VGOS: Voxel Grid Optimization for View Synthesis from Sparse Inputs

Jiakai Sun, Zhanjie Zhang, Jiafu Chen, Guangyuan Li, Boyan Ji, Lei Zhao∗, Wei Xing∗
Zhejiang University
{csjk, cszzj, chenjiafu, ji_by, cszhl, wxing}@zju.edu.cn, lgy1428275037@163.com

Abstract

Neural Radiance Fields (NeRF) has shown great success in novel view synthesis due to its state-of-the-art quality and flexibility. However, NeRF requires dense input views (tens to hundreds) and a long training time (hours to days) for a single scene to generate high-fidelity images. Although using the voxel grids to represent the radiance field can significantly accelerate the optimization process, we observe that for sparse inputs, the voxel grids are more prone to overfitting to the training views and will have holes and floaters, which leads to artifacts. In this paper, we propose VGOS, an approach for fast (3-5 minutes) radiance field reconstruction from sparse inputs (3-10 views) to address these issues. To improve the performance of voxel-based radiance field in sparse input scenarios, we propose two methods: (a) We introduce an incremental voxel training strategy, which prevents overfitting by suppressing the optimization of peripheral voxels in the early stage of reconstruction. (b) We use several regularization techniques to smooth the voxels, which avoids degenerate solutions. Experiments demonstrate that VGOS achieves state-of-the-art performance for sparse inputs with super-fast convergence. Code will be available at https://github.com/SJoJoK/VGOS.

1 Introduction

In 3D vision, novel view synthesis is a long-standing task that aims to synthesize a target image with an arbitrary target camera pose from given source images and their camera poses. Recently, Neural Radiance Fields (NeRF) [Mildenhall et al., 2020], a learning-based neural implicit representation, have emerged as a powerful tool yielding high-fidelity results on this task. However, NeRF requires tens to hundreds of dense inputs and hours to days of training time to get high-quality results. When considering real-world applications such as autonomous driving, AR/VR, and robotics that lack dense data and require real-time performance, NeRF’s limitations of relying on dense input views and lengthy optimization time are even more magnified.

To speed up the optimization, recent works [Sun et al., 2022a; Chen et al., 2022a; Müller et al., 2022; Yu et al., 2022] utilize explicit data structures to represent the radiance field, reducing the training time to minutes. However, these data structures designed to shorten the optimization process of the radiance field do not consider the performance for sparse inputs and still require dense inputs to obtain high-quality results.

To improve NeRF’s performance on sparse inputs, several works [Chen et al., 2021; Yu et al., 2021; Liu et al., 2022] first pre-train a model on the multi-view images dataset of many scenes and then use the pre-trained model and the optional per-scene fine-tuning process to synthesize novel views for sparse inputs. Although these works have obtained promising results, acquiring pre-training data may be expensive, and the pre-training time is also very long. In addition, these methods may not generalize well for domains not covered
by the pre-training data. Other works [Jain et al., 2021; Niemeyer et al., 2022; Deng et al., 2022; Xu et al., 2022] train
the model from scratch for every new scene. To enhance
the performance for sparse inputs, some works [Jain et al., 2021; Niemeyer et al., 2022] regularize appearance or semantics by
introducing models pre-trained on large-scale image datasets.
Although these methods can generate high-quality rendering
results, their results suffer from incorrect geometry, and the
pre-trained model increases the method’s complexity. Be-
sides, some works leverage depth maps to supervise the opti-
mization [Deng et al., 2022] or augment training images [Xu
et al., 2022]. The addition of depth information helps the
model obtain relatively correct geometry from sparse inputs,
but depth maps are not as easy to obtain as RGB images.

To overcome the aforementioned shortcomings and limita-
tions, we present an approach for fast radiance field recon-
struction from sparse inputs, namely VGOS. As shown in the
Fig. 1, our model achieves on-par high-quality results after
minutes of training time compared with the previous state-of-
the-art approaches, which take hours of per-scene optimiza-
tion or days of generalizable pre-training. Specifically, we
directly optimize the voxel grids representing the radiance
field [Sun et al., 2022a]. However, for sparse inputs, the re-
construction of the radiance field (a) is more prone to over-
fitting to the training views, and (b) the voxel grids will have
holes and floaters. In order to solve these two problems, we
propose two methods: (a) incremental voxel training strategy
and (b) voxel smoothing method. With the improvement of
the training strategy and the new regularization method, our
model achieves state-of-the-art performance for sparse input
without any pre-trained model and with only RGB images as
input.

