SurfelGAN: Synthesizing Realistic Sensor Data for Autonomous Driving

Zhenpei Yang\textsuperscript{1*}, Yuning Chai\textsuperscript{2}, Dragomir Anguelov\textsuperscript{2}, Yin Zhou\textsuperscript{2}, Pei Sun\textsuperscript{2}, Dumitru Erhan\textsuperscript{3}, Sean Rafferty\textsuperscript{2}, Henrik Kretzschmar\textsuperscript{2}

\textsuperscript{1}UT Austin, \textsuperscript{2}Waymo, \textsuperscript{3}Google Brain

Abstract

Autonomous driving system development is critically dependent on the ability to replay complex and diverse traffic scenarios in simulation. In such scenarios, the ability to accurately simulate the vehicle sensors such as cameras, lidar or radar is hugely helpful. However, current sensor simulators leverage gaming engines such as Unreal or Unity, requiring manual creation of environments, objects, and material properties. Such approaches have limited scalability and fail to produce realistic approximations of camera, lidar, and radar data without significant additional work.

In this paper, we present a simple yet effective approach to generate realistic scenario sensor data, based only on a limited amount of lidar and camera data collected by an autonomous vehicle. Our approach uses texture-mapped surfels to efficiently reconstruct the scene from an initial vehicle pass or set of passes, preserving rich information about object 3D geometry and appearance, as well as the scene conditions. We then leverage a SurfelGAN network to reconstruct realistic camera images for novel positions and orientations of the self-driving vehicle and moving objects in the scene. We demonstrate our approach on the Waymo Open Dataset and show that it can synthesize realistic camera data for simulated scenarios. We also create a novel dataset that contains cases in which two self-driving vehicles observe the same scene at the same time. We use this dataset to provide additional evaluation and demonstrate the usefulness of our SurfelGAN model.

1. Introduction

Recent advances in deep learning have inspired breakthroughs in multiple areas related to autonomous driving such as perception [16, 27], prediction [5, 7] and planning [13]. These recent trends only underscore the increasingly significant role of data-driven system development. One aspect is that deep learning networks benefit from large training datasets. Another is that autonomous driving system evaluation requires the ability to realistically replay a large set of diverse and complex scenarios in simulation capturing sensor properties, seasons, time of day, and weather. Developing simulators that support the levels of realism required for autonomous system evaluation is a challenging task. There are many ways to design simulators, including simulating mid-level object representations [5, 12]. However, mid-level representations omit subtle perceptual cues that are important for scene understanding, such as pedestrian gestures and blinking lights on vehicles. Furthermore, as end-to-end models that combine perception, prediction, and sometimes even control become an increasingly more popular direction of research, we are faced with the need to faithfully simulate the sensor data, which is the input to such models during scenario replay.

Frameworks for autonomous driving that support realistic sensor simulation are traditionally built on top of gaming engines such as Unreal or Unity [12]. The environment and its object models are created and arranged manually, to approximate real-world scenes of interest. In order to enable realistic LiDAR and radar modeling, material properties often need to be manually specified as well. The overall process is time-consuming and not scalable. Furthermore, simple ray-casting or ray-tracing techniques are often insufficient to generate realistic camera, LiDAR, or radar data for a specific self-driving system, and additional work is required to adapt the simulated sensor statistics to the real sensors.

In this work, we propose a simple yet effective data-driven approach for creating realistic scenario sensor data. Our approach relies on camera and LiDAR data collected in a single pass, or several passes, of an autonomous vehicle through a scene of interest. We use this data to reconstruct the scene using a texture-mapped surfel representation. This representation is simple and computationally efficient to create and preserves rich information about the 3D geometry, semantics, and appearance of all objects in the scene. Given the surfel reconstruction, we can render the scene for novel poses of the self-driving vehicle (SDV) and the other scenario agents. The rendered reconstruction for these novel views may have some missing parts due to occlusion differences between the initial and the new configuration. It can

\*Work done as an intern at Waymo. Correspondence: yzp@utexas.edu
also have visual quality artifacts due to the limited fidelity of the surfel reconstruction. We address this gap by applying a GAN network [15] to the rendered surfel views to produce the final high-quality image reconstructions. An overview of our proposed system is illustrated in Fig. 1.

