IS-NEAR: Implicit Semantic Neural Engine and Multi-Sensor Data Rendering With 3D Global Feature

Tiecheng Sun[1]* Wei Zhang[2,3] Xingliang Dong[1] Tao Lin[1]
[1]Central Media Technology Institute, Huawei 2012 Laboratories
[2]University Of Stuttgart [3]Audiovisual Lab, Huawei Munich Research Center
{suntiecheng1,lintao879,dongxingliang}@huawei.com wei.zhang@ifp.uni-stuttgart.de

Abstract

Data-driven Computer Vision (CV) tasks are still limited by the amount of labeled data. Recently, some semantic NeRFs have been proposed to render and synthesize novel-view semantic labels. Although current NeRF methods achieve spatially consistent color and semantic rendering, the capability of the geometrical representation is limited. This problem is caused by the lack of global information among rays in the traditional NeRFs since they are trained with independent directional rays. To address this problem, we introduce the point-to-surface global feature into NeRF to associate all rays, which enables the single ray representation capability of global geometry. In particular, the relative distance of each sampled ray point to the learned global surfaces is calculated to weight the geometry density and semantic-color feature. We also carefully design the semantic loss and back-propagation function to solve the problems of unbalanced samples and the disturbance of implicit semantic field to geometric field. The experiments validate the 3D scene annotation capability with few feed labels. The quantification results show that our method outperforms the state-of-the-art works in efficiency, geometry, color and semantics on the public datasets. The proposed method is also applied to multiple tasks, such as indoor, outdoor, part segmentation labeling, texture re-rendering and robot simulation.

1. Introduction

The Computer Vision (CV) perception tasks have attracted widespread attention from academia and industry due to the amazing results. However, the breakthrough in CV is still limited by the availability of high-quality annotated data. The existing annotated data is mainly obtained by simulator [6, 16, 26], manually labeling or pseudo labels from perceptual algorithms. These methods have some problems such as domain gaps, low labeling efficiency and unsatisfying quality. Recently, some works are proposed for novel view synthesis, such as Neural Radiance Fields (NeRF) [19] for novel view color image rendering, and semantic NeRF [34] for annotation data rendering. These methods provide a new pipeline for data labeling.

Inspired by NeRFs, we prefer to construct an implicit data engine that can generate high-quality multi-sensor CV data such as images, semantic labels, depth and videos for different tasks training or evaluation by feeding posed images and a small amount of semantic labels. However, the existing methods such as the semantic NeRF [34] and its variants [3, 7, 12, 14, 15, 17, 21, 28, 33, 35, 35] still face some challenges to achieve this goal, *e.g.*, (i) compared to color, we find that dimensionality-reduced semantics interfere somewhat with geometry learning when we introduce semantics into NeRFs. Noisy geometry further degrades the quality of color and semantics. Thus, we prefer decoupling the geometry from the semantic field. (ii) In these NeRF methods, the independent single ray used for training lacks global information, resulting in dense semantics feeding for a complete and high-quality semantic scene training, increasing the labeling workload. (iii) The independent single ray also lacks global geometry information, so the current NeRF methods struggle to learn precise geometry.

In this paper, we propose an **I**mplicit **S**emantic **N**eural **E**ngine for multi-sensor d**A**ta **R**endering, named IS-NEAR. We are the first to introduce the 3D global features into NeRFs as a novel semantic and geometric representation and address the aforementioned challenging issues. Specifically, our implicit engine consists of four modules, namely feature, geometry, semantic, and color modules. To speed up the training and inference, we use the multi-resolution hash table proposed in instant Neural Graphics Primitives (instant NGP) [20] to learn and store spatial grid features in the feature module. These grid features are used for rendering and generating color, semantics, and geometry, ensuring spatial consistency among these elements. In traditional NeRFs, such as instant NGP [20], Mildenhall NeRF [19] or

*Corresponding author

semantic NeRF [34], for each ray, we can only collect the features from the sampling ray points or the grids that intersect the ray. This ray-wise feature is suitable for inferring the ray-wise-different color, but it is unreasonable to infer geometry or semantics that with local and global continuity. Instead, this ray-wise feature can easily lead to their discontinuity. To address this problem, we propose to learn global surfaces like the point-to-surface representation [25] in the geometry module to globally constrain the ray-wise geometry. We calculate the relative distances from each point on the ray to the global geometric surfaces to weight the density distribution and the collected features that used for color and semantics rendering. The experimental results show that the global features can improve geometry, color, semantics and efficiency at the same time.

In order to decouple the geometry from the semantic field, we customize back-propagation function that ignores the geometry gradient from semantics. In color module, we use the direction embedding and grid features to decode the color. While in the semantic module, we only use the weighted features to decode semantics without direction embedding, because semantics are direction-invariant. We also carefully design the cross entropy loss function with the truncated weights to solve the problem of unbalanced semantic samples. For sparse view scenarios, such as autonomous driving, we use depth or point cloud geometry priors to assist in constructing large-scale scene data engines. Experiments validate the effectiveness of the proposed method and show that our method outperforms the state-of-the-art (SOTA) algorithms. For example, the geometric RMSE is reduced from 0.096 m to 0.0122 m, the semantic mIoU is improved from 93.68% to 94.79%, the color PSNR is improved from 31.39 dB to 35.9 dB, and the training and rendering times are reduced from 8 h and 4.82 s to 15 min and 0.1 s, respectively. The proposed engine is also applied to indoor, outdoor and objects semantic labeling, texture re-rendering and data simulation. The contributions of this paper are summarized as follows:

1) To the best of our knowledge, IS-NEAR is the first to associate all independent NeRF rays with 3D global features, which improves the geometry precision and the quality of color and semantics. The global features also reduce the labels feeding and improve efficiency.
2) We propose to customize the back-propagation function to eliminate the differences between geometric and semantic inferences.
3) The carefully loss and network design makes a trade-off among each performance indicator of the engine, *e.g.*, the efficiency, semantics, color and geometry are superior to the SOTA methods.
4) Extensive experiments validate our engine and sample applications, including indoor, outdoor and objects semantic labeling, texture re-rendering, and robot simulation.

