2023 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
AUTONOMY, ARTIFICIAL INTELLIGENCE & ROBOTICS (AAIR) TECHNICAL SESSION
AUGUST 15-17, 2023 - NOVI, MICHIGAN

ARTIFICIAL NEURAL NETWORK BASED TERRAIN RECONSTRUCTION FOR OFF-ROAD AUTONOMOUS VEHICLES USING LIDAR

Sarang Sutavani[1], Andrew Zheng[1], Ajinkya Joglekar[2], Jonathon Smereka[*], David Gorsich[*], Venkat Krovi[2], Umesh Vaidya[1]

[1]Department of Mechanical Engineering, Clemson University, Clemson, SC
[2]Department of Automotive Engineering, Clemson University International Center for Automotive Research (CU-ICAR), Greenville, SC

ABSTRACT

Accurate terrain mapping is of paramount importance for motion planning and safe navigation in unstructured terrain. LIDAR sensors provide a modality, in the form of a 3D point cloud, that can be used to estimate the elevation map of the surrounding environment. But, working with the 3D point cloud data turns out to be challenging. This is primarily due to the unstructured nature of the point clouds, relative sparsity of the data points, occlusions due to negative slopes and obstacles, and the high computational burden of traditional point cloud algorithms. We tackle these problems with the help of a learning-based, efficient data processing approach for vehicle-centric terrain reconstruction using a 3D LIDAR. The 3D LIDAR point cloud is projected on the ground plane, which is processed by a generative adversarial network (GAN) architecture in the form of an image to fill in the missing parts of the terrain heightmap. We train the GAN model on artificially generated datasets and show the method's effectiveness by means of the reconstructed terrains.

Citation: S. Sutavani, A. Zheng, A. Joglekar, J. Smereka., D. Gorsich, V. Krovi, U. Vaidya, "Artificial Neural Network based Terrain Reconstruction for Off-road Autonomous Vehicles using LIDAR," In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 15-17, 2023.

This work was supported by VIPR-GS

1. INTRODUCTION

There has been significant progress in autonomous navigation due to advancements in sensing, planning, and control. Mobile robots that navigate within structured environments such as urban areas are well established, and the scenario-specific plans such as lane changing and path following are well researched [1, 2, 3]. Many of these successes can be attributed to well-defined environments; however, this does not apply to offroad terrain navigation [4]. The offroad navigation problem has many unorthodox structures, such as the degree of traversability, irregular obstacles, secluded obstacles, etc., which makes the problem difficult.

One of the key aspects for successful navigation in unstructured terrains is the ability to sense and model the terrain height. [5] has shown that given solely elevation information from an external sensor, a grid map based on elevation data is sufficient to guide an autonomous system to navigate across unstructured terrain. 3D LIDARs are typically used to obtain accurate spatial data, called point cloud. The 3D LIDAR sweeps a LASER beam across the environment, registering the spatial coordinates of the environment based off the angle and duration the light took to reflect back into the LIDAR. Though the LIDARs provides a full robot centric view of the environment, working with 3D LIDARs in off-road navigation has its own set of challenges.

The LIDAR data generated in unstructured environments is non-uniform in nature. Moreover, LIDARs have a fixed resolution making the point cloud sparser as the distance increases. Terrain construction from sporadic point cloud is an area of active research [6, 7]. Furthermore, there may be not only large gaps due to resolution issues, but gaps in data points due to negative spaces. These gaps cannot be arbitrarily reconstructed due to the highly diverse obstacle structures within off-road environments [8]. Such issues make it difficult to use 3D LIDARs for off-road navigation.

In this work, inspired by the recent advancements in neural networks, we train a generative adversarial network (GAN) to generate appropriate elevation data for the LIDAR map in the unobserved part of the terrain. We show that the GAN model is able of efficiently process the point cloud data and construct reasonably accurate elevation maps.

