MonoPlace3D: Learning 3D-Aware Object Placement for 3D Monocular Detection

Rishubh Parihar[*] Srinjay Sarkar[*] Sarthak Vora[*] Jogendra Nath Kundu R. Venkatesh Babu
IISc Bangalore

Abstract

Current monocular 3D detectors are held back by the limited diversity and scale of real-world datasets. While data augmentation certainly helps, it's particularly difficult to generate realistic scene-aware augmented data for outdoor settings. Most current approaches to synthetic data generation focus on realistic object appearance through improved rendering techniques. However, we show that where and how objects are positioned is just as crucial for training effective 3D monocular detectors. The key obstacle lies in automatically determining realistic object placement parameters - including position, dimensions, and directional alignment when introducing synthetic objects into actual scenes. To address this, we introduce MonoPlace3D, a novel system that considers the 3D scene content to create realistic augmentations. Specifically, given a background scene, MonoPlace3D learns a distribution over plausible 3D bounding boxes. Subsequently, we render realistic objects and place them according to the locations sampled from the learned distribution. Our comprehensive evaluation on two standard datasets KITTI and NuScenes, demonstrates that MonoPlace3D significantly improves the accuracy of multiple existing monocular 3D detectors while being highly data efficient. project page

1. Introduction

Monocular 3D object detection has rapidly progressed recently, enabling its use in autonomous navigation and robotics [18, 32]. However, the performance of 3D detectors relies heavily on the quantity and quality of the training dataset. Given the considerable effort and time required to curate extensive, real-world 3D-annotated datasets, specialized data augmentation for 3D object detection has emerged as a promising direction.

However designing realistic augmentations for 3D tasks, is non-trivial, as the generated augmentations must adhere to the physical constraints of the real world, such as maintaining 3D geometric consistency and handling collisions.

[*]equal contribution

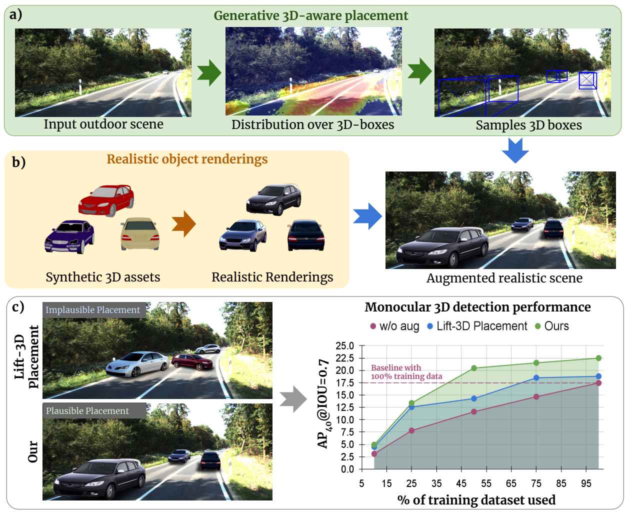
a)
Generative 3D-aware placement
Input outdoor scene
Distribution over 3D-boxes
Samples 3D boxes
b)
Realistic object renderings
Synthetic 3D assets
Realistic Renderings
Augmented realistic scene
c)
Implausible Placement
Lift-3D Placement
Plausible Placement
Our
Monocular 3D detection performance
● w/o aug ● Lift-3D Placement ● Ours
Baseline with 100% training data
AP40@IOU=0.7
% of training dataset used

Figure 1. a) We compare augmentations from our learned placement with heuristic-based placements from Lift3D [22]. In our augmentations, vehicles follow the lane orientations and are placed appropriately. b) These realistic augmentations significantly improve the 3D detection performance (KITTI [6] val set, (easy)). Notably, we achieve detection performance comparable to that of the fully labeled dataset using only 50% of the dataset.

Existing techniques [14, 25] for 3D augmentation use relatively simple heuristics for placing synthetic objects in an input scene. For instance, in the context of road scenes, a recent approach [22] generates realistic cars and places them on the segmented road region. However, such heuristics result in highly unnatural scene augmentations (Fig. 1), resulting in a marginal improvement in 3D detection performance. In this work, we ask the following two crucial questions: (1) *What key factors are essential for generating realistic augmentations to improve monocular 3D object detection?*, and (2) *How can these factors be integrated to generate effective scene-aware augmentations?*

For the first question, we discover two *critical factors* responsible for generating effective 3D augmentations:
1. Object Placement: Plausible placement of augmented objects, with appropriate *object placement (location, scale,*

and orientation), is essential for rendering realistic scene augmentations. For instance, in road scenes, a car should be placed on the road, be of appropriate size based on the distance from the camera, and follow the lane orientation. Augmentations that respect such physical constraints generalize better to real scenes by faithfully modelling the true distribution of the vehicles in the real world. To give an example of how such an augmentation looks, we compare our proposed augmentation approach against heuristic-based placement from Lift3D [22] in Fig. 1. Given the same rendering, our generation looks much more plausible regarding car placement and orientation compared to the baseline approach. Notably, when used for object detection training, our approach leads to significantly greater performance improvement, making the detector not only *performant*, but also *highly data efficient* (refer Fig. 1c)

2. Object Appearance: For 3D augmentation, it is desired that the generated objects exhibit realism and seamlessly integrate with the background to preserve visual consistency. This, in turn, minimizes the domain disparity between real and augmented data. Existing augmentation methods for 3D detection [14, 22, 25] primarily focus on the object appearance. This limits their ability to exploit the full potential of the data augmentations for 3D detection.

To address both these factors, we propose **MonoPlace3D**, a novel *scene-aware* augmentation method that generates effective 3D augmentations, as shown in Fig. 1. For plausible object placement, we train a 3D Scene-Aware Placement Network (**SA-PlaceNet**), which maps a given scene image to a distribution of plausible 3D bounding boxes. It learns realistic object placements that adhere to the physical rules of road scenes, facilitating sampling of diverse and plausible 3D bounding boxes (see Fig. 1a). For training this network, we consider existing 3D detection datasets, which typically contain only a limited number of objects per scene, resulting in a *sparse* training signal. Therefore, to enable *dense* placement prediction, we introduce novel modules based on (1) geometric augmentations of 3D boxes, along with (2) modeling of a continuous distribution of 3D boxes.

For realistic object appearance, we propose a rendering pipeline that leverages synthetic 3D assets and an image-to-image translation model. We translate the synthetic renderings into a realistic version using ControlNet [53](see Fig. 1b) and blend them with the background to get final augmentations. This allows us to utilize amateur-quality 3D assets and transform them into diverse, highly realistic car renderings that resemble real-world scenes.

Our two-stage augmentation approach is *highly effective and modular*, allowing seamless integration with advancements in placement and rendering for enhancing 3D object detection datasets. Using our augmentation method on popular 3D detection datasets led to significant improvements over the prior baselines and set a new state-of-the-art monocular detection benchmark. Notably, as shown in Figure 1, using only 40% of the real training data and our 3D augmentations outperforms a model that is trained on the complete data without any 3D augmentations. Through extensive ablation studies, we thoroughly analyze the role of different components and their effect on detection performance. We summarize our contributions below:

1. We identify the critical role of *3D-aware object placement* and *realistic appearance* for generating effective scene augmentations for 3D object detection.
2. We propose *MonoPlace3D*, a novel approach to generate plausible 3D augmentations for road scenes by realistically placing objects following scene grammar.
3. We demonstrate the effectiveness of the proposed augmentations on multiple 3D detection datasets and detector architectures with significant gains in performance as well as data efficiency.

2. Related Work

Object Placement. There are numerous works [1, 35, 49, 54, 60] which aim to predict object placement by learning a transformation or the bounding box parameters directly for a given background image. A set of works [35, 49] learns the distribution of indoor synthetic objects. Another set of works [20, 21, 34, 44, 54] learns the plausible locations for humans and other outdoor objects in a 2D manner. Few works aim to learn the arrangement conditioned on the scene-graph [19, 31, 52]. Similarly, another set of works [21, 44, 54] train a deep network adversarially in order to learn plausible 2D bounding box locations. Similarly, ST-GAN [26] learns to predict the geometric transformation of a bounding box in the given scene using adversarial training. [24] uses a variational autoencoder to predict a plausible location heatmap over the scene but is limited to placement in restricted indoor environments.

Monocular Object Detection. The current monocular 3D detection methods can be grouped as image-based or pseudo-lidar-based. Image-based detectors [2, 27, 28, 33, 37, 41, 43, 47, 56] estimate the 3D bounding box information for an object from a single RGB image. Due to the lack of depth information, these methods rely on geometric consistency in order to predict the class and the location of the object. Some works [23, 28, 32] use the prediction of key points of 3D bounding boxes as an intermediate task in order to improve it's performance on 3D monocular detection. In this work, we aim to improve the performance of image-based monocular detection models since RGB images are the most commonly used modality and easy to acquire with low acquisition costs, unlike LIDAR and depth sensors.

Scene Data Augmentation. Multiple works use 2D data augmentation techniques to improve the performance of perception tasks [40]. However, these augmentations can-not be lifted directly to 3D without violating the geometric constraints. To alleviate this problem, a recent method augments the training dataset for the task of 3D monocular detection [9, 12, 22, 25, 45]. One approach is to copy-paste cars from an archived dataset by considering the effect of 3D scene geometry, such as the scale and pose of the car [25]. Another approach is to model a synthetic urban scene from real-world datasets [9]. On the contrary, Lift3D [22] learns an object-centric neural radiance field to generate realistic 3D cars with GAN-augmented views and [45] learns a radiance field for the full 3D scene. Another set of approaches fully generates realistic multi-view scenes with diffusion models for generating realistic scenes [11, 13, 48, 50]. All these methods use heuristics such as lane segments to place cars; however, we aim to learn the distribution over car locations, scale, and orientation from the real-world object detection dataset.