2. Related work

2.1. Implicit geometry and color fields

NeRF is first proposed by Mildenhall *et al.* [19] for view synthesis. They formulate the image rendering task as a differentiable optimization problem, achieving amazing results. The geometric information of a scene is also learned by renderer training process simultaneously. Therefore, more and more research [2, 8, 29, 32] focuses on image rendering and 3D scene reconstruction simultaneously with NeRF. Some studies [5, 13, 31] also show that geometric priors can improve the quality of rendering and assist the implicit fields construction with sparse views.

2.2. Implicit semantic field

Recently, scholars start to focus on the combination of implicit fields with geometry, color and semantics. For example, the Panoptic NeRF is proposed by Fu *et al.* [7] to render the spatially consistent 2D semantics for autonomous driving segmentation training. However, this approach relies on prior 3D bounding boxes and 2D pseudo-labels. Zhi *et al.* [34] proposed the Semantic NeRF that encodes appearance, geometry and semantics in one NeRF. It also works with sparse semantic labels. However, this method takes several hours for training, and some details are lost in the rendering images. To speed up labeling, Zhi *et al.* [35] proposed an online labeling method based on the implicit mapping method iMAP [24]. Similarly, Mazur *et al.* [17] propose a real-time feature fusion fields based on iMAP. These iMAP-based scene annotation methods achieve fast semantic scene reconstruction. However, the generated labeled data still cannot be used as truth value due to the limited mean Intersection over Union (mIoU).

Vora *et al.* [28] propose a two-step training method that first trains an original NeRF for extracting geometry representation. Then the parameters of the NeRF are held fixed and another semantic reasoning network is trained for semantics. The 3D network and two-step training processes make the method time-consuming and computational resource-consuming. There are some other methods [12, 15, 21] that use the pre-trained segmentation model to get the prior pseudo-labels for implicit semantic fields training. Although these methods have some performance improvements over semantic segmentation networks, the mIoU of the generated label is still low due to the noisy pseudo-labels.

These methods can effectively solve the problem of spatial inconsistency of the traditional 2D semantic segmentation. However, if the combination of multi-source implicit fields is used in a data engine, further trade-offs and improvements in time cost, image, color, and semantic quality are required. This paper proposes to use 3D global features to trade-off the quality of the implicit fields.

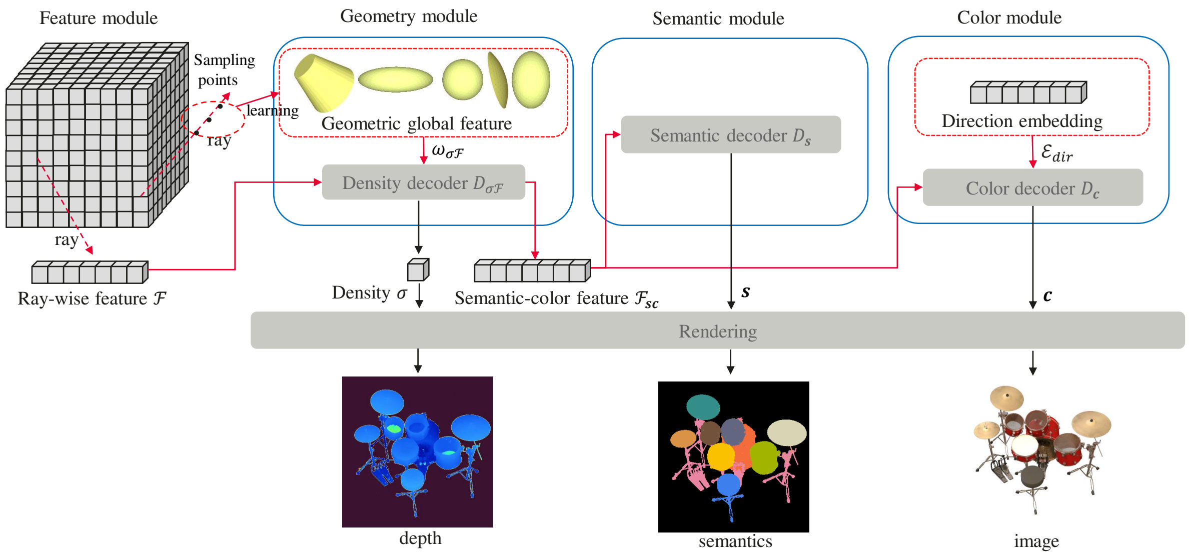

Figure 1. The proposed implicit engine includes four modules, *i.e.*, feature module, geometry module, semantic module and color module. Firstly, the hash features are looked up from our feature module, and then they are converted into density features. These features are subsequently weighted by the point-to-surface weights to obtain the semantic-color feature. Then, the semantic-color feature is fed into decoders to get color and semantic probability. Finally, the image, depth and semantics are obtained by volume rendering.

