NeRF-Det: Learning Geometry-Aware Volumetric Representation for Multi-View 3D Object Detection

Chenfeng Xu\textsuperscript{1} Bichen Wu\textsuperscript{2} Ji Hou\textsuperscript{2,*} Sam Tsai\textsuperscript{2} Ruilong Li\textsuperscript{1} Jialiang Wang\textsuperscript{2} Wei Zhan\textsuperscript{1} Zijian He\textsuperscript{2} Peter Vajda\textsuperscript{2} Kurt Keutzer\textsuperscript{1} Masayoshi Tomizuka\textsuperscript{1}

\textsuperscript{1}University of California, Berkeley \quad \textsuperscript{2}Meta AI

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{NeRF-Det aims to detect 3D objects with only RGB images as input. To enhance detection, we propose to embed a NeRF branch with our designed synergy. The two joint branches share the same geometry representation and are trained end-to-end, which helps achieve state-of-the-art accuracy on multi-view indoor RGB-only 3D detection, and additionally enables a generalizable novel view synthesis on new scenes without per-scene optimization.}
\end{figure}

\section*{Abstract}

We present NeRF-Det, a novel method for indoor 3D detection with posed RGB images as input. Unlike existing indoor 3D detection methods that struggle to model scene geometry, our method makes novel use of NeRF in an end-to-end manner to explicitly estimate 3D geometry, thereby improving 3D detection performance. Specifically, to avoid the significant extra latency associated with per-scene optimization of NeRF, we introduce sufficient geometry priors to enhance the generalizability of NeRF-MLP. Furthermore, we subtly connect the detection and NeRF branches through a shared MLP, enabling an efficient adaptation of NeRF to detection and yielding geometry-aware volumetric representations for 3D detection. Our method outperforms state-of-the-arts by 3.9 mAP and 3.1 mAP on the ScanNet and ARKITScenes benchmarks, respectively. We provide extensive analysis to shed light on how NeRF-Det works.

As a result of our joint-training design, NeRF-Det is able to generalize well to unseen scenes for object detection, view synthesis, and depth estimation tasks without requiring per-scene optimization. Code is available at \url{https://github.com/facebookresearch/NeRF-Det}.

\section{1. Introduction}

In this paper, we focus on the task of indoor 3D object detection using posed RGB images. 3D object detection is a fundamental task for many computer vision applications such as robotics and AR/VR. The algorithm design depends on input sensors. In the past few years, most 3D detection works focus on both RGB images and depth measurements (depth images, point-clouds, etc.). While depth sensors are widely adopted in applications such as autonomous driving, they are not readily available in most AR/VR headsets and mobile phones due to cost, power dissipation, and form factor constraints. Excluding depth input, however, makes 3D object detection significantly more challenging, since we need to understand not only the semantics, but also the...
underlying scene geometry from RGB-only images.

To mitigate the absence of geometry, one straightforward solution is to estimate depth. However, depth estimation itself is a challenging and open problem. For example, most monocular depth-estimation algorithms cannot provide accurate metric depth or multi-view consistency [10, 32, 16, 29]. Multi-view depth-estimation algorithms can only estimate reliable depth in textured and non-occluded regions [8, 34].

Alternatively, ImVoxelNet [31] models the scene geometry implicitly by extracting features from 2D images and projecting them to build a 3D volume representation. However, such a geometric representation is intrinsically ambiguous and leads to inaccurate detection.

On the other hand, Neural Radiance Field (NeRF) [21, 3, 3] has been proven to be a powerful representation for geometry modeling. However, incorporating NeRF into the 3D detection pipeline is a complex undertaking for several reasons: (i) Rendering a NeRF requires high-frequency sampling of the space to avoid aliasing issues [21], which is challenging in the 3D detection pipeline due to limited resolution volume. (ii) Traditional NeRFs are optimized on a per-scene basis, which is incompatible with our objective of image-based 3D detection due to the considerable latency involved. (iii) NeRF makes full use of multi-view consistency to better learn geometry during training. However, a simple stitch of first-NeRF-then-perception [37, 14, 15] (i.e., reconstruction-then-detection) does not bring the advantage of multi-view consistency to the detection pipeline.

