Seppanen, Alvari; Ojala, Risto; Tammi, Kari

4DenoiseNet: Adverse Weather Denoising From Adjacent Point Clouds

Published in:
IEEE Robotics and Automation Letters

DOI:
10.1109/LRA.2022.3227863

Published: 01/01/2023

Document Version
Publisher's PDF, also known as Version of record

4DenoiseNet: Adverse Weather Denoising From Adjacent Point Clouds

Alvari Seppänen, Risto Ojala, and Kari Tammi

***Abstract*—Reliable point cloud data is essential for perception tasks e.g. in robotics and autonomous driving applications. Adverse weather causes a specific type of noise to light detection and ranging (LiDAR) sensor data, which degrades the quality of the point clouds significantly. To address this issue, this letter presents a novel point cloud adverse weather denoising deep learning algorithm (4DenoiseNet). Our algorithm takes advantage of the time dimension unlike deep learning adverse weather denoising methods in the literature. It performs about 10% better in terms of intersection over union metric compared to the previous work and is more computationally efficient. These results are achieved on our novel SnowyKITTI dataset, which has over 40000 adverse weather annotated point clouds. Moreover, strong qualitative results on the Canadian Adverse Driving Conditions dataset indicate good generalizability to domain shifts and to different sensor intrinsics.**

***Index Terms*—AI-based methods, computer vision for transportation, deep learning for visual perception, intelligent transportation systems, visual learning.**

I. Introduction

ADVERSE weather conditions can have a huge impact on light detection and ranging (LiDAR) sensor data. Airborne particles, such as rain droplets [1], [2], [3], fog [1], [3], [4], [5], or snowflakes [3], [6], [7], [8], [9] cause undesired reflections, refractions, and absorptions of the laser, which results in missing and cluttered points. This is a major problem since point clouds are often used for determining the open volume of the environment, e.g. autonomous robots use point clouds for obstacle avoidance. Moreover, the clutter also affects other downstream perception algorithms, such as object detection [10], [11], [12], [13], which are an essential component of autonomous road vehicles. Thus, robust perception data in adverse weather is crucial as fatality rates for human drivers are notably higher in such conditions, as reported by the European Commission [14] and the US Department of Transportation [15].

Our task is to remove points from LiDAR point clouds that are caused by airborne particles. The motivation for this is to provide clean point cloud data for all downstream tasks e.g. mapping, localization, object detection, and navigation. Previous work uses either a classical approach [16], [17], [18], [19], [20] or a learned approach [21]. Unlike previous work, our work utilizes spatial-temporal data as an input to a neural network. We do this by feeding adjacent point clouds $\mathbf{P}^{(t)} \in \mathbb{R}^{n\times 3}$ and $\mathbf{P}^{(t-1)} \in \mathbb{R}^{n\times 3}$ to a novel neural network that predicts the points that are caused by airborne particles. Then the predictions are used for removing these points from the point cloud, yielding a clean point cloud $\mathbf{P}^{(t)'}$. The neural network architecture can be optimized for multiple types of adverse weather e.g. rain, fog, and snowfall. Moreover, our algorithm is also tested with more challenging clutter caused by light, medium, and heavy snowfall, whereas previous work [21] was tested only in fog and rain. We obtained better performance than WeatherNet [21] and general-purpose state-of-the-art semantic segmentation networks [22], [23]. Furthermore, the experiments show that our model generalizes better to other adverse weather conditions.

The problem of removing noise caused by adverse weather is not trivial because of the nature of the LiDAR sensor, the sparsity of the point cloud, occlusions caused by the noise, and the varying density of the noise. This leads to statistical or hard-coded filters [16], [17], [18], [19], [20] to remove valid points and preserve the clutter. To address this problem, we use a deep learning architecture that learns an equivariance function. A deep learning method is also able to utilize the complex patterns of adverse weather noise. Due to the nature of the LiDAR sensor, the reflectance and position of solid structures affect the pattern of the noise. Our method is trained with partly simulated semi-synthetic data. Since labels are "free lunch" from the simulation, the training set can be built effortlessly, and our model can be trained in a supervised manner. However, we want to emphasize that our model is tested quantitatively with real data captured in adverse weather. The tests indicate that our model generalizes to real data and also to different sensor intrinsics.

We show that the key to robust adverse weather denoising is the efficient utilization of spatial-temporal data. Spatial information, in metric space, is useful because clutter has a relatively low density. Temporal information is crucial because valid points follow predictable trajectories. Contrarily, noise points caused by airborne particles have a chaotic nature. Our architecture exploits these phenomena by a k-nearest neighbor convolution kernel that captures spatial-temporal information, and with a motion-guided attention mechanism.

Our contribution is two-fold.

- We present the first deep learning approach for LiDAR adverse weather denoising utilizing spatial and temporal

Manuscript received 15 September 2022; accepted 24 November 2022. Date of publication 8 December 2022; date of current version 14 December 2022. This letter was recommended for publication by Associate Editor B. Thornton and Editor C. C. Lerma upon evaluation of the reviewers' comments. This work was supported in part by Henry Ford Foundation Finland and in part by the Helsinki Institute of Physics. *(Corresponding author: Alvari Seppänen.)*

Alvari Seppänen and Kari Tammi are with the Aalto University, 02150 Espoo, Finland, and also with the University of Helsinki, 00100 Helsinki, Finland (e-mail: alvari.seppanen@aalto.fi; kari.tammi@aalto.fi).

Risto Ojala is with the Aalto University, 02150 Espoo, Finland (e-mail: risto.j.ojala@aalto.fi).

