Temporal Bird's Eye View for 3D Semantic Segmentation

Fabian Duerr

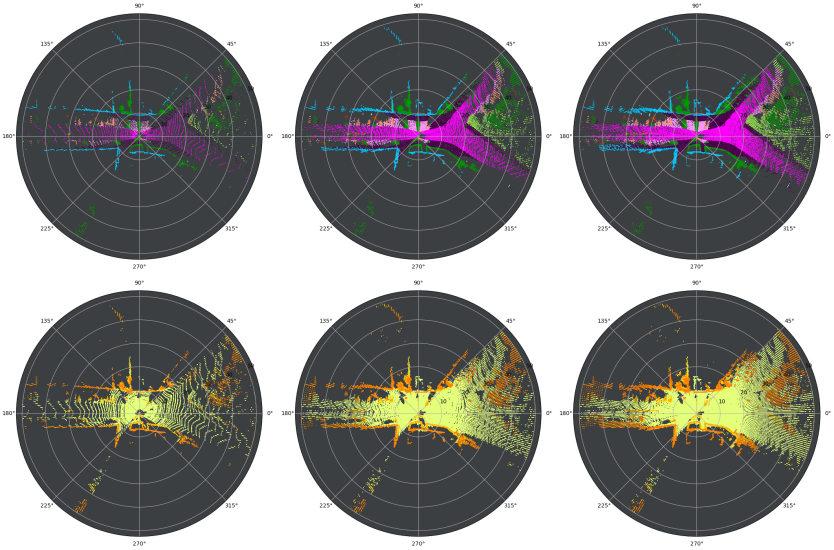
Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
fabian.duerr@partner.kit.edu

Abstract

Due to the growing importance of autonomous robots and vehicles, 3D semantic segmentation, a key task of 3D scene understanding, has become more and more important. Despite its sequential nature in real-time scenarios, 3D semantic segmentation is often approached as single frame problem. However, temporal dependencies and information offer a huge potential to improve the predictions. Therefore, we propose a recurrent temporal architecture for 3D semantic segmentation, which exploits temporal information at the input and feature stage, to maximize the temporal benefits. Aggregated point clouds in bird's eye view increase the information provided to the backbone and temporally fused feature maps exploit temporal dependencies on feature level. The experiments conducted on a challenging and large-scale outdoor dataset show considerable improvements compared to a single frame baseline. The temporal information improve the results for every individual class.

1 Introduction

Living in a 3D world, one of the key challenges for autonomous robots is the understanding and interpretation of their 3D environment. While point clouds



(a) Single frame. (b) Five aggregated frames. (c) Ten aggregated frames.

Figure 1.1: Potential of aggregating consecutive frames in bird's eye view (BEV), visualized by the semantic segmentation at the top and the input point cloud colored by height at the bottom. While the single frame BEV (a) is relatively sparse, the aggregation of five (b) or ten previous frames (c) is much denser in the occupied areas and therefore adds a lot of information. This can not only be exploited for the shown input data but also for feature maps of the backbone.

already provide valuable geometric information, 3D semantic segmentation adds a class label to every individual point and therefore additional semantic information, which is often seen as key enabler for 3D scene understanding.
To tackle semantic segmentation of 3D point clouds, a proper representation or architecture is required to solve this task with established deep learning based approaches. While point-based approaches [21, 27] directly process raw point clouds, they deploy special architectures and convolution operations to deal with the unstructured data. To enable conventional convolutions and architectures, projection-based methods [20, 34] transform the point clouds into a regular space, e.g. grid.
An important property of real-time environment perception is the sequential

nature of the recorded sensor data. Temporal relations and information offer a huge potential to improve 3D semantic segmentation. As the environment does not change drastically during the recording of two consecutive frames, previous frames contain valuable information also for the current frame, see Fig. 1.1. The amount of information naturally diminishes with the temporal distance. For real-time applications, only past frames can be exploited whereas accessing future frames is not possible.
In this work, we present an efficient temporal semantic segmentation approach building upon bird's eye view (BEV) representation and exploiting temporal information at two stages. At the input stage, the point cloud of the current frame and the aggregated past point clouds are fused to increase the point cloud density and therefore input information. At feature stage, features of the current frame are fused with features from a temporal memory, which contains aggregated past information, following the idea of [10]. A feature alignment step in BEV space allows the reuse of computations from previous frames in both stages, which enables an efficient recurrent architecture. The benefits of the temporal fusion are twofold, it improves existing features by fusion, based on aggregated past information and additionally increases the density of the BEV.
To summarize our contributions, we propose:

- A temporal input memory, which efficiently aggregates input point clouds in BEV over time to increase information provided to the backbone.
- A temporal feature memory, which efficiently aggregates feature maps computed by the backbone, to provide aggregated information of the current frame and past frames to the semantic head, to further improve the predicted 3D semantic segmentation.