To mitigate the issue of ambiguous scene geometry, we propose NeRF-Det to explicitly model scene geometry as an opacity field by jointly training a NeRF branch with the 3D detection pipeline. Specifically, we draw inspirations from [41, 49] to project ray samples onto the image plane and extract features from the high-resolution image feature map, rather than from the low-resolution volumes, thereby overcoming the need for high-resolution volumes. To further enhance the generalizability of NeRF model to unseen scenes, we augment the image features with more priors as the input to the NeRF MLP, which leads to more distinguishable features for NeRF modeling. Unlike previous works that build a simple stitch of NeRF-then-perception, we connect the NeRF branch with the detection branch through a shared MLP that predicts a density field, subtly allowing the gradient of NeRF branches to back-propagate to the image features and benefit the detection branch during training. We then take advantage of the uniform distribution of the volume and transform the density field into an opacity field and multiply it with the volume features. This reduces the weights of empty space in the volume feature. Then, the geometry-aware volume features are fed to the detection head for 3D bounding box regression. It is worth noting that during inference, the NeRF branch is removed, which minimizes the additional overhead to the original detector.

Our experiments show that by explicitly modeling the geometry as an opacity field, we can build a much better volume representation and thereby significantly improve 3D detection performance. Without using depth measurements for training, we improve the state-of-the-art by 3.9 and 3.1 mAP on the ScanNet and the ARKITScenes datasets, respectively. Optionally, if depth measurements are also available for training, we can further leverage depth to improve the performance, while our inference model still does not require depth sensors. Finally, although novel-view synthesis and depth estimation are not our focus, our analysis reveals that our method can synthesize reasonable novel-view images and perform depth prediction without per-scene optimization, which validates that our 3D volume features can better represent scene geometry.

2. Related Work

3D Detection in Indoor Scene. 3D detection utilizes various methods depending on inputs, and has achieved great success on point cloud [22, 25, 26, 28, 50] and voxel representations [47, 12, 11, 6, 13]. 3D-SIS [11] uses anchors to predict 3D bounding boxes from voxels fused by color and geometric features. The widely-used VoteNet [27] proposes hough voting to regresses bounding box parameters from point features. However, depth sensors are not always readily available on many devices due to its huge power consumption, such as on VR/AR headsets. To get rid of sensor depth, Panoptic3D [5] operates on point clouds extracted from predicted depth. Cube R-CNN [2] directly regresses 3D bounding boxes from a single 2D image.

Comparably, the multi-view approach is also not limited by depth sensors and is more accurate. However, the current state-of-the-art multi-view method [31] fuses the image naively by duplicating pixel features along the ray, which does not incorporate a sufficient amount of geometric clues. To address this, we leverage NeRF to embed geometry into the volume for better 3D detection.

3D Reconstruction with Neural Radiance Field. Neural Radiance Field (NeRF) [21] is a groundbreaking 3D scene representation that emerged three years ago, and has proven to be powerful for reconstructing 3D geometry from multi-view images [21, 1, 46, 51, 39]. Early works [21, 1, 18, 24, 48] directly optimize for per-scene density fields using differentiable volumetric rendering [20]. Later, NeuS [40] and VolSDF [46] improve the reconstruction quality by using SDF as a replacement of density as the geometry representation. Recently, Ref-NeRF [36] proposes to use reflective direction for better geometry of glossy objects. Aside from aforementioned per-scene optimization methods, there are also works that aim to learn a generalizable NeRF from multiple scenes, such as IBRNet [41] and MVS-NeRF [3], which predict the density and color at each 3D location con-
Our method, termed as NeRF-Det, uses posed RGB images for indoor 3D object detection by extracting image features and projecting them into a 3D volume. We leverage NeRF to help infer scene geometry from 2D observations. To achieve this, we entangle the 3D object detection and NeRF with a shared MLP, with which the multi-view constraint in NeRF can enhance the geometry estimation for the detection, as shown in Fig. 2.

3. Method

3.1. 3D Detection Branch

In the 3D detection branch, we input posed RGB frames to the 2D image backbone, denoting the images as $I_i \in \mathbb{R}^{H_i \times W_i \times 3}$ and the corresponding intrinsic matrix and extrinsic matrix as $K \in \mathbb{R}^{3 \times 3}$ and $R_i \in \mathbb{R}^{3 \times 4}$, where $i = 1, 2, 3, ..., T$ and $T$ is the total number of views. We follow [31], which uses an FPN [19] to fuse multistage features and use high resolution features, denoted as $F_i \in \mathbb{R}^{C \times H_i/4 \times W_i/4}$, to generate a 3D feature volume.

