sensors

Article

Color Image Generation from Range and Reflection Data of LiDAR

Hyun-Koo Kim, Kook-Yeol Yoo and Ho-Youl Jung *

Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38544, Korea; kim-hk@ynu.ac.kr (H.-K.K.); kyoo@yu.ac.kr (K.-Y.Y.)
* Correspondence: hoyoul@yu.ac.kr; Tel.: +82-53-810-3545

Received: 14 July 2020; Accepted: 17 September 2020; Published: 21 September 2020

Abstract: Recently, it has been reported that a camera-captured-like color image can be generated from the reflection data of 3D light detection and ranging (LiDAR). In this paper, we present that the color image can also be generated from the range data of LiDAR. We propose deep learning networks that generate color images by fusing reflection and range data from LiDAR point clouds. In the proposed networks, the two datasets are fused in three ways—early, mid, and last fusion techniques. The baseline network is the encoder-decoder structured fully convolution network (ED-FCN). The image generation performances were evaluated according to source types, including reflection data-only, range data-only, and fusion of the two datasets. The well-known KITTI evaluation data were used for training and verification. The simulation results showed that the proposed last fusion method yields improvements of 0.53 dB, 0.49 dB, and 0.02 in gray-scale peak signal-to-noise ratio (PSNR), color-scale PSNR, and structural similarity index measure (SSIM), respectively, over the conventional reflection-based ED-FCN. Besides, the last fusion method can be applied to real-time applications with an average processing time of 13.56 ms per frame. The methodology presented in this paper would be a powerful tool for generating data from two or more heterogeneous sources.

Keywords: artificial intelligence; fusion technique; heterogeneous source; image generation; LiDAR range; LiDAR reflection; sparse input

1. Introduction

Light detection and ranging (LiDAR) sensors are widely used for the advanced driver-assistance systems (ADAS) and autonomous vehicles. LiDAR sensors provide information that consists of range (distance) and reflection. LiDAR range data have been used for various applications, such as semantic segmentation [1–4], 3D mapping [5,6], and object detection [7–14]. On the other hand, reflection data have been utilized for the recognition of driving-related factors in the environment, such as lanes, road marks, and traffic signs, which have relatively high reflectivity [15–18].

Interestingly, it has been reported that the deep learning-based encoder-decoder structured fully convolution network (ED-FCN) can successfully generate camera-captured-like color images from heterogeneous LiDAR reflection data [19–21]. Note that the ED-FCN network is originally applied for semantic segmentation. The methods [19–21] consist of two steps, as shown in Figure 1. In the first step, LiDAR 3D reflection data are projected into 2D color image coordinate. The color image is generated from the projected reflection data in the second step.

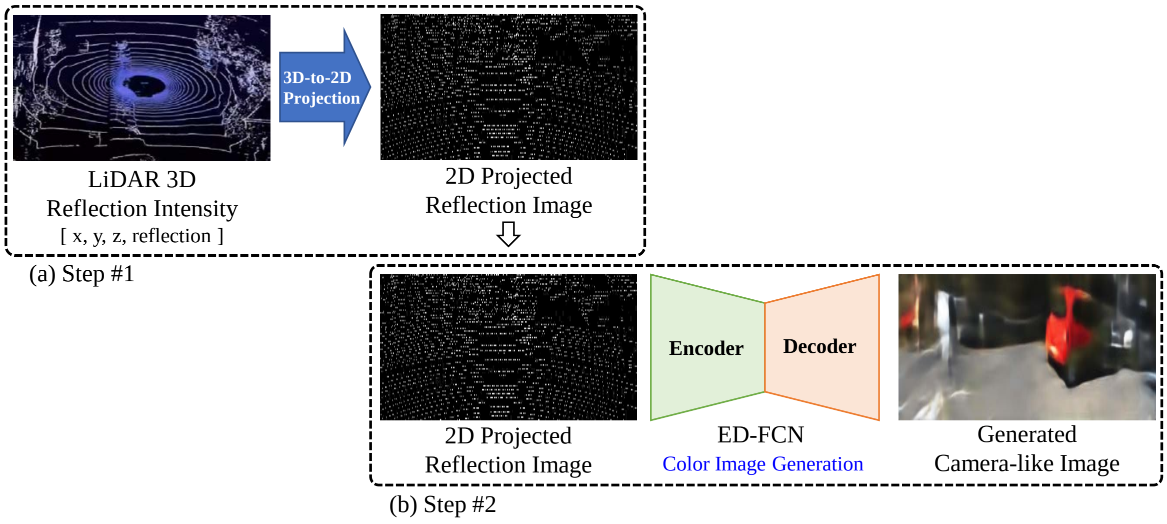

3D-to-2D Projection
LiDAR 3D Reflection Intensity
[x, y, z, reflection]
2D Projected Reflection Image
(a) Step #1
2D Projected Reflection Image
Encoder
Decoder
ED-FCN
Color Image Generation
Generated Camera-like Image
(b) Step #2

Figure 1. The camera-captured-like color-image-generation method using 3D LiDAR point clouds. The method consists of two steps: (**a**) LiDAR 3D-to-2D projection and (**b**) color image generation.

