Implicit Representations via Operator Learning

Sourav Pal [1] **Harshavardhan Adepu** [1] **Clinton Wang** [2] **Polina Golland** [2] **Vikas Singh** [1]

Abstract

The idea of representing a signal as the weights of a neural network, called *Implicit Neural Representations* (INRs), has led to exciting implications for compression, view synthesis and 3D volumetric data understanding. One problem in this setting pertains to the use of INRs for downstream processing tasks. Despite some conceptual results, this remains challenging because the INR for a given image/signal often exists in isolation. What does the neighborhood around a given INR correspond to? Based on this question, we offer an operator theoretic reformulation of the INR model, which we call Operator INR (or O-INR). At a high level, instead of mapping positional encodings to a signal, O-INR maps one function space to another function space. A practical form of this general casting is obtained by appealing to Integral Transforms. The resultant model does not need multi-layer perceptrons (MLPs), used in most existing INR models – we show that convolutions are sufficient and offer benefits including numerically stable behavior. We show that O-INR can easily handle most problem settings in the literature, and offers a similar performance profile as baselines. These benefits come with minimal, if any, compromise. Our code is available at https://github.com/vsingh-group/oinr.

1. Introduction

If we treat a given signal as a map from the domain of measurement to the range space, can neural networks help estimate this mapping? One instantiation of this idea is popularly known as *Implicit Neural Representations* (INRs) (Sitzmann et al., 2020; Tancik et al., 2020; Mildenhall et al., 2021; Fathony et al., 2021) which can parameterize spatial/spatio-temporal data (Gropp et al., 2020; Niemeyer et al., 2019; Jiang et al., 2020) for applications in image super-resolution (Chen et al., 2021), texture synthesis (Oechsle et al., 2019; Sun et al., 2023), inverse problems (Sun et al., 2021; Yu et al., 2021b; Niemeyer et al., 2020), and view synthesis (Mildenhall et al., 2021; Sun et al., 2022).

[1]University of Wisconsin–Madison [2]Massachusetts Institute of Technology. Correspondence to: Sourav Pal <spal9@wisc.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

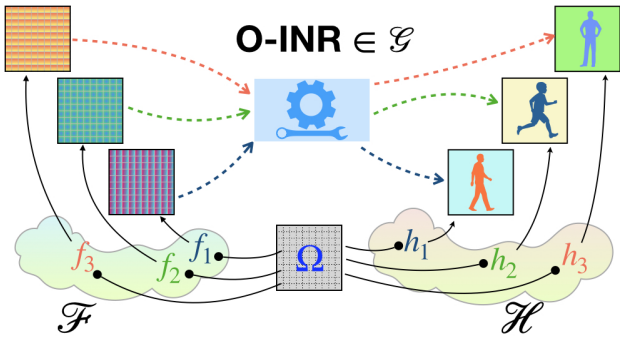


Figure 1. Overview of O-INR: $f_1, f_2, f_3 \in \mathcal{F}$ are input functions acting on the domain Ω. O-INR maps these functions to their corresponding signals (functions) $h_1, h_2, h_3 \in \mathcal{H}$.

From one signal to a set of signals. INRs typically consist of a neural network that is trained to map each coordinate of a given signal's domain to its measurements/values, and so are also known as *coordinate-value networks*. The mapping is learned via a neural network and gives a compact representation of the signal (Sitzmann et al., 2020; Fathony et al., 2021; Srinivasan et al., 2023). The discussion above describes the case of *one signal (or image)*. When given a *set* of signals, we may derive an INR for each signal in the set – Dupont et al. (2022a) then uses these functions (called functa) as data for downstream tasks. Alternatively, one can estimate a meta-learned "base" (INR) network, and associate each data sample (or signal) in the dataset as a *modulation* of the base network (Dupont et al., 2022b), akin to random effects modulating fixed effects in mixed effects models (Lindstrom & Bates, 1990). The modulation can also be accomplished in other ways as we will see later (Feng et al., 2022), via introducing a surrogate vector which is tied to a specific INR through conditioning. Now, if the data samples were *ordered* with respect to a surrogate variable, we get a set of INRs where each sample specific INR can be considered as a level set with discrete values denoting the levels of the surrogate variable. Our goal is to study and develop this interpretation, see Fig. 1.

