# Real2sim: Automatic Generation of Open Street Map Towns For Autonomous Driving Benchmarks

**Avishek Mandal**
University of Oxford, UK

**Panagiotis Tigas**
University of Oxford, UK

**Yarin Gal**
University of Oxford, UK

## Abstract

Research in machine learning for autonomous driving (AD) is a constantly evolving field as researchers strive to build a Level 5 autonomous driving system. However, current benchmarks for such learning algorithms do not satisfactorily allow researchers to evaluate and compare performance across safety-critical metrics such as generalizability, out-of-distribution performance, etc. Reasons for this include the expensive nature of data collection from the real-world for autonomous driving and the limitations of software tools currently available for autonomous driving simulators.

We develop a pipeline that allows for automatic generation of new town maps for simulator environments from `OpenStreetMap` [Haklay and Weber, 2008]. We demonstrate that our pipeline is capable of generating towns that, when perceived via `LiDAR` , share similar footprint to real-world gathered datasets like `NuScenes` [Caesar et al., 2020]. Additionally, we learn a realistic noise augmentation via *Conditional Adversarial Networks* [Isola et al., 2017] to further improve the similarity with real-world `LiDAR` . Our pipeline allows researchers for the first time to benchmark at scale various AD agents, both in-distribution, and out-of-distribution.

## 1 Introduction

The challenge of autonomous driving has received a lot of attention in these past two decades from a number of organizations, ranging from governmental entities such as Defense Advanced Research Projects Agency (DARPA) which ran the DARPA Grand Challenge in the mid/late 2000s [DARPA], to commercial actors like Weymo, Tesla or Toyota. Despite this attention over a long period of time, fully autonomous (Level 5) driving in urban environments remains an elusive problem to solve.

The main paradigm of developing autonomous driving agents is via simulated driving environments, such as `CARLA` [Dosovitskiy et al., 2017]. Simulators allow for quick turnaround of training and testing different Autonomous Driving (AD) agents, with the aim of deploying such agents in real world. Overcoming the so-called reality gap [Jakobi et al., 1995] requires significant engineering effort and measuring the safety and performance of the agents in real-world can be potentially risky. Existing benchmarking tools focus on a limited set of maps or real world data gathered in the wild [Caesar et al., 2020, Sun et al., 2020], which don't allow testing behaviors in an online fashion.

With this project we propose to overcome these limitations by bringing real world maps into a simulation environment (Real2Sim) to allow researchers to quickly test out-of-distribution and in-distribution scenarios.

Correspondence: `ptigas@robots.ox.ac.uk`

Machine Learning for Autonomous Driving Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

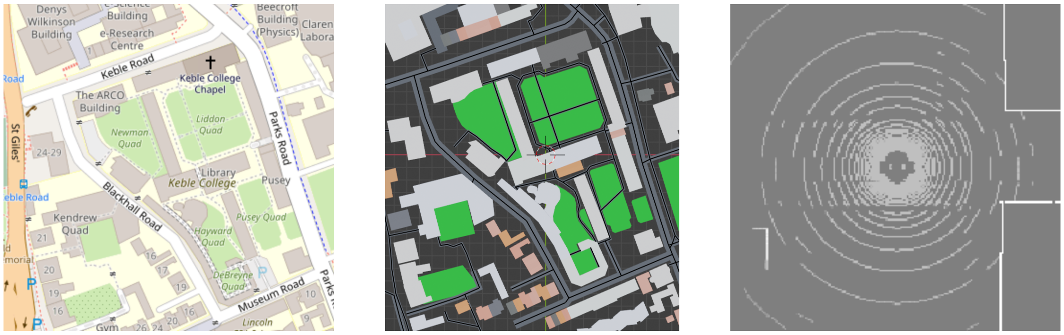


**Figure 1:** An example of the pipeline in action. (a) View of Keble College, Oxford, in Open Street Map. (b) The top down view in Blender, after importing and post processing. (c) `LiDAR` view from inside the `CARLA` simulator of the map object after the fbx and xodr files have been ingested.

We summarize the high-level desiderata below —

1. The pipeline should be easy for everyone to use.
2. The generated towns should contain diversity that mimic the complexity of the real world.
3. The pipeline should be Open Source for it to be usable and accessible to the AD community.

## 2 Related Work

There are several proposed benchmarks for autonomous driving, such as the nuScenes tasks [Caesar et al., 2020], the Waymo competition [Sun et al., 2020] or the CARLA Leaderboard . Many of these benchmarks share some similarities. They often involve solving a series of challenges — object detection challenges, trajectory forecasting challenges etc. — and tend to be in the form of a competition.

