Polynomial Implicit Neural Representations For Large Diverse Datasets

Rajhans Singh Ankita Shukla Pavan Turaga
Geometric Media Lab, Arizona State University
{rsingh70, ashukl20, pavan.turaga}@asu.edu

Figure 1. Samples generated by our Poly-INR model on the ImageNet dataset at various resolutions. Our model generates images with high fidelity without using convolution, upsample, or self-attention layers, i.e., no interaction between the pixels.

Abstract

Implicit neural representations (INR) have gained significant popularity for signal and image representation for many end-tasks, such as superresolution, 3D modeling, and more. Most INR architectures rely on sinusoidal positional encoding, which accounts for high-frequency information in data. However, the finite encoding size restricts the model's representational power. Higher representational power is needed to go from representing a single given image to representing large and diverse datasets. Our approach addresses this gap by representing an image with a polynomial function and eliminates the need for positional encodings. Therefore, to achieve a progressively higher degree of polynomial representation, we use element-wise multiplications between features and affine-transformed coordinate locations after every ReLU layer. The proposed method is evaluated qualitatively and quantitatively on large datasets like ImageNet. The proposed Poly-INR model performs comparably to state-of-the-art generative models without any convolution, normalization, or self-attention layers, and with far fewer trainable parameters. With much fewer training parameters and higher representative power, our approach paves the way for broader adoption of INR models for generative modeling tasks in complex domains. The code is available at https://github.com/Rajhans0/Poly_INR

1. Introduction

Deep learning-based generative models are a very active area of research with numerous advancements in recent years [8, 13, 24]. Most widely, generative models are based on convolutional architectures. However, recent developments such as implicit neural representations (INR) [29, 43] represent an image as a continuous function of its coordinate locations, where each pixel is synthesized independently. Such a function is approximated by using a deep neural network. INR provides flexibility for easy image transformations and high-resolution up-sampling through the use of a coordinate grid. Thus, INRs have become very effective for 3D scene reconstruction and rendering from very few training images [3, 27–29, 56]. However, they are usually trained to represent a single given scene, signal, or image. Recently, INRs have been implemented as a generative model to generate entire image datasets [1, 46]. They perform comparably to CNN-based generative models on perfectly curated datasets

like human faces [22]; however, they have yet to be scaled to large, diverse datasets like ImageNet [7].

INR generally consists of a positional encoding module and a multi-layer perceptron model (MLP). The positional encoding in INR is based on sinusoidal functions, often referred to as Fourier features. Several methods [29, 43, 49] have shown that using MLP without sinusoidal positional encoding generates blurry outputs, i.e., only preserves low-frequency information. Although, one can remove the positional encoding by replacing the ReLU activation with a periodic or non-periodic activation function in the MLP [6, 37, 43]. However, in INR-based GAN [1], using a periodic activation function in MLP leads to subpar performance compared to positional encoding with ReLU-based MLP.

Sitzmann et al. [43] demonstrate that ReLU-based MLP fails to capture the information contained in higher derivatives. This failure to incorporate higher derivative information is due to ReLU's piece-wise linear nature, and second or higher derivatives of ReLU are typically zero. This can be further interpreted in terms of the Taylor series expansion of a given function. The higher derivative information of a function is included in the coefficients of a higher-order polynomial derived from the Taylor series. Hence, the inability to generate high-frequency information is due to the ineffectiveness of the ReLU-based MLP model in approximating higher-order polynomials.

Sinusoidal positional encoding with MLP has been widely used, but the capacity of such INR can be limiting for two reasons. First, the size of the embedding space is limited; hence only a finite and fixed combination of periodic functions can be used, limiting its application to smaller datasets. Second, such an INR design needs to be mathematically coherent. These INR models can be interpreted as a non-linear combination of periodic functions where periodic functions define the initial part of the network, and the later part is often a ReLU-based non-linear function. Contrary to this, classical transforms (Fourier, sine, or cosine) represent an image by a linear summation of periodic functions. However, using just a linear combination of the positional embedding in a neural network is also limiting, making it difficult to represent large and diverse datasets. Therefore, instead of using periodic functions, this work models an image as a polynomial function of its coordinate location.

The main advantage of polynomial representation is the easy parameterization of polynomial coefficients with MLP to represent large datasets like ImageNet. However, conventionally MLP can only approximate lower-order polynomials. One can use a polynomial positional embedding of the form $x^p y^q$ in the first layer to enable the MLP to approximate higher order. However, such a design is limiting, as a fixed embedding size incorporates only fixed polynomial degrees. In addition, we do not know the importance of each polynomial degree beforehand for a given image.

Hence, we do not use any positional encoding, but we progressively increase the degree of the polynomial with the depth of MLP. We achieve this by element-wise multiplication between the feature and affine transformed coordinate location, obtained after every ReLU layer. The affine parameters are parameterized by the latent code sampled from a known distribution. This way, our network learns the required polynomial order and represents complex datasets with considerably fewer trainable parameters. In particular, the key highlights are summarized as follows:

- We propose a Poly-INR model based on polynomial functions and design a MLP model to approximate higher-order polynomials.
- Poly-INR as a generative model performs comparably to the state-of-the-art CNN-based GAN model (StyleGAN-XL [42]) on the ImageNet dataset with $3-4\times$ fewer trainable parameters (depending on output resolution).
- Poly-INR outperforms the previously proposed INR models on the FFHQ dataset [22], using a significantly smaller model.
- We present various qualitative results demonstrating the benefit of our model for interpolation, inversion, style-mixing, high-resolution sampling, and extrapolation.

