2024 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
MODELING, SIMULATION, PROTOTYPING & VALIDATION (MSPV) TECHNICAL SESSION
AUG. 13-15, 2024 - NOVI, MICHIGAN

FIDELITY REQUIREMENTS FOR DIGITAL SCENES IN OFF-ROAD AGV SIMULATION

Chris Goodin[1], Daniel W. Carruth[1], Lalitha Dabbiru[1], Michael Hedrick[1], Brandon Black[1], Zachary Aspin[2], Justin T. Carrillo[2], and John Kaniarz[3]

[1]Center for Advanced Vehicular Systems, Mississippi State University, Starkville, MS
[2]Engineer Research and Development Center, Vicksburg, MS
[3]Ground Vehicles System Center, Warren, MI

ABSTRACT

While simulation is being used extensively in the development and testing of autonomous ground vehicles, many fundamental questions remain regarding the required characteristics of autonomous simulation. One such question is, "How much geometric fidelity is required to realistically simulate off-road digital scenes?" In this work, we construct a series of experiments to answer this question, quantitatively, for both lidar and camera simulations. By using machine learning classification algorithms, we are able to determine the level of geometric fidelity required to have a trained neural-network based algorithm correctly classify trees in images. We find that the meshes must have a triangle density of $> 1000/m^3$ *to perform as well as real data. In additional analysis on lidar scans, we find that vegetation meshes must have* > 1000 *total triangles to yield a realistic distribution of scan ranges.*

Citation: C. Goodin, D.W. Carruth, L. Dabbiru, M. Hedrick, B. Black, J.T. Carrillo, J. Kaniarz, "Fidelity Requirements for Digital Scenes in Off-Road AGV Simulation", In *Proceedings of the 2024 Ground Vehicle Systems Engineering and Technology Symposium (GVSETS)*, NDIA, Novi, MI, Aug. 13-15, 2024 .

1. INTRODUCTION

Simulation has become an important part of developing and testing autonomous ground vehicles (AGV) [1]. However, there are still many unanswered questions related to the properties of simulation tools. The most fundamental question about AGV simulation relates to validation. While simulations of purely physical processes (like fluid flow) may have analytical test cases with exact solutions that enable validation, there are no such cases for AGV. Autonomous vehicles include a mix of hardware and software that often makes duplicating test results difficult, which in turn makes validation difficult, especially since AGV simulation

incorporates multiple physics domains (*ie* tire-soil interaction, sensor physics, vehicle dynamics, etc.).

In the preeminent work on simulation validation [2], Sargent lists a number of validation methods including *Comparison to Other Models*, *Historical Data Validation*, and *Extreme Condition Tests*, among others. However, the method for AGV simulation validation that is used most often is *Face Validity*, defined by Sargent as the process whereby "Individuals knowledgeable about the system are asked whether the model and/or its behavior are reasonable." Other recent studies have relied on the *Event Validity* method by comparing the failure rate in system level tests [3].

One interesting validation method defined by Sargent is the *Turing Test* where "Individuals who are knowledgeable about the operations of the system being modeled are asked if they can discriminate between system and model outputs." In this work, we use a modified version of the *Turing Test* validation technique to measure the importance of geometric fidelity in digital scenes for creating simulated images and lidar point clouds for use in AGV simulations. Our modification to the technique proposed by Sargent is that rather than using "Individuals who are knowledgeable about the operations of the system," we use the classification algorithms that are being used on the raw sensor data as the evaluator. We observe the difference between the classification results when using real and simulated input data. This could also be characterized as relying on "Face Validity", but using a software algorithm as the arbiter of validity, rather than a human.

When studying autonomy and AGV, it is important to use a software-based "expert" in the Turing Test / Face Validity methods, especially when evaluating the validity of synthetic imagery, because human observers often focus on image features that have little relevance to machine-learning algorithms [4]. Therefore, when evaluating the suitability of simulated imagery for AGV simulation, it is important to use the same "arbiter" for validation as will ultimately be consuming the data. Therefore, for this work, we chose the well known and popular googlenet classifier [5] as the "Expert" for the Turing test validation approach.