3. Method

Our goal is to construct an implicit engine with limited accessible data such as pose, image, semantics or depth. Then, we can generate more novel view multi-sensor data. As shown in Figure 1, the engine consists of four learned modules, *i.e.*, feature module, geometry module, semantic module and color module. For each ray, we first collect the learned grid features through the feature module. Then, these features are fed into a Multi-Layer Perceptron (MLP) decoder, and the output of the decoder are weighted by the geometric global features to get density and semantic-color feature. The geometric global features are learned from the sampling points of each ray beside the MLPs training. After that, we use semantic-color feature and MLP decoders to get semantics and color. In color module, the direction embedding is used for color rendering. Finally, we can get depth, semantics and image with the volume rendering.

3.1. Global feature for implicit fields

We sample the ray as instant NGP [20] and the sampling point is denoted by $p(x, y, z)$. The feature module learns a multi-resolution hash table [20], which takes the input of a ray point coordinate (x, y, z) and outputs the feature $\mathcal{F}$ of this point. In this way, the collected features only contain the ray-point-located grid features, lacking of global information. This paper proposes to associate all ray samples with 3D global feature. Inspired by the point-to-surface representation [25] that learns global surfaces to extract the local and global feature of point cloud for segmentation and classification, in this paper, we use the learned quadrics to globally associate all independent rays, attaching global features to each sampling point. The point-to-surface representation is expressed as:

$$d = \boldsymbol{\pi}^T \cdot \boldsymbol{X}, \tag{1}$$

where $\boldsymbol{\pi}$ is the coefficients vector of the quadratic terms, namely the global feature, and the notation $\boldsymbol{X}$ is the vector of quadratic terms, *i.e.*,

$$\boldsymbol{X} = (x^2, y^2, z^2, xy, xz, yz, x, y, z). \tag{2}$$

We map the value of d to a range of $[0, 1]$ with sigmoid function [9]. Then, $1 - sigmoid(d)$ is simply used to indicate proximity of the point to the global surface.

In order to attach this global feature to a point, we use an MLP as embedding layer L_{emb} to get the weights and globally weight the geometry distribution. The weights can be expressed as:

$$\boldsymbol{\omega} = L_{emb}(1 - (sigmoid(\pi^T \cdot \boldsymbol{X}))). \tag{3}$$

Then, as shown in Figure 1, we define the point density σ, semantics $\boldsymbol{s}$ and color $\boldsymbol{c}$ as:

$$[\sigma, \mathcal{F}_{\boldsymbol{sc}}] = \boldsymbol{\omega}_{\sigma\mathcal{F}} \cdot D_{\sigma\mathcal{F}}(\mathcal{F}), \tag{4}$$

$$\boldsymbol{s} = D_{\boldsymbol{s}}(\mathcal{F}_{\boldsymbol{sc}}), \tag{5}$$

$$\boldsymbol{c} = D_{\boldsymbol{c}}([\mathcal{F}_{\boldsymbol{sc}}, \mathcal{E}_{dir}]), \tag{6}$$

where the symbol "$\cdot$" represents the multiplication of corresponding elements. Since the semantics are direction-invariant, we only decode $\mathcal{F}_{\boldsymbol{sc}}$ to obtain $\boldsymbol{s}$ as formulated in Eq. 5. On the contrary, obtaining color requires decoding both $\mathcal{F}_{\boldsymbol{sc}}$ and direction embedding $\mathcal{E}_{dir}$, as shown in Eq. 6.

After getting the point-wise attributes, *i.e.*, σ, $\boldsymbol{s}$ and $\boldsymbol{c}$, we use the volume rendering to get depth, semantics and image. The volume rendering can be formulated as:

$$\mathcal{A} = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i \delta_i))\boldsymbol{a}_i, \tag{7}$$

where, the notation i indexes the point on a ray, δ_i is the distance between two adjacent sampling points, and N represents the total number of the sampling points. The notation T_i is the accumulated transmittance along the ray direction, expressed as:

$$T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j). \tag{8}$$

The notation $\mathcal{A}$ indicates different attributes, such as depth, semantics or image, and the corresponding point attribute $\boldsymbol{a}_i$ is $\sum_{i=1}^{i} \delta_i$, $\boldsymbol{s}_i$, or $\boldsymbol{c}_i$, where $\boldsymbol{s}_i$ is an N_c-dimensional vector, and each dimension represents the probability of a semantic category.

3.2. The loss functions

Color loss: The application scenarios of the data engine are complex and diverse. Therefore, we carefully design the loss functions for different implicit fields. The implicit color field is trained with color loss, defined as:

$$\mathcal{L}_c = \frac{1}{B}\sum_{b=1}^{B} \|\boldsymbol{c}_b - \hat{\boldsymbol{c}}_b\|^2, \tag{9}$$

where B is the batch size, $\boldsymbol{c}_b$ is the predicted color, and $\hat{\boldsymbol{c}}_b$ is the Ground Truth (GT).

Semantic loss: For semantic field, we use a weighted cross-entropy loss function, which is expressed as:

$$\mathcal{L}_s = -\sum_{r \in \mathcal{R}} \sum_{k=1}^{N_c} w_k \hat{p}_k(r) \log p_k(r), \tag{10}$$

where N_c is the number of the semantic categories, r is the sampled ray and $\mathcal{R}$ represents the set of all the training rays in one batch. The notations p_k and $\hat{p}_k$ are the predicted probability and GT, respectively. Different from the traditional cross-entropy loss function used in Semantic NeRF [34] or SS-NeRF [33], in this paper, we weight each category by the weights w_k to balance the contribution of samples across all classes. The weights w_k is defined as:

$$w_k = \lfloor_l \frac{n_k}{\sum_{i=0}^{N_c} n_i} \rceil^h, \tag{11}$$

where n_k is the number of pixels of the k^{th} category. The operation $\lfloor_l x \rceil^h$ represents up and down truncation, defined as:

$$\lfloor_l x \rceil^h = \begin{cases} l, & x < l \\ x, & l \le x \le h \\ h, & x > h \end{cases}. \tag{12}$$

In our subsequent experiments, we set $l = 1$ and $h = 5$ to ensure that the weight of large samples is not too low, and to limit the maximum weight of small samples. This weighted loss helps to solve the problem of unbalanced samples in semantics learning.