2 Related Work

2.1 3D Semantic Segmentation

As an integral part of 3D scene understanding, 3D semantic segmentation has drawn a lot of attention over the past years. Enabled by the availability of a

constantly increasing number of datasets [1, 3, 5, 29] considerable progress has been achieved. In contrast to images, a preliminary consideration about the representation is required, to tackle this task with Convolutional Neural Networks (CNN). Almost all representations proposed so far can be assigned to one of two main categories.
Point-based methods [14, 16, 21, 22, 27, 28] directly operate on the 3D point clouds and rely on adapted convolution operations and special network architectures. Projection-based methods transform the point clouds into a regular space, which enables the application of conventional convolutions and architectures. Based on the target space, these approaches can further be divided into subcategories, like dense and sparse voxel grids [7, 23, 26], range images [9, 20, 30] or bird's eye view (BEV). Zhang et al. [33] build upon a 3D occupancy grid but treat the z-axis as feature dimension and therefore work with a 2D BEV representation as input. PolarNet [34] proposes a 2D polar BEV representation, which is based on a learned PointNet [21] encoding of all points lying inside a BEV cell. In the last stage of the network the 2D feature maps are expanded to a 3D polar grid prediction. Motivated by the promising results of PolarNet and the general potential of the BEV representation for temporal fusion, the presented approach builds upon the polar BEV representation.

2.2 Temporal Point Cloud Fusion

The majority of the methods proposed so far treat semantic segmentation as single frame task. Most of the approaches, which exploit temporal information on feature level aim for 3D object detection [15, 17, 18, 24, 31], only a few approaches for 3D semantic segmentation exist [6, 8, 10, 25].
Yin et al. [31] tackle temporal object detection in BEV representation with an RNN-based architecture building upon an extended ConvGRU [2], called attentive spatio-temporal GRU. It aggregates spatio-temporal information to exploit temporal dependencies of the point cloud sequences. For the same task, Huang et al. [15] exploit a sparse 3D voxel representation and propose a LSTM to fuse sparse features from previous and the current frame. The object detection head is then applied to the temporally fused features.

For semantic segmentation, MinkowskiNet [8] uses the fourth dimension to include previous frames and relies on sparse convolutions to handle the dominating empty cells. One disadvantage is the dependence of the run-time on the number of past frames considered. SpSequenceNet [25], which relies on the backbone of [12], and a voxel-based representation, proposes a cross-frame global attention layer, which highlights features of the current frame based on past information. Cross-frame local interpolation targets the temporal combination of local information. However, this approach is only designed to exploit the last frame. Li et al. [6] exploit temporal information in range image space for moving object segmentation. Past distance information is transformed into the current frame and residual images are computed using the difference of the transformed past distance values and the current values. The residual images are then used as additional input channels to a CNN. TemporalLidarSeg [10] builds upon an RNN-based architecture and passes a recursively aggregated temporal memory through time. An alignment step improves temporal consistency by compensating the ego motion. The temporal information provided by the memory considerably improves the semantic segmentation results.
Most of the listed approaches are not capable of exploiting information over a larger number of past frames due to design [25] or increasing run-time [8, 31]. The presented approach however is able to exploit temporal information of sequences of arbitrary length in constant time, comparable to TemporalLidarSeg [10]. However, instead of using range images like the latter, the presented approach relies on the BEV representation, which offers additional possibilities for temporal fusion.

