sensors

MDPI

Article

Real-Time LiDAR Point-Cloud Moving Object Segmentation for Autonomous Driving

Xing Xie [1], Haowen Wei [2] and Yongjie Yang [1,*]

[1] School of Information Science and Technology, Nantong University, Nantong 226019, China
[2] Department of Computer Science, Columbia University, New York, NY 10027, USA
* Correspondence: yang.yj@ntu.edu.cn

Abstract: The key to autonomous navigation in unmanned systems is the ability to recognize static and moving objects in the environment and to support the task of predicting the future state of the environment, avoiding collisions, and planning. However, because the existing 3D LiDAR point-cloud moving object segmentation (MOS) convolutional neural network (CNN) models are very complex and have large computation burden, it is difficult to perform real-time processing on embedded platforms. In this paper, we propose a lightweight MOS network structure based on LiDAR point-cloud sequence range images with only 2.3 M parameters, which is 66% less than the state-of-the-art network. When running on RTX 3090 GPU, the processing time is 35.82 ms per frame and it achieves an intersection-over-union(IoU) score of 51.3% on the SemanticKITTI dataset. In addition, the proposed CNN successfully runs the FPGA platform using an NVDLA-like hardware architecture, and the system achieves efficient and accurate moving-object segmentation of LiDAR point clouds at a speed of 32 fps, meeting the real-time requirements of autonomous vehicles.

Keywords: moving object segmentation; LiDAR; CNN; FPGA

check for updates

Citation: Xie, X.; Wei, H.; Yang, Y. Real-Time LiDAR Point-Cloud Moving Object Segmentation for Autonomous Driving. *Sensors* **2023**, *23*, 547. https://doi.org/10.3390/s23010547

Academic Editors: Miaohui Wang, Guanghui Yue, Jian Xiong and Sukun Tian

Received: 1 December 2022
Revised: 29 December 2022
Accepted: 29 December 2022
Published: 3 January 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Presently, the key to autonomous navigation in autonomous driving systems is the ability to recognize static and moving objects in the environment, and to support predicting the future state of the environment, avoiding collisions, and planning tasks. Moving object segmentation (MOS) algorithms improve environment perception [1], localization [2], and future state prediction [3] by distinguishing between moving and static objects in 3D LiDAR point cloud data. However, the MOS task is computationally intensive and the network model is complex [4,5], so it is very important to meet the real-time processing requirements of autonomous driving applications.

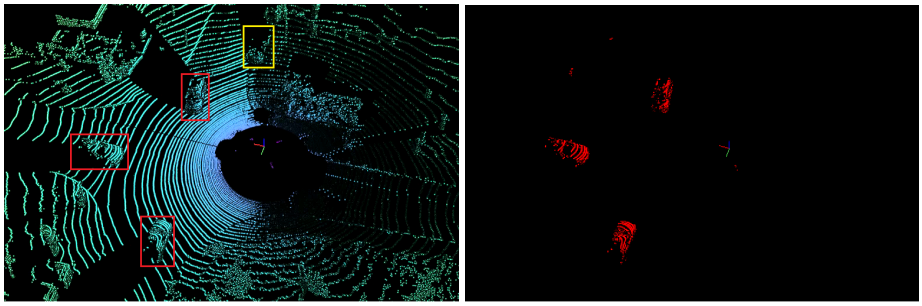
Most of the existing LiDAR-based point-cloud semantic segmentation networks predict the semantic labels of point clouds, such as vehicles, buildings or roads from a single frame. However, comparing to images, LiDAR provides better object location information via consecutive frames, so that we can also use LiDAR to distinguish moving objects. Recently proposed point-based [6] or voxel-based [1] segmentation networks, although superior in performance, are structurally complicated and computationally expensive. In the recent work [4], a segmentation network based on range images was adopted, the 3D LiDAR point cloud was projected onto a 2D plane, the range images of consecutive frames were used as the intermediate representation, and a 2D convolutional neural network (CNN) was used. This network performs the moving segmentation task. Furthermore, almost all state-of-the-art point-cloud moving object segmentation networks target GPUs that may not suitable for edge computing. From a computation point of view, edge deep learning accelerators (such as NVIDIA Deep Learning Accelerator (NVDLA) [7] and Xilinx DPU [8]) do not accelerate all common operations. Therefore, designing neural networks compatible with edge deep learning accelerators is critical for real-time embedded applications.

In this paper, we propose a lightweight multi-branch network structure to solve the problem of 3D LiDAR point-cloud moving object segmentation, which can run in real time on GPU. Figure 1 shows an example scene of our segmentation, red boxes are moving cars, the yellow box is a parked car, and moving objects are represented by red masks, which also verifies the feasibility of our method. Furthermore, the MOS computing system is built for autonomous vehicles, which can perform point-cloud pre-processing and neural network segmentation. Since only post-processing steps are left to the automotive electronic Control Unit ECU, this solution significantly alleviates the computation burden of ECU, thereby reducing the decision making and vehicle reaction latency. Our moving object segmentation network achieved 32 frames per second (fps) on FPGA. The contributions of this paper are summarized below:

(1) To our knowledge, this is one of the first end-to-end FPGA implementations for a real time LiDAR point-cloud moving-object segmentation deep learning platform, a LiDAR is directly connected to the processing system (PS) side. After pre-processing, the point cloud is stored in the DDR memory, which is accessible by the hardware accelerator on the programmable logic (PL) side.

(2) A light-weight and real-time moving-object segmentation network is proposed, targeting to NVDLA. Hardware-friendly layers are used (i.e., by replacement of deconvolution with bi-linear interpolation) to greatly reduce the complexity of computation. Its IoU score on the SemanticKITTI test set is 51.3%. The inference time on NVIDIA RTX 3090TI is about 35.82 ms.

