DeepVoxels: Learning Persistent 3D Feature Embeddings

Vincent Sitzmann[1], Justus Thies[2], Felix Heide[3],
Matthias Nießner[2], Gordon Wetzstein[1], Michael Zollhöfer[1]

[1]Stanford University, [2]Technical University of Munich, [3]Princeton University

vsitzmann.github.io/deepvoxels/

Abstract

In this work, we address the lack of 3D understanding of generative neural networks by introducing a persistent 3D feature embedding for view synthesis. To this end, we propose DeepVoxels, a learned representation that encodes the view-dependent appearance of a 3D scene without having to explicitly model its geometry. At its core, our approach is based on a Cartesian 3D grid of persistent embedded features that learn to make use of the underlying 3D scene structure. Our approach combines insights from 3D geometric computer vision with recent advances in learning image-to-image mappings based on adversarial loss functions. DeepVoxels is supervised, without requiring a 3D reconstruction of the scene, using a 2D re-rendering loss and enforces perspective and multi-view geometry in a principled manner. We apply our persistent 3D scene representation to the problem of novel view synthesis demonstrating high-quality results for a variety of challenging scenes.

1. Introduction

Recent years have seen significant progress in applying generative machine learning methods to the creation of synthetic imagery. Many deep neural networks, for example based on (variational) autoencoders, are able to inpaint, refine, or even generate complete images from scratch [19, 30]. A very prominent direction is generative adversarial networks [13] which achieve impressive results for image generation, even at high resolutions [26] or conditional generative tasks [20]. These developments allow us to perform highly-realistic image synthesis in a variety of settings; e.g., purely generative, conditional, etc.

However, while each generated image is of high quality, a major challenge is to generate a series of coherent views of the same scene. Such consistent view generation would require the network to have a latent space representation that fundamentally understands the 3D layout of the scene; e.g., how would the same chair look from a different viewpoint? Unfortunately, this is challenging to learn

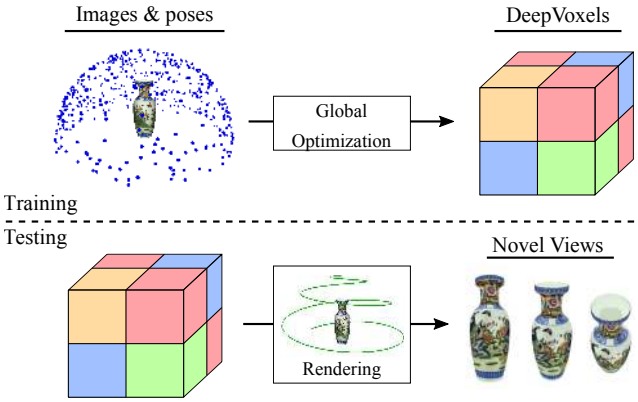

Figure 1: During training, we learn a persistent DeepVoxels representation that encodes the view-dependent appearance of a 3D scene from a dataset of posed multi-view images (top). At test time, DeepVoxels enable novel view synthesis (bottom).

for existing generative neural network architectures that are based on a series of 2D convolution kernels. Here, spatial layout and transformations of a real, 3D environment would require a tedious learning process which maps 3D operations into 2D convolution kernels [22]. In addition, the generator network in these approaches is commonly based on a U-Net architecture with skip connections [47]. Although skip connections enable efficient propagation of low-level features, the learned 2D-to-2D mappings typically struggle to generalize to large 3D transformations, due to the fact that the skip connections bypass higher-level reasoning.

To tackle similar challenges in the context of learning-based 3D reconstruction and semantic scene understanding, the field of 3D deep learning has seen large and rapid progress over the last few years. Existing approaches are able to predict surface geometry with high accuracy. Many of these techniques are based on explicit 3D representations in the form of occupancy grids [35, 43], signed distance fields [46], point clouds [42, 32], or meshes [21]. While these approaches handle the geometric reconstruction task well, they are not directly applicable to the synthesis of realistic imagery, since it is unclear how to rep-

resent color information at a sufficiently high resolution. There also exists a large body of work on learning low-dimensional embeddings of images that can be decoded to novel views [54, 61, 7, 9, 60, 45]. Some of these techniques make use of the object's 3D rotation by explicitly rotating the latent space feature vector [60, 45]. While such 3D techniques are promising, they have thus far not been successful in achieving sufficiently high fidelity for the task of photo-realistic image synthesis.

In our work, we aim at overcoming the fundamental limitations of existing 2D generative models by introducing native 3D operations in the neural network architecture. Rather than learning intuitive concepts from 3D vision, such as perspective, we explicitly encode these operations in the network architecture and perform reasoning directly in 3D space. The goal of the DeepVoxels approach is to condense posed input images of a scene into a persistent latent representation without explicitly having to model its geometry (see Fig. 1). This representation can then be applied to the task of novel view synthesis to generate unseen perspectives of a 3D scene without requiring access to the initial set of input images. Our approach is a hybrid 2D/3D one in that it learns to represent a scene in a Cartesian 3D grid of persistent feature embeddings that is projected to the target view's canonical view volume and processed by a 2D rendering network. This persistent feature volume, which exists in 3D world-space, in combination with a structured, differentiable image formation model, enforces perspective and multi-view geometry in a principled and interpretable manner during training. The proposed approach learns to exploit the underlying 3D scene structure, without requiring supervision in the 3D domain. We demonstrate novel view synthesis with high quality for a variety of scenes based on this new representation. In summary, our approach makes the following technical contributions:

- A novel persistent 3D feature representation for image synthesis that makes use of the underlying 3D scene information.
- Explicit occlusion reasoning based on learned soft visibility that leads to higher-quality results and better generalization to novel viewpoints.
- Differentiable image formation to enforce perspective and multi-view geometry in a principled and interpretable manner during training.
- Training without requiring 3D supervision.

