A Framework for Generating Dangerous Scenes for Testing Robustness

Anonymous Author(s)
Affiliation
Address
email

Abstract

Benchmark datasets for autonomous driving, such as KITTI, nuScenes, Argoverse, or Waymo are realistic but designed to be faultless. These datasets contain limited errors, difficult driving maneuvers, or other corner cases. We propose a framework for perturbing autonomous vehicle datasets, the DANGER framework, which generates edge-case images on top of current autonomous driving datasets. The input to DANGER are photorealistic datasets from real driving scenarios. We present the DANGER algorithm for vehicle position manipulation and the interface towards the renderer module, and present five scenario-level dangerous primitives generation applied to the virtual KITTI and virtual KITTI 2 datasets. Our experiments prove that DANGER can be used as a framework for expanding the current datasets to cover generative while realistic and anomalous corner cases.

1 Introduction

Since the release of the KITTI [Geiger et al., 2012] dataset, autonomous driving research has been data-driven. Autonomous vehicles (AVs) promise to decrease vehicle fatalities and increase safety in the modern automobile. However, the majority of datasets [Chang et al., 2019, Wilson et al., 2021, Sun et al., 2020, Caesar et al., 2020, Geiger et al., 2012] and derived algorithms [Cheng et al., 2020, Xu et al., 2022, Girshick, 2015, Ren et al., 2015, Reading et al., 2021, Ganeshan et al., 2021, Cai et al., 2021, Weng et al., 2020, Zhou et al., 2020] are used for benchmarking and standardized on perfectly curated datasets, This causes two issues: (a) these algorithms are specially designed and hard-code into a workable scenario (b) solutions are focused on accuracy on a single dataset, rather than robustness. Instead, we want autonomous driving solutions to be able to deal with different driving scenarios. A real road scene is often dangerous and full of all kinds of unexpected events. There is an immediate need for AI systems to consider these event, especially in the context of implementing algorithms on actual on-road vehicles [Shalev-Shwartz et al., 2017]. Instead, we propose the development of a framework to mimic the kinds of "one-off" scenarios humans may encounter in driving tests. We use this to validate that AI systems can generalize in real-world driving environments. We describe an iterative procedure for creating out-of-domain examples for autonomous driving, or corner cases, based on the existing AV standard datasets.

It is challenging to design benchmarks and datasets. Therefore, we contribute a general framework for generating photo-level realistic driving scenes with custom trajectory inputs. Our framework, "DANGER", supports user-defined vehicle trajectories and poses to complete a sequence of frames of data generation. For example, a user can define a left-turn maneuver, which can be included in the data set. DANGER also supports the distortion and deletion of vehicles in an individual frame and can simulate illogical special camera failure modes. In summary, our key contributions are:

- We introduce DANGER, a framework for the generation of Danger-Aware datasets. DANGER is a generator that can enhance robustness. It generates new scenarios with user input:

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

a set of primitives. Each primitive is a vehicle driving trajectory and posture over time: a complete sequence of frames of data generation.

- DANGER supports the shifting and deletion of cars in individual frame and can simulate illogical special camera failure modes.
- Our DANGER implementation includes five scenario-level dangerous primitives applied on virtual KITTI and virtual KITTI 2 to generate more robust, "DANGER-vKITTI" datasets.
- We evaluate DANGER on a corner case score evaluation and via human study. Our results demonstrate that DANGER and our generated scenarios are realistic, novel, anomalous, or risky. These dataset augmentations can help increase the robustness and range of scenarios in the original datasets.

To prove that DANGER that can generate anomalous driving scenarios, we implemented quantitative and qualitative experiments on two datasets: Virtual KITTI [Gaidon et al., 2016b] and Virtual KITTI 2 [Cabon et al., 2020]. Our code for the benchmarks and results of experiments have been open-sourced under the MIT License. It is available at: `https://github.com/link` (Link is hidden for double-blind review).

2 Related Work

Today's deep neural networks are highly dependent on goodness-of-data and obey the "garbage in, garbage out" rule of thumb [Vidgen and Derczynski, 2020, Kim et al., 2016]. As Sambasivan et al. [2021] suggested, there is no goodness-of-fit without goodness-of-data. In the absence of standardized metrics to characterize "the goodness" of data, conventional visual perception methods are often not able to detect dangerous scenarios. This is because corner cases have not been witnessed during training [Thomas and Uminsky, 2020, Breitenstein et al., 2021]. However, fitting metrics does not represent the phenomenological fidelity and validity of the data. Their detection is based on sanitized data, lacking anomalous events: low possibility but realistic dangerous driving scenarios. Therefore, we need reliable detection and understanding of corner cases, which will further increase safety in autonomous driving.

2.1 Autonomous Driving Datasets

KITTI [Geiger et al., 2012] is the pioneering benchmark datasets for use in autonomous driving by providing LiDAR sensors, stereo cameras, and GPS/IMU data. Compared with KITTI, Waymo [Sun et al., 2020] provides a large-scale, high-quality dataset with high-intensity annotations and higher annotation frequency using five cameras and five LiDARs from different angles and locations. nuScenes [Caesar et al., 2020] provides 360°coverage from the LiDAR, radar, and camera sensors. Argoverse [Chang et al., 2019] contributes detailed geometric and semantic maps of the environment, and its sibling Argoverse2 [Wilson et al., 2021] has the most extensive self-driving taxonomy with HD maps that include real-world changes. Cityscapes [Cordts et al., 2016] provides semantic, instance-wise annotations for semantic understanding of urban street scenes. The current state-of-the-art (SOTA) datasets are inherently designed for independent model training based on various sensors rather than handling real-world challenges. As Shalev-Shwartz et al. [2017] estimated that AVs need to drive 30 billion miles to get enough statistical evidence to prove that AVs are three times safer than human drivers, yet leading Waymo claims their test fleet has run 20+ million miles on public roads [Schwall et al., 2020]. Therefore, corner case or out-of-distribution data in existing autonomous driving datasets are insufficient.

2.2 Synthetic Datasets

Advances in computer graphics have made it possible to easily annotate and generate virtual datasets, such as SYNTHIA, Virtual KITTI (vKITTI), and Virtual KITTI 2 (vKITTI2), include various scene types under different weather, environment, and lighting conditions [Ros et al., 2016, Gaidon et al., 2016b, Cabon et al., 2020]. Johnson-Roberson et al. [2016] demonstrate that SOTA neural networks trained using only synthetic data perform better than the same architectures trained on real-world dataset. CARLA [Dosovitskiy et al., 2017] supports custom waypoints input for vehicle trajectory generation, but it is projected in a virtual city built in a game engine without any modification compatibility to an existing real-world dataset.

2.3 2D Image Synthesis

Generative Adversarial Networks (GANs) [Goodfellow et al., 2014], have been used for a variety of image synthesis exercises, including image generation [Radford et al., 2015, Arjovsky et al., 2017, Karras et al., 2017], image-to-image translation [Isola et al., 2017, Zhu et al., 2017], text-to-image synthesis [Zhang et al., 2017, Reed et al., 2016], and inpainting [Pathak et al., 2016]. StyleGAN [Karras et al., 2019a] generates high-quality, high-resolution face images. StyleGan can also generate car-like images; however, flaws remain in the generated dataset. Firstly, the generated images were frequently displayed at a 45-degree exhibition angle rather than the perspective view of a car in motion. Second, the shape of some of the cars was distorted and produced a peculiar effect of indistinguishable front and rear of the car. StyleGAN2 [Karras et al., 2019b] fixed the artifacts problem remained in StyleGAN. However, both StyleGAN and StyleGAN2 cannot generate images via controlling car object-dependent appearance, pose, and size in 3D.

Recent studies [Ramesh et al., 2021, 2022, Saharia et al., 2022] showed the SOTA GPT-based methods of generating photorealistic images from text. Nevertheless, the above methods generate images that have low fidelity and cannot generate continuous frame images or videos.

Several studies [Ratner et al., 2017, Cubuk et al., 2019] learned transformation and color adjustment policies, such as rotate, shear, shear, contrast, or, hue. These strategies, however, are limited to 2D images or color space and are challenging to generate new viewpoint data. Contrary to 2D, we need a disentangled 3D-aware image synthesis model that allows a user to edit the viewpoint, object shape, or texture independently [Zhu et al., 2018].

2.4 3D-aware Image Synthesis

Several recent studies [Yao et al., 2018, Schwarz et al., 2020, Liao et al., 2020] developed multiple GAN-based 3D-aware image synthesis methods including 3D-friendly features: interpretable, disentangled scene representation, viewpoint manipulation, and 3D Controllable. Wu et al. [2017], Liao et al. [2020], and Greff et al. [2022] inspired by rendering a 2D image by 3D game engine, which treats images as a projection of the 3D world. The learned 3D space can be potentially useful for various tasks such as image reasoning tasks in the dangerous scene understanding.

Inspired by Mildenhall et al. [2020], GRAF [Schwarz et al., 2020] introduces GANs to implement Neural Radiance Fields, and uses conditional GAN [Mirza and Osindero, 2014] to achieve controllability of the rendered object. GIRAFFE [Niemeyer and Geiger, 2021] uses one Neural Radiance Field per object in order to combine objects from different scenes. This enables the movement and rotation of objects in the generative new image. GRAF and Liao et al. [2020] can generate images of cars in different poses, but the resulting backgrounds are often monochromatic or pure white. 3D-SDN [Yao et al., 2018] and PNF [Kundu et al., 2022], are the algorithm that can modify both the 3D pose and position of the spawning vehicles while remaining realistic city and road scenes.

2.5 Robustness

Our framework is a hybrid knowledge representation layer that improves robustness by introducing out-of-domain semantically-significant samples. Our work is similar to a knowledge-driven cognitive model approach improve AI system robustness [Marcus, 2020], with a focus on robustifying autonomous vehicle data sets with predefined primitives. We argue that using a primitive representation, similar to the abstract script-like representation in language [Schank, 1975, Borchardt, 1992], can make the underlying opaque system more understandable. Amini et al. [2022] demonstrates that VISTA simulator can generate novel camera images. However, the generated photo-realistic pictures can only be obtained via changing the driving angle of the ego vehicle but not the pose of other cars on the road.