(3) An efficient moving-object segmentation network architecture is implemented on the ZCU104 MPSoC FPGA platform, which enables real-time processing at 32 frames per second (fps).



(**a**) (**b**)

Figure 1. Moving object segmentation using our approach. (**a**) Raw Point Cloud ; (**b**) Segmented Point Cloud.

The rest of this paper is organized as follows: Section 2 summarizes the existing research results of moving-object segmentation in LiDAR point clouds and the FPGA implementation of the segmentation network. In Section 3, the proposed moving-object segmentation network model of LiDAR point cloud and its training details are described. The FPGA implementation and its results are discussed in Sections 4 and 5, respectively. Finally, Section 6 summarizes the whole paper.

2. Related Work

2.1. LiDAR Point-Cloud Moving Object Segmentation

Existing LiDAR point cloud moving object segmentation networks can be categorized into two groups: computer-vision-based [9–13] and LiDAR-sensor-based [14–16]. However, the processing of LiDAR data remains challenging due to the uneven distribution and sparsity of LiDAR point clouds. Here, we mainly study the MOS problem of 3D LiDAR point cloud data.

In recent years, great progress has been made in semantic segmentation based on LiDAR sensor point cloud data [17–24], such as the point-cloud compression methods

in [23,24] and so on. Semantic segmentation is a key step in the segmentation of moving objects in LiDAR point clouds. However, most of the existing semantic segmentation convolutional neural networks can only predict the semantic labels of point clouds, such as vehicles, buildings, and people, but cannot distinguish between actual moving objects and static objects, such as moving cars and parked ones.

The state-of-the-art scene flow method, FlowNet3D [25], is designed based on PointNet [6] and PointNet++ [26], which directly processes the original irregular 3D points without any pre-processing, and estimates each LiDAR point for two consecutive frames. The translational flow vectors include moving vehicles and pedestrians. Although these methods perform well on small point clouds, processing power becomes inefficient on larger point cloud datasets, requiring longer runtime. In addition, there are various 3D point-cloud-based semantic segmentation methods, such as SpSequencenet [27], KPConv [28], and SPVConv [29], which are also able to achieve state-of-the-art performance in semantic segmentation tasks. Among them, SpSequencenet [27] uses changes in sequence point clouds to predict moving objects. However, one problem with all networks based on operating directly on the point cloud is the dramatic increase in processing power and memory requirements, causing the point cloud to become larger. Therefore, training is difficult and cannot meet the real-time requirements of the automatic driving system.

Chen et al. [4] developed LMNet, which utilizes the residual between the current frame and the previous frame to be used as an additional input to the semantic segmentation network to achieve class-independent moving object segmentation, as well as in RangeNet++ [17] and SalsaNext [18] for performance evaluation. These networks are capable of real-time moving object segmentation running faster than the frame rate of the LiDAR sensor used. Mohapatra et al. [30] introduced a computationally efficient moving object segmentation framework based on LiDAR bird's eye view (BEV) space. The work in [31] utilizes a dual-branch structure to fuse the spatio-temporal information of LiDAR scans to improve the performance of MOS. In contrast, Kim et al. [32] proposed a network architecture that fuses motion features and semantic features, achieving improvements in computational speed and performance metrics. In the recent work of [33], the autoregressive system identification (AR-SI) theory was used to significantly improve the segmentation effect of the traditional encoder–decoder structure, and the model was deployed in embedded devices for actual measurement.

2.2. FPGA Implementations of Segmentation Networks

Advanced driver-assistance systems (ADAS) are rapidly being integrated into almost all new vehicles. The LiDAR point cloud segmentation algorithm must meet the real-time requirements, which may not be well met by standard CPUs or GPUs. FPGA has the advantages of high energy efficiency ratio and flexible reconfiguration, which can realize the high energy efficiency deployment of semantic segmentation networks in ADAS. Current researchers [5,34–39] mainly study and analyze the lightweight semantic segmentation network algorithm and the accelerated computation combined with the resource characteristics of customized hardware platform. Xilinx will support Continental's new advanced LiDAR sensor ARS 540 through the Zynq UltraScale+ MPSoC platform, partnering to create the automotive industry's first mass-produced 4D imaging sensor, paving the way for L5-level autonomous driving systems. In Ref. [16], a LiDAR sensor is directly connected to FPGA through an Ethernet interface, realizing a deep learning platform of end-to-end 3D point cloud semantic segmentation based on FPGA, which can process point-cloud segmentation in real time.

3. Proposed Network

The design method of the moving object segmentation network is mainly inspired by [4]. The residual image is used as an additional input to the designed semantic segmentation network to achieve moving object segmentation. In the following, we will describe our method in detail.

3.1. Spherical Projection of LiDAR Point Cloud

Following previous work [16,17], we use a 2D neural network convolution to extract features from the range view (RV) of LiDAR. Specifically, we project the LiDAR point (x, y, z) onto a sphere and finally convert it to image coordinates (u, v), defined as:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left[1 - \arctan(y, x)\pi^{-1}\right]w \\ \left[1 - \left(\arcsin\left(zr^{-1}\right) + f_{up}\right)f^{-1}\right]h \end{pmatrix} \tag{1}$$

where (u, v) are the image coordinates, (h, w) are the desired range image according to the height and width, r represents the range of each point as $r = \sqrt{x^2 + y^2 + z^2}$, and $f = |f_{\text{down}}| + |f_{up}|$ for the sensor's vertical field of view.