3. Method

In this section, we first explain why it's important to have specialized methods for creating realistic scene-based augmentations for 3D detection. Then, we delve into the details of our unique approach to 3D augmentation.

Insight-1: *Unlike the object-based augmentations suitable for broad image classification tasks, enhancing structured tasks as 3D object detection requires careful consideration of object-background and object-object interactions for generation of plausible scene-based augmentations.*

Remarks: Synthetic *object-based* augmentation for image classification typically involves placing objects on any suitable background. This method may not always respect the interaction between the object and the background, its impact on the classification task remains minimal. In contrast, for *scene-based* augmentation, which is crucial in tasks like 3D detection, the interactions between objects and backgrounds, as well as between objects, becomes pivotal. For example, implausible placements such as a car in a sky background, two cars occluding each other's 3D volume, or a car-oriented perpendicular to lanes on the road, need to be avoided. While one might argue that random placement could aid in a 3D object detection task by helping the model distinguish objects from the background, empirical evidence suggests otherwise. Hence, it's crucial to devise a placement-based augmentation method that respects the scene-prior, thereby instilling this understanding into the detector model during training.

Insight-2: *The distribution of augmented samples for a given real sample $\mathbf{x_r}$, denoted as $q(\mathbf{x_{aug}}|\mathbf{x_r})$, can be enhanced by better scene-prior modeling; this leads to augmented scenes that closely align with the real distribution, fostering a robust model that is resilient to failures and can achieve superior performance with fewer real samples.*

Remarks: The equation $q(\mathbf{x_{aug}}|\mathbf{x_r}) = q(\mathbf{x_{aug}}|\mathbf{z}, \mathbf{x_r})q(\mathbf{z}|\mathbf{x_r})$ represents the distribution of augmented samples for a given real sample $\mathbf{x_r}$. Here, $q(\mathbf{x}|\mathbf{z}, \mathbf{x_r})$ represents a pipeline that generates the augmented scene image upon applying an effective placement-based augmentation. Here, $q(\mathbf{z}|\mathbf{x_r})$ denotes the *scene-prior* related latent factor $\mathbf{z}$ given the real image. This factor can model the distribution of plausible location, orientation, and scale to place objects given the scene layout. Improved modeling of the *scene prior* ensures that the augmented scene closely matches the real distribution. Training with such augmentations imbues the model with a strong understanding of the *scene prior*, enhancing its robustness and reliability. We demonstrate that this strategy enables efficient training, yielding superior performance with fewer real samples compared to the baseline.

Approach overview. Our method for 3D augmentation consists of two stages. First, we train the placement model that maps a monocular RGB image to a distribution over plausible 3D bounding boxes (Sec. 3.1). Subsequently, we sample a set of 3D bounding boxes from this distribution to place cars. In the second stage, we render realistic cars following the sampled 3D bounding box and blend them with the background road scene. (Sec. 3.2).

3.1. Scene-aware Plausible 3D Placement

Realistic 3D placement in road scenes is extremely challenging due to the high diversity in the scene layouts and underlying *grammatical* rules of the road scenes (Sec.1). Existing methods use simple heuristic placement [22] based on the road segmentation unable to model these complexities and hence result in unnatural augmentations (Fig. 1). We propose a data-driven approach to learn the real-world placement distribution by training a **Scene-Aware Placement Network (SA-PlaceNet)**, that maps a given image to the distribution of plausible 3D bounding boxes.

Learning such a distribution requires dense supervision about object location, scale, and orientation for each 3D point in space. Having such a dense annotated real dataset is impractical and can only be generated in a controlled synthetic setting that does not generalize to the real world. Hence, we take an alternate approach to learn the 3D bounding box distribution from an existing 3D object detection dataset. Object detection datasets only provide information on where *cars are located* but not *where they could be*. To mitigate this, we inpaint the vehicles from the scene to generate a paired image dataset with/without the vehicles. However, detection datasets have only a few vehicles in each scene, which provides only *sparse* signals for plausible 3D bounding boxes. Directly training with such a dataset will lead to overfitting and the model learns the *sparse* point estimate of locations as each scene has only a few car locations in the ground truth. To truly learn the underlying

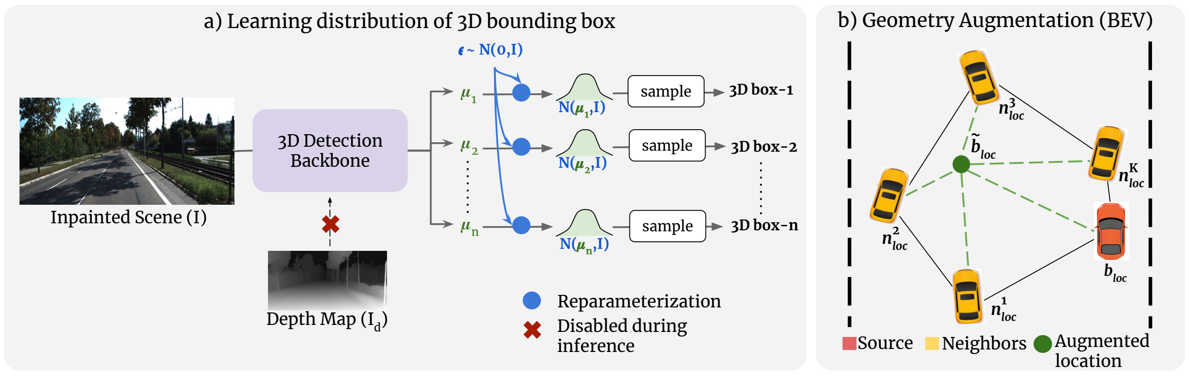

Figure 2. **a) SA-PlaceNet Architecture:** Given an input background image and corresponding depth to predict the means of a multi-dimensional Gaussian distribution over 3D bounding boxes. 3D bounding boxes are sampled from each of these Gaussian to compute the training loss. **b) Geometry-aware augmentation** in BEV (Birds Eye View). For a given source car location (b_{loc}), we first find K nearest neighbors with the same orientation and augment the location to $\tilde{b}_{loc}$ by interpolating with neighboring locations n_{loc} (Alg.1)

distribution of 3D bounding boxes, we propose two novel modules during training of placement network. **Geometry aware augmentation** and predicting *distribution over 3D bounding box* instead of a single estimate. The proposed modules enable diverse placements for a given scene that follow the underlying rules of the road scene.

The complete architecture for placement is shown in Fig. 2a. We build SA-PlaceNet using the backbone of MonoDTR [18]. MonoDTR is designed to perform monocular 3D object detection and is trained with auxiliary depth supervision. However, depth is not required during inference. We adapt the architecture of MonoDTR to learn the mapping from background road images $\mathcal{I}$ to a set of 3D bounding boxes $\mathcal{B}$. Following [18], we define bounding box $\mathbf{b} \in \mathcal{B}$ as 8 dimensional vector $\mathbf{b} = [b_x, b_y, b_z, b_h, b_w, b_l, b_\theta, b_\alpha]$, where (b_x, b_y, b_z) are 3D locations, (b_h, b_w, b_l) are height, width, and length of the box, and b_θ and b_α are orientation angles. Note that b_α can be computed deterministically from b_θ and hence we have only 7 variables defining a given bounding box. As a convention, we consider the xz plane as the road plane.

Dataset preparation. There is no existing real-world dataset that provides plausible placement annotations for a given road scene. Instead, we take advantage of the KITTI [15] dataset with 3D object detection annotations. We preprocess the dataset by inpainting the foreground cars in the scene using off-the-shelf inpainting [38]. Through this process, we obtain an image dataset ($\mathcal{I}$) with no cars on the road and a set of corresponding 3D bounding boxes ($\mathcal{B}$). Next, we obtain depth images $\mathcal{I}_d$ for the inpainted images using [36]. The obtained paired dataset, $\mathcal{D} = \{\mathcal{I}, \mathcal{I}_d, \mathcal{B}\}$, is used to train the SA-PlaceNet.

3.1.1. Geometry aware augmentation

Training SA-PlaceNet directly with the paired dataset $\mathcal{D}$ could easily learn a mapping to sparse 3D locations where

Algorithm 1 Geometry-aware augmentation procedure

```
1. Input:
   query box: $\mathbf{b} = [b_x, b_y, b_z, b_h, b_w, b_l, b_\theta, b_\alpha]$ where $b_{loc} = (b_x, b_y, b_z)$
   number of neighbors: $\mathbf{K}$
   radius of interpolation: $\mathbf{r}$
   amount of jitter: $\mathbf{d_j}$
   orientation threshold: $\epsilon_\theta$
2. Sample $K$ neighbors $\{n^i\}_1^K \in B$, s.t.
      $||n^i_{loc} - b_{loc}||_2 < r \quad \& \quad |n^i_\theta - b_\theta| < \epsilon_\theta$     (1)
3. If there are no neighbours i.e $K = 0$, then do
      $b_x \leftarrow b_x + d_x \quad b_z \leftarrow b_z + d_z$     (2)
    where $d_z > 2d_x$ and $d_x, d_z \in \mathcal{U}(0, d_j)$
   end If
4. Else do
   Generate the augmented location $\tilde{b}_{loc} = (\tilde{b}_x, \tilde{b}_y, \tilde{b}_z)$ using Eq. 7
   end Else
5. Output : Augmented bounding box parameters $\tilde{b}$ : $[\tilde{b}_x, \tilde{b}_y, \tilde{b}_z, b_h, b_w, b_l, b_\theta, b_\alpha]$
```

real cars were present before inpainting. Additionally, the model can cheat by using the inpainting artifacts to predict cars at the source location. To overcome these limitations, we propose *geometry-aware augmentation* $\mathcal{G}$ in the 3D bounding box space. We build on the intuition that the regions' neighboring ground truth car locations are also plausible for placement. The augmentation $\mathcal{G}$ transforms the ground truth bounding box $\mathbf{b} \in \mathcal{B}$ of a car, located at $\mathbf{b_{loc}} = (b_x, b_y, b_z)$ into a plausible neighboring box $\tilde{\mathbf{b}} = \mathcal{G}(\mathbf{b})$ located at $\tilde{\mathbf{b}}_{\mathbf{loc}} = (\tilde{b}_x, \tilde{b}_y, \tilde{b}_z)$ shown in Fig. 2b. The detailed algorithm for geometry-aware augmentation is given in detail in Alg.1. Specifically, we first find a set of K neighboring car boxes $\{n^i\}_{i=1}^{i=K}$ to the given car $\mathbf{b}$. We consider n^i as the neighbor of $\mathbf{b}$ if $||n^i_{loc} - \mathbf{b_{loc}}||_2 < r$ and $|n^i_\theta - b_\theta| < \epsilon_\theta$, for a given threshold r and ϵ_θ. We assume the selected K nearest cars will be in the same lane and