This paper. A prevailing view is to consider the INR as a coordinate-value transform. We study a generalization where we still wish to parameterize a signal (i.e., same goal as INR) but as a transformation between two *function spaces*. Casting INRs in this manner yields an *operator-theoretic* view: our object of interest is the operator that takes us across function spaces. We model these transforms via *integral operators* (or integral transforms) which transform between function spaces via the process of integration. If we further constrain the integral operator to be local and translation-equivariant, we arrive at an efficient parameterization using convolutional layers. Other than its succinctness, we show how this approach gives benefits compared to coordinate-based INRs. The key **contributions** are:

(1) We introduce a new type of INR called Operator INR (O-INR) which yields comparable or superior empirical performance relative to common methods in terms of representation capability on 2D images and 3D scenes;
(2) While most INR parameterizations rely on large MLPs, we show that convolution operations with sinusoidal non-linearities are more efficient to train and evaluate.
(3) Higher order derivatives of O-INRs can be efficiently computed in closed form, allowing efficient processing in downstream tasks such as denoising.
(4) O-INR offers greater convenience (control over both the input function space and the weight space), including explicit control of the spatial interpolation behavior, mitigating the influence of initialization, and more interpretable behavior in weight space.

2. Setting up O-INRs

We denote the standard coordinate-valued network as $m_\theta : \mathbb{R}^D \to \mathbb{R}^R$ where $\theta \in \mathbb{R}^N$ denotes the parameters of the neural network, usually based on multi-layer perceptrons (MLPs). Here, D denotes the dimensionality of the domain and R denotes the range space of the continuous function being learned. For example, when fitting an INR to a 2D RGB image, $D = 2$ and $R = 3$.

Space of Discretized Positional Encodings: It is known that coordinate-value networks fail when coordinate locations are directly given as input (Tancik et al., 2020; Sitzmann et al., 2020). Workarounds suggest lifting the coordinates to higher-dimensions. These *positional encodings* on the coordinate space involve sinusoids across many frequencies (Mildenhall et al., 2021) or hashing (Müller et al., 2022; Xie et al., 2023). Consider the encoding $f((x, y))$

$$= \left[\sin(2^l \pi x), \cos(2^l \pi x), \sin(2^l \pi y), \cos(2^l \pi y), \ldots\right], \quad (1)$$

for $l \in \{0, \cdots, L-1\}$. We have many choices for this input, of the form $f(x, y) = [\sin(\theta x), \cos(\theta x), \sin(\theta y), \cos(\theta y)]$, where θ can even be a tunable parameter as in Zhou et al. (2021). In fact, it even makes sense to consider the entire family of such functions, say by varying θ, which share a common domain and co-domain. The corresponding space we will obtain is commonly referred to as a *function space*.

More importantly, for a sequence of signals defined on the *same* domain, Ω, the corresponding positional encodings $f_1, \cdots, f_n$ also act on Ω. These are different functions but belong to the same family w.r.t. their domain of definition, regularity properties, and so on. We must interpret this set to correspond to a well defined space of functions.

Even in the simplest case of two functions $f_1, f_2 \in \mathcal{F}$, the mapping via standard INRs will be independent. We will obtain two *separate* models. However, both are members within the family of functions discussed above. And due to how these functions are defined, there is extensive structure in the function space that can be utilized. Based on these observations, we define our input function space in terms of sinusoidal positional encodings (Sitzmann et al., 2020). Specifically,

$$\mathcal{F} = \{f\} \text{ where } f : \Omega \to \psi \quad (2)$$

where Ω is the domain of definition, e.g., 2D plane for images, and a 3D cube for volumes whereas ψ defines the space of sinusoidal positional encodings.

