USING DEEP LEARNING TO DIGITIZE ROAD ARROW MARKINGS FROM LIDAR POINT CLOUD DERIVED IMAGES

M. L. R. Lagahit *, Y. H. Tseng

Dept. of Geomatics, National Cheng Kung University, Tainan, Taiwan – (miguellagahit, tseng)@prs.geomatics.ncku.edu.tw

KEYWORDS: Deep Learning, Neural Network, Image Segmentation, Mobile Mapping, LIDAR Point Cloud, Road Marking

ABSTRACT:

The concept of Autonomous vehicles or self-driving cars has recently been gaining a lot of popularity. Because of this, a lot of research is being done to develop the technology. One of which is High Definition (HD) Maps, which are centimeter-level precision 3D maps that contain a lot of geometric and semantic information about the road which can assist the AV when driving. An important component of HD maps is the road markings which indicates a set of rules on how a vehicle should navigate itself on the road. For example, lane lines indicate which part of the road a vehicle can drive on in a certain direction. This research proposes a methodology that uses deep learning techniques to detect road arrows, road markings that show possible driving directions, on LIDAR derived images, and extract them as polyline vector shapefiles. The general workflow consists of (1) converting the LIDAR point cloud to images, (2) training and applying U-Net – a fully convolutional neural network, (3) creating masks from image segmentation results that have been transformed to fit the local coordinates, (4) extracting the polygons and polylines, and finally (5) exporting the vectors in shapefile format. The proposed methodology has shown promising results with object segmentation accuracies comparable with previous related works.

1. INTRODUCTION

1.1 Background

Autonomous vehicles (AVs) or self-driving cars is currently one of the most popular topics of research in the field of science and engineering (Brummelen, O'Brien, Gruyer, & Najjaran, 2018). A lot of advances are being done from how it perceives its environment to how it makes its own decisions on the road. One of those is the concept of High Definition (HD) maps. HD Maps are centimeter-level precision 3-dimensional (3D) maps that contain both geometric and semantic information of everything on or nearby the road (Vardhan, 2017) which helps AVs better navigate on the road.

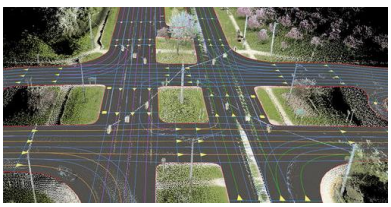

Figure 1. Sample visualization of an HD Map (taken from https://www.geospatialworld.net/)

An essential component of HD Maps is the vectorized representation of road markings. Road markings consist of lane lines that indicate the driving area, road arrows that show possible driving directions, crosswalks which indicate possible human traffic on the road, and much more. As such, a lot of researchers tackle on how to extract those road markings from raw sensor data like images or LIDAR point cloud. Recent trends make use of deep learning techniques for extraction. (Kurz, Azimi, Sheu, & d'Angelo, 2019) used a convolutional neural network (CNN) on aerial images and (Hu, et al., 2019) and (Hoang, Nam, & Park, 2019) used YOLO on mobile mapping images to detect road markings. While (Wen, et al., 2018) and (Wolf, Richter, Discher, & Dollner, 2019) used CNN to detect road markings on images derived from the LIDAR point cloud.

1.2 Objective

This research paper proposes a working methodology that can utilize deep learning techniques such as semantic segmentation, which assigns each pixel to a certain specific classification, using U-Net a kind of convolutional neural network to detect road arrow markings and successfully digitize them as polyline vector shapefiles from LIDAR point cloud derived intensity and color (RGB) images.

1.3 Data Source

The LIDAR point cloud dataset provided by the Department of Geomatics at National Cheng Kung University was taken using a mobile mapping system and contains both color (RGB) and intensity information. It is composed of 28 combined blocks that cover the entire autonomous vehicle (AV) test field located near the Shalun train station in Tainan, Taiwan (ROC). The test field was built to replicate multiple conditions in the real world urban road environment for autonomous vehicle use. It is filled with various road markings (such as road arrows and pedestrian crosswalks) and traffic management objects (such as traffic signs and traffic lights).