2. PROPOSED APPROACH

Perception performance turns out to the bottleneck for many off-road autonomous systems [9]. Advanced driver-assistance systems (ADAS) [10] and automated driving (AD) systems [11] have been the focus of significant research efforts in recent years. The advances in the ADAS/AD technologies, especially the ones related to perception, are accelerated by the advances in the artificial neural network (ANN) based technologies. However, these efforts are skewed towards structured and well-mapped environments. The perception algorithms exploit the structured nature of the problem along with the similarities in the operating environments to achieve good performance. But, this performance does not carry over to the unknown off-road settings due to various factors including lack of appropriate training datasets, lack of testing scenarios that can be generalized to a large number of cases, degree of uncertainty in the operating conditions, etc. In [12], the authors advocated for the use of synthetic data as one of the means to bridge the performance gap. This is of-course predicated on the high fidelity of the synthetic data, to maintain acceptable performance after sim-to-real transfer. The approach we used in this article follows the aforementioned philosophy, as this gives us more control over the data quality and ensures that the data is in the required format for further processing.

We synthesized the required data from a simulation environment using the following process: i) generate a realistic heightmap with desired characteristics, ii) create a terrain in the simulation environment using the heightmap, generate the LIDAR scan in the simulation environment, process

the scan into a ground projected image, iii) use the ground projected image as the input and the original heightmap as the output for training the artificial neural network. There are some key advantages of the proposed ANN based image processing method for terrain reconstruction. The LIDAR data processing pipeline (for individual scans) can be compressed to a single forward pass of the ANN. Therefore, it is possible to run the terrain reconstruction in real time. Additionally, by adjusting the complexity of the neural network it can be implemented on devices with low memory and low compute power to run with low latency. The output of this procedure can be used as a starting point for other (computationally expensive) techniques to speed up the terrain reconstruction process. Or, it can also be used as a backup that requires only limited resources. Since, the LIDAR scan is represented in the form of an image, therefore well established methods from image processing can be used to improve the performance.

3 DATA GENERATION

The proposed approach uses heightmap images of unstructured terrain as the ground truth data. These heightmaps are used to train a neural network to produce terrain heightmaps when fed with sparse LIDAR data images. Since we are proposing a novel method for terrain estimation using heightmaps and ANN, finding labeled LIDAR data as per our requisites was a challenge. We required sufficient data from off-road heightmaps and LIDAR data as 3D point clouds. As a result, there was a need to generate the data set. We used Perlin noise python library, CoppeliaSim simulator, and Matlab software to do the same. A step-by-step layout of the data generation process is provided next.

3.1 Heightmap Generation

The first step in acquiring the data in the required format was to generate the 2.5D heightmap images. To do the same, we utilized the Perlin noise python library. Perlin noise [13] is a gradient noise used for procedurally generating realistic terrains. We utilized it to generate a realistic heightmap representation of rough terrain as given in Fig. 1. The used Perlin noise library implements a parameter called 'octaves'. Each octave represents a particular terrain type, such as mountains, boulders, rocks, etc., and adds a layer of detail to the surface. Increasing the number of octaves increases the variation in the terrain heightmap.

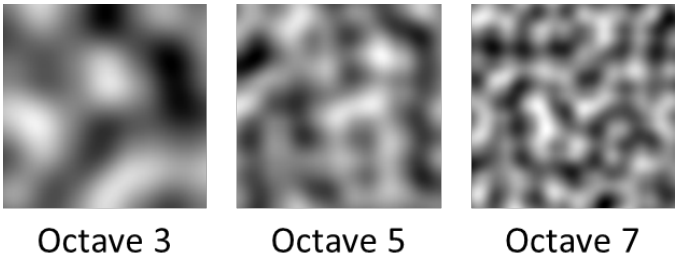

Figure 1: Heightmaps generated using Perlin Noise library for various octave values.

3.2 LIDAR Data generation

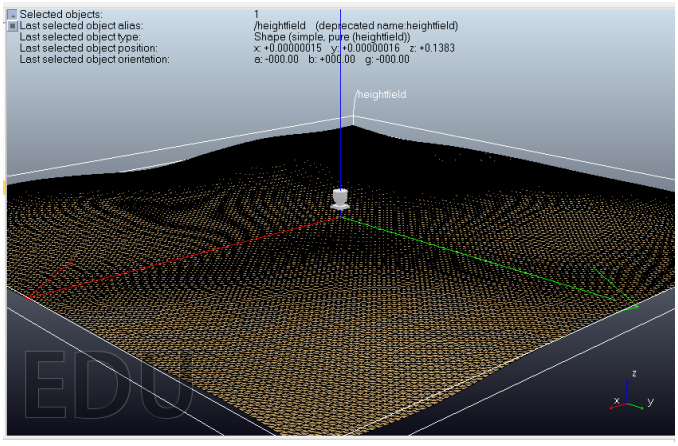

Figure 2: CoppeliaSim simulator is used to construct 3D terrain from the generated heightmap. LIDAR block is implemented to obtain 3D point cloud data.