2. MATERIALS AND METHODS

There are many different aspects of sensor simulation fidelity that may be investigated, including but not limited to the geometric fidelity of the meshes and models comprising the scene, the realism of the algorithm used simulate the transport of radiation through the environment, the quality of the simulation of sensor responses, and the completeness of the material attributes of the objects in the scene. Moreover, there are additional fidelity attributes associated with simulating the vehicle platform itself. In this work, we choose to focus on one quantitative measure of fidelity, namely, the geometric fidelity of the digital scene. Geometric fidelity is straightforward to quantify, and can have a variety of measures such as triangle density (#/m^3), total number of triangles, or the mean maximum dimension of triangles. For this work, we used triangle density as the primary metric, in particular because of the necessity for realistic modeling of vegetation, which has been found to be particularly important for off-road navigation of AGV [6].

2.1. Perception Algorithm

Because of the importance of identifying vegetation for off-road AGV navigation, we chose to use a tree classification algorithm as our arbiter of validity in our modified Turing Test method. We chose the googlenet classifier [5] pretrained on the Imagenet dataset [7] and used the MATLAB Deep Learning toolbox to retrain the classifier on five new classes that are relevant for autonomous navigation. We trained the classifier using real data (Figure 1) and tested on synthetic data.

We chose a relatively small number of distinct classes so that the classifier could be trained with

a b c

Figure 1: Example real training data for the *Tree* (a), *ConstructionBarrel* (b), and *JerseyBarrier* (c) classes.

high accuracy on a dataset relevant to navigation. The training dataset consisted of five classes: *ConstructionBarrel* (35 images), *JerseyBarrier* (30 images), *Tree* (68 images), *HMMWV* (41 images), and *MRZR* (42 images). The input dataset, comprised of real images, was split into 70% training and a 30% test set, and trained over 10 epochs with 15 iterations per epoch. The validation accuracy at the end of this retraining was 96.88%.

2.2. Generating Synthetic Images

There are two important aspects when generating synthetic imagery, the digital scene, and the rendering engine used to create an image on that scene. In order to create the most generalized synthetic dataset possible, we used two different simulators to generate images on the same scenes in this work. The first was the MSU Autonomous Vehicle Simulator (MAVS), a high-fidelity, physics-based simulator for AGV in off-road terrain [8]. The MAVS has been used to study machine learning in a variety of applications using both camera and lidar data [9–11]. In addition to MAVS, the Unreal Engine version 4 (UE4) was used to generate images [12].

Because we were evaluating the differences in tree mesh fidelities, a wide variety of different types of tree meshes were used, as shown in Figure 2. There were 21 different tree meshes total analyzed in this work.

The meshes shown in Figures 2(a) and 2(b)) were actually two-dimensional. In this case, a nominal width of 0.1 meters was assigned in the missing dimension in order to calculate the volume. The resulting datasets are summarized Table 1.

Table 1: Synthetic datasets generated for this work

Dataset	**#**	**Description**
MAVS-1	1000	All classes, ray-traced
MAVS-2	7545	Trees only, ray-traced
MAVS-3	250	Trees only, path-traced
UE4-1	150	Trees only, light-mapped
UE4-2	125	All except trees, light-mapped

2.3. Generating Synthetic Camera Data

MAVS was used to generate synthetic datasets using two different camera modes. The first mode uses a computationally expensive path-tracing technique that solves for full global illumination, while the second mode uses a primary-ray only ray-tracing technique that rendered about 1000 times faster. For both cameras, images were generated using different terrains like deserts, forests, and cities. The camera and lighting properties, as well as the camera position and orientation, were varied randomly.