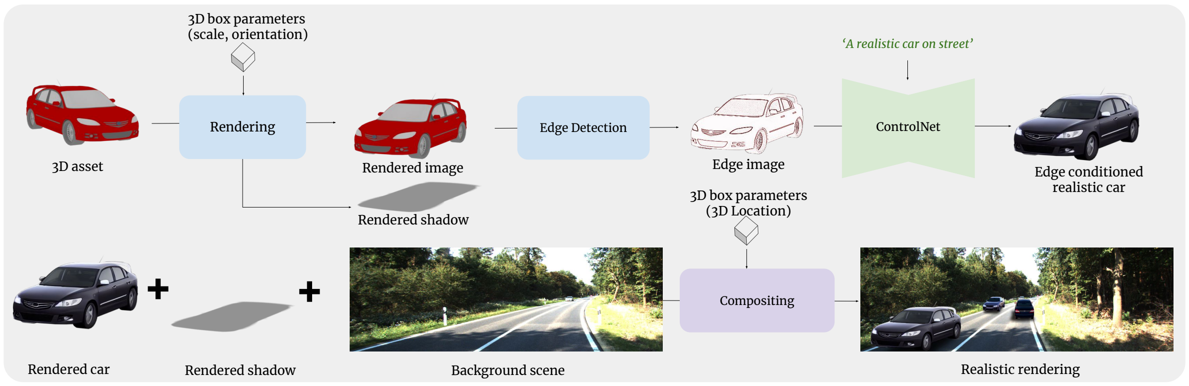

Figure 3. **Rendering pipeline:** Given a 3D asset, we first render an image and shadow from a fixed light source according to the 3D box parameters. Next, we used edge-conditioned ControlNet [53] to generate a realistic car version that follows the same orientation and scale as the rendered image. Finally, we use the obtained shadow, rendered car, and 3D location to place the car and render augmented images.

follow similar orientations. To augment the location $\mathbf{b_{loc}}$, we take a convex combination of neighboring locations n_{loc}^i and $\mathbf{b_{loc}}$ and obtain a location $\tilde{\mathbf{b}}_{\mathbf{loc}}$.

$$\tilde{b}_{loc} = \lambda_0 * b_{loc} + \sum_{i=1}^{k} \lambda_i * n_{loc}^i \tag{3}$$

where $\sum_i \lambda_i = 1$, $\lambda_i \geq 0$ are hyperparameters randomly sampled for each ground truth box $\mathbf{b}$. This transformation enables us to span a large region of plausible locations during training, hence enabling diverse placement locations during inference for each scene. If a car doesn't have any neighboring cars, we apply a uniform jitter along the length and a smaller jitter along the width of the car bounding box.

3.1.2. Distribution over 3D bounding boxes

Geometry-aware augmentation enables the generation of diverse placement locations, but it learns a direct mapping from the input image to a point estimate of bounding boxes. To learn a continuous representation in the output space, we map the input image to the distribution of 3D boxes. This improves the coverage of plausible locations and enables diverse bounding box sampling from a predicted set of mean boxes. Specifically, we approximate each predicted bounding box $\mathbf{b}$ as a multi-dimensional Gaussian distribution with mean μ_b and a fixed covariance matrix as αI, where α is used to control the spread as shown in Fig. 2a. We empirically observed that having a fixed covariance improves training stability. Having a higher α value results in strong augmentations, where the sampled car is far away from the mean location, resulting in a weaker training signal. During the forward pass, the SA-PlaceNet predicts mean bounding box parameters μ_b. To sample a box $\hat{\mathbf{b}}$, we first sample $\epsilon \in \mathcal{N}(\mathbf{0}, \mathbf{I})$ and use the reparametrization trick as follows:

$$\hat{\mathbf{b}} = \mu_b + \epsilon * \alpha \mathbf{I} \tag{4}$$

3.1.3. SA-PlaceNet Training

We train SA-PlaceNet with the acquired paired dataset $\mathcal{D} = \{\mathcal{I}, \mathcal{I}_d, \mathcal{B}\}$, consisting of inpainted background image ($\mathcal{I}$), inpainted depth image ($\mathcal{I}_d$) and the ground truth 3D bounding boxes ($\mathcal{B}$). Following [18], we train the model with $\mathcal{L}_{cls}$ for objectness and class scores, $\mathcal{L}_{dep}$ for depth supervision, and $\mathcal{L}_{reg}$ for bounding box regression.The proposed modules for *geometry-aware augmentation* and *learning distribution over 3D bounding boxes* can be easily integrated into a modified version of the regression loss $\mathcal{L}_{reg}^m$ as discussed below. The total loss is then defined as:

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{reg}^m + \mathcal{L}_{dep} \tag{5}$$

For a given ground-truth bounding box parameter $\mathbf{b}$, we first augment it using geometry-aware augmentation following Eq. (7) to obtain modified bounding box parameters $\tilde{\mathbf{b}} = \mathcal{G}(\mathbf{b})$. To capture the distribution of 3D boxes, we predict a mean bounding box parameter μ_b instead of a point estimate of the box parameters and randomly sample a new bounding box $\hat{\mathbf{b}}$ using the reparameterization trick outlined in Eq. (4). Subsequently, we compute the modified regression loss between the model prediction μ_b and the ground truth box $\mathbf{b}$ as follows:

$$\mathcal{L}_{reg}^m(\mu_b, \mathbf{b}) = \mathcal{L}_{reg}(\hat{\mathbf{b}}, \tilde{\mathbf{b}}) \tag{6}$$

3.2. What to place? Rendering cars

We generate realistic scenes by selecting cars and rendering them within the projected 3D coordinates of the predicted location, as shown in Fig. 3. To accurately render a car based on 3D bounding box parameters, we utilize 3D car assets from ShapeNet [5] that can be adjusted through orientation and scale transformations. Upon acquiring the 3D bounding box predictions, our rendering step entails sampling cars from the ShapeNet. Subsequently, the car model undergoes rotation according to the 3D observation angle of the object before positioning it within the designated scene. We separately render car shadows with predefined lighting in the rendering environment, following [7]. The rendered ShapeNet car images, although following the 3D bounding boxes, look unrealistic when pasted into the scene (Fig. 6,

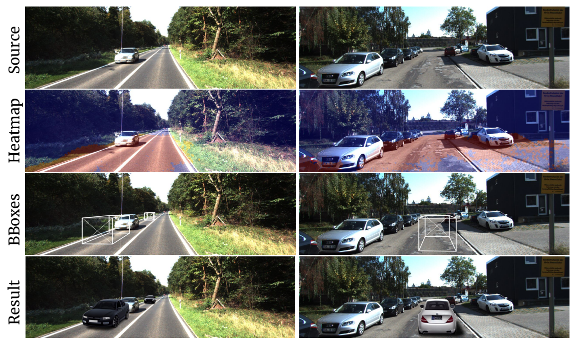

Figure 4. Given an input source image, we plot the heatmaps of the mean objectness score at each pixel location. The generated heatmaps span a large region on the road with plausible locations of objects. Next, we show samples of bounding boxes and realistic renderings of cars in the scene.

row-2). To resolve this, we leverage the advances in conditional generation using text-to-image models.

For the generated synthetic car images, we apply an edge detector to obtain an edge map. The edge map preserves the car's structure and still follows the same orientation and scale as the original car. Next, we use edge-conditioned text-to-image diffusion model ControlNet [53] to render a realistic car using the prompt *'A realistic car on the street.'* We further finetune the backbone diffusion model in ControlNet using LoRA [17] on a subset of 'car' images from the KITTI dataset. This enables us to generate natural-looking versions of cars that blend well with the background scene (Fig. 6). As ControlNet enables diverse generations from the same edge image, we can generate multiple renderings of cars from the edge map of a single ShapeNet car. This enables the generation of many diverse cars from a small, fixed set of 3D assets. The generated renderings look realistic and substantially boost object detection performance, as shown in Tab. We believe, the proposed approach of using a few 3D assets with conditional text-to-image models is promising and can be applied to generate diverse 3D augmentations for other tasks as well. Apart from the proposed rendering technique, we also experiment directly placing ShapeNet [5] and renderings from Lift3D [22], which is a generative radiance field approach that generates realistic 3D car assets.

4. Experiments

In this section, we present results for 3D-aware placement (Sec. 4.1) and car renderings (Sec. 4.2). Next, we present results for 3D detection trained with our generated augmentations (Sec. 4.3). We show additional results for monocular *3D detection on indoor* SUNRGBD [58] dataset, *2D detection on KITTI*, additional ablations, and quantitative analysis of SA-PlaceNet in the suppl.