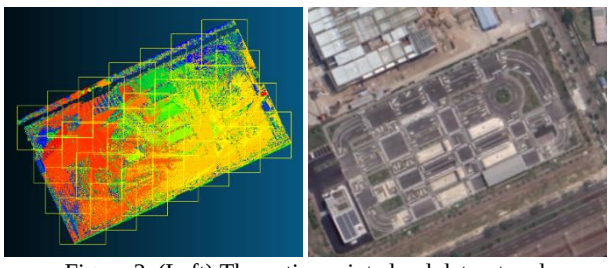

Figure 2. (Left) The entire point cloud dataset and (Right) The AV test field as seen on Google Maps.

* Corresponding author

2. METHODOLOGY

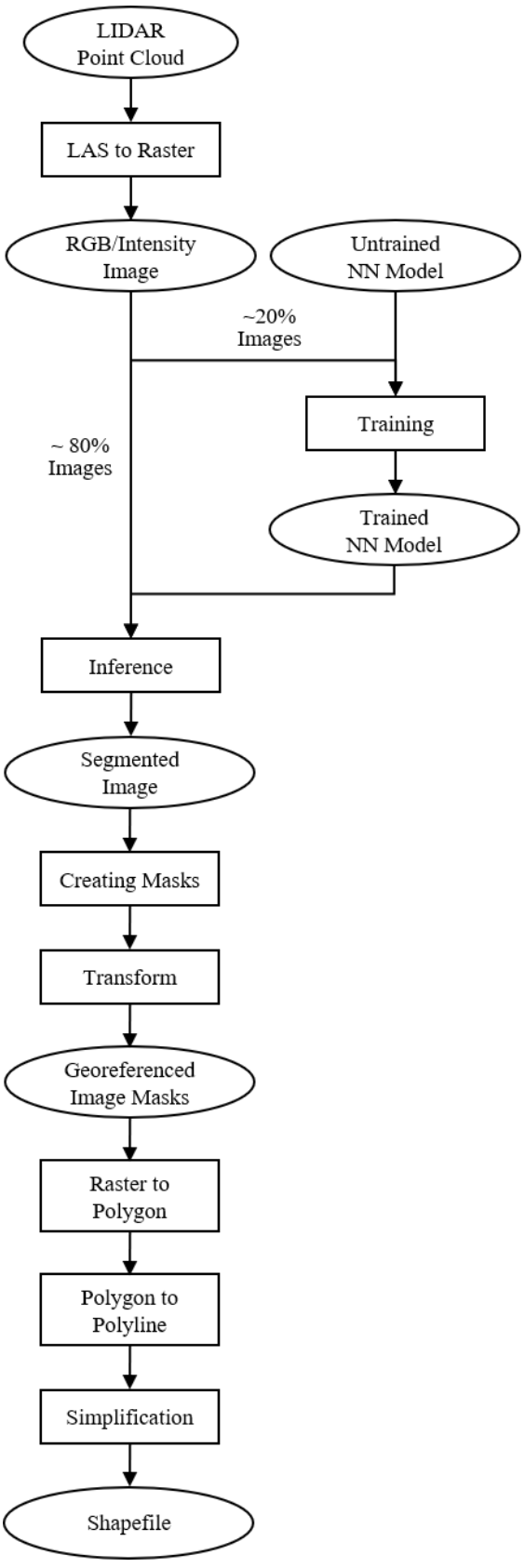

Figure 3. General workflow of the entire methodology.

2.1 LAS to Raster

The initial step is to convert the point cloud blocks, in LAS format, to a raster, PNG format. This is because the convolutional neural network model that will be used for this research is only applicable to images. Converting point cloud to images for neural network classification has been demonstrated in the works of (Wen, et al., 2018), where the point cloud was converted to an image using intensity values. In this case, the point cloud is converted to an image based on intensity and color (RGB) information, on 3 different sampling resolutions: 10, 5, and 1 cm. This was done using ArcMap's (v10.7) LAS to Raster tool. A total of 6 datasets was generated with each containing 28 images.