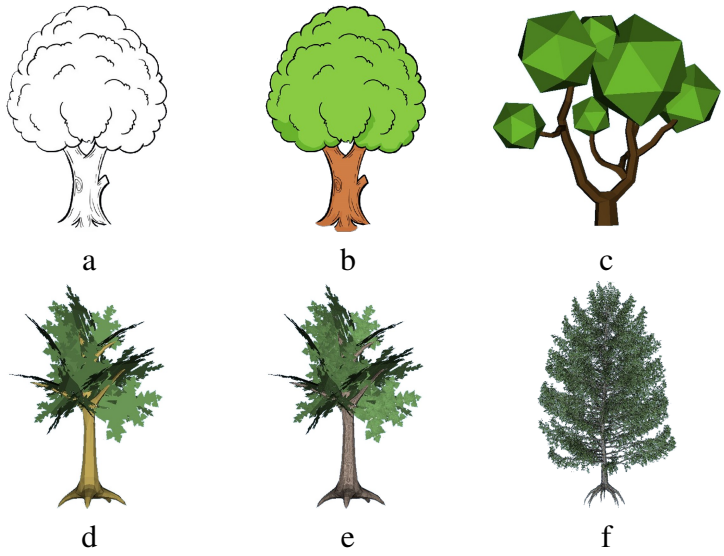

Figure 2: Example of different tree meshes with 2 (a,b), 500 (c), 1094 (d,e), and 496,280 (f) triangles.

MAVS was used to generate three unique datasets. The first dataset had 100 unique scenes and 10 images were generated in each scene, for a total of 1000 images. In the second dataset, 503 different scenes were generated with 15 images acquired in each scene for a total of 7545 images. The second dataset only contained images of the different tree models. Finally, the third MAVS dataset used the path-tracing camera mode and contained 250 images (50 scenes, 5 images per scene) of different tree meshes.

The same tree meshes that were used to create the MAVS scenes were used to create similar scenes in UE4. After importing the trees into a UE4 scene, images were created using the screenshot tool in the game mode. In comparison to the lower fidelity ray-traced MAVS model and the higher fidelity path-traced MAVS model, the UE4 camera model could be considered medium fidelity because it does solve the full global illumination problem like the path-tracer, but it does so using light-maps, which may yield a "clunky" appearance to the shadowing.

2.4. Lidar

We considered lidar point clouds in addition to images in our analysis. Scanning vegetation with lidar will typically produce a distribution of ranges in the reflected returns, in contrast to scanning a solid object which will produce a sharper edge in the point cloud [13]. This has been shown to be true for both terrestrial [14] and aerial [15] lidar sensors. This distributed return range is a characteristic of lidar-vegetation interaction that has been used as a discriminator for vegetation classification in off-road navigation [16]. For this reason, the range distribution of vegetation scans is chosen as a fidelity metric for evaluating the fidelity requirements of meshes for simulations of lidar sensors.

The same trees used in the camera analysis (Figure 2) were also used in the lidar analysis. In the following discussion we appropriate the method of Nie *et al.* [15], who examined the relationship between laser penetration index (LPI) and the distribution of ranges in vegetation. While the work of [15] is for discrete return aerial lidar, the similarity between the operating principle for discrete return aerial lidar and terrestrial lidar allows the findings

from [15] to be applied to the AGV application studied in this work.

The LPI for a discrete return lidar scanning vegetation from above is [15]

$$LPI = \frac{N_g}{N_g + N_v} \tag{1}$$

where N_g is the number of points that return from the ground and N_v is the number of points that return from the vegetation. For a scan from above, the coefficient of variation of the height distribution is [15]

$$CV_h = \frac{\sigma_h}{\mu_h} \tag{2}$$

where σ_h is the standard deviation of the height distribution and μ_h is the mean of the height distribution.

We generated synthetic point clouds by scanning the same trees used in the image analysis from a top-down orientation with lidar sensor, were it was placed above the tree and scanned toward the ground, allowing us to use Equations 1 and 2 to analyze the height distribution of the resulting point cloud.

3. EXPERIMENT RESULTS

The results of the camera and lidar simulation showed distinct crossover points in the chosen metrics (classification accuracy, LPI, respectively), that allowed us to make concrete recommendations for mesh fidelity requirements. These will be discussed in detail in the following subsections.