We create the 3D feature volume by attaching 2D features from each image to their corresponding positions in 3D. We establish a 3D coordinate system, with the z-axis denoting height, and the x- and y-axis denoting two horizontal dimensions. Then, we build a 3D grid of with $N_x \times N_y \times N_z$ voxels. For each voxel center with coordinate $p = (x, y, z)^T$, we project it to view-$i$ to obtain the 2D coordinates as

$$ (u'_i, v'_i, d_i)^T = K' \times R_i \times (p, 1)^T, \quad (1) $$

where $(u_i, v_i) = (u'_i/d_i, v'_i/d_i)$ is to the pixel coordinate of $p$ in view-$i$. $K'$ is the scaled intrinsic matrix, considering the feature map downsampling. After building this correspondence, we assign 3D features as

$$ V_i(p) = \text{interpolate}((u_i, v_i), F_i), \quad (2) $$

where $\text{interpolate}()$ looks up the feature in $F_i$ at location $(u_i, v_i)$. Here we use nearest neighbor interpolation. For $p$ that are projected outside the boundary of the image, or behind the image plane, we discard the feature and set $V_i(p) = 0$. Intuitively, this is equivalent to shooting a ray...
from the origin of camera-i through pixel \((u_i, v_i)\), and for all voxels that are on the ray, we scatter image features to the voxel in \(V_i\). Next, we aggregate multi-view features by simply averaging all effective features as done in ImVoxelNet [31]. Letting \(M_p\) denote the number of effective 2D projections, we compute \(V_{\text{avg}}(p) = \sum_{i=1}^{M_p} V_i(p)/M_p\).

However, volume features generated this way “over populate” projection rays without considering empty spaces and other geometric constraints. This makes the 3D representation ambiguous and causes detection errors. To mitigate this issue, we propose to incorporate a NeRF branch to improve learning geometry for the detection branch.

### 3.2. NeRF Branch

**Feature Sampling.** NeRF [21] is a neural rendering method for novel view synthesis. However, previous NeRFs sample features from the 3D volume with high resolution, such as 128\(^3\) [21]. In 3D detection, we use a lower grid resolution of 40 \times 40 \times 16, which suffers from the aliasing issue and results in degradation of geometry learning. To mitigate this, we draw inspiration from [41, 49] and sample features from higher resolution 2D image feature maps, typically of size 160 \times 120, as shown in Fig. 3. Specifically, we first sample points along the ray originated from the camera, i.e., \(r(t) = o + t \times d\) where \(o\) is the ray origin and \(d\) is the view direction. For a coordinate \(p\) sampled on the ray and a viewing direction \(d\), we can compute the color \(c(p, d)\) and density \(\sigma(p)\) as:

\[
\begin{align*}
\sigma(p, \bar{h}(p)) &= G-\text{MLP}(\bar{V}(p), \gamma(p)), \\
c(p, d) &= C-\text{MLP}(\bar{h}(p), d).
\end{align*}
\]

\(\bar{V}(p)\) is ray features aggregated and augmented from multi-view features, and \(\gamma(p)\) is the position encoding same as [21] while \(\bar{h}\) is latent features. The first MLP is termed \(G\text{-MLP}\) for estimating geometry and the second MLP is termed \(C\text{-MLP}\) for estimating color. For activations, we follow [21] and use ReLU for the density \(\sigma(p)\) and sigmoid for the color \(c(p, d)\).

**Augmenting Features.** Although it is similar to [41, 49] that use image features, it is still difficult to make \(G\text{-MLP}\) estimate accurate geometry across different scenes by simply averaging features from multi-view features as detection branch does. Therefore, we propose to incorporate more priors to help optimize \(G\text{-MLP}\). Beyond averaging the projected features, we augment the sampled features with the variance from multiple views \(V^{\text{var}}(p) = \sum_{i=1}^{M_p} (V_i(p) - V_{\text{avg}}(p))^2 / M_p\). The variance of the color features is able to roughly describe the occupancy of the 3D field, which has been widely used as cost volume in multi-view stereo depth estimation [45]. If the 3D location \(p\) is occupied, the variance of the observed features should be low under the assumption that the scene only contains Lambertian surfaces. On the other hand, if the location is in free space, different appearances would be observed from different views, and therefore the variance of color features would be high. Compared to naive average of features, variance provides a better prior for estimating scene geometry.