In [19], a low-complexity ED-FCN network was used for camera-captured-like gray-scale image generation. Note that the monochrome images were generated from the LiDAR reflection data. The projected reflection image had different sparsity and characteristics, compared with the target image. To increase better generation performance and generate color image, an asymmetric ED-FCN, i.e., a decoder, with greater depth than the encoder, was proposed in [20]. The asymmetric ED-FCN outperformed symmetric ED-FCN and the generative adversarial network (GAN)-based colorization method [22]. Recently, a selected connection UNET (SC-UNET) was proposed in [21], which considers the sparseness of each level in the encoder network and the similarity between the same levels of encoder and decoder networks. The SC-UNET with the connection between encoder and decoder at the two lowest levels outperforms the SC-UNET with other connections and the asymmetric ED-FCN. One interesting result discussed in [19–21] was that shadow-free images were generated since the LiDAR reflection data were originally produced, irrespective of the illumination. Note that these methods [19–21] generate an image only from the reflection data of LiDAR.

In this study, we firstly tried to generate color images using the LiDAR range data. The simulation result confirms that the color image can be also generated from the range data. Accordingly, we propose three color-image-generation methods which fuse both reflection and range data to improve the quality of generated image—early fusion, mid fusion, and last fusion-based LiDAR to color-image-generation methods. KITTI-based evaluation dataset was used for training and performance verification [20,21]. Peak signal-to-noise ratio (PSNR) [23] and structural similarity index measure (SSIM) [23,24] metrics were used for the performance evaluation. The number of weights and the average operation time of the network were used for the comparison of the computational complexities of various image generation networks. The proposed fusion methods improve image generation performance over the conventional image generation methods which use only reflection data. In addition, the proposed fusion methods have operating times applicable to real-time applications. The simulation results show that the proposed fusion methods give better performance compared reflection-only methods.

The rest of this paper is organized as follows. In Section 2, we propose fusion network architectures which generates a camera-captured-like 2D color image from both the range and reflection 3D LiDAR data. The training and inference processes are also described. In Section 3, the performances of the proposed networks are compared with the performance of the conventional reflection-based ED-FCN network. Additionally, the range-based ED-FCN network is compared. Section 4 draws the conclusions.

2. Proposed Method

The conventional color-image-generation methods [19–21] use only LiDAR reflection data. To verify the possibility of color image generation from range data, the range data are applied to the conventional image generation networks. To improve the color image generation performance, we propose three-color image generation networks using LiDAR range data and reflection data. Considering that LiDAR range and reflection data have different characteristics, fusion network architectures were proposed and evaluated through experiments.

2.1. Transformation of 3D LiDAR Point Cloud

The 3D LiDAR point cloud $[X, Y, Z, R]^T$ provided by the LiDAR sensor consists of the world coordinate position $[X, Y, Z]^T$ and the reflection data $R(X, Y, Z)$ of the object. The 3D LiDAR point cloud is projected onto the coordinate $[u, v]^T$ of the color image to be generated by using Equation (1).

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R_L^C & t_L^C \\ 0 & 1 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

where s indicates the scale factor, f_k; c_k and $k = u, v$ are focal length and principal point of the camera, respectively; and $R_L^C \in \mathbb{R}^{3\times3}$ and $t_L^C \in \mathbb{R}^{1\times3}$ represent the rotation and translation matrices for LiDAR-to-camera transform, respectively.

The 2D reflection images are obtained according to Equation (2). Practically, several 3D LiDAR points, $(X_i, Y_i, Z_i), i = 1, \ldots, N$ provided by a LiDAR sensor may be projected onto the same location of (u, v) in the image plane. In this case, the reflection value, $r(u, v)$ is determined by the average of the reflection values of the N LiDAR points.

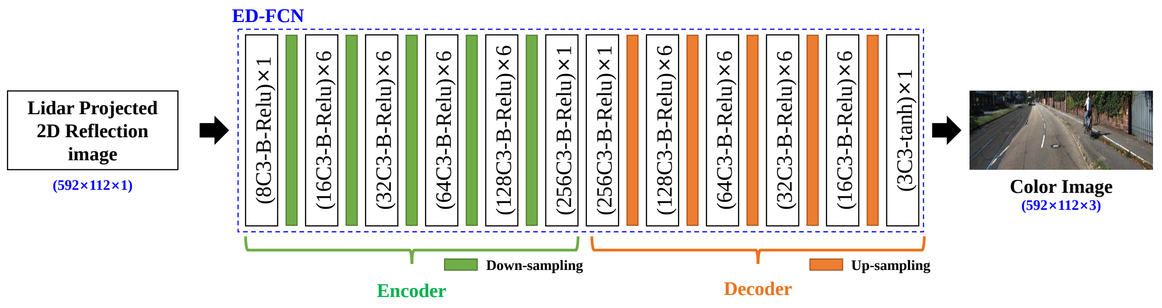
$$r(u, v) = \frac{1}{N}\sum_{i=1}^{N} R(X_i, Y_i, Z_i) \tag{2}$$

The 2D range images are calculated by using Equation (3). Similarly, for N 3D LiDAR points projected onto the same location of (u, v), the distance value, $d(u, v)$, is determined by the minimum of the distance values of the N LiDAR points.

