IMAGE TO POINT CLOUD TRANSLATION USING CONDITIONAL GENERATIVE ADVERSARIAL NETWORK FOR AIRBORNE LIDAR DATA

Takayuki Shinohara, Haoyi Xiu, Masashi Matsuoka

Tokyo Institute of Technology, Department of Architecture and Building Engineering, Yokohama, Japan

KEY WORDS: Airborne LiDAR, Point Clouds, Conditional Adversarial Generative Network, Generative Model.

ABSTRACT:

This study introduces a novel image to a 3D point-cloud translation method with a conditional generative adversarial network that creates a large-scale 3D point cloud. This can generate supervised point clouds observed via airborne LiDAR from aerial images. The network is composed of an encoder to produce latent features of input images, generator to translate latent features to fake point clouds, and discriminator to classify false or real point clouds. The encoder is a pre-trained ResNet; to overcome the difficulty of generating 3D point clouds in an outdoor scene, we use a FoldingNet with features from ResNet. After a fixed number of iterations, our generator can produce fake point clouds that correspond to the input image. Experimental results show that our network can learn and generate certain point clouds using the data from the 2018 IEEE GRSS Data Fusion Contest.

1. INTRODUCTION

In recent years, owing to the increasing number of related applications, analysis methods related to point clouds have been widely studied. Specifically, measurements using airborne LiDAR have been conducted because they enable a wide range of 3D information to be acquired quickly, with high accuracy (Lohani and Ghosh, 2017). As the performance of scanners has improved over the years, high-density point clouds can be acquired and handled in large numbers. Consequently, research into the automatic analysis of point clouds using deep learning has been actively conducted. However, because most methods that use deep learning for point clouds (observed by airborne LiDAR) are discrimination models such as segmentation or object detection, little research has been done on generative models.

Owing to the complexity of 3D point-cloud data in outdoor scenes, the number of labeled open-data sources alone cannot satisfy the enormous data needs of deep-learning research. Automatic generation of 3D point clouds can solve this problem by providing a potentially infinite variety of data sources. Although methods for estimating 3D point clouds from images are widely used to recover simple CAD data or for people using deep learning (Wang et al., 2018), (Tatarchenko et al., 2017), (Fan et al., 2017), (Zhang et al., 2019), (Mescheder et al., 2019), (Park et al., 2019), (Chen and Zhang, 2019) , and (Kato et al., 2020), it is strongly assumed that only a single object is in the image (single-object assumption). This makes the method unsuitable for point clouds and aerial photos observed in an outdoor scene (the subject of this study), and therefore another method is needed. We propose an aerial image to point cloud translation using a conditional generative adversarial network (cGAN) that faithfully generates dense point clouds, observed by airborne LiDAR from aerial photos of the same region without a single object assumption. To the best of our knowledge, this is the first supervised generation of point clouds observed via airborne LiDAR using cGAN.

Figure 1 shows the overview of 3D point cloud generation.

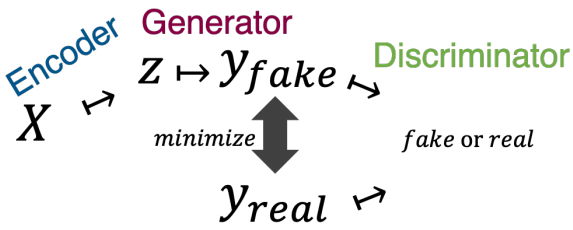

Figure 1. Aerial photo to point clouds translation pipeline using conditional adversarial generative network.

