DeepLabV3-Refiner-Based Semantic Segmentation Model for Dense 3D Point Clouds

Jeonghoon Kwak and Yunsick Sung*

Department of Multimedia Engineering, Dongguk University-Seoul, 30 Pildong-ro 1-gil, Jung-gu, Seoul 04620, Korea; jeonghoon@dongguk.edu
* Correspondence: sung@dongguk.edu; Tel.: +82-2-2260-3338

Abstract: Three-dimensional virtual environments can be configured as test environments of autonomous things, and remote sensing by 3D point clouds collected by light detection and range (LiDAR) can be used to detect virtual human objects by segmenting collected 3D point clouds in a virtual environment. The use of a traditional encoder-decoder model, such as DeepLabV3, improves the quality of the low-density 3D point clouds of human objects, where the quality is determined by the measurement gap of the LiDAR lasers. However, whenever a human object with a surrounding environment in a 3D point cloud is used by the traditional encoder-decoder model, it is difficult to increase the density fitting of the human object. This paper proposes a DeepLabV3-Refiner model, which is a model that refines the fit of human objects using human objects whose density has been increased through DeepLabV3. An RGB image that has a segmented human object is defined as a dense segmented image. DeepLabV3 is used to make predictions of dense segmented images and 3D point clouds for human objects in 3D point clouds. In the Refiner model, the results of DeepLabV3 are refined to fit human objects, and a dense segmented image fit to human objects is predicted. The dense 3D point cloud is calculated using the dense segmented image provided by the DeepLabV3-Refiner model. The 3D point clouds that were analyzed by the DeepLabV3-Refiner model had a 4-fold increase in density, which was verified experimentally. The proposed method had a 0.6% increase in density accuracy compared to that of DeepLabV3, and a 2.8-fold increase in the density corresponding to the human object. The proposed method was able to provide a 3D point cloud that increased the density to fit the human object. The proposed method can be used to provide an accurate 3D virtual environment by using the improved 3D point clouds.

Keywords: dynamic object; deep learning; encoder-decoder model; 3D point cloud; LiDAR; data fusion; dense semantic segmentation

1. Introduction

Recently, 3D virtual environments have been built and used in fields that require learning or verification in various situations, such as autonomous driving or remote monitoring [1–4]. The virtual environment can be reconstructed to provide situations that can occur in the real environment using the 3D point cloud measured by light detection and range (LiDAR) [5–8]. The movement of virtual objects, such as a person, is expressed by using divided moving objects based on the measured 3D point cloud [9–11]. In order to control these objects, a pose predicted as a segmented 3D point cloud is used [12–14]. When performing the segmentation, a problem arises, wherein each human object’s 3D point cloud density decreases because of the spacing of the lasers contained in the LiDAR [15–17]. When the density of the segmented 3D point cloud decreases, it is difficult to provide detailed changes to the shape of a moving object.

In order to increase the density of the 3D point cloud of each segmented human object, the learned encoder-decoder model is applied, which uses a 3D model that provides high density [18,19]. However, it is difficult to account for changing shapes, such as clothes. In
order to increase the density, an RGB image captured by a camera is used [20–24]. However, if the 3D point cloud of the human object is not input into an encoder-decoder model, such as DeepLabV3 [21], the density of the 3D point cloud cannot be accurately increased. In the process of extracting the features of a human’s 3D point cloud with the encoder, not only the features of the human object’s 3D point cloud but also the features of the 3D point cloud around the human object are included. Increasing the density with a decoder based on the features extracted by the encoder is problematic, since a suitable 3D point cloud is not provided to the human object. Depth values are not included in an RGB image, which requires conversion to a 3D coordinate system using the measured 3D point cloud.

This paper proposes a method of automatically segmenting the 3D point cloud for a human object by analyzing the 3D point cloud measured by LiDAR and increasing density using a DeepLabV3-Refiner model; in the learning process, the method is taught to segment with respect to the collected 3D point cloud of a human object and generate a dense segmented image using an RGB image. In the execution process, the input 3D point cloud is analyzed using the learned DeepLabV3-Refiner model, and a dense segmented image with increased density is formed. The dense segmented image is then converted to a 3D coordinate system.

The contributions of this paper through the proposed method are as follows: it is possible to automatically find human objects from measured 3D point clouds with increased density. In the process of extracting features for segmentation of human objects from measured 3D point clouds, not only human objects but also 3D point clouds around the human objects are included. Using the proposed method, removing noise included in the features significantly increases density.

This paper is organized as follows: In Section 2, related research on the segmentation of human objects included in the 3D point clouds and related research on increasing their density are introduced. A process utilizing a DeepLabV3-Refiner model to provide a high-density 3D point cloud of a human from one measured by LiDAR is proposed. In Section 3, the DeepLab-V3-Refiner model is verified. Finally, in Section 4, conclusions about the proposed method and future research are introduced.