Specifically, the incremental voxel training strategy is to
freeze the optimization of peripheral voxels at the early stage
of training and gradually thaw the peripheral voxels as the
training progresses. This strategy prevents the voxels close
to the cameras’ near planes from overfitting to the training
views, thus boosting the quality of radiance field reconstruc-
tion. The voxel smoothing method helps prevent degenerate
solutions by regularizing the depth maps rendered from un-
observed viewpoints [Niemeyer et al., 2022] and penalizing
the sharpness inside the voxel grids with the proposed color-
aware voxel smoothness loss.

In summary, the main contributions of our work can be
summarized as follows:

- We propose an incremental voxel training strategy to
  prevent the voxels from overfitting to the training views
  by suppressing the optimization of peripheral voxels in
  the early stage of radiance field reconstruction.
- We propose a voxel smoothing method to avoid incorrect
  geometry by regularizing the dense voxel grids and
  utilizing depth smooth loss, which eliminates holes and
  floaters in the voxel grids, thus improving the quality of
  radiance field reconstruction in sparse input scenarios.
- Extensive experiments on different datasets demonstrate
  that our proposed model, even without any pre-trained
  model and extra inputs, achieves one to two orders
  of magnitude speedup compared to state-of-the-art ap-
  proaches with on-par novel view synthesis quality.

2 Related Work

2.1 Novel View Synthesis

Novel view synthesis is a time-honored problem at the in-
tersection of computer graphics and computer vision. Previ-
ous works use light field [Levoy and Hanrahan, 1996; Shi
et
al., 2014] and lumigraph [Gortler et al., 1996; Buehler et al., 2001] to synthesize novel views by interpolating the input images. Moreover, explicit representations, such as meshes [Debevec et al., 1996; Hu et al., 2021], voxels [Sitzmann et al., 2019a], and multiplane images [Mildenhall et al., 2019; Flynn et al., 2019], are introduced into this task. Recently, several works [Sitzmann et al., 2019b; Niemeyer et al., 2020; Yariv et al., 2020; Mildenhall et al., 2020] have introduced implicit representation and corresponding differentiable rendering methods due to their convenient end-to-end optimization and high-quality results. Among these works, Neural Radiance Fields [Mildenhall et al., 2020] (NeRF) achieve photorealistic rendering results by representing the radiance field as a multi-layer perceptron (MLP) and differentiable volume rendering method. Subsequent works have improved the performance of NeRF in many aspects, such as training on multi-resolution images [Barron et al., 2021], unconstrained images [Martin-Brualla et al., 2021; Chen et al., 2022b], unbounded scenes [Barron et al., 2022], dark scenes [Mildenhall et al., 2022], and deforming scenes [Park et al., 2021a; Park et al., 2021b]. However, NeRF and these variants require dense inputs to generate high-quality results, which is not always available in real-world applications.

2.2 Fast Radiance Field Reconstruction

Although NeRF can achieve high-fidelity rendering, it takes hours to days of training to reconstruct the radiance field for new scenes. Several works [Sun et al., 2022a; Yu et al., 2022; Chen et al., 2022a; Müller et al., 2022] use explicit or hybrid radiance field representations to reduce training time to a few minutes. DVGO [Sun et al., 2022a] uses dense voxel grids and a shallow MLP to represent the radiance field, while DVGOv2 [Sun et al., 2022b] re-implements some operations in CUDA to achieve improved performance. Plenoxels [Yu et al., 2022] uses a sparse voxel grid and coefficients of spherical harmonic for view-dependent colors to realize a fully explicit representation. TensoRF [Chen et al., 2022a] achieves efficient radiance field reconstruction by decomposing the volume field and modeling the low-rank components. Instant-NGP [Müller et al., 2022] represents the radiance field as a multiresolution hash table and small neural networks, achieving convincing acceleration using C/C++ and fully-fused CUDA kernels. However, these acceleration methods do not reduce the dependence of radiance field reconstruction on dense inputs, while our approach performs high-quality novel view synthesis from sparse inputs in minutes of optimization.