Scope In this paper, we present first steps towards 3D-structured neural scene representations. To this end, we limit the scope of our investigation to allow an in-depth discussion of the challenges fundamental to this approach. We assume Lambertian scenes, without specular highlights or other view-dependent effects. While the proposed approach can deal with light specularities, these are not modeled explicitly. Classical approaches will achieve impressive results on the presented scenes. However, these approaches rely on the explicit reconstruction of geometry. Neural scene representations will be essential to develop generative models that can generalize across scenes to solve reconstruction problems where only few observations are available. We thus compare to such baselines exclusively.

2. Related Work

Our approach lies at the intersection of multiple active research areas, namely generative neural networks, 3D deep learning, deep learning-based view synthesis, and model- as well as image-based rendering.

Neural Image Synthesis Deep models for 2D image and video synthesis have recently shown very promising results. Some of these approaches are based on (variational) auto-encoders (VAEs) [19, 30] or autoregressive models (AMs), such as PixelCNN [38]. The most promising results so far are based on conditional generative adversarial networks (cGANs) [13, 44, 36, 20]. In most cases, the generator network has an encoder-decoder architecture [19], often with skip connections (U-Net) [47], which enable efficient propagation of low-level features from the encoder to the decoder. Approaches that convert synthetic images into photo-realistic imagery have been proposed for the special case of human bodies [64, 2] and faces [28]. In theory, similar architectures could be used to regress the real-world image corresponding to a given viewpoint, i.e., image-based rendering could be learned from scratch. Unfortunately, these 2D-to-2D translation approaches struggle to generalize to transformations in 3D space, such as rotation and perspective projection, since the underlying 3D scene structure cannot be exploited. We compare to this baseline in Sec. 4 and show that DeepVoxels drastically outperforms it.

3D Deep Learning Recently, deep learning has been successfully applied to many 3D geometric reasoning tasks. Current approaches are able to predict an accurate 3D representation of an object from just a single or multiple views. Many of these techniques make use of classical 3D representations, e.g., occupancy grids [35, 43], signed distance fields [46], 3D point clouds [42, 32], or meshes [21]. While these approaches handle the geometric reconstruction task well, they are not directly applicable to view synthesis, since it is unclear how to represent color information at a sufficiently high resolution. View consistency can be explicitly handled using differentiable ray casting [57]. RenderNet [37] learns to render in different styles from 3D voxel grid input. Kulkarni et al. [31] learn a disentangled

representation of images with respect to various scene properties, such as rotation and illumination. Spatial Transformer Networks [22] can learn spatial transformations of feature maps in the network. Even weakly-supervised [62] and unsupervised [23] learning of 3D transformations has been proposed. Our work is also related to CNNs for 3D reconstruction [25, 5] and monocular depth estimation [8]. A "multi-view stereo machine" [25] can learn 3D reconstruction based on 3D or 2.5D supervision. MapNet [18] performs SLAM based on a scene-specific 2D feature grid representation. In contrast to these approaches, which are focused on geometric reasoning, our goal is to learn an embedding for novel view synthesis. To synthesize multi-view consistent images, we optimize for a persistent, scene-specific 3D embedding over all available 2D observations and enable the network to perform explicit occlusion reasoning. We do not require any 3D ground truth but minimize a 2D photometric reprojection loss exclusively.

Deep Learning for View Synthesis Recently, a class of deep neural networks has been proposed that directly aim to solve the problem of novel view synthesis. Some techniques predict lookup tables into a set of reference views [39, 63] or predict weights to blend multi-view images into novel views [11]. A layered scene representation [56] can be learned based on a re-rendering loss. A large corpus of work focuses on embedding 2D views of scenes into a learned low-dimensional latent space that is then decoded into a novel view [54, 61, 7, 9, 60, 45, 6]. Some of these approaches rely on embedding views into a latent space that does not enforce any geometrical constraints [54, 7, 9], others enforce geometric constraints in varying degrees [60, 45, 6, 10], such as learning rotation-equivariant features by explicitly rotating the latent space feature vectors. We focus on optimizing a scene-specific embedding over a training corpus of 2D observations and explicitly account for concepts from 3D vision such as perspective projection and occlusion to constrain the latent space. We demonstrate advantages over weakly structured embeddings in generating high-quality novel views.

Model-Based Rendering Classic reconstruction approaches such as structure-from-motion exploit multi-view geometry [15, 53] to build a dense 3D point cloud of the imaged scene [49, 50, 52, 1, 12]. A triangular surface representation can be obtained using for example the Poisson Surface [27] reconstruction technique. However, the reconstructed geometry is often imperfect, coarse, contains holes, and the resulting renderings thus suffer from visible artifacts and are not fully realistic. In contrast, our goal is to learn a representation that efficiently encodes the view-dependent appearance of a 3D scene without having to explicitly reconstruct a geometric model.

Image-Based Rendering Traditional image-based rendering techniques blend warped versions of the input images to generate new views [51]. This idea was first proposed as a computationally efficient alternative to classical rendering [33, 14, 3]. Multiple-view geometry can be used to obtain the geometry for warping [17]. In other cases, no 3D reconstruction is necessary [11, 41]. Some approaches rely on light fields [24]. Recently, deep-learning has been used to aid image-based rendering via learning a small subtask, i.e., the computation of the blending weights [16, 11]. While this can achieve photorealism, it depends on a dense set of high-resolution photographs to be available at rendering time and requires an error prone reconstruction step to obtain the geometric proxy. Our approach has orthogonal goals: (1) we want to learn an embedding for view synthesis and (2) we want to tackle the problem in a holistic fashion by learning raw pixel output. Thus, our approach is more related to embedding techniques that try to learn a latent space that can be decoded into novel views.

