Multi-Scale Transformer Features in 3D Object Detection

Aidana Nurakhmetova*, Mustaqeem Khan*, Abdulmotaleb El Saddik†* and Wail Gueaieb†*
*Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI)
Masdar City, Abu Dhabi, UAE
Email: {Aidana.Nurakhmetova, Mustaqeem.Khan}@mbzuai.ac.ae
†School of Electrical Engineering and Computer Science
University of Ottawa, Ottawa, ON, Canada K1N 6N5
Email: {elsaddik, wgueaieb}@uottawa.ca

***Abstract*—Transformer architecture in 3DETR is a simple kind with encoder-decoder layers and standard multi-head attention heads. For 3D input, such a style of a transformer may not work as efficiently as in other regular input types, such as long-range text or images. This is due to the three-dimensional view of the 3D point cloud scene, which is sparse and orderless. Therefore, extracting features from a global perspective capturing the context, and learning the locality play significant roles. Transformers are well-designed to understand the context but struggle with localized features. In 3D, local clues are essential to learn the geometry of objects, especially those with smaller sizes. Hence, we propose a multi-scale transformer with multi-head attention which helps extract features hierarchically. The proposed network achieves 62.32% (+0.99) and 43.51% (+4.78) on the ScanNetv2 benchmark dataset, as compared to our baseline.**

***Index Terms*—3D point clouds, Transformer, Object Detection**

I. Introduction

3D object detection poses a challenge due to the nature of point cloud input. The point cloud comprises unstructured and sparse points in a 3D dense scene. The applications of 3D object detection cover a wide range of fields such as robotics [1], [2], healthcare [3], [4], autonomous vehicles [5], and augmented and virtual realities [6]. This work focuses on indoor object detection only, as there is also an outdoor use case that is concerned with self-driving cars application. Various deep learning methodologies were used to extract features of 3D point cloud data efficiently. Amongst them, the PointNet++ [7] backbone played a critical role in allowing robust feature learning. Unlike regular deep learning (DL) 2D inputs and structured text/numeric data, 3D point cloud input is irregular, and thus requires a different way of handling the input before feeding to the DL networks. There are three major approaches to manipulating the point cloud data:

- Point-based methods
- Voxel and projection-based methods
- Transformer or attention-based methods
- Hybrid methods

In the related works section II, papers will be referred based on these different approaches.

Recent works on 3D scene understanding using deep learning have demonstrated significant advancements in various aspects of the field. Researchers have developed novel architectures and techniques to tackle tasks such as shape analysis, semantic segmentation, object alignment, grasp detection, and registration. Studies such as [8], [9], and [10] have explored network designs and convolutional operations tailored explicitly for 3D point cloud data. Other works, such as [11] and [12] have leveraged RGB-D reconstructions and CAD models to enhance scene understanding and alignment.

A number of survey papers were written for 3D point cloud processing with the help of deep learning such as [13], [14], [15], [16].

As of today, self-attention and transformer networks have been widely applied in 3D object recognition and segmentation problems. Usually, the complex and unstructured scene requires extra data manipulation through projection or voxelization, which is costly and complicated. Thanks to efficient feature learning through long-range data capture of transformers, it became possible to directly feed 3D point cloud input to the network. Transformer design allows obtaining a global view of the scene to learn the relationship between different object clusters. In the first end-to-end 3D object detection transformer architecture, 3DETR [17], the encoder is responsible for learning point-point relations. In contrast, the decoder learns the point-object and the object-object associations. However, due to the extensive complexity of scene points fed to the model, the original number of points is downsampled by almost $20\times$. Consequently, most of the information gets lost, and even for the most potent architecture, it is challenging to learn the shapes and attributes of scene objects. Therefore, we propose a technique to tackle this problem by generating more features through an upsampling operation that produces high-resolution feature maps. The upsampled features are processed by the multi-head attention of the decoder along with the original feature points. Hence, the contributions of our research are the following:

- Proposing a novel architecture of 3DETR with multi-scale attention (M-SA) mechanism that can extract finer features of smaller object categories.
- Improved performance in 3DETR (*mAP@25*, *mAP@50*)

and 3DETR-m (*mAP@25*) models, particularly for small-scale objects.