Our work makes the following contributions: 1) We describe a pipeline that builds a detailed reconstruction of a dynamic scene from real-world sensor data. This representation allows us to render novel views in the scene, corresponding to deviations of the SDV and the other agents in the environment from their initially captured trajectories (Sec. 3.1). 2) We propose a GAN architecture that takes in the rendered surfel views and synthesizes images with quality and statistics approaching that of real images (Tab. 1). 3) We build the first dataset for reliably evaluating the task of novel view synthesis for autonomous driving, which contains cases in which two self-driving vehicles observe the same scene at the same time. We use this dataset to provide additional evaluation and demonstrate the usefulness of our SurfelGAN model.

2. Related Work

Simulated Environment for Learning Agent. There have been many efforts towards building simulated environments for various tasks [6, 12, 40, 41, 42]. Much work has focused on indoor environments [6, 40, 42] based on public indoor datasets such as SUNCG [36] or Matterport3D [8]. In contrast to indoor settings where the environment is relatively simple and easy to model, simulators for autonomous driving exhibit significant challenges in modeling the complicated and dynamic scenarios of real-world scenes. TORCS[41] is one of the first simulation environments that support multi-agent racing, but is not tailored for real-world autonomous driving research and development. DeepGTA [1] provides a plugin that transforms the Grand Theft Auto gaming environment into a vision-based self-driving car research environment. CARLA[12] is a popular open-source simulation engine that supports the training and testing of SDVs. All these simulators rely on manual creation of synthetic environments, which is a formidable and laborious process. In CARLA [12], the 3D model of the environment, which includes buildings, road, vegetation, vehicles, and pedestrians, is manually created. The simulator provides one town with 2.9 km of drivable roads for training and another town with 1.4 km of drivable roads for testing. In contrast, our system is easily extendable to new scenes that are driven by an SDV. Furthermore, because the environment we are building is a high-quality reconstruction based on the vehicle sensors, it naturally closes the domain gap between synthetic and real contents, which is present in most traditional simulation environments. AADS [24] shares similar ideas as ours in that it also utilizes real sensor data to synthesize novel views. The major difference is that we reconstruct the 3D environment, while AADS uses a purely image-based novel view synthesis. Reconstructing the 3D environment gives us the freedom to synthesize novel views that could not be easily captured in the real world. Moreover, once our environment is built, we no longer need to store the images as well as the query of the nearest $K$ views upon synthesis, which saves time for deployment.

Learning on Synthetic Data. Besides enabling end-to-end training and evaluation of embodied agents, the simulated environment can also provide a large amount of data for training deep neural networks. [33] uses a synthetic scene to generate a large amount of fully labeled training data for urban scene segmentation. [20] generate images containing...
novel placement of dynamic objects to boost the performance of object detection.

Geometric Reconstruction and 3D Representations. Our work is also closely related to 3D reconstruction for outdoor environments. The typical approach is to use structure from motion [37, 39] and multi-view stereo[14] to recover a dense 3D point cloud from image collections, then optionally use Poisson reconstruction[22] to obtain a mesh representation. Such a paradigm is most suitable for the case where we have multiple images covering the same area from different perspectives, which is not always true in our case. Thanks to the rapid advancement of LiDAR technology, we can have accurate depth information to complement the camera image data. Our work uses this to augment the traditional surfel[32] representation with fine-grained image textures, which not only greatly simplifies the reconstruction process, but it also effectively enhances the representation of the colored real world. Truncated Signed Distance Functions [11] and their most recent variants [30] are also promising alternatives to the surfel. We also noted the recent work [3] that learns a point-wise dense descriptor on point cloud for rendering purpose. Although they’ve shown promising results, however their approach assumes a static environment, and is not practically applicable to outdoor scenario where a scene usually contains tens of millions of points.

GAN-based Image Translation. Generative Adversarial Networks (GAN) [15] have attracted broad interests in both academia and industry. While [15] aims at synthesis realistic images, [19] generalizes its framework to a more practical conditional image synthesis setting. Subsequent research [4, 21, 31, 44] has made great strides in improving the quality of images generated by GAN methods. We refer the readers to [10] for details on this topic. [38] trained a video synthesizer on Cityscape [9] that can convert video of a semantic map into a video of realistic images. However, their training procedure requires pixel-wise annotated semantic images, which is very expensive in practice. In contrast, our approach uses labeled 3D bounding boxes for several dynamic object classes and therefore is more cost-effective. We believe even this requirement can be further relaxed, by replacing the ground-truth 3D boxes with 3D boxes produced by a high-quality offline 3D perception pipeline. Finally, we address the usual problem of GAN evaluation by proposing two new metrics that are suitable for the task of novel view synthesis.