3. Method

The core of our approach is a novel 3D-structured scene representation called DeepVoxels. DeepVoxels is a viewpoint-invariant, persistent and uniform 3D voxel grid of features. The underlying 3D grid enforces spatial structure on the learned per-voxel code vectors. The final output image is formed based on a 2D network that receives the perspective re-sampled version of this 3D volume, i.e., the canonical view volume of the target view, as input. The 3D part of our approach takes care of spatial reasoning, while the 2D part enables fine-scale feature synthesis. In the following, we first introduce the training corpus and then present our end-to-end approach for finding the scene-specific DeepVoxels representation from a set of multi-view images without explicit 3D supervision.

3.1. Training Corpus

Our scene-specific training corpus $\mathcal{C} = \{\mathcal{S}_i, \mathcal{T}_i^0, \mathcal{T}_i^1\}_{i=1}^M$ of M samples is based on a source view $\mathcal{S}_i$ (image and camera pose) and two target views $\mathcal{T}_i^0, \mathcal{T}_i^1$, which are randomly selected from a set of N registered multi-view images; see Fig. 1 for an example. We assume that the intrinsic and extrinsic camera parameters are available. These can for example be obtained using sparse bundle adjustment [55]. For each pair of target views $\mathcal{T}_i^0, \mathcal{T}_i^1$ we then randomly select a single source view $\mathcal{S}_i$ from the top-5 nearest neighbors in terms of view direction angle to target view $\mathcal{T}_i^0$. This sampling heuristic makes it highly likely that points in the source view are visible in the target view $\mathcal{T}_i^0$. While not essential to training, this ensures meaningful gradient flow for every optimization step, while encouraging multi-view consistency to the random target view $\mathcal{T}_i^1$. We sample the training corpus $\mathcal{C}$ dynamically during training.

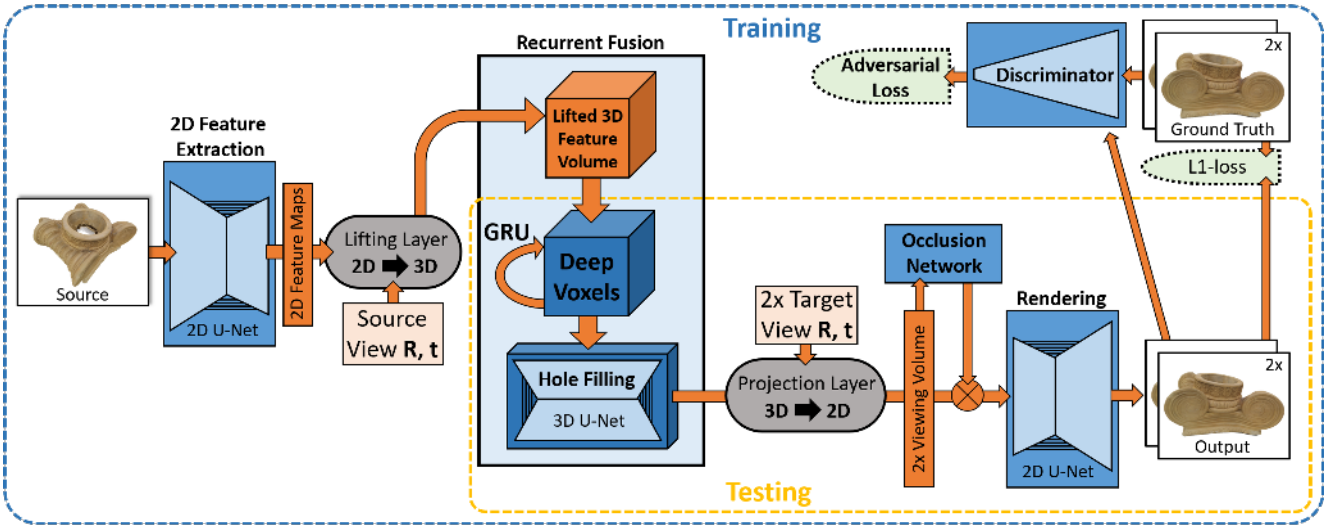

Figure 2: Overview of all model components. At the heart of our encoder-decoder based architecture is a novel viewpoint-invariant and persistent 3D volumetric scene representation called DeepVoxels that enforces spatial structure on the learned per-voxel code vectors.

3.2. Architecture Overview

Our network architecture is summarized in Fig. 2. On a high level, it can be seen as an encoder-decoder based architecture with the persistent 3D DeepVoxels representation as its latent space. During training, we feed a source view $\mathcal{S}_i$ to the encoder and try to predict the target view $\mathcal{T}_i$. We first extract a set of 2D feature maps from the source view using a 2D feature extraction network. To learn a view-independent 3D feature representation, we explicitly lift image features to 3D based on a differentiable lifting layer. The lifted 3D feature volume is fused with our persistent DeepVoxels scene representation using a gated recurrent network architecture. Specifically, the persistent 3D feature volume is the hidden state of a gated recurrent unit (GRU) [4]. After feature fusion, the volume is processed by a 3D fully convolutional network. The volume is then mapped to the camera coordinate systems of the two target views via a differentiable reprojection layer, resulting in the canonical view volume. A dedicated, structured occlusion network operates on the canonical view volume to reason about voxel visibility and flattens the view volume to a 2D view feature map (see Fig. 3). Finally, a learned 2D rendering network forms the two final output images. Our network is trained end-to-end, without the need of supervision in the 3D domain, by a 2D re-rendering loss that enforces that the predictions match the target views. In the following, we provide more details.