2.3 Sparse Input Radiance Field Reconstruction

Many methods have been proposed to overcome the NeRF’s dependence on dense inputs. Several works [Yu et al., 2021; Chen et al., 2021; Chibane et al., 2021] compensate for information scarcity from sparse inputs by pre-train a conditional model of the radiance field. PixelNeRF [Yu et al., 2021] and SRF [Chibane et al., 2021] train convolutional neural network (CNN) encoders to extract features of the input images. MVSNerf [Chen et al., 2021] uses a 2D CNN to get 2D image features from the input images and then uses plane sweeping to obtain a cost volume which will be processed by a 3D CNN. These methods get promising results, but pre-training on multi-view image datasets is expensive and time-consuming. Besides, most of these methods require fine-tuning on new scenes, and the performance of these methods will decline when the data domain changes at test time.

On the other hand, a line of works [Jain et al., 2021; Niemeyer et al., 2022; Deng et al., 2022; Xu et al., 2022] uses models pre-trained on large-scale image datasets and depth maps to train the radiance field from scratch. DietNeRF [Jain et al., 2021] uses prior knowledge about scene semantics learned by pre-trained CLIP ViT [Radford et al., 2021] to constrain a 3D representation. RegNeRF [Niemeyer et al., 2022] uses pre-trained Real-NVP [Dinh et al., 2017] to regularize the colors predicted at unseen viewpoints. DSNeRF [Deng et al., 2022] takes depth maps as input to supervise the reconstruction of the radiance field. Besides, SinNeRF [Xu et al., 2022] uses global structure prior provided by pre-trained DINO-ViT [Caron et al., 2021] and augments data using depth maps.

In addition, InfoNeRF [Kim et al., 2022] is a prior-free model without any extra inputs, which regularizes the reconstruction of the radiance field by minimizing ray entropy and reducing information gain. However, this scheme requires the weights of all sampled points on rays. Therefore, reducing the number of sampling points is difficult, which is commonly used in NeRF acceleration approaches.

In contrast, our approach is $10 \times 100 \times$ faster than state-of-the-art approaches with comparable high-quality results without expensive and time-consuming pre-train process and without additional input or pre-trained model to increase complexity.

3 Method

Our approach, which builds upon DVGOv2 [Sun et al., 2022b] (Sec. 3.1), performs fast radiance field reconstruction from sparse RGB input images without any pre-trained model. We find that unexpected overfitting and holes and floaters of the voxel grids lead to degenerate solutions for sparse inputs. To prevent the radiance field from overfitting to the input views, we introduce an incremental voxel training strategy (Sec. 3.2) that suppresses the optimization of peripheral voxels. Moreover, we smooth the voxels (Sec. 3.3) by regularizing the predicted geometry from sampled views and the shape of the explicit radiance field. We depict an overview of our approach in Fig. 2.

3.1 Background

Neural Radiance Fields

A radiance field is a function that maps a 3D position $x$ and a viewing direction $d$ to the corresponding view-dependent emission color $c$ and volume density $σ$. NeRF [Mildenhall et al., 2020] uses MLP to parameterize this function:

$$\text{MLP}_{θ}(x, d) \rightarrow (c, σ),$$ (1)

where $θ$ is the learnable MLP parameters. Note that the positional encoding [Tancik et al., 2020] is applied to $x$ and $d$ before the MLP to enable the MLP to represent higher frequency functions.
To synthesize novel views, NeRF uses volume rendering techniques. To be specific, the rendered color \( C(r) \) of a target pixel is obtained by integrating colors and densities between near and far bounds \( t_n \) and \( t_f \) along a ray \( r(t) = o + td \) from the camera center \( o \) through the pixel along direction \( d \):
\[
C(r) = \int_{t_n}^{t_f} T(t) \sigma(r(t)) c(r(t), d) dt,
\]
where \( T(t) = \exp \left( -\int_{t_n}^{t} \sigma(r(s)) ds \right) \) is the accumulated transmittance along the ray from \( t_n \) to \( t \), and \( \sigma(\cdot) \) and \( c(\cdot, \cdot) \) indicate the density and color prediction of the radiance field \( F_\Theta \), respectively.

In practice, the integral is approximated by quadrature:
\[
\hat{C}(r) = \sum_{i=1}^{N} T_i \left( 1 - \exp(-\sigma_i \delta_i) \right) c_i,
\]
where \( T_i = \exp \left( -\sum_{j=1}^{i-1} \sigma_j \delta_j \right) \), \( N \) is the number of sampled points along the ray \( r \), \( \sigma_i \), \( c_i \) are the density and color of the \( i \)th sampled point, and \( \delta_i = t_{i+1} - t_i \) is the distance between adjacent samples.