Our work is inspired by prior work on a stress testing framework for autonomous system verification and validation [Falco and Gilpin, 2021]. The key idea for DANGER is that the robustness is not static, but a series of stress tests that can be automatically generated. Our goal is to use these generative examples to generate counterfactuals, similar to previous work using adversarial examples for explaining counterfactuals [Pawelczyk et al., 2022]. Our contribution is a set of semantic primitives, similar to the semantic layer in DNNs discussed in [Browne and Swift, 2020].

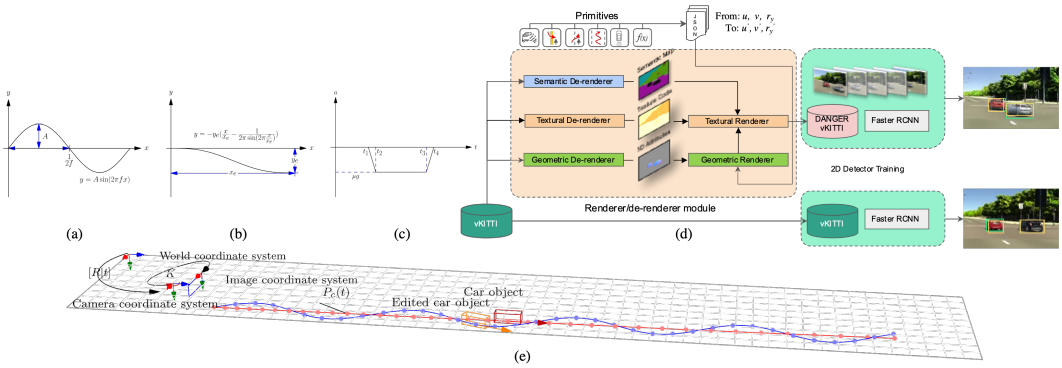


Figure 1: **DANGER framework, primitives, and visualization of scene editing**. (a) Primitive slalom lane change function (b) cut-in function (c) braking function (d) DANGER contains a renderer/de-renderer module, primitive function module, and generated descriptive file that can help users to develop a wide variety of corner-cases (e) a slalom-lane-change scene editing of scene `0006` shown in world coordinate. The orange red curve is the original trajectory of car object `tid` 4, and the light blue curve is the corresponding edited trajectory in world coordinate.

According to the systematization of corner case detection complexity proposed by Breitenstein et al. [2021], our framework is designed to obtain the most dangerous anomalous scenario for a detector, and we also answer the research question, '*How to generate or record corner case from descriptions?*', proposed by Bogdoll et al. [2021]. Breitenstein et al. [2021] defined a corner case as a non-predictable relevant object/class in a relevant location, Breitenstein et al. [2021] extended the definition of corner case level by giving three most dangerous scenes: the anomalous scenario is a potentially dangerous unknown object, the novel scenario is a harmless unknown object, the risky scenario is a potentially dangerous but known object. In the result section, we present the design our primitives.

3 Method

Renderer/de-renderer module The 3D scene de-rendering networks (3D-SDN) [Yao et al., 2018] is an optimal algorithm that generates photo-level realistic synthetic images. It employs an encoder-decoder architecture with three branches: scene semantics, object geometry, 3D pose, and textual appearance of objects and the background. As shown in Figure 1 (d), three branches intend to learn a scene's semantic segmentation, infer the object shape and 3D pose, and encode the appearance of each object and background segment. Disentangling 3D geometry and pose from the given scene enables 3D-aware scene manipulation in image coordinate with a target object location (u, v), orientation pose r_y, and `(delete, modify)` operations described in a JSON file.

In the result section, we use 3D-SDN as the renderer/de-renderer module to demonstrate the feasibility of our framework. In practice, modules such as PNF can also be selected [Kundu et al., 2022].

Primitives To augment the input dataset, we define five danger-aware primitives: `Exit parking`, `Cut-in Opposite`, `Cut-in`, `Slalom Lane Change`, and `Braking`. Detail of these primitives will be introduced in the results section.

3.1 Scene Editing Computation

Our approach is adaptive. The object's position can be edited in the real-world 2D plane according to any arbitrary function. At the same time, the implementation of editing in the world coordinate system requires reading the position information of the object and transforming it into the world coordinate system. The vKITTI dataset was designed to match the multi-object tracking (MOT) evaluation benchmark of KITTI. Therefore, the MOT ground truth and exact position of each car object are provided in the folder `motgt`. The following object annotation and terminology we inherited from vKITTI is detailed in Gaidon et al. [2016b,a] and supplementary material.

We assume that the original driving trajectory of the target vehicle object is a straight line. Given a image $\mathcal{I} \in \mathbb{R}^{W \times H \times 3}$ defined by `scene`, `topic`, `tid`[1], and `frame` with known camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times4}$ and camera extrinsic matrix $[\mathbf{R}|\mathbf{t}]$, we can convert the position of an object in world coordinate, camera coordinate, or pixel coordinate by Equation (1) and Equation (3), where $\mathbf{R} \in \mathbb{R}^{3\times3}$ and $\mathbf{t} \in \mathbb{R}^{3\times1}$ indicate the rotation and translation matrices [Hartley and Zisserman, 2003]. $P_c \in \mathbb{R}^{4\times1}$ is the 3D point position in camera coordinates and $P_w \in \mathbb{R}^{4\times1}$ is the 3D point position in world coordinates. In both cases, they are represented in homogeneous coordinates. A camera extrinsic matrix $\mathbf{M} \in \mathbb{R}^{4\times4}$ is used to denote a projective mapping from world coordinates to pixel coordinates.

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{1}$$

$$\mathbf{P} = \mathbf{K}\mathbf{M} \tag{2}$$

$$P_c = \mathbf{M}P_w \tag{3}$$

The elements of object's center position vector P_c can be acquired from the MOT ground truth data `x3d`, `y3d`, `z3d`, and `h3d`, and the corresponding P_w will be easily obtained by apply the inverse of frame-dependent matrix $\mathbf{M}$. In the x-z plane, arbitrary vehicle poses can be generated according to the primitive function, where $\mathbf{r}'_y$ is a unit tangent vector to the curve at (x'_w, z'_w) representing the target orientation of the car object. The trajectory curve and heading vectors are generated at the origin and then translated to the desired starting point. The curve was further rotated by θ using the rotation matrix $\mathcal{R}_\theta$ to align with the original trajectory's slope a, where a, θ are the slope and angle of original path, $\mathcal{R}$ is a corresponding rotation matrix [Weisstein, 2003]. A detailed algorithm of the implementation of the primitive operation in 3D world coordinate is shown in the supplementary material.

4 Results

In this section, we present the *base* dataset and primitives we used in our generation framework. We evaluate our results in two aspects that focus on the corner cases generation capability and the realistic level of our dangerous maneuver via a human study.

We hypothesize that adding risky maneuvers to the dataset will result in a higher corner case score for the ego vehicle. We performed a numerical analysis of our generated data frames by applying a corner case detector that considers object-level and predictability. We also validated our results with a user study.

4.1 Dangerous Corner Case Generation

4.1.1 Primitives

We designed five scenario-level dangerous corner cases in the world space based on the vKITTI dataset. These primitives are deliberately selected for each scene presented in the vKITTI according to the vehicle object's position and motion. Though these are hand-curated, each primitive follows the definition of a scenario-level corner case that is an anomalous or risky scenario derived from the real-world. We also refer to the safety assist test procedure and autonomous vehicle collision report as templates for our dangerous corner cases [Van Ratingen, 2017, California DMV, 2022e].

Cut-in Many accidents are caused by neighboring vehicles suddenly driving in front of a moving car dangerously, either due to the driver impatiently overtaking or an unintentional aggressively traverse due to forgetting the highway exits. To achieve a realistic cut-in lane change, we define the single-lane change curvature according to the sinusoidal ramp function [Sledge Jr and Marshek, 1997] as following:

[1] A unique track identification number for each object instance in the scene.

$$y = y_e(\frac{x}{x_e} - \frac{1}{2\pi \sin(2\pi \frac{x}{x_e})}), \tag{4}$$

where x_e is the longitudinal offset of the target position, y_e is the lateral lane-change offset of the target position as shown in Figure 1 (b). The forwarding trajectory will be rotated and aligned with the camera space's longitudinal axis z_c. We chose two vehicles with `tid` 1 and 2 in scene `0018` to simulate two scenarios of cut-in ego vehicle and overtaking with nine sets of parameters, respectively.

Exit Parking A careless driver may suddenly place its front end out of a line of parked cars on either side of a narrow street. We assign a trajectory generated by the cut-in function presented above to two distinguished car objects with `tid` 63 and 70 in scene `0001`. Among the nine sets of parameters for each vehicle, x_e was chosen as a distance of 4 m for one and a half vehicle lengths.

Cut-in on Opposite We incorporate the rotated cut-in function to a car (`tid` 0) driving in the opposite direction. The rampage driver's driving leads to an upcoming accident that potentially causes severe injury in scene `0002`. By combining six x_e and three y_e parameters, we obtained eighteen sets of different driving conditions distributed in a two-dimensional space of ninety square meters.

Slalom Lane Change According to Wang [1994]'s study, lane change crashes caused over 244,000 accidents in 1991, accounting for 4.0% of all accidents in the US. Therefore, we aim to design a novel scenario that a driver can barely decide whether other car objects choose to make a lane change or not. As illustrated in Figure 1 (a), we borrow the idea from the slalom test in automotive engineering and assign a parameter-dependent sine-wave to the `tid` 2 and 7 in scene `0006` as follows:

$$y = Asin(2\pi f x) \tag{5}$$

where A is the lateral offset amplitude in meters, and f is the steering input frequency in Hz.

Braking We simulate a constant deceleration maneuver in scene `0020` when `tid` 16 applies a braking pedal by exploiting the inverted trapezoid piecewise function:

$$a_t = \begin{cases} 0, & t < t_1 \\ -\frac{(t-t_1)\mu g}{t_2 - t_1}, & t_1 \leq t < t_2 \\ -\mu g, & t_2 \leq t < t_3 \\ -\frac{(t_4 - t)\mu g}{t_4 - t_3}, & t_3 \leq t < t_4 \\ 0, & t_4 \leq t \end{cases} \tag{6}$$

where a_t is the time dependent acceleration of the editing vehicle, μ and g constitute a uniform deceleration value, t_i is the timestamp when the braking caliper initiated and stopped. As illustrated in Figure 1 (c), we replace the object speed in the original video frame with the target braking speed curve obtained by filtering and integrating the generated acceleration curve. Then a new waypoint can be generated in world coordinates to replace the original.