3. Approach

In this section, we explain the key innovations of this work, which lays the foundation for creating a data-driven simulation environment supporting sensor realism: surfel scene reconstruction and image synthesis via the SurfelGAN.

3.1. Surfel Scene Reconstruction

Enhanced Surfel Map. A good scene reconstruction should achieve faithful preservation of the sensor information while being efficient to compute and storage. We propose a novel texture-enhanced surfel map representation. Surfels are compact, easy to reconstruct, and because of their fixed size, easy to texture and compress. Below we describe our approach, which contains more fine-grained details compared to traditional surfel map representations [32].

We discretize the scene into a 3D voxel grid of fixed size and process LiDAR scans in the order they are captured. For each voxel, we construct a surfel disk by estimating the mean coordinate and the surfel normal, based on all the LiDAR points that fall within that voxel. The surfel disk radius is defined as \( \sqrt{3v} \), where \( v \) denotes the voxel size. For the LiDAR points binned in a voxel, we also have corresponding colors from the camera image, which we can use to estimate the surfel color. Note that traditional surfel maps suffer from the trade-off between geometry consistency and fine-grained details, i.e., a large voxel size gives better geometry consistency but fewer details, while small voxel size results in finer details but less stable geometry. Therefore, we take an alternative approach that aims to achieve both good geometry consistency and rich texture details. Specifically, we discretize each surfel disk into a \( k \times k \) grid centered on its point centroid, as illustrated in subfigure b) in Fig. 1. Each grid center is assigned an independent color to encode higher-resolution texture details.

Since each surfel may have a different appearance across different frames, due to the variations of the lighting conditions and the changes of relative pose (distance and view angle), we propose to enhance the surfel representation by creating a codebook of such \( k \times k \) grids at \( n \) various distances. For each bin, we determine its color from the first observation, which we found is important to obtain a smooth rendering image. During the rendering stage, we determine which \( k \times k \) patch to use based on the camera pose. The final rendering is shown in Fig. 2. We can see that the baseline surfel map introduces many artifacts at object boundaries and yields non-smooth coloring at non-boundary areas. In contrast, our texture-enhanced surfel map eliminates much of the artifacts and gives vivid-looking images. In our experiments, we use \( v = 0.2m, k = 5 \) and \( n = 10 \).

Handling Dynamic Objects. We consider vehicles as rigid dynamic objects and reconstruct a separate model for each. For simplicity, we leverage the high-quality 3D bounding box annotations from the Waymo Open Dataset [2] to accumulate the LiDAR points from multiple scans for each object of interest. We apply the Iterative Closest Point (ICP) [34] algorithm to refine the point cloud registration, producing a dense point cloud that allows an accurate, enhanced surfel reconstruction for each vehicle. Please see Supplementary
Sec. A for reconstructed examples. Our approach does not strictly require 3D box ground-truth; we could also leverage a state of the art vehicle detection and tracking algorithm [28, 35] to get initial estimates for ICP. However, we leave this experiment for future work.

When simulating our environment, the reconstructed vehicle models can be placed in any location of choice. In the case of the pedestrians, which are deformable objects, we reconstruct a separate surfel model for each LiDAR scan separately. We allow placement of the reconstructed pedestrian anywhere in the scene for that scan. We leave the task of accurate deformable model reconstruction from multiple scans to future work.

3.2. Image Synthesis via SurfelGAN

While the surfel scene reconstruction captures a rich environment representation, it produces surfel renderings that have a non-negligible realism gap when compared to real images, due to incomplete reconstruction and imperfect geometry and texturing (see Fig. 2). SurfelGAN is designed to address the issue.

Ultimately, we like to learn a generator (SurfelGAN) that converts surfel renderings rendered from the surfel scene reconstruction to realistically looking images. We treat semantic and instance segmentation maps as part of the surfel rendering. For the sake of simplicity, we omit to mention them explicitly.