We use Equation (1) to extract the range index r, 3D point coordinates (x, y, z) and intensity value i for each point projected to (u, v), and take them as features to be superimposed along the channel dimension. Therefore, we can directly input these features into the network, and then transform point-cloud moving segmentation into image moving segmentation.

3.2. Residual Images

As in ref. [4], the residual image and range view based on LiDAR point cloud are used as the input of the segmentation network, and the temporal information in the residual image is used to distinguish the static object and the pixels on the moving object, so the actual moving object and the static object can be distinguished.

Assuming that there are N time series of LiDAR scans in the SLAM history, $S_j = \{\boldsymbol{p}_i \in \mathbb{R}^4\}$ and M points are represented as homogeneous coordinates, i.e., $\boldsymbol{p}_i = (x, y, z, 1)$. $\boldsymbol{T}_N^{N-1}, \ldots, \boldsymbol{T}_1^0$ is denoted as the transformation matrix between $N + 1$ scan poses, i.e., $\boldsymbol{T}_k^l \in \mathbb{R}^{4\times4}$. Equation (2) represents the coordinate system in which the k^{th} scan transformed into the l^{th} scan

$$\mathcal{S}^{k\to l} = \left\{ \prod_{j=k}^{l+1} T_j^{j-1} p_i \mid p_i \in S_k \right\} \tag{2}$$

In Ref. [4], in order to generate the residual image and fuse it into the current range image, transformation and re-projection are required. First, the transformation estimate defined in the ego-motion is compensated according to Equation (2) by transforming the previous scan to the current given local coordinate system, and next, the $S^{k\to l}$ of the past scans are re-projected to the current range image view using Equation (1). In order to calculate the residual $d_{k,i}^l$ for each pixel i, we use the normalized absolute difference between the range of the current frame and the transformed frame to calculate, as defined by

$$d_{k,i}^l = \frac{\left| r_i - r_i^{k\to l} \right|}{r_i} \tag{3}$$

where r_i is the range value from $\boldsymbol{p}_i$ to the current frame at the image coordinates (u_i, v_i), and $r_i^{k\to l}$ is the range value from the transformed scan to the pixel in the same image. In the scene of moving objects, the displacement of the moving car is relatively large compared to the static background, and the residual image is obvious, while the residual image of the slowly moving object is blurred and the residual pattern is not obvious. Therefore, direct use of residual images for moving object segmentation cannot achieve good performance. Finally, we concatenate the residual image and the range view as the input of the segmentation network, and each pixel fuses the spatial and temporal information.

3.3. Network Architecture

In this paper, our proposed network is mainly divided into two branches: context path and spatial path, which respectively extract feature information and then fuse these feature information. The architecture of our proposed CNN for point-cloud moving object

segmentation is shown in Figure 2. The backbone module utilized in the context path branch is ResNet-18 [40] for eight times fast down-sampling. Subsequently, the extracted features are fed into the Atrous Spatial Pyramid Pooling (ASPP) [41] module in order to connect features from different perceptual domains. ASPP builds convolution kernels with different receptive fields through different dilated rates to increase the receptive fields of the network and enhance the ability of the network to obtain multi-scale context, so as to obtain good performance. However, from the perspective of hardware (GPU or FPGA), dilated convolution has low efficiency and slow inference speed, and the larger the dilated rate, the longer the convolution processing time. Therefore, our network keeps the dilated rate as 2, and this method can simulate the function of ASPP without reducing the computational efficiency of GPU and NVDLA. Next, a global context module (GCM) is introduced to extract contextual information and guide feature learning of the current path. GCM consists of a global average pooling layer and a 1×1 convolution layer that extracts global context features.

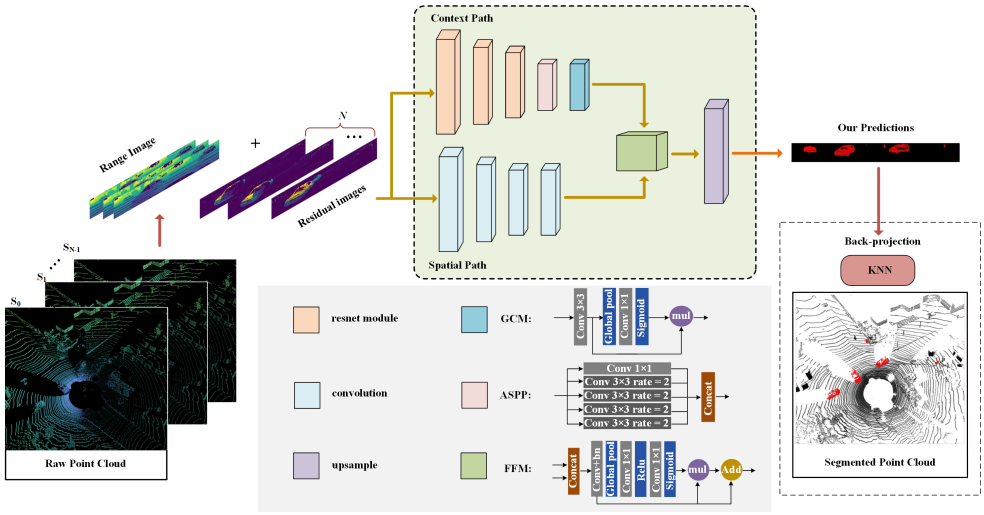


Figure 2. Architecture of the proposed CNN for point-cloud moving object segmentation.