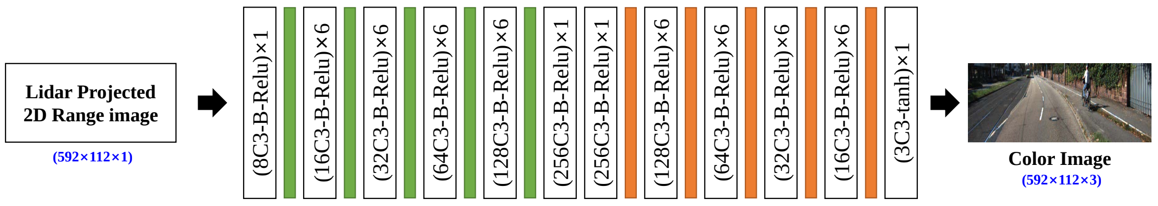
$$d(u, v) = \min_{i=1\ldots N} \sqrt{X_i^2 + Y_i^2 + Z_i^2} \tag{3}$$

2.2. Single-Input-Based Color Image Generation

The color-image-generation methods from the LiDAR 2D projected reflection or range image are shown in Figure 2.



(**a**) The conventional LiDAR reflection-based method [20].



(**b**) The proposed LiDAR range-based method.

Figure 2. The color camera-captured-like image generation methods using 2D projected (**a**) reflection and (**b**) range data.

2.2.1. LiDAR Reflection-Based Method

Figure 2a shows the ED-FCN-based color-image-generation method by using only a reflection image proposed in previous works [19,20]. The ED-FCN network generates the RGB color image (size: 592 × 112 × 3) from the sparse 2D reflection image (size: 592 × 112 × 1). In Figure 2, each block in the network is expressed by $(N_f\mathrm{C}N_k\text{-B-}f_a)\times N_b$ where the N_b is the number of convolution blocks and each convolution block is composed of the convolution layer with N_f filters with $(N_k \times N_k)$ kernel size, a batch-normalization [25] layer, and f_a kind of activation function. For example, (16C3-B-Relu) × 6 means six of the convolution blocks, each of which includes the convolution layer of 16 filters with (3 × 3) kernel size, the batch-normalize layer, and the ReLU [26] activation function which is repeated six times. As another example, (3C3-tanh) × 1 means one convolution block including the convolution layer with 3 filters of (3 × 3) size and the tanh [27] activation function. In each convolution layer, stride 1 and zero-padding are commonly applied. For down and up samplings, max-pooling and un-pooling [28] with a factor of 2 are applied, respectively.

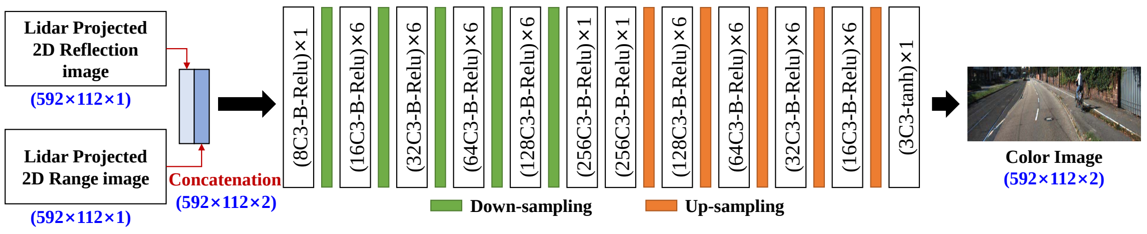
2.2.2. LiDAR Range-Based Method

Figure 2b shows a method which uses the same network of Figure 2a but input is replaced with the range image instead of the reflection image to verify the image generation performance according to the type of LiDAR data.

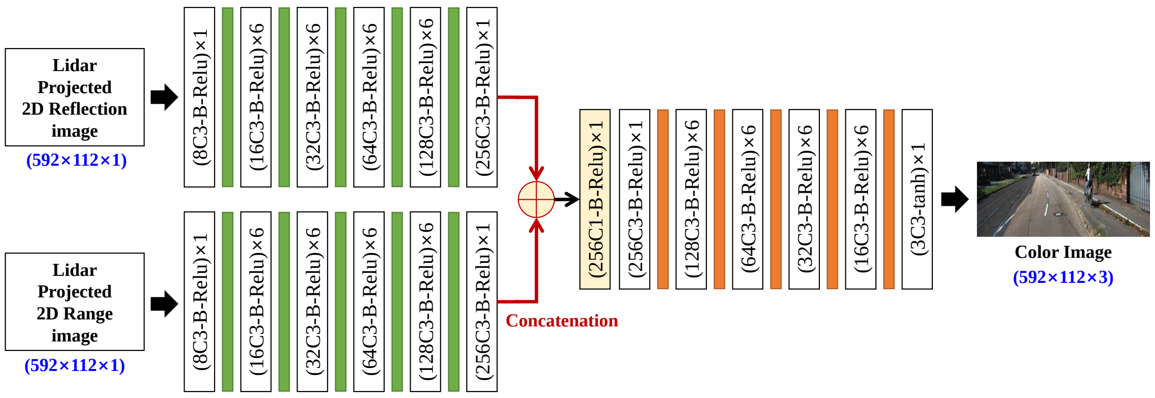
2.3. Proposed Multi-Input-Based Color Image Generation