2. Related work

Implicit neural representations: INRs have been widely adopted for 3D scene representation and synthesis [28, 29, 45]. Following the success of NeRF [29], there has been a large volume of work on 3D scene representation from 2D images [3, 20, 27, 35, 44, 54, 56]. They have also been used for semantic segmentation [11], video [12, 33, 53], audio [12], and time-series modeling [10]. INRs have also been used as a prior for inverse problems [38, 43]. However, most INR approaches either use a sinusoidal positional encoding [29, 49] or a sinusoidal activation function [43], which limits the model capacity for large dataset representation. In our work, we represent our Poly-INR model as a polynomial function without using any positional encoding.

GANs: have been widely used for image generation and synthesis tasks [13]. In recent work, several improvements have been proposed [2, 14, 22, 30, 36] over the original architecture. For example, the popularly used StyleGAN [22] model uses a mapping network to generate style codes which are then used to modulate the weights of the Conv layers. StyleGAN improves image fidelity, as well as enhances inversion [52] and image editing capabilities [15]. StyleGAN has been scaled to large datasets like ImageNet [42], using a discriminator which uses projected features from a pre-trained classifier [41]. More recently, transformer-based models have also been used as generators [26, 57]; however, the self-attention mechanism is computationally costly for achieving higher resolution. Unlike these methods, our generator is

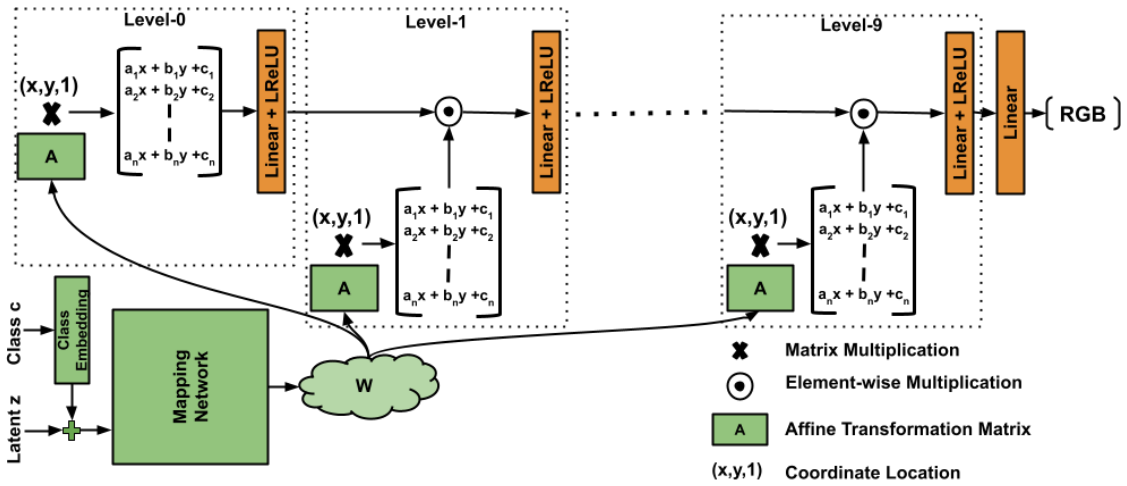

Figure 2. Overview of our proposed Polynomial Implicit Neural Representation (Poly-INR) based generator architecture. Our model consists of two networks: 1) Mapping network, which generates the affine parameters from the latent code z, and 2) Synthesis network, which synthesizes the RGB value for the given pixel location. Our Poly-INR model is defined using only Linear and ReLU layers end-to-end.

free of convolution, normalization, and self-attention mechanisms and only uses ReLU and Linear layers to achieve competitive results, but with far fewer parameters.

GANs + coordinates: INRs have also been implemented within generative models. For example, CIPS [1] uses Fourier features and learnable vectors for each spatial location as positional encoding and uses StyleGAN-like weight modulation for layers in the MLP. Similarly, INR-GAN [46] proposes a multi-scale generator model where a hypernetwork determines the parameters of the MLP. INR-GAN has been further extended to generate an 'infinite'-size continuous image using anchors [47]. However, these INR-based models have only shown promising results on smaller datasets. Our work scales easily to large datasets like ImageNet owing to the significantly fewer parameters.

Other approaches have combined CNN with coordinate-based features. For example, the Local Implicit Image Function (LIIF) [5] and Spherical Local Implicit Image Function (SLIIF) [55] use a CNN-based backbone to generate feature vectors corresponding to each coordinate location. Arbitrary-scale image synthesis [32] uses a multi-scale convolution-based generator model with scale-aware position embedding to generate scale-consistent images. StyleGAN model, further extended by [21] (StyleGAN-3) to use coordinate location-based Fourier features. In addition, StyleGAN-3 uses filter kernels equivariant to the coordinate grid's translation and rotation. However, the rotation equivariant version of the StyleGAN-3 model fails to scale to ImageNet dataset, as reported in [42]. Instead of using convolution layers, the Poly-INR only uses linear and ReLU layers.

