector for each point using a set of multilayer perceptrons, which is semantic segmentation. This approach allowed the authors to successfully apply PointNet to the tasks of segmenting objects in a scene and segmenting parts of objects.
3. An alignment network providing invariance for cloud geometric transformations by canonicalizing the point cloud before feature extraction. The authors used a small neural network T-Net, which predicts the affine transformation required to bring the point cloud to the standard form. The T-Net network is built into the PointNet architecture, and so the predicted transformation is immediately applied to the point cloud. The same approach was used to reduce not only the point cloud but also features obtained from different point clouds to the canonical form.

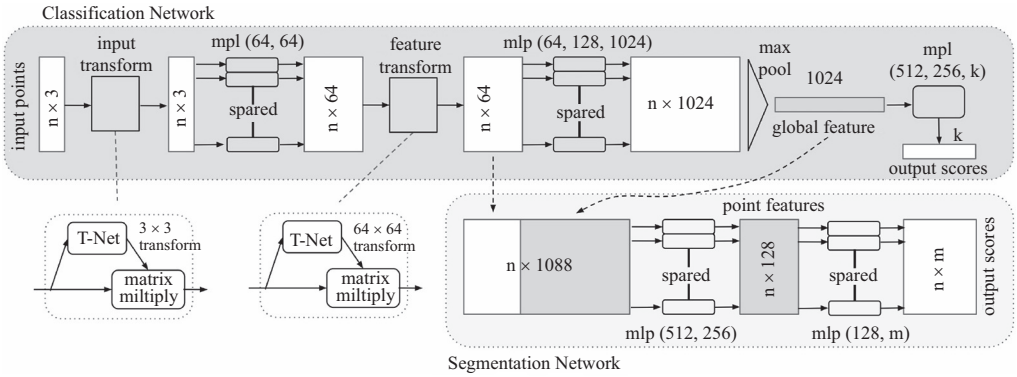

Fig. 5. PointNet Neural Network Architecture [27]. The input is n points represented by three coordinates. The top row shows the main classifier network, and the bottom row shows an additional branch for semantic segmentation. The designation *mlp* implies a multilayer perceptron; the sizes of the layers are indicated in parentheses.

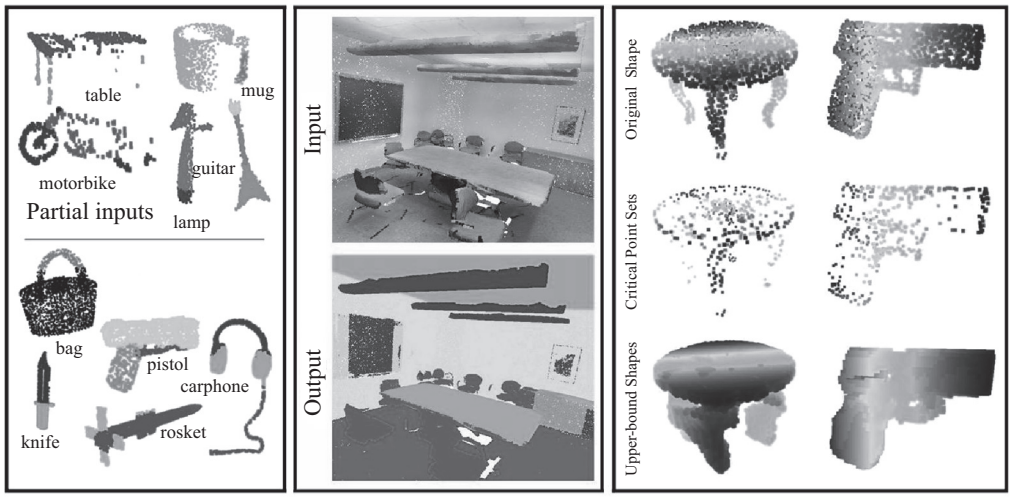

Fig. 6. PointNet results. Left: examples of classification of whole objects and their parts. Center: an example of semantic scene segmentation. Right: for each feature shape (top row), PointNet defines a keypoint set (middle row) such that any point cloud between the keypoint set and the limit shape (bottom row) has the same global feature vector.

However, the PointNet architecture did not include the ability to find local structures and take into account local dependencies between points. Further research was aimed at solving this problem.