2. Materials and Methods

Related materials are described to suggest the proposed method. A method to increase the density of human objects in a 3D point cloud by improving the disadvantages of the materials is introduced.

2.1. Materials

Methods for segmenting a 3D point cloud based on an encoder-decoder model are introduced. Methods for increasing the density of a 3D point cloud using an encoder-decoder model are described.

2.1.1. Segmentation Method

Encoder-decoder models are used for segmentation [25–29]. The object is segmented by using the 3D point cloud measured by LiDAR either as is or by preprocessing it based on voxel. The voxel reconstructs the collected 3D point cloud into values in regular grid units in 3D space. RGB images are used to increase the segmentation performance [30–33]. Since the 3D point cloud and RGB images are used together for segmentation, it and the density are improved. However, the segmentation performance decreases in an environment without lighting, and there is need of a method using only the 3D point cloud.

2.1.2. Increasing the Density of the 3D Point Cloud

Three-dimensional point clouds measured by multiple LiDARs can improve density [34]. By matching the measured features, the 3D point clouds are merged. However, this does not improve parts that are not measured by the laser gap attached to the LiDAR,
and it increases the cost and computing time. Thus, there is a demand for a way to increase the density without using additional LiDARs.

The density can be increased by analyzing the collected 3D point clouds and generating additional ones using a patch-based upsampling network [35], which is used to generate a high-resolution 3D point cloud from other low-resolution 3D point clouds. However, since it is not possible to provide all 3D models with various changes in appearance, such as in the case of humans, the generated 3D point cloud differs from the actual appearance. Thus, there is need of a method that robustly increases the density despite various changes in appearance.

Density can be increased by using an RGB image [23,24] corresponding to a 3D point cloud or a 3D model [18,19,36] generated based on a 3D point cloud. The input of an encoder-decoder model is composed of a 3D point cloud, and the output is composed of a 3D point cloud with increased density. However, in the encoder-decoder model, a 3D point cloud with increased density is generated by utilizing the features of the 3D point cloud. When an unsegmented 3D point cloud is input with an object to increase density, its details cannot be expressed because of other features around the object. There is need of a method that provides a dense 3D point cloud for only the object.

In the proposed method, the 3D point cloud around the object is removed and an increased density suitable for the object is provided. Since the process of segmenting the object to increase density is included in the encoder-decoder model, segmenting the object to increase density from the collected 3D point cloud can be omitted.

2.2. Methods

The DeepLabV3-Refiner model used to fit the human object is introduced. The preprocessing method for inputting a 3D point cloud into the DeepLabV3-Refiner model, which is learned by utilizing an RGB image, is described. A postprocessing process for converting the result of the DeepLabV3-Refiner model into a 3D coordinate system is also described.

2.2.1. Overview

The process is divided into a learning process and an execution process, as shown in Figure 1. The learning process consists of learning the DeepLabV3-Refiner model, preprocessing to convert a collected 3D point cloud into a depth image, and preprocessing to extract a segmented image from a collected RGB image. The depth image constructed by preprocessing the 3D point cloud measured in LiDAR is defined as the 3D point cloud $P_t$. The dense segmented image of the human object in an RGB image is defined as $D_t'$. The result of increasing the density of the 3D point cloud with the DeepLabV3-Refiner model is defined as a dense segmented image $D_t$ and has components identical to $D_t'$. The DeepLabV3-Refiner model is taught to convert $P_t$ into $D_t'$.

The execution process uses the learned DeepLabV3-Refiner model to construct a 3D point cloud with increased density. In order to input the learned DeepLabV3-Refiner model, the 3D point cloud is preprocessed. A dense segmented image $D_t$ is constructed from the learned DeepLabV3-Refiner model using $P_t$. The post-processed image is defined as a dense 3D point cloud to convert $D_t$ into a 3D coordinate system.

As shown in Figure 2, in the learning process and execution process, LiDAR is used to measure the human. An RGB camera, which is only used in the learning process, is used to provide a dense 3D point cloud.
As shown in Figure 2, in the learning process and execution process, LiDAR is used to measure the human. An RGB camera, which is only used in the learning process, is used to provide a dense 3D point cloud.

Figure 1. Process of increasing the density of the 3D point cloud for human objects in the learning and execution processes.

Figure 2. A device that measures the human in the learning and execution processes.
2.2.2. Preprocessing of 3D Point Cloud and RGB Image

A preprocessing method for learning and executing a DeepLabV3-Refiner model is introduced, as shown in Figure 3. The 3D point cloud collected from LiDAR is configured as $P_t$ for input into the DeepLabV3-Refiner model [37]. Then, $P_t$ is constructed by projecting the 3D point cloud measured through LiDAR onto a 2D plane. A 3D point cloud is formed with 3D coordinates measured for the surrounding environment as the time when the lasers configured in LiDAR arrive after being reflected from objects. Then, $P_t$ projects a 3D point cloud measured by LiDAR as a 2D plane, and the place where the 3D point cloud is not measured is set to 0, and it approaches 255 as it gets closer to the LiDAR.