NeRF’s MLP can be optimized over a set of input images and their camera poses by minimizing the photometric MSE between the ground truth pixel color \( C_{GT}(r) \) and the rendered color \( \hat{C}(r) \):
\[
\mathcal{L}_{\text{Photometric}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left\| \hat{C}(r) - C_{GT}(r) \right\|^2,
\]
where \( \mathcal{R} \) denotes a set of rays.

### Direct Voxel Grid Optimization

It is time-consuming to query the color and density of each sampled point through MLP, so DVGO [Sun et al., 2022a] is proposed to accelerate this process by representing the radiance field as voxel grids. Such an explicit scene representation is efficient to query color \( c \) and density \( \sigma \) for any 3D position \( x \) with trilinear interpolation:
\[
\hat{\sigma} = \text{interp}(x, V^{\text{density}}), \quad c = \text{interp}(x, V^{\text{rgb}}), \quad \hat{\sigma} = \log(1 + \exp(\hat{\sigma} + b)),
\]
where the shift \( b = \log \left( \left( 1 - \alpha_{\text{init}} \right)^{-\frac{1}{3}} - 1 \right) \) is the bias term determined by hyperparameter \( \alpha_{\text{init}} \) and voxel size \( s \), \( V^{\text{density}} \) and \( V^{\text{rgb}} \) are the voxel grids storing raw density \( \sigma \) before applying the density activation and color, respectively.

In practice, DVGO uses a coarse-to-fine training strategy. In the fine stage, a shallow MLP is used to process viewing-direction \( d \) and feature \( f \) from a feature voxel grid \( V^{\text{feature}} \) to model view-dependent color emission.
\[
c = \text{MLP}_0(\text{interp}(x, V^{\text{feature}}), x, d),
\]
where \( \Theta \) is the learnable MLP parameters.

Subsequent work, namely DVGOv2 [Sun et al., 2022b], improve DVGO by re-implementing part of the Pytorch operations with CUDA and extending it to support forward-facing and unbounded inward-facing capturing.

### 3.2 Incremental Voxel Training

Although DVGO uses various techniques to avoid degenerate solutions, the radiance field will overfit to input views for sparse scenarios. Specifically, we find that for sparse inputs, the peripheral voxels close to the camera near planes have high density values at the initial stage of training to reproduce the input views. However, the high density value of the outer voxels hinders the optimization of the inner voxels, which makes it difficult for the radiance field to converge to the correct geometry so that the quality of rendering results at novel views will decline.

We propose a simple yet non-trivial incremental voxel training strategy to solve the above-mentioned problem. For the voxel grids \( V \in \mathbb{R}^{C \times N_x \times N_y \times N_z} \) representing the radiance field, where \( C \) is the dimension of the modality, \( N_x, N_y, N_z \) is the total number of voxels, we define an expanding bounding box \( B \) whose corner points are \( P_{\text{min}}, P_{\text{max}} \in \mathbb{R}^3 \):
\[
P_{\text{min}} = (P_{\text{min,init}} \times (1 - r(i))) \odot (N_x, N_y, N_z), \quad P_{\text{max}} = (P_{\text{max,init}} \times (1 - r(i)) + r(i)) \odot (N_x, N_y, N_z),
\]
where \( P_{\text{min,init}} \in [0, 1]^3 \) and \( P_{\text{max,init}} \in [0, 1]^3 \) are the initial ratio of the expanding bounding box \( B \), and \( r(i) = \min \left( \frac{M}{M_i}, 1 \right) \) determine the range of the bounding box \( B \), where \( i \) is the current training iteration and \( M \) is the pre-defined max iteration steps of the increment process. We only optimize the voxels inside the bounding box \( B \); this training strategy freezes the optimization of the peripheral voxels in the early training, avoiding overfitting and leading to better rendering results at novel views. We set \( M = 256 \) for all scenes, \( P_{\text{min,init}} = (0.2, 0.2, 0.2), P_{\text{max,init}} = (0.8, 0.8, 0.8) \) for bounded inward-facing scenes and \( P_{\text{min,init}} = (0, 0, 0.995), P_{\text{max,init}} = (1, 1, 1) \) and \( M = 256 \) for forward-facing scenes in our experiments.