The PointNet++ method [28] groups local regions and applies PointNet on such groups, repeating this procedure in a hierarchical manner. The VoxelNet neural object-detection network [29] takes a point cloud as input and converts it into a voxel mesh; i.e., points are grouped by voxels. For each such voxel group, a feature representation is calculated, which is then fed to the input of the region suggestion network to obtain detections. In SO-Net [30], to reduce the dimension of the initial point cloud, a Kohonen self-organizing map is built with a given number of nodes that generates nodal centroid points. Then, for each such centroid, a given number of neighboring points is selected that are fed to the input of an encoder network, the output of which is a global feature vector for the point cloud. The publication [31] suggests pointwise convolution. For each cloud point, neighboring points are determined that fall into a three-dimensional convolution window, which, like the surrounding pixels, undergo a convolution operation along with the considered point. This is how a fully convolutional neural network is constructed that has a feature vector for

each point as output. The paper [32] presents 3DPointCapsNet—a 3D autoencoder that takes into account the geometric relationships between parts of objects; this improves the network learning and generalization capabilities. The latent space built using the capsule dynamic routing [33] is parameterized by the so-called latent capsules—a set of features of a given size.

Some of the works not only group points but also investigate the correlation between points in local groups. The intuitive assumption is that the points cannot exist in isolation and their clusters display some shape that can be analyzed. In [34], the authors note that the convolution operation performed directly on a group of points does not take into account the relative position of the points and the forms of their crowding, but however, it depends on the order of the points, which is of no importance in point clouds. As a solution of this problem, it is proposed to use the χ-transformation over the coordinates of the points, which is implemented as a multilayer perceptron and allows simultaneously weighing and performing a permutation of the input points. Combining the χ-transform followed by the convolution operation results in χ-Conv, the basic building block of the proposed PointCNN neural network. The paper [35] presents the Pointweb neural network. Here a new learning block AFA (Adaptive Feature Adjustment) is proposed, which receives all possible pairs of points in a local group and calculates the influence of points on each other to form feature vectors that aggregate the properties of the entire group of points.

A large number of papers have been proposed that explore various options for convolutional operations taking into account the relative positions of points and the shapes they form. The paper [36] proposes a version of the point cloud convolution operation—PointConv. Unlike conventional convolutions with discrete kernels (weights), the PointConv offers a trainable continuous function that allows one to get the weight for a point based on its position. In addition, it is supposed to weight points based on estimates of their reciprocal densities: the contribution of individual points that make up dense crowding will thus not be too large. In the RS-CNN [37] and Geo-CNN [38] neural networks, uniformly distributed points are sampled from a cloud, each of the points becoming the centroid of a spherical neighborhood area; however, not the neighborhood points themselves are considered, but their positions relative to the centroid (represented as vectors). A feature of the GeoConv convolution operation, from which the Geo-CNN is constructed, is the projection of these vectors onto the axes, i.e., formation of an orthogonal basis. The features are calculated for each direction, after which they are aggregated taking into account the angles between the vector and the axes into a single final feature vector of the point of a local domain. Then the features of the points of the local domain are aggregated with weights depending on the position of the points relative to the centroid and the radius of the domain. The radius of local domains grows along the network. The paper [39] offers ring convolutions that have a spherical receptive domain with a concentric spherical cavity inside (they take the form of rings when projected onto a plane). The authors note that in modern neural network architectures, multiscale features are usually combined for resilience to scale changes; this leads to overlapping of receptive fields and multiple occurrence of features of some points in the calculation of the final feature vector. A ring convolution can consist of several concentric rings; thus, it is possible to obtain features of different scales without reusing the same points. The SpiderCNN neural network [40] for classification and segmentation is built based on SpiderConv convolutional layers. SpiderConv uses a parameterized filter family to perform the convolution operation. The filters are a composition of two functions—a simple step function to capture coarse geometry and a third-order Taylor expansion to capture more complex structures.

The paper [41] introduces DPRNet—a neural network for the classification and segmentation using sparse convolutions and residual joins and uses the SELU (Scaled Exponential Linear Units) self-normalizing activation function instead of batch normalization.

The paper [43] proposes PATs (Point Attention Transformers)—a transformer neural network designed to work with point clouds. The key PAT operations are Group Shuffle Attention (GSA) and Gumbel Subset Sampling (GSS). GSA is a parametrically efficient self-attention operation for training understanding of relationships between points. GSS serves as an alternative to the Farthest Point Sampling (FPS) operation and has several advantages such as permutation invariance, differentiability, and better resistance to outliers. PAT implies the possibility of classification and segmentation. In both cases, the resulting vector or set of class membership vectors is computed using multilayer perceptrons. The paper [44] also offers sampling operations with trainable parameters. The authors note that despite the high-quality results obtained when using hierarchical architectures, sampling and grouping of points in such methods is performed in the original Euclidean space in a fixed way; this can lead to a decrease in robustness. To solve this problem, the paper proposes a DPAM (Dynamic Points Agglomeration Module) unit. The unit is built on the basis of a set of multilayer perceptrons and forms a trainable agglomeration matrix, which, when multiplied by a cloud of points or a set of features, reduces their dimension; i.e., it is a combination of grouping and subselection (pooling) operations.