Digital Object Identifier 10.1109/LRA.2022.3227863

information. This is realized with a novel k-nearest neighbors search convolution on consecutive point clouds, which captures spatial and temporal information. We surpass existing methods in performance by a large margin, with a lower computational cost.

- Since point-wise annotations are laborious, we train with semi-synthetic data generated by a highly realistic physics-based model [12]. That is, synthetic effects of adverse weather are added to real point clouds captured in clear weather. We are the first to use this data to train a model and test its performance on real data, captured in adverse weather. The excellent performance indicates that our model is robust in this domain shift. Given the excellent performance, our model will be an essential component in outdoor LiDAR sensor applications, enabling clean perception data for all downstream tasks. Moreover, we present the first point-wise annotated adverse weather dataset, i.e. SnowyKITTI, based on this simulation model which has approximately 40000 LiDAR scans.

II. Related Work

Adverse weather denoising from sparse LiDAR point clouds is an emerging field, and only a handful of studies have been conducted. Dense point cloud denoising is a more established field but the methods have not been tested for denoising adverse weather in sparse point clouds. Therefore, we do not cover that area of research.

A. Classical Approaches

Typically, the clutter caused by adverse weather has a relatively low density. Radius outlier removal (ROR) and statistical outlier removal (SOR), presented in [24], remove outliers based on local density. ROR removes points that do not have another point in distance r. SOR computes the mean distance to k nearest neighbors and decides if a point is an outlier based on the global mean distance between the points and the standard deviation. These methods do not work well with point clouds that have inherently varying density, e.g. LiDAR point clouds whose density is proportional to the measured range. This leads to the removal of distant points, and therefore these methods are not suited for adverse weather denoising.

Charron et al. [16] proposed dynamic radius outlier removal (DROR) which adjusts the distance r based on the range of the point from the sensor. They achieved good results in removing points caused by snowfall. Dynamic statistical outlier removal (DSOR) [17] combines SOR and DROR. It removes outliers based on a threshold that is defined by the global mean distance between the points and the standard deviation and the measured range of the point. Low-intensity outlier removal (LIOR) [18] is designed to remove points caused by snowfall. The threshold is derived from LiDAR-measured intensity, and it is a function of the measured range. LIOR includes the ROR filter for preserving points that have low intensity but high density. That is, LIOR takes advantage of the typical low intensity and density of airborne snowflakes. However, valid points are removed from light-absorbent and semi-transparent surfaces. Dynamic distance–intensity outlier removal (DDIOR) [19] fuses DSOR and LIOR to achieve better performance. Li et al. utilized spatial-temporal features for removing points caused by snowfall [20]. They showed that temporal information is valuable in this task by performing well compared to other methods. Their method thresholds points to relative distances in W-T-space where W and T denote spatial dimensions and time, respectively.

B. Learned Approaches

Since our method is based on segmentation with a neural network, we present related work from this area. LiDAR point cloud semantic segmentation networks are focusing on general segmentation, and they can be trained to segment highly abstract shapes. Most successful approaches use voxelized [23], [25], bird's eye view [26], or spherical projection input [27], [28], [29], [30]. However, raw point input approaches exist as well [31], [32], [33], [34], [35], [36], [37], [38] Cylinder3D [23] parts the point cloud into cylindrical voxels. Then a 3D convolutional neural network extracts features and produces the predictions. PolarNet [26] uses a bird's eye view pseudo image representation for feature extraction. Spherical projection image input is beneficial for memory usage because of the dense representation [22], [39]. Another benefit of the projection image is that a 2D convolutional neural network can be used for feature extraction. However, 2D convolution kernels fail to capture local spatial information accurately due to the nature of the projection. Work by Xu et al. combines projection, voxel, and point inputs to produce more accurate segmentation results [40].

The semantic segmentation networks are well-proven and general-purpose. Thus, they can be trained for segmenting the clutter caused by adverse weather. However, they require a lot of labeled training data, memory, and computational power due to the enormous amount of trainable parameters. Recent work by Heinzler et al. [21] proposed the WeatherNet, an optimized semantic segmentation network for segmenting the clutter caused by fog and rain. Their method has a significantly lower amount of trainable parameters while having equal or even better performance compared to the state-of-the-art general-purpose segmentation networks. While their work is focused on segmenting the clutter caused by rain and fog, our method presents a novel spatial-temporal feature encoder and benchmarks the performance of light, medium, and heavy snowfall, which causes more severe clutter. Furthermore, our method has only 0.6 M trainable parameters while theirs has 1.5 M. Therefore, our method generally requires less labeled data and generalizes better.

III. Methods

A. Ordered Point Cloud Representation

Point coordinates from a typical LiDAR sensor $\vec{c} = (x, y, z)$ are mapped $\Gamma : \mathbb{R}^{n\times 3} \rightarrow \mathbb{R}^{s_h \times s_w \times 3}$ to spherical coordinates, and finally to image coordinates, as defined by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1/2(1 - \tan^{-1}(yx^{-1})\pi^{-1})s_w \\ (1 - (\sin^{-1}(z||\vec{c}||^{-1}) + f_{vup})f_v^{-1})s_h \end{bmatrix} \quad (1)$$