For each primitive, we also calculate the rotation Δr_y, zoomed-in factor ρ, and the transformed (u, v) pixels in camera space. All operation parameters are packaged into a JSON file for 3D-SDN processing. For the detailed JSON and parameter settings, see our supplementary material.

4.1.2 Datasets

vKITTI and vKITTI2 [Gaidon et al., 2016b, Cabon et al., 2020] contain five scenes of photo-realistic videos synthesized from virtual worlds under various lighting and weather conditions, respectively. vKITTI2 has more photo-realistic images by utilizing the updated Unity game engine enhancements. In our experiments, we downscale vKITTI images to 624×192. Our inference model was inherited from Yao et al. [2018], and we also trained models followed the setup of Yao et al. [2018] for vKITTI2 on an NVIDIA Tesla P100 (16G) GPU.

Our DANGER-vKITTI dataset contains 4527 pictures with consecutive frames for each of the five scenes with 18 different sets of parameters. Compared to the original five scenes from vKITTI, DANGER has expanded the original dataset into 90 additional scenes using the experimental parameters.

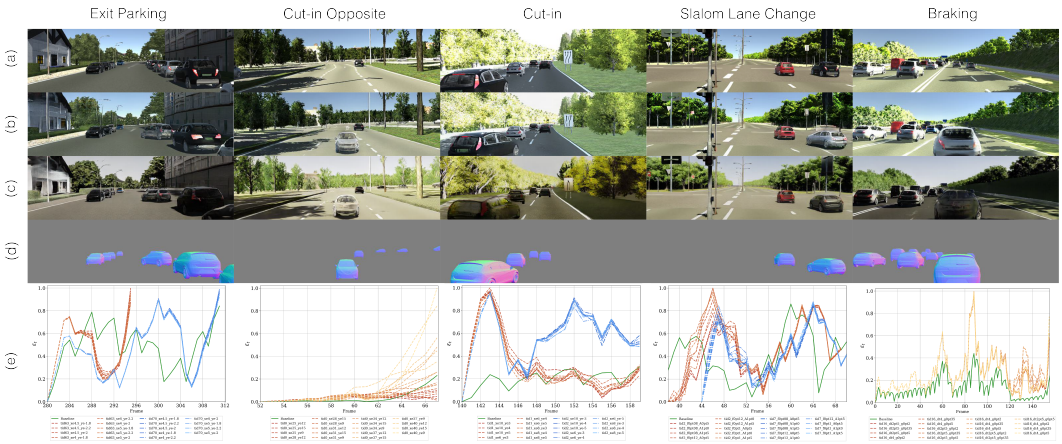


Figure 2: **Primitive dangerous scenarios and evaluation results**. Generation result for each primitive in our study from left to right: `Exit parking` (frame301), `Cut-in Opposite` (frame67), `Cut-in` (frame144), `Lane Change` (frame52), and `Braking` (frame129). The (a) vKITTI, (b) DANGER-vKITTI, (c) DANGER-vKITTI2, (d) normal map, and (e) a normalized dangerous corner scores evaluation of the generated DANGER datasets (two `tid`s in red and blue color cluster, baseline vKITTI in green) are shown from top to bottom.

No.	Metrics	Datasets	Exit Parking	Cut-in Opposite	Cut-in	Slalom Lane Change	Braking
(1)	$\bar{\epsilon}$	vKITTI	0.48	0.04	0.19	0.41	0.13
(2)	ϵ^*	vKITTI	0.84	0.23	0.31	0.86	0.44
(3)	avg. of $\bar{\epsilon}_{\mu\in\mathcal{D}}$	ours	0.46	0.07	0.44	0.45	0.26
(3)/(1)	Ratio		0.97	**1.97**	**2.27**	**1.09**	**1.99**
(4)	avg. of $\epsilon^*_{\mu\in\mathcal{D}}$	ours	0.95	0.29	0.96	0.89	0.98
(4)/(2)	Ratio		**1.13**	**1.26**	**3.15**	**1.03**	**2.23**
(5)	$\epsilon^*_{\mu\in\mathcal{D}}$	ours	1.00	1.00	1.00	1.00	1.00
(5)/(2)	Ratio		**1.19**	**4.33**	**3.27**	**1.16**	**1.21**

Table 1: **Quantitative evaluation of DANGER-vKITTI with the baseline dataset vKITTI**. We compared the mean $\bar{\epsilon}$ and max value ϵ^* of the corner score for each parameter setting in each scene by its average number and maximum number.

4.2 Quantitative Evaluation

We use the conceptual definition of the corner case Bolte et al. [2019] proposed that a *non-predictable relevant object/class* in *relevant location*, and we hypothesize a dangerous movement is a subset of corner cases. Our corner case detector considers semantic segmentation and image non-predictability.

First, we adopt a segmentation network based on the `MobileNet-V2` [Sandler et al., 2018] that allows us to classify and localize the *objects* in the scene for which moving objects are considered as *relevant*. Specifically, we use the model pretrained on ImageNet, MS-COCO, and Cityscapes `train_fine` set provided in `DeepLabv3` [Deng et al., 2009, Lin et al., 2014, Cordts et al., 2016, Chen et al., 2017]. Second, we used the advantage of `PredNet` [Lotter et al., 2016] that can sensing the moving objects to compare the deviation between predicted frame $\hat{\mathbf{x}}_t$ and the real next frame $\mathbf{x}_t$ for time t. As a metric for the corner case, we calculate an error e_t to represent the *non-predictable* component of the detection system.

$$\mathbf{e}_t = \hat{\mathbf{x}}_t - \mathbf{x}_t, \tag{7}$$

where elements $e_t(i), i \in \mathcal{I}$. Here, each pixel $i \in \mathcal{I}$, where $\mathcal{I}$ denotes the set of pixel indices in the given image, and $\|\mathcal{I}\| = H \cdot W$ represents the number of pixels. The input image $\mathbf{x}_t \in \mathbb{G}^{H\times W\times C}$ with image pixel $x_t(i) \in \mathbb{G}$, where $\mathbb{G} = \{0, 1, \ldots, 255\}$ is the set of gray values, H and W are

the image height and width in pixels and $\mathbb{C} = \{1, 2, 3\}$ is the number of color channels. The segmentation network maps the input to output scores $\mathbf{P}_t \in \mathbb{I}^{H \times W \times \|\mathcal{S}\|}$, where $\mathcal{S}$ denotes the set of classes with cardinality $\|\mathcal{S}\| = 19$ and $\mathbb{I} = [0, 1]$. Taking the argmax over the output scores we obtain the $(H \times W)$-dimensional mask $\mathbf{m}_t = \arg\max_{s \in \mathcal{S}} \mathbf{P}_t$, which gives us a pixel-wise classification $m_t(i) \in \mathcal{S}$. Within this work, we limit the target objects $s \in \mathcal{S}_{rel} = \{12, 13, \ldots, 19\}$, which represents the *relevant classes* such as `Person`, `Car`, `Truck`, etc [Cordts et al., 2016].

The information from the two preceding processing steps is combined as a detection system as metric ϵ to evaluate our DANGER datasets. Since we only interested in the target classes defined in $\mathcal{S}_{rel}$, we filter the error map Equation (7) by the following formula:

$$e_{t,rel}(i) = \begin{cases} e_t(i), & m_t(i) \in \mathcal{S}_{rel} \\ 0, & m_t(i) \notin \mathcal{S}_{rel} \end{cases} \tag{8}$$

To assign higher dangerous weighting for the objects close to the ego vehicle, we employ the weighted squared errors of the relevant classes introduced by Bolte et al. [2019] as follows:

$$\epsilon' = \sum_{i \in \mathcal{I}} e^2_{t,rel}(i) \cdot (1 - \frac{h_i}{H-1}), \tag{9}$$

with $h_i \in \{0, 1, \ldots, H-1\}$ being the row index from bottom-up. In contrast to Bolte et al. [2019], we normalize the error scores ϵ'_t of mth dangerous parameter in set $\mathcal{D} = \{1, 2, \ldots, 18\}$ for the same scene to ensure the peak dangerous frame $\mathbf{x}_t$ of the system generates a corner case score $\epsilon_t = 1$. The corner case score is obtained by normalizing the error score ϵ'_t to a value range from 0 to 1 using:

$$\epsilon_t = \frac{\epsilon'_t - \min_{\tau \in \mathcal{T}, \mu \in \mathcal{D}} \epsilon'_{\tau,\mu}}{\max_{\tau \in \mathcal{T}, \mu \in \mathcal{D}} \epsilon'_{\tau,\mu} - \min_{\tau \in \mathcal{T}, \mu \in \mathcal{D}} \epsilon'_{\tau,\mu}} \tag{10}$$

where $\mathcal{T}$ denotes a set of timestamps, and μ denotes the parameter set in $\mathcal{D}$.

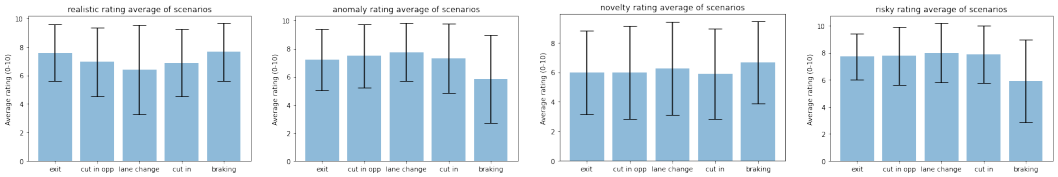
Compared to the green baseline datasets, Figure 2 (e) shows that the corner score increases regardless of which `tid` object is modified. The highest score for `Exit parking` is in the last frame (296 and 312) when the car is completely exiting the side parking; the highest score for `Cut-in Opposite` is in frame 67 when the vehicle is about to crash into the ego; the dangerous score of the `Cut-in` is highest in frame 143 when the black car starts to leave the lane and in frame 152 when the gray car is heading to the opposite lane; the dangerous score is higher in the `Slalom Lane Change` between frames 43 and 52 when the twisting vehicle is relatively close; in the `Braking`, the dangerous score is relatively high in the last frame, 158, and the peak in the middle may be caused by the sudden disappearance of the vehicle object in the frame due to the instability of the 3D-SDN model. Table 1 shows that DANGER's corner case level is between 1 and 3.15 times that of the original datasets considering the average performance across parameters. Considering only the individual parameters, DANGER's hazard factor can be up to 4.33 times the original data.