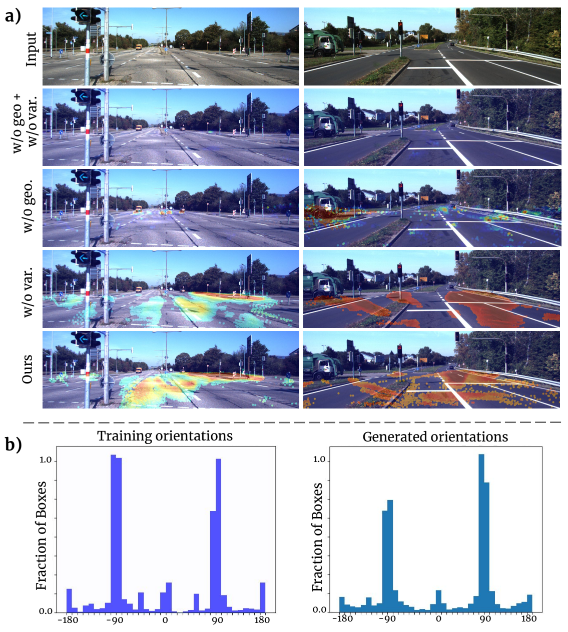

Figure 5. a) Ablation for object placement - For a background road scene, we visualize the heatmaps of aggregated objectness scores at each pixel location. Geometric augmentation and variational inference help to generate diverse and plausible object placements. b) Histogram of the distribution of orientations of the ground truth bounding boxes and the generated bounding boxes.

Dataset. We use the KITTI [15] and NuScenes [3] datasets for our experiments. KITTI consists of a total of 7481 real-world images captured from a camera mounted on a car. Following [6, 22, 46], we split the data into 3712 train and 3679 validation samples. For NuScenes, we use the official split with 700 train scenes containing 28134 samples and 150 validation scenes containing 6019 samples.

4.1. Evaluation of Placement Model

The placement network is trained with RGB images from the train split. We prepare the training data by inpainting the moving objects using [38] and obtain a paired dataset $\mathcal{D} = \{\mathcal{I}, \mathcal{I}_d, \mathcal{B}\}$ as detailed in Sec. 3.1. To visualize the performance of the placement, we generate heatmaps over the center of the bottom face of the bounding box in Fig. 4. For visualization, we use the mean objectness score of the anchor boxes corresponding to each grid cell. Geometry-aware augmentation enables learning of a large region for placing cars even though trained with input scenes with only a few cars. This allows for the sampling of diverse physically plausible placement locations for a given input scene shown as a set of 3D bounding boxes. We sample two sets of boxes from the predicted distribution. The sampled boxes have appropriate locations, scales, and orientations based on the background road. We present a detailed quantitative analysis of our method in the suppl. document.

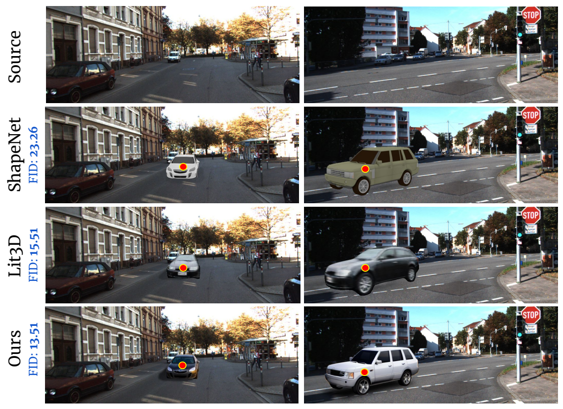

Figure 6. Ablation over rendering methods: Given the source image and predicted 3D bounding boxes, we sample and render a synthetic ShapeNet [5] car; Lift3D [22] rendered method; and our realistic rendering. We show a smaller domain gap between the rendered cars and the original samples.

Analysis. We analyze the impact of each component on placement performance in Fig. 5a). The naive baseline of directly training object placement without geometric augmentation and variational modeling only learns a point estimate and results in a few concentrated spots for placement location. Adding the variational head for learning a distribution of boxes instead expands the space of plausible locations but is still segregated in small regions. For the variational head, we have fixed the $alpha$ as 0.1. This highlights the sparse training signals for placement using ground truth boxes. However, when coupled with the geometry-aware augmentation, the predicted distribution covers a large driveable area on the road. To further analyze the orientations, we plot a histogram of predicted and the ground truth orientations in Fig. 5b), where the predictions closely follow the ground truth.

4.2. Evaluation of object renderings

We augment the road scenes by placing synthetic cars rendered by several approaches in Fig. 6. We compare the rendering quality of the proposed method with **1) ShapeNet** - 3D car assets renderings sampling from ShapeNet [5], **2) Lift3D** [22] - A generalized NeRF method for generating 3D car models. ShapeNet renderings result in unnatural augmentations due to synthetic car appearance and domain gaps from real scenes. On the other hand, Lift3D renderings, although realistic, lack diversity and suffer from artifacts. Our rendering method leverages conditional text-to-image diffusion models and generates extremely realistic cars that blend well with the background and are of high fidelity. Additionally, as our rendering starts from an underlying 3D asset, we use it to render shadows in a synthetic environment and copy the same shadow to the generated

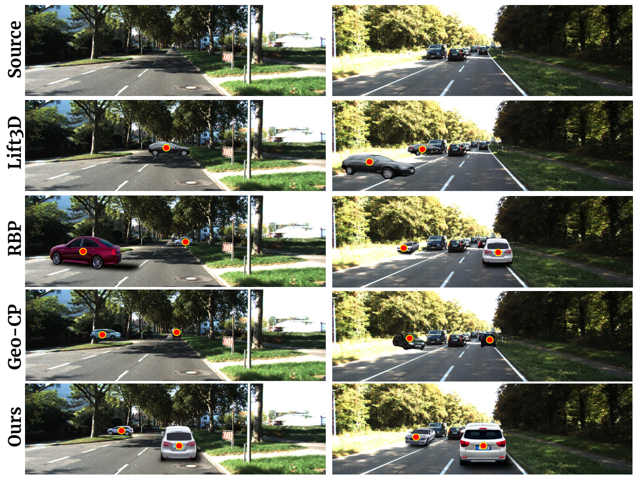

Figure 7. Qualitative comparison of the generated augmentations with all the baseline methods. Our augmentations are highly realistic, place cars following plausible placement properties, and have a minimal domain gap from the training dist.

realistic renderings. The proposed rendering pipeline effectively generates realistic augmentations and results in superior object detection performance (Tab. 1). Further, we report FID of the generated augmentations with the real training set to evaluate the realism.

4.3. Improving 3D Object Detection

We evaluate the effectiveness of our augmentations for monocular 3D object detection. We augment the training set with the same number of images to prepare an augmented version of the dataset. We compare our proposed augmentation method with the following augmentation approaches:

Geometric Copy-paste (Geo-CP) [25]. We use instance-level augmentation from [25], where cars from the training images are archived along with the corresponding 3D bounding boxes to create a dataset. For augmenting a scene, a car, and its 3D box parameters are sampled from the dataset, and the car is pasted in the background.

Lift-3D [22] proposed a generative radiance field network to synthetize realistic 3D cars. The generated cars are then placed on the road using a heuristic-based placement. Specifically, a placement location is sampled on the segmented road, and other 3D bounding box parameters are sampled from a predefined parameter distribution.

CARLA [10]. To compare the augmentations generated by simulated road scene environments, we use state-of-the-art CARLA simulator engine for rendering realistic scenes with multiple cars. It can generate diverse traffic scenarios that are implemented programmatically. However, it's extremely challenging for simulators to capture the true diversity from real-world road scenes and they often suffer

Table 1. Monocular 3D detection performance on KITTI dataset

a) MonoDLE[32]	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard
w/o 3D Augmentation	17.45	13.66	11.69	55.41	43.42	37.81
Geo-CP	17.52	14.60	12.57	58.95	44.23	38.66
CARLA	17.98	14.30	12.17	58.33	44.41	38.81
Lift3D	17.19	14.65	12.48	56.81	44.21	39.13
RBP	20.50	14.32	11.29	60.30	43.69	38.55
Ours	**22.49**	**15.44**	**12.89**	**63.59**	**45.59**	**40.35**

b) GUPNet[30]	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard
w/o 3D Augmentation	22.76	16.46	13.27	57.62	42.33	37.59
Geo-CP	21.81	15.65	13.24	59.12	44.03	39.16
CARLA	22.50	16.17	13.61	59.89	43.52	38.22
Lift3D	19.05	14.84	12.64	57.50	43.81	39.22
RBP	21.67	14.56	11.23	60.40	43.25	36.95
Ours	**23.94**	**17.28**	**14.71**	**61.01**	**47.18**	**41.48**

from a large sim2real gap.

Rule Based Placement (RBP). We create a strong rule-based baseline to show the effectiveness of our learning-based placement. Specifically, we first segment out the road region with [16] and sample placement locations in this region. To get a plausible orientation, we copy the orientation of the closest car in the scene, assuming neighboring cars follow the same orientations. We used our our rendering pipeline to generate realistic augmentations.

Qualitative comparison of generated augmentations are shown in Fig. 7. Lift3D augmentations have cars placed in incorrect orientation as the orientation is sampled from a general predefined distribution. RBP and Geo-CP augmentations are relatively better in terms of orientation but fail to place cars in the correct lanes. The proposed augmentation method follows the underlying grammar of the road well and generates realistic scene augmentations.

4.3.1. Realistic augmentations improves 3D detection

We evaluate our augmentation technique on two state-of-the-art monocular 3D detection networks - MonoDLE [32] and GUPNet [30] in Tab. 1 on KITTI [15] dataset. We generate one augmentation per real image for all the baselines. All the augmentation techniques improve over the baseline for MonoDLE. However, gains from Lift3D, CARLA, and Geo-CP are marginal. RBP performs better than other baselines primarily due to our realistic renderings. For GUPNet, none of the baselines can improve the detection performance overall. Our method significantly improves the score detection scores for both networks. This indicates a strong generalization of our augmentations on various 3D object detection models. We also show results on the current state-of-the-art MonoDETR [55] in the suppl. document.