The PointHop method for classifying a point cloud is proposed in [45]. It has low computational complexity and is trained without a teacher. The method is a cascade of modules of the same name. Features from all modules are aggregated into a single vector for support vector and random forest classification. The PointHop module forms a local domain for each cloud point from k nearest neighbors, then splits the domain into eight octant regions, calculates features, and concatenates them using the Saab transformation (a variation of the principal component method) [46]. The paper was continued in the PointHop++ method [47]. The authors optimized the model by reducing the number of its parameters, bringing it to a tree form, and implementing the selection of distinguishing features according to the entropy criterion. Finally, in 2021, the R-PointHop model [48] was introduced to find a rigid transformation (rotation and offset) between two point clouds that provides an optimal fit. All methods of the PointHop group are not deep learning methods and use unsupervised learning, which makes them suitable for mobile and low-power computing platforms. Another method using unsupervised learning is the one presented in [49]. The proposed neural network extracts key (structural) points from the cloud and allows solving the problems of matching the shapes of point clouds and transferring segmentation markup and potentially the problems of reconstructing and supplementing the shape.

The graph-based approach treats a point cloud as a graph whose nodes are points. The edges of the graph represent the correlation between points. The 2017 paper [50] describes a Kd-network, i.e., a neural network representing a point cloud in the form of a k-d-tree (k-dimensional tree). The leaves of the tree are the points of the cloud (their coordinates), and the vector representation of each nonleaf node is calculated as a nonlinear function of an affine transformation with trainable parameters applied to two neighboring nodes of the considered nonleaf node. After processing the root node, a feature vector of the entire point cloud is obtained. The paper [51] presents a module EdgeConv that generates a feature vector for a point given its neighborhood and is a directed graph. The module computes pairwise edges, called edge features, between the centroid point and its neighbors using a fully connected layer. The output of the EdgeConv module is

obtained by applying the pooling operation to the edge features. A peculiarity of this approach is that the edge features depend not only on the relative position of the points but also on the edge features obtained from the previous EdgeConv layer. Thus, the graphs used are dynamic. The paper [52] proposes spectral convolutions, its own subsampling (pooling) operation, and a neural network architecture for classification and segmentation based on them. In the spectral convolution module, the neighborhood points are the nodes of the graph, and the lengths of the edges are calculated as the distances between the points in the feature space and form an adjacency matrix based on which the spectrum of the graph is calculated. Next, the Fourier transform for the graph, spectral filtering, and finally, the inverse Fourier transform are performed. The recursive cluster pooling module iteratively performs spectral clustering and pooling in such a way that each cluster participates in the formation of the final set of feature vectors. The authors suggest that, unlike the popular maximum pooling, this method will better preserve useful information. The paper [53] proposes the semantic segmentation model Point2Node. The key feature of this model is the ability to use the correlation not only of neighboring points but also of any model points. The graph is constructed from multidimensional nodes obtained from the input points of the cloud using the χ-Conv convolutions proposed in [34]. Further, the DNC (Dynamic Node Correlation) module represents each node in three versions—taking into account autocorrelation (between the channels of the node's feature vector), local correlation (between the nodes of a local group), and nonlocal correlation (between mutually distant nodes). An adaptive learning module for aggregation of multilevel features is used. The paper [54] proposes to divide a point cloud into so-called superpoints. Superpoints are simple semantically homogeneous geometric shapes. For each superpoint, a feature vector is generated using the PointNet model [27], and the resulting vectors form a Superpoint Graph (SPG). This approach permits significantly reducing the size of the input point cloud while preserving important features and can be used on very large point clouds. In the subsequent work [55], the case is considered in which a geometrically homogeneous domain is not semantically homogeneous, and it is proposed to additionally segment superpoints into smaller regions. The paper [56] proposes a new Graph Attention Convolution (GAC) operation, which dynamically determines the weight of each point in a local group. This permits focusing on more important elements of the local group and excluding elements that belong to a different semantic class. The 3D Graph Convolution Network (3D-GCN) proposed in [57] uses convolutional modules in which the kernel, which is a graph and represents a set of local group points, has not only weights that can be adjusted during training but also a shape, i.e., the position of the points themselves. The cosine distance used to determine the similarity of the input data and the convolutional kernel is independent of the length of the vectors and provides scale invariance.