In addition to extracted deep features, which are trained to be invariant to subtle appearance changes, we also augment pixel RGB values into sampled features on the ray. This is inspired by IBRNet [41] where they attach the pixel RGB to the input to the MLP for better appearance modeling. We compute the mean and variance for pixel RGB values in the same way as for deep features. In all, the augmented feature \(\bar{V}\) is represented as a concatenation of \(\{V^{\text{avg}}, V^{\text{var}}, R^{\text{avg}}, G^{\text{avg}}, B^{\text{avg}}, R^{\text{var}}, G^{\text{var}}, B^{\text{var}}\}\), as shown in Fig. 3. The sampled augmented features are passed into NeRF MLPs (Eq. 3) to generate the density and color. We use volumetric rendering to produce the final pixel color and depth,

\[
\hat{C}(r) = \sum_{i=1}^{N_p} T_i \alpha_i e_i, D(r) = \sum_{i=1}^{N_p} T_i \alpha_i t_i, \tag{5}
\]

where \(T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta t_i), \alpha_i = 1 - \exp(-\sigma_i \delta t_i), t_i\) is the distance between sampled \(i\)th point between the camera, \(\delta t_i\) is the distance between sampled points of rays.

### 3.3. Estimating Scene Geometry

We use an opacity field to model the scene geometry. The opacity field is a volumetric representation that reflects the probability of an object’s presence in a particular area, i.e., if there is an object that cannot be seen through, the opacity field would be 1.0 in that area. To generate the opacity field, we follow the same process of augmenting features in the detection branch as we do in the NeRF branch. A key ingredient to the approach is sharing the G-MLP learned
in the NeRF branch with the detection branch. This enables two important capabilities. Firstly, the shared G-MLP subtly connects the two branches, allowing gradients from the NeRF branch to back-propagate and benefit the detection branch during training. Secondly, during inference, we can directly input the augmented volume features of 3D detection into the shared G-MLP since both inputs from two branches are augmented features. The output of G-MLP is the density represented as \( \sigma \), where \( \sigma(p) \in [0, \infty] \). Note that \( p \) is the center position of each voxel in the volume of the detection branch.

Next, we aim to transform the density field into the opacity field by \( \alpha(p) = 1 - \exp(-\sigma(p) \times \delta_t) \). However, it is infeasible to calculate \( \delta_t \) as we cannot decide the ray direction and calculate the point distance within the undirected volume in the detection branch. Here we subtly take advantage of the uniform distribution of the voxels in the space. Thus, the \( \delta_t \) in the volumetric rendering equation can be canceled out as it is a constant. So obtaining opacity can be reduced to \( \alpha(p) = 1 - \exp(-\sigma(p)) \). After generating the opacity field, we multiply it with the feature grid \( V_{avg} \) for 3D detection, denoted as \( \alpha(p) \times V_{avg}(p) \).

### 3.4. 3D Detection Head and Training Objectives

Our geometry-aware volumetric representation can fit to most detection heads. For fair comparison and the simplicity, we use the same indoor detection head as ImVoxelNet [31], in which we select 27 location candidates for each object and we use three convolution layers to predict the categories, the locations and the centerness.

We jointly train detection and NeRF branches end-to-end. No per-scene optimization for the NeRF branch is performed in test time. For the detection branch, we supervise training with ground truth 3D bounding boxes following ImVoxelNet [31], which computes three losses: focal loss for classification \( L_{cls} \), centerness loss \( L_{cntr} \), and localisation loss \( L_{loc} \). For the NeRF branch, we use a photo-metric loss \( L_c = ||C(r) - C_{gt}(r)||_2 \). When depth ground truth is used, we can further supervise the expected depth of the scene geometry as \( L_d = ||D(r) - D_{gt}(r)|| \) where \( D(r) \) is computed with Eq. 5. The final loss \( L \) is given by

\[
L = L_{cls} + L_{cntr} + L_{loc} + L_c + L_d. \tag{6}
\]

Even though we optionally use depth during training, it is not required in inference. Our trained network is generalizable to new scenes which are never seen during training.