Table 2. Rendering ablation with fixed placement

Rendering	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard
w/o 3D Augmentation	17.45	13.66	11.69	55.41	43.42	37.81
ShapeNet	20.91	14.17	12.28	59.54	43.48	37.64
Lift3D	21.35	14.25	11.65	60.38	42.65	37.53
Ours (w/o shadow)	21.45	14.21	11.73	61.23	43.27	38.28
Ours	**22.49**	**15.44**	**12.89**	**63.59**	**45.59**	**40.35**

4.3.2. Impact of object rendering on 3D detection

Table 2 presents an ablation study of various rendering approaches for augmentation in 3D detection. All renderings, when used with our learned placement, significantly outperform the baselines, demonstrating their compatibility with any rendering method. ShapeNet shows the lowest performance due to limited synthetic car diversity and a substantial sim2real gap. Lift3D rendering performs better than ShapeNet but exhibits noticeable artifacts when cars are close to the camera (Fig. 6). Our rendering approach, which uses a generative text-to-image model, outperforms all baselines but also enhances and achieves state-of-the-art performance when combined with shadows.

4.3.3. Augmenting other object categories

Though the car is the major category in the road 3D detection benchmarks, we also perform augmentation for two additional categories of cyclists and pedestrians, given they occur at 3.79% and 11.39% in the KITTI training set. For simplicity, we integrate our placement method with copy-paste rendering as described in the suppl. document (similar to Geo-CP [25]). Note that we trained another placement model to predict the placement of all the classes together. We use the augmented dataset with renderings of cyclists and pedestrians to train MonoDLE [32] object detector. The results are shown in Tab. 3; our augmentation significantly improves the detection performance of both categories over the baselines. We show qualitative results for these classes with copy-paste in the suppl. document.

Table 3. Augmenting multiple categories for 3D detection

Cyclist	3D@IOU=0.50			3D@IOU=0.25		
	Easy	Mod	Hard	Easy	Mod	Hard
w/o 3D Augmentation	4.92	2.03	1.85	18.41	10.82	9.52
Ours	**6.75**	**3.41**	**3.37**	**21.59**	**11.23**	**9.90**

Pedestrian	3D@IOU=0.50			3D@IOU=0.25		
	Easy	Mod	Hard	Easy	Mod	Hard
w/o 3D Augmentation	4.60	3.81	2.99	22.98	18.38	15.12
Ours	**4.98**	**3.89**	**3.34**	**26.28**	**20.81**	**16.16**

4.4. Experiments on large datasets

Table 4. Detection on NuScenes

FCOS3D [3]	MAP	NDS
w/o 3D Augmentation	0.3430	0.415
Lift3D	0.3211	0.371
Ours	**0.3704**	**0.440**

We validate the generalization of our method by training SA-PlaceNet on a large driving dataset - NuScenes [3]. Our approach produces plausible realistic augmentations for the given scene (suppl.) and we show improved performance on the NuScenes dataset with the FCOS3D [3] monocular detection network in Tab. 4.

4.5. Computational Cost of MonoPlace3D

Training SA-PlaceNet for generating augmentation takes a fraction of the time of the overall detection training. Specifically, on the KITTI dataset, SA-PlaceNet takes 12 hours vs 20 hours for a 3D detector (GUPNet) on a single A5000

GPU. Similarly, on a larger nuScenes dataset, SA-PlaceNet takes 32 hours vs 5 days for a 3D detector (FCOS3D). We have provided additional details for computational requirements across configurations in suppl. document.

5. Conclusion

This work proposes a novel scene-aware augmentation technique to improve outdoor monocular 3D detectors. The core of our method is an object placement network, that learns the distribution of physically plausible object placement for background road scenes from a single image. We utilize this information to generate realistic augmentations by placing cars on the road scenes with geometric consistency. Our results with scene-aware augmentation on monocular 3D object detectors suggest that realistic placement is the key to substantially improving the augmentation quality and data efficiency of the detector. The primary limitation of our approach is the dependency on the off-the-shelf inpainting method for data preparation for the training of the placement network. Also, our current framework does not consider more nuanced appearance factors in augmentations such as the lighting of the scene. In conclusion, we provide important insights for designing effective scene-based augmentations to improve monocular 3D object detection.

Acknowledgements. We thank Tejan Karmali for their helpful comments and discussions, and Abhijnya Bhat for reviewing the draft. Rishubh Parihar is supported by PMRF from the Government of India.

References

[1] Diego Martin Arroyo, Janis Postels, and Federico Tombari. Variational transformer networks for layout generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13642–13652, 2021. 2

[2] Garrick Brazil and Xiaoming Liu. M3D-RPN: monocular 3d region proposal network for object detection. *CoRR*, abs/1907.06038, 2019. 2

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *CoRR*, abs/1903.11027, 2019. 6, 8, 12, 14

[4] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. 17

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5, 6, 7, 16, 17

[6] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, page 424–432, Cambridge, MA, USA, 2015. MIT Press. 1, 6, 14

[7] Yun Chen, Frieda Rong, Shivam Duggal, Shenlong Wang, Xinchen Yan, Sivabalan Manivasagam, Shangjie Xue, Ersin Yumer, and Raquel Urtasun. Geosim: Realistic video simulation via geometry-aware composition for self-driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7230–7240, 2021. 5

[8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 12, 16, 17

[9] Shubham Dokania, Anbumani Subramanian, Manmohan Chandraker, and CV Jawahar. Trove: Transforming road scene datasets into photorealistic virtual environments. In *European Conference on Computer Vision*, pages 592–608. Springer, 2022. 3

[10] 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. 7, 15

[11] Ruiyuan Gao, Kai Chen, Enze Xie, HONG Lanqing, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. Magicdrive: Street view generation with diverse 3d geometry control. In *The Twelfth International Conference on Learning Representations*. 3

[12] Ruiyuan Gao, Kai Chen, Enze Xie, Lanqing Hong, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. Magicdrive: Street view generation with diverse 3d geometry control. *arXiv preprint arXiv:2310.02601*, 2023. 3

[13] Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhenguo Li, and Qiang Xu. Magicdrive3d: Controllable 3d generation for any-view rendering in street scenes. *arXiv preprint arXiv:2405.14475*, 2024. 3

[14] Yunhao Ge, Hong-Xing Yu, Cheng Zhao, Yuliang Guo, Xinyu Huang, Liu Ren, Laurent Itti, and Jiajun Wu. 3d copy-paste: Physically plausible object insertion for monocular 3d detection. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2

[15] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 4, 6, 8, 15

[16] Cheng Han, Qichao Zhao, Shuyi Zhang, Yinzi Chen, Zhenlin Zhang, and Jinwei Yuan. Yolopv2: Better, faster, stronger for panoptic driving perception. *arXiv preprint arXiv:2208.11434*, 2022. 8, 12, 15

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 6, 17

[18] Kuan-Chih Huang, Tsung-Han Wu, Hung-Ting Su, and Winston H Hsu. Monodtr: Monocular 3d object detection with depth-aware transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4012–4021, 2022. 1, 4, 5

[19] Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. Layoutvae: Stochastic scene layout generation from a label set. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9895–9904, 2019. 2

[20] Sumith Kulal, Tim Brooks, Alex Aiken, Jiajun Wu, Jimei Yang, Jingwan Lu, Alexei A Efros, and Krishna Kumar Singh. Putting people in their place: Affordance-aware human insertion into scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17089–17099, 2023. 2

[21] Donghoon Lee, Sifei Liu, Jinwei Gu, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Context-aware synthesis and placement of object instances. *Advances in neural information processing systems*, 31, 2018. 2

[22] Leheng Li, Qing Lian, Luozhou Wang, Ningning Ma, and Ying-Cong Chen. Lift3d: Synthesize 3d training data by lifting 2d gan to 3d generative radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 332–341, 2023. 1, 2, 3, 6, 7, 15, 16

[23] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. RTM3D: real-time monocular 3d detection from object keypoints for autonomous driving. *CoRR*, abs/2001.03343, 2020. 2

[24] Xueting Li, Sifei Liu, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Putting humans in a scene: Learning affordance in 3d indoor environments. *CoRR*, abs/1903.05690, 2019. 2

[25] Qing Lian, Botao Ye, Ruijia Xu, Weilong Yao, and Tong Zhang. Exploring geometric consistency for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1685–1694, 2022. 1, 2, 3, 7, 8

[26] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. ST-GAN: spatial transformer generative adversarial networks for image compositing. *CoRR*, abs/1803.01837, 2018. 2

[27] Yuxuan Liu, Yixuan Yuan, and Ming Liu. Ground-aware monocular 3d object detection for autonomous driving. *CoRR*, abs/2102.00690, 2021. 2

[28] Zechen Liu, Zizhang Wu, and Roland Tóth. SMOKE: single-stage monocular 3d object detection via keypoint estimation. *CoRR*, abs/2002.10111, 2020. 2

[29] Hannan Lu, Xiaohe Wu, Shudong Wang, Xiameng Qin, Xinyu Zhang, Junyu Han, Wangmeng Zuo, and Ji Tao. Seeing beyond views: Multi-view driving scene video generation with holistic attention. *arXiv preprint arXiv:2412.03520*, 2024. 13

[30] Yan Lu, Xinzhu Ma, Lei Yang, Tianzhu Zhang, Yating Liu, Qi Chu, Junjie Yan, and Wanli Ouyang. Geometry uncertainty projection network for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3111–3121, 2021. 8, 14

