 DOI: 10.3217/978-3-85125-603-1-10

Bringing classical Robot Vision descriptors to Deep Learning*

Jean-Baptiste Weibel, Timothy Patten, Michael Zillich and Markus Vincze

Abstract— **Robot vision still relies heavily on classical hand-crafted features because of their demonstrated robustness. Recent advances in Deep Learning have been drastically outperforming such classical approaches on images, however, transferring this success to 2.5D data in a robust manner is still an open question, both because of the challenges introduced by an additional dimension and the lack of non-artificial large 3D dataset. In this work, we demonstrate the benefits of using the PointNet[8] architecture to improve upon any histogram-based descriptor. In particular, we introduce a network using a global shape descriptor and a local descriptor that is directly applicable to a point cloud, and that keeps the robustness of the original descriptors (rotation invariance, robustness to variable point density and occlusion). We obtain 83 % accuracy on the ModelNet 40 dataset, using 10-100x less parameters than some competing methods.**

I. INTRODUCTION

Since the emergence of cheap and reasonably precise 2.5D sensors, such as the Kinect, 3D object classification has been intensively studied. It is an essential task for any indoor robot. Many descriptors have been created for this purpose specifically for such sensor. They all use point clouds, which became the de facto data representation for robotic tasks.

The state-of-the-art deep learning methods used in general computer vision have not been widely adopted by the field, partly because they do not mix well with the unstructured representation that is a point cloud, but also because large datasets acquired with such sensor are still quite rare, which limits the direct applicability of such methods.

This problem can be avoided altogether by using artificial computer generated models to train a neural network, but compared to 3D artificial models, data acquired by robots tends to be noisier in a few characteristic ways:

- unaligned objects
- occlusion
- outliers points
- sensor noise

It is impossible to make any assumption about the pose of objects in the scene. As such, the only two options are to use rotation-invariant features, a path chosen by most classical descriptors, or to align the data and compute features on aligned models. If a large body of work exist regarding the computation of robust local reference frame, aligning full objects with noisy data is a challenging problem, and even though they can deal with sensor noise to some extent, most of these methods are quite sensitive to outliers or occlusion. Occlusions can be caused either by other objects in front of our object of interest, or by viewpoint being unaccessible to the robot. Outliers, on the other hand, are often remnants from a previous segmentation step. Finally, sensor noise is inevitable, and tends to increase with the distance to the sensor. Robustness to variable point density and variable noise is therefore important.

Most deep learning architectures developed so far do not consider all type of noise. In particular, most of the recent and best performing works are assuming aligned models, which is, as discussed before, non-trivial to obtain in a robotic context. These noise sources have all been studied with more classical approaches which led to robust descriptors.

We present in this paper a method to bridge the gap between those two worlds. Our proposed architecture makes use of the PointNet[8] architecture to replace the histogram with a learned probabilistic version, making it possible to learn end-to-end architectures that work on the original feature space. Any histogram-based descriptor can be improved this way, keeping the original features advantages, such as rotation invariance (no alignment necessary), robustness to occlusions and variable point density.

Our contribution is the introduction of a general method to improve the descriptiveness of any histogram-based descriptor through learning while keeping their robustness. We also showcase this method on an ESF-like global shape descriptor [18] coined L-ESF and a SHOT-like local descriptor [16] coined L-SHOT.

II. RELATED WORK

A. 3D Classification

Unsurprisingly, deep learning techniques dominate the field of 3D classification. Due to the multiplicity of data representation available when dealing with 3D data, the approaches can differ drastically. We present here a non-exhaustive list focusing on the most common and most promising architecture.

The most straightforward way to apply convolutional neural networks (CNN) to 3D data is to start by extracting 2D views from the full model. These depth maps can then be fed to any available CNN. Many mappings from single-channel to three-channel representation have also been developed[3][2], thus avoiding the issue of the lack of large dataset for training through the re-use of images-learned features. Once a representation for each view is computed, different schemes for pooling them have been developed, from view pooling [15] to more complex view-set reasoning

*This research has received funding from the European Community's Seventh Framework Programme under grant agreement no. 610532, SQUIRREL.

All authors are with the Vision4Robotics group (ACIN - TU Wien), Austria {weibel, patten, zillich,prankl, vincze}@acin.tuwien.ac.at

[17]. While performing very well in practice, thanks to the advances of 2D CNN, these approaches generally lose all information between the views, and raise the problem of view selection. In an indoor robotic context, not all viewpoints are accessible. They tend to use a lot more parameters than other methods.