The spatial path mainly retains rich spatial information to generate high-resolution feature maps, which only contains four convolutional layers. The first three convolutional layers are stride = 2, and 1/8 feature maps are extracted. The feature fusion module (FFM) [42] is used to fuse the features of context branch and spatial branch at different scales. Therefore, the features of the two channels cannot be simply weighted, but superimposed by concatenation method. FFM combines the attention mechanism for feature fusion, mainly including global pooling layer, 1×1 convolution layer, ReLU activation layer and Sigmoid layer. At the end of the network, in order to output the moving object segmentation results of the original image size, the mainstream high-performance segmentation networks, U-NET [43] and FCN [44], use layer skip connection for up-sampling. This requires a GPU or FPGA for more computation and data movement. Therefore, we up-sample the FFM output eight times using a bi-linear interpolation algorithm.

3.4. Training Details

We implemented a 3D LiDAR point-cloud moving-object segmentation network using PyTorch and trained on a single NVIDIA RTX 3090TI GPU. We use the method of [4] to train the network, process all point clouds according to Equations (1)–(3), and generate 64×2048 range views and residual images respectively. The residual images are then concatenated with the current range image and used as input to a 2D convolutional neural network. Trained with the new binary masks, the proposed method can separate moving and static objects label maps. During training, the network is trained with an initial learning rate of 0.01 and a weight decay of 1×10^{-4}.

3.5. Dataset and Evaluation

SemanticKITTI [45] is a large-scale dataset for semantic scene understanding of 3D LiDAR point cloud sequences, including semantic segmentation and semantic scene completion. The dataset contains 28 annotated categories such as pedestrians, vehicles, parking lots, roads, buildings, etc., which further distinguishes static objects from moving objects. The raw odometry data consists of 22 sequences of point cloud data. We follow the same protocol in [17], where the sequences 00–10 are used for training and the sequence 08 is used for validation. The remaining sequences 11–21 are used as the test set. All classes are reorganized into two types: moving and non-moving/static objects according to [4]. The former one contains actually moving vehicles and pedestrians, all other classes are non-moving/static objects.

In order to evaluate the MOS performance, we follow the official guidance, using the Jaccard index or IoU [46], which is:

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{4}$$

where TP, FP, and FN correspond to the number of true-positive, false-positive, and false-negative predictions for the moving classes.

Referring to the evaluation method proposed in [4], we use IoU to evaluate the accuracy of moving object segmentation in LiDAR point clouds. Table 1 shows the MOS performance compared to the state-of-the-art on the SemanticKITTI test set. Table 2 shows the results of the validation set. Since all operations of our network are supported by NVDLA, the network complexity is reduced and no semantic information is added. Thus, when our proposed model uses N = 8 residual images, the best performance IoU score (51.3%) is obtained, but slightly lower than the baseline LMNet [4] on the test benchmark.

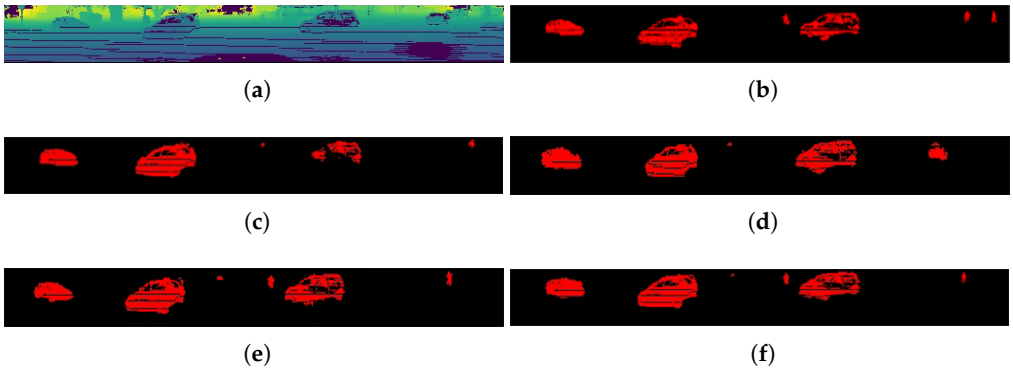
For qualitative evaluation, Figure 3 shows the qualitative results of different methods on the SemanticKITTI test set. Meanwhile, the qualitative results of the point clouds are shown in Figure 4. In the intersection in the figure below, there are a large number of moving objects and non-moving/stationary objects such as moving vehicles and walking people; our method can distinguish between actual moving vehicles and pedestrians, while other methods cannot detect slow-moving objects.

Table 1. MOS performance compared to the state-of-the-art on the SemanticKITTI test set.

Methods	IoU (%)
SceneFlow [25]	28.7
SpSequenceNet [27]	43.2
SalsaNext [18]	46.6
LMNet [4]	62.5
BiSeNet [42]	45.1
Ours	51.3

Table 2. MOS performance compared to the state-of-the-art on the SemanticKITTI validation set.