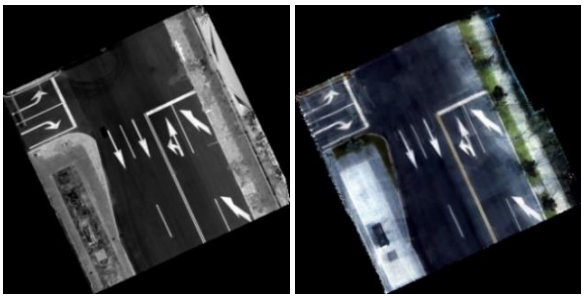

Figure 4. Resulting (Left) Intensity Image and (Right) RGB Image

2.2 Training the Neural Network Model

U-Net v2, a fully convolutional network, was used as the model. U-Net has been proven to be an effective model for road or road segmentation purposes. (Li, Guo, Rao, Xu, & Jin, 2019) and (Yang, et al., 2019) used it on satellite images to extract roads, while (Hu, et al., 2019) and (Wen, et al., 2018) used it on mobile mapping images to extract road markings. The model has been trained using Supervisely, a web-based platform for deep learning. All of the parameters for training the model have been set to default, except the number of epochs which was set to 10, which was the processing limit of the computer used. The parameters are listed below in Table 1:

Table 1. Neural network training parameters

Learning rate: 0.001	Batch size:	Input size:
Epochs: 10	Validation: 1	Width: 256
Iteration: 1 epoch	Training: 4	Height: 256

The computer that was used for this research runs on an Intel i7 @2.80GHz with 4 cores, a 16 GB RAM, and an NVIDIA GTX 1060. In total, 6 models were generated, each from one training dataset. These training datasets have undergone both image annotation and augmentation.

2.2.1 Image annotation: is the process of manually labeling target objects (Ambalina, 2019). In this case, the target object was the road arrows. 5 images (less than 20%) from each dataset were annotated. This means that 5 images were used for each model (ex. 5 images taken from a 1 cm intensity image dataset for the 1 cm intensity image-based model). Only arrows that were fully represented in an image was labeled. This resulted in the initial training dataset.

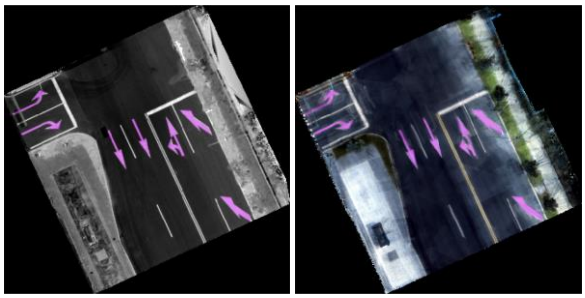

Figure 5. Sample annotated images from the initial training datasets.

2.2.2 Image augmentation: diversifies the training dataset without using any other additional datasets, through a series of image manipulation such as flip, crop, and rotate (Ho, Liang, & Liaw, 2019). The initial training dataset of 5 images went through the first augmentation by flipping copies of the original images horizontally and vertically. After this, the training dataset now had 15 images: 5 of the original, 5 flipped horizontally and 5 flipped vertically. The second augmentation was done by multiplying copies of the images by 10 and applying random crops, random rotations, and both. After this, the training dataset now had a total of 465 images: 15 of the initial augmented, 150 randomly cropped, 150 randomly rotated, and 150 randomly cropped and rotated. Then, the images were tagged at a 95% confidence, as suggested at (Supervisely, 2019), meaning 95% of the images will be used for training and 5% will be used for validation. The images that has undergone this step will be fed to the neural network model for training.

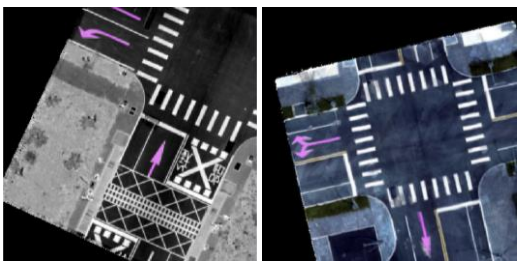

Figure 6. Sample images from the training datasets.

2.3 Creating Masks

Even though segmented image masks were already included in the downloaded datasets from Supervisely it still needed to be changed for it to be able to follow the next procedures. Using Python's (3.X) OpenCV library, the masks were converted to grayscale from color (RGB), then the values were reclassified to only 0 and 1, and the images were exported as TIFF files.