![Preprocessing for 3D Point Cloud](image)

**Figure 3.** Preprocessing the collected 3D point cloud and collected RGB image.

The human object is extracted from the RGB image captured by the camera for preprocessing, forming $D'_t$. It is possible and simple to use an existing deep learning model such as DeepLabV3 [21] for extraction. Finally, $D'_t$ has the same size as $P_t$.

2.2.3. Generation for Segmentation Image

The goal of using a segmented image is to teach the DeepLabV3-Refiner model to provide a dense 3D point cloud with that measured by LiDAR, as shown in Figure 4. The depth image for the preprocessed 3D point cloud is input to the learned DeepLabV3-Refiner model. The base model provides a 3D point cloud for the human object with roughly high density based on the preprocessed 3D point cloud. The features of the human object’s 3D point cloud include the 3D point cloud around the human object. In order to remove the noise of the added features, the refined model provides increased density by fitting the 3D point cloud to that for the inferred human object.
Figure 4. Proposed DeepLabV3-Refiner model architecture. The base model provides segmentation for a 3D point cloud and with roughly increased density. In the refiner model, the 3D point cloud with increased density in the base model is refined to fit the human object.

If the environment to be located is input, the accuracy of the dense 3D point cloud can be improved. In advance, a 3D point cloud is collected for the environment where the human will be located, and the depth image is defined as background B. Then, P_t and B project the measured 3D point cloud onto a 2D plane to form a depth image, similar to the 3D point cloud preprocessing. In an environment where LiDAR does not move, it is possible to increase the accuracy of the segmented dense 3D point cloud by additionally inputting B.

The coarse output consists of a 3D point cloud with increased density and a 3D point cloud for the segmented human object. A 3D point cloud that is half its original size and has increased density is defined as a coarse dense segmented image, S^C_t. The base model of the DeepLabV3-Refiner model infers coarse output based on a preprocessed 3D point cloud.

The model architecture of the proposed DeepLabV3-Refiner model is described. The base model is used as a fully-convolutional encoder-decoder model, which is the architecture of DeepLabV3 [21,22], and provides an effective structure for semantic segmentation. The segmentation model N^S is composed of a backbone, atrous spatial pyramid pooling (ASPP), and decoder structures. The backbone is configured based on the encoder and utilizes ResNet-50 [38], which can be utilized by ResNet-100 to increase accuracy, or by replacing it with MobileNetV2 [39], you can increase speed. ASPP is the method proposed by DeepLabV3, the result of the backbone, which is robust to multi-scaling and derives the features of the 3D point cloud used for segmentation. The decoder performs upsampling to compose S^C_t and S^P_t based on features derived from ASPP.

The base model is configured as shown in Table 1. The backbone consists of ResNet-50, and the ASPP’s dilation rates are set to 3, 6, and 9. The decoder consists of 4 convolutions of 288, 152, 80, and 4. In each convolution, the intermediate result of the backbone is additionally input.
The model used to increase the density to fit the segmented 3D point cloud is defined as the refiner model, where the dense segmented image $D_t$ is constructed using the input 3D point cloud and coarse output. The refiner model is responsible for configuring the dense 3D point cloud to fit the human object and performs fine-tuning using both $P_t$ and the coarse output predicted from the base model; it decides whether the roughly composed $S^C_t$ fits $D_t$ and also removes noise included in $S^C_t$. In order to remedy the fact that the original information is removed, the 3D point cloud input to the base model is used. Based on the extracted features, it up-samples to the original resolution, and finally, a dense segmented image $D_t$ is formed.

The refiner model is composed as shown in Table 2. The 3D point cloud and coarse output input before the first convolution are concatenated. Each of the convolutions consists of 6, 24, 16, 12, and 2 filters. After the second convolution, upsampling is performed to fit the original size. The input 3D point cloud and upsampling result are concatenated for comparison. The remaining convolution proceeds using the concatenated result.

A method for learning the proposed DeepLabV3-Refiner model is described, wherein it learns by transforming input data with actions such as horizontal flip, random noise, resize, and crop, which occur in real situations.

The ground truth for $S^P_t$ is defined as segmented 3D point cloud $S^{P}_t$, which is configured by designating the 3D point cloud corresponding to the human object in $P_t$ reduced to half its size. The ground truth for $D_t$ is defined as $D^{P}_t$, which consists of an RGB image that provides a high density image for the 3D point cloud. The ground truth of $S^C_t$ is defined as $S^C_t$, which is constructed by roughly increasing the density by halving the size of $D^{P}_t$. The loss $L_{\text{segmentation}}$ for $S^P_t$ is calculated as Equation (1).

$$L_{\text{segmentation}} = ||S^P_t - S^{P}_t||^2$$  \hspace{1cm} (1)

Loss $L_{\text{coarse}}$ of the coarse dense segmented image $S^C_t$ is calculated as Equation (2).

$$L_{\text{coarse}} = ||S^C_t - S^{C}_t||^2$$  \hspace{1cm} (2)

Loss $L_{\text{refine}}$ of the dense segmented image $D_t$ is calculated as Equation (3).

$$L_{\text{refine}} = ||D_t - D^{P}_t||^2$$  \hspace{1cm} (3)
Finally, the loss for the DeepLabV3-Refiner model is shown in Equation (4). The DeepLabV3-Refiner model is taught based on $L_{total}$.

$$L_{total} = L_{segmentation} + L_{coarse} + L_{refine}$$ (4)

A dense segmented image $D_t$ is formed by inputting a 3D point cloud $P_t$ using the learned encoder-decoder model.