Let the generator $G_{φ_2}^{S→I}$ be an encoder-decoder model parameterized by the learnable variable $θ_2$. Given pairs of surfel renderings $S_p$ and images $I_p$, the supervised loss can be applied to train the generator. We call SurfelGAN that is trained solely with supervised learning SurfelGAN-S.

Additionally, we can add an adversarial loss dictated by a real image discriminator $D_{φ_1}^I$ parameterized by trainable variable $φ_1$. SurfelGANs trained with this additional loss is named SurfelGAN-SA.

However, paired training data between surfel rendering and real image is very limited. Unpaired data is however easy to obtain. We leverage unpaired data for two purposes, i.e. improving the generalization of discriminator by training with more unlabeled examples, and regularizing the generator by enforcing cycle consistency. Let the reverse generator $G_{θ_1}^{I→S}$ be another encoder-decoder model which has the same architecture as $G_{θ_2}^{S→I}$ except more output channels for semantic/instance map, any surfel rendering, paired $S_p$ or unpaired $S_u$ can be translated to a real image and translated back to a surfel rendering, where a cycle consistency loss can be applied. The same applies to any paired $I_p$ or unpaired $I_u$ real image as well. Finally, we add the surfel rendering discriminator $D_{φ_2}^S$ that judges generated surfel images. We call SurfelGANs trained with additional cycle consistency SurfelGAN-SAC. An intuitive overview of the training strategy is shown in Fig. 3. And please refer to Sec. 4 for a detailed description of paired and unpaired data. We optimize the following objective:

$$
\max_{θ_2, φ_1} \min_{φ_2, θ_1} \mathcal{L}_r(G_{θ_2}^{S→I}, S_p, I_p) + \lambda_1 \mathcal{L}_c(G_{θ_1}^{I→S}, I_p, S_p) + \lambda_2 \mathcal{L}_a(G_{θ_2}^{S→I}, D_{φ_1}^I, S_p, I_u) + \lambda_3 \mathcal{L}_u(G_{θ_1}^{I→S}, D_{φ_2}^S, I_p, I_u) + \lambda_4 \mathcal{L}_c(G_{θ_2}^{S→I}, G_{θ_1}^{I→S}, S_p, I_u) + \lambda_5 \mathcal{L}_c(G_{θ_2}^{S→I}, G_{θ_1}^{I→S}, S_p, I_u)
$$

where $\mathcal{L}_r$, $\mathcal{L}_a$, $\mathcal{L}_c$ are supervised reconstruction, adversarial, and cycle consistency loss. We use hinged Wasserstein loss for adversarial training [25, 29, 43] in our experiments as it...
We set the initial learning rate to $2e^{-4}$ for both the generator and the discriminator and set $\beta_1 = 0.5$ and $\beta_2 = 0.9$. We use batch normalization [18] after ReLU activation. We set $\lambda_1 = 1$, $\lambda_2 = \lambda_3 = 0.001$, $\lambda_4$, $\lambda_5 = 0.1$ in all of our experiments.

Figure 3. (Best viewed in color) **SurfelGAN training paradigm.** The training setup has two symmetric encoder-decoder generators mapping from surfel renderings to real images $G^{S\to I}$ and vice versa $G^{I\to S}$. Additionally, there are two discriminators, $D^S$, $D^I$, which specialize in the surfel and the real domain. The losses are shown as colored arrows. Green: supervised reconstruction loss. Red: adversarial loss. Blue/Yellow: cycle-consistency losses. When training with paired data, e.g. WOD-TRAIN, the surfel renderings translate to real images, and we can apply a one-directional supervised reconstruction loss (SurfelGAN-S) only or add an additional adversarial loss (SurfelGAN-SA). When training with unpaired data, we can go either from the surfel renderings (e.g. WOD-TRAIN-NV) or the real images (e.g. Internal Camera Dataset), use one of the encoder-decoder networks to get to the other domain and back. We can then apply a cycle consistency loss. (SurfelGAN-SAC). The encoder-decoder networks consist of 8 convolutional and 8 deconvolutional layers. Discriminators consist of 5 convolutional layers. All network operate on $256 \times 256$ sized input.

Helps to stabilize the training. We use $\ell^1$-loss as reconstruction and cycle-consistency loss for renderings and images and cross entropy loss for semantic and instance maps.