Figure 7. Sample resulting segmented image mask.

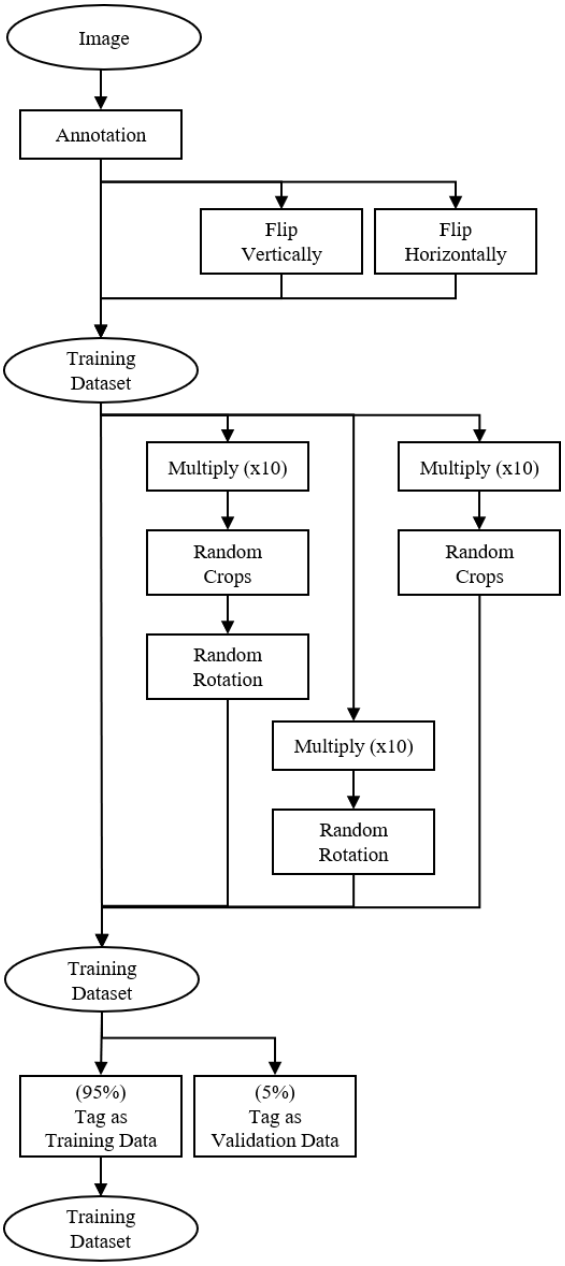

Figure 8. Workflow for creating the training dataset.

2.4 Transforming the Image

Using Python's GDAL library, the segmented image masks were transformed to the same local coordinates as the original TIFF image (from when LAS was converted to raster). This was necessary since downloaded images from Supervisely did not retain their coordinate information.

2.5 Raster to Polygon

Using Python's GDAL library the segmented image masks (raster) were converted to polygon shapefiles (vector). This was done by grouping connecting pixels of the same value, making individual polygon objects for each arrow and the background. The result was individual shapefiles containing multiple objects with different attributes for each one.

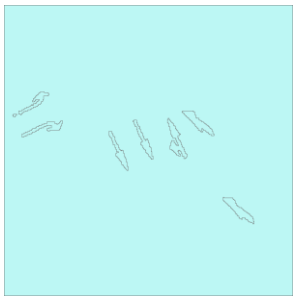

Figure 9. Sample resulting polygon shapefile.

2.6 Polygon to Polyline

The conversion of the polygon to polyline was done using Python's Fiona and Shapely libraries. Having the background polygon removed, the boundary of the polygons were used to represent the polylines. The result was individual shapefiles for each road arrow.

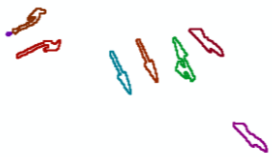

Figure 10. Sample resulting polyline shapefiles.

2.7 Simplification

Finally, the polyline was further improved by simplification. Using Python's Shapely library the line was simplified to remove the jagged effect caused by following the edges of the pixel boundary. In a simple test to figure out which simplification parameter worked visually well, 10 cm and 20 cm values were tested. It turns out that 20 cm had better results and this was used for the entire dataset.