In 2021, the paper [58] was presented, in which it was proposed to group points not into local groups but along curves, i.e., chains of points that form a broken line. When using local groups, a similar distribution of points in any group leads to similar features, disallowing separation into regions. Curves, on the other hand, reflect the geometric shape well and permit constructing more distinguishable features. A curve construction (curvilinear aggregation) module is trainable—each subsequent point is selected as the point from the local group that has the highest numerical score obtained using multilayer perceptrons. The CurveNet neural network using this approach permits one to classify and segment a point cloud.

In 2019, the first neural network algorithm was presented that works with triangular polygon meshes—MeshNet [59]. The input polygonal mesh was considered as a set of face elements. Each

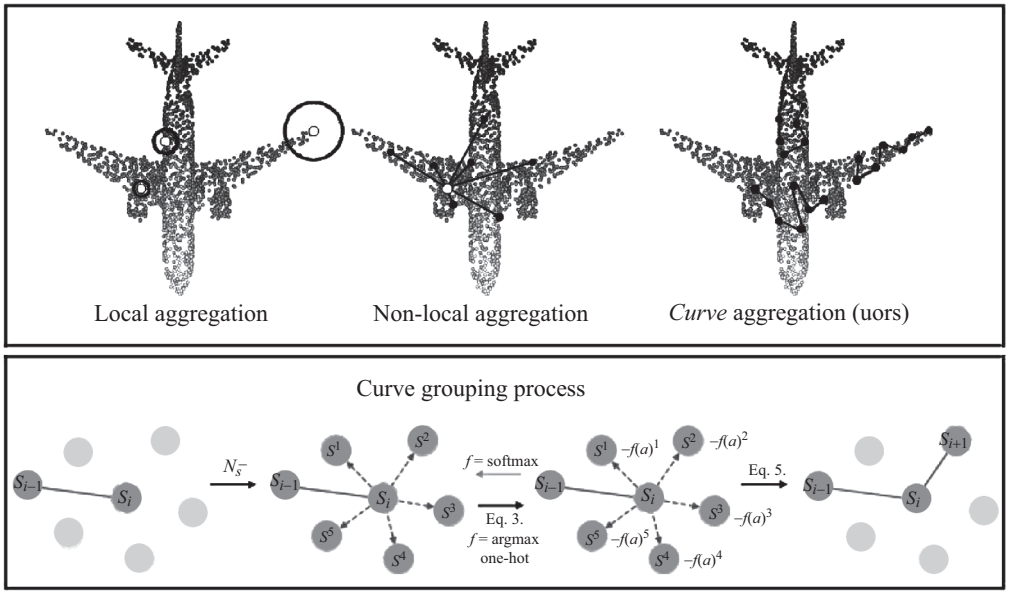

Fig. 7. Top: grouping methods—local groups, nonlocal aggregation, and curvilinear aggregation. Bottom: an illustration of the curve construction process in the paper [58].

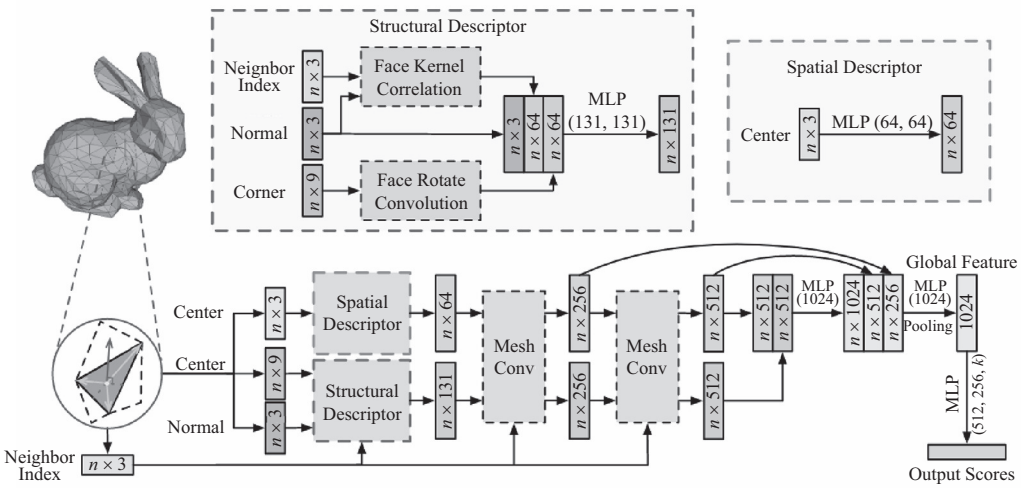

Fig. 8. MeshNet architecture [59].

face was described by the following set of parameters: face center coordinates, face vertex vectors drawn from the face center, unit length normal vector, and mating face indices. The parameters are processed in structural descriptor and spatial descriptor blocks, which receive primary structural and spatial features, respectively. The features are then passed through polygon mesh convolution blocks. Finally, by aggregating and subsampling the features, the final global feature vector is obtained, which can be used for various tasks, including classification.