However, there are limitations to evaluating autonomous driving frameworks on these *static* sets of hold out data. These benchmarks are once again built on snapshots of a particular urban environment in a particular point in time. They don't allow a researcher to evaluate the performance of an autonomous driving agent in rare but plausible scenarios. For example, if we would want to examine how a AD vehicle would react if say, an animal jumped out from out of sight into the vehicle's path, these static datasets, most of which have not collected data about such rare occurrences, will not let the evaluation of the performance under such circumstances.

There are benchmarks available that allow for assessments of AD frameworks in such rare situations. One such benchmark, `CARNOVEL` (car novel-scene benchmark), introduced in Filos et al. [2020], is built on top a town provided by the `CARLA` simulator. The training set provided in this benchmark does not contain a number of situations — such as roundabouts and abnormal turns, and collectively referred to as out of distribution (OOD) scenarios — are only exposed during the evaluation. This benchmark too has limitations — The number of OOD scenarios that the performance of the AD framework can be evaluated on is bounded.

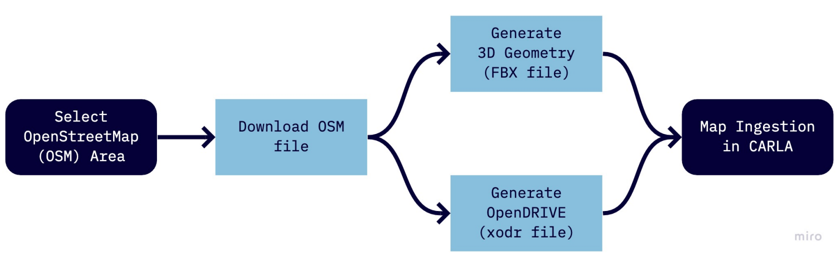


**Figure 2:** Over of the pipeline.

https://leaderboard.carla.org/challenge/

Table 1: Performance of the models with different LiDARs

| Source of LiDAR | NuScenes | CARLA | Blank |
|---|---|---|---|
| **Off-road metric** | 0.0719 | 0.100 | 0.177 |

## 3 The Pipeline

Here we describe an end-to-end automated pipeline. Each step of this process runs in a Docker instance which makes the system deployable as a service and scalable under load.

**Generation of 3D geometry**

Our target is to generate towns for a popular autonomous driving simulator, CARLA . A town consists of 3D geometry (FBX file) and annotated street layouts (OpenDRIVE), bundled together as a binary map which CARLA loads and allows for agents to be interact with it.

We use OpenStreetMap as a source to generate the towns, where we receive annotated a topology and additional metadata which help us generate the appropriate assets.

First, we use Blender to generate the geometries as it is a scriptable, open source, 3D editing software. We used blender-osm to generate the FBX files which we use for the generation of the towns. The main limitation of this approach is that although the high-level geometries are matched, details and textures are not generated, therefore we focus mostly on the high-level features. For automating blender we used Docker images provided by New York Times which allowed us to use it as a headless service.

**Generation of OpenDRIVE**

OpenDRIVE is an open format specification which is used by CARLA to extract the layout and metadata of the streets. To build a map, it is a requirement to extract an OpenDRIVE file from the OpenStreetMap location. For this, we are using netconvert, a tool which is part of the SUMO open-source package.

**Compile the map**

Given the FBX (geometry) file from Blender and the OpenDRIVE file from netconvert, we run the build process from CARLA to generate a binary map. The full pipeline can be seen in figure 2.

## 4 Experiments and Results

Our goal is to automate a pipeline that generates towns which can be used for autonomous driving benchmarking. However, removing too much fidelity can cast the generated town unusable for evaluation of AD agents.

### 4.1 Comparison with NuScenes

To evaluate the quality of the generated maps we devised a methodology which helped us compare against existing real-world, like NuScenes dataset.

We select one thousand scenes from NuScenes randomly. For each scene we use the lat/lon coordinate of the recording car to extract the corresponding OpenStreetMap map. We generate the town for that part of the city and we start CARLA , with an agent placed at the same lat/lon of the NuScenes scene. Then we use the LiDAR from CARLA to take a snapshot . That way, we construct a dataset which consists of the LiDAR images as taken from inside CARLA , for each of the NuScenes scenes. We call this CARLA LiDAR .

Next, we train a density model $p_\theta(s|\phi)$, like the one suggested in Rhinehart et al. [2018] ($s$ is the future trajectory and $\phi$ the LiDAR ) on the NuScenes dataset. We then use the trained model to assess the quality of our town by replacing the LiDAR from NuScenes with the CARLA gathered LiDAR image. First, we evaluate the downstream task performance (summarized in Table 1), as well

https://github.com/vvoovv/blender-osm
https://en.wikipedia.org/wiki/FBX
https://github.com/nytimes/rd-blender-docker
Please note that we have made sure that the LiDAR settings matched the ones from NuScenes .