Figure 11. Polyline simplification parameter used: (Left) None, (Middle) 10cm and (Right) 20cm.

3. RESULTS AND DISCUSSION

To asses the results, I have separated the generated road arrow polyline vectors into 4 classifications namely: Fully Detected, Partially Detected, Wrongly Detected, and Not Detected. Fully detected means that the arrow has been well represented and the head and tail can be clearly seen. Partially detected means that not all of the arrow has been represented but it was detected. Wrongly detected means that a feature that is not an arrow was detected. Not Detected means that an arrow that is present in the image was not detected. Only arrows that were fully visible in the image was taken into account, those that were just partially visible were excluded in the assessment.

Table 2. The number of detections per classification.

Image	Sampling Resolution	Manually Digitized	Fully Detected	Partially Detected	Wrongly Detected	Not Detected
Int…	10	61	51	1	15	9
	5		52	4	18	5
	1		41	12	20	8
RGB	10		40	15	26	6
	5		38	9	20	14
	1		24	20	17	17

Considering both the fully detected and the partially detected as correct identifications, one of the models achieved an accuracy of 92% (computed by dividing the correct identifications by the number of manually digitized arrows). This result did not deviate from the results of previous similar works, of (Wolf, Richter, Discher, & Dollner, 2019) which achieved a 91% accuracy and of (Wen, et al., 2018) which achieved a 96% accuracy.

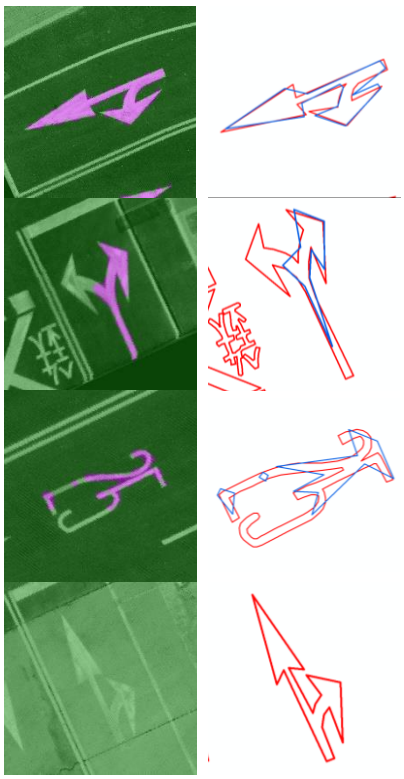

Figure 12. Road arrow polyline classifications: (Top to Bottom) Fully Detected, Partially Detected, Wrongly Detected, and Not Detected. (Left) Segmented Image and (Right) arrow polyline, in which Red is manually digitized and Blue is detected from the neural network model.

For images derived from intensity values, there is not much change from the number of fully detected arrows from 10 cm and 5 cm. But, the number of partially detected arrows did increase at 5 cm. This means that there is an increase in the detection accuracy from 10 cm to 5 cm. However, the number of wrongly detected features has gradually increased from 10 cm to 1 cm. This indicates that higher resolutions or more detailed images can provide more noise that confuses the network. For images derived from color (RGB) values, the number of fully detected

arrows decrease and the number of arrows not detected increase from 10 cm to 1 cm. This means that the detection process is getting worse as the sampling resolution gets higher. This can be caused by pixels having an insufficient number of points with wrong color information (due to wrong projections of color in the LIDAR point cloud). However, the number of wrongly detected features also decreases. This means that, in contrast with the images derived from the intensity, color images provided less noise, which can be due to a pixel having a more distinct representation of itself as compared to a grayscale representation of the intensity-based image. In general, images derived from intensity values provided more fully detected road arrows than the images derived from color (RGB) values. Having around 20%, 25%, and 40% differences in their number of detections for sampling resolutions of 10cm, 5cm, and 1 cm, respectively.