3.1. Camera

The classification algorithm was run on each dataset described in Section 2.2. The retrained googlenet algorithm gives a classification and a confidence level for each image that is classified, as shown in Figure 3. Figures $3a$, $3c$, $3d$, and $3e$ were generated by the MAVS ray-tracing rendering mode. Figures $3c$, $3d$, and $3h$ were generated by UE4, and Figures $3b$ and $3e$ were generated by the MAVS path-tracing mode. The dependent variable in this analysis is classification confidence. Googlenet provides a classification and confidence for each image. The confidence numbers were averaged for each different mesh or model. In the case that an image was misclassified, a zero was included in the confidence average.

Figure 4 shows the average confidence of the classification tests for vegetation only, as a function of triangle density in the mesh. The triangle density is plotted on a log scale, while the confidence has a theoretical maximum of 1.

We note that there were also differences in the different rendering modes (the MAVS ray-tracing, UE4 light-mapping, and MAVS path-tracing). In order to quantify the differences in rendering modes, we consider tree models with $> 1000/m^3$. Table 2 shows the summarized results for the different classes and cameras.

Table 2: Classification confidence for different camera simulations, compared to real

Class	Low	Med	High	Real
ConstructionBarrel	0.948	0.998		0.998
HMMWV	0.796	0.999		0.999
JerseyBarrier	0.846	0.998		0.997
MRZR	0.870	0.993		0.997
Tree	0.930	0.999	0.992	0.985

3.2. Lidar

The results of the lidar simulation analysis is shown in Figure 5. In this plot the LPI (Equation 1) is plotted on the y-axis versus the coefficient of variation CV_h (Equation 2) on the x-axis. According to [15], these values should be linearly related. Indeed, we see in our results that almost all of the tree meshes we simulated fall along a linear relationship.

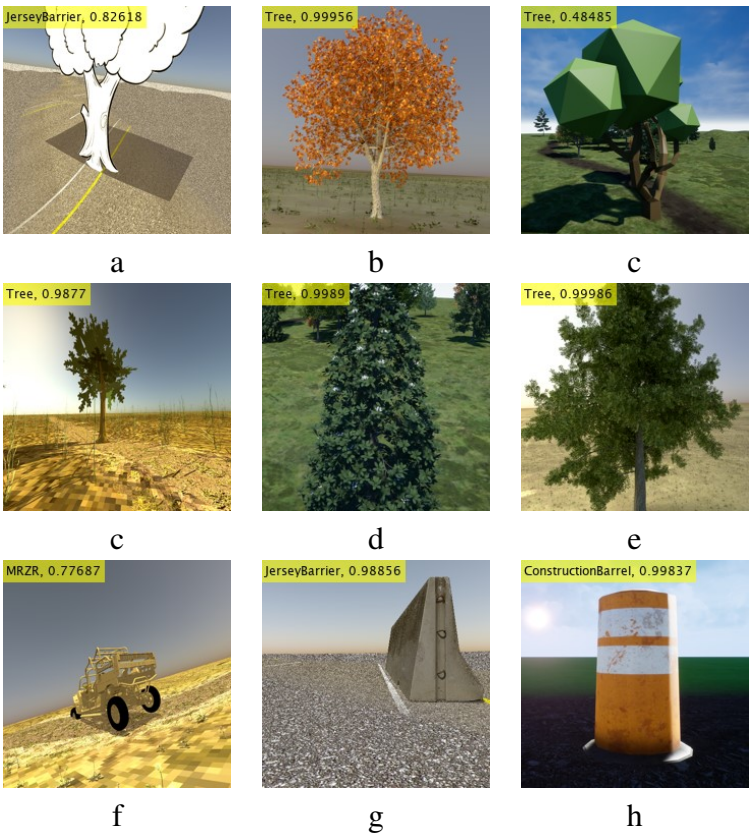

Figure 3: Classification results for different models and cameras

However, the two lowest fidelity meshes we tested, shown as image overlays in Figure 5, did not fit the linear trend observed with the rest of the tree meshes. Both of the non-conforming meshes had less than 1000 triangles, while all meshes with more than 1000 triangles conformed to the linear trend that is to be expected for real vegetation. Therefore, we conclude that 1000 triangles is the cutoff point for realistic vegetation modeling with respect to lidar. We note too because LPI and CV_h are unitless numbers, we can make no conclusions about mesh triangle density as we did for the images, only the total number of triangles.