In [60], it is proposed to transform the mesh into a three-dimensional grid containing curvature vectors in the cells and feed this mesh as input to the CurveNet convolutional neural network presented there (not to be confused with CurveNet in [58]). LaplacianNet [61] (Laplacian Encoding and Pooling Network) takes as input the vertices and normals of a polygon mesh as well as eigenvectors and clusters obtained by spectral clustering. The neural network consists of two MPB (Mesh Pool-

ing Block) units for analyzing data at different resolutions and generating features for each vertex, after which these features are used in a segmentation or classification network, depending on the problem in hand. The initial formation of clusters (similar to superpixels for images) as local groups and varying the size of clusters from small (detailed) to large (coarse) permits processing polygon meshes hierarchically. The MeshWalker neural network [62] accepts as input a chain-path composed randomly from the vertices of a polygon mesh with a volume of 40% of their total number. The network contains recurrent layers for aggregating information along the entire path. The advantage of the MeshWalker is its ability to produce high-quality results even with a small amount of training data. The paper [63] proposes a LRFConv (Local Reference Frame Convolution) convolutional layer with a pass-through connection. In the LRFConv, a group of neighboring points is determined by geodetic distances for each point, represented by its coordinates and normal vector, processed in a three-dimensional model. The coordinates of the neighbors relative to the centroid point are recalculated; together with the distances, they are fed to the input of multilayer perceptrons to bring them to the required dimension and are concatenated with the features of the previous layer and fed to the input of continuous convolution.

The authors of [64] note that modern approaches to the segmentation of three-dimensional shapes (objects) depend on the quality and volume of labeled training data sets and present the ROSS algorithm, which requires only one labeled example for training. The ROSS learns to transfer markup from an example to similar input data.

There is also development in the direction of generative methods. The TM-Net [65] is a network for generating textured polygon meshes and consists of three main parts: TextureVAE, PixelSNAIL, and PartVAE. Each of these parts is based on two variational autoencoders. The PartVAE generates the geometry of the object, the TextureVAE generates the texture, and the PixelSNAIL ensures that the resulting texture matches the geometry generated.

CONCLUSIONS

Despite the significant development of deep learning methods, including those for two-dimensional images, their application in the field of three-dimensional recognition has been difficult until recently. Point clouds, which are quite easily obtained from various sensors or a set of images of an object or scene, are used in many vision systems; however, their irregularity and unstructured and disordered nature make it difficult to develop neural network solutions. Early methods involved converting point clouds into regular and structured representations such as voxel grids, higher-dimensional meshes, and a set of images from various angles.

In 2017, PointNet was introduced—the first neural network that works directly with point clouds and solves the problems of classifying and segmenting a point cloud. However, the neural network did not take into account local dependencies between points, and further research in the area was aimed at solving this problem. At the moment, there are a large number of approaches and techniques that have been successfully applied in the field of three-dimensional recognition of point clouds: considering a point cloud as a graph, superpoints, convolutional kernels with customizable weights and shape, using a local orthogonal basis, trainable continuous functions as convolutional cores, etc.

With the development of deep learning methods for point clouds, methods using voxel representations have become less popular. The reason for this is high requirements for computing resources—due to the sparseness of the voxel grid, most of the operations occur in empty areas, as

a result of which the resolution of the voxel grid is very limited and does not allow high-quality representation of large scenes or objects with many small details. However, voxel representations remain popular in the field of medical problems related to the recognition of radiological images owing to the nature of scanning.

Methods that use sets of images taken at different angles for three-dimensional recognition (usually segmentation) are also losing popularity and, as a rule, lose in accuracy to methods that use other types of representations. Nevertheless, the recent paper [24] showed that this direction has prospects; however, the algorithm proposed assumed the use of renders made according to the three-dimensional model constructed rather than original images.

