Geometry-Free View Synthesis: Transformers and no 3D Priors

Robin Rombach*  Patrick Esser*  Björn Ommer
Heidelberg Collaboratory for Image Processing, IWR, Heidelberg University, Germany

*Both authors contributed equally to this work. Code is available at https://git.io/JOnwn.

Figure 1. We present a probabilistic approach to Novel View Synthesis based on transformers, which does not require explicit 3D priors. Given a single source frame and a camera transformation (center), we synthesize plausible novel views that exhibit high fidelity (right). For comparison, SynSin [71] (left) yields uniform surfaces and unrealistic warps for large camera transformations.

Abstract

Is a geometric model required to synthesize novel views from a single image? Being bound to local convolutions, CNNs need explicit 3D biases to model geometric transformations. In contrast, we demonstrate that a transformer-based model can synthesize entirely novel views without any hand-engineered 3D biases. This is achieved by (i) a global attention mechanism for implicitly learning long-range 3D correspondences between source and target views, and (ii) a probabilistic formulation necessary to capture the ambiguity inherent in predicting novel views from a single image, thereby overcoming the limitations of previous approaches that are restricted to relatively small viewpoint changes. We evaluate various ways to integrate 3D priors into a transformer architecture. However, our experiments show that no such geometric priors are required and that the transformer is capable of implicitly learning 3D relationships between images. Furthermore, this approach outperforms the state of the art in terms of visual quality while covering the full distribution of possible realizations.

1. Introduction

Imagine looking through an open doorway. Most of the room on the other side is invisible. Nevertheless, we can estimate how the room likely looks. The few visible features enable an informed guess about the height of the ceiling, the position of walls and lighting etc. Given this limited information, we can then imagine several plausible realizations of the room on the other side. This 3D geometric reasoning and the ability to predict what the world will look like before we move is critical to orient ourselves in a world with three spatial dimensions. Therefore, we address the problem of novel view synthesis [34, 23, 9] based on a single initial image and a desired change in viewpoint. In particular, we aim at specifically modeling large camera transformations, e.g. rotating the camera by 90° and looking at previously unseen scenery. As this is an underdetermined problem, we present a probabilistic generative model that learns the distribution of possible target images and synthesizes them at high fidelity. Solving this task has the potential to transform the passive experience of viewing images into an interactive, 3D exploration of the depicted scene. This necessitates an approach that both understands the geometry of the scene and, when rendering novel views of an input image, considers their semantic relationships to the visible content.

Interpolation vs. Extrapolation Recently, impressive synthesis results have been obtained with geometry-focused approaches in the multi-view setting [53, 54, 43], where not just a single but a large number of images or a video of a scene are available such that the task is closer to a view interpolation than a synthesis of genuinely novel views. In contrast, if only a single image is available, the synthesis of novel views is always an extrapolation task. Solving this task is appealing because it allows a 3D exploration of a scene starting from only a single picture. While existing approaches for single-view synthesis make small camera transformations, such as a rotation by a few
degrees, possible, we aim at expanding the possible camera changes to include large transformations. The latter necessitates a probabilistic framework: Especially when applying large transformation, the problem is underdetermined because there are many possible target images which are consistent with the source image and camera pose. This task cannot be solved with a reconstruction objective alone, as it will either lead to averaging, and hence blurry synthesis results, or, when combined with an adversarial objective, cause a significant mode-dropping when modeling the target distribution. Thus, in order to remedy these issues, we propose to model this task with a powerful, autoregressive transformer model, trained to directly maximize the likelihood of the target data.

**Explicit vs. Implicit Geometry** The success of transformers is often attributed to the fact that they enforce less inductive biases compared to convolutional neural networks (CNNs), which are biased towards local context. Relying mainly on CNNs, this locality-biased approach required previous approaches for novel view synthesis to explicitly model the overall geometric transformation, thereby enforcing yet another inductive bias regarding the three dimensional structure. In contrast, by modeling interactions between far-flung regions of source and target images, transformers have the potential to learn to represent the required geometric transformation implicitly without requiring such hand-engineered operations. This raises the question whether it is at all necessary to explicitly include such biases in a transformer model. To address this question, we perform several experiments with varying degrees of inductive bias and find that our autoregressively trained transformer model is indeed capable of learning this transformation completely without built-in priors and can even learn to predict depth in an unsupervised fashion.

**To summarize our contributions**, we (i) propose to learn a probabilistic model for single view synthesis that properly takes into account the uncertainties inherent in the task and show that this leads to significant benefits over previous state-of-the-art approaches; see Fig. 1. We (ii) analyze the need for explicit 3D inductive biases in transformer architectures and find that transformers make it obsolete to explicitly code 3D transformations into the model and instead can learn the required transformation implicitly themselves. We also (iii) find that the benefits of providing them geometric information in the form of explicit depth maps are relatively small, and investigate the ability to recover an explicit depth representation from the layers of a transformer which has learned to represent the geometric transformation implicitly and without any depth supervision.

## 2. Related Work

**Novel View Synthesis** We can identify three seminal works which illustrate different levels of reliance on geometry to synthesize novel views. [34] describes an approach which requires no geometric model, but requires a large number of structured input views. [23] describes a similar approach but shows that unstructured input views suffice if geometric information in the form of a coarse volumetric estimate is employed. [9] can work with a sparse set of views but requires an accurate photogrammetric model. Subsequent work also analyzed the commonalities and trade-offs of these approaches [5]. Ideally, an approach could synthesize novel views from a single image without having to rely on accurate geometric models of the scene and early works on deep learning for novel view synthesis explored the possibility to directly predict novel views [13, 14, 33, 74, 64] or their appearance flows [78, 47, 62] with convolutional neural networks (CNNs). However, results of these methods were limited to simple or synthetic data and subsequent works combined geometric approaches with CNNs.