**Distance Weighted Loss.** Due to the limited coverage of the surfel map, our surfel rendering contains large areas of unknown regions. The uncertainty in those regions is much higher than that of the region with surfel information. Also, the distance between the camera and the surfel introduces another factor of uncertainty. Therefore, we use a distance weighted loss to stabilize our GAN training. Specifically, during data pre-processing, we generate a distance map that records the nearest distance to the observed region and then uses the distance information as weighting coefficients to modulate our reconstruction loss.

**Training Details.** We use Adam[23] optimizer for training.

We collected additional 9.8k short sequences (100 frames for each) similar to WOD images. These unannotated images are used for unpaired training of real images.

**Dual-Camera-Pose Dataset (DCP)** Finally, we built a unique dataset tailored for measuring the realism of our model. The dataset contains scenarios where two vehicles observe the same scene at the same time. Specifically, we find the interval where two vehicles are within $20m$ of each other. We use the sensor data from the first vehicle to reconstruct the scene, and render the surfel image at the exact pose
of the second vehicle. After filtering cases where the scene reconstruction is too incomplete, we obtain around 1k pairs, for which we can directly measure the pixel-wise accuracy of the generated image.

4.1. Model Variants and Baseline

Most experiments were performed on three variants of our proposed model. **Supervised (S)**: we train the surfel-rendering-to-image model in a supervised way by minimizing an $\ell_1$-loss between the generated image and the groundtruth real image. This type of training requires paired data. Hence, it is only possible to train on WOD-TRAIN. **Supervised + Adversarial (SA)**: we still only consider WOD-TRAIN as the training data. However, we add an adversarial loss alongside the $\ell_1$-loss. **Supervised + Adversarial + Cycle (SAC)**: in this variation, we also use WOD-TRAIN-NV and the Internal Camera Image Dataset. Since these two sets are unpaired, the supervised loss does not apply. We propose to use a cycle-consistency loss in addition to the adversarial loss, as discussed in Sec. 3.2.

The baseline for our applications is the direct surfel renderings (Surfel) that serve as the input to our model.

4.2. Vehicle Detector Realism

Since the primary application of this work is the simulation of the camera data, a natural quality metric is to evaluate it using a downstream module. To this end, we like to know, without any fine-tuning, how well an off-the-shelf vehicle detector performs on the generated images, and, whether the detector perceived the generated images similarly to real images. We chose to use a vehicle detector with a ResNet architecture [17] and an SSD detection head [27], trained and evaluated on resized images of a $512 \times 512$ resolution. It is trained on a mixture of datasets that include WOD-TRAIN.

We trained our SurfelGAN model variants on a mixture of WOD-TRAIN, WOD-TRAIN-NV, and the Internal Camera Image Dataset, and generated images on WOD-TRAIN-NV, WOD-EVAL and WOD-EVAL-NV. Tab. 1 shows the quantitative comparison of the detector’s quality on the original surfel renderings that is the input SurfelGAN, images generated from the variants of SurfelGAN and real images. A few examples are displayed in Fig. 4. Please find additional visu-
It is notable that our texture-enhanced surfel scene reconstruction already produces surfel renderings that achieve good detection quality on the WOD-EVAL set at 52.1% AP@50. But there is still a significant gap between these surfel renderings and real images at 61.9%, which motivates our SurfelGAN work. As shown in Tab. 1, SurfelGAN-S, -SA and -SAC variants gradually improve over the baseline surfel renderings. SurfelGAN-SAC ultimately improves the AP@50 metric from 52.1% to 62.0% on WOD-EVAL, which is on par to the real images at 61.9%. It shows that images generated by SurfelGAN-SAC are close to real images in the eyes of the detector, which is the primary motivation of this work.

There are two types of generalization worth discussing. The first type is whether a SurfelGAN model trained on one set of scenes (e.g. WOD-TRAIN and WOD-TRAIN-NV) generalizes to new scenes (e.g. WOD-EVAL and WOD-EVAL-NV). We believe that the SurfelGAN model generalizes well since the relative improvement of SurfelGAN over the baseline is very similar between the WOD-TRAIN-NV and WOD-EVAL-NV columns in Tab. 1.