Relation to classical geometric moment: Polynomial functions have been explored earlier in the form of geometric moments for image reconstruction [9, 18, 19, 50]. Unlike the Fourier transform, which uses the sinusoidal functions as the basis, the geometric moment method projects the 2D image on a polynomial basis of the form $x^p y^q$ to compute the moment of order $p+q$. The moment matching method [50] is generally used for image reconstruction from given finite moments. In moment matching, the image is assumed to be a polynomial function, and the coefficients of the polynomial are defined to match the given finite moments. Similar to geometric moments, we also represent images on a polynomial basis; however, our polynomial coefficients are learned end-to-end and defined by a deep neural network.

3. Method

We are interested in a class of functions that represent an image in the form:

$$G(x,y) = g_{00} + g_{10}x + g_{01}y + ... + g_{pq}x^p y^q, \quad (1)$$

where, (x,y) is the normalized pixel location sampled from a coordinate grid of size $(H \times W)$, while the coefficients of the polynomial (g_{pq}) are parameterized by a latent vector z sampled from a known distribution and are independent of the pixel location. Therefore, to form an image, we evaluate the generator G for all pixel locations (x,y) for a given fixed z:

$$I = \{G(x,y;z) \,|\, (x,y) \in CoordinateGrid(H,W)\}, \quad (2)$$

where, $CoordinateGrid(H,W) = \{(\frac{x}{W-1}, \frac{y}{H-1}) \,|\, 0 \leq x < W, 0 \leq y < H\}$. By sampling different latent vectors z, we generate different polynomials and represent images over a distribution of real images.

Our goal is to learn the polynomial defined by Eq. 1 using only Linear and ReLU layers. However, the conventional definition of MLP usually takes the coordinate location as input, processed by a few Linear and ReLU layers. This definition of INR can only approximate low-order polynomials and hence only generates low-frequency information. Although, one can use a positional embedding consisting

of polynomials of the form $x^p y^q$ to approximate a higher-order polynomial. However, this definition of INR is limiting since a fixed-size embedding space can contain only a small combination of polynomial orders. Furthermore, we do not know which polynomial order is essential to generate the image beforehand. Hence, we progressively increase the polynomial order in the network and let it learn the required orders. We implement this by using element-wise multiplication with the affine-transformed coordinate location at different levels, shown in Fig 2. Our model consists of two parts: 1) **Mapping network**, which takes the latent code z and maps it to affine parameters space $\mathbf{W}$, and 2) **Synthesis network**, which takes the pixel location and generates the corresponding RGB value.

Mapping Network: The mapping network takes the latent code $z \in \mathbb{R}^{64}$ and maps it to the space $\mathbf{W} \in \mathbb{R}^{512}$. Our model adopts the mapping network used in [42]. It consists of a pre-trained class embedding, which embeds the one hot class label into a 512 dimension vector and concatenates it with the latent code z. Then the mapping network consists of an MLP with two layers, which maps it to the space $\mathbf{W}$. We use this $\mathbf{W}$ to generate affine parameters by using additional linear layers; hence we call $\mathbf{W}$ as affine parameters space.

Synthesis network: The synthesis network generates the RGB ($\mathbb{R}^3$) value for the given pixel location (x, y). As shown in Fig. 2, the synthesis network consists of multiple levels; at each level, it receives the affine transformation parameters from the mapping network and the pixel coordinate location. At *level*-0, we affine transform the coordinate grid and feed it to a Linear layer followed by a Leaky-ReLU layer with $negative\_slope = 0.2$. At later levels, we do element-wise multiplication between the feature from the previous level and the affine-transformed coordinate grid, and then feed it to Linear and Leaky-ReLU layers. With the element-wise multiplication at each level, the network has the flexibility to increase the order for x or y coordinate position, or not to increase the order by keeping the affine transformation coefficient $a_j = b_j = 0$. In our model, we use 10 levels, which is sufficient to generate large datasets like ImageNet. Mathematically, the synthesis network can be expressed as follows:

$$G_{syn} = \dots \sigma(W_2((A_2 X) \odot \sigma(W_1((A_1 X) \odot \sigma(W_0(A_0 X)))))), \quad (3)$$

where $X \in \mathbb{R}^{3\times HW}$ is the coordinate grid of size $H \times W$ with an additional dimension for the bias, $A_i \in \mathbb{R}^{n\times 3}$ is the affine transformation matrix from the mapping network for *level-i*, $W_i \in \mathbb{R}^{n\times n}$ is the weight of the linear layer at *level-i*, σ is the Leaky-ReLU layer and $\odot$ is element-wise multiplication. Here n is the dimension of the feature channel in the synthesis network, which is the same for all levels. For large datasets like ImageNet, we choose the channel dimension $n = 1024$, and for smaller datasets like FFHQ, we choose $n = 512$. Note that with this definition, our model only uses Linear and ReLU layers end-to-end and synthesizes each pixel independently.