3 Recurrent 3D Semantic Segmentation

The goal of the presented approach is the exploitation of temporal information in BEV to improve 3D semantic segmentation. Therefore, the architecture is designed to use past information at two stages, see Fig. 3.1. Starting at the input stage, the BEV image fed to the backbone does not only contain the point cloud of the current time step but also the aggregated point clouds of the previous frames. The second temporal fusion stage is designed to fuse the feature maps

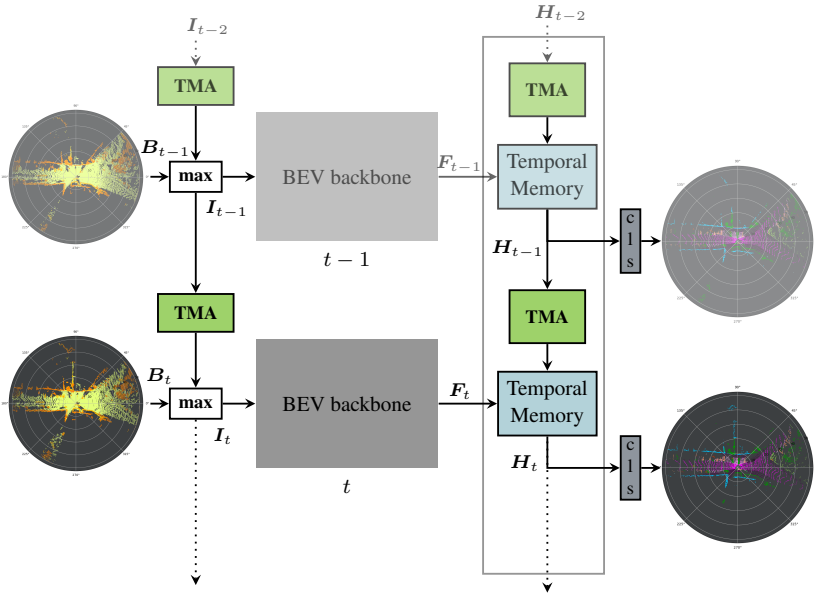


Figure 3.1: Overview of the recurrent temporal architecture, unrolled for two time steps. Aggregated input point clouds in BEV are fed to the backbone, which computes intermediate feature maps. Based on these features a temporal memory containing the aggregated past information is updated. The final semantic segmentation is computed from these temporally fused features.

computed by the BEV backbone. Therefore, they are used to update a temporal memory, which contains the aggregated past feature maps. The temporally fused and aggregated features are then expanded to a 3D polar grid and used for the final semantic predictions. Both stages efficiently reuse computations from the previous time steps.

Bird's Eye View Backbone Like single frame approaches, the presented architecture has a backbone responsible for computing feature maps of point clouds represented as BEV images. The differences however are twofold. First, while still taking only a single BEV image as input, it contains the aggregate point clouds of the current frame as well as past frames. Secondly, the intermediate

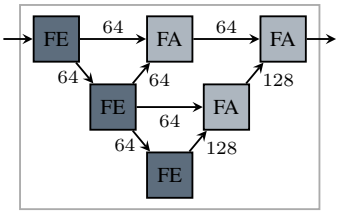


Figure 3.2: Backbone for computing feature maps for point clouds represented in BEV. The edges are labeled with the channel size of the feature maps.

feature maps are computed for the temporal memory instead of the semantic head, see Fig. 3.1.
In general, we build upon the ideas of PolarNet [34] but use a different backbone and reduce the channels of the PointNet, which encodes the polar input, to 16, 24 and 32. The deployed backbone is based on deep layer aggregation [32] and additionally motivated by LaserNet [19]. It is build of three feature extractors (FE) with four, five and six residual blocks [13] and a downsampling ratio of two. Feature aggregators (FA) apply a transposed convolution to upsample their lower resolution input and concatenate both inputs, followed by two residual blocks. The backbone architecture is depicted in Fig. 3.2.

Temporal BEV Alignment In order to realize the mentioned temporal aggregation of BEV images containing input point clouds or deep features, a recursive transformation of BEV images from the last to the current time step is required. This allows to reuse already aggregated point clouds or computed features of past frames, following the idea of [10]. The temporal alignment itself is required because of the ego motion, which changes the sensor origin.
Generally, an important relation for 3D semantic segmentation based on BEV images is the mapping from a 3D point $\boldsymbol{p} = (x, y, z)^T$ to its corresponding 2D BEV cell $\boldsymbol{u} = (u, v)^T$, which is described by