Signal spaces: In the above discussion, the elements of the function space $\mathcal{F}$ were considered to belong to the family of sinusoidal embedding functions. But INRs learn a map from the positional encoding space to the signal space. So, what do $f_1, f_2, \cdots, f_n$ yield after going through such a map? The answer is clear when we think of the associated co-domain of this map simply as another function space. We denote this function space of signals by $\mathcal{H}$, also defined on the domain Ω. Elements of $\mathcal{H}$, namely $h_1, h_2, \ldots, h_n$ are essentially the different signals (for example, frames in a video) whose corresponding embedding in $\mathcal{F}$ is $f_1, f_2, \ldots, f_n$.

From INRs to O-INRs: Many tasks can be posed as learning a map between two function spaces $\mathcal{F}$ and $\mathcal{H}$. We parameterize the transformation between these function spaces via a neural network (Rosasco et al., 2010; Que et al., 2014):

$$\mathcal{G}_\phi : f \to h \quad (3)$$

where ϕ represents the parameters of a DNN, and $f \in \mathcal{F}$ and $h \in \mathcal{H}$ are functions. We refer to this operator based formulation of an implicit representation as O-INR. While (3) gives a very general transform, we need a little more structure on the operator to allow efficient learning.

Integral operators: Let us assume a simplified setup. We want to learn a map from $f_1 \to h_1$, where $f_1 \in \mathcal{F}$ and $h_1 \in \mathcal{H}$. Consider the common domain to be $\Omega = \mathbb{R}$, the 1D real number line. The simplest map would be an identity mapping, resulting in $h_1(x) = f_1(x), \forall x \in \Omega$. At the

Ground truth | O-INR (32.14) | SIREN (31.84) | WIRE (22.1) | MFN (32.67)

Figure 2. Performance comparisons of O-INR in multi-resolution training setting. We show the ground truth together with reconstructions from O-INR and other baselines (L to R), with the PSNR value in dB. O-INR achieves comparable/better performance to baselines.

other extreme, we can write $h_1(x) = \mathcal{C}(x, \{f_1(y)|y \in \Omega\})$, where the value of $h_1(x)$ depends on all evaluations of f_1 via a functional $\mathcal{C}$. This can be written as an integral *along* the domain Ω,

$$h_1(x) = \int_{y\in\Omega} \mathcal{C}(x, f_1(y))\mathrm{d}y \tag{4}$$

This means that an integral operator achieves the transformation between the function spaces via integrating over the domain of definition. This is helpful: since integral operators are defined using their associated kernels, the only parameterization we need within O-INR will be this kernel!

Consider $f \in \mathcal{F}$ and $h \in \mathcal{H}$ as functions over the domain Ω, we learn an integral operator $\mathcal{G}_\phi$ with the associated kernel $\mathcal{K}_\phi$, where ϕ denotes the parameterization involved. Then, the integral transform can be represented as:

$$h(\omega) = \mathcal{G}\big[f\big](\omega) = \int_{\omega'\in\Omega} \mathcal{K}_\phi(\omega, \omega')f(\omega')\mathrm{d}\omega', \quad \omega \in \Omega \tag{5}$$

where $\mathcal{G}\big[f\big]$ denotes the application of the transform on the function f. Note that we recover the behavior of a standard coordinate-valued network if the kernel is modulated by a Dirac delta function: $\mathcal{K}_\phi(\omega, \omega') = K_\phi(\omega)\delta_{\omega'}(\omega)$. In which case, we have $h(\omega) = K_\phi(\omega)f(\omega) = \tilde{K}_{\tilde{\phi}}(f(\omega))$. Here, $\tilde{K}_{\tilde{\phi}}$ represents the corresponding standard INR with its own parameterization, $\tilde{\phi}$.