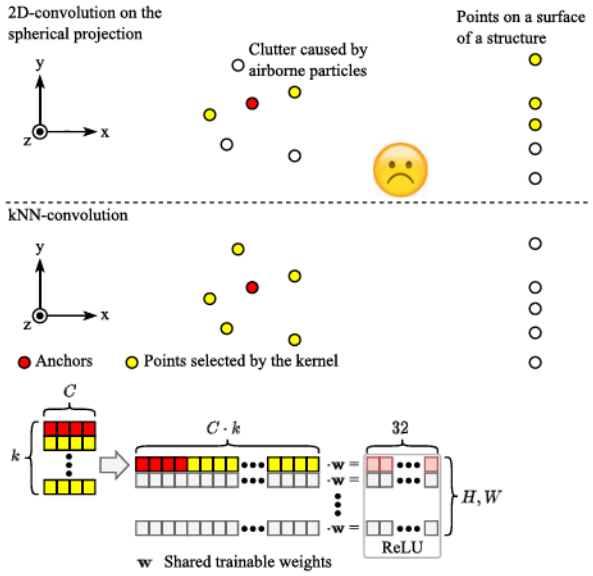

Fig. 1. A highlight of an issue of a 2D-convolution on a spherical projection image. It fails to capture local points in metric space, whereas kNN-convolution captures local points, which is important in our task as the clutter is not continuous on the projection image.

where (s_h, s_w) are the height and width of the desired projection image representation, f_v is the total vertical field-of-view of the sensor, and f_{vup} is the vertical field-of-view spanning upwards from the horizontal origin plane. The resulting list of image coordinates is used to construct a (x, y, z)-channel image i.e. ordered point cloud $\mathbf{P}_o \in \mathbb{R}^{s_h \times s_w \times (3+C_f)}$, where C_f denotes the number of feature channels, in our case, $C_f = 1$ for intensity.

B. Utilization of Spatial Information

Classical methods DROR [16], DSOR [17], and DDIOR [19] show that local point density is a useful indicator for determining if a given point is caused by an airborne particle. Therefore, enabling the neural network to capture this information is crucial. Since our method is projection-based and a traditional convolution fails to capture local points [41], a measure has to be taken. We define the first convolution layer to capture k-nearest neighbors (kNN) in metric space via kNN-convolution. An illustrative schematic is presented in Fig. 1. This convolution kernel considers the closest k points in the metric space instead of the neighboring points based on the pixel coordinates, enabling better spatial information for the network. To mitigate the computational burden of the kNN-search, we search only in the neighboring area of the anchor point in the ordered point cloud. We show an improvement compared to a traditional convolution in an ablation study (Section IV-C). For simplicity, a pixel coordinate is denoted as

$$\vec{p} = (u, v). \tag{2}$$

The spatial-kNN-convolution becomes

$$\mathbf{\Theta}(\vec{p}) = \mathbf{w} * \mathbf{P}_o^{(t)} = \sum_{\partial\vec{p}=0}^{k} \mathbf{w}(\partial\vec{p}) \cdot \mathbf{P}_o^{(t)}(\psi(\vec{p}))(\partial\vec{p}) \tag{3}$$

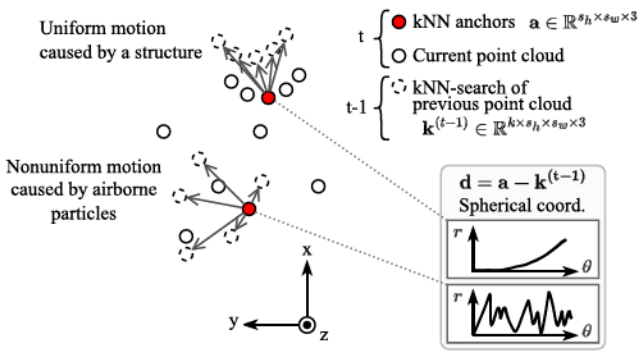

Fig. 2. The Temporal-kNN-kernel capturing temporal information. Distributions given by $\mathbf{d} = \mathbf{a} - \mathbf{k}^{(t-1)}$, in the spherical coordinate system, are smoother for a set of points with a uniform motion. On the contrary, a set of points caused by a nonuniform motion are more random.

where $\mathbf{w}$ denotes trainable weights and $\psi(\vec{p})$ is defined as

$$\psi(\vec{p}) = argmink\left(|\mathbf{P}_{or}^{(t)}\left(\vec{p} - \vec{\xi}, \ldots, \vec{p} + \vec{\xi}\right) - \mathbf{P}_{or}^{(t)}(\vec{p})|\right) \tag{4}$$

where $argmink(\cdot)$ returns indicies of minimum-k elements, and $\mathbf{P}_{or}^{(t)}$ denotes the range channel. $\vec{\xi}$ is a hyperparameter that defines the number of elements that are considered in the kNN-search. Gathered kernel inputs are activated with a $ReLU(\mathbf{\Theta}(\vec{p}))$ function.

C. Utilization of Temporal Information

The effects of adverse weather in LiDAR point clouds have a more chaotic nature than valid points. This is caused by a reflection of the beam from an airborne particle. The reflections caused by these particles are more unpredictable since they are small and are moved by turbulent airflow. This chaotic behavior is on a much smaller scale in other points. Thus, temporal information can be utilized in our task. An empirical study shows that a reflection from an airborne particle, e.g. snowflake, is extremely unlikely to be occurring twice in the same place. That is, a single beam reflected from an airborne particle is highly unlikely to occur in the adjacent scan for a given beam, whereas reflections of other surfaces are more predictable.