Among these deep learning approaches that explicitly model geometry, we can distinguish between approaches relying on a proxy geometry to perform a warping into the target view, and approaches predicting a 3D representation that can subsequently be rendered in novel views. For the proxy geometry, [42] relies on point clouds obtained from structure from motion (SfM) [1, 56] and multi-view stereo (MVS) [57, 18]. To perform the warping, [17, 73] use plane-sweep volumes, [31] estimates depth at novel views and [8, 72] a depth probability volume. [53, 54] post-process MVS results to a global mesh and [26] relies on per-view meshes [27]. Other approaches learn 3D features per scene, which are associated with a point cloud [2] or UV maps [66], and decoded to the target image using a CNN. However, all of these approaches rely on multi-view inputs to obtain an estimate for the proxy geometry.

Approaches which predict 3D representations mainly utilize layered representations such as layered depth images (LDIs) [58, 24, 25], multi-plane images (MPIs) [63, 77, 60, 16] and variants thereof [48, 35]. While this allows an efficient rendering of novel views from the obtained representations, their layered nature limits the range of novel views that can be synthesized with them. Another emerging approach [43] represents a five dimensional light field directly with a multi-layer-perceptron (MLP), but still requires a large number of input views to correctly learn this MLP.

In the case of novel view synthesis from a single view, SfM approaches cannot be used to estimate proxy geometries and early works relied on human interaction to obtain a scene model [29]. [61] uses a large scale, scene-specific light field dataset to learn CNNs which predict light fields from a single image. [38] assumes that scenes can be represented by a fixed set of planar surfaces. To handle more general scenes, most methods rely on monocular depth estimation [51, 19, 20, 76, 21] to predict warps [45, 71, 37] or LDIs [10, 32, 59], and [67] directly predicts an MPI from a
3.1. Probabilistic View Synthesis in Latent Space

Learning the distribution in Eq. (1) requires a model which can capture long-range interactions between source and target view to implicitly represent geometric transformations. Transformer architectures naturally meet these requirements, since they are not confined to short-range relations such as CNNs with their convolutional kernels and exhibit state-of-the-art performance [69]. Since likelihood-based models have been shown [55] to spend too much capacity on short-range interactions of pixels when modeling images directly in pixel space, we follow [15] and employ a two-stage training. The first stage performs adversarially...
guided discrete representation learning (VQGAN), obtaining an abstract latent space that has proved to be well-suited for efficiently training generative transformers [15].

**Modeling Conditional Image Likelihoods** VQGAN consists of an encoder $E$, a decoder $G$ and a codebook $Z = \{z_i\}_{i=1}^{Z}$ of discrete representations $z_i \in \mathbb{R}^{d_z}$. The trained VQGAN allows to encode any $x \in \mathbb{R}^{H \times W \times 3}$ into the discrete latent space as $E(x) \in \mathbb{R}^{h \times w \times d_z}$. Unrolled in raster-scan order, this latent representation corresponds to a sequence $s \in \mathbb{R}^{h \cdot w \cdot d_z}$ and can be equivalently expressed as a sequence of integers which index the learned codebook $Z$. Following the usual designation [69] we refer to the sequence elements as “tokens”. An embedding function $g = g(s) \in \mathbb{R}^{h \cdot w \cdot d_z}$ maps each of these tokens into the embedding space of the transformer $T$ and adds learnable positional encodings. Similarly, to encode the input view $x^{src}$ and the camera transformation $T$, both are mapped into the embedding space by a function $f$:

$$f : (x^{src}, T) \mapsto f(x^{src}, T) \in \mathbb{R}^{n \times d_z},$$

where $n$ denotes the length of the conditioning sequence. By using different functions $f$ various inductive biases can be incorporated into the architecture as described in Sec. 3.2. The transformer $T$ then processes the concatenated sequence $[f(x^{src}, T), g(s^{dst})]$ to learn the distribution of plausible novel views conditioned on $x^{src}$ and $T$,

$$p_T(s^{dst} | f(x^{src}, T)) = \prod_i p_T(s_i^{dst} | s_{<i}^{dst}, f(x^{src}, T)).$$

Hence, to train an autoregressive transformer by next-token prediction $p_T(s_i | s_{<i}, f(x^{src}, T))$ we maximize the log-likelihood of the data, leading to the training objective

$$\mathcal{L}_T = \mathbb{E}_{x^{src}, x^{dst}} p(x^{src}, x^{dst}) \left[ - \log p_T(s^{dst} | f(x^{src}, T)) \right].$$

**3.2. Encoding Inductive Biases**

Besides achieving high-quality novel view synthesis, we aim to investigate to what extent transformers depend on a 3D inductive bias. To this end, we compare approaches where a geometric transformation is built explicitly into the conditioning function $f$, and approaches where no such transformation is used. In the latter case, the transformer itself must learn the required relationship between source and target view. If successful, the transformation will be described implicitly by the transformer.

**Geometric Image Warping** We first describe how an explicit geometric transformation results from the 3D relation of source and target images. For this, pixels of the source image are back-projected to three dimensional coordinates, which can then be re-projected into the target view. We assume a pinhole camera model, such that the projection of 3D points to homogeneous pixel coordinates is determined through the intrinsic camera matrix $K$. The transformation between source and target coordinates is given by a rigid motion, consisting of a rotation $R$ and a translation $t$. Together, these parameters specify the desired control over the novel view to be generated, i.e. $T = (K, R, t)$.