Relation to StyleGAN: StyleGANs [21–23] can be seen as a special case of our formulation. By keeping the coefficients (a_j,b_j) in the affine transformation matrix of x and y coordinate location equal to zero, the bias term c_j would act as a style code. However, our affine transformation adds location bias to the style code, rather than just using the same style code for all locations in StyleGAN models. This location bias makes the model very flexible in applying a style code only to a specific image region, making it more expressive. In addition, our model differs from the StyleGANs in many aspects. First, our method does not use weight modulation/demodulation or normalizing [23] tricks. Second, our model does not employ low-pass filters or convolutional layers. Finally, we do not inject any spatial noise into our synthesis network. We can also use these tricks to improve the model's performance further. However, our model's definition is straightforward compared to other GAN models.

4. Experiments

The effectiveness of our model is evaluated on two datasets: 1) ImageNet [7] and 2) FFHQ [22]. The ImageNet dataset consists of $1.2M$ images over $1K$ classes, whereas the FFHQ dataset contains $\sim 70K$ images of curated human faces. All our models have 64 dimensional latent space sampled from a normal distribution with mean 0 and standard deviation 1. The affine parameters space $\mathbf{W}$ of the mapping network is 512 dimensions, and the synthesis network consists of 10 levels with feature dimension $n = 1024$ for the ImageNet and $n = 512$ for FFHQ. We follow the training scheme of the StyleGAN-XL method [42] and use a projected discriminator based on the pre-trained classifiers (DeiT [51] and EfficientNet [48]) with an additional classifier guidance loss [8].

We train our model progressively with increasing resolution, i.e., we start by training at low resolution and continue training with higher resolutions as training progresses. Since the computational cost is less at low resolution, the model is trained for large number of iterations, followed by training for high resolution. Since the model is already trained at low resolution, fewer iterations are needed for convergence at high resolution. However, unlike StyleGAN-XL, which freezes the previously trained layers and introduces new layers for higher resolution, Poly-INR uses a fixed number of layers and trains all the parameters at every resolution.

4.1. Quantitative results

We compare our model against CNN-based GANs (BigGAN [4] and StyleGAN-XL [42]) and diffusion models (CDM [17], ADM, ADM-G [8], and DiT-XL [34]) on the ImageNet dataset. We also report results on the FFHQ dataset

Table 1. Quantitative comparison of Poly-INR method with CNN-based generative models on ImageNet datasets. (d) compares the number of parameters used in all models at various resolutions. The results for existing methods are quoted from the StyleGAN-XL paper.

(a) ImageNet 128×128

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	6.02	7.18	6.09	145.83	0.86	0.35
CDM	3.52	-	-	128.80	-	-
ADM	5.91	5.09	13.29	93.31	0.70	0.65
ADM-G	2.97	5.09	3.80	141.37	0.78	0.59
StyleGAN-XL	1.81	3.82	1.82	200.55	0.77	0.55
Poly-INR	2.08	3.93	2.76	179.64	0.70	0.45

(b) ImageNet 256×256

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	6.95	7.36	75.24	202.65	0.87	0.28
ADM	10.94	6.02	125.78	100.98	0.69	0.63
ADM-G	3.94	6.14	11.86	215.84	0.83	0.53
DiT-XL/2-G	2.27	4.60	-	278.54	0.83	0.57
StyleGAN-XL	2.30	4.02	7.06	265.12	0.78	0.53
Poly-INR	2.86	4.37	7.79	241.43	0.71	0.39

(c) ImageNet 512×512

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	8.43	8.13	312.00	177.90	0.88	0.29
ADM	23.24	10.19	561.32	58.06	0.73	0.60
ADM-G	3.85	5.86	210.83	221.72	0.84	0.53
DiT-XL/2-G	3.04	5.04	-	240.82	0.84	0.54
StyleGAN-XL	2.41	4.06	51.54	267.75	0.77	0.52
Poly-INR	3.81	5.06	54.31	267.44	0.70	0.34

(d) Number of parameters in millions (M)

Model	64^2	128^2	256^2	512^2
BigGAN	-	141.0	164.3	164.7
ADM	296.0	422.0	554.0	559.0
DiT-XL	-	-	675.0	675.0
StyleGAN-XL	134.4	158.7	166.3	168.4
Poly-INR	46.0	46.0	46.0	46.0

Table 2. Quantitative comparison of Poly-INR method with CNN and INR-based generative models on FFHQ dataset at 256×256.

Model	params (M)	FID ↓	Inference Time (sec/img)
StyleGAN2	30.0	3.83	0.016
StyleGAN-XL	67.9	2.19	0.047
CIPS	45.9	4.38	0.067
INR-GAN	72.4	4.95	0.024
Poly-INR	13.6	2.72	0.054

for INR-based GANs (CIPS [1] and INR-GAN [46]) as they do not train models on ImageNet.