It is also possible to extend the design of 2D CNN to 3D CNN, and to learn features over voxel grids. This direction has been explored in [19] but also [7]. While this direction is a meaningful extension of CNN, they bring with them two main problem inherent to the design of the network. First, the additional dimension is an optimization burden, and because of the explosion of the number of parameters, such approaches have been forced to work on relatively coarse voxel grids, making the data much less informative. Partitioning the space as in [11] is a way to tackle this issue. Second, because of the design of the convolution, they are not rotation invariant. They would thus need either to separately learn each orientation of the object, making learning harder, or align the model beforehand, which is in itself hard to perform robustly.

The last direction is to work directly on point clouds. This data representation loses any explicit neighbourhood information. One way to get around it is to rely on the creation of a KD-Tree over the set of points as in [6]. However, the KD-Tree itself depends on the orientation of the model. Moreover, a reasonably large noise can also affect the structure of the KD-Tree. Another option is to rely only on the implicit information of the coordinates, as in [8] or [10]. The architecture from [8] will be referred to as the PointNet architecture. By drastically expanding the feature dimension of the coordinates (from 3 to 1024 in multiple steps), the network can then learn up to 1024 function expressing a probability of presence in a certain area of the space. This approach also requires aligned data. The author relies on a spatial transformer network[5], to learn a data dependent alignment. Learning such an alignment over a whole object, however, is vulnerable to outliers and occlusion in two ways: the alignment itself will be affected and the representation will then be computed on a not only incomplete but also potentially misaligned set of points. The same author also demonstrated the possibility of hierarchical feature learning with the PointNet architecture[9], using a second PointNet to learn over a set of local descriptors. In [10], the author follows a similar path, except that it assumes that the models are already aligned, and improve on the layers of the network by subtracting a weighted version of the maximum activation value over the whole set for each filter.

B. Robotic classification

As in classical vision, handcrafted features were carefully developed to capture either local, regional or global information.

To capture local information, a whole family of descriptors were created following variants of the scheme introduced by the Point Feature Histogram (PFH) and Fast Point Feature Histogram (FPFH) [13]. They both rely on a set of angle between the normal of a point of interest and its neighborhood.

The best performing local handcrafted descriptor is probably the Signature of Histograms of OrienTations (SHOT) descriptor [16]. It first aligns the neighborhood along a local reference frame. This local reference frame is computed as a repeatable representation of the statistical distribution of points. Once aligned, the sphere around the point of interest is divided in 32 spatial bins, and a histogram is computed over each one of them.

The idea behind PFH/FPFH was extended to capture viewpoint specific global information, by considering angles with the vector going from the viewpoint to the centroid of the considered object. This approach was coined Viewpoint Feature Histogram (VFH)[12]. On the other hand, the Ensemble of Shape Functions (ESF) [18] relies on simple randomly sampled geometrical construct, such as pairs and triangles, instead of relying on the direct neighborhood of a point of interest.

Through their use of randomization, histogram as a pooling strategy, or other methods, all these descriptor have proven their robustness to most of the type of noise encountered in data acquired by a robotic platform. However, their descriptiveness is no match for more modern deep learning approaches, and as such, it is typically challenging to reach state-of-the-art results using such descriptors.

III. Learned Descriptors for Robotic Classification

Considering both the desire for robustness and the need for good accuracy and generalization, we propose to learn descriptors using the PointNet architecture, as a histogram-like pooling solution thus providing robust yet descriptive features.

In this section, we will first detail how we propose to learn such a representation, then provide two concrete applications, one on a global shape representation coined Learned-ESF (L-ESF), and one on a local shape representation coined Learned-SHOT (L-SHOT).

A. Learning histogram-like features

As illustrated in the previous section, most of the descriptors used in robotic rely on histograms, which is a crucial part of their robustness. Approaching noisy data as a set is a good trade-off between the amount of information discarded and the robustness of the description, and a histogram is the most straightforward way to approach set pooling. The PointNet architecture, by working on one data point at a time but optimizing over the whole set, provides a structure to learn functions that activate when a data point is present nearby. In [8], those functions behave like probabilistic bins over the Euclidean space. The global optimization ensures that those functions are optimally spread. An ensemble of such functions can therefore be seen as a probabilistic histogram that is trainable end-to-end. This idea can be extended to any other space.