2.2.4. Postprocessing for Dense 3D Point Cloud

A method for constructing a dense 3D point cloud using a dense segmented image with the learned DeepLabV3-Refiner model is introduced, as shown in Figure 5. A dense 3D point cloud is composed of a depth image that can express 3D coordinates such as $P_t$. The dense 3D point cloud is constructed using $P_t$ and $D_t$. In $D_t$, the dense 3D point cloud is calculated as the average of $P_t$ around the pixels set to 255.

![Dense Segmentation Image $D_t$](image)

![3D Point Cloud $P_t$](image)

*Figure 5. Postprocessing to construct a dense 3D point cloud by utilizing the collected 3D point cloud and dense segmented image.*

3. Experiment

The result of increasing the density of the collected 3D point cloud was analyzed using the proposed DeepLabV3-Refiner model. The dataset used in the experiment is explained in this section, and the results learned using the DeepLabV3-Refiner model are detailed.

3.1. Dataset and Preprocessing Results

The KITTI dataset [40, 41] was used in this experiment, which is a dataset provided for the learning necessary for autonomous vehicles. The 3D point cloud and RGB images of objects around the vehicle are measured through LiDAR and a camera attached to the vehicle. The LiDAR attached to the vehicle was rotated 360° around the vehicle using the HDL-64E, and a 3D point cloud was measured. In addition, an RGB camera was installed so that RGB images could be measured. A depth image of $850 \times 230$ was constructed using the 3D point cloud measured with HDL-64E, which was used as a 3D point cloud in a position between $-24.9$ degrees and 2 degrees for a horizontal angle and $-90$ to 90 degrees for a vertical angle. The RGB image was composed in the same manner by matching the position difference between the LiDAR and RGB camera. A dataset was provided by cropping the composed depth image and RGB image at $230 \times 230$, centered on the location of the human, as shown in Figure 6.
Using the difference between the depth image without the human and the measured depth image, a segmented 3D point cloud was constructed and resized by half, as shown in Figure 6c. The image of the human in the RGB image was segmented, and a dense segmented image was provided based on the segmented area as shown in Figure 6d. The dense segmented image in Figure 6d was resized by half and designated as the coarse dense segmented image. The density of Figure 6d was higher than that of the actual human’s 3D point cloud in Figure 6c.

3.2. Training Results of Generation for the Segmentation Image

Table 3 shows the proposed DeepLabV3-Refiner model and DeepLabV3 [21], which estimates a dense segmented image using the collected depth image. The accuracy was calculated as the similarity between the dense segmented image estimated by the 3D point cloud collected by LiDAR and the dense segmented image estimated by the RGB image. In the case of the proposed DeepLabV3-Refiner model, when the background is also input, the accuracy is 92.38%, and when the background is not used, the accuracy is 91.75%. When the learning process included the background, the dense segmented images improved by 0.63%. It also was different for the overall loss. In the case of DeepLabV3, the accuracy was 91.74% when the background was added. When the background was not added, the accuracy increased by 0.34% to 91.40%. It was confirmed that the accuracy of the refined, dense 3D point cloud can be improved if the background is available.

Table 3. Training results of each model.

| Model                        | Loss       | Accuracy |
|------------------------------|------------|----------|
| DeepLabV3 [21]              | 0.1103     | 91.40%   |
| DeepLabV3 [21] (Use Background) | 0.1039    | 91.74%   |
| Proposed Method             | 0.1464 ($L_{total}$) | 91.75% |
|                             | 0.1053 ($L_{refine}$) |         |
|                             | 0.00004 ($L_{segmentation}$) |       |
|                             | 0.0410 ($L_{coarse}$) |         |
| Proposed Method (Use Background) | 0.1294 ($L_{total}$) | 92.38% |
|                             | 0.0917 ($L_{refine}$) |         |
|                             | 0.00001 ($L_{segmentation}$) |   |
|                             | 0.0376 ($L_{segmentation}$) |       |