4.3 User Study

We designed a user study to validate our synthetically generated data samples, e.g., scenarios. Our hypothesis was that users would find the scenarios realistic and the level of "dangerousness" would correlate with Table 2 and Figure 2 (e). We recruited 100 subjects, who were described the domain and problem setup: that we have augmented a data set to create new scenarios that they are to rank. Users were also presented with the definitions of novel, anomalous, risky and unknown/known events, which are consistent with Breitenstein et al. [2021]. Subjects were recruited via Amazon Mechanical Turk and compensated for completing the survey.

Users were presented with a driving scenario and answered four questions for each scenario: to rank whether the scenario was realistic, novel, anomalous and risky. Users were presented a Likert scale from 0-10, with 10 indicating very realistic/novel/anomalous/risk, and 0 being not realistic/novel-/anomalous/risk at all. We presented a random sample of scenarios, one per primitive. The questions were presented in a random order. The results are in Figure 3.

We found that users generally agreed with our classifications. Users found our scenarios to be realistic, rating them with average scores in the 7-8 range. Users found the cut-in opposite and lane change scenarios to be the most anomalous. Users found the lane change and braking scenarios to be the



(a) Realistic rating (b) Anomaly rating (c) Novelty rating (d) Risky rating

Figure 3: Aggregated user study results for rating our scenarios as realistic, anomalous, novel and risky. Averages are plotted as the bars, and error bars show the standard deviation.

Primitive	Level	Testimony
Exit Parking	Novel	*'...a parked pickup ...pulled out ...and made contact with ...the Waymo AV...'* [California DMV, 2022c]
Cut-in Opposite	Anomalous	*'...a sedan was traveling the wrong way on a one-way section ...made contact with the Cruise AV...'* [California DMV, 2022b]
Lane Change	Novel	N/A
Cut-in	Risky	*'...a passenger vehicle in the right adjacent lane abruptly cut in front of the Zoox vehicle...'* [California DMV, 2022d]
Braking	Risky	*'...The passenger vehicle ahead ...transitioned into reverse and began to accelerate, making contact with the front sensor of the Waymo AV...'* [California DMV, 2022a]

Table 2: We classify our primitives under the taxonomy of corner cases defined by Breitenstein et al. [2021]. We also provide descriptive prompts derived from Autonomous Vehicle Collision Reports in 2022 for each primitive [California DMV, 2022e].

most novel. Users found most of the scenarios to be risky. Notably, they did not find the braking maneuvers to be very risky. This might be because the braking scenarios are almost 1 minute in length, and perhaps users did not watch, and therefore analyze, the full scenario.

5 Conclusion and Discussion

In this paper, we present DANGER: a framework for generating anomalous driving scenarios from existing self-driving vehicle datasets. We show how to define a set of primitives which align with corner cases [Brazil et al., 2020] and user feedback. We promote the use of DANGER to robustify existing autonomous vehicle datasets that may not contain error cases. In fact, it might be implausible to collect such corner cases, either because of the intractability of the scenario, or because of the ethical consequences. We present a framework and technique to define, extract, and classify these scenarios *from existing data*.

Our work has limitations. The 3D-SDN model may generate a jitter frame due to the geometry estimation failure, this might be improved by choosing a more robust renderer/de-renderer module. Similarly, we sometimes get a missing car in a single frame, which may lead to errors in the calculation of the dangerous score. Currently, we can only complete modifications with 3 degrees of freedom. These limitations highlight the challenge of generating corner cases.

We show that "dangerousness" is a superclass consisting of novel, anomalous and risky behaviors and actions. Our initial set of primitives are intentionally distributed over anomalous, novel, and risky scenarios. Users generally agree with labels and categorizations. While we demonstrated DANGER on vKITTI, it can be used with any autonomous vehicle dataset that supports semantic and 3D annotations with a given renderer.

In summary, our DANGER framework is a robustness generator for self-driving car datasets. It is adaptable to multiple types of primitives, and it can cover a wide range of dangerous levels: novel, risky, and anomalous. Our work has opened a new area of robustness data generation, where users,

stakeholders, and system designers can identify and easily generate corner cases to augment datasets in order to make them more robust.

References

Alexander Amini, Tsun-Hsuan Wang, Igor Gilitschenski, Wilko Schwarting, Zhijian Liu, Song Han, Sertac Karaman, and Daniela Rus. Vista 2.0: An open, data-driven simulator for multimodal sensing and policy learning for autonomous vehicles. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2419–2426. IEEE, 2022.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Daniel Bogdoll, Jasmin Breitenstein, Florian Heidecker, Maarten Bieshaar, Bernhard Sick, Tim Fingscheidt, and Marius Zöllner. Description of corner cases in automated driving: Goals and challenges. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1023–1028, 2021.

Jan-Aike Bolte, Andreas Bar, Daniel Lipinski, and Tim Fingscheidt. Towards corner case detection for autonomous driving. In *2019 IEEE Intelligent vehicles symposium (IV)*, pages 438–445. IEEE, 2019.

Gary C Borchardt. Understanding causal descriptions of physical systems. In *AAAI*, pages 2–8, 1992.

Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *European Conference on Computer Vision*, pages 135–152. Springer, 2020.

Jasmin Breitenstein, Jan-Aike Termöhlen, Daniel Lipinski, and Tim Fingscheidt. Corner cases for visual perception in automated driving: Some guidance on detection approaches. *arXiv preprint arXiv:2102.05897*, 2021.

Kieran Browne and Ben Swift. Semantics and explanation: why counterfactual explanations produce adversarial examples in deep neural networks. *arXiv preprint arXiv:2012.10076*, 2020.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

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.

Yingfeng Cai, Lei Dai, Hai Wang, and Zhixiong Li. Multi-target pan-class intrinsic relevance driven model for improving semantic segmentation in autonomous driving. *IEEE Transactions on Image Processing*, 30: 9069–9084, 2021.

California DMV. *Waymo June 5, 2022*. STATE OF CALIFORNIA, DEPARTMENT OF MOTOR VEHICLES, 2022a. URL `https://www.dmv.ca.gov/portal/file/waymo_060522-pdf/`.

California DMV. *Cruise April 25, 2022*. STATE OF CALIFORNIA, DEPARTMENT OF MOTOR VEHICLES, 2022b. URL `https://www.dmv.ca.gov/portal/file/cruise_ol316_042522-pdf/`.

California DMV. *Waymo April 29, 2022*. STATE OF CALIFORNIA, DEPARTMENT OF MOTOR VEHICLES, 2022c. URL `https://www.dmv.ca.gov/portal/file/waymo_ol316_042922-pdf/`.

California DMV. *Zoox July 8, 2022*. STATE OF CALIFORNIA, DEPARTMENT OF MOTOR VEHICLES, 2022d. URL `https://www.dmv.ca.gov/portal/file/zoox_070822-pdf/`.

California DMV. Autonomous vehicle collision reports, Aug 2022e. URL `https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/autonomous-vehicle-collision-reports/`.

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019.

Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Hongdong Li, Tom Drummond, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. *Advances in Neural Information Processing Systems*, 33:22158–22169, 2020.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

Gregory Falco and Leilani H Gilpin. A stress testing framework for autonomous system verification and validation (v&v). In *2021 IEEE International Conference on Autonomous Systems (ICAS)*, pages 1–5. IEEE, 2021.

Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual kitti dataset. `https://europe.naverlabs.com/research/computer-vision/proxy-virtual-worlds-vkitti-1/`, 2016a. (Accessed on 08/13/2022).

Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtualworlds as proxy for multi-object tracking analysis. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4340–4349, 2016b. doi: 10.1109/CVPR.2016.470.

Aditya Ganeshan, Alexis Vallet, Yasunori Kudo, Shin-ichi Maeda, Tommi Kerola, Rares Ambrus, Dennis Park, and Adrien Gaidon. Warp-refine propagation: Semi-supervised auto-labeling via cycle-consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15499–15509, 2021.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3749–3761, 2022.

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei 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.

Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*, 2016.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019a.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019b. URL https://arxiv.org/abs/1912.04958.

Yoonsang Kim, Jidong Huang, Sherry Emery, et al. Garbage in, garbage out: data collection, quality assessment and reporting standards for social media data use in health research, infodemiology and digital disease detection. *Journal of medical Internet research*, 18(2):e4738, 2016.

Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic Neural Fields: A Semantic Object-Aware Neural Scene Representation. In *CVPR*, 2022.

Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5871–5880, 2020.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

Gary Marcus. The next decade in ai: four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*, 2020.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

Martin Pawelczyk, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In *International Conference on Artificial Intelligence and Statistics*, pages 4574–4594. PMLR, 2022.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. URL https://arxiv.org/abs/2102.12092.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

Alexander J Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Learning to compose domain-specific transformations for data augmentation. *Advances in neural information processing systems*, 30, 2017.

Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8555–8564, 2021.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. URL https://arxiv.org/abs/2205.11487.

Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445518. URL https://doi.org/10.1145/3411764.3445518.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

Roger C Schank. The primitive acts of conceptual dependency. In *Theoretical issues in natural language processing*, 1975.

Matthew Schwall, Tom Daniel, Trent Victor, Francesca Favaro, and Henning Hohnhold. Waymo public road safety performance data, 2020. URL https://arxiv.org/abs/2011.00038.

Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *CoRR*, abs/1708.06374, 2017. URL http://arxiv.org/abs/1708.06374.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*, 2017.

Nathaniel H Sledge Jr and Kurt M Marshek. Comparison of ideal vehicle lane-change trajectories. *SAE transactions*, pages 2004–2027, 1997.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin 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.

Rachel Thomas and David Uminsky. The problem with metrics is a fundamental problem for ai. *arXiv preprint arXiv:2002.08512*, 2020.

Michiel R Van Ratingen. The euro ncap safety rating. In *Karosseriebautage Hamburg 2017*, pages 11–20. Springer, 2017.