Camera Model We follow a perspective pinhole camera model that is fully specified by its extrinsic $\mathbf{E} = [\mathbf{R}|\mathbf{t}] \in \mathbb{R}^{3\times4}$ and intrinsic $\mathbf{K} \in \mathbb{R}^{3\times3}$ camera matrices [15]. Here, $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the global camera rotation and $\mathbf{t} \in \mathbb{R}^3$ its translation. Assume we are given a position $\mathbf{x} \in \mathbb{R}^3$ in 3D coordinates, then the mapping from world space to the canonical camera volume is given as:

$$\mathbf{u} = \begin{pmatrix} u \\ v \\ d \end{pmatrix} = \mathbf{K}(\mathbf{R}\mathbf{x} + \mathbf{t}) \ . \tag{1}$$

Here, u and v specify the position of the voxel center on the screen and d is its depth from the camera. Given a pixel and its depth, we can invert this mapping to compute the corresponding 3D point $\mathbf{x} = \mathbf{R}^T(\mathbf{K}^{-1}\mathbf{u} - \mathbf{t})$.

Feature Extraction We extract 2D feature maps from the source view based on a fully convolutional feature extraction network. The image is first downsampled by a series of stride-2 convolutions until a resolution of 64×64 is reached. A 2D U-Net architecture [48] then extracts a 64×64 feature map that is the input to the subsequent volume lifting.

Lifting 2D Features to 3D Observations The lifting layer lifts 2D features into a temporary 3D volume, representing a single 3D observation, which is then integrated into the persistent DeepVoxels representation. We position the 3D feature volume in world space such that its center roughly aligns with the scene's center of gravity, which can be obtained cheaply from the keypoint point cloud obtained from sparse bundle adjustment. The spatial extent is set such that the complete scene is inside the volume. We try to bound the scene as tightly as possible to not lose spatial resolution. Lifting is implemented by a gathering operation. For each voxel, the world space position of its center is projected to the source view's image space following Eq. 1. We extract a feature vector from the feature map using bilinear sampling and store the result in the code vector associated

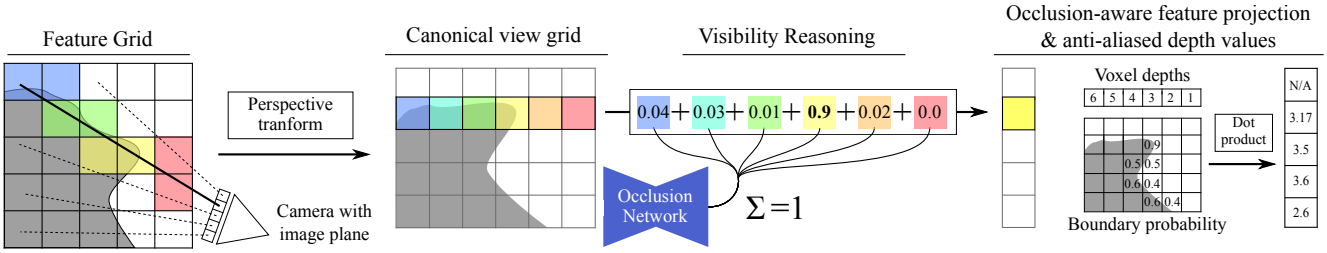

Figure 3: Illustration of the occlusion-aware projection operation. The feature volume (represented by feature grid) is first resampled into the canonical view volume via a projection transformation and trilinear interpolation. The occlusion network then predicts per-pixel softmax weights along each depth ray. The canonical view volume is then collapsed along the depth dimension via a softmax-weighted sum of voxels to yield the final, occlusion-aware feature map. The per-voxel visibility weights can be used to compute a depth map.

with the voxel. Note, our approach is based only on a set of registered multi-view images and we do not have access to the scene geometry or depth maps, rather our approach learns automatically to resolve the depth ambiguity based on a gated recurrent network in 3D.

Integrating Lifted Features into DeepVoxels Lifted observations are integrated into the DeepVoxels representation via an integration network that is based on gated recurrent units (GRUs) [4]. In contrast to the standard application of GRUs, the integration network operates on the same volume across the full training procedure, i.e., the hidden state is *persistent* across all training steps and never reset, leading to a geometrically consistent representation of the whole training corpus. We use a uniform volumetric grid of size $w \times h \times d$ voxels, where each voxel has f feature channels, i.e., the stored code vector has size f. We employ one gated recurrent unit for each voxel, such that at each time step, all the features in a voxel have to be updated jointly. The goal of the gated recurrent units is to incrementally fuse the lifted features and the hidden state during training, such that the best persistent 3D volumetric feature representation is discovered. The gated recurrent units implement the mapping

$$\mathbf{Z}_t = \sigma(\mathbf{W}_z \mathbf{X}_t + \mathbf{U}_z \mathbf{H}_{t-1} + \mathbf{B}_z) \ , \tag{2}$$

$$\mathbf{R}_t = \sigma(\mathbf{W}_r \mathbf{X}_t + \mathbf{U}_r \mathbf{H}_{t-1} + \mathbf{B}_r) \ , \tag{3}$$

$$\mathbf{S}_t = \mathrm{ReLU}(\mathbf{W}_s \mathbf{X}_t + \mathbf{U}_s (\mathbf{R}_t \circ \mathbf{H}_{t-1}) + \mathbf{B}_s) \ , \tag{4}$$

$$\mathbf{H}_t = (1 - \mathbf{Z}_t) \circ \mathbf{H}_{t-1} + \mathbf{Z}_t \circ \mathbf{S}_t \ . \tag{5}$$

Here, $\mathbf{X}_t$ is the lifted 3D feature volume of the current timestep t, the $\mathbf{W}_\bullet$ and $\mathbf{U}_\bullet$ are trainable 3D convolution weights, and the $\mathbf{B}_\bullet$ are trainable tensors of biases. We follow Cho et al. [4] and employ a sigmoid activation σ to compute the response of the tensor of update gates $\mathbf{Z}_t$ and reset gates $\mathbf{R}_t$. Based on the previous hidden state $\mathbf{H}_{t-1}$, the per-voxel reset values $\mathbf{R}_t$, and the lifted 3D feature volume $\mathbf{X}_t$, the tensor of new feature proposals $\mathbf{S}_t$ for the current time step t is computed. $\mathbf{U}_s$ and $\mathbf{W}_s$ are single 3D convolutional layers. The new hidden state $\mathbf{H}_t$, the DeepVoxels representation for the current time step, is computed as a per-voxel linear combination of the old state $\mathbf{H}_{t-1}$ and the new DeepVoxel proposal $\mathbf{S}_t$. The GRU performs one update step per lifted observation. Afterwards, we apply a 3D inpainting U-Net that learns to fill holes in this feature representation. At test time, only the optimally learned persistent 3D volumetric features, the DeepVoxels, are used to form the image corresponding to a novel target view. The 2D feature extraction, lifting layer and GRU gates are discarded and are not required for inference, see Fig. 2.