In terms of arrows not detected, there were also some cases that arrows were visible on both the intensity-based and color-based images but not detected. These are arrows that were located inside of tunnels, as shown in Figure 13. These images were not included in the training dataset and can be a source of confusion for the model since it has a very different condition. In total there were 2 blocks of this condition and it contained 7 fully represented arrows. So, if this were to be excluded in the assessment the 5 cm intensity-based image and the 10 cm color-based image would no longer have arrows in the not detected classification. In terms of wrongly detected features, since a closed polyline was counted as 1 feature no matter the size and its proximity to another polyline, cases like the one in Figure 15 greatly increases the number of features classified as wrongly detected. There was also a case like Figure 16, in which that type of marking was not found in the training data which confused the model, which also greatly contributed to the increase of wrongly detected features.

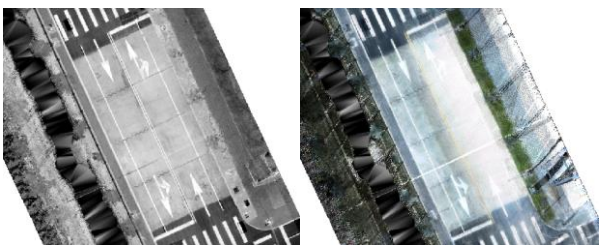

Figure 13. Sample zoomed-in for both intensity-based and color-based images that contain arrows inside tunnels.

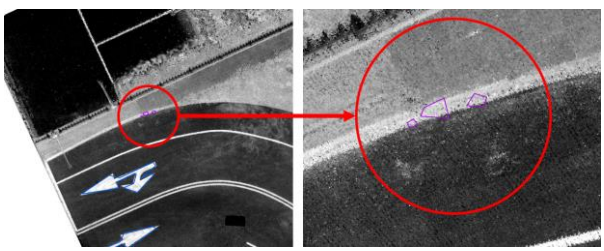

Figure 14. Sample of 3 wrongly detected features.

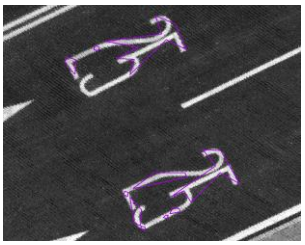

Figure 15. Road markings of the number 25 (in this case it indicates speed limit).

Figures 17, 19 and 21 show sample results of road arrow detection and vectorization for intensity-based images and Figures 20, 22 and 24 show the results for color-based images. Each of those figures was arranged in the manner of:

- from top to bottom: image, segmented image, and polyline representation;
- from left to right: 10cm, 5cm, and 1cm sampling resolutions;
- and for polyline representations, the red polyline is the manually digitized and the blue polyline is the neural network model detection results.

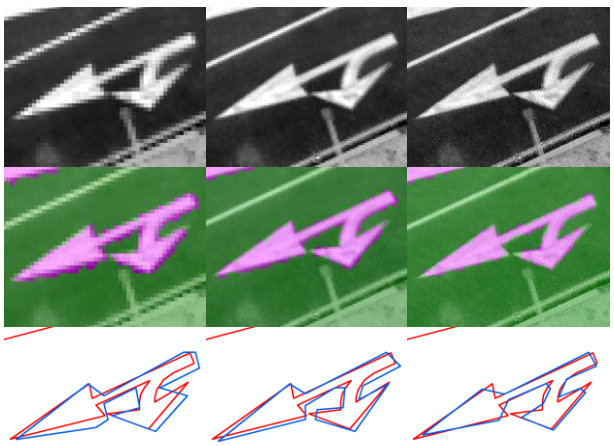

Figure 16. Sample arrow #1 – head straight or turn right (intensity-based).

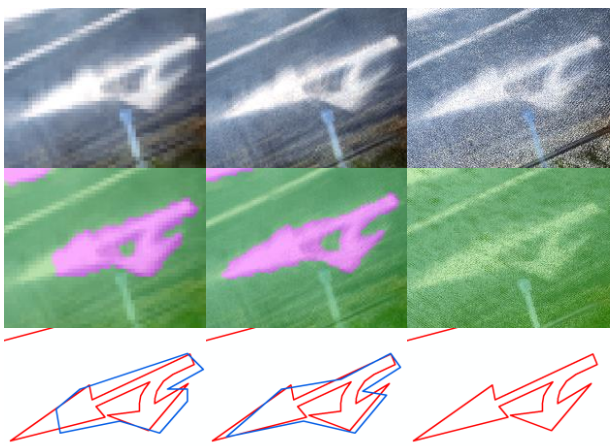

Figure 17. Sample arrow #1 – head straight or turn right (color-based).