Even when the background of the proposed method was not entered, the performance was similar to that of the case without the background in DeepLabV3. However, although the accuracy is similar, the difference in the performance of the dense segmented image can be confirmed, as shown in Table 4.
| Index | Collected RGB Image | Collected 3D Point Cloud | Ground Truth | Difference Image-Based Density Correction [20] |
|-------|---------------------|--------------------------|--------------|-----------------------------------------------|
| 1     | ![Collected RGB Image](image1.png) | ![Collected 3D Point Cloud](image2.png) | ![Ground Truth](image3.png) | ![Difference Image-Based Density Correction](image4.png) |
| 2     | ![Collected RGB Image](image5.png) | ![Collected 3D Point Cloud](image6.png) | ![Ground Truth](image7.png) | ![Difference Image-Based Density Correction](image8.png) |
| 3     | ![Collected RGB Image](image9.png) | ![Collected 3D Point Cloud](image10.png) | ![Ground Truth](image11.png) | ![Difference Image-Based Density Correction](image12.png) |
| 4     | ![Collected RGB Image](image13.png) | ![Collected 3D Point Cloud](image14.png) | ![Ground Truth](image15.png) | ![Difference Image-Based Density Correction](image16.png) |
| 5     | ![Collected RGB Image](image17.png) | ![Collected 3D Point Cloud](image18.png) | ![Ground Truth](image19.png) | ![Difference Image-Based Density Correction](image20.png) |
Table 4. Dense segmentation image result of each method.

| Index | Collected RGB Image | Collected 3D Point Cloud | Ground Truth | Difference Image-Based Density Correction [20] |
|-------|----------------------|---------------------------|--------------|-----------------------------------------------|
| 6     |                      |                           |              |                                               |

| Index | DeepLabV3 [21] | Proposed Method |
|-------|----------------|-----------------|
|       | (Using Background) | (Using Background) |
| 1     |                 |                 |
| 2     |                 |                 |
| 3     |                 |                 |
| 4     |                 |                 |
Table 4. Cont.

| Index | DeepLabV3 [21] | DeepLabV3 [21] (Using Background) | Proposed Method | Proposed Method (Using Background) |
|-------|----------------|-----------------------------------|-----------------|-----------------------------------|
| 5     | ![Image](image1) | ![Image](image2)                  | ![Image](image3) | ![Image](image4)                  |
| 6     | ![Image](image5) | ![Image](image6)                  | ![Image](image7) | ![Image](image8)                  |

3.3. Execution Results of Generation for the Segmentation Image

Table 4 shows the results of constructing a dense segmented image using the learned proposed DeepLabV3-Refiner model. Similar results for the proposed method and the existing method can be found in Appendix A, Table A1. When the background was used in the proposed method, it was confirmed that the dense segmented image became even denser and was most similar to the human object. Information in the 3D point clouds collected, such as that for an arm, can be expressed. If the background of the proposed method was not added, there was a problem with some of the 3D point clouds collected such as those for a neck, arm, and leg. In the case of DeepLabV3, it was confirmed that a larger dense segmented image appeared compared to the collected 3D point cloud. It is predicted that a dense segmented image will be set as features of the surrounding 3D point cloud. When the density was corrected with the different image-based density correction [20], it was confirmed that the human object’s shadow, clothes, and the background were similar, and noise was generated by the shaking of the RGB camera.

As shown in the third picture in Table 4, there was no distinction between the human’s legs in the ground truth. However, in the proposed method, it was possible to distinguish between the two legs. The first and second pictures of Table 4 confirm that the proposed method using the background can increase the density of the arm. However, if the background is not added to the proposed method, the arm cannot be expressed if the number of collected 3D point clouds is small. In the fourth picture, the arm was raised in front, but the number of 3D point clouds was small, so it could not be expressed in methods using deep learning.

Table 5 shows the average number of 3D point clouds that have increased density through the learned model from those measured. From a total of 1,000 verification data points, 617,407 points were collected from the human object. If the distance difference between pixels in the collected human’s 3D point cloud was within 5, the correct 3D point cloud was calculated, and if greater than 5, it was calculated as an inappropriate 3D point cloud, and the 3D point cloud was verified if the density suitable for the human object was increased.
Table 5. Performance evaluation of dense segmentation image.

| Method                                      | Number of 3D Point Clouds for Human Object | Number of 3D Point Clouds | Ratio with of Number of 3D Point Clouds for Human Object |
|---------------------------------------------|-------------------------------------------|---------------------------|----------------------------------------------------------|
|                                             | Suitable 3D Point Clouds                  | Unsuitable 3D Point Clouds| Suitable 3D Point Clouds                                 |
| Difference Image-Based Density Correction [20] | 1,937,792                                 | 2,173,354                 | 3.138 3.520                                               |
| DeepLabV3 [21]                              | 3,116,982                                 | 1,1407,208                | 5.048 2.279                                               |
| DeepLabV3 [21] (Using Background)           | 2,710,732                                 | 459,594                   | 4.390 0.744                                               |
| Proposed Method                             | 2,063,935                                 | 342,018                   | 3.343 0.553                                               |
| Proposed Method (Using Background)          | 2,901,549                                 | 324,842                   | 4.699 0.526                                               |