In applications with two or more multi-input datasets with different characteristics, fusion networks are classified into three types: early fusion [1,2,9,15], mid fusion [3,7–11,29], and last fusion [2,13] methods, depending on where the data are combined together. In the case of the early fusion method, multiple sources are concatenated into a single input before being applied to the image generation network. The mid fusion method merges the features of the intermediate output of each network with different input data. The last fusion method combines the outputs of each network into a final output.

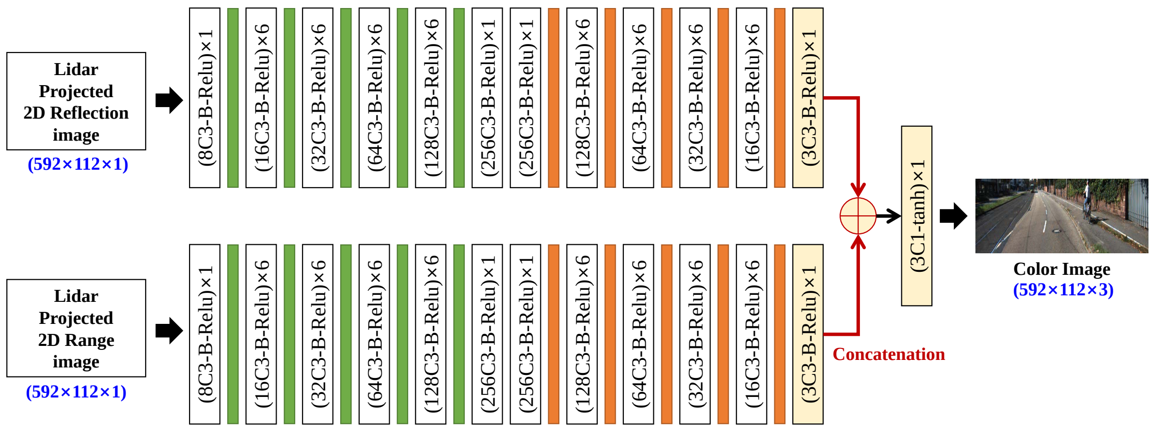
Figure 3 shows the proposed fusion-based LiDAR to color image generation networks using two heterogeneous datasets, i.e., LiDAR reflection and range data, where ED-FCN is used.



(**a**) Early fusion-based LiDAR to color image generation network.



(**b**) Mid fusion-based LiDAR to color image generation network.



(**c**) Last fusion-based LiDAR to color image generation network.

Figure 3. The proposed fusion-based LiDAR to color image generation networks using LiDAR reflection and range data: (**a**) early fusion, (**b**) mid fusion, and (**c**) last fusion-based LiDAR to color image generation networks.

2.3.1. Early Fusion-Based LiDAR to Color Image Generation Network

The projected 2D reflection and range images are concatenated in channel direction to form an input. For instance, reflection and range images with the resolution of (592 × 112 × 1) are combined into a (592 × 112 × 2) concatenated image. The concatenated input is applied to the same network as shown in Figure 2. The overall architecture of the early fusion network is depicted in Figure 3a.

2.3.2. Mid Fusion-Based LiDAR to Color Image Generation Network

For the mid fusion network, as shown in Figure 3b, two encoder networks independently extract the feature-maps from the projected 2D range and reflection images, respectively. The two feature-maps are merged by concatenation operation. Additionally, then, the convolution layers with the 256 (1 × 1) filters, batch-normalize layer, and ReLU activation are repeatedly applied. Finally, the color image is generated by using a decoder network which is the same decoder network used in image generation network with single input.

2.3.3. Last Fusion-Based LiDAR to Color Image Generation Network

For the last fusion network, as shown in Figure 3c, two decoder feature maps are independently extracted from the projected 2D reflection and range images by using single-input-based image generation network. It should be noted that the activation functions of the last convolution layer use ReLU activation, unlike other networks wherein tanh is used. The resultant feature-maps are also merged by using a concatenation. The final color image is generated by using convolution layer with the three (1×1) filters and following tanh activation.