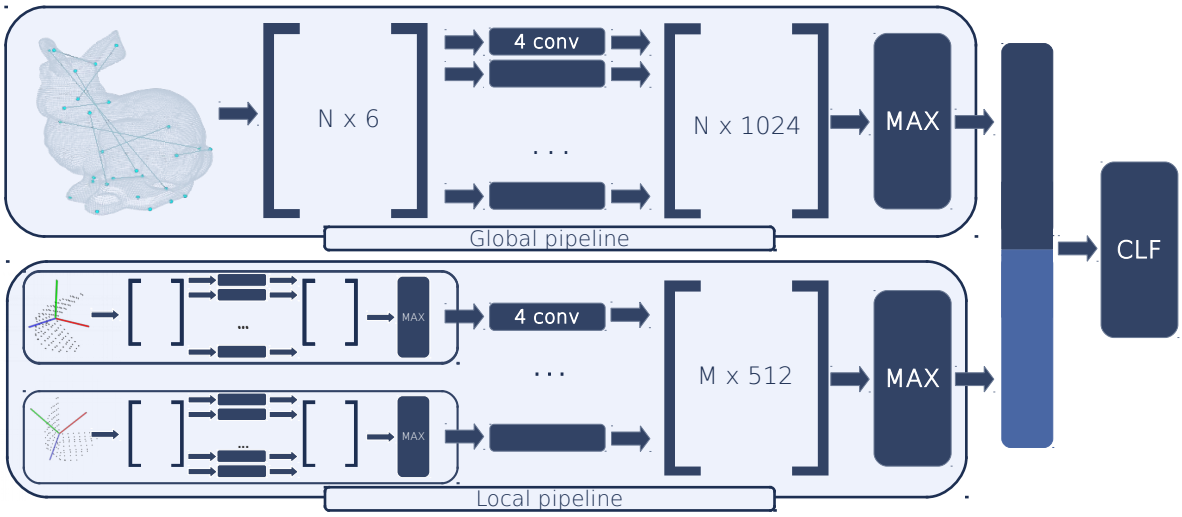

Fig. 1. System overview of our proposed method. The classifier layer is composed of two fully connected layers. N is the number of pairs sampled, M is the number of local descriptors sampled.

In this setup, the spatial transformer network of the original PointNet architecture is not necessary as we are not working on Euclidean coordinates. Instead, we use 4 one-dimensional convolutional layers, with a kernel size of 1, and a progressively increasing number of filters, and then a max-pooling layer to implement our histogram-like feature learning scheme. Each of the layer include a batch normalization step [4]. We also subtract a weighted version of the maximum value of a given filter over the whole set to each output, as described in [10]. We choose to use ReLU as an activation function.

B. Global Pipeline - Learned ESF

In a first step, we want to extract robust shape informations globally. This pipeline is strongly inspired by the ESF descriptor [18], which is to our knowledge the best performing global handcrafted shape descriptor that does not depend on the viewpoint choice.

The ESF descriptor computation first samples randomly pairs and triplets of points and extracts handcrafted features for each one. It then creates various histograms over them. For a pair of points, the features chosen are the distance between them, and the percentage of the line from one point to the other that is filled with surface points. This is done by tracing the line with the help of the 3D Bresenham algorithm in a voxel grid of size 64x64x64. For triplets of points, the angles of the triangle and the area covered by the triangle is computed. The robustness of this descriptor comes both from the use of histogram and its randomness.

In our pipeline, we decided to focus on pairs of points. On top of the features extracted for the ESF descriptor, we also draw inspiration from the Point Pair Features[1], and Fast Point Feature Histogram[13] descriptors. All the features we extract from each pair are rotation invariant, thus making the whole pipeline rotation invariant as well. That allows us to not have to rely on any form of alignment.