Bertie Vidgen and Leon Derczynski. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *Plos one*, 15(12):e0243300, 2020.

J-S Wang. *Lane change/merge crashes: problem size assessment and statistical description*. US Department of Transportation, National Highway Traffic Safety Administration, 1994.

Eric W Weisstein. Rotation matrix. *https://mathworld. wolfram. com/*, 2003.

Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. Ab3dmot: A baseline for 3d multi-object tracking and new evaluation metrics. *arXiv preprint arXiv:2008.08063*, 2020.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021.

Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. Neural scene de-rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 699–707, 2017.

Gangwei Xu, Junda Cheng, Peng Guo, and Xin Yang. Acvnet: Attention concatenation volume for accurate and efficient stereo matching. *arXiv preprint arXiv:2203.02146*, 2022.

Shunyu Yao, Tzu Ming Harry Hsu, Jun-Yan Zhu, Jiajun Wu, Antonio Torralba, William T. Freeman, and Joshua B. Tenenbaum. 3d-aware scene manipulation via inverse graphics. In *Advances in Neural Information Processing Systems*, 2018.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017.

Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3d representations. *Advances in neural information processing systems*, 31, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See appendix.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [No]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Datasets

A.1.1 Visualization

Our dataset is hosted on the Google drive[2], where contains all the DANGER-vKITTI1 gifs generated by the dataset without the raw data for the parameter sets shown in Table 4. You can navigate to `./DANGER-vKITTI1/main.html` to check all gifs for different parameter sets. vKITTI2 only generates parameter sets in Table 3, but both gifs and images are available in the folder.

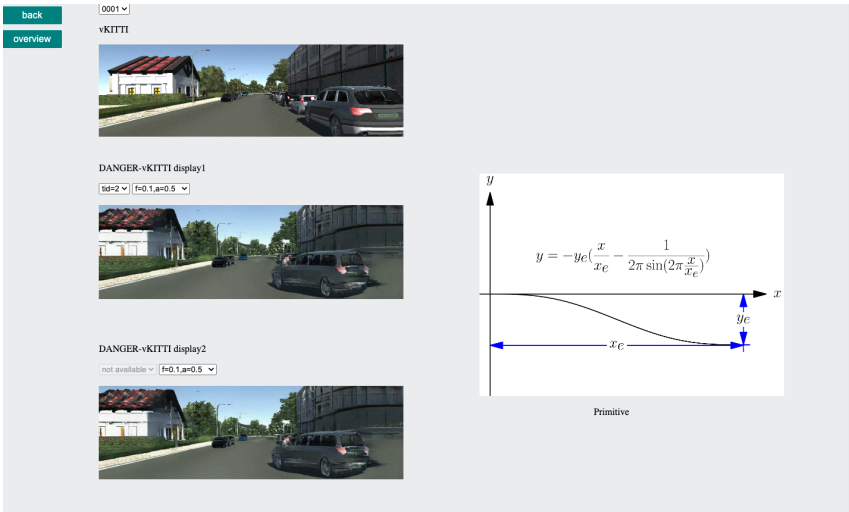


Figure 4: Visualizer for DANGER-vKITTI1

	Exit Parking	Cut-in Opposite	Slalom Lane Change	Cut-in	Braking
DANGER-vKITTI	tid70_epp_xe5_ye-2.2	tid0_cio_xe40_ye15	tid7_lc_f0pt12_A1pt5	tid2_ci_xe10_ye-5	tid16_br_t010_dt4_g0pt35
DANGER-vKITTI2	tid70_epp_xe4_ye-2	tid0_cio_xe37_ye12	tid7_lc_f0pt1_A1pt0	tid1_ci_xe10_ye3	tid16_br_t02_dt2pt5_g0pt5

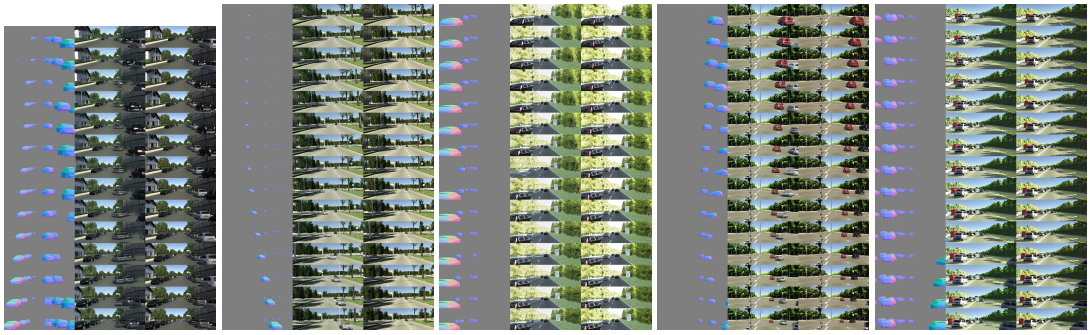
Table 3

A.1.2 Datasets Generation Code

Our Code is open sourced on the Colab server in Jupyter Notebook format. Here we provide the PDF version of the original notebook. The functionality of each file are as follows:

- `./DANGER_Design_JSON.pdf` Load metadata from vKITTI dataset, and create modified JSON files for 3D-SDN.

[2]https://drive.google.com/file/d/14ZjBi3ewvf2J92aKImEJgQ2j-ZzhqA_7/view?usp=sharing



(a) 0001 (b) 0002 (c) 0018 (d) 0006 (e) 0020

Figure 5: Overview of generated datasets.

Algorithm 1 Transformation algorithm used in cut-in.

Input: Input scene **S**, topic **T**, car object **tid**
Output: dataframe **df**

Let $t \in [t_s, t_e]$
$P_{c,t} \leftarrow \begin{bmatrix} x_{c,t} & y_{c,t} & z_{c,t} & 1 \end{bmatrix}^\intercal$
$P_{w,t} \leftarrow \mathbf{M}^{-1} P_{c,t}$
Apply linear regression on $P_{w,t}$ to get slope a
Generate a curve by *primitive function* at origin $P^*_{w,t} \leftarrow \begin{bmatrix} x'_w & y'_w & z'_w & 1 \end{bmatrix}^\intercal w/ \ \mathbf{r}'_y$
Rotation matrix $\mathcal{R}_\theta$, where $\theta \leftarrow \arctan(a)$ $t \leftarrow t_s$ to t_e
$\begin{bmatrix} x_{w,t} & z_{w,t} \end{bmatrix}^\intercal \leftarrow \mathcal{R}_\theta \begin{bmatrix} x'_{w,t} & z'_{w,t} \end{bmatrix}^\intercal$
$P_{c,t} \leftarrow \mathbf{M} P_{w,t}$
$\mathbf{r} \leftarrow \mathcal{R}_\theta \mathbf{r}'_y$
$r_y \leftarrow \text{arctan2}(r_2 - z_c(t), r_1 - x_{c,t})$
$x_c \leftarrow x_c + x_o$
$z_c \leftarrow z_c + z_o$
`df['x3d']` $\leftarrow x_c$
`df['y3d']` $\leftarrow y_c$
`df['z3d']` $\leftarrow z_c$
`df['ry']` $\leftarrow r_y$
df

- `./DANGER_CUDA9-3D-SDN.pdf` Hardware setup for 3D-SDN and inference for the test.JSON obtained from the above step.
- `./DANGER_Eval_Score.pdf` Qualitative evaluation for the generated dataset and plot generation.

A.1.3 Datasets Sample

Posted at the end, see Figure 13.

A.1.4 Scene Editing Transformation

We only show the data generation and coordinate system conversion pseudocode applied in slalom lane change, see Section A.2.1 for a detailed visualization.

A.2 Details of experiments

A.2.1 Primitive Generation

In this section, we provide more detailed pictures of primitive generation, mainly focusing on the visualization of trajectories in the world coordinate system and the camera coordinate system.

Exit Parallel Parking In the world coordinate system, we generate the primitive function (blue curve) at the origin, push it to the target point, and convert to the camera coordinate system. The pink color in the figure is the original position of the vehicle parked on the side of the road in the camera coordinate system, while the dark blue point and the light blue arrow are the converted coordinate points and the vehicle head direction converted to the camera coordinate system according to the primitives, respectively.

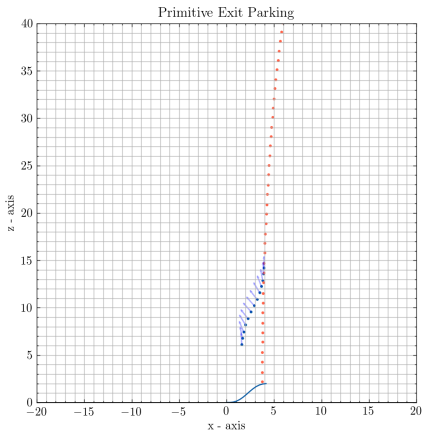


Figure 6: Visualization of primitive operation in camera space.

Cut-in Opposite In the world coordinate system, we generate the primitive function (blue curve) at the origin and push it to the target starting point shown as the green curve. The pink color in the figure is the original path curve, while the dark blue point and light blue arrow are the rotated coordinate point and the direction of the converted car head, respectively. The following sections color markings are the same.

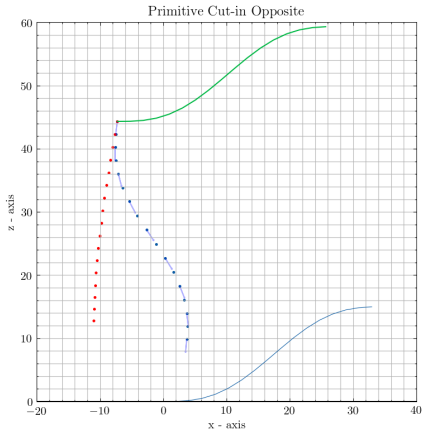


Figure 7: Visualization of primitive operation in world space.

Slalom

Cut-in

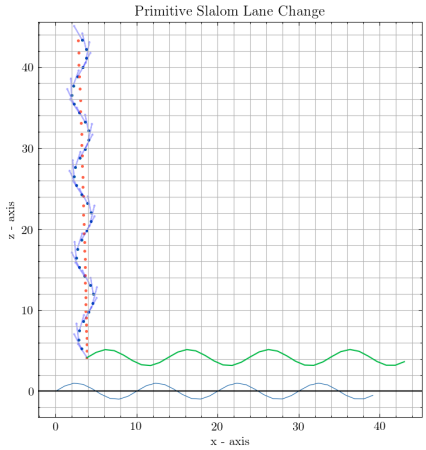


Figure 8: Visualization of slalom lane change operation in world space.