- Simple implementation and lightweight network which is utilized only in the first layer of the decoder.

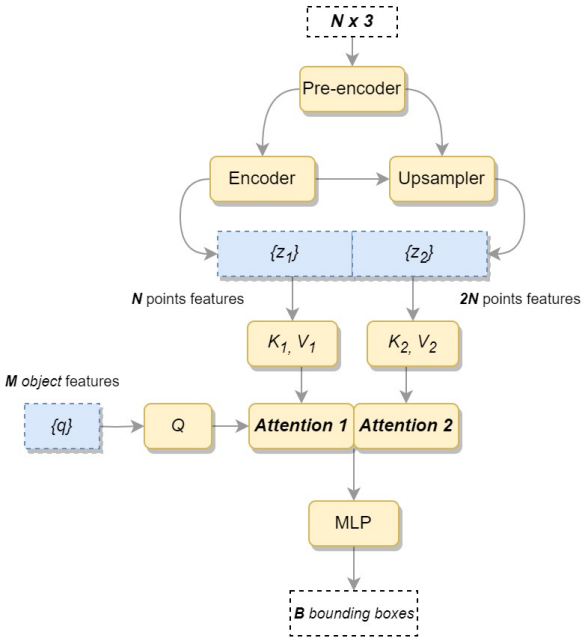


Fig. 1: New MHA layer in the proposed network. Note that the pre-encoder is the same SA layer 1.

II. Related Works

A. Point-based approaches

The PointMixer [18] employs a versatile operator for point sets, which facilitates the extraction of shared knowledge about 3D scenes. In this regard, the token-mixing multi-layer perceptrons (MLPs) are substituted with a softmax function. The EdgeConv [8] network module, which relies on convolutional neural networks (CNN), processes graph inputs at each layer and gathers local neighborhood features. The 3D-MPA [19] method proposes employing graph convolutional networks to capture the inter-proposal relations, thereby enhancing the understanding of the interactions between various objects and per-point features. BRNet [20] draws inspiration from the back-tracing strategy found in the traditional Hough voting algorithm. This strategy is utilized to improve the estimation of representative points surrounding the voted centers, and it also reassesses the seed points to achieve precise object localization.

B. Projection/Voxel-based approaches

MV3D [21] integrates LIDAR point cloud and RGB images using both 2D and 3D convolutions to generate features and predict oriented 3D boxes. The network achieves accurate object detection by utilizing a proposal generation subnetwork that projects the 3D point cloud into a bird's eye view. FCAF3D [22] introduces a new method for 3D object detection that eliminates the use of predefined anchor boxes. The approach utilizes a fully convolutional network and directly regresses the bounding box parameters, resulting in the accurate detection of objects in 3D point cloud data. The paper [23] presents a novel methodology for object detection by combining sparse detection networks with generative adversarial networks (GANs). By leveraging GANs to generate high-resolution object proposals, the approach enhances detection performance, especially in situations with limited training data.

C. Transformer-based approaches

Point Cloud Transformer (PCT) [24], specifically designed for analyzing point cloud data. By leveraging self-attention mechanisms and positional encoding, PCT effectively captures long-range dependencies and preserves spatial information. The Point Transformer [25] network is specifically developed for tasks involving classification and dense prediction, serving as a potential substitute backbone for various 3D scene analysis models. The unique aspect of the PT layer is its utilization of vector self-attention instead of traditional scalar attention, incorporating the subtraction relation function. The Pointformer [26] paper introduces an architecture that closely resembles a U-net and is solely based on transformers. It takes advantage of the transformers' capability to capture long-range dependencies.

D. Hybrid architectures

There are hybrid architectures in 3D point cloud processing that leverage the advantages of different methods. For instance, these architectures merge voxel-based techniques with attention mechanisms, or combine point or graph-based approaches with voxels or attention modules, creating unified and robust networks.