2.4. Training and Inference Processes

We used a dataset [19,20] composed of a triplet of sparse LiDAR 2D projected reflection images, sparse 2D projected range images, and their corresponding dense RGB color images. The projected reflectance and range images were obtained from a LiDAR 3D point cloud with reflection intensity through Equations (2) and (3). The color images synchronized with the reflection and range data were captured from the camera and were used for ground truth (GT) data. In the case of the single-input-based image generation network shown in Figure 2, the 2D reflection image or range image was used. Additionally, in the multi-input-based image generation network shown in Figure 3, both the reflection image and the range image were used as input data of the network. The range and reflection data are normalized into $[0, 1]$, before feeding to the network. The target image of the image generation network, in other words, GT image, is the RGB color image which corresponds to the two LiDAR 2D projected images. As the tanh activation function [27] in the last layer in the image generation networks is used, the dynamic range of generated output data is confined to $[-1, 1]$. The target color images are, therefore, normalized to have the same dynamic range during the training process. As in the previous works [19–21], mean squared error (MSE) is used as a loss function for training.

For training, the proposed network architectures are trained by using adaptive moment estimation (Adam) solver [30] with a maximum of 2000 epochs. All other parameters are the same as those of the Adam solver used in [20]. The early stopping technique with a patience parameter of 25 is applied for validation loss [31].

In the inference process, three-channel output data having the dynamic range $[-1, 1]$ are generated and normalized for the final RGB color images to have the dynamic range of $[0, 255]$.

3. Experimental Results

3.1. Simulation Environment

3.1.1. Evaluation Dataset

The evaluation dataset was reconstituted from the KITTI raw dataset [32], as in [19,20]. The dataset consists of a triplet of images—a projected 2D LiDAR reflection image, a 2D LiDAR range image, a and color image—that were recorded simultaneously. The reflection and range images were used for the input and the color image was used for ground truth. The triplets of the images that were recorded under heavy shadows were manually excluded in the evaluation dataset, because one advantage of the proposed method is to generate shadow-free color images from LiDAR data. The evaluation dataset consisted of 4308 triplets—2872 triplets for training, 718 for validation, and 718 for testing as used in [19,20]. All evaluation images had the same resolution of 592×112 (66,304 pixels). Both the reflection and range images had on average 3502 valid values. In other words, the density of both the projected 2D LiDAR images was 5.28% [21]. Therefore, both the LiDAR images were very sparse and even irregular compared to the target RGB color image.

3.1.2. Measurement Metrics

To evaluate the image quality between the generated color image and target color image, PSNR [23] and SSIM [23,24] were used. In the case of PSNR performance evaluation, since both the generated image and the GT image were in color, PSNR was calculated based on the RGB color image. In other words, PSNRs were separately calculated for each R, G, and B channel, and their average, denoted as PSNR$_c$ was used for the evaluation. Additionally, the gray-scale PSNR, denoted as PSNR$_g$ was measured using only the gray-scale Y component between the generated and the GT images. For the SSIM, only the gray-scale Y component was used.

For the evaluation of the complexity, the average inference computation time and the total number of weights of the network were compared. We used a workstation with an Intel Core i7-6850 CPU 3.60 GHz and a Nvidia Titan X Pascal GPU. The software environments were Ubuntu 16.04, Python 3.5.6, Tensorflow 1.13.1 [33], and Keras 2.3.1 [34].

3.2. Performances of Single-Input-Based Methods

In this section, the performances of the single-image input-based color-image-generation methods are compared. LiDAR reflection-based and LiDAR range-based networks shown in Figure 2a,b were implemented and evaluated respectively. As mentioned in the previous section, the network was the same, but input data were different.

Table 1 shows the results of the image quality from the single-input-based color image generation network according to the input data type. Average PSNR and average SSIM are listed with standard deviation in parenthesis. The LiDAR range input-based image generation can also generate a color image as the reflection input-based method. However, the range-based network has on average 0.67 dB in PSNR$_g$, 0.64 dB in PSNR$_c$, and 0.05 in SSIM less than the reflection-based network, respectively. The gray-scale PSNR (PSNR$_g$) is higher than color-scale PSNR$_c$, irrespective of input data. The results show the possibility of color image generation from LiDAR range data.

Table 1. Generated image quality performance using single-input-based color-image-generation methods in terms of average gray-scale PSNR (PSNR$_g$), color PSNR (PSNR$_c$), and SSIM, where standard deviation is listed in parenthesis.

Input Data	**Validation**			**Test**		
	PSNR$_g$	**PSNR$_c$**	**SSIM**	**PSNR$_g$**	**PSNR$_c$**	**SSIM**
LiDAR reflection	19.57 (±2.49)	19.18 (±2.36)	0.51 (±0.11)	19.53 (±2.47)	19.15 (±2.34)	0.51 (±0.11)
LiDAR range	18.90 (±2.01)	18.55 (±1.95)	0.46 (±0.09)	18.86 (±2.00)	18.51 (±1.94)	0.46 (±0.09)

Table 2 shows the computational complexity according to the input data type in the single-input-based color image generation network. As both input-data-based networks use the same image generation network, the number of weights and average inference computation time were the same, 3,350,243 and 6.91 ms, respectively.

Table 2. Complexity performance results of single-input-based color-image-generation methods in terms of the number of weights and average inference processing time.

Input Data	**The Number of Weights**	**Average Processing Time [ms]**
LiDAR Refection	3,350,243	6.91
LiDAR Range	3,350,243	6.91

3.3. Performances of Proposed Multi-Input-Based Methods

For evaluation of the performances of the multi-input-based color-image-generation methods (here, two-inputs: reflection and range), the three fusion networks shown in Figure 3 were tested and evaluated.