$$\boldsymbol{\mathcal{P}} : \mathbb{R}^3 \rightarrow [1, H] \times [1, W] \subset \mathbb{N}^2 \;\Rightarrow\; \boldsymbol{\mathcal{P}}(\boldsymbol{p}) = \begin{pmatrix} \left\lfloor \frac{\sqrt{x^2+y^2}}{\tilde{r}} \right\rfloor \\ \left\lfloor \frac{\text{atan2}(y,x)}{\tilde{\alpha}} \right\rfloor \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}, \quad (3.1)$$

with the image resolution $(\tilde{r}, \tilde{\alpha})$ and size $H \times W$. The z-coordinate does not have any influence on this projection. For the transformation of a BEV image from the last to the current time step, the cell centers and not the contained points are considered. Therefore, the cartesian coordinates of every cell's center are required and computed following

$$\begin{gathered} \boldsymbol{\mathcal{C}} : [1, H] \times [1, W] \subset \mathbb{N}^2 \rightarrow \mathbb{R}^3 \Rightarrow \\ \boldsymbol{\mathcal{C}}(\boldsymbol{u}) = \begin{pmatrix} \tilde{r} \cdot (u + 0.5) \cdot \cos((v + 0.5) \cdot \tilde{\alpha}) \\ \tilde{r} \cdot (u + 0.5) \cdot \sin((v + 0.5) \cdot \tilde{\alpha}) \\ 0 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \boldsymbol{p}. \end{gathered} \tag{3.2}$$

The cartesian cell centers are then transformed from the last sensor pose $\boldsymbol{T}_{t-1}$ to the current sensor pose $\boldsymbol{T}_t$:

$$\boldsymbol{\mathcal{T}} : \mathbb{R}^4 \rightarrow \mathbb{R}^4 \;\Rightarrow\; \boldsymbol{\mathcal{T}}(\boldsymbol{p}_{t-1}) = \boldsymbol{T}_t^{-1} \cdot \boldsymbol{T}_{t-1} \cdot \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ z_{t-1} \\ 1 \end{pmatrix} = \begin{pmatrix} {}^t x_{t-1} \\ {}^t y_{t-1} \\ {}^t z_{t-1} \\ 1 \end{pmatrix}. \tag{3.3}$$

The combination of these steps provides the position of the cells of the last BEV in the BEV of the current time step:

$$\begin{gathered} \boldsymbol{\mathcal{A}} : [1, H] \times [1, W] \subset \mathbb{N}^2 \rightarrow [1, H] \times [1, W] \subset \mathbb{N}^2 \Rightarrow \\ \boldsymbol{\mathcal{A}}(\boldsymbol{u}_{t-1}) = (\boldsymbol{\mathcal{P}} \circ \boldsymbol{\mathcal{T}} \circ \boldsymbol{\mathcal{C}})(\boldsymbol{u}_{t-1}) = ({}^t u_{t-1}, {}^t v_{t-1})^T = {}^t \boldsymbol{u}_{t-1}. \end{gathered} \tag{3.4}$$

This temporal transformation can then be used to transform the content of the BEV image at time $t-1$ to the BEV image at time t.

Input Alignment and Fusion The temporal transformation presented in the previous section is used for the first time at the input stage. The input memory $\boldsymbol{I}_{t-1}$ containing the aggregated point clouds until frame $t-1$ is transformed to the current time step t using the indices computed by Eq. 3.4. Cells of the memory, which lie outside the current BEV after the transformation are discarded. The transformed input memory is then fused with the input BEV $\boldsymbol{B}_t$, containing the current point cloud, see Fig. 3.1. The fusion is done by channel-wise maximum, following the PointNet encoding, which performs a channel-wise maximum over the feature vectors of all points lying inside one cell.

Feature Alignment and Fusion Following the same temporal transformation, the temporal memory $\boldsymbol{H}_{t-1}$ at feature level, which contains the aggregate output features of the past frames up to $t-1$, is transformed to the current frame. The features $\boldsymbol{F}_t$ of the current input, computed by the backbone, are then used to update the transformed temporal memory, following the residual update strategy presented in [10].

4 Experiments

4.1 SemanticKITTI

The experiments for the presented approach are conducted on the large-scale and challenging SemanticKITTI dataset [3, 11]. The single scan benchmark provides point-wise annotations of 19 classes for $360°$-Velodyne-HDL-64E scans. Over $43{,}000$ scans are divided into 22 sequences of varying length and recorded at 10 Hz. The first half of the sequences are provided with labels for training and validation while the test split is defined by the second half, with no labels published. We follow the official recommendation and use sequence 08 for validation and report the mean Intersection-over-Union (mIoU) as evaluation metric.