Interpretation via Green's Theorem: Recall that the Green's function is the fundamental solution to a linear differential operator in its Dirac delta inhomogeneous form. Given a linear differential operator $\mathcal{L}$, the Green's function $G(\omega, \omega')$ of the operator is any solution of:

$$\mathcal{L}G(\omega, \omega') = \delta(\omega' - \omega) \tag{6}$$

where δ is the Dirac delta function. This property is used to solve differential equations of the form:

$$\mathcal{L}u(\omega) = f(\omega) \tag{7}$$

which is given by

$$u(x) = \int G(\omega, \omega')f(\omega')\mathrm{d}\omega' \tag{8}$$

Note that (8) is exactly the same as (5). Viewing INRs as realizations of Green's function allows us to use various operators as INRs that have not been tried before. In particular, this allows us to use operators such as convolution and the Calderon-Zygmund operator (Beylkin et al., 1991; Pal et al., 2023) in place of MLPs.

How to parameterize O-INR? From (5), the *only* parameterization in our formulation is through $\mathcal{K}_\phi$. In its maximum capacity, the bi-variate function $\mathcal{K}_\phi$ can take distinct parameters for each pair of distinct (ω, ω'). While nearly all INR formulations perform pointwise evaluations with an MLP decoder, we can take advantage of our model and use convolution layers to parameterize O-INR. Considering the associated kernel to be a convolutional kernel, we have: $\mathcal{K}_\phi(\omega, \omega') = g_\phi(\omega - \omega')$. Therefore, with g_ϕ being the standard convolutional kernel, (5) becomes:

$$h(\omega) = \mathcal{G}\big[f\big](\omega) = \int_{\omega'\in\Omega} g_\phi(\omega - \omega')f(\omega')\mathrm{d}\omega', \quad \omega \in \Omega \tag{9}$$

In standard INRs, the mapping is a point-wise map, hence in the latent space (of INRs), adjacency does *not* have a semantic correspondence with the spatial dimension. But in O-INRs, the transform is obtained over the entire domain of definition and hence the use of location bias is permissible.

Remark 2.1. While we focus on convolutions because it is easy to understand, easy to train, and sufficient for the tasks we study, our formulation of O-INR is general and can be extended to other operators including Fourier neural operators and similar techniques that are used for efficient PDE solvers. We explore O-INRs parameterized as Calderon-Zygmund (CZ) operator in §5.

Multi-resolution training & Continuous convolutions: *How to sample at arbitrary resolution?* When using convolution kernels to parameterize O-INR, one drawback arises

Ground Truth | **O-INR (ours)** (32.9dB) | SIREN (30.4dB) | WIRE (31.5dB) | MFN (30.9dB)

Figure 3. Performance comparison of O-INR with other INR methods for 2D image representation. We show the ground truth together with reconstructions from O-INR and other baselines (L to R), with the PSNR value in dB.

when we want to sample the signal at *any arbitrary* resolution. This is because discrete convolutions cannot adapt their weights to different spacings, resulting in poor performance when changing resolution. A remedy is available via continuous convolutional kernels (Romero et al., 2021b). We also parameterize each INR as a continuous convolutional network, which is trained to map multiple resolutions of positional encodings of the domain of definition of the signal (e.g., the 2D plane) to corresponding resolutions of the desired signal (e.g., images).

Remark 2.2. For spatio-temporal data, the time dimension is treated differently in the context of the positional encoding as we will discuss in §4.

Remark 2.3. While the rationale of positional encoding is to provide high frequency signals as inputs to the model, our use of convolutional layers also makes it possible to simply use noise as a proxy for the high frequency positional encoding term. But this is a poor choice within INRs with MLP layers due to the lack of location bias.

Miscellaneous implementation details: While we use the aforementioned positional encoding in our experiments for CNNs, we also train O-INR with noise as an additional channel to provide high frequency components and it works well. For ease of implementation, and in cases where the sole purpose is to fit to one resolution of the data point, we can use discrete convolutions. We now discuss a set of experiments starting with the use of continuous convolutions for multi-resolution training.

3. Representation capability of O-INR

We first check the representation capability of O-INRs relative to standard INRs. We evaluate performance on 2D images as well as 3D volumes. Additionally we show that our proposed model can handle inverse problems such as image denoising. For 2D images, we use images from several sources including Agustsson & Timofte (2017) Kodak Image Suite, scikit-image, etc. For 3D volumes, we use data from the Stanford 3D Scanning Repository and Saragadam et al. (2022; 2023).