4. DISCUSSION AND CONCLUSIONS

Figure 4 shows the classification confidence from the retrained googlenet classifier for the vegetation meshes as a function of triangle density. Although the data are noisy, it is clear that the confidence is near 100% for tree meshes with a triangle density $> 1000/m^3$. From Figure 4 it is also clear that the highest fidelity camera (MAVS path-tracer) and intermediate camera (UE4 screen grabber) both matched the confidence level of the real data (red line), while the lowest camera fidelity (MAVS ray-tracer) did not. While the path-tracer and light-map rendering both solve for full global illumination, the ray-tracer does not. Therefore, we conclude that solving for global illumination, even using a lower-fidelity method like light mapping, is important for matching the real results.

For lidar, using the well-known linear relationship between LPI and return height variation when scanning lidar from an aerial perspective, we

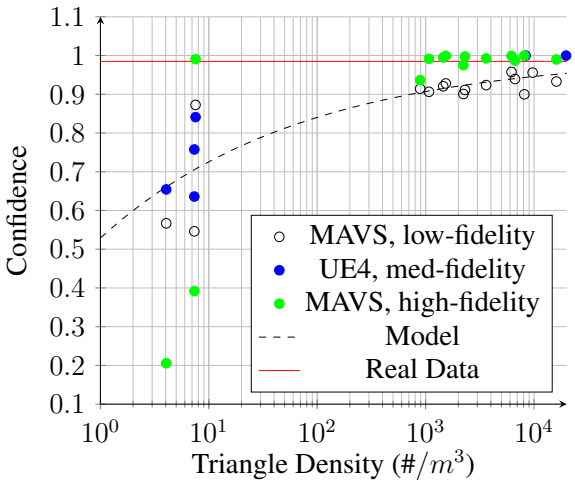

Figure 4: Tree classification confidence for MAVS and UE4, versus triangle density of the tree mesh

determined that meshes of $>$ 1000 triangles are required to properly simulate the statistical characteristics of lidar point clouds generated by scanning vegetation. Therefore, we draw the following conclusions for vegetation mesh fidelity requirements for AGV simulation:

- Tree meshes should have a triangle density $>$ $1000/m^3$ for images
- The rendering engine should solve full global illumination for images
- Tree meshes should have $>$ 1000 total triangles for lidar simulation

To summarize the findings of this paper, we attempted to validated synthetic image generation for off-road AGV simulators using a modified "Turing Test" approach where the arbiter was itself a retrained googlenet algorithm, trained on real data. We examined the influence of both geometric and rendering fidelity and found that vegetation meshes with triangle density $>$ $1000/m^3$ and rendered using global illumination are necessary to "fool" the Turing Test arbiter into classifying the images with the same confidence as is it does the real training

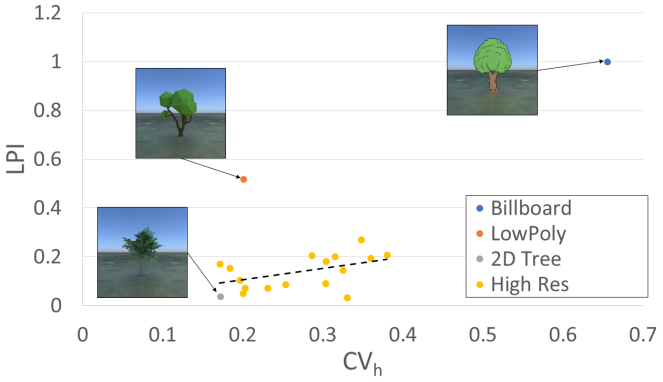

Figure 5: Lidar scan porosity versus coefficient of variation

data. Furthermore, we conclude that tree meshes should have $>$ 1000 triangles to give realistic range distributions in lidar simulation. Future work in this area will focus on continuing to address other aspects of simulation fidelity like material attributes.