An example of such an approach is VoxSet [27], which utilizes a Transformer model to process 3D voxels and facilitate the retrieval of local and global information within a scene's points. The architecture incorporates a voxel-based set attention (VSA) module that replaces self-attention with two cross-attention mechanisms within each voxel. In Point-Voxel Transformer (PVT) [28], the point cloud is first converted into a voxel grid representation. Each voxel is then processed using a 3D CNN to extract local features. However, instead of directly using the voxel features, PVT leverages the Point-Voxel Transformer, which consists of multiple transformer layers. PatchFormer [29] is an efficient variant of the Point Transformer model. PatchFormer introduces a Patch Attention mechanism that divides the point cloud into patches, enabling more efficient computation while still capturing important spatial relationships. By focusing on patch-level interactions rather than point-level interactions, PatchFormer achieves competitive performance in point cloud tasks such as object classification and part segmentation with reduced computational complexity. Group-Free 3D [30] is a hybrid approach that leverages transformer models without the need for grouping or voxelization. The proposed method directly processes point cloud data using a point-based transformer

network, which captures the relationships and dependencies between individual points.

III. Approach

Recently, the groundbreaking work of Vaswani et al. [31] revolutionized feature extraction and learning by replacing convolutions with the attention mechanism. This shift has had a significant impact on 2D object detection, where transformer architectures have emerged as powerful tools for accurately predicting bounding boxes. Transformers excel in learning object relations and global context, eliminating the need for manual anchor design and non-maximum suppression steps.

A. Multi-scale transformer network

As can be seen in Figure 2, the input point cloud is passed through the first layer of set abstraction (SA) from PointNet++ where the original 40,000 points are downsampled to much lower proportion such as 1024 or 2048 points. Then, those fewer points representing the entire scene that was sampled by FPS (farthest points sampling) were further fed to transformer encoder layers. In a similar matter, a twice larger proportion is sampled with the SA layer for upsampling operation. The upsampled features are passed to the transformer decoder.

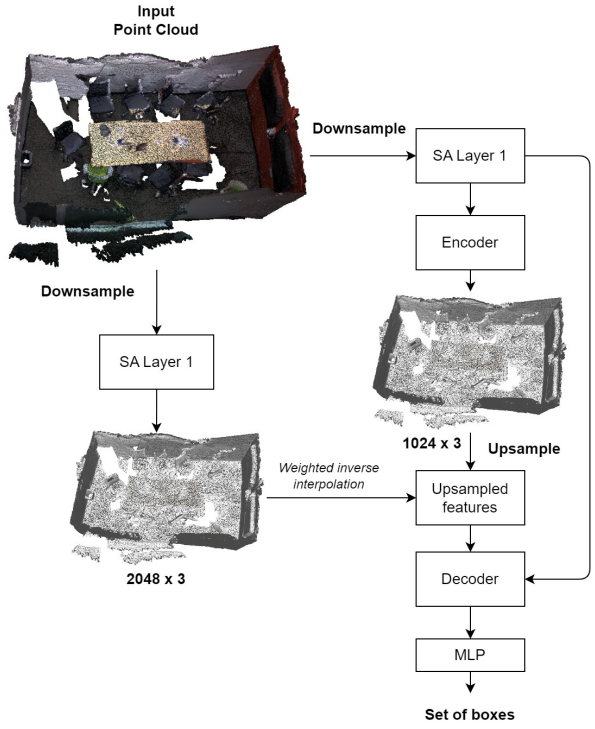


Fig. 2: The architecture of 3DETR model with multi-scale feature extraction

B. Upsampling features

As can be seen in Figure 3, from the bigger perspective, the upsampling algorithm consists of 3 main steps. First, the nearest three points $(x1, x2, x3)$ to the sampled point **p** are picked by the k-nearest-neighbors algorithm from the set of N encoder points according to Euclidean distance. The sampled points originate from the set of $2N$ points, as demonstrated in the figure (note that the size of the spheres is for illustration purposes only, the real point cloud scene is scattered and non-homogeneous). Secondly, weighted inverse distance (WID) interpolation is applied between the query points and the neighbor points. The output of interpolated values is the up-sampled features. Lastly, the multi-layer perceptron (MLP) is used to project the 3D channels into a larger 256-dimensional vector that would be passed to the decoder later.