Neural network algorithms working with polygonal meshes were first introduced in 2019. The advantage of meshes is the presence of information about the surface and the connectivity of elements. This direction allows one both to use advances in point cloud recognition, considering the mesh as a set of mesh vertices with additional information (normal vectors, mating vertices, etc.) and to apply new approaches (to form a mesh of curvature vectors or a chain-path along the mesh as input data, use geodetic distances, etc.) At the moment, it was not possible to notice any established practices in this direction.

Thus, the state of the art in the field of three-dimensional recognition allows one to work with any popular three-dimensional representations—point cloud, voxel grid, and polygon mesh—as well as with a set of multiview images; however, the greatest progress and activity can be traced in studies related to point clouds.

FUNDING

This work was supported by the Russian Foundation for Basic Research, project no. 20-37-90039.

REFERENCES

1. COLMAP Project Page on Github.io—Main Page. https://colmap.github.io. Cited August 3, 2021.
2. COLMAP Project Page on Github.io—Datasets. https://colmap.github.io/datasets.html. Cited August 3, 2021.
3. Pérez, P. and Iván, R., Blurring the boundaries between real and artificial in architecture and urban design through the use artificial intelligence, *PhD Thesis,* Univ. Coruña, 2017.
4. Neubauer, W., Doneus, M., Studnicka, N., and Riegl, J., Combined high resolution laser scanning and photogrammetrical documentation of the pyramids at Giza, *CIPA XX Int. Symp.* (Citeseer, 2005), pp. 470–475.
5. McCarthy, J.K., Benjamin, J., Winton, T., and van Duivenvoorde, W., *3D Recording and Interpretation for Maritime Archaeology*, Springer Nature, 2019.
6. Hoiem, D. and Savarese, S., Representations and techniques for 3D object recognition and scene interpretation, *Synth. Lect. Artif. Intell. Mach. Learn.,* 2011, vol. 5, no. 5, pp. 1–169.
7. Biederman, I., On the semantics of a glance at a scene, *Perceptual Organ.*, 1981, vol. 213, p. 253.
8. Bello, S.A., Yu, S., Wang, C., Adam, J.M., and Li, J., Deep learning on 3D point clouds, *Remote Sensing*, 2020, vol. 12, no. 11, p. 1729.
9. Maturana, D. and Scherer, S., 3D convolutional neural networks for landing zone detection from lidar, *IEEE ICRA. IEEE,* 2015, pp. 3471–3478.
10. Maturana, D. and Scherer, S., Voxnet: a 3D convolutional neural network for real-time object recognition, *IEEE/RSJ IROS. IEEE,* 2015, pp. 922–928.

11. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., and Guibas, L.J., Volumetric and multi-view CNNs for object classification on 3D data, *Proc. CVPR,* 2016, pp. 5648–5656.

12. Wang, C., Cheng, M., Sohel, F., Bennamoun, M., and Li, J., NormalNet: a voxel-based CNN for 3D object classification and retrieval, *Neurocomputing*, 2019, vol. 323, pp. 139–147.

13. Ghadai, S., Lee, X., Balu, A., Sarkar, S., and Krishnamurthy, A., Multi-resolution 3D convolutional neural networks for object recognition. 2018. arXiv:1805.12254.

14. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J., 3D ShapeNets: a deep representation for volumetric shapes, *Proc. CVPR,* 2015, pp. 1912–1920.

15. Riegler, G., Osman Ulusoy, A., and Geiger, A., Octnet: learning deep 3D representations at high resolutions, *Proc. CVPR,* 2017, pp. 3577–3586.

16. Tatarchenko, M., Dosovitskiy, A., and Brox, T., Octree generating networks: efficient convolutional architectures for high-resolution 3D outputs, *Proc. IEEE Int. Conf. Comput. Vision,* 2017, pp. 2088–2096.

17. Su, Hang., Maji, S., Kalogerakis, E., and Learned-Miller, E., Multi-view convolutional neural networks for 3D shape recognition, *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 945–953.

18. Leng, B., Guo, S., Zhang, X., and Xiong, Z., 3D object retrieval with stacked local convolutional autoencoder, *Signal Process.*, 2015, vol. 112, pp. 119–128.

19. Bai, S., Bai, X., Zhou, Z., Zhang, Z., and Jan Latecki, L., Gift: a real-time and scalable 3D shape search engine, *Proc. CVPR*, 2016, pp. 5023–5032.

20. Kalogerakis, E., Averkiou, M., Maji, S., and Chaudhuri, S., 3D shape segmentation with projective convolutional networks, *Proc. CVPR*, 2017, pp. 3779–3788.