To project pixels back to 3D coordinates, we require information about their depth $d$, since this information has been discarded by their projection onto the camera plane. Since we assume access to only a single source view, we require a monocular depth estimate. Following by previous works [59, 37], we use MiDaS [51] in all of our experiments which require monocular depth information.

The transformation can now be described as a mapping of pixels $i \in \{1, \ldots, H\}, j \in \{1, \ldots, W\}$ in the source image $x^{src} \in \mathbb{R}^{H \times W \times 3}$ to pixels $i', j'$ in the target image. In homogeneous coordinates, their relationship is given by

$$\begin{pmatrix} j' \\ i' \\ 1 \end{pmatrix} \simeq K \begin{pmatrix} RK^{-1}d(i, j) \\ 1 \end{pmatrix} + t$$

This relationship defines a forward flow field $F^{src \rightarrow dst} = F^{src \rightarrow dst}(K, R, t, d) \in \mathbb{R}^{H \times W \times 3}$ from source to target as a function of depth and camera parameters. The flow field can then be used to warp the source image $x^{src}$ into the target view with a warping operation $S$:

$$x^{warp} = S(F^{src \rightarrow dst}, x^{src}).$$

Because the target pixels obtained from the flow are not necessarily integer valued, we follow [44] and implement $S$ by bilinearly splatting features across the four closest target pixels. When multiple source pixels map to the same target pixels, we use their relative depth to give points closer to the camera more weight—a soft variant of z-buffering.

In the simplest case, we can now describe the difference between explicit and implicit approaches in the way that they receive information about the source image and the desired target view. Here, explicit approaches receive source information warped using the camera parameters, whereas implicit approaches receive the original source image and the camera parameters themselves, i.e.

$$\begin{cases} \text{explicit:} & S(F^{src \rightarrow dst}(K, R, t, d), x^{src}) \\ \text{implicit:} & (K, R, t, d, x^{src}) \end{cases}$$

Thus, in explicit approaches we enforce an inductive bias on the 3D relationship between source and target by making this relationship explicit, while implicit approaches have to learn it on their own. Next, we introduce a number of different variants for each, which are summarized in Fig. 2.

**Explicit Geometric Transformations** In the following, we describe all considered variants in terms of the transformer’s conditioning function $f$. Furthermore, $e$ denotes a
Table 1. To assess the effect of encoding different degrees of 3D prior knowledge, we evaluate all variants on RealEstate and ACID using negative log-likelihood (NLL), FID [28] and PSIM [75], PSNR and SSIM [70]. We highlight best, second best and third best scores.

| Method            | RealEstate10K         | ACID            |
|-------------------|-----------------------|-----------------|
|                   | FID ↓ | NLL ↓ | PSIM ↓ | SSIM ↑ | PSNR ↑ | FID ↓ | NLL ↓ | PSIM ↓ | SSIM ↑ | PSNR ↑ |
| impl.-nodepth     | 48.59 | 4.956 | 3.17 ± 0.43 | 0.42 ± 0.03 | 12.16 ± 1.54 | 42.88 | 5.365 ± 0.007 | 2.90 ± 0.03 | 0.40 ± 0.15 | 15.17 ± 3.40 |
| hybrid            | 48.84 | 4.913 | 3.09 ± 0.46 | 0.44 ± 0.03 | 12.51 ± 1.49 | 44.47 | 5.341 ± 0.000 | 2.83 ± 0.04 | 0.41 ± 0.15 | 15.54 ± 3.52 |
| impl.-depth       | 49.15 | 4.836 | 3.05 ± 0.46 | 0.44 ± 0.03 | 12.66 ± 1.68 | 42.93 | 5.353 ± 0.011 | 2.86 ± 0.02 | 0.41 ± 0.15 | 15.53 ± 3.34 |
| expl.-img         | 49.63 | 4.924 | 3.18 ± 0.46 | 0.43 ± 0.03 | 12.11 ± 1.66 | 47.72 | 5.414 ± 0.006 | 3.00 ± 0.01 | 0.40 ± 0.14 | 14.83 ± 3.20 |
| impl.-catedepth   | 50.04 | 4.860 | 3.10 ± 0.48 | 0.43 ± 0.03 | 12.43 ± 1.66 | 47.44 | 5.350 ± 0.004 | 2.86 ± 0.03 | 0.42 ± 0.15 | 15.54 ± 3.57 |
| expl.-emb         | 50.35 | 5.004 | 3.15 ± 0.45 | 0.43 ± 0.03 | 12.30 ± 2.66 | 47.08 | 5.416 ± 0.007 | 2.88 ± 0.04 | 0.42 ± 0.15 | 15.45 ± 3.61 |
| expl.-feat        | 54.82 | 5.159 | 3.31 ± 0.43 | 0.41 ± 0.03 | 11.75 ± 2.58 | 52.65 | 5.657 ± 0.003 | 3.14 ± 0.02 | 0.38 ± 0.15 | 14.06 ± 3.28 |

Implicit Geometric Transformations

Next, we describe implicit variants that we use to analyze if transformers—with their ability to attend to all positions equally well—require an explicit geometric transformation built into the model. We use the same notation as for the explicit variants.