Projection Layer The projection layer implements the inverse of the lifting layer, i.e., it maps the 3D code vectors to the canonical coordinate system of the target view, see Fig. 3 (left). Projection is also implemented based on a gathering operation. For each voxel of the canonical view volume, its corresponding position in the persistent world space voxel grid is computed. An interpolated code vector is then extracted via a trilinear interpolation and stored in the feature channels of the canonical view volume.

Occlusion Module Occlusion reasoning is essential for correct image formation and generalization to novel viewpoints. To this end, we propose a dedicated occlusion network that computes soft visibility for each voxel. Each pixel in the target view is represented by one column of voxels in the canonical view volume, see Fig. 3 (left). First, this column is concatenated with a feature column encoding the distance of each voxel to the camera, similar as in [34]. This allows the occlusion network to reason about voxel order. The feature vector of each voxel in this canonical view volume is then compressed to a low-dimensional feature vector of dimension 4 by a single 3D convolutional layer. This compressed volume is input to a 3D U-Net for occlusion reasoning. For each ray, represented by a single-pixel column, this network predicts a scalar per-voxel visibility weight based on a softmax activation, see Fig. 3 (middle). The canonical view volume is then flattened along the depth dimension with a weighted average, using the predicted visibility values. The softmax weights can further be used to

compute a depth map, which provides insight into the occlusion reasoning of the network, see Fig. 3 (right).

Rendering and Loss The rendering network is a mirrored version of the feature extraction network with higher capacity. A 2D U-Net architecture takes as input the flattened canonical view volume from the occlusion network and provides reasoning across the full image, before a number of transposed convolutions directly regress the pixel values of the novel view. We train our persistent DeepVoxels representation based on a combined ℓ_1-loss and adversarial cross entropy loss [13]. We found that an adversarial loss accelerates the generation of high-frequency detail earlier on in training. Our adversarial discriminator is a fully convolutional patch-based discriminator [58]. We solve the resulting minimax optimization problem using ADAM [29].

4. Analysis

In this section, we demonstrate that DeepVoxels is a rich and semantically meaningful 3D scene representation that allows high-quality re-rendering from novel views. First, we present qualitative and quantitative results on synthetic renderings of high-quality 3D scans of real-world objects, and compare the performance to strong machine-learning baselines with increasing reliance on geometrically structured latent spaces. Next, we demonstrate that DeepVoxels can also be used to generate novel views on a variety of real captures, even if these scenes may violate the Lambertian assumption. Finally, we demonstrate quantitative and qualitative benefits of explicitly reasoning about voxel visibility via the occlusion module, as well as improved model interpretability. Please see the supplement for further studies on the sensitivity to the number of training images, the size of the voxel volume, as well as noisy camera poses.

Dataset and Metrics We evaluate model performance on synthetic data obtained from rendering 4 high-quality 3D scans (see Fig. 4). We center each scan at the origin and scale it to lie within the unit cube. For the training set, we render the object from 479 poses uniformly distributed on the northern hemisphere. For the test set, we render 1000 views on an Archimedean spiral on the northern hemisphere. All images are rendered in a resolution of 1024×1024 and then resized using area averaging to 512×512 to minimize aliasing. We evaluate reconstruction error in terms of PSNR and SSIM [59].

Implementation All models are implemented in PyTorch [40]. Unless specified otherwise, we use a cube volume with 32^3 voxels. We average the ℓ_1 loss over all pixels in the image. The ℓ_1 and adversarial loss are weighted $200:1$. Models are trained until convergence using ADAM with a learning rate of $4 \cdot 10^{-4}$. One model is trained per scene. The proposed architecture has 170 million parameters. At test time, rendering a single frame takes 71ms.

Baselines We compare to three strong baselines with increasing reliance on geometry-aware latent spaces. The first baseline is a Pix2Pix architecture [20] that receives as input the per-pixel view direction, i.e., the normalized, world-space vector from camera origin to each pixel, and is trained to translate these images into the corresponding color image. This baseline is representative of recent achievements in 2D image-to-image translation. The second baseline is a deep autoencoder that receives as input one of the top-5 nearest neighbors of the target view, and the pose of both the target and the input view are concatenated in the deep latent space, as proposed by Tatarchenko et al. [54]. The inputs of this model at training time are thus identical to those of our model. The third baseline learns an interpretable, rotation-equivariant latent space via the method proposed in [60, 6] and used previously in [45], by being fed one of the top-5 nearest neighbor views and then rotating the latent embedding with the rotation matrix that transforms the input to the output pose. At test time, the previous two baselines receive the top-1 nearest neighbor to supply the model with the most relevant information. We approximately match the number of parameters of each network, with all baselines having equally or slightly more parameters than our model. We train all baselines to convergence with the same loss function. For the exact baseline architectures and number of parameters, please see the supplement.