Eq. 7 indicates that the obtained semantics are related to both semantic attribute $\boldsymbol{s}_i$ and density σ_i. In automatic back-propagation algorithm, the semantics will feed back a gradient to density σ. Thus, if the automatic back-propagation algorithm is used directly in semantics learning, the geometry will be interfered with semantics and more seriously, the network will fail to converge. To avoid this, Siddiqui *et al.* [21] proposed to use the detach function in PyTorch to stop gradients to the density. However, stopping gradients results in features not being updated for semantics, increasing the burden on the decoder, and reducing performance. To solve this problem, we customize the back-propagation function that omits the partial derivative of σ from $\boldsymbol{s}$ for calculating the gradient of σ.

Depth loss: In some special scenarios, such as self-driving, sparse views result in noisy geometry, which further leads to noisy color and semantics. In this case, we introduce the depth supervision with some geometric prior such as depth or point cloud to construct the implicit engine. Similar to DepthFormer [1], we define the depth loss function as:

$$\mathcal{L}_d = \sqrt{\frac{1}{N_d}\sum_i g_i^2 + \frac{\lambda}{{N_d}^2}(\sum_i g_i)^2}, \tag{13}$$

where $g_i = \log d_i - \log \hat{d}_i$, $\hat{d}_i$ is GT and d_i is the predicted depth. The number of the GT pixels is denoted by N_d. The notation $\lambda = 0.15$ in our experiments.

3.3. Implementation details

Our method is implemented by PyTorch based on the open-source project ngp-pl [30]. We use the CUDA code to customize our back-propagation function. For scenarios with dense views, the engine is trained without depth supervision, while for sparse views, depth supervision can be used to effectively construct a data engine. All kinds of engines are trained on a RTX-A6000 GPU machine. For each scene, we train the network with the batch size of 8192. Each epoch contains 1000 steps. The learning rate is 0.01.

Table 1. The evaluation of efficiency, color, semantics, and geometry on Replica dataset [23]. The evaluated indicators include training time T_t, rendering time T_r, PSNR, SSIM, mIoU, total accuracy ACC$_t$, class average accuracy ACC$_a$, AbsDiff, SqRel and RMSE as defined in [34]. The best ones are bolded.

Performance	Efficiency		Color		Semantics			Geometry		
Method	T_t	T_r	PSNR (dB)	SSIM	mIoU (%)	ACC$_t$ (%)	ACC$_a$ (%)	AbsDiff	SqRel	RMSE
SS-NeRF [33]	9 h	–	30.2	–	92.4	–	–	–	–	–
Semantic-NeRF [34]	8 h	4.82 s	31.39	0.930	93.68	99.00	96.53	0.032	0.007	0.096
Ours	**15 min**	**0.1 s**	**35.9**	**0.970**	**94.79**	**99.29**	**97.68**	**0.0056**	**0.0006**	**0.0122**

4. Experiments

4.1. Datasets and experiments

The data engine focuses more on the quality of the generated data, so we select three high-quality public datasets to validate and evaluate our method, namely the indoor Replica dataset [23], outdoor scene Virtual KITTI (VKITTI) [4], and object scene Synthetic-NeRF [19].

We first analyze the performance of our method and the compared baselines from the aspects of efficiency, color, semantics, and geometry. Then, we conduct ablation study to validate the proposed method. Finally, we present the applications of the proposed engine.

4.2. The performance of IS-NEAR

We compare the efficiency, color, semantics, and geometry of the IS-NEAR and the SOTA works, namely semantic NeRF [34] and SS-NeRF [33]. Different from the pseudo-label methods [12, 15, 21], we use high-quality GT to train and evaluate the engine. Table 1 shows the comparison results, which are the mean of 7 Replica [23] scenes (room:0-1, office:0-4).

Efficiency: IS-NEAR takes 15 minutes to train one scene, while the compared methods take about 8 to 9 hours. The time cost of IS-NEAR is less than 3% of that of the compared methods. The use of hash features significantly improves the efficiency. Moreover, since the time is proportional to the number of volume rendering samples on a ray, the proposed global feature reduces the number of sampling points as shown in Figure 2, and further reduces the training time. For a VKITTI [4] scene, with the absence of 3D global feature, the example scene takes 1019 s for training. While in 3D global feature case, it takes 648 s, further improved by 36.4% ($\frac{1019-648}{1019}$) on the basis of hash method [20]. Similarly, the rendering time of our method takes only 0.1 s for an image of 320×240, which is a significant improvement over semantic NeRF's 4.82 s. Our method also supports multi-GPU parallel training. The training time can be further reduced proportionally.

Color: As for image quality, the average PSNR of novel view synthesis is 35.9 dB, and the Structure Similarity Index Measure (SSIM) is 0.97, outperforming the comparison methods. Compared with Semantic-NeRF [34], the IS-NEAR can reconstruct more details as the visualization results shown in Figure 3. These details are crucial for data generation and simulation in some applications such as multi-view 3D reconstruction, feature point extraction, *etc*. Therefore, our method is closer to the basic characteristics of a data engine, as implied by its name, IS-NEAR.