Quantitative metrics: We use Inception Score (IS) [40], Frechet Inception Distance (FID) [16], Spatial Frechet Inception Distance (sFID) [31], random-FID (rFID) [42], precision (Pr), and recall (Rec) [25]. IS (higher the better) quantifies the quality and diversity of the generated samples based on the predicted label distribution by the Inception network but does not compare the distribution of the generated samples with the real distribution. The FID score (lower the better) overcomes this drawback by measuring the Frechet distance between the generated and real distribution in the Inception feature space. Further, sFID uses higher spatial features from the Inception network to account for the spatial structure of the generated image. Like StyleGAN-XL, we also use the rFID score to ensure that the network is not just optimizing for IS and FID scores. We use the same randomly initialized Inception network provided by [42]. In addition, we also compare our model on the precision and recall metric (higher the better) that measures how likely the generated sample is from the real distribution.

Table 1 summarizes the results on the ImageNet dataset at different resolutions. The results for existing methods are quoted from the StyleGAN-XL paper. We observe that the performance of the proposed model is third best after DiT-XL and StyleGAN-XL on the FID and IS metrics. The proposed model outperforms the ADM and BigGAN models at all resolutions and performs comparably to the StyleGAN-XL at 128×128 and 256×256. We also observe that with the increase in image size, the FID score for Poly-INR drops much more than StyleGAN-XL. The FID score drops more because our model does not add any additional layers with the increase in image size. For example, the StyleGAN-XL uses 134.4M parameters at 64×64 and 168.4M at 512×512, whereas Poly-INR uses only 46.0M parameters at every resolution, as reported in Table 1(d). The table shows that our model performs comparably to the state-of-the-art CNN-based generative models, even with significantly fewer parameters. On precision metric, the Poly-INR method performs comparably to other methods; however, the recall value is slightly lower compared to StyleGAN-XL and diffusion models at higher resolution. Again, this is due to the small model size, limiting the model's capacity to represent much finer details at a higher resolution.

We also compare the proposed method with other INR-based GANs: CIPS and INR-GAN on the FFHQ dataset. Table 2 shows that the proposed model significantly outperforms these models, even with a small generator model. Interestingly the Poly-INR method outperforms the StyleGAN-2 and performs comparable to StyleGAN-XL, using significantly fewer parameters. Table 2 also reports the inference speed of these models on a Nvidia-RTX-6000 GPU. StyleGANs and INR-GAN use a multi-scale architecture, resulting in faster inference. In contrast, CIPS and Poly-INR models perform all computations at the same resolution as the output image, increasing the inference time.

4.2. Qualitative results

Fig. 1 shows images sampled at different resolutions by the Poly-INR model trained on 512×512. We observe that

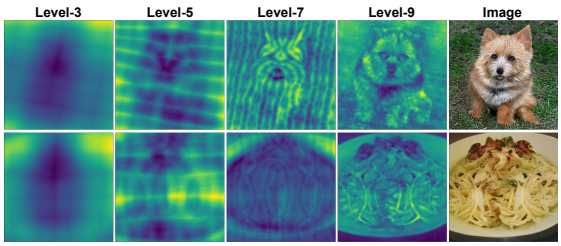

Figure 3. Heat-map visualization at different levels of the synthesis network. At initial levels, the model captures the basic shape of the object, and at higher levels, the image's finer details are captured.

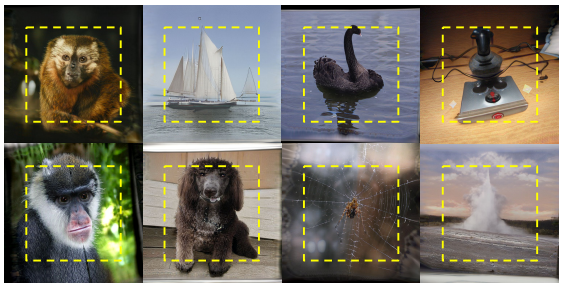

Figure 4. Few example images showing extrapolation outside the image boundary (yellow square). The Poly-INR model is trained to generate images on the coordinate grid $[0, 1]^2$. For extrapolation, we use the grid size $[-0.25, 1.25]^2$. Our model generates continuous image outside the conventional boundary.

our model generates diverse images with very high fidelity. Even though the model does not use convolution or self-attention layers, it generates realistic images over datasets like ImageNet. In addition, the model provides flexibility to generate images at different scales by changing the size of the coordinate grid, making the model efficient if low-resolution images are needed for a downstream task. In contrast, CNN-based models generate images only at the training resolution due to the non-equivariant nature of the convolution kernels to image scale.

Heat-map visualization: Fig. 3 visualizes the heat-map at different levels of our synthesis network. To visualize a feature as a heat-map, we first compute the mean along the spatial dimension of the feature and use it as a weight to sum the feature along the channel dimension. In the figure, we observe that in the initial levels (0-3), the model forms the basic structure of the object. Meanwhile, in the middle levels (4-6), it captures the object's overall shape, and in the higher levels (7-9), it adds finer details about the object. Furthermore, we can interpret this observation in terms of polynomial order. Initially, it only approximates low-order polynomials and represents only basic shapes. However, at higher levels, it approximates higher-order polynomials representing finer details of the image.