Object-specific Novel View Synthesis We train our network and all baselines on synthetic renders of four high-quality 3D scans. Table 1 compares PSNR and SSIM of the proposed architecture and the baselines. The best-performing baseline is Pix2Pix [20]. This is surprising, since no geometrical constraints are enforced, as opposed to the approach by Worrall et al. [60]. The proposed architecture with strongly structured latent space outperforms all baselines by a wide margin of an average 7dB. Fig. 4 shows a qualitative comparison as well as further novel views sampled from the proposed model. The proposed model displays robust 3D reasoning that does not break down even in challenging cases. Notably, other models have a tendency to "snap" onto views seen in the training set, while the proposed model smoothly follows the test trajectory. Please see the supplemental video for a demonstration of this behavior. We hypothesize that this improved generalization to unseen views is due to the explicit multi-view constraints enforced by the proposed latent space. The baseline models are not explicitly enforcing projective and epipolar geometry, which may allow them to parameterize latent spaces that are not properly representing the low-dimensional man-

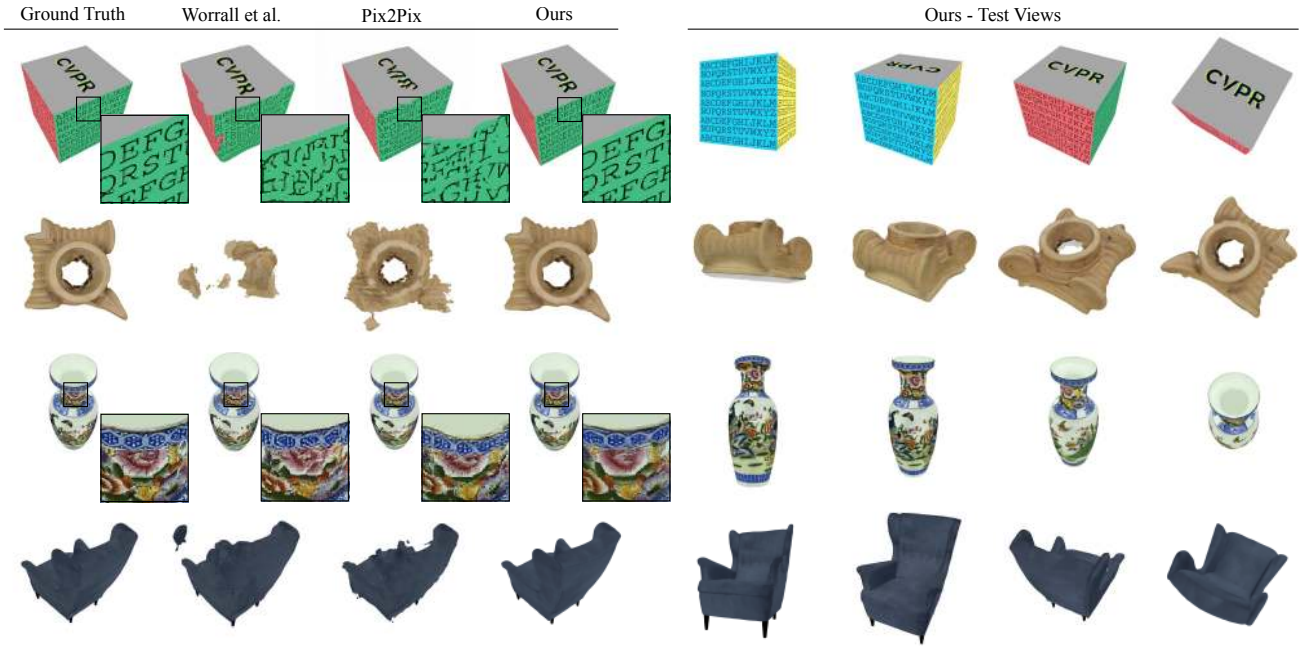

Figure 4: Left: Comparison of the best three performing models to ground truth. From Left to right: Ground truth, Worrall et al. [60], Isola et al. [20] (Pix2Pix), and ours. Our outputs are closest to the ground truth, performing well even in challenging cases such as the strongly foreshortened letters on the cube or the high-frequency detail of the vase. Right: Other samples of novel views generated by our model.

	Vase PSNR / SSIM	Pedestal PSNR / SSIM	Chair PSNR / SSIM	Cube PSNR / SSIM	Mean PSNR / SSIM
Nearest Neighbor	23.26 / 0.92	21.49 / 0.87	20.69 / 0.94	18.32 / 0.83	20.94 / 0.89
Tatarchenko et al. [54]	22.28 / 0.91	23.25 / 0.89	20.22 / 0.95	19.12 / 0.84	21.22 / 0.90
Worrall et al. [60]	23.41 / 0.92	22.70 / 0.89	19.52 / 0.94	19.23 / 0.85	21.22 / 0.90
Pix2Pix (Isola et al.) [20]	26.36 / 0.95	25.41 / 0.91	23.04 / 0.96	19.69 / 0.86	23.63 / 0.92
Ours	**27.99 / 0.96**	**32.35 / 0.97**	**33.45 / 0.99**	**28.42 / 0.97**	**30.55 / 0.97**

Table 1: Quantitative comparison to four baselines. Our approach obtains the best results in terms of PSNR and SSIM on all objects.

ifold of rotations. Although the resolution of the proposed voxel grid is 16 times smaller than the image resolution, our model succeeds in capturing fine detail much smaller than the size of a single voxel, such as the letters on the sides of the cube or the detail on the vase. This may be due to the use of trilinear interpolation in the lifting and projection steps, which allow for a fine-grained representation to be learned. Please see the video for full sequences, and the supplemental material for two additional synthetic scenes.