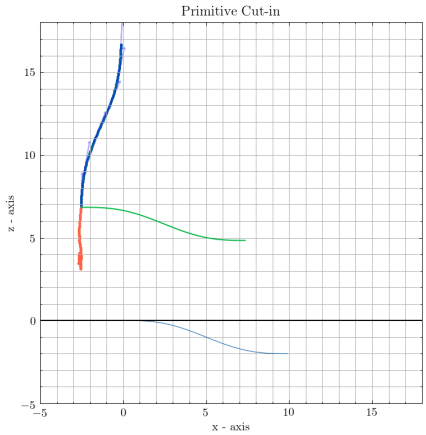


Figure 9: Visualization of cut-in operation in world space.

Brake Unlike the above method, we simulate the real braking deceleration in the brake primitive and transform it into the velocity and displacement profile of the vehicle object by integration. By re-projecting them onto the original driving trajectory in the world coordinate system, they are then transformed into relative position points in the camera coordinate system. Our approach is not just to bring the vehicle closer by a distance, but to pull the original vehicle to the real trajectory points based on the real physical travel.

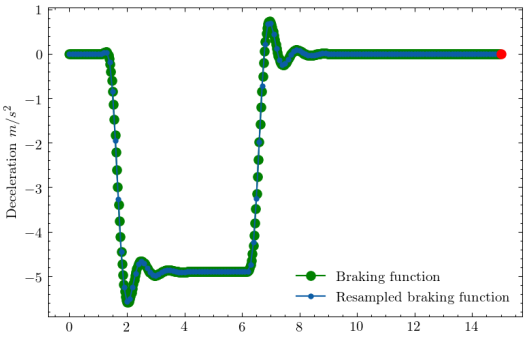


Figure 10: Braking acceleration based on the parameters input.

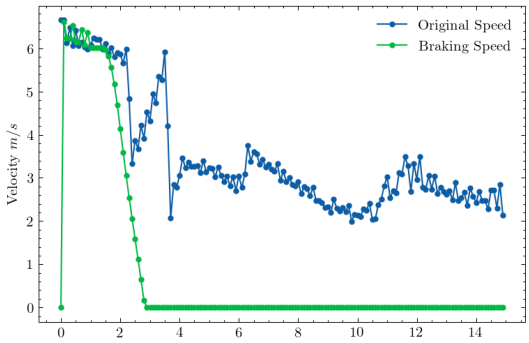


Figure 11: Original and modified speed profile.

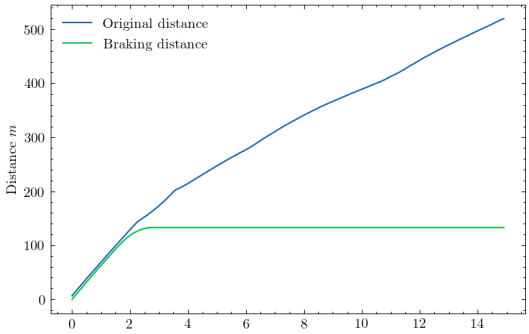


Figure 12: Original and modified travel distance profile

A.2.2 JSON file naming

We saved the generated JSON file by the following rule,

`<world>_<topic>_<tid>_<prim>_variable1#1_variable2#2.json`

where

- `<world>` is the name of a virtual world, which is the sequence number of the corresponding original "seed" real-world KITTI sequence (`0001`, `0002`, `0006`, `0018`, `0020`) in the 3D Object Detection Evaluation challenge **?**.
- `<topic>` denotes one of the 10 different rendering variations in terms of imaging or weather conditions. `clone`: rendering as close as possible to original real-world KITTI conditions; `morning`: typical lighting conditions after dawn on a sunny day; `sunset`: lighting typical of slightly before sunset; `overcast`: typical overcast weather (diffuse shadows, strong ambient lighting).
- `tid` The operation will apply to which car object with number `'<tid>'`. Please look up the number under column tid in Table. 4.
- `prim` Primitive parameters defined in Table. 4, and corresponding value `variable`.

A sample JSON file that contains the (u, v), `zoom`, and r_y information is listed as follows:

```
[
    {
        "world": "0018",
        "topic": "clone",
        "source": "00140",
        "target": "00140",
        "operations": [
            {
                "type": "modify",
                "from": {
                    "u": "350.3",
                    "v": "287.0"
                },
                "to": {
                    "u": "247.9",
                    "v": "319.6",
                    "roi": [ 0, 0, 0, 0]
                },
                "zoom": "1.3252036238820235",
                "ry": "0.056247797876731065"
            }
        ]
    },
    ...
]
```

	"0001" Exit Parking (tid, x_e, y_e)	"0002" Cut-in Opposite (tid, f, A)	"0006" Slalom (tid, f, A)	"0018" Cut-in (tid, x_e, y_e)	"0020" Braking (tid, t_1, dt, μ)
Case 1	(63, 4, 1.8)	(0, 20, 9)	(7, 0.1, 0.5)	(1, 6, 4)	(6, 5, 2.5, 0.35)
Case 2	(63, 4, 2)	(0, 23, 9)	(7, 0.1, 1.0)	(1, 8, 4)	(6, 5, 2.5, 0.2)
Case 3	(63, 4, 2.2)	(0, 26, 9)	(7, 0.1, 1.5)	(1, 10, 4)	(6, 5, 2.5, 0.5)
Case 4	(63, 4.5, 1.8)	(0, 29, 9)	(7, 0.08, 0.5)	(1, 6, 3)	(6, 5, 4, 0.35)
Case 5	(63, 4.5, 2)	(0, 32, 9)	(7, 0.08, 1.0)	(1, 8, 3)	(6, 5, 4, 0.5)
Case 6	(63, 4.5, 2.2)	(0, 35, 9)	(7, 0.08, 1.5)	(1, 10, 3)	(6, 5, 4, 0.2)
Case 7	(63, 5, 1.8)	(0, 20, 12)	(7, 0.12, 0.5)	(1, 6, 5)	(6, 10, 2.5, 0.35)
Case 8	(63, 5, 2)	(0, 23, 12)	(7, 0.12, 1.0)	(1, 8, 5)	(6, 10, 2.5, 0.2)
Case 9	(63, 5, 2.2)	(0, 26, 12)	(7, 0.12, 1.5)	(1, 10, 5)	(6, 10, 2.5, 0.5)
Case 10	(70, 4, 1.8)	(0, 29, 12)	(2, 0.1, 0.5)	(2, 6, -4)	(6, 10, 4, 0.35)
Case 11	(70, 4, 2)	(0, 32, 12)	(2, 0.1, 1.0)	(2, 8, -4)	(6, 10, 4, 0.5)
Case 12	(70, 4, 2.2)	(0, 35, 12)	(2, 0.1, 1.5)	(2, 10, -4)	(6, 10, 4, 0.2)
Case 13	(70, 4.5, 1.8)	(0, 20, 15)	(2, 0.08, 0.5)	(2, 6, -3)	(6, 2, 2.5, 0.35)
Case 14	(70, 4.5, 2)	(0, 23, 15)	(2, 0.08, 1.0)	(2, 8, -3)	(6, 2, 2.5, 0.2)
Case 15	(70, 4.5, 2.2)	(0, 26, 15)	(2, 0.08, 1.5)	(2, 10, -3)	(6, 2, 2.5, 0.5)
Case 16	(70, 5, 1.8)	(0, 29, 15)	(2, 0.12, 0.5)	(2, 6, -5)	(6, 2, 4, 0.35)
Case 17	(70, 5, 2)	(0, 32, 15)	(2, 0.12, 1.0)	(2, 8, -5)	(6, 2, 4, 0.5)
Case 18	(70, 5, 2.2)	(0, 35, 15)	(2, 0.12, 1.5)	(2, 10, -5)	(6, 2, 4, 0.2)

Table 4: Parameters in primitives.

A.2.3 Class used in `MobileNet-V2`

No.	Class	No.	Class	No.	Class	No.	Class
1	Road	6	Pole	11	Sky	**16**	**Bus**
2	Sidewalk	7	Traffic ligh	**12**	**Person**	**17**	**Train**
3	Building	8	Traffic sign	**13**	**Rider**	**18**	**Motorcycle**
4	Wall	9	Vegetation	**14**	**Car**	**19**	**Bicycle**
5	Fence	10	Terrain	**15**	**Truck**		

Table 5

Glossary

A Amplitude of the a car in sine-wave slalom maneuver. Unit in m. 21, 22

μ The constantly steady braking acceleration we designed to reach. Unit in $\#g$. For example, $0.5g = 4.9m/s^2$. 21, 22

dt The duration of applying a braking. $d_t = t_3 - t_2$. Unit in s. 21, 22

f Frequency of the a car in sine-wave slalom maneuver. Unit in Hz. 21, 22

r_y KITTI-like 3D object `rotation about y`, rotation around Y-axis (yaw) in camera coordinates $[-\pi..\pi]$ (KITTI convention is $r_y == 0$ iff object is aligned with x-axis and pointing right). 22

t_1 Starting time of applying a braking. Unit in s. 21, 22

x_e Longitudinal movement of a car in cut-in. 21, 22

y_e Lateral movement of a car in cut-in. 21, 22

delete A description used in JSON. Delete an object from current view by 3D-SDN. 22

frame Frame index in the video (starts from 0). 22

modify A description used in JSON. Moving an object from a position to another by 3D-SDN. 22

motgt Multi-Object Tracking Ground Truth. 22

scene Scene shares the **world** definition in vKITTI. It's the name of a virtual world, which is the sequence number of the corresponding original "seed" real-world KITTI sequence (0001, 0002, 0006, 0018, 0020). 22

tid Track identification number (unique for each object instance). In the context of DANGER, tid means the id of a specific car object. 21, 22

topic Denotes one of the 10 different rendering variations in terms of imaging or weather conditions: **clone**: rendering as close as possible to original real-world KITTI conditions; **15-deg-right**: horizontal rotation of the camera 15 degrees to the right; **15-deg-left**: horizontal rotation of the camera 15 degrees to the left; **30-deg-right**: horizontal rotation of the camera 30 degrees to the right; **30-deg-left**: horizontal rotation of the camera 30 degrees to the left; **morning**: typical lighting conditions after dawn on a sunny day; **sunset**: lighting typical of slightly before sunset; **overcast**: typical overcast weather (diffuse shadows, strong ambient lighting); **fog**: fog effect implemented via a volumetric formula; **rain**: simple rain particle effect (ignoring the refraction of water drops on the camera). Only inference **CLONE** in this implementation. 22

w3d, h3d, l3d KITTI-like 3D object 'dimensions', respectively width, height, length in meters. 22

x3d, y3d, z3d KITTI-like 3D object 'location', respectively x, y, z in camera coordinates in meters (center of bottom face of 3D bounding box). 22

