Artifact: Generating Realistic and Diverse Tests for LiDAR-Based Perception Systems

Garrett Christian
University of Virginia
Charlottesville, Virginia, USA
rda2tc@virginia.edu

Trey Woodlief
University of Virginia
Charlottesville, Virginia, USA
adw8dm@virginia.edu

Sebastian Elbaum
University of Virginia
Charlottesville, Virginia, USA
selbaum@virginia.edu

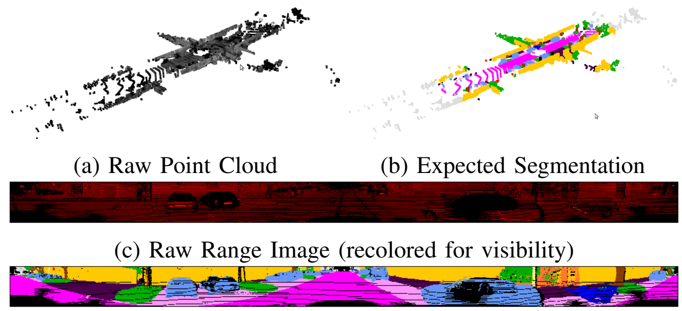
The artifact to evaluate Generating Realistic and Diverse Tests for LiDAR-Based Perception Systems is reusable and publicly released on GitHub along with additional study details and discussion. The Artifact includes the tool, perception systems, evaluation scripts, and data used in the original study and evaluation. The tool is written in Python and containerized in Docker with supporting Unix shell scripts to validate the tool and detailed descriptions for how to set up and extend the evaluation pipeline for adoption in future research. We provide a small data set along with scripts to demonstrate the end-to-end usage of the tool, with detailed instructions on how to obtain and set up the full data set from its authors. In addition to the tool itself, the evaluated versions of each of the perception systems are available along with the data obtained from the study.

I. Introduction

We introduce a new technique for testing LiDAR-based perception systems that takes as input existing real-world labeled LiDAR test cases, as shown in Fig 1, and mutates them to generate novel, realistic test cases that can expand the diversity of tests to explore the long-tail of possible scenes. We provide a research prototype tool that we use to explore and validate the technique. The tool is publicly released at https://archive.softwareheritage.org/browse/origin/directory/?origin_url=https://github.com/less-lab-uva/semLidarFuzz

The data set consists of pairs of LiDAR point clouds (PCs) collected from the real world and a labeling of these PCs based on the semantic interpretation of the points. From this data, the tool will produce realistic and novel test cases consisting of a novel PC and its labeling. Although many data sets may be applicable, our tool uses the publicly available SemanticKITTI [1] data set as the source for the labeled LiDAR data. The tool is written in Python and includes a MongoDB database for storage and supporting shell scripts for initialization. The tool is containerized in Docker with Docker Compose support for ease of use and replication.

Our technique also offers a new paradigm for evaluating the performance of LiDAR perception systems under test (SUTs). The five SUTs evaluated in the study are also containerized in Docker and publicly available. The tool provides Python code to evaluate the performance of each SUT on the generated test cases, and is extensible to allow for the addition of new SUTs.



(a) Raw Point Cloud (b) Expected Segmentation

(c) Raw Range Image (recolored for visibility)

(d) Expected Segmentation Range Image

Fig. 1: Example LiDAR Point Cloud (PC) used in perception (best viewed on a screen). Fig. 1a shows an example PC in perspective. Figs. 1a and 1c are colored with the reflected intensities of the LiDAR beams. Figs. 1b and 1d are colored with the human-annotated semantic class of each point. Figs. 1c and 1d show a "range image" where the 360° view is projected onto a cylinder and unrolled.

II. Motivation

Autonomous Systems (ASs) operate in safety-critical situations and preserving safety begins with an ASs ability to accurately perceive its surroundings. Perception failures of ASs have led to fatalities [2], including due to LiDAR-based perception [12]. However, it is particularly challenging to test LiDAR perception systems—a critical barrier as LiDAR is adopted by industry (e.g. [11], [13], [17]). Current testing techniques rely on large amounts of real world data, which is impractical due to the cost of collection and human labeling efforts leading to limited data, or simulation, which is limited in applicability by the simulation-reality gap [9], [10], [15]. This necessitates an approach that can provide novel, realistic data at a reasonable cost. Prior research has identified data augmentation as a method to diversify existing data sets; however, these techniques have focused on either diversity (e.g. [4], [5], [14], [16], [18]) or realism (e.g. [6], [8]) at the expense of the other. We address both realism and diversity at once by designing a set of mutation operations that increase test diversity, while also crafting a set of realism invariants that must hold during test generation. The final result is a set of novel and realistic test cases for LiDAR perception systems.