Extrapolation: The INR model is a continuous function of the coordinate location; hence we extrapolate the image by feeding the pixel location outside the conventional image boundary. Our Poly-INR model is trained to generate images on the coordinate grid defined by $[0, 1]^2$. We feed the grid size $[-0.25, 1.25]^2$ to the synthesis network to generate the extrapolated images. Fig. 4 shows a few examples of extrapolated images. In the figure, the region within the yellow square represents the conventional coordinate grid $[0, 1]^2$. The figure shows that our INR model not only generates a continuous image outside the boundary but also preserves the geometry of the object present within the yellow square. However, in some cases, the model generates a black or white image border, resulting from the image border present in some real images of the training set.

Table 3. FID score (lower the better) evaluated at 512×512 for models trained at a lower resolution and compared against classical interpolation-based upsampling.

Training Resolution	Nearest Neighbour	Bilinear	Bicubic	Poly-INR
32×32	184.39	112.28	73.86	65.15
64×64	89.24	72.41	42.97	36.30

Sampling at higher-resolution: Another advantage of using our model is the flexibility to generate images at any resolution, even if the model is trained on a lower resolution. We generate a higher-resolution image by sampling a dense coordinate grid within the $[0, 1]^2$ range. Table 3 shows the FID score evaluated at 512×512 for models trained on the lower-resolution ImageNet dataset. We compare the quality of upsampled images generated by our model against the classical interpolation-based upsampling methods. The table shows that our model generates crisper upsampled images, achieving a significantly better FID score than the classical interpolation-based upsampling method. However, we do not observe significant FID score improvement for our Poly-INR model trained on 128×128 or higher resolution against the classical interpolation techniques. This could be due to the limitations of the ImageNet dataset, which primarily consists of lower-resolution images than the 512×512. We used bilinear interpolation to prepare the training dataset at 512×512. As per our knowledge, there are currently no large and diverse datasets like ImageNet with high-resolution images. We believe this performance can be improved when the model has access to higher-resolution images for training. We also compare the upsampling performance with other INR-based GANs by reporting the FID scores at 1024×1024 for models trained on FFHQ-256×256 as follows: **Poly-INR:13.69, INR-GAN: 18.51, CIPS:29.59**. Our Poly-INR model provides better high-resolution sampling than the other two INR-based generators.

Interpolation: Fig. 5 shows that our model generates smooth interpolation between two randomly sampled images. In the first two rows of the figure, we interpolate in the latent space, and in the last two rows, we directly interpolate between the affine parameters. In our synthesis network, only the affine parameters depend on the image, and other parameters are fixed for every image. Hence interpolating in affine parameters space means interpolation in INR space.

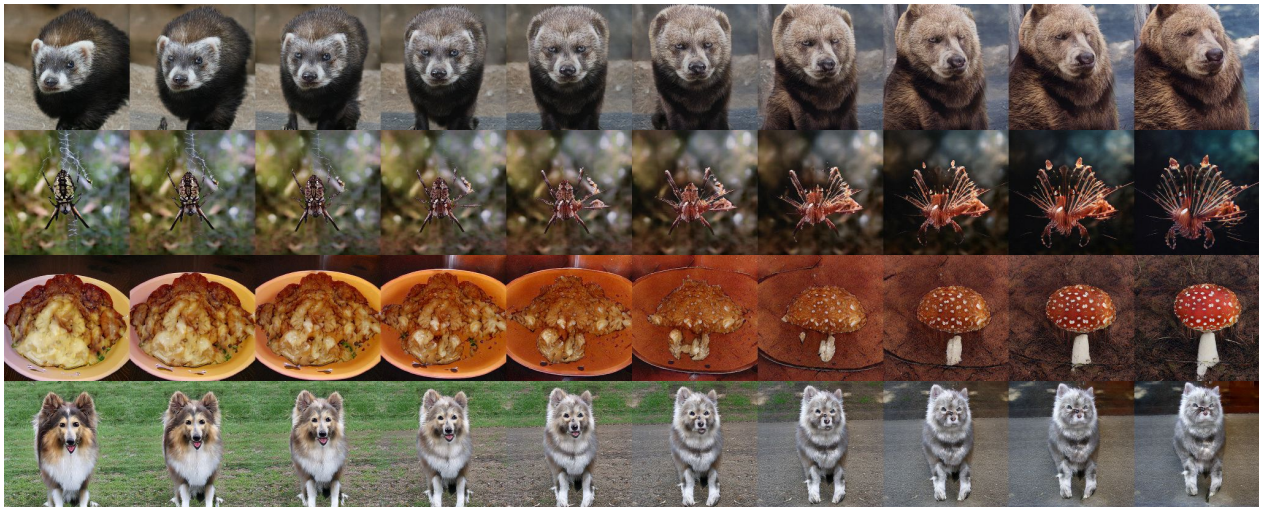

Figure 5. Linear interpolation between two random points. The first two rows represent interpolation in the latent space, while in the last two, we directly interpolate between the affine parameters. Poly-INR provides smooth interpolation even in a high dimension of affine parameters. Our model generates high-fidelity images similar to state-of-the-art models like StyleGAN-XL but without the need for convolution or self-attention mechanism. Comparisons with existing methods are present in the supplementary material.