Table 3 shows the results of the image quality performances of the multi-input-based color image generation networks. Like the experimental results in Section 3.2, gray-scale PSNR (PSNR_g) was higher than the color-scale PSNR (PSNR_c), irrespective of fusion type. The results show that the mid fusion-based method had better image generation performance than the early fusion method, and the last fusion was better than the mid fusion. On average, the last fusion method showed higher improvements of 0.35 dB in PSNR_g, 0.33 dB in PSNR_c, and 0.01 in SSIM over the early fusion method, respectively. Among all image generation methods, including single-input and multi-input-based methods, the last fusion method had the best performance. In addition, on average, the last fusion method achieved a color image generation performance of 20.06 dB in PSNR_g, 19.64 dB in PSNR_c, and 0.53 in SSIM.

Table 3. Image quality performance results of the proposed fusion-based color image generation in terms of average gray-scale PSNR (PSNR_g), color PSNR (PSNR_c), and SSIM.

Fusion-Based Network	**Validation**			**Test**		
	PSNR_g	**PSNR_c**	**SSIM**	**PSNR_g**	**PSNR_c**	**SSIM**
Early fusion-based network	19.79 (±2.51)	19.37 (±2.37)	0.52 (±0.11)	19.71 (±2.52)	19.31 (±2.37)	0.52 (±0.11)
Mid fusion-based network	19.96 (±2.46)	19.55 (±2.34)	0.52 (±0.11)	19.91 (±2.44)	19.50 (±2.30)	0.52 (±0.11)
Last fusion-based network	20.10 (±2.62)	19.68 (±2.44)	0.53 (±0.11)	20.06 (±2.61)	19.64 (±2.42)	0.53 (±0.11)

Table 4 shows the computational complexities of the three types of fusion-based networks. The networks used in the early fusion, the mid fusion, and the last fusion-based networks have the about 1.06, 1.45, and 2.00 times of the weights of the single-input-based network, respectively. Additionally, the average computation times of the early fusion, the mid fusion, and the last fusion-based methods are about 1.02, 1.44, and 1.96 times slower than the single-input-based image generation, respectively.

Table 4. Complexity performance results of the proposed fusion-based color image generation in terms of the number of weights and average inference processing time.

Fusion-Based Network	**The Number of Weights**	**Average Processing Time [ms]**
Early fusion-based network	3,550,315	7.03
Mid fusion-based network	4,863,219	9.93
Last fusion-based network	6,700,531	13.56

3.4. Subjective Image Quality Evaluation

For the evaluation of subjective image quality, two representative inference examples are given in Figure 4, where GT color images are shown in the first row, and the second to sixth rows are the images generated by range-based (in the second row) and reflection-based (in the third row) single-input networks and early fusion (in the fourth row), mid fusion (in the fifth row), and last fusion-based multi-input networks (in the sixth row), respectively.

All image generation methods generate blurred images compared to GT images. As shown in the second row of Figure 4, the range-based image generation method generated a somewhat camera-captured-like image. However, the range-based method did not produce color information properly and produced the most blurred image compared to the other methods. The images in the third row show that the reflection-based, single-input image generation can generate color images.