In the case of the proposed method, the density of the 3D point cloud increased by about 4.6 times when the background was included and by 3.3 times when the background was not included. In the case of including the background using DeepLabV3, the reason that the difference was 0.3 times that of the result including the background with the proposed method, is that the background was set to be larger than the human object and the distance difference between pixels was within 5, so there was no significant difference. However, it was also confirmed that there were 0.2 times more cases where the distance difference between pixels was measured to be larger than 5. In the case where the background is not included in the proposed method, the density increase rate is low because the density is higher than that of the human’s 3D point cloud. In DeepLabV3, when the background was not included, the density increased by about 5 times, but it increased so that the human’s pose could not be confirmed. It was confirmed that as the density increased, the number of incorrect 3D point clouds also increased. In the case of the difference image-based density correction [20], it was confirmed that the 3D point cloud suitable for the human object was not generated better than that generated by the proposed method, and the noise of the 3D point cloud was increased by 3.5 times.

In addition, it was confirmed that the execution speed of the method proposed by GTX1050Ti is approximately 0.09 s faster than that of DeepLabV3, as shown in Table 6. The speed was reduced because the features were extracted by reducing them by 1/4 in the segmentation model. In each method, 0.01 s was added when the background was included. In order to increase the accuracy, ResNet-100 can be used instead of ResNet-50, and MobileNetV2 [39] can be used to increase the speed.

Table 6. Inference time for the encoder-decoder models.

| Method                                      | Inference Time (s) |
|---------------------------------------------|--------------------|
| DeepLabV3 [21]                              | 0.22               |
| DeepLabV3 [21] (Using Background)           | 0.23               |
| Proposed Method                             | 0.13               |
| Proposed Method (Using Background)          | 0.14               |

3.4. Postprocessing Results

The dense 3D point cloud provided after postprocessing using the dense segmentation image is shown in Table 7, which can be expressed to have a depth value similar to that of the collected 3D point cloud. In the dense 3D point cloud, the 3D point cloud for the
non-human object part has been removed, and the density of the 3D point cloud for the human object is increased.

Table 7. Dense 3D point cloud result.

| Index | Collected RGB Image | Collected 3D Point Cloud | Proposed Method | Proposed Method (Using Background) |
|-------|---------------------|--------------------------|-----------------|-------------------------------------|
| 1     | ![Image](image1.png) | ![Cloud1.png]             | ![Proposed1.png] | ![ProposedBackground1.png]          |
| 2     | ![Image](image2.png) | ![Cloud2.png]             | ![Proposed2.png] | ![ProposedBackground2.png]          |
| 3     | ![Image](image3.png) | ![Cloud3.png]             | ![Proposed3.png] | ![ProposedBackground3.png]          |
| 4     | ![Image](image4.png) | ![Cloud4.png]             | ![Proposed4.png] | ![ProposedBackground4.png]          |
| 5     | ![Image](image5.png) | ![Cloud5.png]             | ![Proposed5.png] | ![ProposedBackground5.png]          |
4. Discussion

In the DeepLabV3-Refiner model, the dense segmentation image for the human object could be provided by removing the parts with increased density in the non-human object based on the coarse dense segmentation image inferred from the base model. The reason why it is possible to remove the non-human object parts from the refined model is that the segmented 3D point cloud for the user inferred from the base model can be used, so a dense segmentation image can be provided to fit the human object. In addition, if it is possible to provide a 3D point cloud for the environment in which a human is located, it is possible to infer the 3D point cloud for the human object with higher accuracy than when there is no environment, so it is possible to improve the dense segmentation image.

DeepLabV3-Refiner model affected segmented human objects based on the RGB image provided by ground truth, which provided dense segmentation images to non-human object parts. The reason why dense segmentation images were provided for non-human object parts is because some dense segmentation images include non-human object parts in the ground truth. If the dense segmentation image of the ground truth is improved, it will be possible to improve the dense segmentation image to fit the human object.

The new model can be used to construct a 3D virtual environment by using a 3D point cloud with increased density. If a 3D virtual environment is constructed using a 3D point cloud with increased density, it will be possible to provide a 3D point cloud that expresses the human’s external changes in more detail.

5. Conclusions

This paper proposed a DeepLabV3-Refiner model to segment a 3D point cloud of a human using one collected by a LiDAR to increase the density. Through the segmentation model of the proposed DeepLabV3-Refiner model, the portions corresponding to human objects were found to have automatically increased density to fit the human objects. The result of the learned DeepLabV3-Refiner model was converted into 3D coordinates to provide a dense 3D point cloud. Thus, it was confirmed that only human objects increase the density of the 3D point cloud collected by LiDAR when measuring an environment including the human and provide a higher density of 3D point cloud.