21. Cao, Z., Huang, Q., and Karthik, R., 3D object classification via spherical projections, *2017 Int. Conf. 3D Vision (3DV), IEEE*, 2017, pp. 566–574.

22. Zhang, L., Sun, J., and Zheng, Q., 3D point cloud recognition based on a multi-view convolutional neural network, *Sensors*, 2018, vol. 18, no. 11, p. 3681.

23. Kanezaki, A., Matsushita, Y., and Nishida, Y., RotationNet: joint object categorization and pose estimation using multiviews from unsupervised viewpoints, *Proc. CVPR*, 2018, pp. 5010–5019.

24. Kundu, A., Yin, X., Fathi, A., Ross, D., Brewington, B., Funkhouser, T., and Pantofaru, C., Virtual multi-view fusion for 3D semantic segmentation, *Eur. Conf. Comput. Vision (ECCV),* Springer, 2020, pp. 518–535.

25. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.-H., and Kautz, J., Splatnet: sparse lattice networks for point cloud processing, *Proc. CVPR*, 2018, pp. 2530–2539.

26. Rao, Y., Lu, J., and Zhou, J., Spherical fractal convolutional neural networks for point cloud recognition, *Proc. CVPR*, 2019, pp. 452–460.

27. Qi, C.R., Su, H., Mo, K., and Guibas, L.J., Pointnet: deep learning on point sets for 3D classification and segmentation, *Proc. CVPR*, 2017, pp. 652–660.

28. Qi, C.R., Yi, L., Su, H., and Guibas, L.J., Pointnet++: deep hierarchical feature learning on point sets in a metric space, 2017. arXiv:1706.02413.

29. Zhou, Y. and Tuzel, O., Voxelnet: end-to-end learning for point cloud based 3D object detection, *Proc. CVPR,* 2018, pp. 4490–4499.

30. Li, J., Chen, B.M., and Lee, G.H., So-Net: self-organizing network for point cloud analysis, *Proc. CVPR,* 2018, pp. 9397–9406.

31. Hua, B.-S., Tran, M.-K., and Yeung, S.-K., Pointwise convolutional neural networks, *Proc. CVPR,* 2018, pp. 984–993.

32. Zhao, Y., Birdal, T., Deng, H., and Tombari, F., 3D point capsule networks, *Proc. CVPR,* 2019, pp. 1009–1018.

33. Sabour, S., Frosst, N., and Hinton, G.E., Dynamic routing between capsules, 2017. arXiv:1710.09829.

34. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., and Chen, B., PointCNN: Convolution on χ-transformed points, 2018. arXiv:1801.07791.

35. Zhao, H., Jiang, L., Fu, C.-W., and Jia, J., Pointweb: enhancing local neighborhood features for point cloud processing, *Proc. CVPR,* 2019, pp. 5565–5573.

36. Wu, W., Qi, Z., and Fuxin, L., PointConv: deep convolutional networks on 3D point clouds, *Proc. CVPR,* 2019, pp. 9621–9630.

37. Liu, Y., Fan, B., Xiang, S., and Pan, C., Relation-shape convolutional neural network for point cloud analysis, *Proc. CVPR,* 2019, pp. 8895–8904.

38. Lan, S., Yu, R., Yu, G., and Davis, L.S., Modeling local geometric structure of 3D point clouds using Geo-CNN, *Proc. CVPR,* 2019, pp. 998–1008.

39. Komarichev, A., Zhong, Z., and Hua, J., A-CNN: annularly convolutional neural networks on point clouds, *Proc. CVPR,* 2019, pp. 7421–7430.

40. Xu, Y., Fan, T., Xu, M., Zeng, L., and Qiao, Y., Spidercnn: deep learning on point sets with parameterized convolutional filters, *Proc. ECCV,* 2018, pp. 87–102.

41. Arshad, S., Shahzad, M., Riaz, Q., and Fraz, M.M., DPRNet: deep 3D point based residual network for semantic segmentation and classification of 3D point clouds, *IEEE Access*, 2019, vol. 7, pp. 68892–68904.

42. Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S., Self-normalizing neural networks, 2017. arXiv:1706.02515.

43. Yang, J., Zhang, Q., Ni, B., Li, L., Liu, J., Zhou, M., and Tian, Q., Modeling point clouds with self-attention and Gumbel subset sampling, *Proc. CVPR*, 2019, pp. 3323–3332.

44. Liu, J., Ni B., Li, C., Yang, J., and Tian, Q., Dynamic points agglomeration for hierarchical point sets learning, *Proc. CVPR,* 2019. pp. 7546–7555.