https://doi.org/10.5194/isprs-archives-XLIII-B5-2020-123-2020

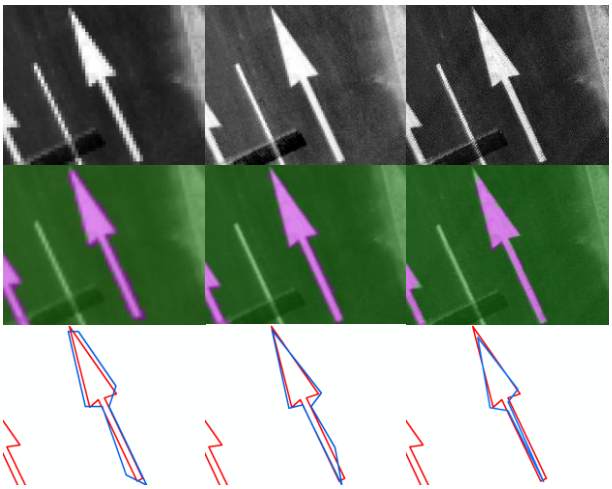

Figure 18. Sample arrow #2 – head straight (intensity-based).

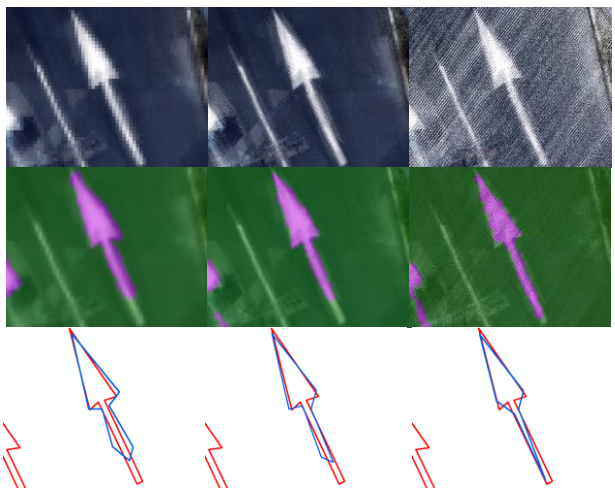

Figure 19. Sample arrow #2 – head straight (color-based).

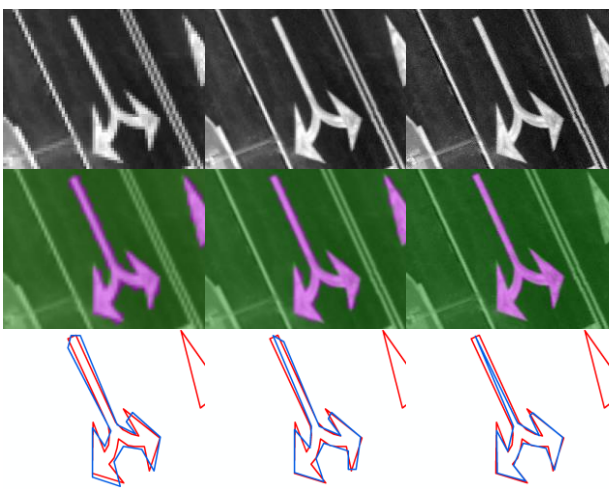

Figure 20. Sample arrow #3 – turn left or right (intensity-based).

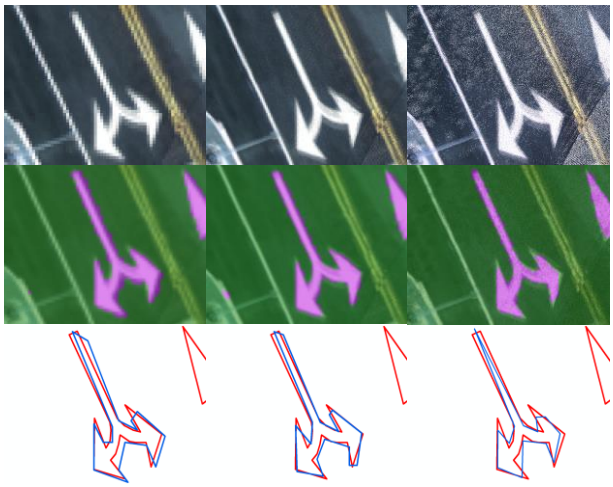

Figure 21. Sample arrow #3 – turn left or right (color-based).