4.2 Implementation Details

The approach is implemented in PyTorch and trained in mixed precision mode on four Tesla V100 GPUs using distributed data parallel training.
Cross entropy and Lovász loss [4] are optimized equally weighted by Adam for $75k$ iterations. To prevent overfitting, weight decay of 0.0005 is applied as well as extensive data augmentation. Before being projected, the point clouds are randomly flipped along x- and y-axis with a probability of 0.5, randomly rotated around the z-axis and randomly cropped to a $180°$ crop. Additionally, objects of underrepresented classes, like bicycle or motorcycle, are randomly pasted into the point clouds. These objects are extracted upfront from the training set and placed on corresponding ground classes like road or sidewalk.

Backbone	TIM	TFM	mIoU (%)
✓			58.0
✓	✓		58.7
✓		✓	64.5
✓	✓	✓	**64.7**

Table 4.1: Improvements on the validation set achieved by the temporal input memory (TIM) and the temporal feature memory (TFM), compared to the single frame backbone.

Initially, the backbone is trained on single frames with a batch size of 16 and learning rate of 0.001, which decays by $e^{-5 \cdot 10^{-5} \cdot i}$. The BEV grid has a resolution of $[480, 360, 32]$. Using the pretrained backbone, the overall architecture is trained with the same batch size and learning rate and follows the temporal training proposed by [10].

4.3 Temporal BEV Segmentation

In order to evaluate the benefits of the presented recurrent temporal approach, the improvements achieved by the individual components are investigated, with the results shown in Table 4.1. The BEV backbone, which is also considered as baseline, achieves a mIoU of 58.0%. Temporal fusion at the input stage with the presented temporal input memory (TIM) improves the results to 58.7%. The fusion on feature stage has an even greater impact and considerably improves the segmentation results to a mIoU of 64.5%. Noticeably, temporal fusion of deep feature maps computed by a CNN backbone exploits temporal information and dependencies more effective than an early fusion of the input point clouds. Nevertheless, combining both stages achieves the best results and obtains an overall improvement of $+6.7\%$ in terms of mIoU.
For a more detailed analysis, the results of the individual classes are investigated and compared to the baseline, depicted in Table 4.2. Static classes are constantly improved, like fence with $+11.2\%$, and even classes with already high values benefit from temporal information. In addition, dynamic classes are considerably

Approach	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist
Backbone	58.0	93.6	43.5	60.0	54.0	42.1	56.3	73.4	10.4
TemporalBEV	**64.7**	**95.3**	**46.5**	**76.3**	**54.3**	**66.3**	**70.4**	**76.8**	**44.2**

Approach	road	sidewalk	parking	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic sign
Backbone	92.2	77.9	43.3	1.2	89.0	47.0	85.6	60.0	72.7	57.7	42.9
TemporalBEV	**93.6**	**79.0**	**43.5**	**1.4**	**91.1**	**58.2**	**86.9**	**65.2**	**74.1**	**60.2**	**46.9**

Table 4.2: Results for the individual classes on the validation set of SemanticKITTI. The temporal approach outperforms the single frame backbone for every class. Values are given as IoU (%).

improved as well, especially motorcycle, other-vehicle, person and motorcyclist. This requires a deeper investigation, because movement of dynamic objects can cause alignment errors and complicates the correct temporal association. However, only fast movement causes noticeable errors, so solely a few fast moving dynamic objects do not benefit, the majority of the dynamic objects significantly benefits from the temporal information.

5 Conclusion

In this work, we presented an efficient recurrent temporal architecture for semantic segmentation of 3D point clouds relying on BEV representation. Temporal information and dependencies are exploited twice, at the input as well as feature stage. Point clouds of the last frames are aggregated to improve the information provided to the backbone. The feature maps of the backbone are then used to update a temporal feature memory, which contains the aggregated and fused features from the current frame and the past. Based on these enhanced features an improved semantic segmentation is predicted. The evaluation showed a considerable improvement achieved by the usage of temporal information and

as a result the presented approach outperforms the single frame baseline by a large margin and also for every individual class.

References

[1] Iro Armeni et al. “3D Semantic Parsing of Large-Scale Indoor Spaces”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[2] Nicolas Ballas et al. “Delving Deeper into Convolutional Networks for Learning Video Representations”. In: *arXiv* (Nov. 2016). arXiv: 1511.06432v4.