		Train	Fern	Coffee	Walnut	Rocket
	Size	510×339	510×339	400×600	510×339	427×640
PSNR	SIREN	24.88	28.27	29.91	26.09	30.39
	WIRE	34.13	37.16	31.51	33.21	31.46
	MFN	26.82	31.26	32.27	28.36	30.93
	iNGP	33.13	35.15	37.99	34.11	39.55
	NFFB	22.17	31.93	30.09	26.15	35.25
	O-INR	34.04	36.4	32.04	32.12	32.91
Time	SIREN	38.43	53.8	68.67	51.3	89.39
	WIRE	100.36	109.41	151.27	102.24	198.49
	MFN	89.7	88.05	117.76	82.06	151.99
	iNGP	88.0	68.2	101.3	64.83	109.63
	NFFB	39.0	39.3	39.7	39.8	40.5
	O-INR	57.1	57.0	56.0	57.0	56.2

Table 1. O-INR and baselines for 2D image representation. PSNR (in dB) and time in seconds show O-INR is comparable/better than baselines.

3.1. Multi-resolution training is possible

Task. We will assess the effectiveness of the multi-resolution training approach for O-INR. Given an image at a particular resolution, we train our model using its lower resolution versions (obtained by down sampling). Can our model effectively reconstruct images at an arbitrary resolution?

Setup. We compare our method to baselines including SIREN (Sitzmann et al., 2020), WIRE (Saragadam et al., 2023) and MFN (Fathony et al., 2021). Following (Saragadam et al., 2023), we train the baselines on the best resolution image seen by the O-INR during training. We then compare performance of all methods for reconstructions at the original (higher) resolution. Note that O-INRs with continuous convolutions can be trained at multiple resolutions.

Results summary. As seen in Fig. 2, O-INR achieves comparable or slightly better performance than baselines in terms of the Peak Signal to Noise Ratio (PSNR). Due to the use of continuous convolutions, the number of parameters required for O-INR are much smaller (100K) compared to baseline models ($\sim$130K) to achieve parity in performance.

3.2. 2D Image representation effectiveness

Task. A prominent use case of INRs is in representing spatio-temporal signals. So, is O-INR effective at fitting 2D images of varying resolutions?

Setup. We compare O-INR with sinusoidal representation

Ground Truth | Noisy | O-INR (24.28) | SIREN (20.94) | WIRE (24.48) | MFN **(25.22)**

Figure 4. Performance comparison of O-INR to other INRs for image denoising. For each method, we show the PSNR for the image in dB. Among all methods, SIREN achieves the lowest PSNR, while O-INR and other baselines perform similarly.

networks (SIREN) (Sitzmann et al., 2020), wavelet implicit neural representations (WIRE) (Saragadam et al., 2023), multiplicative filter networks (MFN) (Fathony et al., 2021), Instant NGP (iNGP) (Müller et al., 2022), and neural Fourier filter banks (NFFB) (Wu et al., 2023) based on PSNR and training time to reach the best possible PSNR for that specific model. While other INR models use MLP layers, our model is solely parameterized by convolution layers. The number of parameters in each model is comparable.

Results summary. Table 1/Fig. 3 shows that O-INR achieves comparable or better performance than baseline methods in terms of PSNR, only outperformed by iNGP. In terms of training time, O-INR is faster than every method except NFFB and SIREN, but O-INR's quality is better.

3.3. Application to Image Denoising

Task. Our task is to assess the robustness of O-INR: is it effective at representing noisy images?

Setup. Given an image, following (Saragadam et al., 2023), we add photon noise for each pixel via independently distributed Poisson r.v. (maximum mean photon count 30, readout count 2). These noisy images are then used to learn O-INR models. We compare performance with SIREN, WIRE, MFN, Instant NGP and NFFB. As in previous experiments, all models have comparable number of parameters.