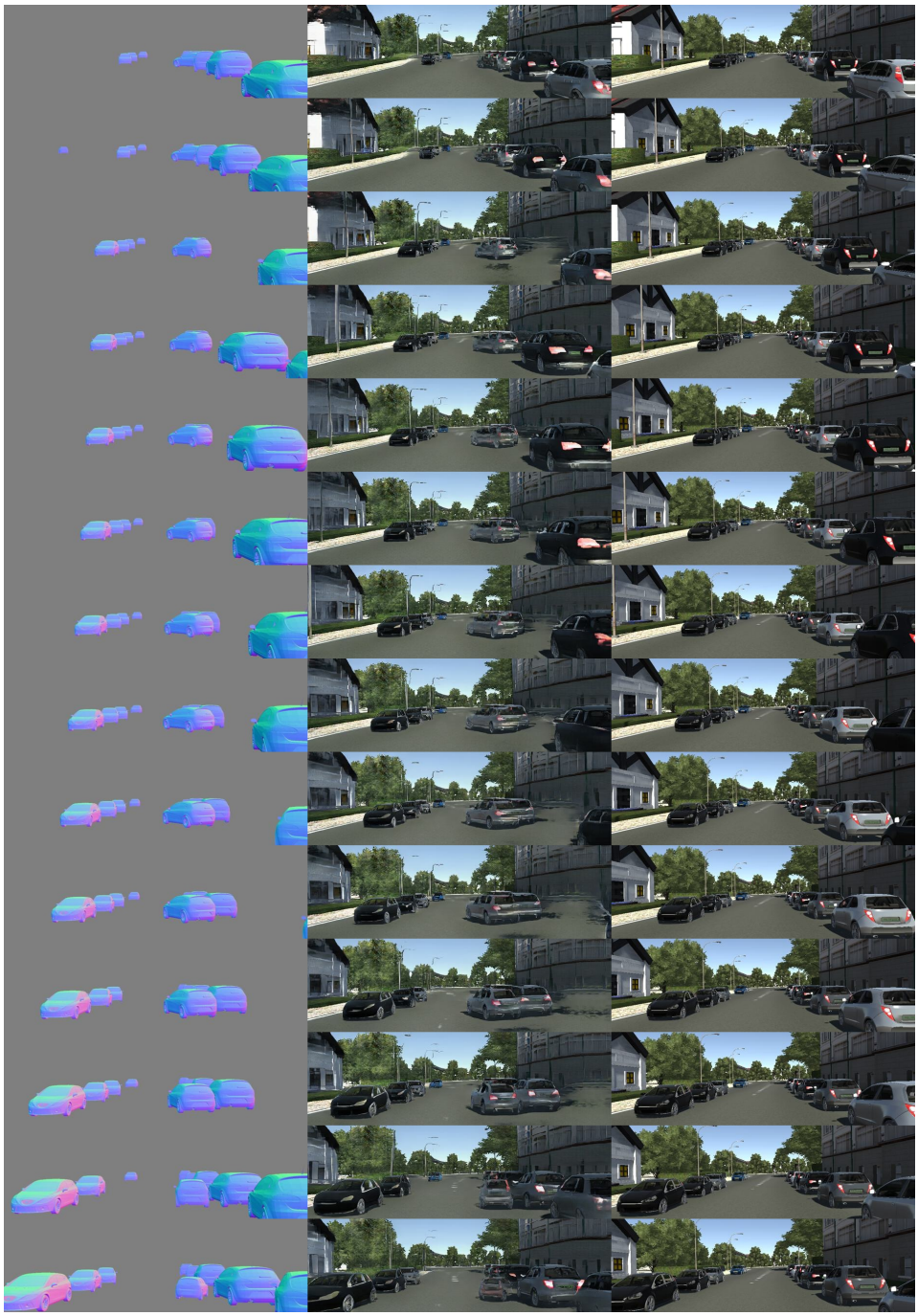


Figure 13: `0001_clone_tid70_epp_xe5_ye2.2`

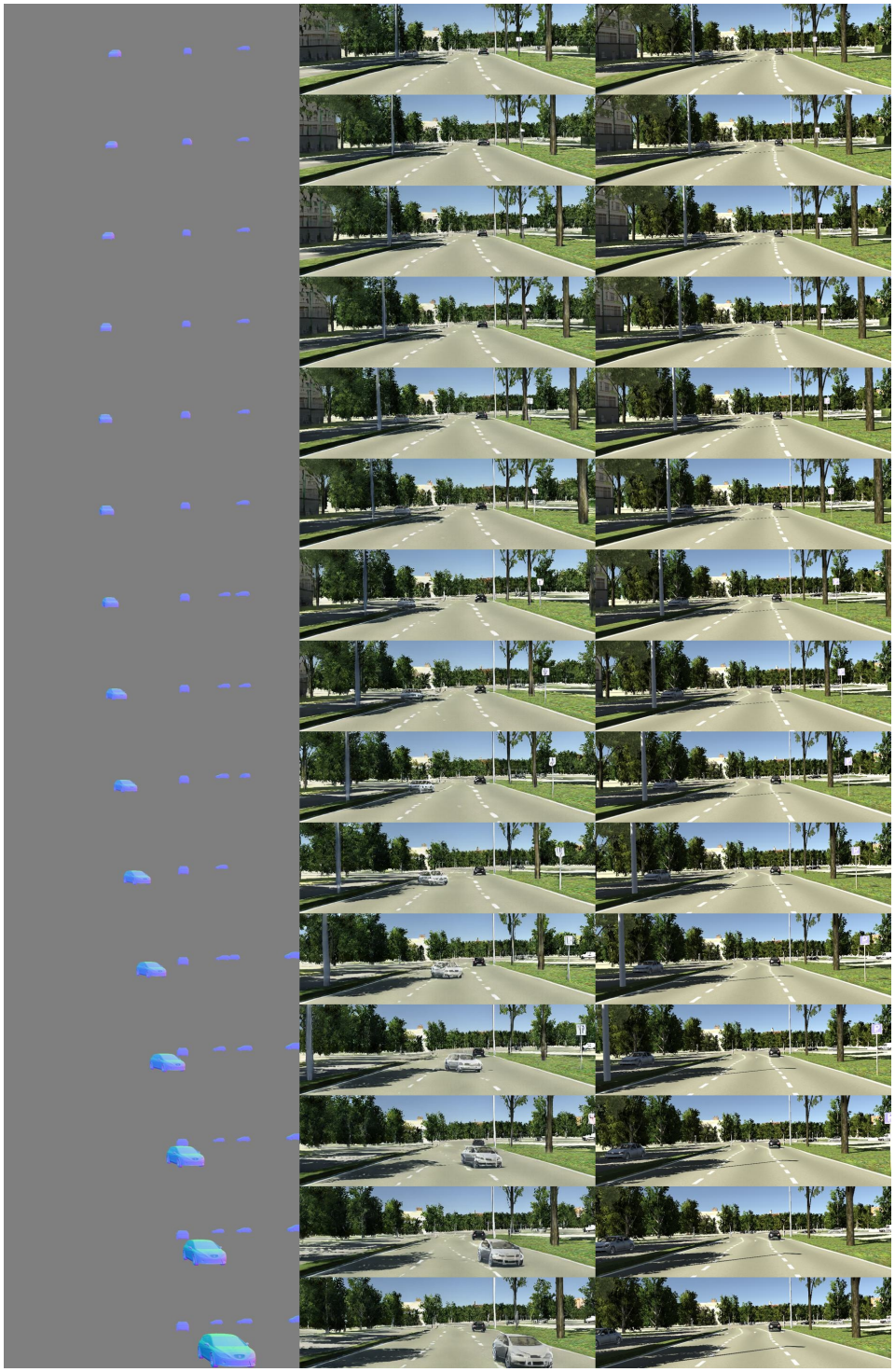


Figure 14: `0002_clone_tid0_cio_xe40_ye15`

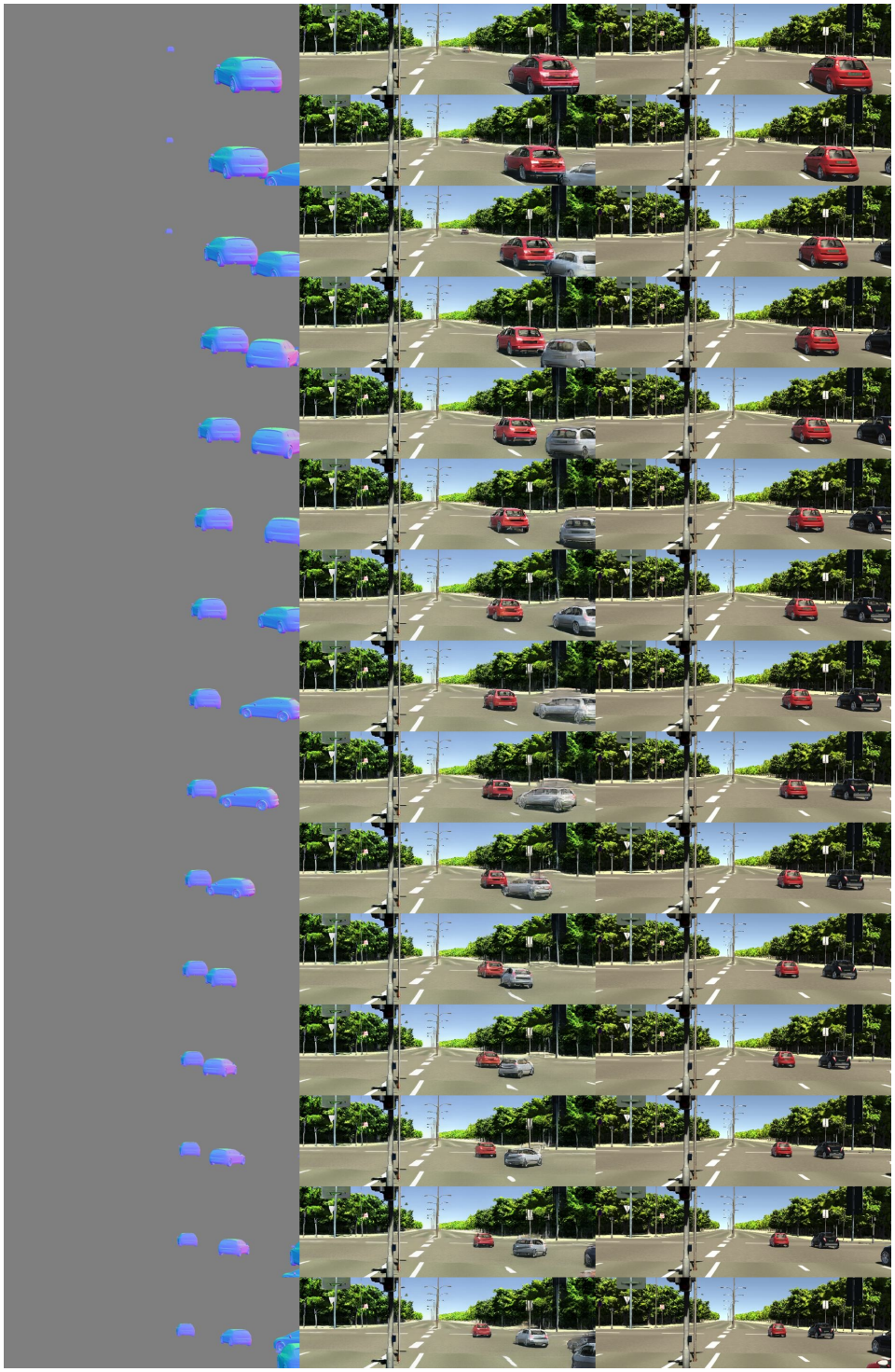