As illustrated in Fig. 2, we capture temporal information by searching for a kNN-set of points from a previous point cloud $\mathbf{P}^{(t-1)}$ using anchor points $\mathbf{a} \in \mathbb{R}^{s_h \times s_w \times 3}$ i.e. the Cartesian channels of the current point cloud $\mathbf{P}^{(t)}$. Similarly to the $\mathbf{P}^{(t)}$ kNN-search, the search considers only the neighboring area in the ordered point cloud. The temporal-kNN-convolution is defined as

$$\begin{aligned}\mathbf{\Delta}(\vec{p}) &= \mathbf{w}_\Delta * \mathbf{d} \\ &= \sum_{\partial\vec{p}=0}^{k} \mathbf{w}_\Delta(\partial\vec{p}) \cdot (\mathbf{a}(\vec{p}) - \mathbf{P}_o^{(t-1)}(\psi_\Delta(\vec{p})))(\partial\vec{p})\end{aligned} \tag{5}$$

where $\psi_\Delta(\vec{p})$ is defined as

$$\psi_\Delta(\vec{p}) = argmink\left(\left|\mathbf{P}_{or}^{(t-1)}(\vec{p} - \vec{\xi}, \ldots, \vec{p} + \vec{\xi}) - \mathbf{P}_{or}^{(t)}(\vec{p})\right|\right). \tag{6}$$

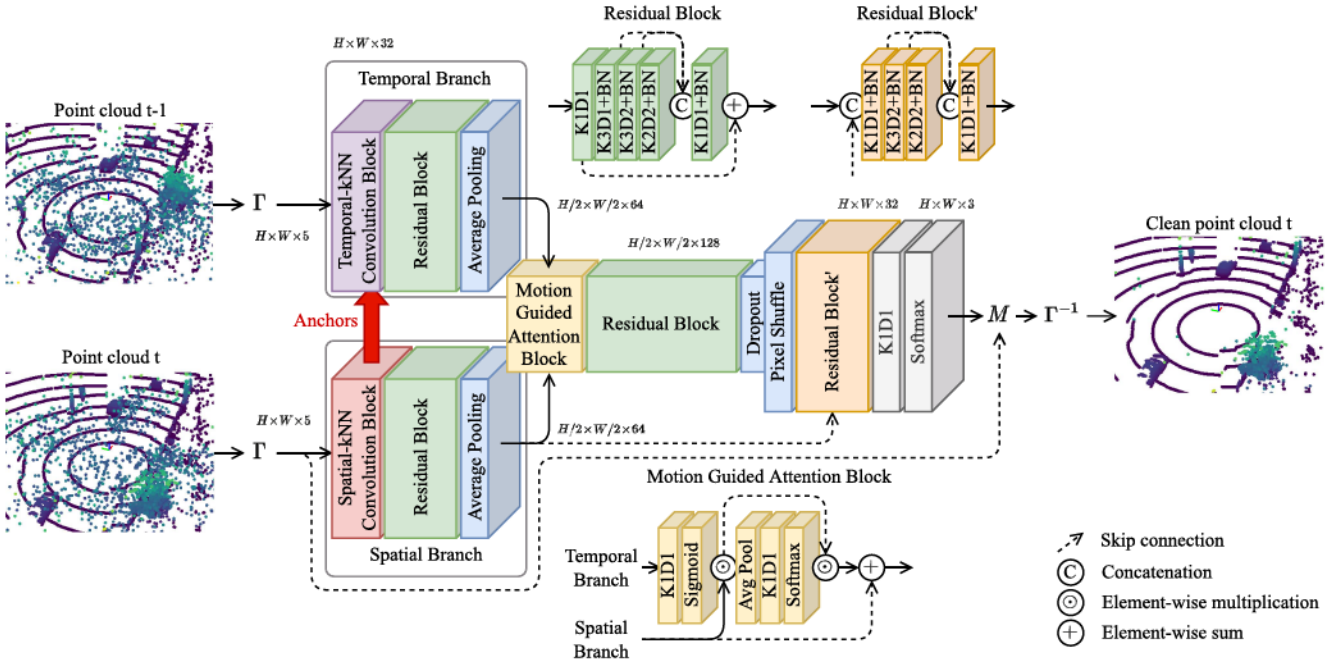

Fig. 3. 4DenoiseNet architecture. K, D, and BN indicate the kernel size, dilation, and batch normalization, respectively.

$\mathbf{d} \in \mathbb{R}^{k \times s_h \times s_w \times 3}$ is an approximation of the motion of a local manifold. $\mathbf{d}$ is converted from Cartesian into a spherical coordinate system to have a more relevant representation of the data, i.e. r and (θ, ϕ) are the magnitude and the direction of the motion, respectively,

$$\Gamma_{\mathbf{d}} : \begin{cases} \mathbb{R}^3 \to \mathbb{R}^3 \\ (x, y, z) \mapsto (r, \theta, \phi), \end{cases} \tag{7}$$

this is done before convolving with w_{Δ}. Similarly to the spatial-kNN-convolution, kernel output is activated with a $ReLU(\mathbf{\Delta}(\vec{p}))$ function.

D. Neural Network Design and Training

Based on the above insights, a function that captures both spatial and temporal features should have superior performance over its counterparts. We approximate this function by a novel neural network, namely 4DenoiseNet (Fig. 3), where 4D is a reference to time as the fourth dimension. It has two branches: spatial and temporal. The spatial branch processes the spatial features of the current point cloud $\mathbf{P}^{(t)}$, and the temporal branch processes the temporal features of the previous point cloud $\mathbf{P}^{(t-1)}$. The spatial branch has a kNN-convolution block that captures kNN for each point. The kNN-search is also computed in the temporal branch. However, anchors are provided by the spatial branch and subtracted from the kNN points of the previous point cloud $\mathbf{P}^{(t-1)}$. This captures temporal information as illustrated in Fig. 2. Both branches have a residual block for encoding, inspired by the work of Cortinhal et al. [22] and Aksoy et al. [42]. The branches are fused by the motion-guided attention (MGA) block, which fuses temporal features with spatial features using the motion-guided attention mechanism, described in [43]. After another residual block, a pixel shuffle decreases channel dimension and increases spatial dimensions. The residual block' connects the skip connection from the spatial branch with the main pipeline. A standard convolution and Softmax normalization layers give us the desired logits. Finally, a mask module M removes points caused by airborne particles using the Softmax confidences, giving us a clean point cloud $\mathbf{P}^{(t)'}$.