In the experiment, the density of 3D point clouds collected using the KITTI dataset was increased. A human object was segmented from the 3D point clouds measured based on the DeepLabV3-Refiner model, and their densities were also increased. A 0.2% increase was observed when the background was included compared with the increase seen by using DeepLabV3 [21], and 1.7% of the noise was removed when the background was not included. Further research is needed to predict the human object’s pose by using a 3D point cloud with increased density. In addition, a method is required to increase the density of 3D point clouds measured in LiDAR in real time.
Author Contributions: Methodology, J.K., Y.S.; software, J.K.; validation, J.K.; writing—original draft preparation, J.K.; writing—review and editing, Y.S.; visualization, J.K.; funding acquisition, Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the High-Potential Individuals Global Training Program) (2019-0-01585, 2020-0-01576) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The KITTI Dataset [40,41] used for this study can be accessed at http://www.cvlibs.net/datasets/kitti/ (accessed on 17 April 2021).

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

Appendix A

Table A1. Dense segmentation image result of each method.

| Index | Collected RGB Image | Collected 3D Point Cloud | Ground Truth | Difference Image-Based Density Correction [20] |
|-------|---------------------|--------------------------|--------------|-----------------------------------------------|
| 1     | ![Image](image1.png) | ![Image](image2.png)     | ![Image](image3.png) | ![Image](image4.png)                        |
| 2     | ![Image](image5.png) | ![Image](image6.png)     | ![Image](image7.png) | ![Image](image8.png)                        |
| 3     | ![Image](image9.png) | ![Image](image10.png)    | ![Image](image11.png) | ![Image](image12.png)                       |

| Index | DeepLabV3 [21] (Using Background) | Proposed Method (Using Background) |
|-------|-----------------------------------|-----------------------------------|
| 1     | ![Image](image13.png)             | ![Image](image14.png)              |
**Appendix A**

**Table A1. Dense segmentation image result of each method.**

| Index | DeepLabV3 [21] | DeepLabV3 [21] (Using Background) | Proposed Method | Proposed Method (Using Background) |
|-------|----------------|-----------------------------------|-----------------|-----------------------------------|
| 2     | ![Image](image1.png) | ![Image](image2.png) | ![Image](image3.png) | ![Image](image4.png) |
| 3     | ![Image](image5.png) | ![Image](image6.png) | ![Image](image7.png) | ![Image](image8.png) |

**References**

1. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. *Field Serv. Robot.* 2017, 5, 621–635.
2. Meftah, L.H.; Braham, R. A Virtual Simulation Environment using Deep Learning for Autonomous Vehicles Obstacle Avoidance. In Proceedings of the 2020 IEEE International Conference on Intelligence and Security Informatics (ISI), Arlington, VA, USA, 9–11 November 2020; pp. 1–7.
3. Bojarski, M.; Testa, D.D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L.D.; Monfort, M.; Muller, U.; Zhang, J.; et al. End to End Learning for Self-Driving Cars. *arXiv* 2016, arXiv:1604.07316.
4. Zhou, T.; Zhu, Q.; Du, J. Intuitive Robot Teleoperation for Civil Engineering Operations with Virtual Reality and Deep Learning Scene Reconstruction. *Adv. Eng. Inform.* 2020, 46, 101170–101191. [CrossRef]
5. Yi, C.; Lu, D.; Xie, Q.; Liu, S.; Li, H.; Wei, M.; Wang, J. Hierarchical Tunnel Modeling from 3D Raw LiDAR Point Cloud. *Comput.-Aided Des.* 2019, 114, 143–154. [CrossRef]
6. Zhu, Q.; Wu, J.; Hu, H.; Xiao, C.; Chen, W. LiDAR Point Cloud Registration for Sensing and Reconstruction of Unstructured Terrain. *Appl. Sci.* 2018, 8, 2318. [CrossRef]
7. Zhang, T.; Nakamura, Y. Moving Humans Removal for Dynamic Environment Reconstruction from Slow-Scanning LiDAR Data. In Proceedings of the 2018 15th International Conference on Ubiquitous Robots (UR), Honolulu, HI, USA, 26–30 June 2018; pp. 449–454.
8. Fang, Z.; Zhao, S.; Wen, S.; Zhang, Y. A Real-Time 3D Perception and Reconstruction System Based on a 2D Laser Scanner. *J. Sens.* 2018, 2018, 1–14. [CrossRef]
9. Yang, Z.; Wang, S.; Manivasagam, S.; Huang, Z.; Ma, W.; Yan, X.; Yumer, E.; Urtasun, R. S3: Neural Shape, Skeleton, and Skinning Fields for 3D Human Modeling. *arXiv* 2021, arXiv:2101.06571.
10. Tian, Y.; Chen, L.; Sung, Y.; Kwak, J.; Sun, S.; Song, W. Fast Planar Detection System Using a GPU-based 3D Hough Transform for LiDAR Point Clouds. *Appl. Sci.* 2020, 10, 1744. [CrossRef]
11. Yang, B.; Luo, W.; Urtasun, R. RIXOR: Real-time 3D Object Detection from Point Clouds. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7652–7660.
12. Iskakov, K.; Burkov, E.; Lempiäisky, V.; Malkov, Y. Learnable Triangulation of Human Pose. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 7718–7727.
13. Nibali, A.; He, Z.; Morgan, S.; Prendergast, L. 3D Human Pose Estimation with 2D Marginal Heatmaps. In Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa Village, HI, USA, 7–11 January 2019; pp. 1477–1485.
14. Luo, Z.; Golestanineh, S.A.; Kitani, K.M. 3D Human Motion Estimation via Motion Compression and Refinement. In Proceedings of the 2020 Asian Conference on Computer Vision (ACCV), Virtual, 30 November–4 December 2020; pp. 1–17.
15. Te, G.; Hu, W.; Zheng, A.; Guo, A. RGCN: Regularized Graph CNN for Point Cloud Segmentation. In Proceedings of the 26th ACM Multimedia Conference Multimedia (MM), Seoul, Korea, 22–26 October 2018; pp. 746–754.
16. Meng, H.; Gao, L.; Lai, Y.; Manocha, D. VV-net: Voxel VAE Net with Group Convolutions for Point Cloud Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 8500–8508.