We used CoppeliaSim software for generating 3D LIDAR point cloud data. CoppeliaSim, previously known as V-REP, is a simulator used in industry, education, and research. We imported the heightmap

generated in the previous step to the simulation environment of CoppeliaSim to construct a 3D representation (Fig. 2) of the terrain. Then using a LIDAR block available in the CoppeliaSim, we generate the required LIDAR point cloud in the simulation. This point cloud data is used to generate the LIDAR images.

3.3 Generation of 2D LIDAR images

The 3D point cloud data obtained from the CoppeliaSim simulator cannot be directly used for the training of neural network. This is because of the fact that there is no reference orientation for point cloud data. Different terrain surfaces will generate different point cloud data and without proper representation of the point cloud data, the data can not be utilized to train the neural network. Therefore some data pre-processing is required. For this purpose, we project the LIDAR data on a 2D plane to get an image of the LIDAR data. This is done with the help of MATLAB software.

3.4 Data Preprocessing and augmentation

After the generation of LIDAR images, we perform standard augmentation techniques such as rotating the image frames, flipping, etc., to increase the number of images in the dataset. We also add some noise to the LIDAR images to simulate the effect of sensor noise. This step helps restrict over-fitting and improve generalization. A sample image set of the LIDAR images can be seen in Fig. 3

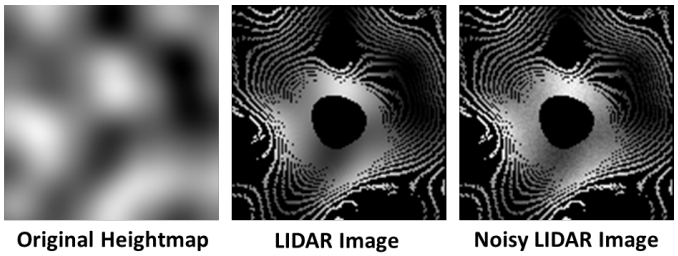

Figure 3: Ground projected images of a LIDAR scan with and without noise.

4 GENERATIVE ADVERSARIAL NETWORK ARCHITECTURE

A generative adversarial network (GAN) synthesizes images that are expected to look like real images. GANs consist of two main elements, a generator and a discriminator. The generator constructs a synthetic image from a latent noise vector and/or on some external input. The discriminator is tasked to distinguish the fake images, which the generator model generates, from the real images, which are the ground truth. Training the GAN is equivalent to solving for a solution of a two player adversarial game. In this game the generator tries to fool the discriminator by constructing an accurate looking synthetic image. The discriminator is trained to identify the fake images. To assist with the convergence of the game to some equilibrium the generator parameters are kept constant while the discriminator is training and vice versa.

Pix2Pix [14] is a conditional GAN (cGAN) architecture designed for image-to-image translation, where the generation target image is conditioned on the input. The synthetic output image can be compared with the source image to calculate the loss and update the weights of the GAN using gradient descent. Pix2Pix provides a general solution to various image synthesis tasks such as image inpainting, image colorization, edge to image reconstruction, etc. The generator is trained with the adversarial loss function and an additional term, i.e., L_1 loss function, which captures the loss between the generated image and the ground truth. The discriminator uses the PatchGAN structure, which allows for processing of the input image at various scales, i.e., from whole image to patches in the image as small as a pixel. The discriminator learns to distinguish between the patches that are part of a real image from those coming from a synthetic image. This approach also helps reduce the number of parameters and can be applied to high-resolution images. The Pix2Pix GAN uses pairwise image data (input image and the lable/ground truth image) for

training. We use the data obtained from the data generation pipeline described in the previous section to train the Pix2Pix architecture. The trained model takes an image that contains partial information of the terrain obtained through the LIDAR scan and provides an estimate on the geometry of the terrain not observed in the scan.