(IV) The first variant, impl.-catedepth, provides the transformer with all the same components which are used in the explicit variants: Camera parameters $K, R, t$, estimated depth $d$ and source image $x_{\text{src}}$. Camera parameters are flattened and concatenated to $T$, which is mapped via $W_{\text{cam}} \in \mathbb{R}^{d_x \times 1}$ to the embedding space. Depth and source images are encoded by VQGAN encoders $E^{d}$ and $E$ to obtain:

$$f(x_{\text{src}}, T) = [W_{\text{cam}} \hat{T}, e(E^{d}(d)), e(E(x_{\text{src}}))] + e^{\text{pos}}$$ (12)

Compared to the other variants, this sequence is roughly $\frac{3}{2}$ times longer, resulting in twice the computational costs.

(V) Therefore, we also include a impl.-depth variant, which concatenates the discrete codes of depth and source image, and maps them with a matrix $W \in \mathbb{R}^{d_x \times 2 \times d_z}$ to the embedding space to avoid an increase in sequence length:

$$f(x_{\text{src}}, T) = [W^{\text{cam}} \hat{T}, W[E^{d}(d), E(x_{\text{src})] + e^{\text{pos}}$$ (13)

(VI) Implicit approaches offer an intriguing possibility: Because they do not need an explicit estimate of the depth to perform the warping operation $S$, they hold the potential to solve the task without such a depth estimate. Thus, impl.-nodepth uses only camera parameters and source image—the bare minimum according to our task description.

$$f(x_{\text{src}}, T) = [W^{\text{cam}} \hat{T}, e(E(x_{\text{src}}))] + e^{\text{pos}}$$ (14)

(VII) Finally, we analyze if explicit and implicit approaches offer complementary strengths. Thus, we add a hybrid variant whose conditioning function is the sum of the f’s of expl.-emb in Eq. (11) and impl.-depth in Eq. (13).

3.3. Depth Readout

To investigate the ability to learn an implicit model of the geometric relationship between different views, we propose to extract an explicit estimate of depth from a trained model. To do so, we use linear probing [6], which is commonly used to investigate the feature quality of unsupervised approaches. More specifically, we assume a transformer model consisting of $L$ layers and of type impl.-nodepth, which is conditioned on source frame and transformation parameters only. Next, we specify a certain layer $0 \leq l \leq L$ (where $l = 0$ denotes the input) and extract its latent representation $e^l$, corresponding to the positions of the provided source frame $x_{\text{src}}$. We then train a position-wise linear classifier $W$ to predict the discrete, latent representation of the depth-encoder $E^{d}$ (see Sec. 3.2) via a cross-entropy objective from $e^l$. Note that both the weights of the transformer and the VQGANs remain fixed.
4. Experiments

First, Sec. 4.1 integrates the different explicit and implicit inductive biases into the transformer to judge if such geometric biases are needed at all. Following up, Sec. 4.2 compares implicit variants to previous work and evaluates both the visual quality and fidelity of synthesized novel views. Finally, we evaluate the ability of the least biased variant, impl.-nodepth, to implicitly represent scene geometry, observing that they indeed capture such 3D information.

4.1. Comparing Implicit and Explicit Transformers

To investigate if transformers need (or benefit from) an explicit warping between source and target view we first compare how well the different variants from Sec. 3.2 (see also Fig. 2) can learn a probabilistic model for novel view synthesis. Afterwards, we directly evaluate both the quality and fidelity of samples obtained from these models.

To prepare, we first train VQGANs on frames of the RealEstate10K [77] and ACID [37] datasets, whose preparation is described in the supplementary. We then train the various transformer variants on the latent space of the respective first stage models. Note that this procedure ensures comparability of different settings within a given dataset, as the space in which the likelihood is measured remains fixed.

Comparing Density Estimation Quality

A basic measure for the performance of probabilistic models is the likelihood assigned to validation data. Hence, we begin our evaluation of the different variants by comparing their (minimal) negative log-likelihood (NLL) on RealEstate and ACID. Based on the results in Tab. 1, we can identify three groups with significant performance differences on ACID: The implicit variants impl.-catdepth, impl.-depth, and impl.-nodepth and hybrid achieve the best performance, which indicates an advantage over the purely explicit variants. Adding an explicit warping as in the hybrid model does not help significantly.

Moreover, expl.-feat is unfavorable, possibly due to the features $E(x^{src})$ remaining fixed while training the transformer. The learnable features which are warped in variant expl.-emb obtain a lower NLL and thereby confirm the former hypothesis. Still there are no improvements of warped features over warped pixels as in variant expl.-img.

The results on RealEstate look similar but in this case the implicit variant without depth, impl.-nodepth, performs a bit worse than expl.-img. Presumably, accurate depth information obtained from a supervised, monocular depth estimation model are much more beneficial in the indoor setting of RealEstate compared to the outdoor setting of ACID.

Visualizing Entropy of Predictions

The NLL measures the ability of the transformer to predict target views. The entropy of the predicted distribution over the codebook entries for each position captures the prediction uncertainty of the model. See Fig. 4 for a visualization of variant impl.-nodepth. The model is more confident in its predictions for regions which are visible in the source image. This indicates that it is indeed able to relate source and target via their geometry instead of simply predicting an arbitrary novel view.

Measuring Image Quality and Fidelity

Because NLL does not necessarily reflect the visual quality of the images [65], we evaluate the latter also directly. Comparing predictions for novel views with ground-truth helps to judge the faithfulness with which the model transforms the source view into the novel view as specified by the camera. However, since we are especially interested in the case of large camera movements where large parts of the target image have not been observed in the source view, we must also evaluate the quality of the content imagined by the model. Note that a sample from the model might be fairly different from the content in the ground-truth, since the latter is just one of many possible realizations of the real-world.