Based on these considerations, after scaling our point cloud to the unit sphere, our global shape descriptor computation starts by randomly sampling pairs of points. For $\vec{p}_1$ and $\vec{p}_2$ the two sampled points, and $\vec{n}_1$ and $\vec{n}_2$ their respective normal vectors (which are assumed to be normalized), we first extract the distance d between the points as in (1).

$$d = \left\| \vec{p}_1 - \vec{p}_2 \right\| \tag{1}$$

We also consider the cosine similarity between the normals (2), and the absolute value of the cosine similarity between the vector $\vec{p}_1 - \vec{p}_2$ and each of the normals (3)

$$cos(\angle(\vec{n}_1, \vec{n}_2)) = \vec{n}_1 . \vec{n}_2 \tag{2}$$

$$\left| cos(\angle(\vec{p}_1 - \vec{p}_2, \vec{n}_{\{1,2\}})) \right| = \left| \frac{(\vec{p}_1 - \vec{p}_2) . \vec{n}_{\{1,2\}}}{\left\| \vec{p}_1 - \vec{p}_2 \right\|} \right| \tag{3}$$

Finally, as in the ESF descriptor, we consider the percentage of $\vec{p}_1 - \vec{p}_2$ that is on the surface of the object. We follow the PointNet[8] architecture for the classification of the global pipeline: the set of features is first fed to a convolutional neural network, always operating on a single pair at a time. After enlarging the feature dimension, the set is then max-pooled.

Due to the intractable number of potential pairs, we tend to lose any information relative to finer structures during the sampling step. For this reason, we introduce a local descriptor pipeline.

C. Local Pipeline - Learned SHOT

As mentioned in the previous section, we need a way to capture finer structure to improve the descriptiveness of our approach. However, computing local descriptor densely

would be wasteful. As such, we need a way to direct our sampling of local patches.

1) Attention Model: Given the nature of our global shape, we choose to base our attention model on the statistical consistency of the normal orientations. Indeed, finer structures are characterized by higher angle between neighboring normals. However, we cannot solely rely on the local variations of the angle between normals, otherwise, any rounded surface would be considered salient. Consider the example of a flower pot: local descriptors on the leaf are probably more informative than redundant local descriptors on the pot itself.

We therefore chose to first create a histogram of angles between normals and their neighbors. We then normalize the histogram such that its sum is equal to one. With $\|P\|_0$ the number of non-zeros elements in our histogram and P_k the k-th entry in the histogram, we apply the following transformation:

$$\tilde{P}_k = \begin{cases} \|P\|_0 - P_k \text{ if } P_k \leq \|P\|_0 \\ 0 \text{ otherwise} \end{cases} \tag{4}$$

That allows us to only capture the statistically significant angles and remove all highly recurrent angles. We can then reproject the saliency value for each point by simply looking up the saliency value corresponding to the angle, and then summing over the whole neighborhood.

The actual sampling of salient points is done using a Poisson sampling. For each point sampled, we set a neighborhood of twice the number of points used for the descriptor computation to be invalid for sampling. However, we use the mean distribution between the distribution described before, and a uniform sampling to better capture the model specificities.

2) Local Descriptor: To design our local descriptor, we followed the design of the SHOT descriptors[16]. The first step for its computation is to compute a robust and repeatable local reference frame (LRF) to align our point against. Then, the sphere of all neighbors is divided in 32 bins, a histogram of normal angles being computed for each of them.

Considering the success of the PointNet architecture [8] over raw point coordinates, we chose a similar path when designing our local descriptor. We first transform all the neighboring points in the LRF computed as in the SHOT descriptor, instead of relying on a learned alignment, and use our sampled salient point as the origin. We can then directly feed the coordinates of our points to a smaller scale PointNet architecture. Indeed, by not having to include a spatial transformer network, we can drastically reduce our number of parameters.

IV. EXPERIMENTS

A. Invariance and Robustness

We designed our network around robust features. To demonstrate their intrinsic power, we devised a set of transformations applied to the input point cloud, and report the corresponding accuracy. **It is important to note that those experiments are performed without re-training the network, which is only trained on clean data.** The following experiments demonstrate that the network carry over the robustness of the chosen features. We report the results of our proposed architecture in comparison to the original descriptors, whose robustness has already been proven. No experiments are required regarding rotation invariance, as all features fed to the network are rotation invariant.