4.1 GAN loss function

We consider the conditional GAN loss function from [14]. $\mathcal{L}_{GAN}$ is the generative adversarial network cost. The cost function represents a *min-max game* between the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$. While training the discriminator, the generator training parameters are kept fixed and we have

$$\mathcal{L}^{\mathcal{D}}_{GAN}(x) = y\ log(\mathcal{D}(x)) + (1-y)\ log(1-\mathcal{D}(\mathcal{G}(u))). \quad (1)$$

The input u to the generator is generally a combination of random noise and some additional features. We use the ground projected LIDAR image as our feature. x is the input to the discriminator. The label associated with x is $y = 1$ for the real images and $y = 0$ for the fake images fed to the discriminator. When training the generator, the discriminator training parameters kept fixed, and we have

$$\mathcal{L}^{\mathcal{G}}_{GAN} = log(\mathcal{D}(\mathcal{G}(u))) + \lambda\ L_1, \quad (2)$$

which accounts for the generator being able to fool the discriminator. The L_1 loss term is also called the reconstruction loss function.

Around 1000 ground truth and LIDAR images were included in our training datasets. We used data augmentation techniques to expand the dataset for more robustness. The set of hyperparameters used is shown in Table 1. The learning rate for the discriminator and generator was kept constant. The generator in Pix2Pix follows the U-Net architecture. We designed the U-Net structure according to our dataset requirement, to handle the input images of size $121 \times 121 \times 1$ (i.e., single channel or gray scale images). The kernel size and number of filters in the encoder and decoder are shown in Table 2. We trained a network for 60 epochs per experiment using the NVIDIA GeForce RTX 2070 SUPER graphics card. The average time for each training session was approximately two hours.

Table 1: Hyperparameters for the experiment.

Hyperparameter	**Value**
Epochs	60
Batch size	1
Learning rate of discriminator	$2e^{-4}$
Learning rate of generator	$2e^{-4}$
Adam β_1	0.5
Lambda (λ)	100

Table 2: Generator U-net architecture

Stage	Filters	Size	Stride	Padding
Encoder (Convolution 2D)	64	3	2	valid
	128	4	2	same
	256	4	2	valid
	512	4	2	same
	512	4	2	same
	512	4	2	same
	512	4	2	same
Decoder (Transpose Convolution 2D)	512	4	2	same
	512	4	2	same
	512	4	1	valid
	512	4	2	same
	256	4	2	valid
	128	4	2	same

5 RESULTS

We conducted several experiments on our dataset to test the reliability of the approach. We used octave 3 and octave 5 data along with 5% and 10% noise to make the dataset more realistic. In Figure 4, the variation in GAN losses against the epochs is depicted. There are three loss terms represented: loss of the discriminator, loss of the generator, and the L_1 loss. The trend of the losses suggests convergence

towards a saddle point. Discriminator loss shows the capability of the discriminator to predict real vs. fake images. L_1 loss is the error in the target and generated image. Among all presented cases, as shown in Figure 4, the network generates synthetic images efficiently for the octave 5 dataset since the loss is lesser compared to the other cases. A smaller value of L_1 loss indicates a higher degree of similarity between ground truth and generated image.

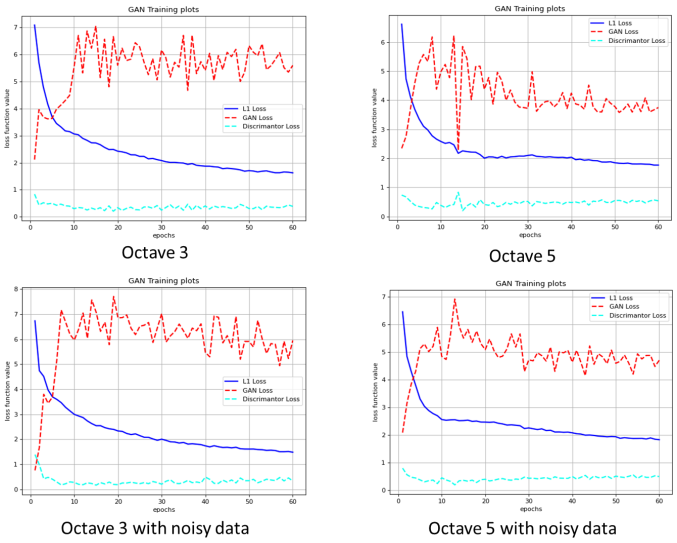

Figure 4: GAN losses.

In Figure 5, we used an octave 3 dataset with 10% noise to reconstruct the terrain. It can be seen that even with 10% noise, reconstructed terrain looks similar to that of ground truth terrain. We can observe from Figure 6, that the predicted image matches very well with the ground truth image from the octave 3 dataset with 10% noise.