Semantics: The mIoU, ACC$_t$ and ACC$_a$ of IS-NEAR are 94.79%, 99.29%, 97.68%, respectively, superior to the Semantic-NeRF and the SS-NeRF [33]. In order to more intuitively compare the semantic reconstruction effect, some rendering results are shown in Figure 4. These results indicate that our engine can reconstruct more semantic details, which is benefit from the use of the designed 3D global fea-

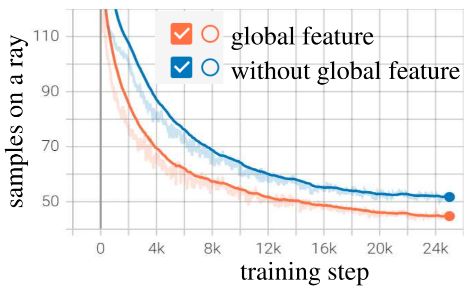

Figure 2. Ray samples with or without 3D global feature.

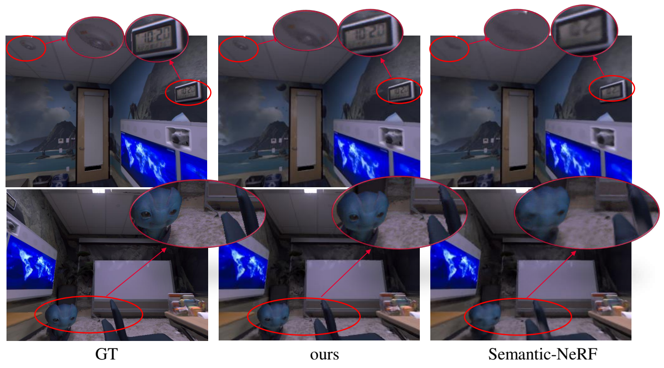

Figure 3. The image rendering quality comparison between our method and the Semantic-NeRF [34]. The red circle marks the contrasting details. Please zoom in for details.

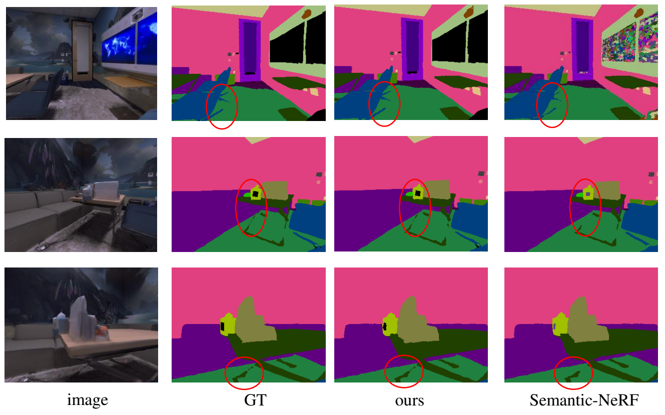

Figure 4. The visualization results of the rendered semantics. Some details to be contrasted are marked in red circles. Please zoom in for a good view.

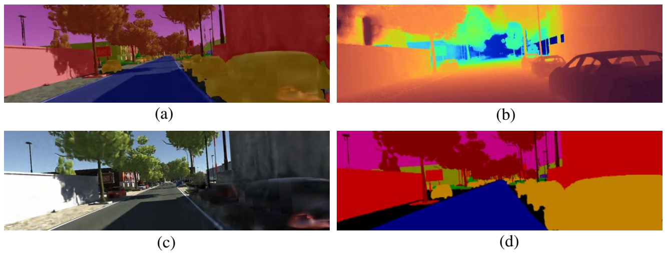

Figure 5. The rendered novel view on VKITTI [4]: (a) image with semantic mask, (b) depth, (c) image, (d) label.

ture and the truncated-weights semantic loss. These details are important for a data engine. We know that the upper limit of GT determines the upper limit of the perception algorithm. We prefer it to be near to the limit (100%). However, the closer it is to the limit, the more difficult it is to achieve. Nevertheless, the experimental results indicate that our method IS NEAR.

Geometry: For fair comparison, AbsDiff, SqRel and RMSE are used to evaluate the geometry of IS-NEAR as they are also used in Semantic-NeRF [34]. The comparison results show that our geometric indicators are superior to the Semantic-NeRF, and the geometric error is reduced by about an order of magnitude. The pipeline in Figure 1 indicates that geometry is the foundation of color and semantics. Therefore, geometry is essential for semantic and color rendering.

Sparse view: Sparse view data is challenging due to the little geometric constraints from different views, such as the car driving data. To address this issue, we introduce geometric priors, where geometry is supervised by the existing depth images. In this case, we use the VKITTI [4] dataset to evaluate the semantics and colors. To do this, a static sequence is selected for training and testing. Figure 5 shows the rendering results of a novel view. The test PSNR, mIoU, ACC_t and ACC_a are 22.5 dB, 77.7%, 95.9% and 88.2%, respectively.

4.3. Ablation studies

Global feature: In this section, the details of the proposed method are analyzed in ablation study. Firstly, we analyze the effect of the global feature. We compare the proposed method to the baseline instant NGP [20]. Although the baseline is equipped with a semantic decoder, the performance of the baseline is equivalent to that of instant NGP due to the use of custom backpropagation function. The results of Table 2 show that the point-to-surface global feature simultaneously improves the performance of geometry, color and semantics, which indicates that the global feature effectively improves the representation ability of geometry in the implicit fields. In addition, the improved geometry further enhances semantics and color.