Figure 15: 0006_clone_tid7_lc_f0pt12_A1pt5

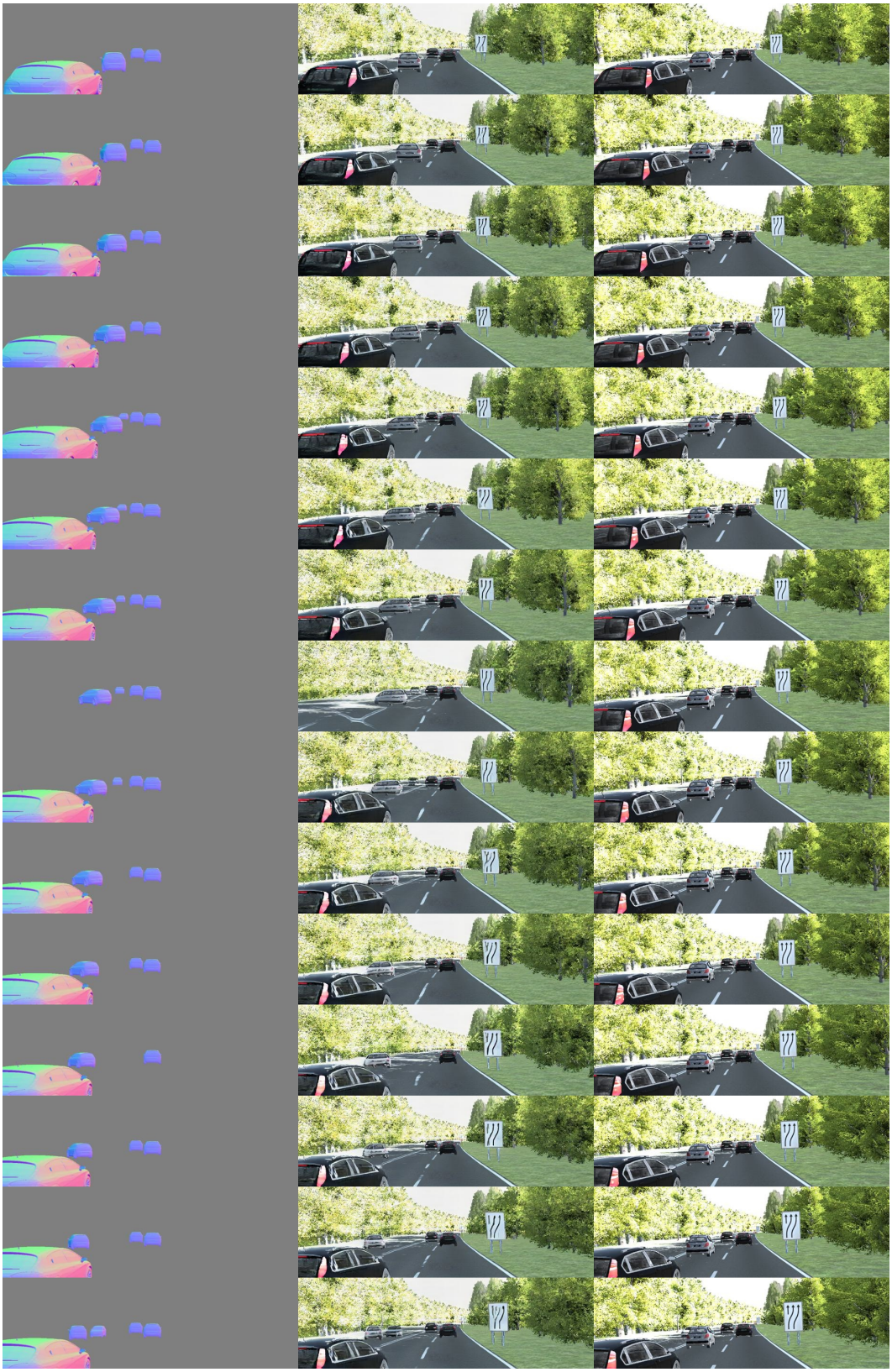


Figure 16: 0018_clone_tid2_ci_xe10_ye5

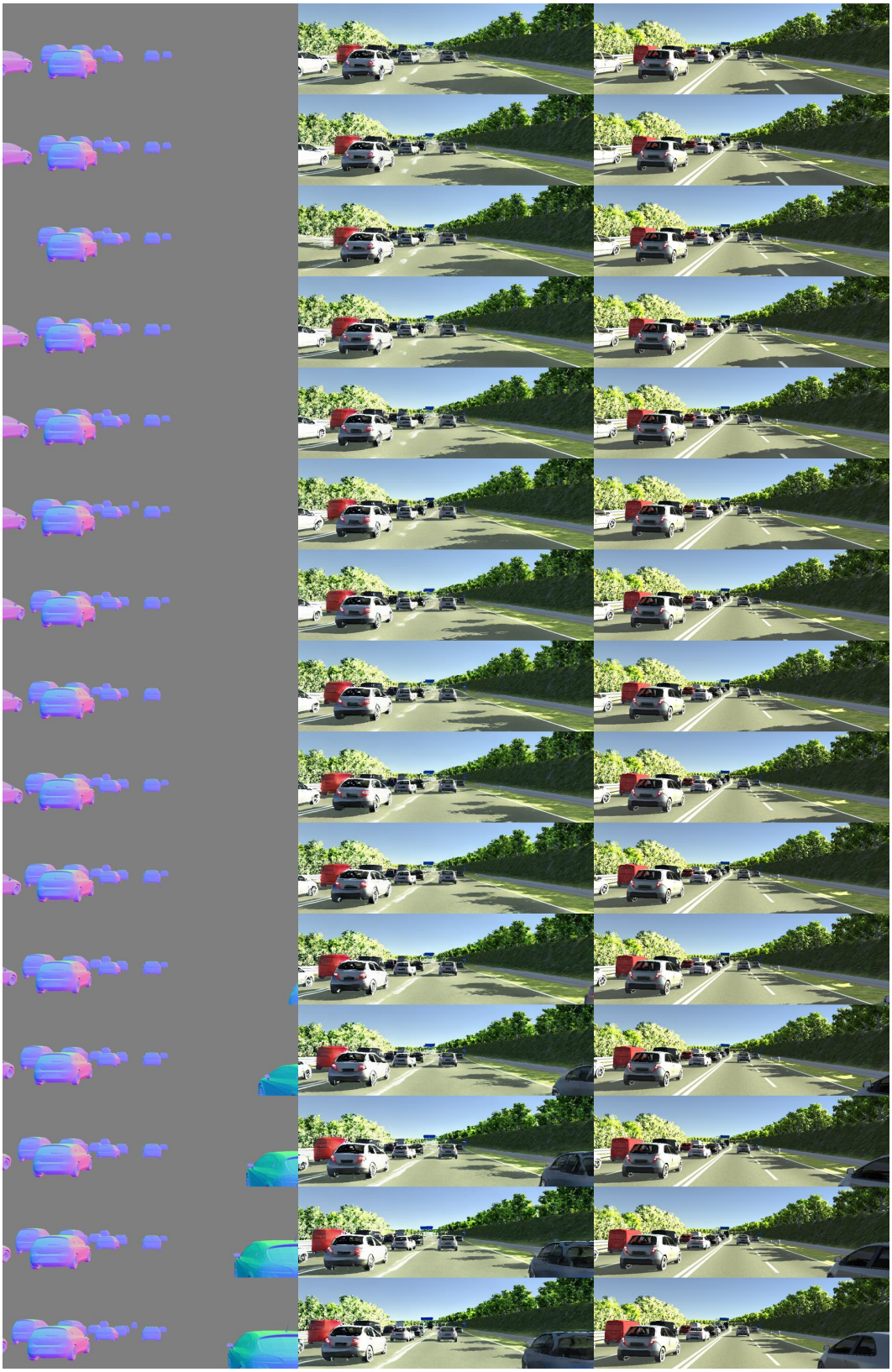


Figure 17: `0020_clone_tid16_br_t010_dt4_g0pt35`