1) Point Density robustness: We want to evaluate how well our network behave depending on the point density of the point cloud. We chose to randomly remove points from the original instead of a more structured form of downsampling. The results of the experiments are reported in the figure IV-A.1

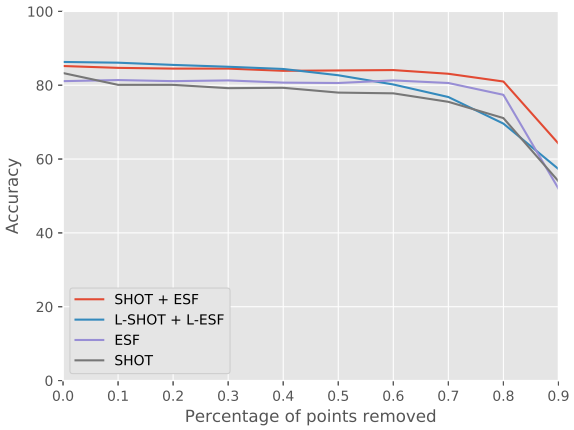

Fig. 2. Influence of the point density on the accuracy

2) Occlusion robustness: To simulate occlusion, we chose to remove a neighborhood around a randomly sampled point. The size of the neighborhood corresponds to the percentage of the points being removed. The results are reported in the figure IV-A.2

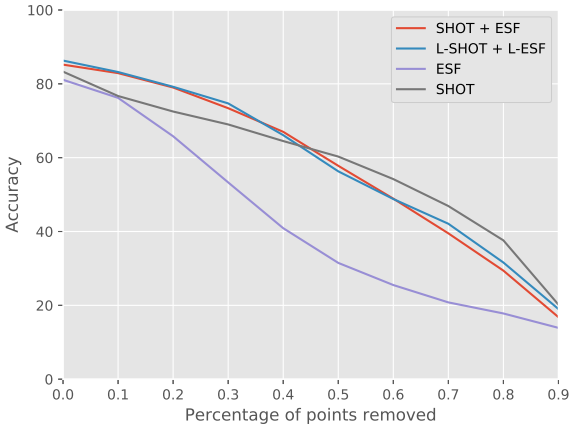

Fig. 3. Influence of occlusion on the accuracy

3) Sensor noise: Working with full models prevents us from using a realistic sensor noise model. However, we can still model a generic noise by adding Gaussian noise. We choose the standard deviation of our Gaussian noise as a proportion of the longest distance between points in the point cloud. The results are reported in the figure IV-A.3

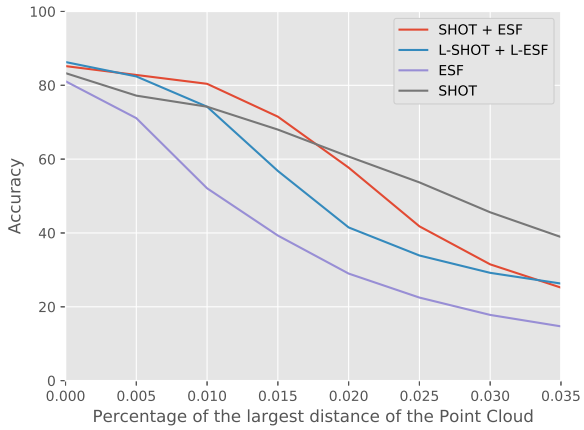

Fig. 4. Influence of sensor noise on the accuracy

B. ModelNet

We perform our evaluation on the ModelNet dataset[19], a CAD model dataset which has two variants, ModelNet40 with 40 classes, and ModelNet10 which is a 10 class subset of the ModelNet40.

Our method was designed to be used on Point Clouds, so as a pre-processing step, we extract Point cloud from the original CAD models by first down-sizing the CAD model to the unit sphere, and then sampling a point every 1mm of the surface of the point cloud for ModelNet40, and every 2mm for ModelNet10. Due to the random nature of our algorithm, we average the accuracy over 5 run on the test set.

For the global pipeline, we sample 5000 pairs on ModelNet40, and 3200 on ModelNet10. For the local pipeline, we sample 50 salient points drawn using Poisson sampling from a distribution which is the average of a uniform distribution and our attention model described above.

For the SHOT, ESF and SHOT+ESF results, we simply replace our learned version of the descriptor with the original descriptor, as implemented in [14]. We keep the same classification layers, and still merge the set of SHOT descriptors with a PointNet architecture for a fair comparison.

The accuracy results can be found in table I for ModelNet

C. Discussion

As described above, we achieve results that are better or on par with the methods based on voxel grids and classical descriptors, but worst than the view-based methods. It should however be noted that view-based methods uses architecture with a lot more parameters (10-100 millions compared to around 1 million for ours). Compared to point cloud based methods, Kd-Networks performs best but requires aligned

TABLE I
ACCURACY ON THE MODELNET DATASET [19]

	MN10	MN40	Input
3DShapeNets[19]	83.5	77	Voxel grid
VoxNet[7]	92	83	Voxel grid
PointNet[8]		89.2	Point Cloud
Kd-Networks[6]	94	91.8	KD-Tree
MVCNN[15]		90.1	Views
SHOT	83.3	73.9	Point Cloud
ESF	81.1	70.4	Point Cloud
SHOT+ESF	85.2	76.9	Point Cloud
Ours	**86.3**	**83.0**	Point cloud

models, so it is not as robust, and PointNet learns an alignment which is itself sensitive to occlusion and outliers.