Generating point clouds is a difficult and computationally expensive task because the number of features and level of detail increase with resolution. In traditional generative adversarial networks (Goodfellow et al., 2014), the generator attempts to generate fake data to fool a discriminator in an unsupervised manner. To change the unsupervised task into a supervised task, generative adversarial networks with a condition on the input data (Isola et al., 2017) generate fake data that minimize the distance generated from fake data and real (ground truth) data. Additionally, an image-to-voxel translation method was proposed using the same cGAN pipeline (Knyaz et al., 2019). However, because the voxel format is not memory-efficient for representing data over a wide area and is difficult to use for airborne LiDAR data, the point cloud format is more suitable for wide-area data. This study proposes an end-to-end aerial image to the point-cloud translation model that learns the distribution of the point clouds, as well as maps, from image to point clouds with a cGAN-based network.

The key contributions of this work are threefold:

- We introduce a conditional generative adversarial network that transforms aerial images into 3D point clouds in the absence of supervision.
- Our network follows pix2pix pipeline and consists of a ResNet-based encoder to extract latent features of aerial

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-V-2-2021-169-2021

photos, FoldingNet-based generator to convert point clouds from latent features, and discriminator.

- We include both qualitative and quantitative analyses of the synthesized data collected by our network using a dataset that includes a pair of airborne LiDAR data and an aerial image.

The rest of this paper is organized as follows: In Section 3, we define the problem mathematically and provide essential background information. We also show the proposed conditional generative adversarial network (GAN) for point cloud generation and the loss function for the training process. We present the experimental setup and results in Section 4. A conclusion of this work is given in Section 5.

2. RELATED STUDY

In 2014, the original GAN (Goodfellow et al., 2014), an algorithm for unsupervised learning, was published. GANs can acquire probability distributions of input data. Deep Convolutional GAN (DCGAN) is used for image generation (Radford et al., 2015). Since the publication of the original GAN, a large number of papers on the GAN-based model have been published. In 2018, 11,800 papers related to GANs were reportedly published (Gui et al., 2020). Although GAN was born in the framework of image generation, it is now being applied in various fields, such as natural language processing and speech processing.

To summarize, a GAN is an adversarial learning process between a generator (G) and a discriminator (D). The original GAN, G, and D are composed of a multi-layer perceptron (MLP). By contrast, the DCGAN (Radford et al., 2015) uses convolutional layers to enhance the expressiveness of G and D for suitable image generation. G of the DCGAN is trained to make the generated image as close as possible to the real image, and D of the DCGAN is trained to accurately identify whether the input image is real or fake generated by the G. The G learns to generate an image from the noise z that is close to the real one. When D can no longer distinguish between the real image and the image generated by G, the learning process converges. In other words, G indicates that it has learned the distribution of the training data. The original GAN comes down to the problem of minimizing the distance between fake and real, called the JS divergence, which is calculated by D. Here, the JS divergence is a symmetric distance to the KL divergence. The original GAN and DCGAN could achieve image generation.

GANs have two major problems in the training process; however, the gradient vanishing problem and mode collapse. In the early stages of training, real and fake data generated by G are easily distinguishable. In other words, $D(G(z))$ is close to zero; when it is near zero, the gradient is almost zero, and G does not train well. This is called the gradient vanishing problem. Additionally, when G produces an image that can fool D, the loss function will be small if we only produce that image. In other words, the learned G produces only the same type of images. This is called mode collapse.

Here, we introduce the typical variants of GAN to overcome these problems. Wasserstein GAN (WGAN) (Arjovsky et al., 2017) is a GAN that proposes a loss function using a distance to measure between two probability distributions called the Wasserstein distance to stabilize the learning of the GAN to solve mode collapse. The Wasserstein distance is the minimum required to match a probability distribution to the target probability distribution.

GANs have many applications beyond image generation. Conditional GANs (cGANs) (Mirza and Osindero, 2014) are designed to generate images under certain conditions. For example, cGANs can generate images based on class labels or sentence features. A famous example of cGAN is pix2pix (Isola et al., 2017), a method of image translation between different modal images, such as adding color to a black-and-white image. In the fields of computer graphics, image processing, and computer vision, the input image is processed to produce the output image in many transformation problems. However, the algorithms used in these problems have been developed separately for each application and are not generalizable. The motivation behind Pix2Pix is to generalize the algorithms for these transformation problems into a simple common framework.