Methods	IoU (%)
SalsaNext [18]	48.6
LMNet [4]	65.3
BiSeNet [42]	46.1
Ours	52.4

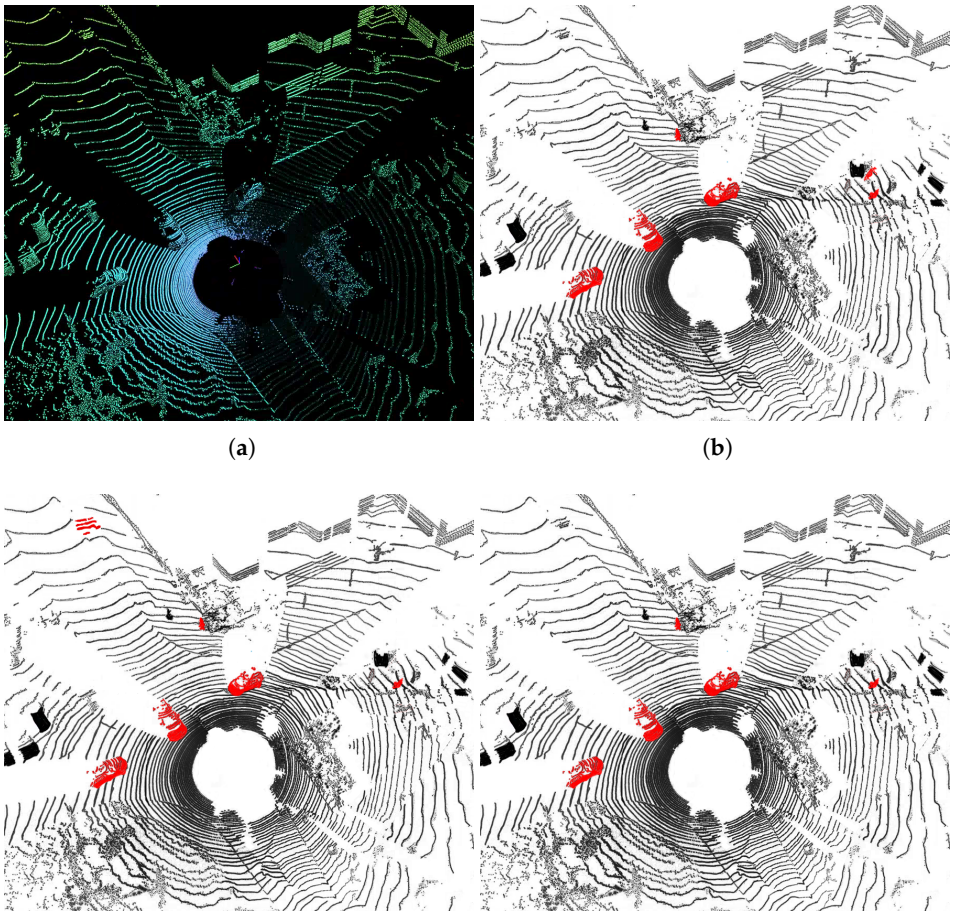


(**a**) (**b**)

(**c**) (**d**)

(**e**) (**f**)

Figure 3. Qualitative results of different methods on the SemanticKITTI test set, where red pixels correspond to moving objects (range view images). (**a**) Range Image; (**b**) Ground Truth Labels; (**c**) BiSeNet (retrained); (**d**) SalsaNext (retrained); (**e**) LMNet; (**f**) ours.



(**a**) (**b**)

(**c**) (**d**)

Figure 4. Qualitative results shown as point clouds. (**a**) Raw Point Cloud; (**b**) Ground Truth Labels, and (**c**,**d**) prediction results, where red points correspond to the class moving (**c**) LMNet; (**d**) Ours.

3.6. Ablation Studies

In this section, some ablation experiments on the validation set (sequence 08) of the SemanticKITTI dataset are conducted to analyze the effect of each component's performance shown in Table 3. As shown in Table 3, it can be observed that the IoU of the dual-branch architecture can be increased by 9.6% compared to that of the single-branch architecture.

On this basis, FFM, GCM, ASPP, and their combination are added. It is worth noting that the IoU of our proposed final setup can achieve 52.4%.

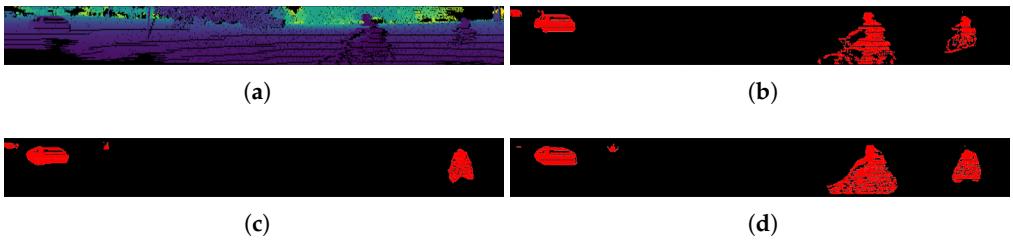
Table 3. Ablation study of components on the validation set. CP: Context Path; SP: Spatial Path; GCM: Global Context Module; FFM: Feature Fusion Module.

Methods	IoU (%)
CP	33.6
CP + SP + Sum	43.2
CP + SP + FFM	45.1
CP + SP + FFM + GCM	47.4
CP + SP + FFM + GCM+ASPP	52.4

In our design, the K-Nearest Neighbor (KNN) post-processing is used to back-project the 2D prediction result to the 3D point cloud. In order to verify the attractive performance of the KNN post-processing in our proposed design, the comparison with regard to the back-projection between the Conditional Random Field (CRF) post-processing and the KNN post-processing is provided in Table 4. As shown in Table 4, compared to CRF post-processing, KNN post-processing results in better IoU performance. Figure 5 shows the qualitative results of different post-processing methods for MOS on the validation set. The qualitative results prove that the KNN can handle the blurred boundary of moving objects in a better way. This also obeys our expectation. Due to the fact that, during dimension reduction, different 3D points belonging to different categories might project into the same pixel in the 2D range image. Considering the principle of CRF, it contributes little to solve this issue if it is applied to a 2D range image. While KNN counts nearest points in 3D space rather than 2D.

Table 4. Comparison of the post-processing between the KNN and the CRF on the validation set.

	Methods	IoU (%)
(1)	KNN	52.4
(2)	CRF	49.1



(**a**) (**b**)

(**c**) (**d**)

Figure 5. Qualitative results of different post-processing methods for MOS on the validation set, where red pixels correspond to moving objects (range view images). (**a**) Range Image; (**b**) Ground Truth Labels; (**c**) CRF; (**d**) KNN.