Voxel Embedding vs. Rotation-Equivariant Embedding As reflected in Tab. 1, we outperform [60] by a wide margin both qualitatively and quantitatively. The proposed model is constrained through multi-view geometry, while [60] has more degrees of freedom. Lacking occlusion reasoning, depth maps are not made explicit. The model may thus parameterize latent spaces that do not respect multi-view geometry. This increases the risk of overfitting, which we observe empirically, as the baseline snaps to nearest neighbors seen during training. While the proposed voxel embedding is memory hungry, it is very parameter efficient. The use of 3D convolutions means that the parameter count is independent of the voxel grid size. Giving up spatial structure means Worrell et al. [60] abandon convolutions and use fully connected layers. However, to achieve the same latent space size of $32^3 \times 64$ features would necessitate more than $4.4 \cdot 10^{12}$ parameters between just the fully connected layers before and after the feature transformation layer, which is infeasible. In contrast, the proposed 3D inpainting network only has $1.7 \cdot 10^7$ parameters, five orders of magnitude less. To address memory inefficiency, the dense grid may be replaced by a sparse alternative in the future.

Occlusion Reasoning and Interpretability An essential part of the rendering pipeline is the depth test. Similarly, the rendering network ought to be able to reason about occlusions when regressing the output view. A naive approach might flatten the depth dimension of the canonical camera volume and subsequently reduce the number of features using a series of 2D convolutions. This leads to a drastic in-

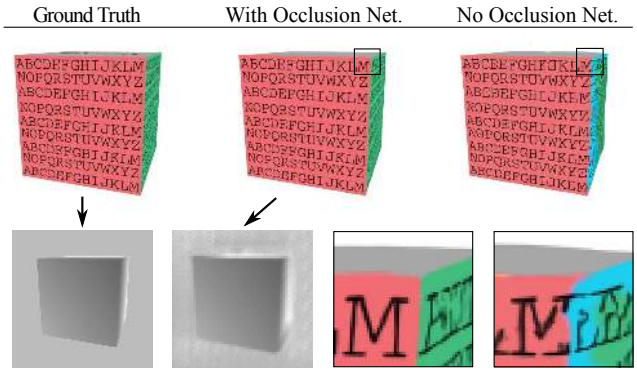

Figure 5: The occlusion module is critical to model performance. It boosts performance from 23.26dB to 28.42dB (cube), and from 30.02dB to 32.35dB (pedestal). Left: ground truth view and depth map. Center: view generated with the occlusion module and learned depth map (64×64 pixels). Note that the object background is unconstrained in the depth map and may differ from ground truth. Right: without the occlusion module, the occluded, blue side of the cube (see Fig. 4) "shines through", and severe artifacts appear (see inset). In addition to decreasing parameter count and boosting performance, the occlusion module generates depth maps fully unsupervised, demonstrating 3D reasoning.

crease in the number of network parameters. At training time, this further allows the network to combine features from several depths equally to regress on pixel colors in the target view. At inference time, this results in severe artifacts and occluded parts of the object "shining through" (see Fig. 5). Our occlusion network forces learning to use a softmax-weighted sum of voxels along each ray, which penalizes combining voxels from several depths. As a result, novel views generated by the network with the occlusion module perform much more favorably at test time, as demonstrated in Fig. 5, than networks without the occlusion module. The depth map generated by the occlusion model further demonstrates that the proposed model indeed learns the 3D structure of the scene. We note that the depth map is learned in a fully unsupervised manner and arises out of the pure necessity of picking the most relevant voxel. Please see the supplement for more examples of learned depth maps.

Novel View Synthesis for Real Captures We train our network on real captures obtained with a DSLR camera. Camera poses, intrinsic camera parameters and keypoint point clouds are obtained via sparse bundle adjustment. The voxel grid origin is set to the respective point cloud's center of gravity. Voxel grid resolution is set to 64. Each voxel stores 8 feature channels. Test trajectories are obtained by linearly interpolating two randomly chosen training poses. Scenes depict a drinking fountain, two busts, a globe, and a bag of coffee. See Fig. 6 for example model outputs. The drinking fountain and the globe have noticeable specularities, which are handled gracefully. While the coffee bag is

Figure 6: Novel views of real captures. Please refer to the video for full sequences with nearest neighbor comparisons.

generally represented faithfully, inconsistencies appear on its highly specular surface. Generally, results are of high quality, and only details that are significantly smaller than a single voxel, such as the tiles in the sink of the fountain, show artifacts. Please refer to the supplemental video for detailed results as well as a nearest-neighbor baseline.

5. Limitations

Although we have demonstrated high-quality view synthesis results for a variety of challenging scenes, the proposed approach still has limitations that can be tackled in the future. By construction, the employed 3D volume is memory inefficient, thus we have to trade local resolution for spatial extent. The proposed model can be trained with a voxel resolution of 64^3 with 8 feature channels, filling a GPU with 12GB of memory. Future work on sparse neural networks may replace the dense representation at the core. Please note, compelling results can already be achieved with quite small volume resolutions. Synthesizing images from viewpoints that are significantly different from the training set, i.e., generalization, is challenging for all learning-based approaches. While this is also true for DeepVoxels and detail is lost when viewing scenes from poses far away from training poses, DeepVoxels generally deteriorates gracefully and the 3D structure of the scene is preserved. Please refer to the supplemental material for failure cases as well as examples of pose extrapolation.

6. Conclusion

We have proposed a novel 3D-structured scene representation, called DeepVoxels, that encodes the view-dependent appearance of a 3D scene using only 2D supervision. Our approach is a first step towards 3D-structured neural scene representations and the goal of overcoming the fundamental limitations of existing 2D generative models by introducing native 3D operations into the network.

Acknowledgements: We thank Robert Konrad, Nitish Padmanaban, and Ludwig Schubert for fruitful discussions, and Robert Konrad for the video voiceover. Vincent Sitzmann was supported by a Stanford Graduate Fellowship. Michael Zollhöfer and Vincent Sitzmann were supported by the Max Planck Center for Visual Computing and Communication (MPC-VCC). Gordon Wetzstein was supported by a National Science Foundation CAREER award (IIS 1553333), by a Sloan Fellowship, and by an Okawa Research Grant. Matthias Nießner and Justus Thies were supported by a Google Research Grant, the ERC Starting Grant Scan2CAD (804724), a TUM-IAS Rudolf Mößbauer Fellowship (Focus Group Visual Computing), and a Google Faculty Award.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *Proc. CVPR*, pages 72–79, 2009.