### 4. Experimental Results

#### Implementation details

Our detection branch mainly follows ImVoxelNet, including backbones, detection head, resolutions and training recipe etc. Please refer to supplemental material for more details.

Our implementation is based on MMDetection3D [4]. To the best of our knowledge, we are the first to implement NeRF in MMDetection3D. We are also the first to conduct NeRF-style novel view synthesis and depth estimation on the whole ScanNet dataset, while prior works only test on a small number of scenes [44, 43]. The code will be released for future research.

#### 4.1. Main results

**Quantitative results.** We compare NeRF-Det with point-cloud based methods [42, 11, 27], RGB-D based methods [11, 9] and the state-of-the-art RGB-only method ImVoxelNet [31] on ScanNet, as shown in Table 1.

With ResNet50 as the image backbone, we observe that NeRF-Det-R50-1x outperforms ImVoxelNet-R50-1x by 2.0 mAP. On top of that, NeRF-Det with depth supervision, denoted as NeRF-Det-R50-1x*, further improves the detection performance by 0.6 mAP compared to only RGB supervision NeRF-Det-R50-1x.

We denote the total training iterations of ImVoxelNet from the official code as 1x in the Tab. 1. Yet the 1x setting only iterates through each scene roughly 36 times, which is far from sufficient for optimizing the NeRF branch, which requires thousands of iterations to optimize one scene, as indicated in [21, 41, 1, 18]. Thus, we further conduct experiments with 2x training iterations to fully utilize the potential of NeRF, and we observe that NeRF-Det-R50-2x reaches 52.0 mAP, surpassing ImVoxelNet by 3.6 mAP under the same setting (ImVoxelNet-R50-2x). It is worth mentioning that we do not introduce any extra data/labels to get such an improvement. If we further use depth supervision to train the NeRF branch (NeRF-Det-R50-2x*), the detection branch is further improved by 1.3 mAP@.50 as shown in Tab. 3. This validates better geometry modeling (via depth supervision) could be helpful to the 3D detection task. While NeRF-Det provides an efficient method to incorporate depth supervision during the training process, introducing depth supervision in ImVoxelNet is difficult.

Moreover, when substituting ResNet50 with ResNet101, we achieve 52.9 mAP@.25 on ScanNet, which outperforms ImVoxelNet on the same setting over 3.9 points. With depth supervision, NeRF-Det-R101-2x* reduces the gap between RGB-based method ImVoxelNet [31] and point-cloud based method VoteNet [27] by half (from 10 mAP \( \rightarrow \) 5 mAP). Besides, we conduct experiments on the ARKitScenes (see Tab. 2). The 3.1 mAP improvement further demonstrates the effectiveness of our proposed method.