Training objective is a standard cross-entropy and Lovász-Softmax loss [44], which optimizes for the Jaccard index [45] i.e. intersection over union (IoU) metric

$$\mathcal{L}_{ls} = \frac{1}{|C|} \sum_{c \in C} \overline{\Delta_{J_c}}(m(c)) \qquad \text{and}$$

$$m_i(c) = \begin{cases} 1 - x_i(c), & \text{if } c = y_i(c) \\ x_i(c), & \text{otherwise.} \end{cases} \tag{8}$$

where the number of classes in denoted with C, $\overline{\Delta_{J_c}}$ defines the Lovász extension of the Jaccard index, $y_i(c) \in \{0, 1\}$ and $x_i(c) \in [0, 1]$ hold the ground truth label and the prediction of pixel i for class c, respectively. The complete loss function is formulated as follows

$$\mathcal{L} = \mathcal{L}_{ls} + \mathcal{L}_{ce} \tag{9}$$

where $\mathcal{L}_{ce}$ denotes the standard cross-entropy loss.

IV. Experiments

A. Experimental Setup

Datasets: We conduct qualitative tests on the Canadian Adverse Driving Conditions dataset [46]. It is captured in real-world adverse weather conditions which include light, medium, heavy, and extreme snowfall. Since this dataset does not have

TABLE I
DEFINITIONS OF DIFFERENT ADVERSE WEATHER CONDITIONS AND TRAINING SUBSETS

Classification	Light	Medium	Heavy
Snowfall rate	$[0.5, 1.5[$	$[1.5, 2.5[$	$[2.5, 3.0]$
Terminal velocity	$[1.0, 2.0]$	$[1.0, 2.0]$	$[1.0, 2.0]$
All	✓	✓	✓
Subset 1	✓	✓	-
Subset 2	✓	-	✓
Subset 3	-	✓	✓
Subset 4	-	-	✓
Subset 5	-	✓	-
Subset 6	✓	-	-

TABLE II
TRAINED WITH ALL CONDITIONS. GPU: NVIDIA GTX 1060, CPU: 4.0 GHZ INTEL I5-7600 K 5TH GENERATION

Method	Light IoU	Medium IoU	Heavy IoU	Runtime ms	Param. $\cdot 10^6$
DROR* [16]	0.442	0.445	0.437	120	10^{-5}
LIOR* [18]	0.445	0.442	0.430	120	10^{-5}
SalsaNext [22]	0.947	0.947	0.951	26	8.8
Cylinder3D [23]	0.949	0.943	0.942	65	41.7
WeatherNet [21]	0.884	0.889	0.865	44	1.5
4DenoiseNet	**0.975**	**0.976**	**0.977**	**19**	**0.6**

* - No training required. Bolded font indicates the best values.

point-wise annotations, we train our model on our SnowyKITTI dataset. SnowyKITTI is a modified KITTI odometry benchmark dataset [47] with synthetic snowfall created with a highly realistic physics-based simulation model presented in [12]. SnowyKITTI is divided into training, validation, and testing splits with a ratio of 40/10/50. We also evaluate our model with the SnowyKITTI testing set. Testing and validation sets are different sequences compared to the training set. Furthermore, the training set is divided into subsets depending on the parameters of the snowfall simulation. The subsets are described in Table I. This is done to investigate the generalizability of the models. That is, a network is trained only with e.g. light snowfall conditions and tested in all conditions. We have created a diverse selection of training subsets that validate robust performance across a wide range of adverse weather conditions.

Evaluation metrics: We use the Jaccard index [45] i.e. IoU, which is formulated as follows

$$J_c(\mathbf{y}, \mathbf{x}) = \frac{|\{\mathbf{y} = c\} \cap \{\mathbf{x} = c\}|}{|\{\mathbf{y} = c\} \cup \{\mathbf{x} = c\}|}. \tag{10}$$

where $\mathbf{y}$ and $\mathbf{x}$ denote ground truth and predicted labels, respectively. We are only interested in the IoU value of the noise class. Thus, from here forward IoU is for noise class.

Training and inference details: The parameters of the network are optimized with the Adam optimizer [48]. We fix the spatial dropout probability to 0.2. The initial learning rate is set to 0.01 which is decayed by 0.01 after each epoch. An L2 penalty is applied with $\lambda = 1 \cdot 10^4$ and a momentum of 0.9. Data is augmented by randomly dropping points before creating the projection, applying a random translation and rotation, and flipping randomly relative to the y-axis. These augmentation methods are applied independently of each other with a probability of 0.5. Inference runtime is benchmarked with an Nvidia GTX 1060 GPU.

The code and the data are made publically available here.[1]

B. Quantitative Results

The main results are presented in Table II. Our 4DenoiseNet is compared with the other adverse weather filtering model WeatherNet [21], and with general-purpose state-of-the-art semantic segmentation networks [22], [23]. The models are evaluated in terms of Jaccard index, runtime, and parameter count in Light, Medium, and Heavy snowfall. Furthermore, the models are trained with different subsets (Table I) to investigate generalizability to conditions differing from training conditions. All learned models in Table II are trained and tested with all conditions, albeit training and testing do not share any sequences. The training set is 80% of the size of the testing set as described in Section IV-A. Overall our model performs the best in both accuracy and runtime. Moreover, our model has fewer trainable parameters compared to the other learned models. There is a large gap in accuracy between classical DROR and LIOR compared to learned approaches, highlighting the difficulty of this task.