Results summary. From Tab. 2/Fig. 4, we see that O-INR is able to recover the true signal to a similar degree as other methods. The methods with an explicit spatial component in their representation (iNGP and NFFB) tend to memorize the signal and thus overfit the noise. This experiment (also see appendix) indicates O-INR's effectiveness in solving some inverse problems.

3.4. 3D Volume representation

Task. INRs are commonly used as a continuous representation of 3D volumes or surfaces. How well can O-INR encode 3D volumetric data?

Setup. We consider occupancy volume sampled over a $512 \times 512 \times 512$ voxel grid, where each voxel *within* the volume is assigned a value of 1 inside an object and 0 otherwise. We compare O-INR with SIREN, WIRE and MFN based on intersection over union (IoU). We ensure a similar number of parameters when comparing with baselines.

Results summary. Fig. 5 shows that O-INR performs well in IoU in all cases. For SIREN and WIRE, we report the best performance (achieved with a model with slightly fewer number of parameters). Increasing the parameters of SIREN and WIRE involves an interplay with other hyperparameters. We should clarify that the convolutional parameterization of O-INR in 3D and higher dimensions can be computationally expensive.

4. Representing a sequence of signals/functions

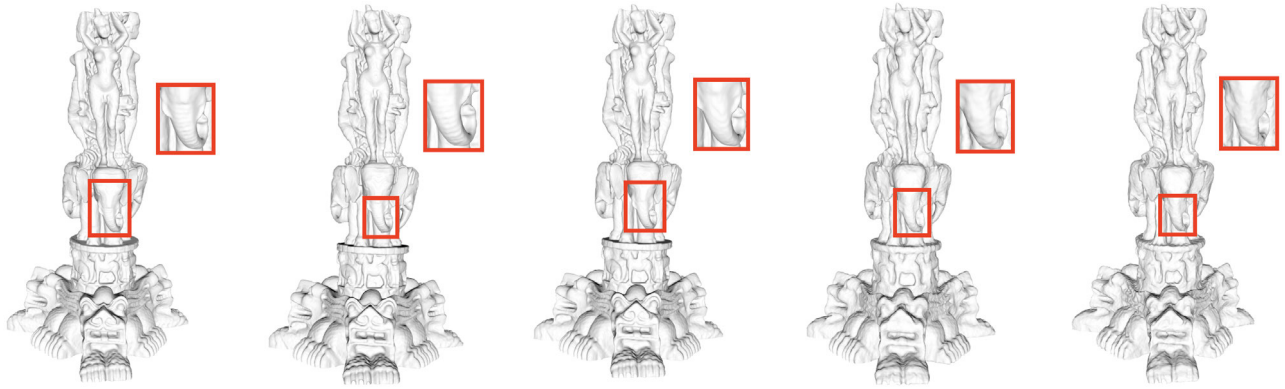
Task. Given a sequence of signals captured over a predefined fixed domain, it is natural to consider the data as a sequence of functions defined over the domain yielding a sequence of functions (or signals). For example, frames in a short-burst video are a sequence of images (captured by different functions over the same domain).

In standard INR formulations, such signals are represented by considering an additional parameter (usually time) in the domain of definition and parameterized using $m_\theta : (x, y, t) \to (r, g, b)$. While this is reasonable, a more natural approach from a operator (functional) perspective is to consider the sequence of frames as different (but related) functions acting on the same domain rather than a function (with a rather large redundancy) acting on the spatio-temporal volume. Our experiment checks if O-INR is effective here.

Setup. We consider learning a transform between spaces consisting of sequence of functions. More precisely, in this case O-INR takes the following form:

$$\begin{aligned} \mathcal{G}_\phi : \mathcal{F}_N \to \mathcal{H}_N; \quad \mathcal{F}_N = \{f_n(\Omega) | n \in \mathbb{N}\} \\ \mathcal{H}_N = \{h_n(\Omega) | n \in \mathbb{N}\} \end{aligned} \tag{10}$$

where ϕ denotes the to-be-learned parameters and Ω is the domain of definition. We use sinusoidal positional encodings as the input function space. A key question here is how to define a sequence of functions over the domain under consideration, while still ensuring that all such functions