III. Implementation

Each mutation is implemented using Numpy [7], SemanticKITTI's developer tools [1], and open3D [19] Python libraries. The tool and all SUTs evaluated in the study are containerized in Docker for ease of replication.

The tool relies on the SemanticKITTI [1] data set as the basis for generating new test cases. However, the technique described in the paper is broadly applicable to any labeled LiDAR perception data set. To use a different data set, the tool must be adapted to its particular file format and labeling customs—we note that a consistent standard does not exist for either. For example, the SemanticKITTI data set has separate classes for the trunk of a tree and the tree's top, whereas the NuScenes [3] data set labels both as 'vegetation'. Such differences can impact the mutation operation, enforcement of realism invariants, and how SUTs are evaluated and transitioning to a different data set requires careful examination throughout the tool pipeline.

IV. Running the Artifact

A. Generating Tests

The `tool/` folder in the repository contains all of the tool source code. The `tool/tool_demo.sh` script runs a minimal example of the complete test generation pipeline. Initially, the script downloads a small subset of the SemanticKITTI data set and performs one-time processing to catalog available assets[1] Next, the script generates 5 test cases for each mutation; the script may be re-run to generate additional test cases. The perception systems tested reside in separate repositories; the tool repository contains additional instructions and scripts to automatically set up the SUTs.

B. Examining Study Results

The `study/` folder contains the code used to generate the tables and figures from the paper, as well as supplementary figures. The raw study results are publicly available[2]; the `study/generate_figures.sh` script downloads the study results and generates the figures and LaTeX-styled tables. The `study/README.md` file contains descriptions for each of the images produced and additional discussion about the parameters chosen for the study.

V. Requirements

The tool requires CUDA 10.0 support and was tested on Ubuntu 18.04. The shell (`.sh`) scripts provided to automate test execution are compatible with appropriate Unix shell environments. Tests were run on a pop-os 18.04 desktop with an Intel Xeon Silver 4216, 128GB of RAM, and an NVIDIA TITAN RTX GPU with 24GB of VRAM. The Docker environment files contain complete lists of all packages needed to run the tool and are automatically setup inside of the Docker container; if running outside of Docker, ensure that the appropriate packages are manually installed.

[1] To run on the full SemanticKITTI data set, the user must download and extract the data directly from the distributor: http://www.semantic-kitti.org/dataset.html. This is not included automatically due to the data set's size. Further instructions can be found in the repository.

[2] https://zenodo.org/record/7569212#.Y9P3KRzMK_4

Acknowledgements

This work was supported in part by funds provided by NSF#1924777, NSF#1909414, and AFOSR#FA9550-21-1-0164. Trey Woodlief was supported by a University of Virginia SEAS Fellowship.

References

[1] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[2] NTS Board. Collision between vehicle controlled by developmental automated driving system and pedestrian. nat. transpot. saf. board, washington, dc. Technical report, USA, Tech. Rep. HAR-19-03, 2019. URL https://www. ntsb. gov/investigations . . . , 2019.

[3] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[4] Jaeseok Choi, Yeji Song, and Nojun Kwak. Part-aware data augmentation for 3d object detection in point cloud. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3391–3397, 2021.

[5] Jin Fang, Xinxin Zuo, Dingfu Zhou, Shengze Jin, Sen Wang, and Liangjun Zhang. Lidar-aug: A general rendering-based augmentation framework for 3d object detection. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4708–4718, 2021.

[6] An Guo, Yang Feng, and Zhenyu Chen. Lirtest: Augmenting lidar point clouds for automated testing of autonomous driving systems. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 480–492, New York, NY, USA, 2022. Association for Computing Machinery.

[7] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[8] Frederik Hasecke, Martin Alsfasser, and Anton Kummert. What can be seen is what you get: Structure aware point cloud augmentation. 2022.

[9] Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*, pages 704–720. Springer, 1995.

[10] Waymo LLC. Waymo's safety methodologies and safety readiness determinations. Technical report, October 2020.

[11] Post author By Luminar Marketing. Luminar and volvo cars announce plans for next generation suv to be revealed in 2022, Jan 2022.

[12] Aarian Marshall. Uber video shows the kind of crash self-driving cars are made to avoid, Mar 2018.

[13] Jake Port. From the vault: Lidar: How self-driving cars 'see' where they're going, Jan 2022.

[14] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. *CoRR*, abs/1812.04244, 2018.

[15] Patrik Vacek, Otakar Jašek, Karel Zimmermann, and Tomáš Svoboda. Learning to predict lidar intensities. *IEEE Transactions on Intelligent Transportation Systems*, 23(4):3556–3564, 2022.

[16] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018.

[17] Yingzhi Yang, Zhang Yan, and Brenda Goh. Baidu ceo says ev arm's autonomous driving tech will be ahead of tesla. *Reuters*, Aug 2022.

[18] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *CoRR*, abs/2006.11275, 2020.

[19] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.