Table 2. The ablation study of the global features.

Performance	RMSE (m)	PSNR(dB)	SSIM	mIoU(%)	ACC_t(%)	ACC_a(%)
Baseline	0.09	36.5	0.970	91.4	98.9	96.0
Ours	0.027	41.2	0.992	94.2	99.3	97.8

Sparse labels: We prefer to feeding less labels to train an engine and keep the quality of the generated data. Therefore, we test the performance with sparse labels. Table 3 records the quantization results of different sparse rate labels on Replica dataset. The results of the first five rows in Table 3 illustrates that the performance of the dense and sparse labels training are similar, which is conducive to speeding up the semantic scene labeling. We also conduct an experiment to validate the effectiveness of global feature in reducing pixel-wise labels feeding for semantic field training. The results recorded in the last three rows marked by w/o PS, indicat that the networks are trained by these sparse labels without point-to-surface global feature. The comparison between the first and last three rows results shows that the global feature can significantly improve the performance for sparse labels. These results indicate that the global feature enhances the representation capability of a single ray. Therefore, the IS-NEAR help us to reduce the manual labels feeding for annotation.

Truncated-weights semantic loss function: In the training process of the implicit semantic field, unbalanced samples make it difficult to learn the categories with a small number of samples. To solve this problem, the small samples categories are weighted with a larger weight when solving the loss as formulated in Eq. 10. Figure 6 is the comparison results about this loss. The results show that the weighted loss helps to recover the detailed semantics.

Semantic rendering without direction embedding: As

Table 3. The performance of the data engine trained with sparse labels. The rate R represents the ratio of the number of semantic labels to the images.

R	PSNR (dB)	SSIM	mIoU (%)	ACC$_t$ (%)	ACC$_a$ (%)
10%	40.7	0.989	93.1	99.1	96.1
20%	40.5	0.988	93.1	99.1	96.2
25%	40.3	0.988	92.4	99.0	96.2
50%	40.7	0.989	92.6	99.0	96.1
100%	40.6	0.989	92.3	99.0	95.9
10%w/o PS	39.6	0.985	83.6	97.3	91.0
20%w/o PS	39.1	0.983	89.3	98.5	94.5
25%w/o PS	39.1	0.982	89.4	98.6	94.6

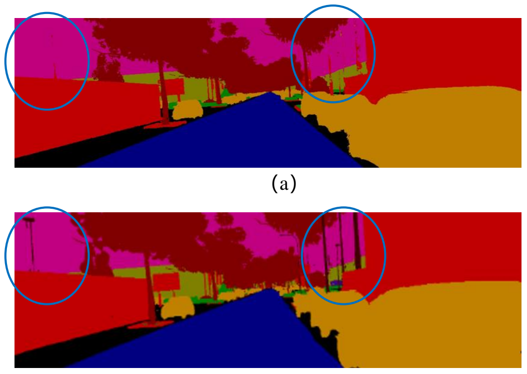

Figure 6. The comparison of truncated-weights semantic loss function. (a) The engine is trained with the traditional cross-entropy loss. (b) The engine is trained with the truncated-weights cross-entropy loss.

mentioned above, the view direction is irrelevant to semantics. Similar to SS-NeRF [33], we use the semantic-color feature to render the semantics without the direction embedding. In order to verify its effect, we conduct a comparative experiment on Synthetic-NeRF [19] dataset. The comparison results are shown in Figure 7. Figure 7(a) shows the novel view rendering result of adding direction embedding to the semantic decoder D_s. In this case, the noise occurs when rendering a novel view even though the training view is clean. When the direction embedding is removed, the rendering image is noise-free as shown in Figure 7(b).

4.4. Application

To further verify the effectiveness of the proposed method, the proposed implicit engine is applied to part segmentation labeling, texture re-rendering and robot simulation.

Part segmentation labeling: Scene labeling is an important application of the implicit semantic field. The traditional manual labeling methods are not only slow, but also low quality. In this paper, we use the proposed engine to construct high-quality implicit semantic field with a few artificially coarse labels. We test our approach on Synthetic-NeRF [19] datasets without semantic labels. Some examples are given in Figure 8. We take the ficus object as an example. We use the traditional interactive-annotation tool EISeg [10] to obtain sparse and coarse semantic labels, which only takes little time. In this ficus case, we only annotate two coarse labels as shown in the first row of Figure 8. Then these sparse and coarse labels and posed images are used to train the model. Semantic loss is not calculated when the ray is not attached with a semantic label during the training. It takes about 5 minutes for training a model on the four RTX-A6000 platform. When rendering, we mask the background semantics with the estimated depth range and get the final labeling results as shown in the second row of Figure 8. The results show that we can use coarse label to train the engine and get the detailed result that is impossible for the traditional artificial. In addition, only few views labels are enough for a complete scene annotation, which shows that the IS-NEAR achieves semantic propagation. This is also impossible for the traditional 3D

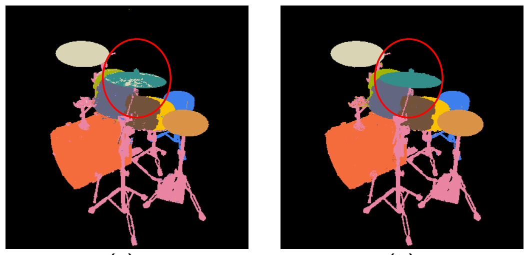

Figure 7. The comparison of direction embedding. (a) The rendering novel view semantics have noises when the direction embedding is added to the semantic decoder. (b) No noise occurs when the direction embedding is removed.