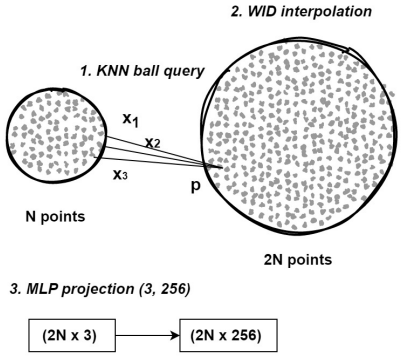


Fig. 3: Upsampling procedure in our proposed multi-scale transformer-based model according to Algorithm 1

Algorithm 1 Upsampling implementation

Ensure: $xyz, features = point\_clouds$
  $pre\_enc\_xyz \leftarrow pre\_encoder(xyz, features)$
Require: $sampled\_points = pre\_enc\_xyz$
  while $k \leq sampled\_points.length$ **do**
    $data \leftarrow point\_features[k]$
    $neighbors \leftarrow NearestNeighbors(3,' ball\_tree')$
    $query\_points \leftarrow sampled\_points[k]$
    $indices \leftarrow nbrs.kneighbors(query\_points, 3)$
    $neighbors \leftarrow data[indices]$
    $wid \leftarrow WID(neighbors, query\_points, 2)$
    $upsampled\_feats[k] \leftarrow wid$
  end while
  $feats \leftarrow mlp\_proj(upsampled\_feats)$

C. MHA with two attention maps

The self-attention operation, as described in [31], involves taking the dot product between a *Key* and a *Query*, which is then scaled based on the embedding vector dimension, denoted as d. This scaled attention $Q \times K^T/d$ is multiplied by a Value matrix, resulting in a weighted sum of matrix elements. In equation 1, it is shown that the input matrices for key, query, and value are all derived from the input X, where K, Q, and V are equal to X and belong to the matrix space $\mathbb{R}^{N \times d}$.

$$Attention = Z(Q, K, V) = \sigma(\frac{QW^q(KW^k)^T}{\sqrt{d}})(VW^v) \quad (1)$$

Algorithm 2 MS-A implementation

$WID(neighbors, queries, power)$:
$dist1 \leftarrow linalg.norm(queries[:] - neighbors[0,:])$
$dist2 \leftarrow linalg.norm(queries[:] - neighbors[1,:])$
$dist3 \leftarrow linalg.norm(queries[:] - neighbors[2,:])$
Ensure: $\epsilon = 1e-8$
Require: $indices = where(dist1 == 0)[0]$
Require: $indices = where(dist2 == 0)[0]$
Require: $indices = where(dist3 == 0)[0]$
if $min(dist1) \leq \epsilon$ **then**
$min_v \leftarrow neighbors[0, indices, 0]$
$dist1[indices, 0] \leftarrow min_v$
end if
if $min(dist2) \leq \epsilon$ **then**
$min_v \leftarrow neighbors[0, indices, 0]$
$dist2[indices, 0] \leftarrow min_v$
end if
if $min(dist3) \leq \epsilon$ **then**
$min_v \leftarrow neighbors[0, indices, 0]$
$dist3[indices, 0] \leftarrow min_v$
end if
$w1 \leftarrow 1/(dist1 * *power)$
$w2 \leftarrow 1/(dist2 * *power)$
$w3 \leftarrow 1/(dist3 * *power)$
$w\_sum \leftarrow w1 + w2 + w3$
$w1\_norm \leftarrow w1/w\_sum$
$w2\_norm \leftarrow w2/w\_sum$
$w3\_norm \leftarrow w3/w\_sum$
$value \leftarrow w1\_norm * neighbors[0,:] + w2\_norm * neighbors[1,:] + w3\_norm * neighbors[2,:]$

where, $W_q, W_k, W_v \in R^{d \times d}$ represent learnable matrices and σ is for Softmax operation.

$$MHA = Z(QW_i^q, KW_i^k, VW_i^v) \quad for\ i = 1, ..., h \tag{2}$$

The final attention map, represented as Z, is obtained by applying a formula similar to the one shown in equation 1, across the different heads. In the Multi-Head Attention (MHA) mechanism, the input set is divided into multiple heads. Each head independently and in parallel learns distinct features. The outputs from all the heads are then combined using linear projection, which preserves the original dimensionality.