[31] Andrew Luo, Zhoutong Zhang, Jiajun Wu, and Joshua B Tenenbaum. End-to-end optimization of scene layout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3754–3763, 2020. 2

[32] Xinzhu Ma, Yinmin Zhang, Dan Xu, Dongzhan Zhou, Shuai Yi, Haojie Li, and Wanli Ouyang. Delving into localization errors for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4721–4730, 2021. 1, 2, 8, 14, 15

[33] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. *CoRR*, abs/1612.00496, 2016. 2

[34] Rishubh Parihar, Harsh Gupta, Sachidanand VS, and R Venkatesh Babu. Text2place: Affordance-aware text guided human placement. In *European Conference on Computer Vision*, pages 57–77. Springer, 2024. 2

[35] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. *Advances in Neural Information Processing Systems*, 34:12013–12026, 2021. 2

[36] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 4

[37] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *CoRR*, abs/1811.08188, 2018. 2

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 4, 6, 15

[39] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2397–2406, 2022. 13

[40] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48, 2019. 2

[41] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. *CoRR*, abs/1905.12365, 2019. 2

[42] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. *CoRR*, abs/1905.12365, 2019. 13

[43] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Elisa Ricci, and Peter Kontschieder. Single-stage monocular 3d object detection with virtual cameras. *CoRR*, abs/1912.08035, 2019. 2

[44] Jin Sun, Hadar Averbuch-Elor, Qianqian Wang, and Noah Snavely. Hidden footprints: Learning contextual walkability from 3d human trails. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 192–207. Springer, 2020. 2

[45] Wenwen Tong, Jiangwei Xie, Tianyu Li, Hanming Deng, Xiangwei Geng, Ruoyi Zhou, Dingchen Yang, Bo Dai, Lewei

Lu, and Hongyang Li. 3d data augmentation for driving scenes on camera, 2023. 3

[46] Wenwen Tong, Jiangwei Xie, Tianyu Li, Hanming Deng, Xiangwei Geng, Ruoyi Zhou, Dingchen Yang, Bo Dai, Lewei Lu, and Hongyang Li. 3d data augmentation for driving scenes on camera. *arXiv preprint arXiv:2303.10340*, 2023. 6

[47] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. *CoRR*, abs/2107.14160, 2021. 2

[48] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European Conference on Computer Vision*, pages 55–72. Springer, 2024. 3

[49] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19037–19047, 2023. 2

[50] Yuqing Wen, Yucheng Zhao, Yingfei Liu, Fan Jia, Yanhui Wang, Chong Luo, Chi Zhang, Tiancai Wang, Xiaoyan Sun, and Xiangyu Zhang. Panacea: Panoramic and controllable video generation for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6902–6912, 2024. 3

[51] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 15

[52] Cheng-Fu Yang, Wan-Cyuan Fan, Fu-En Yang, and Yu-Chiang Frank Wang. Layouttransformer: Scene layout generation with conceptual and spatial diversity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3732–3741, 2021. 2

[53] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 2, 5, 6, 16, 17

[54] Lingzhi Zhang, Tarmily Wen, Jie Min, Jiancong Wang, David Han, and Jianbo Shi. *Learning Object Placement by Inpainting for Compositional Data Augmentation*, pages 566–581. 2020. 2

[55] Renrui Zhang, Han Qiu, Tai Wang, Xuanzhuo Xu, Ziyu Guo, Yu Qiao, Peng Gao, and Hongsheng Li. Monodetr: Depth-guided transformer for monocular 3d object detection. *ICCV 2023*, 2022. 8, 12, 13, 14

[56] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. *CoRR*, abs/2104.02323, 2021. 2

[57] Hanqing Zhao, Dianmo Sheng, Jianmin Bao, Dongdong Chen, Dong Chen, Fang Wen, Lu Yuan, Ce Liu, Wenbo Zhou, Qi Chu, Weiming Zhang, and Nenghai Yu. X-paste: Revisiting scalable copy-paste for instance segmentation using clip and stablediffusion. In *International Conference on Machine Learning*, 2023. 14

[58] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. *Advances in neural information processing systems*, 27, 2014. 6, 13

[59] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, abs/1904.07850, 2019. 13

[60] Sijie Zhu, Zhe Lin, Scott Cohen, Jason Kuen, Zhifei Zhang, and Chen Chen. Topnet: Transformer-based object placement network for image compositing, 2023. 2

MonoPlace3D: Learning 3D-Aware Object Placement for 3D Monocular Detection

Supplementary Material

Contents

A. Additional placement results

A.1. Quantitative evaluation

To quantify the performance of placement, we compute the following three metrics on the training set of KITTI: **1) Overlap:** As road regions can cover most of the plausible locations for cars, we evaluate the predicted location by checking whether the *center of the base of the 3D bounding box* is on the road. Specifically, we compute the fraction of boxes that overlap with the road segmentation obtained using [16]. **2)** $\theta_{\mathbf{KL}}$: We evaluate the KL-divergence between the distribution of orientation of the predicted 3D bounding box and the ground truth boxes. We present quantitative results in Tab. 5, where our method achieves superior overlap scores, suggesting the superiority of placement.

Table 5. Ablation over SA-PlaceNet components

Method	Random	w/o var & geo	w/o geo	w/o var	Ours
Overlap ↑	0.20	0.15	0.17	0.35	**0.36**
θ_{KL} ↓	1.37	0.66	1.18	0.32	**0.30**

A.2. Placement on nuScenes [3] dataset

We validate the generalization of our method by training SA-PlaceNet on a subset of a recent driving dataset - NuScenes [3] in Fig. 8. We visualize predicted 3D bounding boxes and realistic renderings from our method. Our approach produces plausible placements and authentic augmentations for the given scene.

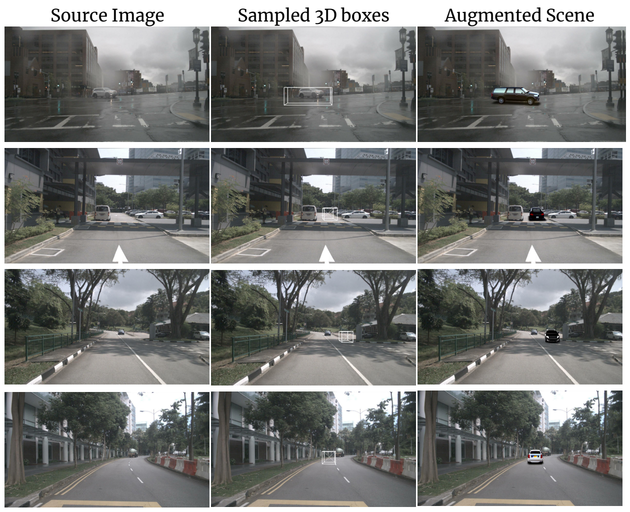

Figure 8. Placement on nuScenes [3] dataset.

A.3. Controlling traffic density in scenes

Our augmentation method enables us to control the traffic density of vehicles in the input scenes by controlling the number of bounding boxes to be sampled. We present results for generating low-density ($1-3$ cars added) and high-density ($3-5$ cars added) traffic scenes in Fig. 9.

A.4. Placing other categories

Our method enables us to learn placement for other categories from KITTI datasets. Specifically, we trained a joint placement model to learn the distribution of 3D bounding boxes for cars, pedestrians, and cyclists. To render the pedestrians and cyclists, we leverage simple copy-paste rendering as discussed in Sec. F.1. We present placement results in additional categories in Fig. 10. The pro-

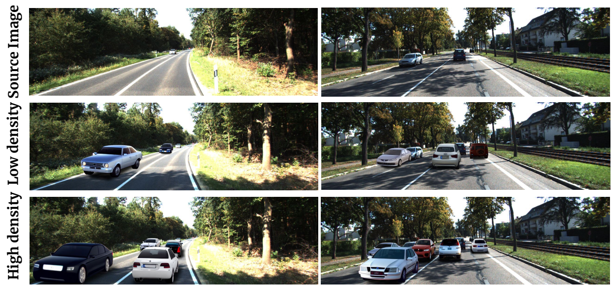

Figure 9. Augmented training dataset for 3D object detection: Given a sparse scene with few cars, we place cars at the predicted 3D bounding box locations using our rendering algorithm. We present two sets of results, one with low density ($1-3$ cars added) and another with high density ($4-5$ cars added) for each scene.

posed method predicts plausible locations, orientation, and shape of the object, enabling rich scene augmentations. Using these augmentations for training leads to significant improvement in performance for less frequent cyclist and pedestrian categories (Tab.3 main paper).

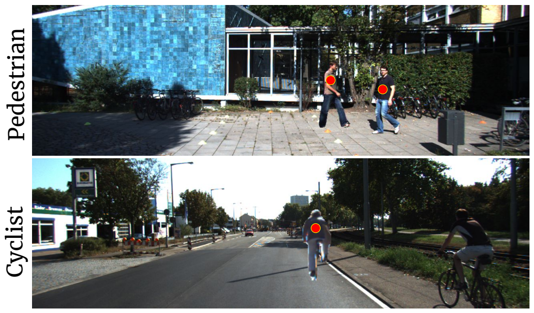

Figure 10. Placement results for pedestrian and cycle categories on KITTI dataset. Note that we applied copy-paste in the predicted 3D object box locations to generate the augmentations. Though copy-pasting causes image artifacts, these augmentations still improve 3D detection performance, as shown in the main paper.

B. Additional object detection results

B.1. Monocular 3D detection in indoor scenes

Our proposed method is generalizable for 3D detection in indoor environments. To demonstrate this, we performed a preliminary experiment involving monocular 3D detection on SunRGBD [58] dataset. We adapt our placement network building on an indoor detection network - ImVoxelNet [39]. We used copy-paste along with the predicted object locations to generate data augmentations. The generated augmentations are highly effective and improve upon the monocular 3D detection performance, as shown in Tab. 6. This indicates the superior generalization of our method for diverse environments. We believe a detailed exploration of our work for indoor environments is a promising future direction.