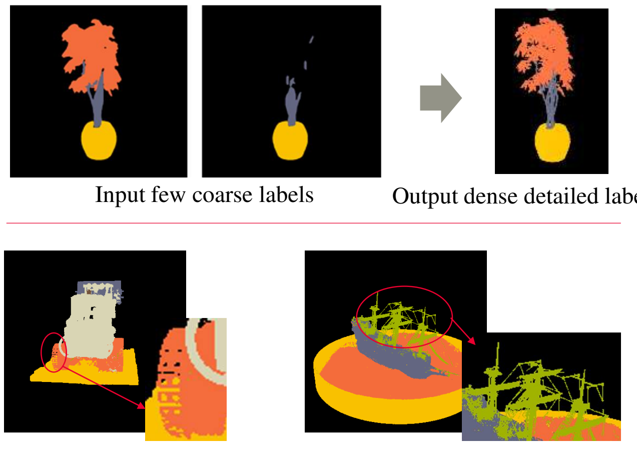

Figure 8. Part segmentation labeling with the IS-NEAR engine. We train the engine with sparse and coarse labels and get the dense and detailed novel view images and labels.

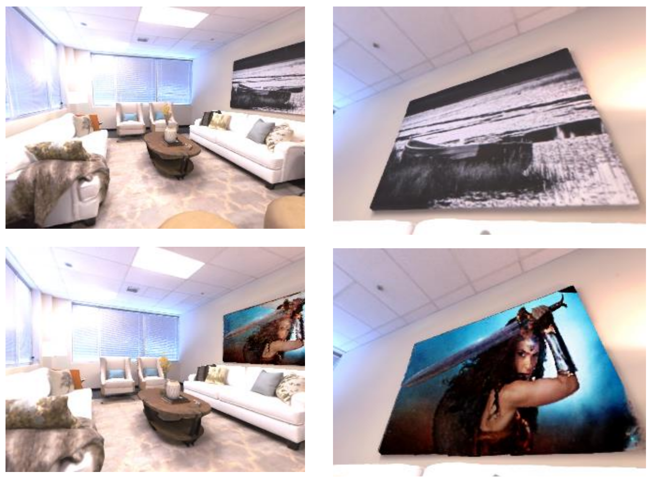

Figure 9. Replacing a picture on the wall with the IS-NEAR for spatially consistent texture re-rendering.

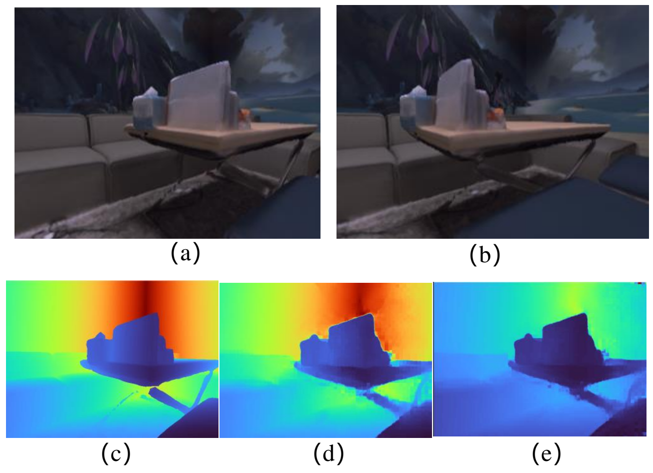

Figure 10. The engine is used for stereo data generation with GT depth. (a) and (b) are the rendered left and right view, (c) is the rendered depth. (d) is the stereo depth estimation result with DROID-SLAM [27], and (e) is the monocular depth estimation result.

reconstruction methods [11, 18, 22].

Texture re-rendering: In order to improve the generalization ability of perceptual algorithms, data augmentation is an effective way. However, traditional generation networks, such as the Generative Adversarial Network (GAN) based methods [36], are difficult to achieve consistent style and texture replacement. To solve this problem, we propose to implement texture re-rendering by the interaction of the multiple implicit fields. For example, if we want to replace a picture in a room scene as shown in Figure 9, we need to give a new texture image and select a view to map it onto the original region. Then, the new texture is mapped into the implicit fields through the model training. In the training, as

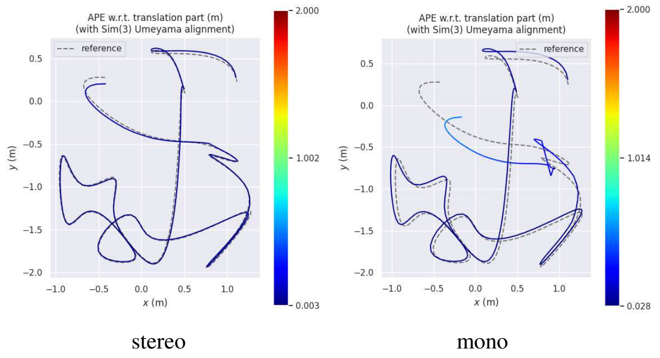

Figure 11. The trajectory estimation results with stereo and mono sequence.

for the selected view, if the semantic category is a picture, the color GT is replaced by the corresponding new texture image, otherwise we use the original image. With regard to the other views, the color loss is masked by the picture semantic, which ensures that other views do not interfere with the learning of the new texture and that the texture is spatially consistent. The results show that the implicit geometry field ensures the consistency of different views textures, and the semantic information is the key to update the specified view texture.