3.7. Run-Time Evaluation on GPU

In autonomous driving systems, the processing speed of the moving object segmentation network must meet real-time requirements. To get a fair performance evaluation, all measurements are evaluated on the SemanticKITTI dataset 08 sequence using a single NVIDIA RTX 3090TI-24GB card and the network performances are shown in Table 5. Compared to the state-of-the-art network LMNet [4], our model clearly shows better performance, the running time is 35.82 ms, and the amount of network parameters is about 2.3 M, which is reduced to 1/3 of that in LMNet [4].

Table 5. Run-time performance on the SemanticKITTI validation set.

	Processing Time	Speed	Parameters
BiSeNet [42]	56.3 ms	18 fps	13.76 M
LMNet [4]	42.21 ms	23 fps	6.71 M
MotionSeg3D [31]	116.71 ms	8 fps	6.73 M
Ours (GPU)	35.82 ms	28 fps	2.3 M
Ours (FPGA)	31.69 ms	32 fps	2.3 M

Figure 6 shows the inference speed vs. IoU on the validation set. For practical use in embedded systems on autonomous vehicles, the IoU of our design in GPU and FPGA is sacrificed to achieving the higher inference speed compared to the other methods. Note that our implementation runs significantly faster than the 10 Hz sampling rate of mainstream LiDAR sensors [47].

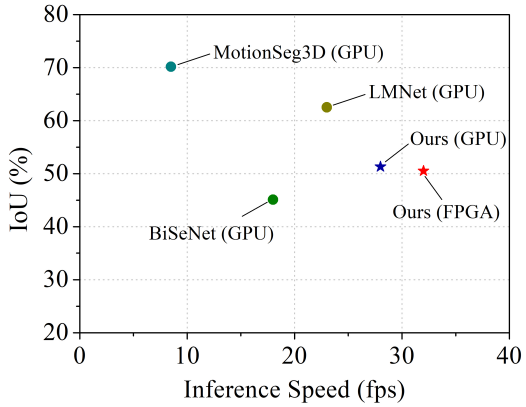


Figure 6. Inference speed vs. IoU on the SemanticKITTI validation set. Red star indicates our method in FPGA, the blue star indicates our method in GPU, and colored dots represent other methods.

4. Hardware Architecture

The hardware architecture of the point cloud moving object segmentation network is shown in Figure 7. It consists of processing system (ARM core) and programmable logic (FPGA) parts. ARM core is used to complete pre-processing and post-processing, such as point cloud reading, image resizing [48], result showing, etc. On the FPGA side, an NVIDIA Deep Learning Accelerator (NVDLA) like system is implemented. We tailor it and adopt it into FPGA.

The core of the convolutional engine is the MAC array. In this work, the size of the MAC array is chosen to be 32×32. To improve the processing speed and alleviate the bandwidth requirement between FPGA and DDR memory, NVDLA adopts a Ping-Pong buffer. Different from a micro-controller in NVDLA, we implement a finite state machine (FSM) to control the running order of CNN operations. In this study, we quantize the neural network to INT8 for high computation efficiency.

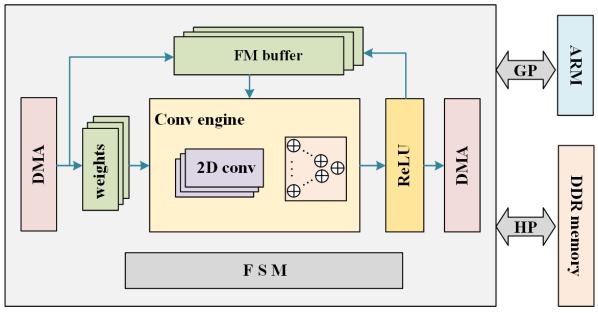


Figure 7. Hardware architecture of the CNN accelerator on the FPGA.

5. Results and Discussion

The target hardware platform is the Zynq UltraScale+ MPSoC ZCU104 development board. Figure 8 exhibits the overall system setup with LiDAR connected to the FPGA board directly, which demonstrates our experiment setup. The LiDAR driver is implanted in the ARM processor on ZCU104 board and connected to LiDAR via UDP protocol on the Ethernet port. The ARM processor receives each set of point cloud data from LiDAR and stores it into DDR memory for NVDLA fetching.

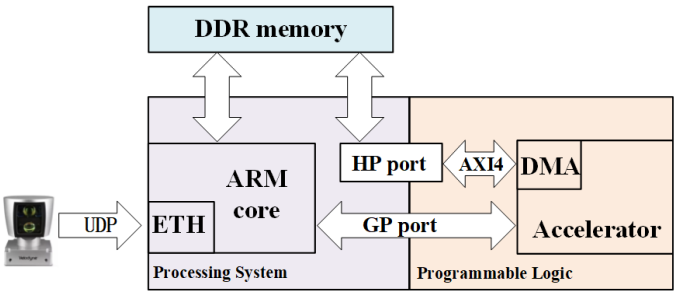


Figure 8. Overall system setup with LiDAR connected to the FPGA board directly.

The hardware resource usage of our proposed neural network is shown in Table 6. This design has used 91.84% of the DSP resources, if the parallelism is increased, a larger FPGA needs to be used. Table 7 shows the run-time performance of the proposed approach on SemanticKITTI Dataset. It can be observed that when running at 250 MHz, this accelerator's processing speed is 32 fps. The estimated power consumption of the FPGA implementation is 12.8 W. The only real-time solution currently available, SalsaNext [18], runs on Nvidia Quadro P6000 GPUs and requires 600–650 W PC power support. Therefore, our solution provides a balanced and practical approach for running LiDAR point cloud moving object segmentation tasks on embedded devices. Since there are few 3D point-cloud moving object segmentation implementations on FPGA, the performance and hardware resource utilization comparison with similar works is not yet available.