### 3.3 Voxel Smoothing

Although we use the incremental voxel training strategy to alleviate the overfitting, if we only use the photometric MSE loss (Equation (4)) to supervise the training from sparse inputs, the radiance field will still overfit to the input views.
To solve this problem, we propose a novel color-aware voxel smoothness loss on the dense voxel grids and utilize the depth smoothness loss on the sampled views to smooth the voxels.

**Regularization on Dense Voxels**

To prevent the outliers and noises in the explicit model, previous works [Yu et al., 2022; Chen et al., 2022a; Sun et al., 2022b] utilize total variation (TV) loss [Rudin and Osher, 1994]:

\[
L_{TV}(V) = \sum_{v \in V} \Delta(v),
\]

where \(\Delta(v)\) shorthand for the mean of the loss (L1, L2, or Huber Loss) between the value in voxel \(v\) and its six nearest-neighbor voxels and \(V\) indicates the voxel grids storing density, color or feature, which is indeed effective. However, these works calculate the TV loss of density and color separately, not taking advantage of the correlation between density and color in the explicit radiance field.

We observe that, in the radiance field, the density change is not smooth where the color changes sharply. According to the above observation, we propose color-aware total variance (CATV) loss, which uses the activated value in the color voxel grid to guide TV loss of the density voxel grid and is formulated as:

\[
F_{CA}(V, v) = \Delta_{\text{activate}}(v), \quad v \in V,
\]

\[
L_{CATV} = \sum_{v \in V^{\text{density}}} e^{-F_{CA}(V^{\text{feature}}, v)} \Delta(v),
\]

with \(\Delta_{\text{activate}}(v)\) indicates that the activated values are used calculating \(\Delta(v)\). In practice, we use L1 loss in \(F_{CA}\), and Huber Loss is in \(L_{CATV}\). Sigmoid is used in \(F_{CA}\) to normalize the feature values to \([0, 1]\) and align the choices of activation functions in DVGO. In Fig. 3, we show the differences between using \(L_{TV}\) and \(L_{CATV}\) to regularize \(V^{\text{density}}\).

To ensure flexibility, we use both \(L_{TV}\) and \(L_{CATV}\). Therefore, the color-aware voxel smoothness (CAVS) loss which is used to regularize the dense voxels is formulated as:

\[
L_{CAVS} = \lambda_{TVF}L_{TV}(V^{\text{feature}}) + \lambda_{TVD}L_{TV}(V^{\text{density}}) + \lambda_{CATV}L_{CATV},
\]

where \(\lambda_{TVF}\), \(\lambda_{TVD}\) and \(\lambda_{CATV}\) are the corresponding weights. Since computing \(L_{CAVS}\) is time-consuming, we implement it in CUDA kernel to speedup the process. Besides, we only backpropagate the gradient of \(L_{CATV}\) to \(V^{\text{density}}\). We set \(\lambda_{TVD} = 5 \cdot 10^{-4}\), \(\lambda_{TVF} = \lambda_{CATV} = 5 \cdot 10^{-5}\) in coarse-stage training and \(\lambda_{TVD} = 5 \cdot 10^{-3}\), \(\lambda_{TVF} = 10^{-5}\), \(\lambda_{CATV} = 5 \cdot 10^{-6}\) in fine-stage training for bounded inward-facing scenes. For
forward-facing scenes which only need fine-stage training, we set \( \lambda_{TVD} = 5 \cdot 10^{-5} \) and \( \lambda_{TVF} = \lambda_{CAV} = 5 \cdot 10^{-6} \).

**Regularization on Sampled Viewpoints**

The piecewise-smooth of geometry is a classic hypothesis in depth and disparity estimation [Scharstein and Szeliski, 2002]. Hence we utilize the depth smoothness (DS) loss introduced by RegNeRF [Niemeyer et al., 2022] on the unseen views to improve scene geometry.

To get unobserved views, we sample camera pose \( p \sim \pi \) where \( \pi \) is the distribution of camera poses if \( \pi \) is available. For bounded inward-facing scenes such as one from the Realistic Synthetic 360° dataset [Mildenhall et al., 2020], \( \pi \) is the uniform distribution over the hemisphere with the known radius. For forward-facing scenes like those from the LLFF dataset [Mildenhall et al., 2019], \( \pi \) is the uniform distribution over a 2D plane with given boundaries. If \( \pi \) is not available, we generate new poses by interpolating between input poses.