The second type of generalization is whether surfel rendering has a strong bias towards the poses from which the scene was reconstructed. We compare the metric values between WOD-EVAL and WOD-EVAL-NV. Although SurfelGAN improved by roughly 10% over the baseline in both cases, there is a noticeable quality difference between the two columns. To better understand this difference, in Tab. 2, we breakdown the metrics of SurfelGAN-SAC on WOD-EVAL-NV according to how much each pose deviates from the original poses in WOD-EVAL. The deviation \( d(t,R) \) is defined as a weighted sum of both translational and rotational differences of the poses:

\[
d((t,R),(t',R')) = ||t-t'|| + \lambda_R \frac{||log(RTR')||}{\sqrt{2}} \tag{2}
\]

where \( t \) and \( R \) are the pose (translation and rotation) of the novel view in WOD-EVAL-NV, and \( t', R' \) the pose of its closest pose in WOD-EVAL, \( \lambda_R \) is chosen to be 1.0.

It is an indication that the surfel renderings do have a quality bias w.r.t viewing direction, which means we should not perturb too much from the original poses if we want higher quality synthesized data. However, we believe that this problem can be negated if we were to reconstruct the surfel scene from multiple runs which bring in more accuracy. We did not explore this direction but left it to future work.

### 4.3. Image-Pixels Realism

The Dual-Camera-Pose (DCP) Dataset contains scenarios where two vehicles observe the same scene at the same time so that we can reconstruct the surfel scene using one camera and generate images at the poses of the second camera. We then match the generated image to the real one and report the \( \ell^1 \)-distance error on the pixels that are covered by the surfel rendering. This is to ensure that there is a fair comparison between the surfel renderings and the generated images. Like in the previous experiment, the model is trained using WOD-TRAIN, WOD-TRAIN-NV, and the Internal Camera Image Dataset. The results are showing in Tab. 3. SurfelGAN improves on top of the surfel renderings, generating images that are closer to real images in \( \ell^1 \)-distance. However, it is worth noting that the SurfelGAN-S version outperforms...
both SA and SAC that used additional losses and data during training. This finding is not unexpected since SurfelGAN-S optimizes for the $\ell^1$-distance.

### 4.4. Improving Detection by Data Augmentation

In a bonus experiment, we explore whether the generated images are helpful as a tool of data augmentation to training a vehicle object detector. For the baseline, we trained a vehicle detector on WOD-TRAIN and evaluated the detector’s quality on WOD-EVAL. We then trained another vehicle detector using both WOD-TRAIN and surfel images generated from WOD-TRAIN-NV, and also evaluated on WOD-EVAL.

WOD-TRAIN-NV only inherits 3D bounding boxes from WOD-TRAIN, and does not contain tightly fitting 2D bounding boxes like those in WOD-TRAIN. We approximate the latter by projecting all surfels in the 3D bounding boxes to the 2D novel view and take the axis-aligned bounding box as an approximation. The results are shown in Tab. 4. The data augmentation significantly boosted the detector’s metric on average precision, boosting the AP@50 score from 21.9% to 25.4%, the AP@75 from 10.8% to 12.1%, and the average AP from 11.9% to 13.0%. It is worth noting that these AP scores are much lower than those in Tab. 1. The main reason for the discrepancy is that images are resized differently in order to use the off-the-shelf detector in Tab. 1. We also trained on surfel renderings directly. There is a slight improvement compared to training only on WOD-TRAIN. But there is still a significant gap between augmenting using SurfelGAN synthesized images, which further demonstrate the realism of the SurfelGAN model.

### 5. Conclusion

In this work, we proposed a simple yet effective data-driven approach, which can synthesize camera data for autonomous driving simulations. Based on camera and lidar data captured by a vehicle pass through a scene, we reconstruct a 3D model using our Enhanced Surfel Map representation. Given this representation, we are able to render novel views and configurations of objects in the environment. We use our SurfelGAN image synthesis model to fix any reconstruction, occlusion or rendering artifacts. To the best of our knowledge, we have built the first purely data-driven camera simulation system for autonomous driving. Experimental results not only demonstrate the high level of realism of our synthesized sensor data, but also show the data can be used for training dataset augmentation for deep neural networks. In future work, we plan to enhance camera simulation further by improving the dynamic object modeling process and by investigating temporally consistent synthesis module.
References

[1] Deepgta v2. 2
[9] 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 CVPR, 2016. 3
[33] 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 CVPR, 2016. 2