Table 6. Indoor 3D detection

config.	mAP@0.25
ImVoxelNet - w/o 3D augm.	0.410
ImVoxelNet - w ours	**0.430**

B.2. Improving 2D object detection

As our approach provides consistent 3D augmentations, it also enables to improve the performance of 2D object detectors. Specifically, our placement model also predicts the 2D bounding box along with the 3D bounding box (followed in most of the 3D detection works). We use these predicted 2D bounding box annotations to obtain a labeled 2D detection dataset. We evaluate the gains from our augmentations on 2D object detection on off-the-shelf 2D detector CenterNet [59] in Tab. 7. Following [42], we use a standardized approach to report AP_{40} metric instead of the AP_{11} for evaluation. Notably, our proposed augmentation method, though designed for 3D detection, can also improve the performance of 2D object detection, proving the task generalization of the proposed approach.

Table 7. 2D Detection Performance on '*Car*' category with CenterNet [59]

config.	AP2D@IOU=0.5		
	Easy	Mod.	Hard
w/o 3D Aug.	86.03	73.74	65.08
Ours	**89.56**	**76.79**	**72.28**

B.3. 3D object detection on BEV based detector

Our method generalizes to BEV-based detection, as our placement model predicts 3D bounding boxes in the world coordinate space. We train BEV-based **DeTR3D** on multi-view nuScenes, augmenting individual camera views by placing our ***2D car renderings*** in non-overlapping image regions. Since overlapping regions are mostly confined to the peripheries of adjacent camera views [29], our augmentations effectively improve detection performance (Tab. 8). For overlapping image regions, a possible solution is to use 3D cars and render consistent multi-views for placement.

Table 8. Detection on BEV based 3D detector DeTR3D

config.	NDS	mAP
Detr3D - w/o 3D augm.	0.434	0.349
Detr3D - w ours	**0.451**	**0.381**

B.4. 3D object detection on MonoDETR [55]

To validate the generalizability of our approach, we evaluate proposed 3D augmentation on a recent 3D monocular detection model MonoDETR [55] on the KITTI dataset in Tab. 9. We report the baseline results without our augmentation from the original paper. Our method consistently outperforms the baseline in all three settings. The comprehensive evaluation across several detectors (also in the main paper) evidently shows the generalization of our proposed

3D augmentation method.

Table 9. 3D Detection Performance on Car with MonoDETR [55]

MonoDETR	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard
w/o 3D Augmentation	28.84	20.61	16.38	68.86	48.92	43.57
Geo-CP	23.26	16.41	14.58	60.65	43.93	37.71
Lift3D	22.00	16.61	14.59	63.45	47.34	38.57
RBP	24.92	17.75	15.90	61.99	44.02	38.04
Ours	**29.90**	**21.91**	**16.85**	**69.63**	**49.10**	**43.63**

B.5. Effect of Poisson Blending

We use Poisson blending to enhance the quality of the composition of synthetic cars with the background scene. We observe a slight dip in the detection performance using the obtained augmentations as reported in Tab. 10. A similar observation was made in [57], where improved blending does not positively affect the detection performance.

Table 10. Monocular 3D detection performance of Poisson Blending on our Rendering on KITTI [6] validation set.

(a) MonoDLE[32] on Car with and without Poisson Blending

Rendering	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod	Hard
w/o 3D Aug.	17.45	13.66	11.69	55.41	43.42	37.81
Ours	**22.49**	**15.44**	**12.89**	**63.59**	**45.59**	**40.35**
Ours (+Poisson)	21.34	14.44	12.81	59.60	44.11	38.15

(b) GUPNet[30] on Car with and without Poisson Blending

Rendering	3D@IOU=0.7			3D@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod	Hard
w/o 3D Aug.	22.76	16.46	13.27	57.62	42.33	37.59
Ours	**23.94**	**17.28**	**14.71**	**61.01**	**47.18**	**41.48**
Ours (+Poisson)	22.43	17.03	14.55	60.00	45.28	39.60

Table 11. Analysis of Training Time

Model	Dataset	Training Time	#GPU's	GPU Model
SA-PlaceNet	KITTI	12h	1	A5000
SA-PlaceNet	NuScenes	32h	1	A5000
GUPNet	Original KITTI	20h	1	A5000
GUPNet	Augmented KITTI	22h	1	A5000
FCOS3D	Original NuScenes	5d18h	2	A5000
FCOS3D	Augmented NuScenes	6d	2	A5000

C. Computational cost of MonoPlace3D

Training of SA-PlaceNet takes a fraction of the time of the detection training. The relative training time is significantly reduced for large datasets such as NuScenes. We present the computational requirements of our augmentation in comparison to the training time in Table 11. We train GUPNet and MonoDLE for an additional 10 epochs and FCOS3D for an additional 5 epochs when training with our augmented data.

C.1. Data Efficiency on KITTI

In this section, we demonstrate the data efficiency of our method. As observed in Tab.12 our method can significantly reduce the dependence on real data when training monocular detection networks. Specifically, augmenting just 50 % of the real data can achieve better performance than training with 100 % of the original training data.

Table 12. Data efficiency of SA-PlaceNet on KITTI dataset

MonoDLE		3D@IOU=0.7			3D@IOU=0.5		
% Real Data	% Aug. Data	Easy	Mod.	Hard	Easy	Mod.	Hard
10	10	4.94	3.90	3.26	27.21	21.03	18.06
25	25	13.38	9.78	8.23	48.28	36.99	30.83
50	50	20.46	13.70	11.71	58.04	43.83	37.87
75	75	21.53	14.95	12.38	60.94	45.19	39.99
100	100	**22.49**	**15.44**	**12.89**	**63.59**	**45.59**	**40.35**
100	0	17.45	13.66	11.69	55.41	43.42	37.81

C.2. Scalability of generated augmentations

To evaluate the effectiveness of the scale of our augmentations, we perform a scalability experiment on a large nuScenes dataset consisting of $\approx 35K$ images. We use different fractions of real and augmented data to train a monocular 3D detector and achieve consistent gains across the amount of data.

Table 13. Scaling on NuScenes dataset

% Data	mAP (w/o aug)	mAP (ours)	NDS (w/o aug)	NDS (ours)
15	0.131	**0.151**	0.223	**0.239**
30	0.231	**0.253**	0.311	**0.339**
50	0.310	**0.342**	0.392	**0.411**
100	0.343	**0.371**	0.415	**0.440**

C.3. Rendering Ablation on NuScenes

We also present an ablation study of various rendering approaches for augmentation in 3D detection for NuScenes. All renderings, when used with our learned placement, outperform the baseline, demonstrating the compatibility of our placement with different rendering methods.

Table 14. Ablation on NuScenes

FCOS3D [3]	MAP	NDS
w/o 3D Augmentation	0.3430	0.415
ShapeNet	0.3441	0.414
Lift3D	0.3460	0.416
Ours	**0.3704**	**0.440**

D. Data Augmentation for Corner Cases

We aim to approximate the training data distribution $p(x)$, with a learned distribution $q_\theta(x)$, which can be sampled $(x \sim q_\theta(x))$ to generate augmentations. In principle, our approach can also model abnormal cases by learning a distribution to approximate the conditional distribution $p(x|state = \text{'}abnormal\text{'})$. During inference from SA-PlaceNet we sample the least likely positions from the learned distribution to simulate corners cases for autonomous driving . We augment the training data with these corner cases and train MonoDLE [32] . In Fig 11 we show qualitatively how training with our data can improve the model performance on corner cases .

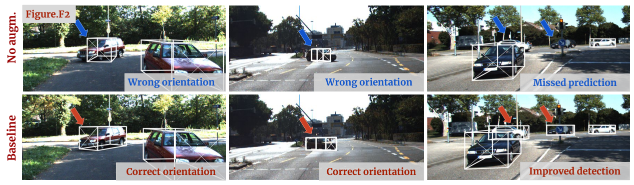

Figure 11. Detection improvement in corner cases.

E. Implementations details

E.1. Placement data Preprocessing

We use the state-of-the-art Image-to-Image Inpainting method [38] to remove vehicles and objects from the KITTI dataset [15]. The input prompt *'inpaint'* is passed to the inpainting pipeline. A few outputs from this method can be seen in Fig. 12

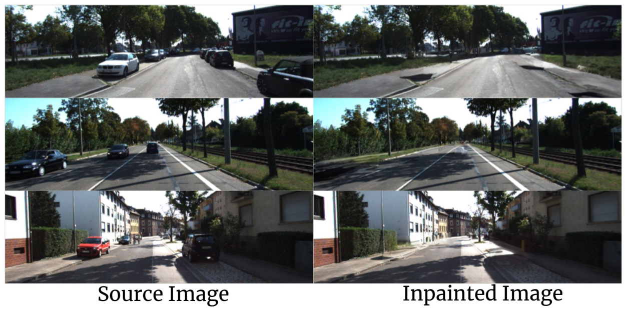

Figure 12. Outputs generated from Stable Diffusion Inpainting pipeline [38]. These inpainted images are used for training our placement model.