![Figure 4. Visualization of the entropy of the predicted target code distribution for impl.-nodepth. Increased confidence (darker colors) in regions which are visible in the source image indicate its ability to relate source and target geometrically, without 3D bias.](image-url)
To evaluate the image quality without a direct comparison to the ground-truth, we report FID scores [28]. To evaluate the fidelity to the ground-truth, we report the low-level similarity metrics SSIM [70] and PSNR, and the high-level similarity metric PSIM [75], which has been shown to better represent human assessments of visual similarity. Tab. 1 contains the results for RealEstate10K and ACID. In general, these results reflect the findings from the NLL values: Image quality and fidelity of implicit variants with access to depth are superior to explicit variants. The implicit variants outperform previous approaches in all metrics except for IS, with drastic improvements for FID.

| Source | Target | 3DPhoto [59] | SynSin [71] | expl.-det | impl.-depth | impl.-nodepth |
|--------|--------|--------------|--------------|-----------|-------------|--------------|
| method | FID ↓  | IS ↑ | PSIM ↓ | SSIM ↑ | PSNR ↑ |
| impl.-nodepth | 48.59 | 4.24 ±0.16 | 2.95 ±0.43 | 0.49 ±0.12 | 14.06 ±2.45 |
| impl.-depth | 49.15 | 4.17 ±0.52 | 2.86 ±0.45 | 0.50 ±0.12 | 14.47 ±2.51 |
| expl.-det | 66.66 | 4.47 ±0.49 | 2.97 ±0.45 | 0.42 ±0.15 | 13.60 ±2.36 |
| 3DPhoto [59] | 85.43 | 5.10 ±0.38 | 3.20 ±0.34 | 0.49 ±0.12 | 12.80 ±2.33 |
| SynSin [71] | 113.88 | 3.70 ±0.30 | 3.30 ±0.31 | 0.47 ±0.13 | 12.87 ±2.46 |
| MiDaS [51] | 132.13 | 5.63 ±0.77 | 3.38 ±0.36 | 0.46 ±0.15 | 13.09 ±2.16 |

Table 2. Quantitative comparison on RealEstate. Reconstruction metrics are reported with 32 samples, see Fig. 3 for other values. Our implicit variants outperform previous approaches in all metrics except for IS, with drastic improvements for FID.

Figure 5. Qualitative Results on RealEstate10K: We compare three deterministic convolutional baselines (3DPhoto [59], SynSin [71], expl.-det) to our implicit variants impl.-depth and impl.-nodepth. Ours is able to synthesize plausible novel views, whereas others produce artifacts or blurred, uniform areas. The depicted target is only one of many possible realizations; we visualize samples in the supplement.

To evaluate the image quality without a direct comparison to the ground-truth, we report FID scores [28]. To evaluate the fidelity to the ground-truth, we report the low-level similarity metrics SSIM [70] and PSNR, and the high-level similarity metric PSIM [75], which has been shown to better represent human assessments of visual similarity. Tab. 1 contains the results for RealEstate10K and ACID. In general, these results reflect the findings from the NLL values: Image quality and fidelity of implicit variants with access to depth are superior to explicit variants. The implicit variant without depth (impl.-nodepth) consistently achieves the same good FID scores as the implicit variants with depth (impl.-catdepth & impl.-depth), but cannot achieve quite the same level of performance in terms of reconstruction fidelity. However, it is on par with the explicit variants, albeit requiring no depth supervision.

4.2. Comparison to Previous Approaches

Next, we compare our best performing variants impl.-depth and impl.-nodepth to previous approaches for novel view synthesis: 3DPhoto [59], SynSin [71] and InfNat [37]. 3DPhoto [59] has been trained on MSCOCO [36] to work on arbitrary scenes, whereas SynSin [71] and InfNat [37] have been trained on RealEstate and ACID, respectively.

To assess the effect of formulating the problem probabilistically, we introduce another baseline to compare probabilistic and deterministic models with otherwise equal architectures. Specifically, we use the same VQGAN encoder and decoder architectures as used in the first stage described in Sec. 3.1. However, they are not trained as an autoencoder, but instead the encoder receives the warped source image $x^{wp}$, and the decoder predicts the target image $x^{dt}$. This model, denote by expl.-det, represents an explicit and deterministic baseline. Finally, we include the warped source image itself as a baseline denoted by MiDaS [51].

Utilizing the probabilistic nature of our model, we analyze how close we can get to a particular target image with a fixed amount of samples. Tab. 2 and 3 report the reconstruction metrics with 32 samples per validation example of RealEstate and ACID. The probabilistic variants consistently achieve the best values for the similarity metrics PSIM, SSIM and PSNR on RealEstate, and are always among the best three values on ACID, where expl.-det achieves the best PSIM values and the second best PSNR values after impl.-depth. We also show the reconstruction metrics on RealEstate as a function of the number of samples in Fig. 3. We observe that already with four samples, the performance of impl.-depth is better than all other approaches except for the SSIM values of 3DPhoto [59], which are overtaken by impl.-depth with 16 samples, and does not saturate even when using 32 samples, which demonstrates the advantages of a probabilistic formulation of novel view synthesis.