In Pix2Pix, the original input image x translates into fake data $G(x)$ produced by G. D trains to correctly identify real (ground truth) images y and x (real pair) or transformed images $G(x)$ and x (fake pair). On the other hand, G trains to mislead D to identify that x and $G(x)$ are real pairs. This process allows G to learn how to make natural transformations. It is conditional in the sense that it is learning with the pair (x, y). We show each network G and D of Pix2Pix. First, we show the detail of G. Pix2Pix uses U-Net (Ronneberger et al., 2015) as a G. Since capturing the fine details of pixels in the image transformation problem, as well as in semantic segmentation, is important, U-Net is suitable because it can propagate the detailed information of the image with the encoder-–decoder structure and skip connections. In Pix2Pix, the noise vector z is given to each layer of the U-Net by adding dropout to multiple U-Net layers, which enable the generation of high-quality images.

Further, we show the detail of D. The D incorporates the idea of PatchGAN (Li and Wand, 2016). The input image is decomposed into patches of $M \times M$ resolution, and each patch is identified as real or fake. The output of D (i.e., the real/fake image) is the average of the real/fake image of all patches. By having D learn to discriminate patch by patch, rather than on the entire image, we can concentrate the cGAN training on modeling the image's high-frequency components. In Pix2Pix, the overall image (low-frequency component) is captured by the L1 loss function, whereas the detailed image (high-frequency component) is captured by the cGAN so that the shortcomings of each are complemented to improve accuracy. The convolutional layers of the generator and discriminator both use the Convolution-Batch Normalization-ReLU module.

We show the remainder of this paper. Herein, we propose a cGAN-based image-to-point translation method to translate 3D point clouds observed by airborne LiDAR from an aerial photo.

3. IMAGE TO POINT TRANSLATION METHOD

3.1 Problem Statement

We consider an input aerial photo (input image) $X \in \mathbb{R}^{3\times H\times W}$ in which 3 denotes the RGB channel, H is the height of the image, and W is the width; our final output is a 3D fake point cloud, $y_{fake} \in \mathbb{R}^{N\times 3}$, where N is the number of points. In this study, N is $2,025$, depending on the specifications of the computing environment. Rather than directly convert the input

image into a 3D point cloud, we use a trained CNN to project it to a low-dimensional latent feature z, as follows:

$$z = E(X).$$

The fake point cloud (y_{fake}) is then generated from the low-dimensional latent feature z:

$$y_{fake} = G(z).$$

In the training process, G attempts to minimize the distance between y_{fake} and y_{real}. Additionally, G learns to mislead D to classify fake data as real data. Conversely, D learns to classify fake data as fake data.

3.2 Optimization

Here, we describe the conditional GAN used in this study.

3.2.1 GAN First, we explain the basic generative adversarial network (GAN) (Goodfellow et al., 2014). GAN is a generative model of a neural network that attempts to learn the probability distribution, P_r, of the input data to generate fake samples. The GAN consists of a generator G and an identifier D, and these models fool each other through non-cooperative games, with min-max objectives. The generator G tries to convert a random vector z, sampled from a fixed distribution P_g, into the generated fake data (y_{fake}) that are indistinguishable from the real data (y_{real}). The discriminator seeks to differentiate between y_{fake} and y_{real}. More formally, the loss function of a GAN $\mathcal{L}_{GAN}(G, D)$ can be written as

$$\mathbb{E}_{y_{real} \sim P_r}[\log D(y_{real})] + \mathbb{E}_{z \sim P_g}[\log(1 - D(G(z))].$$

3.2.2 cGAN Second, we explain the image-to-points translation pipeline. To add conditions to the input image when generating point clouds, we use conditional GAN (cGAN) instead of the pure original GAN. Pix2Pix (Isola et al., 2017) used GANs for image-to-image translation and regression loss to force the output to be conditioned on the input.