As for the updated *MHA* in our proposed network provided in Figure 1, there are two distinct attention maps that have separate *Key, Value* pairs. In the first group of key-value pairs K_1, V_1 represent the original point features from the encoder, whereas K_2, V_2 represent the upsampled point features. Following this, two attention maps were created where *Query* is common, as shown in the equation below.

$$A_1 = Z(Q, K_1, V_1) = \sigma(\frac{QW_1^q(K_1W_1^k)^T}{\sqrt{d}})(V_1W_1^v) \tag{3}$$

$$A_2 = Z(Q, K_2, V_2) = \sigma(\frac{QW_2^q(K_2W_2^k)^T}{\sqrt{d}})(V_2W_2^v) \tag{4}$$

The final attention map is the concatenation of A_1 and A_2 in channel dimension. It is important to note that the weight matrix is divided across two key-value pairs in a logical way that mimics having two attention heads.

IV. Experiments

Dataset and metrics. ScanNetv2 [32] is an indoor dataset that contains highly detailed 3D reconstructions of 1513 indoor scenes, covering 18 different object categories. Users have the option to include color and height information, in addition to the (x, y, z) positions, as extra features. The dataset provides comprehensive annotations, including 3D bounding boxes and per-point instance semantic labels. Evaluation is performed using the standard mean Average Precision (mAP) metric with IoU thresholds of 0.25 and 0.5, without considering oriented bounding boxes. The dataset is split into a training set with 1201 scenes and a separate test set consisting of 312 scenes for validation. After training, the same test set with shuffled data points is used for evaluation purposes.

Implementation details. 2048 original points are fed to an encoder, and 4096 points are fed to the SA1 layer of the PointNet++ from where higher resolution features are obtained. Regarding complexity, MS-A is not heavy as it is used only in the first layer of a transformer. However, compared to the original model, the training and inference are longer due to more points being processed.