E.2. Baseline methods

Geometric Copy-paste (Geo-CP). To augment a given scene, a car is randomly sampled from the database, and its 3D parameters are altered before placement. Specifically, the depth of the box (z coordinate) is randomly sampled, and corresponding x and y are transformed using geometric operations. Other parameters, such as bounding box size and orientation, are kept unchanged. The sampled car is then pasted using simple blending on the background scene.
CARLA [10]. To compare the augmentations generated by simulated road scene environments, we use state-of-the-art CARLA simulator engine for rendering realistic scenes with multiple cars. It can generate diverse traffic scenarios that are implemented programmatically. However, it's extremely challenging for simulators to capture the true diversity from real-world road scenes and they often suffer from a large sim2real gap.
Rule Based Placement (RBP). We create a strong rule-based baseline to show the effectiveness of our learning-based placement. Specifically, we first segment out the road region with [16] and sample placement locations in this region. To get a plausible orientation, we copy the orientation of the closest car in the scene, assuming neighboring cars follow the same orientations. We used our proposed rendering pipeline to generate realistic augmentations.
Lift-3D [22] proposed a generative radiance field network to synthetize realistic 3D cars. Lift3D trains a conditional NeRF on multi-view car images generated by StyleGANs. However, the car shape is changed following the 3D bounding box dimensions. The generated cars are then placed on the road using a heuristic based on road segmentation. We used a single generated 3D car provided in the official code to augment the dataset as the training code is unavailable. Specifically, road region is segmented using off-the-shelf drivable area segmentor [16]. Next, the 3D bounding box of cars is sampled from a predefined distribution of box parameters as given in Tab.15, and the ones outside the drivable area are filtered out. For a sampled 3D bounding box parameters b=$[b_x, b_y, b_z, b_w, b_h, b_l, b_\theta]$, we render the car at adjusted orientation angle $\tilde{\theta}$ using Eq. 7. We place the camera at the fixed height of $1.6m$, with an elevation angle of 0. Also, we used (b_w, b_h, b_l) to render the car of a particular shape. We render the car image for 512x512 resolution using volume rendering and the defined camera parameters. Along with the RGB image, Lift3D also outputs the segmentation mask for the car which is used to blend it with the background. Fig. 13 shows some sample renderings from Lift3D.

Figure 13. Sampled views rendered from Lift3D [22].

F. Rendering details

F.1. Copy-Paste

In simple copy-paste rendering, the cars from the training corpus are added to the predicted 3D bounding boxes. We extract cars of various orientations from the training set images through instance segmentation using Detectron2 [51].

Table 15. Preset distribution of bounding boxes. Lift3D [22] samples bounding boxes from the predefined parameter distribution.

Pose	Distribution	Parameters
x	Uniform	$\{[-20m, 20m]\}$
y	Gaussian	$\mu = height, \sigma = 0.2$
z	Uniform	$\{[5m, 45m]\}$
l	Gaussian	$\mu = l_{\text{mean}}, \sigma = 0.5$
w	Gaussian	$\mu = w_{\text{mean}}, \sigma = 0.5$
h	Gaussian	$\mu = h_{\text{mean}}, \sigma = 0.5$
θ	Gaussian	$\mu = \pm\pi/2, \sigma = \pi/2$

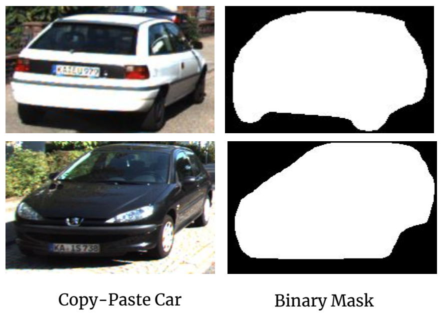

Figure 14. Sample cars from the Copy-Paste Database

These cars are archived in a database with their corresponding 3D orientation and binary segmentation mask data. During inference, given a 3D bounding box, we query and search for cars whose orientation closely aligns with the given 3D box orientation. A certain degree of randomness is introduced in selecting the nearest-matching car, contributing to increased diversity and seamless integration with the input scene. Next, we compose the retrieved car image onto the background scene using the 2D-coordinated obtained from the 3D bounding box and the binary mask. This simple rendering essentially captures the diverse cars present in the training dataset and helps in generating scenes that are close to training distribution. However, such rendering has a problem with shadows as the composition is not 3D-aware, given the placed cars are stored as images.

F.2. ShapeNet

ShapeNet [5] is a large-scale synthetic dataset that provides 3D models for various object categories, including cars. The ShapeNet Cars dataset focuses specifically on providing 3D models of different car models from various viewpoints. We leverage the high diversity of cars (nearly 7500 models) in the dataset and render the cars at the predicted box locations with 3D bounding box parameters using Blender [8] software. We employ a random sampling technique to select a 3D car model from this extensive dataset, which is then loaded in the Blender [8] environment. To ensure consistency in the car shapes, we initially calculated the average dimensions of the cars within the dataset. We exclude any car model with dimensions exceeding 50% of the computed average, and we repeat this random sampling procedure until the specified conditions are satisfied. Following that, we align and render the car by a 3D rotation angle. Specifically, as the orientation angle θ is defined in 3D, using it directly to render the image does not take care of perspective projection. Eg. all the cars following a lane will have similar orientation angles (close to zero) but look visually different when projected on the image as shown in Fig. 16. Both the rendered cars have 0 orientation angle in 3D but when projected onto the image planes, the rendered orientation changes with the location. To this end, we adjust the car orientation by a correction factor to incorporate the perspective view, as described in equation (7),

$$\tilde{\theta} = \theta + \tan^{-1}(\frac{x}{z}) \tag{7}$$

where x and z are the respective 3D coordinates of the bounding box. We use the final corrected $\tilde{\theta}$ value for rendering the ShapeNet car. We render car images at 512x512, with a white background, which can be later used as a segmentation mask to blend the rendered image. A few examples of the ShapeNet cars rendered with different orientations are visualized in Fig. 15.

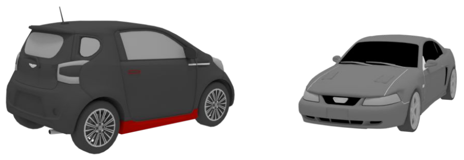
Figure 15. Sample of ShapeNet [5] cars rendered at different views.

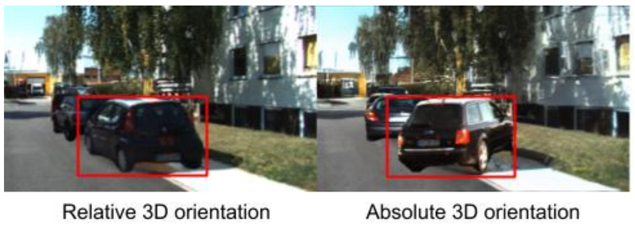

Figure 16. Perspective and Absolute projection of cars with the same 3D orientation.

F.3. Reaslistic rendering with Text-to-image models.

We leverage a state-of-the-art image-to-image translation method based on the powerful StableDiffusion model [53] to convert the synthetic ShapeNet renderings into realistic cars. We use edge-conditioned ControlNet [53], which takes an edge image and a text prompt to generate images following the edge map and the prompt. Specifically, we utilize a canny edge detector to create edge maps for synthetic car images rendered using ShapeNet [5], preserving the car's structure while maintaining its original orientation and scale. These edge maps, generated through the Canny Edge Detection algorithm [4], serve as input for the edge-conditioned ControlNet [53], enabling the rendering of realistic cars using the prompt *'A realistic car on the street'*. Furthermore, given an edge map and hence a ShapeNet-rendered car, we can obtain various realistic renderings at each iteration, facilitating diverse scene generations (Fig. 17). We further enhance ControlNet's backbone diffusion model using LoRA [17] on a subset of 'car' images from the KITTI dataset. This process enables the generation of natural-looking car versions that seamlessly blend with the background scene. Finally, we integrate the ControlNet-rendered car and its shadow base into the predicted location within the scene to achieve a realistic rendering.

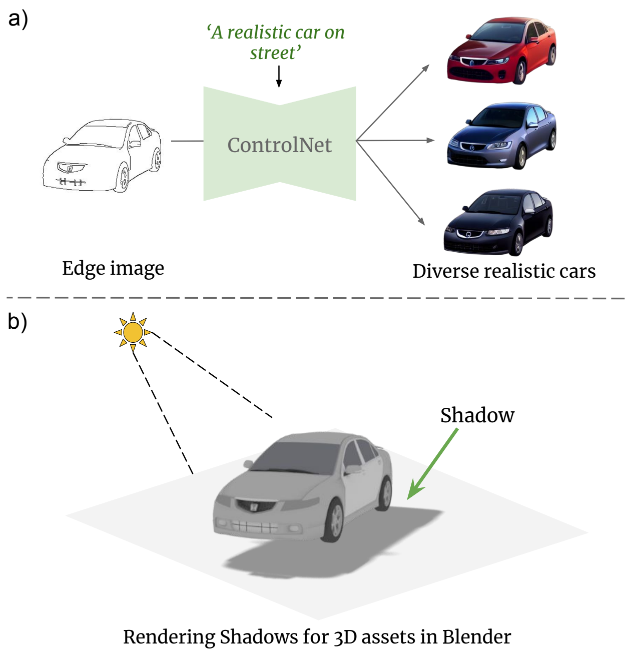
a)
'A realistic car on street'
ControlNet
Edge image
Diverse realistic cars
b)
Shadow
Rendering Shadows for 3D assets in Blender

Figure 17. a) Diverse renderings generated with edge-conditioned ControlNet. B) Shadows are generated by rendering 3D assets with a point light source in the blender [8] environment

F.4. Rendering shadows in Blender [8]

To generate a realistic composition of the augmented cars, we generate realistic shadows for cars using the ShapeNet [5] dataset and rendered with Blender. We modify the rendering method to generate shadows by introducing a 2D mesh plane beneath the car base and adding a uniform 'Sun' Light source along the z-axis of the blender environment, placed at the top on the z-axis of the car (Fig. 17). Additionally, we introduce slight variations across all axes for the light source position. Once the cars are positioned within the Blender [8] environment with suitable orientation, we render the entire scene while setting both the car and the 2D plane as transparent. This method enables us to create a collection of shadow renderings with a transparent background for each car in the placement setting.