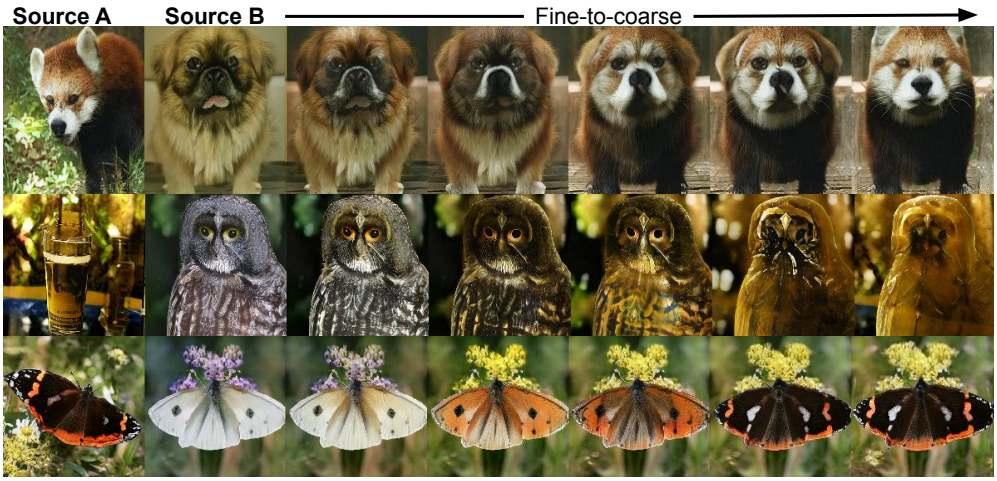

Figure 6. Source A and B images are generated corresponding to random latent codes, and the rest of the images are generated by copying the affine parameters of source A to source B at different levels. Copying the higher levels' (8 and 9) affine parameters leads to finer style changes, whereas copying the middle levels' (7, 6, and 5) leads to coarse style changes.

Figure 7. The Poly-INR model generates smooth interpolation with embedded images in affine parameters space. The leftmost image (first row) is from the ImageNet validation set, and the last two (rightmost) are the OOD images.

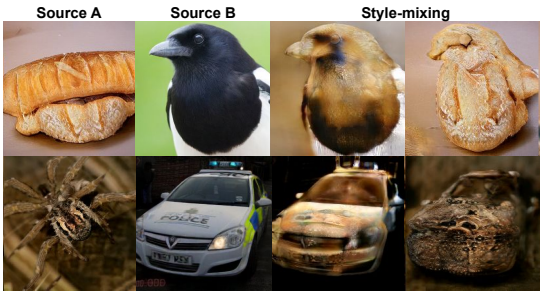

Figure 8. Style-mixing with embedded images in affine parameters space. Source B is the embedded image from the ImageNet validation set, mixed with the style of randomly sampled source A image.

Our model provides smoother interpolation even in the affine parameters space and interpolates with the geometrically coherent movement of different object parts. For example, in the first row, the eyes, nose, and mouth move systematically with the whole face.

Style-mixing: Similar to StyleGANs, our Poly-INR model transfers the style of one image to another. Our model generates smooth style mixing even though we do not use any style-mixing regularization during the training. Fig. 6 shows examples of style-mixing from source A to source B images. For style mixing, we first obtain the affine parameters corresponding to the source A and B images and then copy the affine parameters of A to B at various levels of the synthesis network. Copying affine parameters to higher levels (8 and 9) leads to finer changes in the style, while copying to middle levels (7, 6, and 5) leads to the coarse style change. Mixing the affine parameters at initial levels changes the shape of the generated object. In the figure, we observe that our model provides smooth style mixing while preserving the original shape of the source B object.

Inversion: Embedding a given image into the latent space of the GAN is an essential step for image manipulation. In our Poly-INR model, for inversion, we optimize the affine parameters to minimize the reconstruction loss, keeping the synthesis network's parameters fixed. We use VGG feature-based perceptual loss for optimization. We embed the ImageNet validation set in the affine parameters space for the quantitative evaluation. Our Poly-INR method effectively embeds images with high PSNR scores (**PSNR:26.52** and **SSIM:0.76**), better than StyleGAN-XL (PSNR:13.5 and SSIM:0.33). However, our affine parameters dimension is much larger than the StyleGAN-XL's latent space. Even though the dimension of the affine parameters is much higher, the Poly-INR model provides smooth interpolation for the embedded image. Fig. 7 shows examples of interpolation with embedded images. In the figure, the first row (leftmost) is the embedded image from Val set, and the last two rows (rightmost) are the out-of-distribution images. Surprisingly, our model provides smooth interpolation for OOD images. In addition, Fig. 8 shows smooth style-mixing with the embedded images. In some cases, we observe that the fidelity of the interpolated or style-mixed image with the embedded image is slightly less compared to samples from the training distribution. This is due to the large dimension of the embedding space, which sometimes makes the embedded point farther from the training distribution. It is possible to improve interpolation quality further by using the recently proposed pivotal tuning inversion method [39], which fine-tunes the generator's parameters around the embedded point.