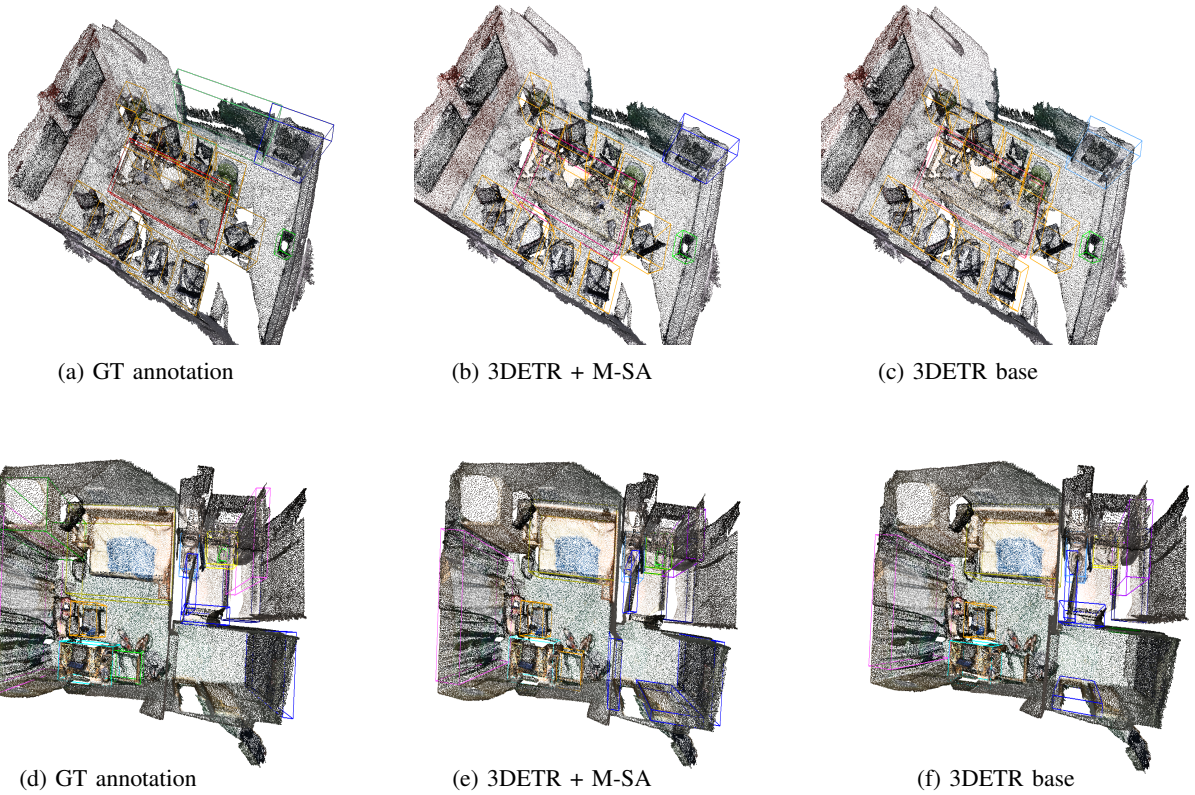
V. Results and Discussion

The quantitative outputs are provided in Table I, where 3DETR + M-SA shows **62.32% (+0.99)** and **43.51% (+4.78)**, as compared to our base model. With the proposed network, we can notice how it can detect smaller objects better compared to paper results such as the **picture (+3.51)** and **garbage bin (+1.4)** , as in Table III. We can observe the increased performance in objects which are semantically related to each other such as the table and chair, desk, cabinet, and bookshelf. These objects usually co-occur and are placed near physically. This shows with the new MHA, local clues were better extracted while not losing the context. Qualitative visualizations are provided in Figure 4.

According to qualitative outputs, the proposed model can detect objects such as doors and garbage bins as shown in Figure 4, while the 3DETR base model misses those object categories. Although the proposed model seems to have more false positive examples, it does not omit most objects like the base.

VI. Ablation Studies

Figure II shows the outcomes for the 3DETR-m model variant with M-SA applied. The performance does not improve for this model in both mAPs (in mAP@25, there is +0.79 gain), most probably due to the interim downsampling stage after the encoder, which eliminates half of the original



(a) GT annotation (b) 3DETR + M-SA (c) 3DETR base

(d) GT annotation (e) 3DETR + M-SA (f) 3DETR base

Fig. 4: Qualitative outputs (from left to right): (a) Original ground-truth labels. (b) The window object is missing. (c) The door object is detected wrongly as a bookshelf and the window is missing. (d) Original ground-truth labels. (e) The small-size object garbage bin is detected. (f) The garbage bin and one of the doors are missing. Hence, the proposed 3DETR + M-SA is superior to the 3DETR base.

TABLE I: Mean Average Precision (mAP) performance of the popular models trained on ScanNetv2 dataset. PN++ is for PointNet++. PN++w2 shows the backbone has a width of 2 in MLP networks. SA1 is the first layer of set abstraction in PointNet++.

Architecture	**Backbone**	**mAP@25**	**mAP@50**
VoteNet	PN++	58.6	33.5
MLCVNet	PN++	64.5	41.4
Group-Free (L6, O256)	PN++w2 ×	67.3 (66.2)	48.9 (48.4)
3DETR (paper)	SA1	62.7	37.5
3DETR (our baseline)	SA1	61.33	38.73
3DETR + MS-A	SA1	**62.32 (+0.99)**	**43.51 (+4.78)**

TABLE II: Mean Average Precision (mAP) performance of the popular models trained on ScanNetv2 dataset. PN++ is for PointNet++. PN++w2 shows the backbone has a width of 2 in MLP networks. SA1 is the first layer of set abstraction in PointNet++.