Fig. 4 presents the performance when models are trained with different training subsets. It should be noted, that the testing sets are separate from the training sets. In the figure, the x-axis presents which training set was used. Most notably, our model is the most robust as there is a smaller variance in IoU compared to other models. When trained with Subset2, SalsaNext is on par with our model. We hypothesize that it is more unlikely to overfit with Subset2 because it has more variance in conditions. This hypothesis is supported by the performance with Subset5 where the performance of SalsaNext is lower. When the other models are trained with all conditions, their performance is lower, this might be caused by conditions imbalance of the dataset, which causes the models to overfit to the most frequent condition. However, our model performs the best when trained with all conditions in Medium and Heavy snowfall.

C. Ablation Study

To study the importance of spatial-temporal kNN-convolution, we conducted an ablation study where four 4DenoiseNet variants were compared. Table III describes the variants and their performance, runtime, and the number of trainable parameters. The kNN-convolution module was replaced with a traditional 2D convolution layer. In the variants without the temporal branch, MGA is not used, and the spatial branch connects straight to the second residual block. Based on the ablation study, the spatial kNN-convolution increases the performance slightly more than the temporal kNN-convolution. Overall, both spatial and temporal kNN-convolution modules improved the performance significantly.

[1] [Online]. Available: https://github.com/alvariseppanen/4DenoiseNet

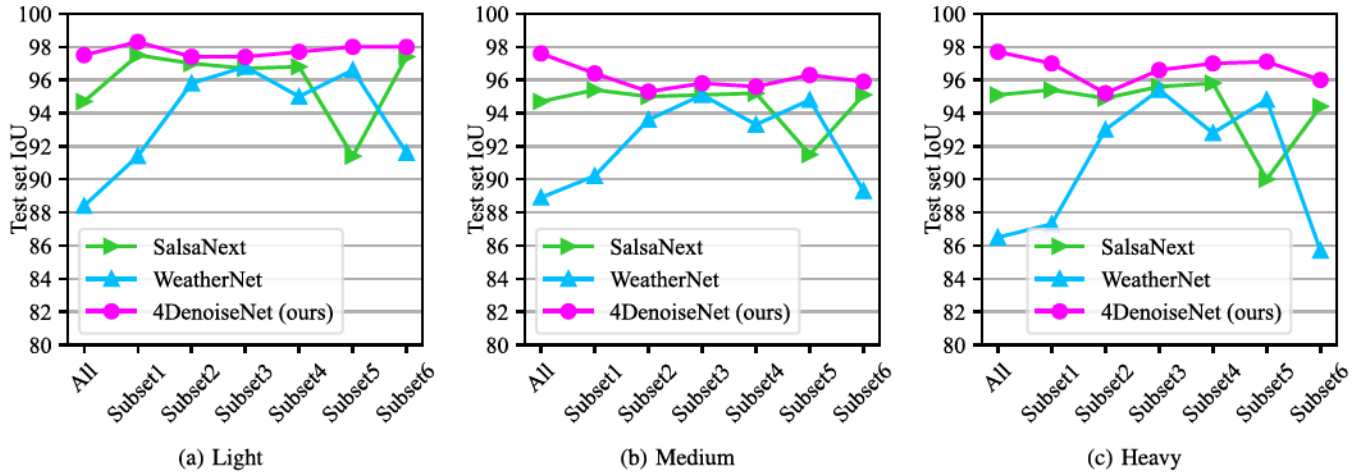

Fig. 4. Quantitative results on the Light, Medium, and Heavy test sets with different training sets (Table I). The performance of 4DenoiseNet remains high regardless of the training dataset.