[2] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros. Everybody Dance Now. *ArXiv e-prints*, 2018.

[3] S. E. Chen and L. Williams. View interpolation for image synthesis. In *Proc. ACM SIGGRAPH*, pages 279–288, 1993.

[4] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[5] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proc. ECCV*, pages 628–644, 2016.

[6] T. S. Cohen and M. Welling. Transformation properties of learned visual representations. *arXiv preprint arXiv:1412.7659*, 2014.

[7] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox. Learning to generate chairs, tables and cars with convolutional networks. *IEEE Trans. PAMI*, 39(4):692–705, 2017.

[8] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Proc. NIPS*, pages 2366–2374, 2014.

[9] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[10] L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. *ICML Workshops*, 2018.

[11] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proc. CVPR*, pages 5515–5524, 2016.

[12] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. PAMI*, 32(8):1362–1376, 2010.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proc. NIPS*, 2014.

[14] N. Greene. Environment mapping and other applications of world projections. *IEEE CG&A*, 6(11):21–29, 1986.

[15] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[16] P. Hedman, J. Philip, T. Price, J.-M. Frahm, G. Drettakis, and G. Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph. (SIGGRAPH Asia)*, 37(6), 2018.

[17] P. Hedman, T. Ritschel, G. Drettakis, and G. Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph. (SIGGRAPH Asia)*, 35(6):231, 2016.

[18] J. F. Henriques and A. Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *Proc. CVPR*, pages 8476–8484, 2018.

[19] G. E. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proc. CVPR*, pages 5967–5976, 2017.

[21] D. Jack, J. K. Pontes, S. Sridharan, C. Fookes, S. Shirazi, F. Maire, and A. Eriksson. Learning free-form deformations for 3d object reconstruction. *CoRR*, abs/1803.10932, 2018.

[22] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu. Spatial transformer networks. In *Proc. NIPS*, pages 2017–2025. 2015.

[23] D. Jimenez Rezende, S. M. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images. In *Proc. NIPS*, pages 4996–5004. 2016.

[24] N. K. Kalantari, T.-C. Wang, and R. Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Trans. Graph. (SIGGRAPH Asia)*, 35(6):193, 2016.

[25] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. In *Proc. NIPS*, pages 365–376, 2017.

[26] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. ICLR*, 2018.

[27] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proc. SGP*, pages 61–70, 2006.

[28] H. Kim, P. Garrido, A. Tewari, W. Xu, J. Thies, N. Nießner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt. Deep Video Portraits. *ACM Trans. Graph. (SIGGRAPH)*, 2018.

[29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[31] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Proc. NIPS*, pages 2539–2547. 2015.

[32] C.-H. Lin, C. Kong, and S. Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *AAAI*, 2018.

[33] A. Lippman. Movie-maps: An application of the optical videodisc to computer graphics. In *ACM SIGGRAPH*, volume 14, pages 32–42, 1980.

[34] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *arXiv preprint arXiv:1807.03247*, 2018.

[35] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Proc. IROS*, page 922 928, September 2015.

[36] M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv:1411.1784, 2014.

[37] T. H. Nguyen-Phuoc, C. Li, S. Balaban, and Y. Yang. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Proc. NIPS 2018*, pages 7902–7912. 2018.

[38] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. In *Proc. NIPS*, pages 4797–4805, 2016.

[39] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg. Transformation-grounded image generation network for novel 3d view synthesis. *CoRR*, abs/1703.02921, 2017.

[40] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[41] E. Penner and L. Zhang. Soft 3d reconstruction for view synthesis. *ACM Trans. Graph. (SIGGRAPH Asia)*, 36(6):235, 2017.

[42] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. CVPR*, 2017.

[43] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proc. CVPR*, 2016.

[44] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proc. ICLR*, 2016.

[45] H. Rhodin, M. Salzmann, and P. Fua. Unsupervised geometry-aware representation for 3d human pose estimation. *Proc. ECCV*, 2018.

[46] G. Riegler, A. O. Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proc. CVPR*, 2017.

[47] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI*, pages 234–241, 2015.

[48] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI*, pages 234–241, 2015.

[49] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016.

[50] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. ECCV*, 2016.

[51] H. Shum and S. B. Kang. Review of image-based rendering techniques. In *Proc. VCIP*, pages 2–14, 2000.

[52] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM Trans. Graph. (SIGGRAPH)*, volume 25, pages 835–846, 2006.

[53] R. Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[54] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Single-view to multi-view: Reconstructing unseen views with a convolutional network. *CoRR abs/1511.06702*, 1(2):2, 2015.

[55] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proc. ICCV Workshops*, pages 298–372, 2000.

[56] S. Tulsiani, R. Tucker, and N. Snavely. Layer-structured 3d scene inference via view synthesis. In *Proc. ECCV*, 2018.

[57] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proc. CVPR*, 2017.

[58] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *Proc. CVPR*, 2018.

[59] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Im. Proc.*, 13(4):600–612, 2004.

[60] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow. Interpretable transformations with encoder-decoder networks. In *Proc. ICCV*, volume 4, 2017.

[61] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Proc. NIPS*, pages 1696–1704, 2016.

[62] J. Yang, S. E. Reed, M.-H. Yang, and H. Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *Proc. NIPS*, pages 1099–1107, 2015.

[63] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. In *Proc. ECCV*, pages 286–301. Springer, 2016.

[64] H. Zhu, H. Su, P. Wang, X. Cao, and R. Yang. View extrapolation of human body from a single image. *CoRR*, abs/1804.04213, 2018.