**Qualitative results.** We visualize the predicted 3D bounding boxes from NeRF-Det-R50-2x on scene meshes, as shown in Fig. 4. We observe that the proposed method gets accurate detection prediction even on the ex-
Table 1: 3D Detection with multi-view RGB inputs on ScanNet. The first block of the table includes point-cloud based and RGBD-based methods, and the rest are multi-view RGB-only detection methods. † means our reproduction of ImVoxelNet [31] using the official code. * indicates the NeRF-Det with depth supervision. 1x and 2x refer that we train the model with the same and two times of the training iteration wrt. the original iterations of ImVoxelNet, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>cab</th>
<th>bed</th>
<th>chair</th>
<th>sofa</th>
<th>tabl</th>
<th>door</th>
<th>wind</th>
<th>bkshfr</th>
<th>pic</th>
<th>cntr</th>
<th>desk</th>
<th>cart</th>
<th>frig</th>
<th>shower</th>
<th>tv</th>
<th>sink</th>
<th>bath</th>
<th>ofurn</th>
<th>mAP@.25</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImVoxelNet-R50-1x</td>
<td></td>
<td></td>
<td>31.6</td>
<td>81.8</td>
<td>74.4</td>
<td>69.3</td>
<td>53.6</td>
<td>29.7</td>
<td>12.9</td>
<td>50.0</td>
<td>1.3</td>
<td>32.6</td>
<td>69.2</td>
<td>12.7</td>
<td>54.6</td>
<td>31.8</td>
<td>93.1</td>
<td>55.8</td>
<td>68.2</td>
</tr>
<tr>
<td>ImVoxelNet-R50-2x*</td>
<td>34.5</td>
<td>83.6</td>
<td>72.6</td>
<td>71.6</td>
<td>54.2</td>
<td>30.3</td>
<td>14.8</td>
<td>42.6</td>
<td>0.8</td>
<td>40.8</td>
<td>65.3</td>
<td>18.3</td>
<td>52.2</td>
<td>40.9</td>
<td>90.4</td>
<td>53.3</td>
<td>74.9</td>
<td>33.1</td>
<td>48.4</td>
</tr>
<tr>
<td>NeRF-Det-R50-1x</td>
<td>37.7</td>
<td>84.1</td>
<td>74.5</td>
<td>71.8</td>
<td>54.2</td>
<td>34.2</td>
<td>17.4</td>
<td>51.6</td>
<td>0.1</td>
<td>54.2</td>
<td>71.3</td>
<td>16.7</td>
<td>54.5</td>
<td>55.0</td>
<td>92.1</td>
<td>50.7</td>
<td>73.8</td>
<td>34.1</td>
<td>51.8</td>
</tr>
<tr>
<td>NeRF-Det-R50-2x*</td>
<td></td>
<td></td>
<td>38.5</td>
<td>86.1</td>
<td>76.6</td>
<td>56.9</td>
<td>35.1</td>
<td>18.6</td>
<td>47.5</td>
<td>0.0</td>
<td>44.4</td>
<td>65.5</td>
<td>19.6</td>
<td>58.2</td>
<td>32.8</td>
<td>92.3</td>
<td>40.1</td>
<td>77.6</td>
<td>28.0</td>
</tr>
<tr>
<td>NeRF-Det-R101-2x*</td>
<td>36.8</td>
<td>85.0</td>
<td>77.0</td>
<td>73.5</td>
<td>56.9</td>
<td>36.7</td>
<td>14.3</td>
<td>48.1</td>
<td>0.8</td>
<td>49.7</td>
<td>68.3</td>
<td>23.5</td>
<td>54.0</td>
<td>60.0</td>
<td>96.5</td>
<td>49.3</td>
<td>78.4</td>
<td>38.4</td>
<td>52.9</td>
</tr>
<tr>
<td>NeRF-Det-R101-2x*</td>
<td>37.6</td>
<td>84.9</td>
<td>76.2</td>
<td>75.7</td>
<td>57.7</td>
<td>36.4</td>
<td>17.8</td>
<td>47.0</td>
<td>2.5</td>
<td>49.2</td>
<td>52.0</td>
<td>29.2</td>
<td>68.2</td>
<td>49.3</td>
<td>97.1</td>
<td>57.6</td>
<td>83.6</td>
<td>35.9</td>
<td>53.3</td>
</tr>
</tbody>
</table>

Table 2: Comparison experiments "whole-scene" of ARKITScenes validation set.

<table>
<thead>
<tr>
<th>Methods</th>
<th>cab</th>
<th>frig</th>
<th>shlf</th>
<th>stove</th>
<th>bed</th>
<th>sink</th>
<th>wshr</th>
<th>tolt</th>
<th>thbth</th>
<th>oven</th>
<th>dshwr</th>
<th>frplce</th>
<th>stood</th>
<th>chr</th>
<th>btle</th>
<th>TV</th>
<th>sofa</th>
<th>mAP@.25</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImVoxelNet-R50</td>
<td>32.2</td>
<td>34.3</td>
<td>4.2</td>
<td>4.0</td>
<td>64.7</td>
<td>20.5</td>
<td>15.8</td>
<td>68.9</td>
<td>80.4</td>
<td>9.9</td>
<td>4.1</td>
<td>10.2</td>
<td>0.4</td>
<td>2.2</td>
<td>11.6</td>
<td>3.1</td>
<td>35.6</td>
<td>23.6</td>
</tr>
<tr>
<td>NeRF-Det-R50</td>
<td></td>
<td>40.7</td>
<td>4.9</td>
<td>0.0</td>
<td>69.3</td>
<td>24.4</td>
<td>17.3</td>
<td>75.1</td>
<td>84.6</td>
<td>14.0</td>
<td>7.4</td>
<td>10.9</td>
<td>0.2</td>
<td>4.0</td>
<td>14.2</td>
<td>5.3</td>
<td>44.0</td>
<td>26.7 (+3.1)</td>
</tr>
</tbody>
</table>