In this study, we expand this Pix2Pix pipeline into an image-to-point cloud translation problem. The loss function of a cGAN can be expressed as

$$\mathcal{L}_{cGAN}(G, D) = L_{GAN}(G, D) + \lambda L_{point}, \qquad (1)$$

in which L_{point} is the pointwise loss between the generated and real points, and λ is a coefficient determined by the hyperparameter. Next, we show the L_{point}. In this study, $y_{real} \subseteq \mathbb{R}^{N \times 3}$ is defined as a set of real points, and $y_{fake} \subseteq \mathbb{R}^{N \times 3}$ is defined as sets of fake points. For the distance between the real and generated point clouds(L_{point}), we used a *chamfer* (pseudo)-distance (CD). For two subsets of the same size y_{real} and y_{fake}, the CD ($L_{CD}(y_{real}, y_{fake})$) between these subsets is defined by

$$\sum_{s_1 \in Y_{real}} \min_{s_2 \in y_{fake}} \|s_1 - s_2\|_2^2 + \sum_{s_2 \in Y_{fake}} \min_{s_1 \in y_{real}} \|s_1 - s_2\|_2^2.$$

We use $L_{CD}(y_{real}, y_{fake})$ as a L_{point}.

Our final loss function, G, tries to minimize this objective against an adversarial D that attempts to maximize it as follows:

$$\arg \min_G \max_D \mathcal{L}_{cGAN}(G, D). \qquad (2)$$

3.3 Overall Architecture

Figure 2 shows the architecture of our image to 3D point cloud translation. Our model has the following three main networks: an encoder E, a generator G, and a discriminator D. Unless stated otherwise, we refer to a real point cloud y_{real} as a 2-dimensional matrix with N points—that is, $y_{real} \in \mathbb{R}^{N \times 3}$, in which each point represents geometric information (x, y, z).

In this paper, we set N as $2,025$. For image-to-point cloud translation, both the generator and the discriminator are conditioned on a latent image feature z from a pre-trained CNN as an encoder E. To optimize the generator and discriminator, we used the cGAN techniques discussed in Section 3.2.2. Figure 2 shows the overall architecture of our image-to-point cloud translation model.

3.3.1 Encoder First, we show the encoder E. To obtain a prior distribution for the input to the generator, we need to extract the latent features from the input aerial photo. We used an encoder with the ResNet50 architecture (He et al., 2016) (pre-trained on ImageNet (Deng et al., 2009)) as a feature extractor for the input image. The pre-trained ResNet can provide the latent feature z, including meaningful context information of the input image. In this study, to obtain latent feature z, we use the output of the last pooling layer of ResNet with a $2,025$ dimensional 1D array—that is, the feature vector—before connecting to the fully connected layer.

3.3.2 Generator Second, we show the generator G, which translates into fake point clouds (y_{fake}) from the input latent feature z. The z controls the generation of point clouds with a desired geometric distribution.

In this paper, the generator is comprised of a FoldingNet (Yang et al., 2018), which generates point clouds by concatenating the latent features of the input image extracted from the E to the 2D grid (45×45)—the seed of point cloud generation—and then applies two folding blocks to the concatenated 2D grid. Each folding block consists of three fully connected layers, each of which has 1024, 1024, and three features. The first folding block is used to convert this 2D grid into an "intermediate" 3D point cloud. The structure of the second folding block is used to convert an "intermediate" 3D point cloud into a final 3D point cloud.

3.3.3 Discriminator Finally, we show the discriminator D. Given a real-point cloud y_{real} or a generated fake-point cloud y_{fake}, the discriminator D tries to predict the probability that y_{real} is real or generated. In this paper, we use PointNet++ (Qi et al., 2017) as the discriminator. We follow Pix2Pix (Isola et al., 2017) and use PatchGAN(Demir and Unal, 2018) to classify each small region as fake or real.