We can estimate depth \( \hat{d} \) along the ray \( r \) cast from the sampled camera pose the similar way we render color in Equation (3):

\[
\hat{d}(r) = \sum_{i=1}^{N} T_{i}(1 - \exp(-\sigma_{i}\delta_{i})) .
\]

By estimating depth from sets of neighboring rays, we can render depth patches and regularize them by the DS loss:

\[
L_{DS} = \frac{\lambda_{DS}}{|R|} \sum_{r_{c} \in R} \sum_{(x,y)} \| \nabla D(r_{c}, y) \|^2 ,
\]

where \( R \) indicates a set of rays cast from the sampled poses, \( D \) is the depth patch centered at \( r_{c} \), and \( \lambda_{DS} \) is the loss weight. In practice, finite difference formula is used to compute \( \nabla D \). We set \( \lambda_{DS} = 5 \cdot 10^{-4} \) in coarse-stage training and \( \lambda_{DS} = 10^{-5} \) in fine-stage training for bounded inward-facing scenes. For forward-facing scenes, we set \( \lambda_{DS} = 5 \cdot 10^{-4} \).

### 3.4 Total Loss Function

The total loss function of our model is given by:

\[
L_{Total} = L_{Photometric} + L_{CAVS} + L_{DS} .
\]

**Metrics.** We measure the mean of peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM) [Wang et al., 2004], and learned perceptual image patch similarity (LPIPS) [Zhang et al., 2018] to evaluate our model.

**LLFF Dataset.** The LLFF Dataset consists of 8 complex real-world scenes captured by a handheld cellphone. Each scene has 20 to 62 forward-facing images. We hold out 1/8 of the images as test sets following the standard protocol [Mildenhall et al., 2020] and report results for 3 input views randomly sampled from the remaining images.

**4 Experiments**

**4.1 Datasets and Evaluations**

We perform experiments on inward-facing scenes from the Realistic Synthetic 360° dataset [Mildenhall et al., 2020] and forward-facing scenes from the LLFF dataset [Mildenhall et al., 2019].

**Realistic Synthetic 360° Dataset.** The Realistic Synthetic 360° dataset contains path traced images of 8 synthetic scenes with complicated geometry and realistic non-Lambertian materials. Each scene has 400 images rendered from inward-facing virtual cameras with different viewpoints. Following the protocol of InfoNeRF [Kim et al., 2022], we randomly sample 4 views out of 100 training images as sparse inputs and evaluate the model with 200 testing images.

**4.2 Implementation Details**

We implement our model on the top of DVGOv2 codebase using Pytorch [Paszke et al., 2019]. Following DVGO, We use the Adam [Kingma and Ba, 2015] to optimize the voxel grids with the initial learning rate of 0.1 for all voxels and \( 10^{-3} \) for the shallow MLP and exponential learning rate decay is applied.

For scenes in the Realistic Synthetic 360° dataset, we train the voxel grids for 5K iterations with a batch size of \( 2^{13} \) rays for input views and \( 2^{14} \) rays for sampled views in both stages.

For scenes in the LLFF dataset, we train the voxel grids for 9K iterations with a batch size of \( 2^{12} \) rays for input views and \( 2^{14} \) rays for sampled views in only one stage.

Please refer to the supplementary material for more details.

**4.3 Comparisons**

Following InfoNeRF [Kim et al., 2022], the presented metrics for comparisons are the average score of five experiments with different viewpoint samples.

**Table 1: Quantitative comparison on Realistic Synthetic 360° in the 4-view setting.** The asterisk (*) denotes that early-stopping is used instead of the default setting. The star (⋆) denotes the generalizable pre-training time. Bold and underline indicate the best and the second-best values for each metric.

**Realistic Synthetic 360° Dataset**

We compare our model with NeRF [Mildenhall et al., 2020], DietNeRF [Jain et al., 2021], PixelNeRF [Yu et al., 2021], InfoNeRF [Kim et al., 2022], and DVGOv2 [Sun et al., 2022a; Sun et al., 2022b] on the Realistic Synthetic 360° dataset in the 4-view setting. Since PixelNeRF is pre-trained on the DTU dataset, we fine-tune it for 20K iterations similar to [Deng et al., 2022] for improved performance.