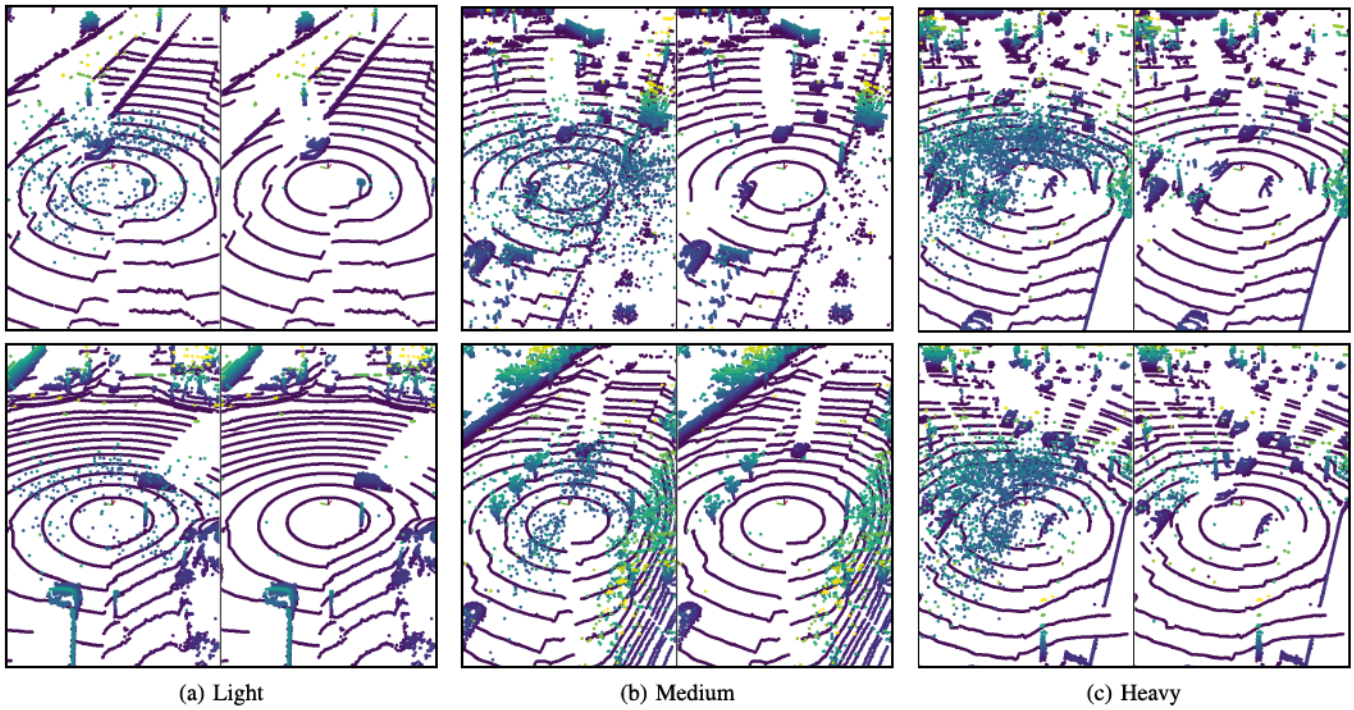

Fig. 5. Denoising performance on 6 individual sample point clouds from real snowfall. Raw input and denoised output are on the left and right sides of each image pair, respectively. We suggest zooming in for better detail.

D. Qualitative Results and Discussion

We conducted qualitative tests on real LiDAR data captured in adverse weather. The data is from the Canadian Adverse Driving Conditions dataset [46]. The performance was analyzed in snowfall as it causes more noise to the point cloud compared to other conditions. Fig. 5 illustrates the denoising performance. The reader should focus on the framed image pairs, where the image on the left side is the raw input, and the image on the right side is the denoised output. In medium and heavy snowfall, some objects are not visible in the input point cloud, but after applying our algorithm they are clearly visible. Based on visual analysis, our model does not seem to remove valid points albeit being sparse (see the first Medium pair).

To further highlight the challenge of this task, the LiDAR used from this dataset has different intrinsic parameters compared to the LiDAR used for training. The training LiDAR has a vertical resolution of 0.44°. The testing LiDAR has a vertical resolution of 1.25°. This means that the training point clouds are denser compared to the testing point clouds. Furthermore, the performance on the domain shift improves when the intensity channel is omitted. The trained models that we tested (Table II)

TABLE III
ABLATIONS OF DIFFERENT MODULES AND THEIR CONTRIBUTION TO THE IOU AND RUNTIME

First conv.	$\mathbf{P}_{t-1}$	Light IoU	Medium IoU	Heavy IoU	Runtime ms	Param. $\cdot 10^3$
2D	-	0.599	0.566	0.562	8	477.84
2D	✓	0.676	0.654	0.666	13	568.92
kNN	-	0.882	0.864	0.866	16	480.53
kNN	✓	0.975	0.976	0.977	19	571.61

Our KNN-convolution module is replaced with a traditional 2-dimensional convolution (first conv. 2D) on the ordered point cloud. ✓Indicates that previous point cloud P_{t-1} and the temporal branch are used.

get biased on the intensity. Therefore, we omitted the intensity channel and trained our model with (r, x, y, z)-input channels. After this, the model generalizes well to the real data. Note that the intensity channel was not omitted in the quantitative experiments.

V. CONCLUSION

We presented 4DenoiseNet, the first deep learning algorithm for adverse weather denoising on adjacent LiDAR point clouds. Quantitive results on our annotated SnowyKITTI dataset, qualitative results on the Canadian Adverse Driving Conditions dataset, and runtime indicate that our model is the new state-of-the-art on adverse weather denoising. 4DenoiseNet is based on the spatial-temporal kNN-convolution module that is presented in this work. We show via ablation study the importance of spatial and temporal information in metric space in adverse weather denoising tasks. Given the light computational demand and the performance, our algorithm will be an essential component in outdoor LiDAR applications, enabling clean point cloud data for all downstream tasks. For more qualitative results we refer to our repository page 1.

REFERENCES

[1] A. M. Wallace, A. Halimi, and G. S. Buller, "Full waveform LiDAR for adverse weather conditions," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7064–7077, Jul. 2020.

[2] C. Goodin, D. Carruth, M. Doude, and C. Hudson, "Predicting the influence of rain on LiDAR in ADAS," *Electronics*, vol. 8, no. 1, 2019, Art. no. 89.

[3] M. Bijelic et al., "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11682–11692.

[4] M. Bijelic, T. Gruber, and W. Ritter, "A benchmark for lidar sensors in fog: Is detection breaking down?," in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 760–767.

[5] R. Heinzler, P. Schindler, J. Seekircher, W. Ritter, and W. Stork, "Weather influence and classification with automotive lidar sensors," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2019, pp. 1527–1534.

[6] M. Jokela, M. Kutila, and P. Pyykönen, "Testing and validation of automotive point-cloud sensors in adverse weather conditions," *Appl. Sci.*, vol. 9, no. 11, 2019, Art. no. 2341.