Because PointNet++ includes down-sampling layers, we used three down-sampling layers with [1024, 512, and 256]. The feature dimension for the layers of D was set to $[3, 64]$ in the first down-sampling layer, $[128, 256]$ in the second down-sampling layer, $[512, 1]$ in the last down-sampling layer, and the probability of belonging to real data in the last layer.

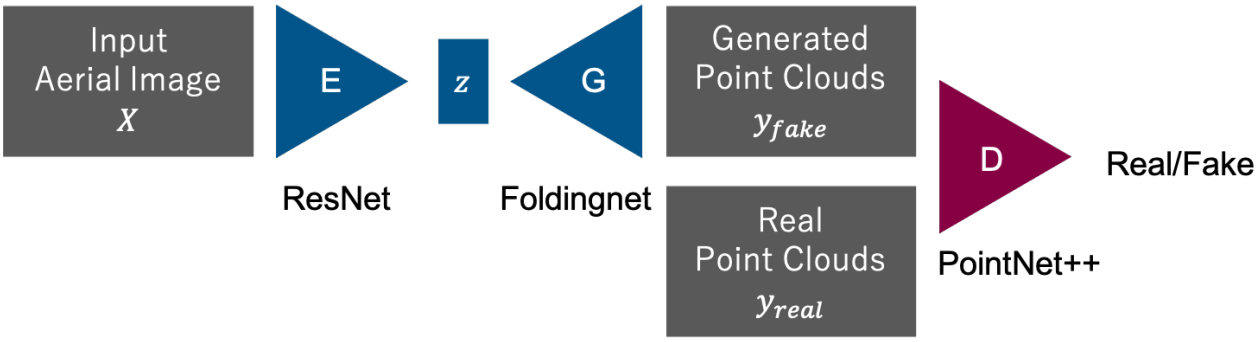

Figure 2. Given a latent feature z from input image X using encoder E, the generator G produces a point cloud y_{fake}. The real point cloud y_{real} and the generated fake point cloud y_{fake} are fed into the discriminator which then tries to predict the probability of the data being real data or generated fake data. The encoder is pre-trained ResNet-50, the decoder is FoldingNet, and the discriminator is PointNet++. These models are trained in an end-to-end manner.

3.4 Training

We show the training setup. The weights of the encoder are fixed daring training process. For both generator and discriminator, we used LeakyReLU nonlinearity with a negative slope of 0.2. The learning rate α was set to 10^{-4}, and we used the Adam optimizer (Kingma and Ba, 2014) with coefficients $\beta_1 = 0.0$ and $\beta_2 = 0.95$. The coefficient of the distance between the real and generated point clouds(L_{point}) in λ in $\mathcal{L}_{cGAN}(G, D)$ is 10.

4. EXPERIMENTAL RESULT

4.1 Dataset

We used the point clouds observed by airborne LiDAR data and aerial photos of an outdoor scene to conduct our experiments (see Figure 3). This dataset, published in the 2018 IEEE GRSS Data Fusion Contest (Saurabh Prasad, 2020), contains houses, ground, trees, and an American football field, so a wide variety of objects have been observed (Figure 4). Because a single piece of an image, or point cloud, is incredibly large—on the order of several hundred meters—it must be cut into small patches to be computed on the GPU used for training. In this study, we cut both the point cloud and aerial photo into 25 m squares. Also, because this point cloud data contains noise, such as points that have gone underground, we removed isolated points as noise and subsample point clouds into $2,025$ by using open3D (Zhou et al., 2018). We moved each point cloud in an area in the range $[0, 25]$ m.

To train and evaluate the model, we divided all patch data into two datasets: the training data used to train our model and the test data (as shown by the blue rectangle in Figure 4) used to evaluate the trained model.

4.2 Result

We trained on the above dataset by using the TSUBAME 3.0 (NVIDIA P100 GPU with 16 GB video memory). The training time for our model was three days for 100 epochs.