4.3. Discussion

The proposed Poly-INR model performs comparably to state-of-the-art generative models on large ImageNet datasets without using convolution or self-attention layers. In addition to smooth interpolation and style-mixing, the Poly-INR model provides attractive flexibilities like image extrapolation and high-resolution sampling. In this work, while we use our INR model for 2D image datasets, it can be extended to other modalities like 3D datasets.

Challenges: One of the challenges in our INR method is the higher computation cost compared to the CNN-based generator model for high-resolution image synthesis. The INR method generates each pixel independently; hence all the computation takes place at the same resolution. In contrast, a CNN-based generator uses a multi-scale generation pipeline, making the model computationally efficient. In addition, we observe common GAN artifacts in some generated images. For example, in some cases, it generates multiple heads and limbs, missing limbs, or the object's geometry is not correctly synthesized. We suspect that the CNN-based discriminator only discriminates based on the object's parts and fails to incorporate the entire shape.

5. Conclusion

In this work, we propose polynomial function based implicit neural representations for large image datasets while only using Linear and ReLU layers. Our Poly-INR model captures high-frequency information and performs comparably to the state-of-the-art CNN-based generative models without using convolution, normalization, upsampling, or self-attention layers. The Poly-INR model outperforms previously proposed positional embedding-based INR GAN models. We demonstrate the effectiveness of the proposed model for various tasks like interpolation, style-mixing, extrapolation, high-resolution sampling, and image inversion. Additionally, it would be an exciting avenue for future work to extend our Poly-INR method for 3D-aware image synthesis on large datasets like ImageNet.

Acknowledgements

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112290073. Approved for public release; distribution is unlimited.

References

[1] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14278–14287, 2021. 1, 2, 3, 5

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 2

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 1, 2

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018. 4

[5] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8628–8638, 2021. 3

[6] Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 264–280. Springer, 2022. 2

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 4

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 1, 4

[9] Jan Flusser, Barbara Zitova, and Tomas Suk. *Moments and moment invariants in pattern recognition*. John Wiley & Sons, 2009. 3

[10] Elizabeth Fons, Alejandro Sztrajman, Yousef El-Laham, Alexandros Iosifidis, and Svitlana Vyetrenko. Hypertime: Implicit neural representations for time series. In *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*. 2

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

[12] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 2

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 1, 2

[14] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017. 2

[15] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Advances in Neural Information Processing Systems*, 33:9841–9850, 2020. 2

[16] 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. *Advances in neural information processing systems*, 30, 2017. 5

[17] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23:47–1, 2022. 4

[18] Barmak Honarvar, Raveendran Paramesran, and Chern-Loon Lim. Image reconstruction from a complete set of geometric and complex moments. *Signal Processing*, 98:224–232, 2014. 3

[19] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962. 3

[20] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021. 2

[21] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021. 3, 4

[22] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 2, 4

[23] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 4

[24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. 2013. 1

[25] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[26] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. In *International Conference on Learning Representations*, 2021. 2

[27] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, pages 7210–7219, 2021. 1, 2

[28] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 1, 2

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[30] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. 2

[31] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter Battaglia. Generating images with sparse representations. In *International Conference on Machine Learning*, pages 7958–7968. PMLR, 2021. 5

[32] Evangelos Ntavelis, Mohamad Shahbazi, Iason Kastanis, Radu Timofte, Martin Danelljan, and Luc Van Gool. Arbitrary-scale image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11533–11542, 2022. 3

[33] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

[34] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 4

[35] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[36] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[37] Sameera Ramasinghe and Simon Lucey. Beyond periodicity: towards a unifying framework for activations in coordinate-mlps. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 142–158. Springer, 2022. 2

[38] Albert W Reed, Hyojin Kim, Rushil Anirudh, K Aditya Mohan, Kyle Champley, Jingu Kang, and Suren Jayasuriya. Dynamic ct reconstruction from limited views with implicit neural representations and parametric motion fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2258–2268, 2021. 2

[39] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG)*, 42(1):1–13, 2022. 8

[40] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 5

[41] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. *Advances in Neural Information Processing Systems*, 34:17480–17492, 2021. 2

[42] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 2, 3, 4, 5

[43] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 1, 2

[44] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34:19313–19325, 2021. 2

[45] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[46] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10753–10764, 2021. 1, 3, 5

[47] Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. Aligning latent and image spaces to connect the unconnectable. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14144–14153, 2021. 3

[48] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 4

[49] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 2

[50] Michael Reed Teague. Image analysis via the general theory of moments. *Josa*, 70(8):920–930, 1980. 3

[51] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 4

[52] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 2

[53] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 2

[54] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural

surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33:2492–2502, 2020. 2

[55] Youngho Yoon, Inchul Chung, Lin Wang, and Kuk-Jin Yoon. Spheresr: 360deg image super-resolution with arbitrary projection via continuous spherical image representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5677–5686, 2022. 3

[56] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1, 2

[57] Long Zhao, Zizhao Zhang, Ting Chen, Dimitris Metaxas, and Han Zhang. Improved transformer for high-resolution gans. *Advances in Neural Information Processing Systems*, 34:18367–18380, 2021. 2