Table 6. FPGA Resource utilization for the CNN accelerator.

FPGA Resource	Used	Available	Utilization
LUT	102,707	230,400	44.58%
FF	114,221	460,800	24.79%
DSP	1587	1728	91.84%
BRAM	162	312	51.92%

Table 7. Run-time Performance of the Proposed Approach on SemanticKITTI Dataset.

Device	Precision	Processing time	Speed
GPU	FP32	35.82 ms	28 fps
FPGA	INT8	31.69 ms	32 fps

6. Conclusions

In this study, we proposed a lightweight CNN architecture for LiDAR point-cloud moving object segmentation. Edge deep learning accelerators are designed with their limitations and computational efficiency in mind, and our proposed network structure fully supports all operations of NVDLA. On SemanticKITTI dataset, the processing time of the network on RTX 3090TI GPU is 35.82 ms and the IoU score of 51.3% is achieved. When compared to the state-of-the-art network, the network achieves a similar error performance, but using only 34% of the parameters. In addition, the proposed network successfully targets the MPSoC FPGA platform using NVDLA hardware architecture. The system

successfully achieves efficient and accurate moving-object segmentation of LiDAR point clouds at 32 fps, which meets the real-time requirements of autonomous vehicles.

However, some potential problems need to be solved in the future. First of all, in order to reduce the computational complexity of our proposed network, we plan to simplify the original structure and remove the time-consuming cross-layer connections while ensuring its high performance. Secondly, due to acceleration of the model inference, the low-level details mostly sacrificed, which leads to a considerable decrease in accuracy. In view of this, the spatial path would be improved by capturing the low-level details with wide channels and shallow layers in our future works.

Author Contributions: Conceptualization, X.X. and H.W.; methodology, X.X., H.W.; software, X.X.; formal analysis, X.X., H.W.; data curation, H.W. and X.X.; writing—original draft preparation, X.X; writing—review and editing, Y.Y.; supervision, X.X.; project administration, Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yin, Z.; Tuzel, O. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
2. Zhang, C.; Luo, W.; Urtasun, R. Efficient Convolutions for Real-Time Semantic Segmentation of 3D Point Clouds. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; pp. 399–408.
3. Yin, H.; Wang, Y.; Ding, X.; Tang, L.; Xiong, R. 3D LiDAR-Based Global Localization Using Siamese Neural Network. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 1380–1392. [CrossRef]
4. Chen, X.; Li, S.; Mersch, B.; Wiesmann, L.; Stachniss, C. Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robot. Autom. Lett.* **2021**, *6*, 6529–6536. [CrossRef]
5. Lyu, Y.; Bai, L.; Huang, X. Real-Time Road Segmentation Using LiDAR Data Processing on an FPGA. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018.
6. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85.
7. Nvidia. Nvidia Deep Learning Accelerator. 2018. Available online: https://developer.nvidia.com/deep-learning-accelerator (accessed on 2 July 2022).
8. Xilinx. Dpuczdx8g for Zynq Ultrascale+ Mpsocs Product Guide. 2021. Available online: https://docs.xilinx.com/r/en-US/pg338-dpu?tocId=Bd4R4bhnWgMYE6wUISXDLw (accessed on 3 August 2022).
9. Barnes, D.; Maddern, W.; Pascoe, G.; Posner, I. Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1894–1900.
10. Patil, P.W.; Biradar, K.M.; Dudhane, A.; Murala, S. An end-to-end edge aggregation network for moving object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8146–8155.
11. Cai, Y.; Luan, T.; Gao, H.; Wang, H.; Li, Z. YOLOv4-5D: An Effective and Efficient Object Detector for Autonomous Driving. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–13. [CrossRef]
12. Bell, A.; Mantecón, T.; Díaz, C.; del-Blanco, C.R.; Jaureguizar, F.; García, N. A Novel System for Nighttime Vehicle Detection Based on Foveal Classifiers with Real-Time Performance. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 5421–5433. [CrossRef]
13. Wang, H.; Chen, Y.; Cai, Y.; Chen, L.; Li, Y.; Sotelo, M.A.; Li, Z. SFNet-N: An Improved SFNet Algorithm for Semantic Segmentation of Low-light Autonomous Driving Road Scenes. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 21405–21417. [CrossRef]
14. Yan, J.; Chen, D.; Myeong, H.; Shiratori, T.; Ma, Y. Automatic Extraction of Moving Objects from Image and LIDAR Sequences. In Proceedings of the 2014 2nd International Conference on 3D Vision, Tokyo, Japan, 8–11 December 2014.
15. Postica, G.; Romanoni, A.; Matteucci, M.; Shiratori, T.; Ma, Y. Robust moving objects detection in lidar data exploiting visual cues. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016.
16. Xie, X.; Bai, L.; Huang, X. Real-Time LiDAR Point Cloud Semantic Segmentation for Autonomous Driving. *Electronics* **2022**, *11*, 11. [CrossRef]