The qualitative evaluation of the trained model is shown below. Figure 5 shows a set of translated fake point clouds from the aerial image, along with their real point clouds. The left figures are the input images for the condition to the generator, the middle figures are the translated fake point cloud, and the right figures are the real point cloud. As shown in the translated

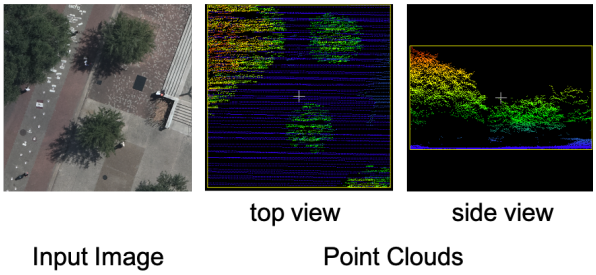

Figure 3. Experimental dataset used in this paper. We used pairs of aerial photo and point clouds observed by airborne LiDAR, published in 2018 IEEE GRSS Data Fusion Contest (Saurabh Prasad, 2020).

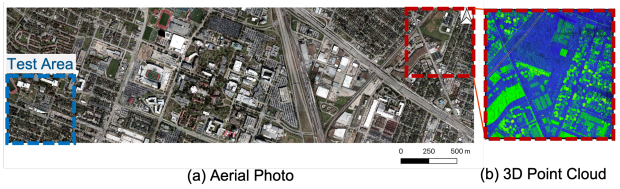

Figure 4. Target area used in this paper. The data are published in the 2018 IEEE GRSS Data Fusion Contest (Saurabh Prasad, 2020).

point clouds, translating a point cloud from the input image is possible. Overall, our model could translate point clouds with a layout similar to that of an aerial photo. For example, the point cloud translated in the area with the building indicated by the red circle also shows a building-like 3D object. Also, vegetation-like point clouds were also translated into areas with vegetation, as shown by the green circle.

Overall, the conversion from aerial photos to point clouds was successful. However, some failure cases showed such issues as walls of buildings and points concentrated in the corners. This was because the FoldingNet used in the decoder uses a two-dimensional grid as the initial value or seed of point clouds. We need to use a deep learning-based generator that can generate more detailed 3D structures in the future. Additionally, CD loss, which evaluates the 3D reconstruction results, calculated the distance between two sets y_{fake} and y_{real}, so it did not evaluate the points one by one. Instead, it learned the generator so that the set as a whole had a low error. The generator

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-V-2-2021-169-2021 | © Author(s) 2021. CC BY 4.0 License.