Robot simulation: We also utilize our implicit data engine to generate video data sequence of both mono and stereo types in order to verify its effectiveness of simulation. Figure 10 shows the rendering results, including the (a) left and (b) right views and (c) the corresponding depth GT. We run DROID-SLAM [27] on the generated data in both mono and stereo modes. For depth estimation, the stereo result outperforms the mono type as shown in Figure 10 (d) and (e). At the same time, we find the estimated stereo depth is close to the generated depth GT. Furthermore, with stereo data, the method produces better video trajectory estimation as shown in Figure 11, with an ATE of 3.5 cm compared to the mono type with 29 cm. These results indicate that the generated data is valid and can be used for robot simulation and algorithm validation.

5. Conclusion

This paper proposes a 3D global feature based data engine IS-NEAR with implicit fields. By introducing the global features to implicit fields, our engine outperforms the existing methods in terms of efficiency, geometry, color, and semantics. The proposed method can be used for fast and high-quality scene labeling, and generating multi-sensor data for various computer vision tasks, such as part segmentation, texture re-rendering and robot algorithms simulation.

For sparse view input, prior geometry is required to guide geometry learning. This is a limitation of our method that should be further studied in the future.

References

[1] Ashutosh Agarwal and Chetan Arora. Depthformer: Multi-scale vision transformer for monocular depth estimation with global local information fusion. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3873–3877, 2022. 4

[2] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6290–6301, 2022. 2

[3] Wang Bing, Lu Chen, and Bo Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *arXiv preprint arXiv:2208.07227*, 2022. 1

[4] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2, 2020. 5, 6

[5] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12882–12891, 2022. 2

[6] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 1

[7] Xiao Fu, Shangzhan Zhang, Tianrun Chen, Yichong Lu, Lanyun Zhu, Xiaowei Zhou, Andreas Geiger, and Yiyi Liao. Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation. In *International Conference on 3D Vision (3DV)*, 2022. 1, 2

[8] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5511–5520, 2022. 2

[9] Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *From Natural to Artificial Neural Computation*, pages 195–201, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. 3

[10] Yuying Hao, Yi Liu, Zewu Wu, Lin Han, Yizhou Chen, Guowei Chen, Lutao Chu, Shiyu Tang, Zhiliang Yu, Zeyu Chen, et al. Edgeflow: Achieving practical interactive segmentation with edge-guided flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1551–1560, 2021. 7

[11] Alexander Hermans, Georgios Floros, and Bastian Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2631–2638, 2014. 8

[12] Junwen Huang, Alexey Artemov, Yujin Chen, Shuaifeng Zhi, Kai Xu, and Matthias Nießner. S4r: Self-supervised semantic scene reconstruction from rgb-d scans, 2023. 1, 2, 5

[13] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18365–18375, 2022. 2

[14] Fangfu Liu, Chubin Zhang, Yu Zheng, and Yueqi Duan. Semantic ray: Learning a generalizable semantic field with cross-reprojection attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17386–17396, 2023. 1

[15] Zhizheng Liu, Francesco Milano, Jonas Frey, Marco Hutter, Roland Siegwart, Hermann Blum, and Cesar Cadena. Unsupervised continual semantic adaptation through neural rendering, 2022. 1, 2, 5

[16] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[17] Kirill Mazur, Edgar Sucar, and Andrew Davison. Feature-realistic neural fusion for real-time, open set scene understanding. In *2023 International Conference on Robotics and Automation (ICRA)*, 2023. 1, 2

[18] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4628–4635, 2017. 8

[19] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision - ECCV 2020*, pages 405–421, Berlin, Heidelberg, 2020. Springer-Verlag. 1, 2, 5, 7

[20] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1, 3, 5, 6

[21] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3d scene understanding with neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9043–9052, 2023. 1, 2, 4, 5

[22] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 8

[23] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 5

[24] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time. In

Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 6229–6238, 2021. 2

[25] Tiecheng Sun, Guanghui Liu, Ru Li, Shuaicheng Liu, Shuyuan Zhu, and Bing Zeng. Quadratic terms based point-to-surface 3d representation for deep learning of point cloud. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(5):2705–2718, 2022. 2, 3

[26] Tao Sun, Mattia Segu, Janis Postels, Yuxuan Wang, Luc Van Gool, Bernt Schiele, Federico Tombari, and Fisher Yu. Shift: A synthetic driving dataset for continuous multi-task domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21371–21382, 2022. 1

[27] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. In *Advances in Neural Information Processing Systems*, pages 16558–16569. Curran Associates, Inc., 2021. 8

[28] S. Vora, N. Radwan, K. Greff, H. Meyer, K. Genova, Msm Sajjadi, E. Pot, A. Tagliasacchi, and D. Duckworth. Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes. 2021. 1, 2

[29] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems*, pages 27171–27183. Curran Associates, Inc., 2021. 2

[30] Project webpage. ngp-pl. `https://github.com/kwea123/ngp_pl`. 4

[31] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5438–5448, 2022. 2

[32] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[33] Mingtong Zhang, Shuhong Zheng, Zhipeng Bao, Martial Hebert, and Yu-Xiong Wang. Beyond rgb: Scene-property synthesis with neural radiance fields. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 795–805, 2023. 1, 4, 5, 7

[34] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15838–15847, 2021. 1, 2, 4, 5, 6

[35] Shuaifeng Zhi, Edgar Sucar, Andre Mouton, Iain Haughton, Tristan Laidlow, and Andrew J. Davison. ilabel: Revealing objects in neural fields. *IEEE Robotics and Automation Letters*, 8(2):832–839, 2023. 1, 2

[36] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 8