17. Milioto, A.; Vizzo, I.; Behley, J.; Stachniss, C. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4213–4220.
18. Cortinhal, T.; Tzelepis, G.; Aksoy, E.E. SalsaNext: Fast, Uncertainty-aware Semantic Segmentation of LiDAR Point Clouds for Autonomous Driving. *arXiv* **2020**, arXiv:2003.03653.
19. Li, S.; Chen, X.; Liu, Y.; Dai, D.; Stachniss, C.; Gall, J. Multiscale interaction for real-time lidar data segmentation on an embedded platform. *arXiv* **2020**, arXiv:2008.09162.
20. Wu, B.; Zhou, X.; Zhao, S.; Yue, X.; Keutzer, K. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 4376–4382.
21. Zhang, Y.; Zhou, Z.; David, P.; Yue, X.; Foroosh, H. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 9598–9607.
22. Xu, C.; Wu, B.; Wang, Z.; Zhan, W.; Vajda, P.; Keutzer, K.; Tomizuka, M. *SqueezeSegV3: Spatially-Adaptive Convolution for Efficient Point-Cloud Segmentation*; Springer: Berlin/Heidelberg, Germany, 2020.
23. Sun, X.; Wang, M.; Du, J.; Sun, Y.; Cheng, S.S.; Xie, W. A Task-Driven Scene-Aware LiDAR Point Cloud Coding Framework for Autonomous Vehicles. *IEEE Trans. Ind. Inform.* **2022**. [CrossRef]
24. Sun, X.; Wang, S.; Wang, M.; Cheng, S.S.; Liu, M. An Advanced LiDAR Point Cloud Sequence Coding Scheme for Autonomous Driving. In Proceedings of the 28th ACM International Conference on Multimedia (MM '20), Seattle, WA, USA, 12–16 October 2020; pp. 2793–2801.
25. Liu, X.; Qi, C.R.; Guibas, L.J. Flownet3d: Learning scene flow in 3d point clouds. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 529–537.
26. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv* **2017**, arXiv:1706.02413.
27. Shi, H.; Lin, G.; Wang, H.; Hung, T.Y.; Wang, Z. Spsequencenet: Semantic segmentation network on 4d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 4573–4582.
28. Thomas, H.; Qi, C.; Deschaud, J.; Marcotegui, B.; Goulette, F.; Guibas, L. KPConv: Flexible and Deformable Convolution for Point Clouds. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27–28 October 2019.
29. Tang, H.; Liu, Z.; Zhao, S.; Lin, Y.; Lin, J.; Wang, H.; Han, S. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In Proceedings of the 16th European Conference on Computer Vision (ECCV), Edinburgh, UK, 23–28 August 2020.
30. Mohapatra, S.; Hodaei, M.; Yogamani, S.; Milz, S.; Gotzig, H.; Simon, M.; Rashed, H.; Maeder, P. LiMoSeg: Real-time Bird's Eye View based LiDAR Motion Segmentation. *arXiv* **2021**, arXiv:2111.04875.
31. Sun, J.; Dai, Y.; Zhang, X.; Xu, J.; Rui, A.; Gu, W.; Chen, X. Efficient Spatial-Temporal Information Fusion for LiDAR-Based 3D Moving Object Segmentation. *arXiv* **2022**, arXiv:2207.02201.
32. Kim, J.; Woo, J.; Im, S. RVMOS: Range-View Moving Object Segmentation Leveraged by Semantic and Motion Features. *IEEE Robot. Autom. Lett.* **2022**, *7*, 8044–8051. [CrossRef]
33. He, Z.; Fan, X.; Peng, Y.; Shen, Z.; Jiao, J.; Liu, M. EmPointMovSeg: Sparse Tensor Based Moving Object Segmentation in 3D LiDAR Point Clouds for Autonomous Driving Embedded System. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2022**, *42*, 41–53. [CrossRef]
34. Bai, L.; Lyu, Y.; Huang, X. RoadNet-RT: High Throughput CNN Architecture and SoC Design for Real-Time Road Segmentation. *IEEE Trans. Circ. Syst. Regul. Pap.* **2020**, 68, 704–714. [CrossRef]
35. Bai, L.; Zhao, Y.; Elhousni, M.; Huang, X. DepthNet: Real-Time LiDAR Point Cloud Depth Completion for Autonomous Vehicles. *IEEE Access* **2020**, *8*, 7825–7833. [CrossRef]
36. Shen, J.; You, H.; Qiao, Y.; Mei, W.; Zhang, C. Towards a Multi-array Architecture for Accelerating Large-scale Matrix Multiplication on FPGAs. In Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 27–30 May 2018; pp. 1–5.
37. Wu, G.; Yong, D.; Miao, W. High performance and memory efficient implementation of matrix multiplication on FPGAs. In Proceedings of the International Conference on Field-Programmable Technology, FPT 2010, Beijing, China, 8–10 December 2010.
38. Chang, J.W.; Kang, S.J. Optimizing FPGA-based convolutional neural networks accelerator for image super-resolution. In Proceedings of the 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), Jeju, Republic of Korea, 22–25 January 2018; pp. 343–348.
39. Lyu, Y.; Bai, L.; Huang, X. ChipNet: Real-Time LiDAR Processing for Drivable Region Segmentation on an FPGA. *IEEE Trans. Circ. Syst. Regul. Pap.* **2019**, *66*, 1769–1779. [CrossRef]
40. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.

41. Chen, L.-C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv* **2017**, arXiv:1706.05587.
42. Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; Sang, N. BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation. In Proceedings of the 15th European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
43. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), Munich, Germany, 5–9 October 2015; pp. 234–241.
44. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 640–651.
45. Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; Gall, J. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27–28 October 2019.
46. Everingham, M.; Gool, L.V.; Williams, C.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
47. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
48. Bradski, G. The OpenCV Library. Available online: https://github.com/opencv/opencv/wiki/CiteOpenCV (accessed on 20 August 2022).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.