These results should be considered along with the competitive FID scores in Tab. 2 and 3 (where the implicit variants always constitute the best and second best value) and the qualitative results in Fig. 5 and 6, underlining the high quality of our synthesized views. Furthermore, it is striking that IS assign the best scores to 3DPhoto [59] and MiDaS [51], the two variants which contain large and plain regions of gray color in regions where the source image does not provide information about the content. In cases where the monocular depth estimation is accurate, 3DPhoto [59] shows good results but it can only inpaint small areas. SynSin [71] and InfNat [37] can fill larger areas but, for large camera motions, their results quickly become blurry.
Based on the experiments in Sec. 4.1 and Sec. 4.2, which showed that the unbiased variant impl.-nodepth is mostly on-par with the others, we investigate the question whether this model is able to develop an implicit 3D “understanding” without explicit 3D supervision. To do so, we perform linear probing experiments as described in Sec. 3.3. Fig. 7 plots the negative cross-entropy loss and the negative PSIM reconstruction error of the recovered depth maps against the layer depth of the transformer model. Both metrics are consistent and quickly increase when probing deeper representations of the transformer model. Furthermore, both curves exhibit a peak for $l = 4$ (i.e., after the third self-attention block) and then slowly decrease with increasing layer depth. The depth maps obtained from this linear map resemble the corresponding true depth maps qualitatively well as shown in Fig. 7. This figure demonstrates that a linear estimate of depth only becomes possible through the representation learned by the transformer ($l = 4$) but not by the representation of the VQGAN encoder ($l = 0$). We hypothesize that, in order to map an input view onto a target view, the transformer indeed develops an implicit 3D representation of the scene to solve its training task.

### 5. Conclusion

We have introduced a probabilistic approach based on transformers for novel view synthesis from a single source image with strong changes in viewpoint. Comparing various explicit and implicit 3D inductive biases for the transformer showed that explicitly using a 3D transformation in the architecture does not help their performance. Moreover, even with no depth information as input the model learns to infer depth within its internal representations. Both of our implicit transformer approaches showed significant improvements over the state of the art in visual quality and fidelity.
Geometry-Free View Synthesis
Transformers and no 3D Priors

Supplementary Material

In this supplementary, we provide additional results obtained with our models in Sec. A. Sec. B summarizes models, architectures and hyperparameters that were used in the main paper. After describing details on the training and test data in Sec. C, Sec. D concludes the supplementary material with details on the visualization of the transformer’s uncertainty.

A. Additional Results

Interactive Scene Exploration Interface  Fig. 9 shows a preview of the videos available at https://git.io/JOnwn, which demonstrate an interface for interactive 3D exploration of images. Starting from a single image, a user can use keyboard and mouse to move the camera freely in 3D. To provide orientation, we warp the starting image to the current view using a monocular depth estimate (corresponding to the MiDaS [51] baseline in Sec. 4.2). This enables a positioning of the camera with real-time preview of the novel view. Once a desired camera position has been reached, the spacebar can be pressed to autoregressively sample a novel view with our transformer model.

In the videos available at https://git.io/JOnwn, we use camera trajectories from the test sets of RealEstate10K and ACID, respectively. The samples are produced by our impl.-depth model, and for an additional visual comparison, we also include results obtained with the same methods that we compared to in Sec. 4.2.

Figure 9. Preview of the videos available at https://git.io/JOnwn, which demonstrate an interface for interactive 3D exploration of images. Starting from a single image, it allows users to freely move around in 3D. See also Sec. A.

Additional Qualitative Results  For convenience, we also include additional qualitative results directly in this supplementary. Fig. 11 and 13 show additional qualitative comparisons on RealEstate10K and ACID, as in Fig. 5 and 6 of the main paper. Fig. 12 and 14 demonstrate the diversity and consistency of samples by showing them along with their pixel-wise standard deviation. Fig. 15 contains results from the depth-probing experiment of Sec. 4.3, and Fig. 16 from the entropy visualization of Sec. 4.1.

Transformer Variants Over the Course of Training  Fig. 10 reports the negative log-likelihood (NLL) over the course of training on RealEstate and ACID, respectively. The models overfit to the training split of ACID early which makes training on ACID much quicker and thus allows us to perform multiple training runs of each variant with different initializations. This enables an estimate of the significance of the results by computing the mean and standard deviation over three runs (solid line and shaded area in Fig. 10, respectively).
In contrast to the global attention operation, the MLP is applied position-wise.

The architecture of all transformer models discussed in this work follows the GPT-2 architecture [49]. More specifically, the transformer consists of $L$ transformer blocks, where each block performs the following operation on an input sequence $z \in \mathbb{R}^{|z| \times d_{c}}$ (with $|z|$ the length of $z$):

\begin{align*}
  z_{1} &= \text{LayerNorm}(z) \\
  z_{2} &= \text{MultiHeadSelfAttention}(z_{1}) + z \\
  z_{3} &= \text{LayerNorm}(z_{2}) \\
  z_{4} &= \text{MLP}(z_{3}) + z_{2}
\end{align*}

In contrast to the global attention operation, the MLP is applied position-wise.

Given an input sequence $s$ and an embedding produced by the conditioning function $f(x^{\text{src}}, T) \in \mathbb{R}^{n \times d_{c}}$ (see Sec. 3.2), the transformer maps $s$ to a learnable embedding $e(s) + e^{\text{pos}} =: e^{0} \in \mathbb{R}^{|s| \times d_{c}}$, applies the $L$ transformer blocks on the concatenated sequence $[f(x^{\text{src}}, T), e^{0}]$ and finally projects to $|Z|$ logits $\pi^{L}$ via a linear transformation $W_{\text{head}}$, which correspond to a categorical distribution over sequence elements, i.e.,

\begin{align*}
  e^{0} &= [f(x^{\text{src}}, T), e(s) + e^{\text{pos}}] \\
  e^{l} &= \text{TransformerBlock}(e^{l-1}), \quad l = 1 \ldots L \\
  \pi^{L} &= W_{\text{head}} \cdot \text{LayerNorm}(e^{L}).
\end{align*}

Note that non-conditioning elements, i.e., the last $|s|$ elements, are masked autoregressively [69]. For all experiments, we use an embedding dimensionality $d_{c} = 1024$, $L = 32$ transformer blocks, 16 attention heads, two-layer MLPs with hidden dimensionalities of $4 \cdot d_{c}$ and a codebook of size $|Z| = 16384$. This setting results in a transformer with 437M parameters. We train the model using the AdamW [40] optimizer (with $\beta_{1} = 0.9$, $\beta_{2} = 0.95$) and apply weight decay of 0.01 on non-embedding parameters. We train for 500k steps, where we first linearly increase the learning rate from $2.5 \cdot 10^{-6}$ to $1.5 \cdot 10^{-4}$ during the first 5k steps, and then apply a cosine-decay learning rate schedule [39] towards zero.