Architecture	**Backbone**	**mAP@25**	**mAP@50**
VoteNet	PN++	58.6	33.5
MLCVNet	PN++	64.5	41.4
Group-Free (L6, O256)	PN++w2 ×	67.3 (66.2)	48.9 (48.4)
3DETR-m (paper)	SA1	65.0	47.0
3DETR-m (our baseline)	SA1	64.88	46.58
3DETR-m + MS-A	SA1	**65.67 (+0.79)**	45.35

number of points by masking out. Hence, it affects the process of creating higher-resolution feature maps. Thus, 3DETR-m requires adapting the upsampling if we want to use M-SA. Due to the scope of the paper, it can be considered as future research.

VII. CONCLUSION AND FUTURE WORK

In summary, we suggested an enhanced transformer-based multi-scale feature extraction tailored to small objects in this research work. Our results show a performance gain of 62.32% (+0.99) and 43.51% (+4.78) on the ScanNetv2 dataset compared to baseline 3DETR. Future research can be done to improve the feature extraction step for transformer networks.

REFERENCES

[1] H.-Y. Kuo, H.-R. Su, S.-H. Lai, and C.-C. Wu, "3d object detection and pose estimation from depth image for robotic bin picking," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, 2014, pp. 1264–1269.

[2] S. M. Ahmed, Y. Z. Tan, C. M. Chew, A. A. Mamun, and F. S. Wong, "Edge and corner detection for unorganized 3d point clouds with application to robotic welding," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7350–7355.

TABLE III: Per-category evaluation of our proposed 3DETR on validation ScanNetV2 at AP@25 IoU. Numbers highlighted in bold show increases as compared to baseline results.

Method	cab	bed	chair	sofa	table	door	window	bkshf	pic	contr	desk	curtain	frdge	shower	toilet	sink	bath	grbin	mAP
3DETR (paper)	50.2	87.0	86.0	87.1	61.6	46.6	40.1	54.5	9.1	62.8	69.5	48.4	50.9	68.4	97.9	67.6	85.9	45.8	62.7
3DETR (our baseline)	44.98	82.84	86.7	89.66	63.37	46.57	37.7	43.38	9.95	59.84	75	60.19	40.74	60.71	98.72	68.23	88.97	46.34	61.33
3DETR + MS-A	**49.58**	82.65	**87.78**	86.82	**66.51**	**47.48**	34.78	**43.73**	**13.46**	**61.42**	**78.41**	57.21	**49.29**	**70.51**	98.11	65.80	80.52	**47.74**	**62.32**
3DETR-m	52.71	80.71	89.75	89.92	66.04	54.91	38.14	51.48	13.52	58.76	75.08	47.09	56.6	62.86	94.34	70.17	85.96	51.42	64.88
3DETR-m + MS-A	51.08	**82.66**	**90.11**	87.86	65.85	53.45	**41.06**	**55.00**	**16.20**	**60.33**	**77.49**	**55.66**	53.96	**68.38**	**98.28**	**73.31**	**93.16**	51.25	**65.67**

TABLE IV: The color map of object categories visualized in bounding boxes.

Object name	Bounding box color	Object name	Bounding box color	Object name	Bounding box color
Cabinet	Green	Counter	Salmon	Window	Sea Green
Bed	Lime Green	Desk	Sky Blue	Sink	Yellow
Chair	Orange	Curtain	Violet Red	Bookshelf	Periwinkle
Sofa	Lavender	Refrig.	Bitter Sweet	Bathtube	Blue Violet
Table	Rhodamine	Shower Curt.	Purple	Picture	Olive Green
Door	Blue	Toilet	Cadet Blue	Garb.bin	Yellow Green

[3] I. Birri, B. S. B. Dewantara, and D. Pramadihanto, "3d object detection and recognition based on rgbd images for healthcare robot," in *2021 International Electronics Symposium (IES)*, 2021, pp. 173–178.