Acknowledgments

This work was performed using funding from the US Department of Defense (DOD) High Performance Computing Modernization Program (HPCMP) under contract W912HZ-22-C-0004. DISTRIBUTION A. Approved for public release; distribution unlimited. OPSEC #:ERDC 24-033-CHG.

References

[1] J. Brabbs, S. Lohrer, P. Kwashnak, P. Bounker, and M. Brudnak, "M&s as the key enabler for autonomy development, acquisition and testing," in *Ground Vehicle Systems Engineering and Technology Symposium*, 2019.

[2] R. G. Sargent, "Verification and validation of simulation models," in *Proceedings of the 2010 winter simulation conference*. IEEE, 2010, pp. 166–183.

[3] C. Goodin, D. W. Carruth, L. Dabbiru, C. H. Hudson, L. D. Cagle, N. Scherrer, M. N. Moore, and P. Jayakumar, "Simulation-based testing of autonomous ground vehicles," in *Autonomous Systems: Sensors, Processing and Security for Ground, Air, Sea and Space Vehicles and Infrastructure 2022*, vol. 12115. SPIE, 2022, pp. 167–174.

[4] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox, "What makes good synthetic training data for learning disparity and optical flow estimation?" *International Journal of Computer Vision*, vol. 126, pp. 942–960, 2018.

[5] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha, "Deep learning algorithm for autonomous driving using googlenet," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 89–96.

[6] C. Goodin, M. Doude, C. Hudson, and D. Carruth, "Enabling off-road autonomous navigation-simulation of lidar in dense vegetation," *Electronics*, vol. 7, no. 9, p. 154, 2018.

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

[8] C. Hudson, C. Goodin, Z. Miller, W. Wheeler, and D. Carruth, "Mississippi state university autonomous vehicle simulation library," in *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium*, 2020, pp. 11–13.

[9] C. Goodin, S. Sharma, M. Doude, D. Carruth, L. Dabbiru, and C. Hudson, "Training of neural networks with automated labeling of simulated sensor data," SAE Technical Paper, Tech. Rep., 2019.

[10] L. Dabbiru, C. Goodin, N. Scherrer, and D. Carruth, "Lidar data segmentation in off-road environment using convolutional neural networks (cnn)," *SAE International Journal of Advances and Current Practices in Mobility*, vol. 2, no. 2020-01-0696, pp. 3288–3292, 2020.

[11] J. Boone, C. Goodin, L. Dabbiru, C. Hudson, L. Cagle, and D. Carruth, "Training artificial intelligence algorithms with automatically labelled uav data from physics-based simulation software," *Applied Sciences*, vol. 13, no. 1, p. 131, 2022.

[12] A. Sanders, *An introduction to Unreal engine 4*. AK Peters/CRC Press, 2016.

[13] J. Macedo, R. Manduchi, and L. Matthies, "Ladar-based discrimination of grass from obstacles for autonomous navigation," in *Experimental Robotics VII*. Springer, 2001, pp. 111–120.

[14] J.-E. Deschaud, D. Prasser, M. F. Dias, B. Browning, and P. Rander, "Automatic data driven vegetation modeling for lidar simulation," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 5030–5036.

[15] S. Nie, C. Wang, P. Dong, X. Xi, S. Luo, and H. Zhou, "Estimating leaf area index of maize using airborne discrete-return lidar data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 7, pp. 3259–3266, 2016.

[16] D. M. Bradley and J. A. Bagnell, "Domain adaptation for mobile robot navigation," Number CMU-RI-TR, Tech. Rep., 2010.