Figure 4: Qualitative results of the predicted 3D bounding box on top of NeRF-Det-R50-2x. Note that our approach takes only posed RGB images as input, the reconstructed mesh is only for visualization purpose.

tremendously dense scenes, e.g., the first row and the left of the third row. The chairs are crowded in the room, and some of which are inserted into tables and are heavily occluded. Yet our method is able to detect them accurately. On the other hand, NeRF-Det can also tackle multiple scales, as shown in the second row and the right side of the third row, in which
there are variant objects with difference sizes, like garbage bins, chairs, tables, doors, windows, and sofas etc.

**Analysis of scene geometry modeling.** As stated in the method section, we mitigate the ambiguity of the volume representation by learning an opacity field. Furthermore, we explore different scene geometry modeling methods, from using depth map to cost volume [35, 45], in Tab. 3.

**Using the Depth Map.** In this experiment, we assume we have access to depth maps during both training and inference. When building the voxel feature grid, instead of scattering the features on all points along the ray, we only place features to a single voxel cell according to the depth maps. Intuitively, this leads to less ambiguity in the volume representation, so we should observe better performance for detection. As a proof of concept, we first use ground-truth depth that comes with the dataset. This serves as an upper-bound for NeRF-Det as it provides a perfect geometry model for the scene geometry. As shown in the gray part of Tab. 3, with the opacity field modeled using NeRF, the performance is significantly improved by +3.6 mAP@.25 and +2.5 mAP@.50 compared to the baseline. After using depth supervision, NeRF-Det is able to achieve larger improvement with +3.7 mAP@.50 (the last row). As shown in Tab. 3, our method of using NeRF to model scene geometry is more effective than using predicted depth or cost volume.

**Comparison to NeRF-then-Det method.** We compare our proposed NeRF-Det, a joint NeRF-and-Det method, to NeRF-RPN [14] which is a NeRF-then-Det method, as shown in Tab. 4. We choose 4 scenes as validation set. We use the official code and provided pre-trained models of NeRF-RPN to evaluate AP. Experiments in the first group demonstrate that our proposed joint-NeRF-and-Det paradigm is more effective than first-NeRF-then-Det method NeRF-RPN, with much faster speed.

The second group of Tab. 4 shows that directly using our model (NeRF-Det-R50-2x in Tab. 1) has drastic improvements relative to NeRF-RPN. Although our model is
trained on large-scale dataset, we emphasize that this is our advantages since it is hard to apply NeRF-RPN on the whole ScanNet (∼1500 scenes) given the heavy overhead.

Is NeRF branch able to learn scene geometry? We hypothesize that the better detection performance comes from better geometry. To verify this, we perform novel view synthesis and depth estimation from the prediction of the NeRF branch. The underlying assumption is that if our model has correctly inferred the scene geometry, we should be able to synthesize RGB and depth views from novel camera positions. We first visualize some of the synthesized images and depth estimation in Fig. 5. The image and depth map quality look reasonable and reflects the scene geometry well.

Quantitatively, we evaluate novel-view synthesis and depth estimation by following the protocol of IBRNet [41]. We select 10 novel views and randomly sample 50 nearby source views. Then, we average the evaluation results of the 10 novel views for each scene, and finally report average results on all 312 scenes in the validation set, as shown in Tab. 9. Although novel-view synthesis and depth estimation are not the main focus of this paper, we achieve an average of 20+ PSNR for novel view synthesis, without per-scene training. For depth estimation, we achieve an RMSE of 0.756 on all scenes. While this performance falls behind state-of-the-art depth estimation methods, we note that [43] reported average RMSE of 1.0125 and 1.0901 using Colmap [33] and vanilla NeRF [21] for depth estimation on selected scenes in ScanNet. This comparison verifies that our method can render reasonable depth.

4.2. Ablation Study

We conduct multiple ablation studies on how different components affect the performance of NeRF-Det, including different feature sampling strategies, whether to share G-MLP, different losses and different features feed into the NeRF branch. Besides, although novel-view synthesis is not our focus in this paper, we also provide some analysis for the novel-view synthesis results coming out of the NeRF branch, and show how the NeRF branch is influenced by the detection branch during joint training. All experiments in the ablation studies are based on NeRF-Det-R50-1x*.