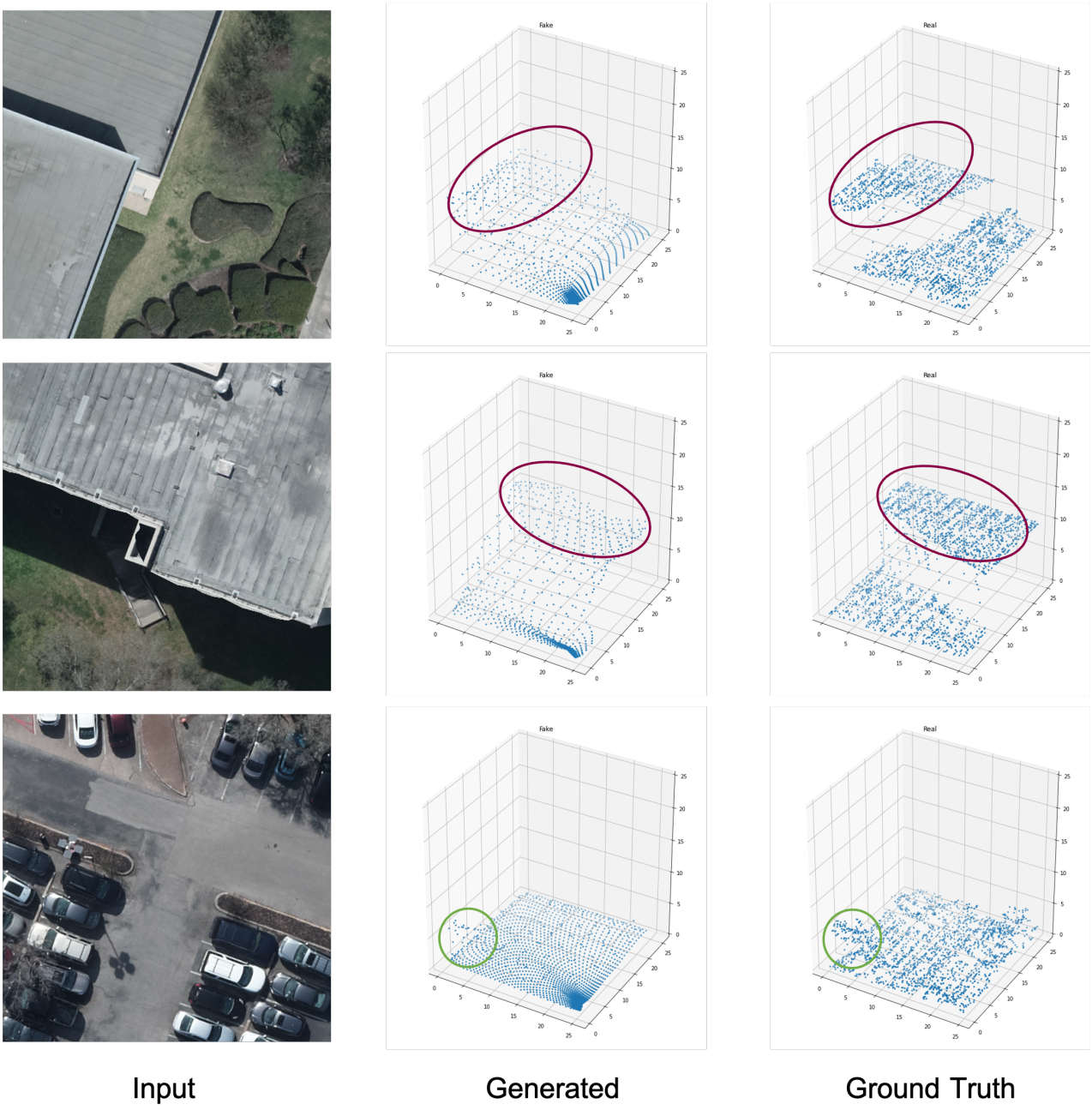

Figure 5. The generated fake point clouds y_{fake} via latent feature z from input image X. The left figures are the input aerial image X, the middle figures are y_{fake}, and the right figure are real (ground truth) point cloud y_{real}.

is trained to lower the CD loss for the set as a whole to obtain an average 3D point cloud. Therefore, for point clouds containing complex geological objects, such as LiDAR data measured outdoors, defining an error function that can evaluate point-to-point is necessary. Also, we need to improve the generator's performance in the future and to feed the latent feature obtained from the images to the generator.

5. CONCLUSION

This paper introduced the translation of aerial images to point clouds, which were observed by airborne LiDAR. The translation model was developed using a conditional generative adversarial network in supervised learning. Our pipeline consists of three networks: the encoder, which extracts the latent vector of the input image; the decoder, which produces fake point clouds from latent vectors; and a discriminator, which measures the distance between real data and fake data. To evaluate our proposed pipeline, we experimented using a pair of aerial photos and airborne LiDAR data (2018 IEEE GRSS Data Fusion Contest). Our pipeline has shown the potential for 3D reconstruction via the visualization of fake point clouds.