[7] M. Kutila, P. Pyykönen, M. Jokela, T. Gruber, M. Bijelic, and W. Ritter, "Benchmarking automotive LiDAR performance in arctic conditions," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst.*, 2020, pp. 1–8.

[8] S. Michaud, J.-F. Lalonde, and P. Giguere, "Towards characterizing the behavior of LiDARs in snowy conditions," in *Proc. Int. Conf. Intell. Robots Syst.*, 2015, pp. 1–6.

[9] W. H. O'Brien, "Visibility and light attenuation in falling snow," *J. Appl. Meteorol. Climatol.*, vol. 9, no. 4, pp. 671–683, 1970.

[10] A. T. Do and M. Yoo, "LossDistillNet: 3D object detection in point cloud under harsh weather conditions," *IEEE Access*, vol. 10, pp. 84882–84893, 2022.

[11] A. D. The and M. Yoo, "MissVoxelNet: 3D object detection for autonomous vehicle in snow conditions," in *Proc. IEEE 13th Int. Conf. Ubiquitous Future Netw.*, 2022, pp. 479–482.

[12] M. Hahner et al., "LiDAR snowfall simulation for robust 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16364–16374.

[13] M. Hahner, C. Sakaridis, D. Dai, and L. V. Gool, "Fog simulation on real LiDAR point clouds for 3D object detection in adverse weather," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15283–15292.

[14] European Commission, "Road safety in the European union, 2018, Accessed: Aug. 31, 2022. [Online]. Available: https://doi.org/10.2832/169706

[15] U. S. "Department of Transportation: Federal Highway Administration, How do weather events impact roads?," 2019, Accessed: Aug. 31, 2022. [Online]. Available: https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm

[16] N. Charron, S. Phillips, and S. L. Waslander, "De-noising of lidar point clouds corrupted by snowfall," in *Proc. IEEE 15th Conf. Comput. Robot Vis.*, 2018, pp. 254–261.

[17] A. Kurup and J. Bos, "DSOR: A scalable statistical filter for removing falling snow from LiDAR point clouds in severe winter weather, 2021, *arXiv:2109.07078*.

[18] J.-I. Park, J. Park, and K.-S. Kim, "Fast and accurate desnowing algorithm for LiDAR point clouds," *IEEE Access*, vol. 8, pp. 160202–160212, 2020.

[19] W. Wang, X. You, L. Chen, J. Tian, F. Tang, and L. Zhang, "A scalable and accurate de-snowing algorithm for LiDAR point clouds in winter," *Remote Sens.*, vol. 14, no. 6, 2022, Art. no. 1468.

[20] B. Li, J. Li, G. Chen, H. Wu, and K. Huang, "De-snowing LiDAR point clouds with intensity and spatial-temporal features," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 2359–2365.

[21] R. Heinzler, F. Piewak, P. Schindler, and W. Stork, "CNN-based lidar point cloud de-noising in adverse weather," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2514–2521, Apr. 2020.

[22] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "SalsaNext: Fast, uncertainty-aware semantic segmentation of LiDAR point clouds," in *Proc. Int. Symp. Vis. Comput.*, 2020, pp. 207–222.

[23] X. Zhu et al., "Cylindrical and asymmetrical 3D convolution networks for LiDAR segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9939–9948.

[24] B. R. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 1–4.

[25] H. Tang et al., "Searching efficient 3D architectures with sparse point-voxel convolution," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 685–702.

[26] Z. Zhou, Y. Zhang, and H. Foroosh, "Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13194–13203.

[27] B. Wu, A. Wan, X. Yue, and K. Keutzer, "SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1887–1893.

[28] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4376–4382.

[29] C. Xu et al., "SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 1–19.

[30] R. Razani, R. Cheng, E. Taghavi, and L. Bingbing, "Lite-HDSeg: LiDAR semantic segmentation using lite harmonic dense convolutions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 9550–9556.

[31] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

[32] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[33] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4558–4567.

[34] H. Su et al., "SPLATNet: Sparse lattice networks for point cloud processing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2530–2539.

[35] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, "Tangent convolutions for dense prediction in 3D," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3887–3896.

[36] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11108–11117.

[37] R. A. Rosu, P. Schütt, J. Quenzel, and S. Behnke, "LatticeNet: Fast spatio-temporal point cloud segmentation using permutohedral lattices," *Auton. Robots*, vol. 46, no. 1, pp. 45–60, 2022.

[38] X. Yan et al., "Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion," in *Proc. AAAI Conf. Artif. Intell.*, 2021 vol. 35, pp. 3101–3109.

[39] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4213–4220.

[40] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, "RPVNet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16024–16033.

[41] K. Sirohi, R. Mohan, D. Büscher, W. Burgard, and A. Valada, "EfficientLPS: Efficient LiDAR panoptic segmentation," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1894–1914, Jun. 2022.

[42] E. E. Aksoy, S. Baci, and S. Cavdar, "SalsaNet: Fast road and vehicle segmentation in LiDAR point clouds for autonomous driving," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 926–932.

[43] H. Li, G. Chen, G. Li, and Y. Yu, "Motion guided attention for video salient object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7274–7283.

[44] M. Berman, A. R. Triki, and M. B. Blaschko, "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, 2018, pp. 4413–4421.

[45] P. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura," *Bull. Soc. Vaudoise Sci. Nat.*, vol. 37, pp. 547–579, 1901.

[46] M. Pitropov et al., "Canadian adverse driving conditions dataset," *Int. J. Robot. Res.*, vol. 40, no. 4/5, pp. 681–690, 2021.

[47] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[48] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Int. Conf. Learn. Representations*, 2015.