We further applied the proposed method for estimating the obstacle shapes only using sparse LIDAR scans. For training we used flat terrain with cylindrical obstacles. Figure 7 shows the obstacle estimation of the GAN on a test example from obstacle only setup. We can observe that the neural network provides a good estimation of obstacle positions and shapes, except for the cases where an obstacle is in the cover shadow of another obstacle (which is expected).

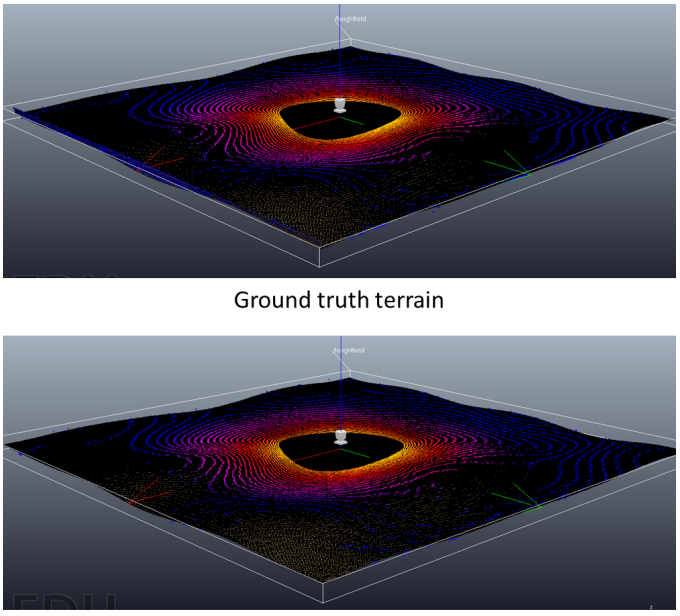

Figure 5: Original and reconstructed terrains.

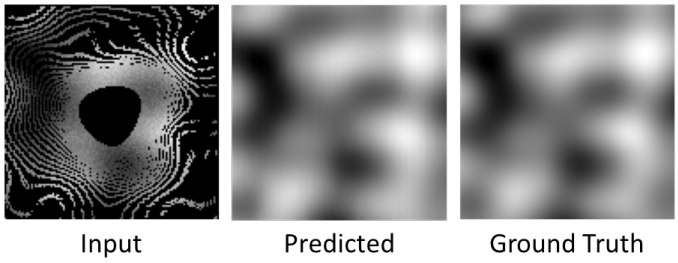

Figure 6: GAN prediction on test example.

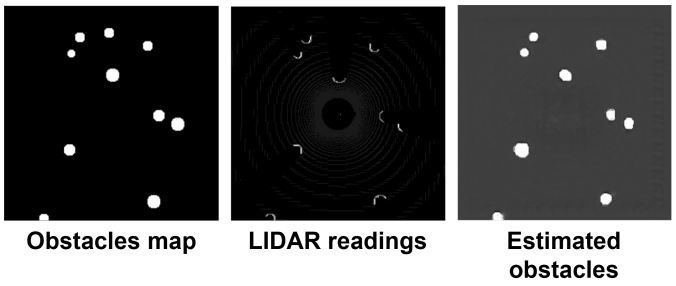

Figure 7: Obstacle prediction on a flat surface.

Figure 8 shows the performance of the proposed approach for the case of slight terrain variation and negative obstacles (representing holes or ditches in the terrain). Negative obstacle prediction is

in general more challenging and we can see here that the generator is struggling with the obstacle reconstruction. The accuracy in the estimation is partly affected by both the sparsity of the LIDAR data, as can be seen from in Figure 8, and partly due to the fact that the negative obstacle do not allow the LASER beams to return to the LIDAR, eliminating very important pieces of information.