However, the quality of the generated point clouds remains an issue, and we will try to improve the performance further by using a modern generator such as a transformer-based model (Mazur and Lempitsky, 2020), (Engel et al., 2020), which has been used in recent years for the 3D point cloud generation. Also, we have shown the effectiveness of the conditional GAN

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-V-2-2021-169-2021 | © Author(s) 2021. CC BY 4.0 License.

for airborne LiDAR, and we will apply it to various applications in the future.

ACKNOWLEDGEMENTS

We gratefully acknowledge the benchmark data (2018 IEEE GRSS Data Fusion Contest) owners for providing aerial photo and point clouds. This work was supported by JSPS KAKENHI (Grant Number 19H02408). The numerical calculations were performed using the TSUBAME 3.0, supercomputer at the Tokyo Institute of Technology.

REFERENCES

Arjovsky, M., Chintala, S., Bottou, L., 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Chen, Z., Zhang, H., 2019. Learning implicit fields for generative shape modeling. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5939–5948.

Demir, U., Unal, G., 2018. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database. *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 248–255.

Engel, N., Belagiannis, V., Dietmayer, K., 2020. Point Transformer. *arXiv preprint arXiv:2011.00931*.

Fan, H., Su, H., Guibas, L. J., 2017. A point set generation network for 3d object reconstruction from a single image. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 605–613.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.

Gui, J., Sun, Z., Wen, Y., Tao, D., Ye, J., 2020. A review on generative adversarial networks: Algorithms, theory, and applications. *arXiv preprint arXiv:2001.06937*.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Identity mappings in deep residual networks. *European conference on computer vision*, Springer, 630–645.

Isola, P., Zhu, J.-Y., Zhou, T., Efros, A. A., 2017. Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125–1134.

Kato, H., Beker, D., Morariu, M., Ando, T., Matsuoka, T., Kehl, W., Gaidon, A., 2020. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*.

Kingma, D. P., Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Knyaz, V. A., Kniaz, V. V., Remondino, F., 2019. Image-to-voxel model translation with conditional adversarial networks. L. Leal-Taixé, S. Roth (eds), *Computer Vision – ECCV 2018 Workshops*, Springer International Publishing, Cham, 601–618.

Li, C., Wand, M., 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. *European conference on computer vision*, Springer, 702–716.

Lohani, B., Ghosh, S., 2017. Airborne LiDAR technology: a review of data collection and processing systems. *Proceedings of the National Academy of Sciences, India Section A: Physical Sciences*, 87(4), 567–579.

Mazur, K., Lempitsky, V., 2020. Cloud transformers. *arXiv preprint arXiv:2007.11679*.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A., 2019. Occupancy networks: Learning 3d reconstruction in function space. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4460–4470.

Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Park, J. J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S., 2019. Deepsdf: Learning continuous signed distance functions for shape representation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 165–174.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, 5099–5108.

Radford, A., Metz, L., Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, Springer, 234–241.

Saurabh Prasad, Bertrand Le Saux, N. Y. R. H., 2020. 2018 ieee grss data fusion challenge – fusion of multispectral lidar and hyperspectral data. IEEE Dataport.

Tatarchenko, M., Dosovitskiy, A., Brox, T., 2017. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. *Proceedings of the IEEE International Conference on Computer Vision*, 2088–2096.

Wang, L., Gao, C., Yang, L., Zhao, Y., Zuo, W., Meng, D., 2018. Pm-gans: Discriminative representation learning for action recognition using partial-modalities. *Proceedings of the European Conference on Computer Vision (ECCV)*, 384–401.

Yang, Y., Feng, C., Shen, Y., Tian, D., 2018. Foldingnet: Point cloud auto-encoder via deep grid deformation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 206–215.

Zhang, Y., Liu, Z., Liu, T., Peng, B., Li, X., 2019. RealPoint3D: An efficient generation network for 3D object reconstruction from a single image. *IEEE Access*, 7, 57539–57549.

Zhou, Q.-Y., Park, J., Koltun, V., 2018. Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*.