In the case of ModelNet10, most of the misclassification are caused by confusion between `night_stand` and `dresser`, and between `table` and `desk`. In the case of ModelNet40, most of the misclassification are caused by confusion between `flower_pot` and `plant`, on top of the confusion made over the ModelNet10 dataset. A deeper observation of the CAD models in each of these class show that those mistakes are quite reasonable and shows promising potential application of this architecture as a robust generic shape feature.

In our study of the robustness of our model, we can see that most of the desirable properties of the classical descriptors are kept. Only the robustness to Gaussian noise seems lower. This is due to the setup of our experiment: by not retraining our network with any kind of data augmentation, this study focuses on the intrinsic properties of the features used, rather than on the already demonstrated learning capabilities of neural network. However, in a classical descriptor, the robustness to Gaussian noise mostly comes from the use of a histogram, and if our learned equivalent of a histogram has not faced noisy data during training, it is unlikely to cope with it during testing.

The key benefit of our method is its robustness: it carries over the benefits of classical handcrafted features, and it does not require any alignment of the models, as the feature extracted are themselves rotation-invariant. This allows us to use a more compact network, as we do not require parameters for a spatial transformer network, or additional parameters that would be necessary to cover the representation over many different orientations. Through the use of randomization, smaller parameter number and batch normalization, our model is less likely to overfit as every representation of a given instance is slightly different, both during training and testing.

V. CONCLUSIONS AND FUTURE WORK

We have shown in this paper a novel architecture that provide robust yet descriptive shape features. Moreover, the scheme devised in this paper can be used to adapt any local or global histogram-based handcrafted feature into a learned descriptor, thus providing better task specific performance and end-to-end learning.

The flexible nature of the architecture also allows its use in both a single and multi-view scenario. In the case of multiple views, it can perform efficiently, as information already computed over previous sets can easily be included to the next step through the max-pooling layer over the whole set, all that while still preserving inter-views information.

Finally, by keeping track of the indices used during the max pooling step, we can gather interpretable information regarding the contribution of local structures. As an extension, such local descriptors could be use for correspondence problems, such as pose estimation which can be done from sampling pairs, as demonstrated in [1].

REFERENCES

[1] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model Globally, Match Locally: Efficient and Robust 3d Object Recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[2] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal Deep Learning for Robust RGB-D Object Recognition," *arXiv:1507.06821 [cs]*, July 2015, arXiv: 1507.06821. [Online]. Available: http://arxiv.org/abs/1507.06821

[3] S. Gupta, R. Girshick, P. Arbelez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *Proceedings of the European Conference on Computer Vision*, ser. ECCV'14, 2014.

[4] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*, June 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[5] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, "Spatial Transformer Networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2017–2025. [Online]. Available: http://papers.nips.cc/paper/5854-spatial-transformer-networks.pdf

[6] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 922–928.

[8] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3d Classification and Segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[9] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5099–5108. [Online]. Available: http://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature-learning-on-point-sets-in-a-metric-space.pdf

[10] S. Ravanbakhsh, H. Su, J. Schneider, and B. Poczos, "Deep Learning with Sets and Point Clouds," *International Conference on Learning Representations (ICLR)*, 2017.

[11] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[12] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 2155–2162.

[13] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, ser. ICRA'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 1848–1853. [Online]. Available: http://dl.acm.org/citation.cfm?id=1703435.1703733

[14] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[15] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *Proc. ICCV*, 2015.

[16] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Proceedings of the 11th European Conference on Computer Vision Conference on Computer Vision: Part III*, ser. ECCV'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 356–369. [Online]. Available: http://dl.acm.org/citation.cfm?id=1927006.1927035

[17] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling for multi-view 3d object recognition." in *Proceedings of British Machine Vision Conference (BMVC). 2017*, 2017.

[18] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3d object classification," in *2011 IEEE International Conference on Robotics and Biomimetics*, Dec. 2011, pp. 2987–2992.

[19] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1912–1920.