VQGAN  The architecture and training procedure of the VQGANs is adopted from [15], where we use a downsampling factor of $f = 2^4$. For the codebook $Z$, we use $|Z| = 16384$ entries and a dimensionality of $d_{z} = 256$. This means that any input $x \in \mathbb{R}^{H \times W \times C}$ will be mapped to a latent representation of size $E(x) \in \mathbb{R}^{H/2^4 \times W/2^4 \times 256}$. For our experiments on RealEstate and ACID, where $H = 208$ and $W = 368$, this corresponds to a latent code of size $13 \times 23$ (which is then unrolled to a sequence of length $|s| = 299$). We use the authors’ official implementation and pretrained models2 and perform finetuning on frames of RealEstate and ACID for 50’000 steps on either dataset, resulting in two dataset-specific VQGANs.

\footnote{2\url{https://github.com/CompVis/taming-transformers}}
Other models  For monocular depth estimates, we use MiDaS v2.1\(^3\). We use the official implementations and pretrained models for the comparison with 3DPhoto [59]\(^4\), SynSin [71]\(^5\) and InfNat [37]\(^6\).

C. Training and Testing Data

Training our conditional generative model requires examples consisting of \((x^{\text{dst}}, x^{\text{src}}, T)\). Such training pairs can be obtained via SfM [77] applied to image sequences, which provides poses \((R_i, t_i)\) for each frame \(x^i\) with respect to an arbitrary world coordinate system. For two frames \(x^{\text{src}}, x^{\text{dst}}\) from the sequence, the relative transformation is then given by \(R = R^{\text{dst}}R^{-1}_{\text{src}}\) and \(t = t^{\text{dst}} - R t^{\text{src}}\). However, the scale of the camera translations obtained by SfM is also arbitrary, and without access to the full sequence, underspecified.

To train the model and to meaningfully compute reconstruction errors for the evaluation, we must resolve this ambiguity. To do this, we also triangulate a sparse set of points for each sequence using COLMAP [56]. We then compute a monocular depth estimate for each image using MiDaS [51] and compute the optimal affine scaling to align this depth estimate with the scale of the camera pose. Finally, we normalize depth and camera translation by the minimum depth estimate.

All qualitative and quantitative results are obtained on a subset of the test splits of RealEstate10K [77] and ACID [37], consisting of 564 source-target pairs, which have been selected to contain medium-forward, large-forward, medium-backward and large-backward camera motions in equal parts. We will make this split publicly available along with our code.

D. Details on Entropy Evaluation

As discussed in Sec. 4.1, the relationship between a source view \(x^{\text{src}}\) and a target view \(x^{\text{dst}}\) can be quantified via the entropy of the probability distribution that the transformer assigns to a target view \(x^{\text{dst}}\), given a source frame \(x^{\text{src}}\), camera transformation \(T\) and conditioning function \(f\). More specifically, we first encode target, camera and source via the encoder \(E\) and the conditioning function \(f\) (see Sec. 3.1), i.e. \(s^{\text{dst}} = E(x^{\text{dst}})\) and \(f(x^{\text{src}}, T)\). Next, for each element in the sequence \(s^{\text{dst}}\), the (trained) transformer assigns a probability conditioned on the source and camera:
\[
p(s^{\text{dst}}_i \mid s^{\text{dst}}_{<i}, f(x^{\text{src}}, T))
\]
where for our experiments the length of the target sequence is always \(|s^{\text{dst}}| = 13 \cdot 23 = 299\), see also Sec. B. The entropy \(\mathbb{H}(s^{\text{dst}}_i, x^{\text{src}})\) for each position \(i\) is then computed as
\[
\mathbb{H}(s^{\text{dst}}_i, x^{\text{src}}) = - \sum_k |Z| p_k(s^{\text{dst}}_i \mid s^{\text{dst}}_{<i}, f(x^{\text{src}}, T)) \log p_k(s^{\text{dst}}_i \mid s^{\text{dst}}_{<i}, f(x^{\text{src}}, T))
\]
Reshaping to the latent dimensionality \(h \times w\) and bicubic upsampling to the input’s size \(H \times W\) then produces the visualizations of transformer entropy as in Fig. 4 and Fig. 16. Note that this approach quantifies the transformers uncertainty/surprise from a single example only and does not need to be evaluated on multiple examples.