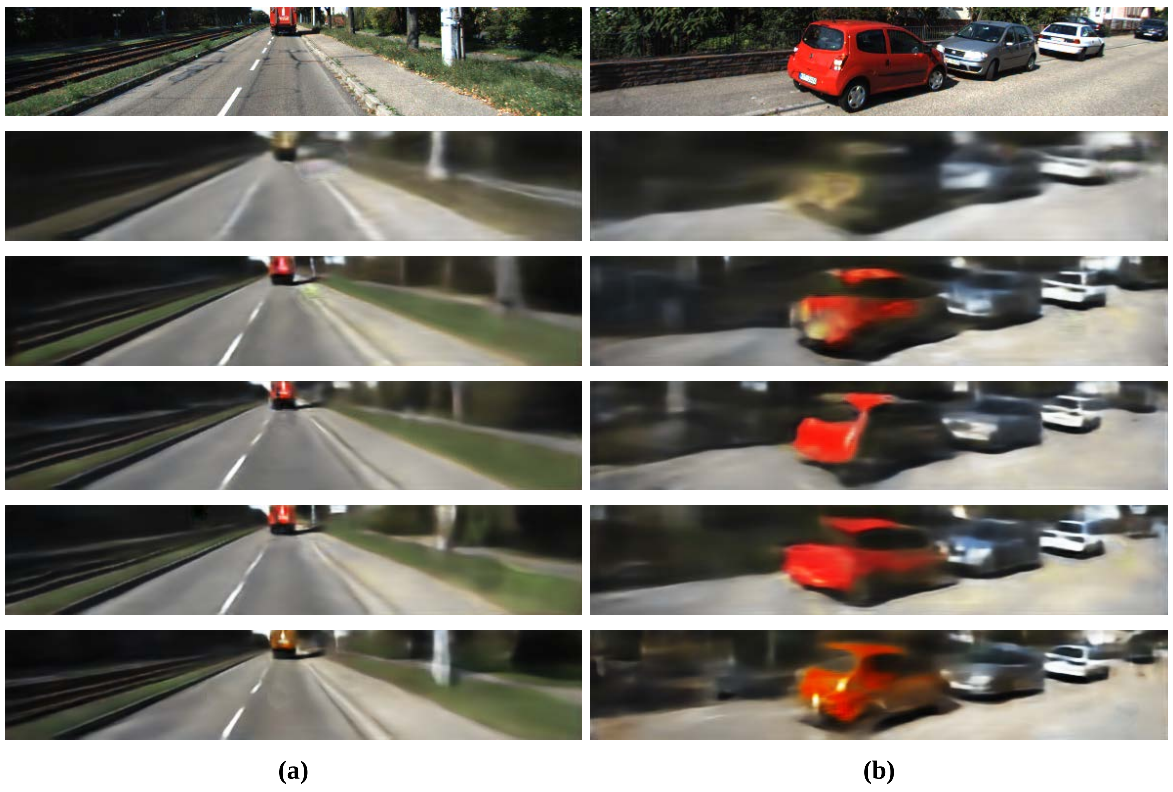


Figure 4. Two representative inference examples. (**a**) shows that the white road-pillar located on the right side does not appear in the reflection-based single-input network. (**b**) shows that the visual qualities are similar to the objective performances shown in Tables 1 and 3. The GT color image is shown in the top row. The second to sixth rows are the images generated by range-based (in the second row) and reflection-based (in the third row) single-input networks and early fusion (in the fourth row), mid fusion (in the fifth row), and last fusion-based multi-input networks (in the sixth row), respectively.

The proposed three fusion methods generate color images more faithfully than reflection-based, single-input image generation. That means the LiDAR range data are clearly useful for the color image generation from LiDAR.

One interesting result is the generation of the white road-pillar on the right side of the image shown in Figure 4a; the reflection-based single-input network could not produce the pillar but the range-based network produced it. That means that the LiDAR range data are helpful in generating geometric structures of objects, thereby producing generated images with better subjective qualities. For the cases of the multi-input networks, the mid fusion and the last fusion networks produced the pillar. In particular, the last fusion-based multi-input network demonstrated faithful generation of the pillar.

Figure 4b shows that the five image generation methods have visual performances similar to the objective performance shown in Tables 1 and 3.

4. Conclusions

In this paper, we examined the effectiveness of the LiDAR range data for camera-like color image generation. LiDAR range data are used as input data in the existing reflection-based single-input color generation network that consists of ED-FCN. Through the simulations, we showed that the range-based single-input method can generate camera-like images, even though the visual quality of the generated image is slightly less than for the reflection-based method. Clearly, the LiDAR range data are useful for LiDAR to color image generation. We considered the use of both LiDAR reflection and range data, and then proposed three kinds of fusion networks based on multiple inputs, i.e., two inputs. The early fusion, mid fusion, and last fusion-based LiDAR-to-color image generation networks were designed and tested.

The proposed last fusion-based method achieved higher improvements of 0.53 dB in PSNR$_g$, 0.49 dB in PSNR$_c$, and 0.02 in SSIM over the previous reflection-based single input method. In addition, the last fusion method is applicable to real-time applications with an average processing time of 13.56 ms.

Therefore, these results show that the fusion of the feature-maps of the decoder networks is better than the fusion of the feature-maps of the encoder networks when the input and output of the image generation network have different characteristics. These results can be applied to various applications, such as object recognition, segmentation, and 3D map-generation using LiDAR data and image generation.

Author Contributions: H.-K.K. performed conception, design, methodology, data collection, implementation, testing, validation, and writing; K.-Y.Y. helped in conception, design, methodology, data analysis, and writing, guided the research; H.-Y.J. supervised the whole research project. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by Basic Science Research Programs of the Ministry of Education (grant NRF-2018R1A2B6005105), in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (number 2019R1A5A8080290), and in part by Ministry of Science, ICT (MSIT), Korea, under the Information Technology Research Center (ITRC) support program (IITP-2020-2016-0-00313) supervised by the IITP(Institute for Information and communications Technology Planning and Evaluation).