4. CONCLUSIONS

The proposed methodology has successfully been able to digitize road arrow markings as polyline shapefiles using deep learning techniques from LIDAR point cloud derived images, with comparable accuracy from previous works in terms of correctly detected features. Intensity-based images have proven to be better than color-based images in terms of acting as training data for road marking extraction. A higher sampling resolution does not automatically provide better results, it can lead to the production of more noise which can greatly confuse the network model. One simplification value does not fit well with all of the features, as can be visually compared from the segmentation results and the output polylines. It is recommended that (1) the point cloud be converted to an image using a different method, (2) test out different sets of training images and augmentation technique combinations, and (3) assess multiple simplification parameter values to determine what would work better for all of the detections. In future works (1) the types of road markings would be expanded (eg. crosswalks, lane lines, etc.), and (2) more detailed ways of assessing the accuracy of all the results will be implemented.

ACKNOWLEDGEMENTS

The authors would like to acknowledge all the support provided by the Ministry of the Interior of Taiwan (ROC).

REFERENCES

Ambalina, L., 2019. What is image annotation? – an intro to 5 image annotation services. Hackernoon. https://hackernoon.com/what-is-image-annotation-an-intro-to-5-image-annotation-services-yt6n3xfj. (30 January 2020)

Brummelen, J., O'Brien, M., Gruyer, D., & Najjaran, H., 2018. Autonomous vehicle perception: The technology of today and tomorrow. *ELSEVIER Transportation Research Part C: Emerging Technologies*, 384-406.

Ho, D., Liang, E., & Liaw, R., 2019. 1000x Faster data augmentation. Berkley artificial intelligence research. https://bair.berkeley.edu/blog/2019/06/07/data_aug/. (30 January 2020)

Hoang, T., Nam, S., & Park, K., 2019. Enhanced detection and recognition of road markings based on adaptive region of interest and deep learning. *IEEE Access*, 109817-109832.

Hu, J., Abubakar, S., Liu, S., Dai, X., Yang, G., & Sha, H., 2019. Near-infrared road-marking detection based on a modified faster regional convolutional neural network. *Hindawi Journal of Sensors*, 1-11.

Kurz, F., Azimi, S., Sheu, C.-Y., & D'Angelo, P., 2019. Deep learning segmentation and 3D reconstruction of road markings using multiview aerial imagery. *ISPRS International Journal of Geo-Information*, 1-16.

Li, Y., Guo, L., Rao, J., Xu, L., & Jin, S., 2019. Road segmentation based on hybrid convolutional network for high-resolution visible remote sensing image. *IEEE Geoscience and Remote Sensing Letters*, 613-617.

Supervisely Development Team. 2019. Automatic road segmentation. Supervisely. https://docs.supervise.ly/cookbook/real-world-use/auto-roads-segm/auto-roads-segm/. (30 January 2020)

Vardhan, H., 2017. HD maps: new age maps powering autonomous vehicles. Geospatial world. https://www.geospatialworld.net/article/hd-maps-autonomous-vehicles/. (30 January 2020)

Wen, C., Sun, X., Li, J., Wang, C., Guo, Y., & Habib, A., 2018. A deep learning framework for road marking extraction, classification, and completion from mobile laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 178-192.

Wolf, J., Richter, R., Discher, S., & Dollner, J., 2019. Applicability of neural networks for image classification on object detection in mobile mapping 3D point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 111-115.

Yang, X., Li, X., Ye, Y., Lau, R., Zhang, X., & Huang, X., 2019. Road detection and centerline extraction via deep recurrent convolutional neural network U-Net. *IEEE Transactions on Geoscience and Remote Sensing*, 7209-7220.