Ablation on G-MLP and feature sampling strategy. As indicated in Sec. 3.2, the key ingredient in our pipeline is a shared G-MLP which enables the constraint of multi-view consistency to be propagated from NeRF branch to detection branch. We conduct an ablation study as shown in Tab. 5. Without shared G-MLP, the performance drops drastically from 50.1 to 48.1 at mAP@.25, shown in the fifth row of Tab. 3. In this case, the influence of multi-view consistency is only propagated into image backbone, significantly limiting the improvement created by NeRF.

Moreover, as mentioned in Sec. 3.2, we sample point features along the ray from the multi-view image features instead of the low-resolution volume features. This ablation is shown in Tab. 3. We observe that with shared G-MLP, both approaches outperform the baseline ImVoxelNet, and sampling from image features yields better performance (+0.7 in mAP@0.25) than sampling from volume features. For the novel-view synthesis task using NeRF branch, sampling from image features achieves 20.51 in PSNR comparing to...
18.93 with volume sampling.

The fact that the performance of the NeRF branch is proportional to the performance of the detection branch also indicates that better NeRF optimization could lead to better detection results.

Ablation study on different loss. We study how different losses work for the NeRF branch, as shown in Tab. 6. It shows that with only photo-metric loss, the performance is closed to purely using depth supervision (third row) in term of mAP@.50, indicating that the multi-view RGB consistency already provides sufficient geometry cues to enable the NeRF branch to learn geometry. When using both photo-metric loss and depth loss, the performance can be further improved. When neither photo-metric loss nor depth loss is used (last row), the performance falls back to that of a cost-volume based method. The performance is dropped by 1.2 mAP@.25 and 0.5 mAP@.50, which demonstrates that the NeRF branch is more effective.

Ablation study on different features. We then study how different features affect performance, as shown in Tab. 7. The experiment shows that introducing variance features improves performance significantly – over 0.7 mAP@.25 and 0.6 mAP@.50 compared to using only average features, which demonstrates that variance features indeed provide a good geometry prior. Moreover, incorporating image features also improves performance, indicating that low-level color information also provides good geometry cues.

Ablation study on geometry descriptions. We clarify that all the input of G-MLP is the augmented features, while the output opacity grid is multiplied with the average feature grids for fair comparison to ImVoxelNet. We provide more alternatives in Tab. 8. We can observe that multiplying density with features achieves similar performance to that of opacity, which is reasonable as opacity is reduced to simple non-linearity of density in our case.

Ablation Study on detection branch affecting novel view synthesis. We keep the target views and source views the same with and without the detection branch. Results are shown in Tab. 9. While the NeRF branch significantly improves 3D detection by improving scene geometry modeling, the detection branch does not benefit the NeRF branch. In fact, disabling the detection branch brings a 0.43db improvement. We hypothesize that the detection branch is prone to erase low-level details which are needed for the NeRF, which we aim to address in our future work.

5. Conclusion

In this paper, we present NeRF-Det, a novel method that uses NeRF to learn geometry-aware volumetric representations for 3D detection from posed RGB images. We deeply integrate multi-view geometry constraints from the NeRF branch into 3D detection through a subtle shared geometry MLP. To avoid the large overhead of per-scene optimization, we propose leveraging augmented image features as priors to enhance the generalizability of NeRF-MLP. In addition, we sample features from high resolution images instead of volumes to address the need for high-resolution images in NeRF. Our extensive experiments on the ScanNet and ARKITScene datasets demonstrate the effectiveness of our approach, achieving state-of-the-art performance for indoor 3D detection using RGB inputs. Notably, we observe that the NeRF branch also generalizes well to unseen scenes. Furthermore, our results highlight the importance of NeRF for 3D detection, and provide insights into the key factors for achieving optimal performance in this direction.

6. Acknowledgment

We sincerely thank Chaojian Li for the great help on Neural-Recon experiments, Benran Hu for the valuable advice on the NeRF-RPN experiments, Feng (Jeff) Liang and Hang Gao for the insightful discussions, as well as Patrick Rim, Matthew Yu and Jinhyung Park for the paper proofreading.
References


[22] Yinyu Nie, Ji Hou, Xiaoguang Han, and Matthias Nießner. Rfd-net: Point scene understanding by semantic instance reconstruction. In CVPR, 2021. 2