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

References

1. Aksoy, E.E.; Baci, S.; Cavdar, S. SalsaNet: Fast Road and Vehicle Segmentation in LiDAR Point Clouds for Autonomous Driving. *arXiv* **2019**, arXiv:1909.08291.
2. Caltagirone, L.; Bellone, M.; Svensson, L.; Wahde, M. LIDAR–camera fusion for road detection using fully convolutional neural networks. *Robot. Auton. Syst.* **2019**, *111*, 125–131. [CrossRef]
3. Chen, Z.; Zhang, J.; Tao, D. Progressive lidar adaptation for road detection. *IEEE/CAA J. Autom. Sin.* **2019**, *6*, 693–702. [CrossRef]
4. Radi, H.; Ali, W. VolMap: A Real-time Model for Semantic Segmentation of a LiDAR surrounding view. *arXiv* **2019**, arXiv:1906.11873.
5. Gao, Y.; Zhong, R.; Tang, T.; Wang, L.; Liu, X. Automatic extraction of pavement markings on streets from point cloud data of mobile lidar. *Meas. Sci. Technol.* **2017**, *28*, 085203. [CrossRef]
6. Wurm, K.M.; Kümmerle, R.; Stachniss, C.; Burgard, W. Improving robot navigation in structured outdoor environments by identifying vegetation from laser data. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009.
7. Du, X.; Ang, M.H.; Karaman, S.; Rus, D. A general pipeline for 3d detection of vehicles. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018.
8. Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; Waslander, S.L. Joint 3d proposal generation and object detection from view aggregation. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
9. Liang, M.; Yang, B.; Chen, Y.; Hu, R.; Urtasun, R. Multi-task multi-sensor fusion for 3d object detection. In Proceedings of the 2019 Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
10. Liang, M.; Yang, B.; Wang, S.; Urtasun, R. Deep continuous fusion for multi-sensor 3d object detection. In Proceedings of the European Conference on Computer Vision (ECCV) 2018, Munich, Germany, 8–14 Sptember 2018.

11. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3d object detection from rgb-d data. In Proceedings of the 2018 Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 918–927.
12. Reymann, C.; Lacroix, S. Improving LiDAR point cloud classification using intensities and multiple echoes. In Proceedings of the IROS 2015—IEEE/RSJ International Conference on Intelligent Robots and Systems, Hamburg, Germany, 28 September–3 October 2015.
13. Vora, S.; Lang, A.H.; Helou, B.; Beijbom, O. PointPainting: Sequential Fusion for 3D Object Detection. *arXiv* **2019**, arXiv:1911.10150.
14. Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; Jia, J. Std: Sparse-to-dense 3d object detector for point cloud. In Proceedings of the IEEE International Conference on Computer Vision 2019, Seoul, Korea, 27 October–2 November 2019.
15. Piewak, F.; Pinggera, P.; Schafer, M.; Peter, D.; Schwarz, B.; Schneider, N.; Enzweiler, M.; Pfeiffer, D.; Zollner, M. Boosting lidar-based semantic labeling by cross-modal training data generation. In Proceedings of the European Conference on Computer Vision (ECCV) 2018, Munich, Germany, 8–14 September 2018.
16. Riveiro, B.; Díaz-Vilariño, L.; Conde-Carnero, B.; Soilán, M.; Arias, P. Automatic segmentation and shape-based classification of retro-reflective traffic signs from mobile LiDAR data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *9*, 295–303. [CrossRef]
17. Tatoglu, A.; Pochiraju, K. Point cloud segmentation with LIDAR reflection intensity behavior. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012.
18. Zhao, X.; Yang, Z.; Schwertfeger, S. Mapping with Reflection–Detection and Utilization of Reflection in 3D Lidar Scans. *arXiv* **2019**, arXiv:1909.12483.
19. Kim, H.K.; Yoo, K.Y.; Park, J.H.; Jung, H.Y. Deep Learning Based Gray Image Generation from 3D LiDAR Reflection Intensity. *IEMEK J. Embed. Sys. Appl.* **2019**, *14*, 1–9.
20. Kim, H.K.; Yoo, K.Y.; Park, J.H.; Jung, H.Y. Asymmetric Encoder-Decoder Structured FCN Based LiDAR to Color Image Generation. *Sensors* **2019**, *19*, 4818. [CrossRef] [PubMed]
21. Kim, H.K.; Yoo, K.Y.; Jung, H.Y. Color Image Generation from LiDAR Reflection Data by Using Selected Connection UNET. *Sensors* **2020**, *20*, 3387. [CrossRef] [PubMed]
22. Isola, P.; Zhu, J.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017, Honolulu, HI, USA, 21–26 July 2017.
23. Hore, A.; Ziou, D. Image quality metrics: PSNR vs. SSIM. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010.
24. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
25. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
26. Dahl, G.E.; Sainath, T.N.; Hinton, G.E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013.
27. Kalman, B.L.; Kwasny, S.C. Why tanh: Choosing a sigmoidal function. In Proceedings of the IJCNN International Joint Conference on Neural Networks, Baltimore, MD, USA, 7–11 June 1992.
28. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the European Conference on Computer Vision 2014, Zurich, Switzerland, 6–12 September 2014; pp. 818–833.
29. Yang, G.; Zhao, H.; Shi, J.; Deng, Z.; Jia, J. Segstereo: Exploiting semantic information for disparity estimation. In Proceedings of the European Conference on Computer Vision (ECCV) 2018, Munich, Germany, 8–14 September 2018.
30. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
31. Prechelt, L. Automatic early stopping using cross validation: Quantifying the criteria. *Neural Netw.* **1998**, *11*, 761–767. [CrossRef]
32. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]

33. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for large-scale machine learning. *arXiv* **2016**, arXiv:1605.08695.
34. Keras. Available online: https://keras.io (accessed on 8 October 2019).