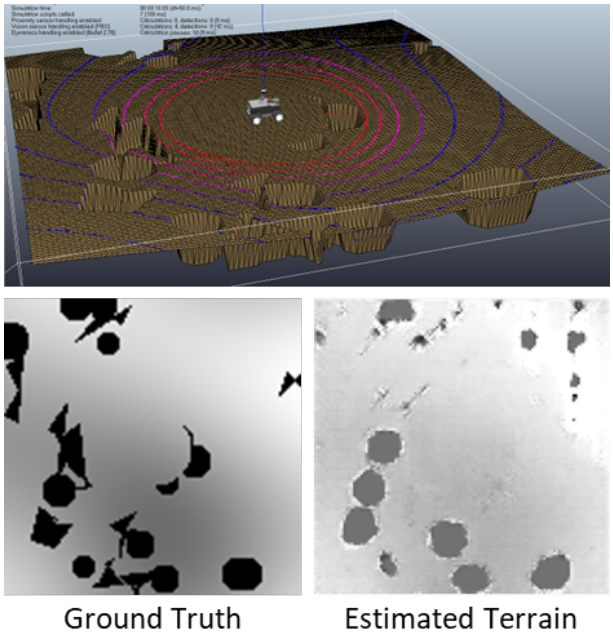

Figure 8: Negative obstacle prediction.

6 CONCLUSION AND FUTURE WORK

The problem of terrain heightmap reconstruction from the LIDAR data is addressed with the help of a deep convolutional neural network approach. The approach of the paper is novel and intuitive. The results obtained using Pix2Pix conditional GAN architecture are promising and indicate that this is a good avenue for further exploration. The reconstructed terrains look similar to the original terrains, but some unwanted artifacts could be observed in the prediction. There could be multiple ways to improve the predictions, including high-frequency noise filters, additional prediction masks, etc. The datasets used were synthetic ones due to the unavailability of the suitably labeled dataset. This work could be naturally extended by creating and training on real datasets.

7. REFERENCES

References

[1] C. Fernández, M. Gavilán, D. F. Llorca, I. Parra, R. Quintero, A. G. Lorente, L. Vlacic, and M. Sotelo, "Free space and speed humps detection using lidar and vision for urban autonomous navigation," in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 698–703.

[2] A. C. Murtra, E. Trulls, O. Sandoval, J. Pérez-Ibarz, D. Vasquez, J. M. Mirats-Tur, M. Ferrer, and A. Sanfeliu, "Autonomous navigation for urban service mobile robots," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4141–4146.

[3] Z. Zheng, "Recent developments and research needs in modeling lane changing," *Transportation research part B: methodological*, vol. 60, pp. 16–32, 2014.

[4] J.-w. Hu, B.-y. Zheng, C. Wang, C.-h. Zhao, X.-l. Hou, Q. Pan, and Z. Xu, "A survey on multi-sensor fusion based obstacle detection for intelligent ground vehicles in off-road environments," *Frontiers of Information Technology & Electronic Engineering*, vol. 21, no. 5, pp. 675–692, 2020.

[5] P. Fankhauser, M. Bjelonic, C. D. Bellicoso, T. Miki, and M. Hutter, "Robust rough-terrain locomotion with a quadrupedal robot," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5761–5768.

[6] J. Serafin, E. Olson, and G. Grisetti, "Fast and robust 3d feature extraction from sparse

point clouds," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4105–4112.

[7] Y. Tatebe, D. Deguchi, Y. Kawanishi, I. Ide, H. Murase, and U. Sakai, "Pedestrian detection from sparse point-cloud using 3dcnn," in *2018 international workshop on Advanced Image Technology (IWAIT)*. IEEE, 2018, pp. 1–4.

[8] J. Larson and M. Trivedi, "Lidar based off-road negative obstacle detection and analysis," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2011, pp. 192–197.

[9] A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski *et al.*, "Toward reliable off road autonomous vehicles operating in challenging environments," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 449–483, 2006.

[10] A. Shaout, D. Colella, and S. Awad, "Advanced driver assistance systems-past, present and future," in *2011 Seventh International Computer Engineering Conference (ICENCO'2011)*. IEEE, 2011, pp. 72–82.

[11] C.-Y. Chan, "Advancements, prospects, and impacts of automated driving systems," *International journal of transportation science and technology*, vol. 6, no. 3, pp. 208–216, 2017.

[12] D. W. Carruth, C. T. Walden, C. Goodin, and S. C. Fuller, "Challenges in low infrastructure and off-road automated driving," in *2022 Fifth International Conference on Connected and Autonomous Driving (MetroCAD)*. IEEE, 2022, pp. 13–20.

[13] K. Perlin, "An image synthesizer," *ACM Siggraph Computer Graphics*, vol. 19, no. 3, pp. 287–296, 1985.

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