Tab. 1 presents the overall quantitative results, and Fig. 4 shows the qualitative results. As the baseline, NeRF has degenerate solutions for sparse inputs. DietNeRF and PixelNeRF outperform the baseline relatively by introducing pre-trained models. Although DVGOv2 aims to accelerate the reconstruction process, it achieves superior results to NeRF in
sparse input scenarios, which we observe as another advantage of explicit models. InfoNeRF outperforms the previous methods in terms of all image quality metrics. However, it takes twice the training time than NeRF and is unsuitable for common acceleration approaches since it requires the weights of all sampled points on rays. Our model achieves state-of-the-art performance with an outstanding convergence speed.

**LLFF Dataset**
We compare our model with Mip-NeRF [Barron et al., 2022], DietNeRF [Jain et al., 2021], PixelNeRF [Yu et al., 2021], SRF [Chibane et al., 2021], MVSNeRF [Chen et al., 2021], RegNeRF [Niemeyer et al., 2022] and DVGOv2 [Sun et al., 2022a; Sun et al., 2022b] on the LLFF Dataset in the 3-view setting. Similar to the experiments on the Realistic Synthetic 360° Dataset, we fine-tune PixelNeRF, SRF, and MVSNeRF on each scene of the LLFF dataset to handle the domain shift issue since these methods are pre-trained on the DTU dataset.

The overall quantitative results are presented in Tab. 2. Besides, we provide the qualitative results in the supplementary material. Our model is superior to previous works in each metric except for LPIPS, which measures human perception. However, pre-trained model extracting high-level information is not used in our approach, which is a trade-off between complexity and performance, leading to relatively higher LPIPS on our model’s evaluation results.

## 4.4 Ablation Study
**Effectiveness of Proposed Components**
We conduct ablation studies on the *room* scene to evaluate the contributions of each component of our proposed model.

### 4.4.1 Ablation Study

<table>
<thead>
<tr>
<th>Variant</th>
<th>Inc.</th>
<th>( \mathcal{L}_{DS} )</th>
<th>( \mathcal{L}_{CVS} )</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
<th>LPIPS↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>14.47</td>
<td>0.5684</td>
<td>0.5830</td>
</tr>
<tr>
<td>w/ Inc.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>15.86</td>
<td>0.6303</td>
<td>0.5281</td>
</tr>
<tr>
<td>w/ ( \mathcal{L}_{CVS} )</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>16.91</td>
<td>0.6756</td>
<td>0.5104</td>
</tr>
<tr>
<td>w/ ( \mathcal{L}_{DS} )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>20.08</td>
<td>0.7939</td>
<td>0.4258</td>
</tr>
<tr>
<td>w/o Inc.</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>18.57</td>
<td>0.7688</td>
<td>0.4404</td>
</tr>
<tr>
<td>w/o ( \mathcal{L}_{CVS} )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>17.84</td>
<td>0.7357</td>
<td>0.4730</td>
</tr>
<tr>
<td>w/o ( \mathcal{L}_{DS} )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>19.79</td>
<td>0.7663</td>
<td>0.4183</td>
</tr>
<tr>
<td>Full Model</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>21.82</td>
<td>0.8220</td>
<td>0.3860</td>
</tr>
</tbody>
</table>

Table 3: Ablation study on the *room* scene in the 3-view setting.

## 5 Conclusion
NeRF suffers from a long training time and the requirement of dense inputs. To overcome the above shortages, we propose VGOS, an approach to improve the performance of the voxel-based radiance field from sparse inputs. By directly optimizing voxel grids, the incremental voxel training strategy, and the voxel smoothing method, VGOS is \( 10 \times - 100 \times \) faster than previous few-shot view synthesis methods with state-of-the-art render quality while avoiding the degenerate solutions for explicit radiance field methods in sparse input scenarios.
Acknowledgements
This work was supported in part by the projects No. 2020YFC1523202-2, 2021YFF0900604, 19ZDA197, LY21F020005, 2021009, 2019011, 2022C01222, Research and Application Demonstration of Key Technologies in Digital Archaeology of Ningbo Prehistoric Sites, Key Technologies and Product Research and Development for Cultural Relics Protection and Trading Circulation, MOE Frontier Science Center for Brain Science & Brain-Machine Integration (Zhejiang University), National Natural Science Foundation of China (62172365).

References
[Niemeyer et al., 2020] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable vol-