45. Zhang, M., You, H., Kadam, P., Liu, S., and Kuo, C.-C.J., Pointhop: an explainable machine learning method for point cloud classification, *IEEE Trans. Multimedia*, 2020, vol. 22, no. 7, pp. 1744–1755.

46. Kuo, C.-C.J., Zhang, M., Li, S., Duan, J., and Chen, Y., Interpretable convolutional neural networks via feedforward design, *J. Visual Commun. Image Representation*, 2019, vol. 60, pp. 346–359.

47. Zhang, M., Wang, Y., Kadam, P., Liu, S., and Kuo, C.-C.J., Pointhop++: a lightweight learning model on point sets for 3D classification, *IEEE Int. Conf. Image Process. (ICIP)*, 2020, pp. 3319–3323.

48. Kadam, P., Zhang, M., Liu, S., and Kuo, C.-C.J., R-PointHop: a green, accurate and unsupervised point cloud registration method, 2021. arXiv:2103.08129.

49. Chen, N., Liu, L., Cui, Z., Chen, R., Ceylan, D., Tu, C., and Wang, W., Unsupervised learning of intrinsic structural representation points, *Proc. CVPR*, 2020, pp. 9121–9130.

50. Klokov, R. and Lempitsky, V., Escape from cells: deep Kd-networks for the recognition of 3D point cloud models, *Proc. IEEE Int. Conf. Comput. Vision,* 2017, pp. 863–872.

51. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., and Solomon, J.M., Dynamic graph CNN for learning on point clouds, *ACM Trans. Graphics (TOG)*, 2019, vol. 38, no. 5, pp. 1–12.

52. Wang, C., Samari, B., and Siddiqi, K., Local spectral graph convolution for point set feature learning, *Proc. ECCV,* 2018, pp. 52–66

53. Han, W., Wen, C., Wang, C., Li, X., and Li, Q., Point2Node: correlation learning of dynamic-node for point cloud feature modeling, *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, pp. 10925–10932.

54. Landrieu, L. and Simonovsky, M., Large-scale point cloud semantic segmentation with superpoint graphs, *Proc. CVPR,* 2018, pp. 4558–4567.

55. Landrieu, L. and Boussaha, M., Point cloud oversegmentation with graph-structured deep metric learning, *Proc. CVPR,* 2019, pp. 7440–7449.

56. Wang, L., Huang, Y., Hou, Y., Zhang, S., and Shan, J., Graph attention convolution for point cloud semantic segmentation, *Proc. CVPR,* 2019, pp. 10296–10305.

57. Lin, Z.-H., Huang, S.-Y., and Wang, Y.-C.F., Convolution in the cloud: learning deformable kernels in 3D graph convolution networks for point cloud analysis, *Proc. CVPR,* 2020, pp. 1800–1809.

58. Xiang, T., Zhang, C., Song, Y., Yu, J., and Cai, W., Walk in the cloud: learning curves for point clouds shape analysis, 2021. arXiv:2105.01288.

59. Feng, Y., Feng, Y., You, H., Zhao, X., and Gao, Y., MeshNet: mesh neural network for 3D shape representation, *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 8279–8286.

60. Muzahid, A., Wan, W., Sohel, F., Wu, L., and Hou, L., CurveNet: curvature-based multitask learning deep networks for 3D object recognition, *IEEE/CAA J. Autom. Sin.*, 2020, vol. 8, no. 6, pp. 1177–1187.

61. Qiao, Y.-L., Gao, L., Rosin, P., Lai, Y.-K., and Chen, X., Learning on 3D meshes with Laplacian encoding and pooling, *IEEE Trans. Visualization Comput. Graphics.*, 2020.

62. Lahav, A. and Tal, A., MeshWalker: deep mesh understanding by random walks, *ACM Trans. Graphics (TOG)*, 2020, vol. 39, no. 6, pp. 1–13.

63. Yang, Z., Litany, O., Birdal, T., Sridhar, S., and Guibas, L., Continuous geodesic convolutions for learning on 3D shapes, *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vision*, 2021, pp. 134–144.

64. Yuan, S. and Fang, Y., Ross: robust learning of one-shot 3D shape segmentation, *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vision*, 2020, pp. 1961–1969.

65. Gao, L., Wu, T., Yuan, Y.-J., Lin, M.-X., Lai, Y.-K., and Zhang, H., TM-Net: deep generative networks for textured meshes, 2020. arXiv:2010.06217.

This paper was recommended for publication by D.V. Vinogradov, a member of the Editorial Board