[4] S. T. L. Pöhlmann, E. F. Harkness, C. J. Taylor, and S. M. Astley, "Evaluation of kinect 3d sensor for healthcare imaging," *Journal of Medical and Biological Engineering*, vol. 36, no. 6, pp. 2199–4757, 2016.

[5] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3d object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019.

[6] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3300061.3300116

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," 2017. [Online]. Available: https://arxiv.org/abs/1706.02413

[8] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," 2018. [Online]. Available: https://arxiv.org/abs/1801.07829

[9] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on $\mathcal{X}$-transformed points," 2018.

[10] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "Pointsift: A sift-like network module for 3d point cloud semantic segmentation," 2018.

[11] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," 2017.

[12] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Nießner, "Scan2cad: Learning cad model alignment in rgb-d scans," 2018.

[13] A. Xiao, J. Huang, D. Guan, X. Zhang, and S. Lu, "Unsupervised point cloud representation learning with deep neural networks: A survey," 2022. [Online]. Available: https://arxiv.org/abs/2202.13589

[14] J. Lahoud, J. Cao, F. S. Khan, H. Cholakkal, R. M. Anwer, S. Khan, and M.-H. Yang, "3d vision with transformers: A survey," 2022. [Online]. Available: https://arxiv.org/abs/2208.04309

[15] M. M. Rahman, Y. Tan, J. Xue, and K. Lu, "Notice of violation of ieee publication principles: Recent advances in 3d object detection in the era of deep neural networks: A survey," *IEEE Transactions on Image Processing*, vol. 29, pp. 2947–2962, 2020.

[16] A. Xiao, X. Zhang, L. Shao, and S. Lu, "A survey of label-efficient deep learning for 3d point clouds," 2023.

[17] I. Misra, R. Girdhar, and A. Joulin, "An end-to-end transformer model for 3d object detection," 2021. [Online]. Available: https://arxiv.org/abs/2109.08141

[18] J. Choe, C. Park, F. Rameau, J. Park, and I. S. Kweon, "Pointmixer: Mlp-mixer for point cloud understanding," 2021. [Online]. Available: https://arxiv.org/abs/2111.11187

[19] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, and M. Nießner, "3d-mpa: Multi proposal aggregation for 3d semantic instance segmentation," 2020. [Online]. Available: https://arxiv.org/abs/2003.13867

[20] B. Cheng, L. Sheng, S. Shi, M. Yang, and D. Xu, "Back-tracing representative points for voting-based 3d object detection in point clouds," 2021. [Online]. Available: https://arxiv.org/abs/2104.06114

[21] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," 2016. [Online]. Available: https://arxiv.org/abs/1611.07759

[22] D. Rukhovich, A. Vorontsova, and A. Konushin, "Fcaf3d: Fully convolutional anchor-free 3d object detection," 2021. [Online]. Available: https://arxiv.org/abs/2112.00322

[23] J. Gwak, C. Choy, and S. Savarese, "Generative sparse detection networks for 3d single-shot object detection," 2020. [Online]. Available: https://arxiv.org/abs/2006.12356

[24] 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, apr 2021.

[25] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," 2020. [Online]. Available: https://arxiv.org/abs/2012.09164

[26] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," 2020. [Online]. Available: https://arxiv.org/abs/2012.11409

[27] C. He, R. Li, S. Li, and L. Zhang, "Voxel set transformer: A set-to-set approach to 3d object detection from point clouds," 2022. [Online]. Available: https://arxiv.org/abs/2203.10314

[28] C. Zhang, H. Wan, X. Shen, and Z. Wu, "Pvt: Point-voxel transformer for point cloud learning," 2021. [Online]. Available: https://arxiv.org/abs/2108.06076

[29] ——, "Patchformer: An efficient point transformer with patch attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 11 799–11 808.

[30] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong, "Group-free 3d object detection via transformers," 2021. [Online]. Available: https://arxiv.org/abs/2104.00678

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[32] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner, "Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans," 2017. [Online]. Available: https://arxiv.org/abs/1712.10215