[3] Jens Behley et al. “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019.

[4] Maxim Berman, Amal Rannen Triki, and Matthew B. Blaschko. “The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks”. In: May 2018, pp. 4413–4421. eprint: 1705.08790v2.

[5] Holger Caesar et al. “nuScenes: A Multimodal Dataset for Autonomous Driving”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[6] X. Chen et al. “Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data”. In: *IEEE Robotics and Automation Letters(RA-L)* (2021).

[7] Ran Cheng et al. “(AF)2-S3Net: Attentive Feature Fusion With Adaptive Feature Selection for Sparse Semantic Segmentation Network”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 12547–12556.

[8] Christopher Choy, JunYoung Gwak, and Silvio Savarese. “4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Apr. 2019, pp. 3075–3084.

[9] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. "SalsaNext: Fast, Uncertainty-aware Semantic Segmentation of LiDAR Point Clouds for Autonomous Driving". In: *International Symposium on Visual Computing (ISVC)* (Mar. 2020), pp. 207–222.

[10] Fabian Duerr et al. "LiDAR-based Recurrent 3D Semantic Segmentation with Temporal Memory Alignment". In: *International Conference on 3D Vision (3DV)* (2020), pp. 781–790.

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? The KITTI vision benchmark suite". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[12] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. "3D Semantic Segmentation with Submanifold Sparse Convolutional Networks". In: 2018, pp. 9224–9232.

[13] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[14] Qingyong Hu et al. "RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[15] Rui Huang et al. "An LSTM Approach to Temporal 3D Object Detection in LiDAR Point Clouds". In: *IEEE European Conference on Computer Vision (ECCV)*. 2020.

[16] Yangyan Li et al. "PointCNN: Convolution On $\mathcal{X}$-Transformed Points". In: *Advances in Neural Information Processing Systems*. 2018.

[17] Wenjie Luo, Binh Yang, and Raquel Urtasun. "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 3569–3577.

[18] Scott McCrae and Avideh Zakhor. "3d Object Detection For Autonomous Driving Using Temporal Lidar Data". In: *IEEE International Conference on Image Processing (ICIP)*. IEEE. 2020, pp. 2661–2665.

[19] Gregory P. Meyer et al. “LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[20] A. Milioto et al. “RangeNet++: Fast and Accurate LiDAR Semantic Segmentation”. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2019.

[21] Charles Ruizhongtai Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[22] Charles Ruizhongtai Qi et al. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *Advances in Neural Information Processing Systems*. 2017.

[23] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. “OctNet: Learning Deep 3D Representations at High Resolutions”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[24] Ahmad El Sallab et al. “YOLO4D: A Spatio-temporal Approach for Real-time Multi-object Detection and Classification from LiDAR Point Clouds”. In: *NIPS Workshop MLITS*. 2018.

[25] Hanyu Shi et al. “SpSequenceNet: Semantic Segmentation Network on 4D Point Clouds”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR*. 2020.

[26] Lyne P. Tchapmi et al. “SEGCloud: Semantic Segmentation of 3D Point Clouds”. In: *International Conference on 3D Vision (3DV)*. 2017.

[27] Hugues Thomas et al. “KPConv: Flexible and Deformable Convolution for Point Clouds”. In: *IEEE International Conference on Computer Vision (ICCV)*. 2019.

[28] Shenlong Wang et al. “Deep Parametric Continuous Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[29] Jun Xie et al. “Semantic Instance Annotation of Street Scenes by 3D to 2D Label Transfer”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[30] Chenfeng Xu et al. "SqueezeSegV3: Spatially-Adaptive Convolution for Efficient Point-Cloud Segmentation". In: *European Conference on Computer Vision (ECCV)*. 2020.

[31] Junbo Yin et al. "LiDAR-based Online 3D Video Object Detection with Graph-based Message Passing and Spatiotemporal Transformer Attention". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Apr. 2020, pp. 11492–11501.

[32] Fisher Yu et al. "Deep Layer Aggregation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[33] Chris Zhang, Wenjie Luo, and Raquel Urtasun. "Efficient Convolutions for Real-Time Semantic Segmentation of 3D Point Clouds". In: *International Conference on 3D Vision (3DV)*. 2018.

[34] Yang Zhang et al. "PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.