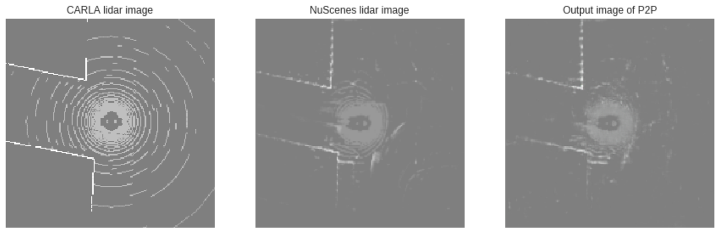


**Figure 3:** (a) `LiDAR` from `CARLA` (b) corresponding `LiDAR` from `NuScenes` (c) Results of applying P2P on `LiDAR` images from the `CARLA` images

as the difference of the distributions between $p_\theta(s|\phi$, for different $\phi$ settings, measured with *Kullback-Leibler* (KL) divergence. Specifically we compare the following `LiDAR` settings: `NuScenes` (NS), `CARLA` (CARLA), corrupted `NuScenes` `LiDAR` with *Bernoulli* noise of probability $\alpha$ (NS-$\alpha$) and finally blank `LiDAR` (blank). We summarize the comparisons $\mathrm{KL}_{\texttt{NS\_}\alpha\texttt{-NS}}$, $\mathrm{KL}_{\texttt{NS\_CARLA}}$, $\mathrm{KL}_{\texttt{NS\_blank}}$ on Table 2.

| **Experiment** | $\mathrm{KL}_{\texttt{NS\_}\alpha\texttt{-NS}}$, $\alpha = 0.2$ | $\mathrm{KL}_{\texttt{NS\_CARLA}}$ | $\mathrm{KL}_{\texttt{NS\_blank}}$ |
|---|---|---|---|
| **Mean** | 0.38334 | 0.38994 | 0.41349 |

**Table 2:** Table showing the average of KL divergences between using `LiDAR` images from the `NuScenes` dataset, and using `LiDAR` images from the `CARLA` simulator and lidar images after applying various values of $\alpha$ zeroing on the nuScenes dataset.

We found the distribution induced when `LiDAR` is fed from `NuScenes` , has the same KL divergence when we corrupt `NuScenes` with Bernoulli noise of $p = 0.2$ and when we replace `NuScenes` with `CARLA` dataset.

As summarized in table 1, we can see that the downstream task, although is affected when swapping `NuScenes` `LiDAR` with the one from `CARLA` , it's less that the performance evaluated with empty `LiDAR` (blank `LiDAR` case), suggesting the our generated towns maintain the fidelity required to solve the task.

### 4.2 Realistic Noise Augmentation via Conditional Adversarial Networks

Although the `LiDAR` in `CARLA` captures the high level geometry (obstacles) correctly, in the real world there is noise that is not modeled in the `LiDAR` sensor of `CARLA` , making the received `LiDAR` images look unrealistic. We cast the problem of learning a noise model into an Image to Image translation. We used Pix2Pix [Isola et al., 2017], a GAN based model, to learn the conditional noise model which make the `LiDAR` look closer to `NuScenes` . As we see in Figure 3, the image after the applied transformation look closer to the original `NuScenes` `LiDAR` . Although this is a promising direction, extra care need to be taken as the learned noise can introduce artifacts that can increase the uncertainty of the model.

## 5 Discussion And Future Work

With this work we demonstrated a pipeline for automatic generation of towns for autonomous driving from real maps. The generated maps share similar `LiDAR` footprint with real-world gathered data and therefore can allow for scaled-up benchmarks or data gathering of in-distribution and out-of-distribution scenarios. While this is a promising first step of brining real-world maps in simulated environments, we are currently working in bring more detailed geometry and texture generation to allow also camera based perception systems as well.

An open challenge remains to generate diverse towns with various difficulties. We aim to extend this work equipping it with an active process which will use the epistemic uncertainty of the trained models to select map locations that will improve the policy.

Finally, we aim to make the tools and source code available online for the research community to use and build upon.

## References


H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. 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.

DARPA. The grand challenge. URL `https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles`.

A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017.

A. Filos, P. Tigas, R. McAllister, N. Rhinehart, S. Levine, and Y. Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? *arXiv preprint arXiv:2006.14911*, 2020.

M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7 (4):12–18, 2008.

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*, pages 1125–1134, 2017.

N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*, pages 704–720. Springer, 1995.

N. Rhinehart, K. M. Kitani, and P. Vernaza. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 772–788, 2018.

P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.