17. 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. 652–660.

18. Chibane, J.; Alldieck, T.; Pons-Moll, G. Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 6970–6981.

19. Rao, Y.; Lu, J.; Zhou, J. Global-Local Bidirectional Reasoning for Unsupervised Representation Learning of 3D Point Clouds. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 6970–6981.

20. Kwak, J.; Sung, Y. Automatic 3D Landmark Extraction System based on an Encoder-decoder using Fusion of Vision and LiDAR for Feature Extraction. Remote Sens. 2020, 12, 1142. [CrossRef]

21. Chen, L.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv 2017, arXiv:1706.05587.

22. Chen, L.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In Proceedings of the 2018 European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 1–12.

23. Lin, C.; Kong, C.; Lucey, S. Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction. Thirty-Second Aaai Conf. Artif. Intell. 2018, 32, 7114–7121.

24. Park, K.; Kim, S.; Sohn, K. High-Precision Depth Estimation Using Uncalibrated LiDAR and Stereo Fusion. IEEE Trans. Intell. Transp. Syst. 2020, 21, 321–335. [CrossRef]

25. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Proceedings of the 2017 Advances in Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 5099–5108.

26. Tian, Y.; Song, W.; Chen, L.; Sung, Y.; Kwak, J.; Sun, S. A Fast Spatial Clustering Method for Sparse LiDAR Point Clouds Using GPU Programming. Sensors 2020, 20, 2309. [CrossRef] [PubMed]

27. Tina, Y.; Chen, L.; Song, W.; Sung, Y.; Woo, S. DGCNet: Dynamic Graph Convolutional Broad Network for 3D Object Recognition in Point Cloud. Remote Sens. 2021, 13, 66.

28. Zhou, Y.; Tuzel, O. Vox2Net: 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, 18–22 June 2018; pp. 4490–4499.

29. Moon, G.; Chang, J.Y.; Lee, K.M. V2V-PoseNet: Voxel-to-Voxel Prediction Network for Accurate 3D Hand and Human Pose Estimation from a Single Depth Map. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 5079–5088.

30. 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 918–927.

31. Simon, M.; Milz, S.; Amende, K.; Gross, H. Complex-YOLO: An Euler-Region-Proposal for Real-time 3D Object Detection on Point Clouds. arXiv 2018, arXiv:1803.06199.

32. Ali, W.; Abdelkarim, S.; Zidan, M.; Zahran, M.; Sallab, A.E. YOLO3D: End-to-End Real-time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud. In Proceedings of the 2018 European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 1–12.

33. Qi, X.; Liao, R.; Jia, J.; Fidler, S.; Urtasun, R. 3D Graph Neural Networks for RGBD Semantic Segmentation. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5199–5208.

34. Gojcic, Z.; Zhou, C.; Wegner, J.D.; Wieser, A. The Perfect Match: 3D Point Cloud Matching with Smoothed Densities. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–21 June 2019; pp. 5545–5554.

35. Yifan, W.; Wu, S.; Huang, H.; Cohen-Or, D.; Sorkine-Hornung, O. Patch-based Progressive 3D Point Set Upsampling. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2019; pp. 3593–3601.

36. Dai, H.; Shao, L. PointAE: Point Auto-encoder for 3D Statistical Shape and Texture Modelling. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seoul, Korea, 15–21 June 2019; pp. 5410–5419.

37. Ge, L.; Liang, H.; Yuan, J.; Thalmann, D. Robust 3D Hand Pose Estimation in Single Depth Images: From Single-View CNNs to Multi-View CNNs. 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.
39. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.

40. 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 (CVPR), Cambridge, MA, USA, 18–20 June 2012; pp. 3354–3361.

41. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets Robotics: The KITTI Dataset. *Int. J. Robot. Res.* 2013, 32, 1231–1237. [CrossRef]