---

3 see https://github.com/intel-isl/MiDaS
4 see https://github.com/vt-vl-lab/3d-photo-inpainting
5 see https://github.com/facebookresearch/synsin/
6 see https://github.com/google-research/google-research/tree/master/infinite_nature
Figure 11. Additional qualitative comparisons on RealEstate10K.
| Source | $\sigma$ | samples of variant $impl.-depth$ |
|--------|---------|-------------------------------|

Figure 12. Additional samples on RealEstate10K. The second column depicts the pixel-wise standard deviation $\sigma$ obtained from $n = 32$ samples.
| Source | Target | 3DPhoto [59] | InfNat [37] | expl.-det | impl.-depth | impl.-nodepth |
|--------|--------|-------------|-------------|-----------|-------------|---------------|

Figure 13. Additional qualitative comparisons on ACID.
Figure 14. Additional samples on ACID. The second column depicts the pixel-wise standard deviation $\sigma$ obtained from $n = 32$ samples.
Figure 15. Additional results on linearly probed depth maps for different transformer layers as in Fig. 8. See Sec. 4.3.

Figure 16. Additional visualizations of the entropy of the predicted target code distribution for impl.-nodepth. Increased confidence (darker colors) in regions which are visible in the source image indicate its ability to relate source and target geometrically, without 3D bias. See also Sec. 4.1 and Sec. D.
References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In ICCV, pages 72–79. IEEE Computer Society, 2009. 2
[2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor S. Lempitsky. Neural point-based graphics. In ECCV (22), volume 12367 of Lecture Notes in Computer Science, pages 696–712. Springer, 2020. 2, 5
[3] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. CoRR, abs/1607.06450, 2016. 3
[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In ICLR, 2015. 3
[5] Chris Buehler, Michael Bosse, Leonard McMillan, Steven J. Gortler, and Michael F. Cohen. Unstructured lumigraph rendering. In SIGGRAPH, pages 425–432. ACM, 2001. 2
[6] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In ICML, volume 119 of Proceedings of Machine Learning Research, pages 1691–1703. PMLR, 2020. 2, 5
[7] Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. In ICLR (Poster). OpenReview.net, 2017. 3
[8] Inchang Choi, Orazio Gallo, Alejandro J. Troccoli, Min H. Kim, and Jan Kautz. Extreme view synthesis. In ICCV, pages 7780–7789. IEEE, 2019. 2
[9] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In SIGGRAPH, pages 11–20. ACM, 1996. 1, 2
[10] Helisa Dhamo, Keisuke Tateno, Iro Laina, Nassir Navab, and Federico Tombari. Peeking behind objects: Layered depth prediction from a single image. Pattern Recognit. Lett., 125:333–340, 2019. 2
[11] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. CoRR, abs/2005.00341, 2020. 3
[12] Sander Dieleman. Musings on typicality, 2020. 3
[13] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In CVPR, pages 1538–1546. IEEE Computer Society, 2015. 2
[14] Alexey Dosovitskiy, Jost Tobias Springenberg, Maxim Tatarchenko, and Thomas Brox. Learning to generate chairs, tables and cars with convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell., 39(4):692–705, 2017. 2
[15] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. CoRR, abs/2012.09841, 2020. 3, 4, 10
[16] John Flynn, Michael Broxton, Paul E. Debevec, Matthew DuVall, Graham Fyffe, Ryan S. Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In CVPR, pages 2367–2376. Computer Vision Foundation / IEEE, 2019. 2
[17] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. CoRR, abs/1506.06825, 2015. 2
[18] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. IEEE Trans. Pattern Anal. Mach. Intell., 32(8):1362–1376, 2010. 2
[19] Ravi Garg, B. G. Vijay Kumar, Gustavo Carneiro, and Ian D. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In ECCV (8), volume 9912 of Lecture Notes in Computer Science, pages 740–756. Springer, 2016. 2
[20] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In CVPR, pages 6602–6611. IEEE Computer Society, 2017. 2
[21] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In ICCV, pages 3827–3837. IEEE, 2019. 2
[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In NIPS, pages 2672–2680, 2014. 3
[23] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In SIGGRAPH, pages 43–54. ACM, 1996. 1, 2
[24] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. Casual 3d photography. ACM Trans. Graph., 36(6):234:1–234:15, 2017. 2
[25] Peter Hedman and Johannes Kopf. Instant 3d photography. ACM Trans. Graph., 37(4):101:1–101:12, 2018. 2
[26] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel J. Brostow. Deep blending for freeviewpoint image-based rendering. ACM Trans. Graph., 37(6):257:1–257:15, 2018. 2
[27] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel J. Brostow. Scalable inside-out image-based rendering. ACM Trans. Graph., 35(6):231:1–231:11, 2016. 2
[28] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In NIPS, pages 6626–6637, 2017. 5, 7
Jonathan Shade, Steven J. Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In SIGGRAPH, pages 231–242. ACM, 1998.

Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In CVPR, pages 8025–8035. IEEE, 2020.

Pratul P. Srinivasan, Richard Tucker, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In CVPR, pages 175–184. Computer Vision Foundation / IEEE, 2019.

Pratul P. Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4d RGBD light field from a single image. In ICCV, pages 2262–2270. IEEE Computer Society, 2017.

Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J. Lim. Multi-view to novel view: Synthesizing novel views with self-learned confidence. In ECCV (3), volume 11207 of Lecture Notes in Computer Science, pages 162–178. Springer, 2018.

Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. Int. J. Comput. Vis., 32(1):45–61, 1999.

Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Multi-view 3d models from single images with a convolutional network. In ECCV (7), volume 9911 of Lecture Notes in Computer Science, pages 322–337. Springer, 2016.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In ICLR, 2016.

Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: image synthesis using neural textures. ACM Trans. Graph., 38(4):66:1–66:12, 2019.

Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In CVPR, pages 548–557. IEEE, 2020.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In NIPS, pages 6306–6315, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In NIPS, pages 5998–6008, 2017.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process., 13(4):600–612, 2004.

Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In CVPR, pages 7465–7475. IEEE, 2020.