Ground Truth | **O-INR (0.9999)** | MFN (0.9946) | SIREN (0.9665) | WIRE (0.9658)

Figure 5. Performance comparison of O-INR to other INRs for surface reconstruction. O-INR achieves the best performance among all methods, capturing fine details in the geometry (see inset).

provide both low and high frequency signals as an input to our O-INR. Here, we consider the domain Ω as the 2D plane over which frames are defined, with $(x, y) \in \Omega$ and $\gamma_n = \alpha + ((\beta - \alpha)/N)n$ as the frame-specific offset which is added to the standard coordinate embeddings in order to fit different frames with the same convolutional weights.

$$f_n(x, y) = [\sin(2^l \pi x) + \gamma, \cos(2^l \pi x) + \gamma, \sin(2^l \pi y) + \gamma, \cos(2^l \pi y) + \gamma, \ldots] \tag{11}$$

where $l \in \{0, \cdots, L-1\}$ for some $L \in \mathbb{N}$, levels of frequencies chosen to be part of the input signal. Here, α, β are empirically determined constants and N is the total number of functions that the O-INR is trained to encode.

Results summary. We trained O-INR on 100 randomly sampled videos from the UCF-101 (Soomro et al., 2012) dataset and 300 randomly sampled GIFs from the TGIF dataset (Li et al., 2016). Fig. 6, shows an example from the UCF dataset (appendix includes more examples). We find that these results are comparable with SIREN. We achieve an average PSNR of 43.76dB and 42.78dB on the UCF and the TGIF datasets respectively. Thus, we can conclude that O-INR can accurately represent frames in videos exploiting the regularity property of the function space associated with a related sequence of signals.

5. Calderon-Zygmund Operators as O-INR

Task. Does the formulation allow the applicability of the model to operators other than convolution?

Setup. We consider the Calderon-Zygmund (CZ) operator (Beylkin et al., 1991; Pal et al., 2023) (Appendix A) to map the coordinates of the pixels in an image to their RGB values.

Result Summary. As can be seen from Fig. 7, this choice of CZ operator in O-INR enables it to faithfully represent

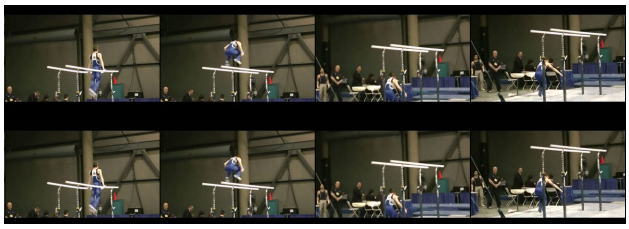
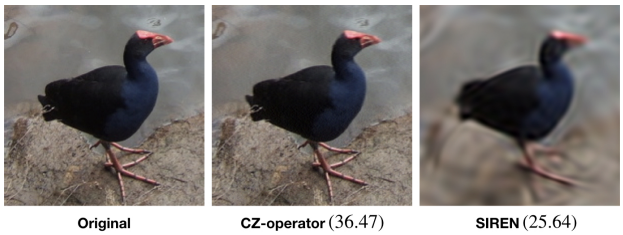


Figure 6. Rows show frames from a video from the UCF-101 dataset (Soomro et al., 2012). (Top) Original and (Bottom) predicted from O-INR trained on sparsely sampled frames of a long video sequence. O-INR represents the scenes in the sequence well.



Original | **CZ-operator** (36.47) | **SIREN** (25.64)

Figure 7. The CZ-Operator in O-INR can successfully represent images with high PSNR values.

an example image from ImageNet-1K (Deng et al., 2009) dataset. We notice that a PSNR of 36.47dB is achieved which is higher than SIREN.

6. Brain Imaging: Slice Imputation

Task. Available software tools (like FreeSurfer (Fischl, 2012)) for the analysis of brain imaging data target high resolution scans, acquired within research studies.

However, as noted in (Dalca et al., 2018), typical clinical (non-research) scans have much lower out of plane resolution (often for slice-by-slice reading by radiologists).

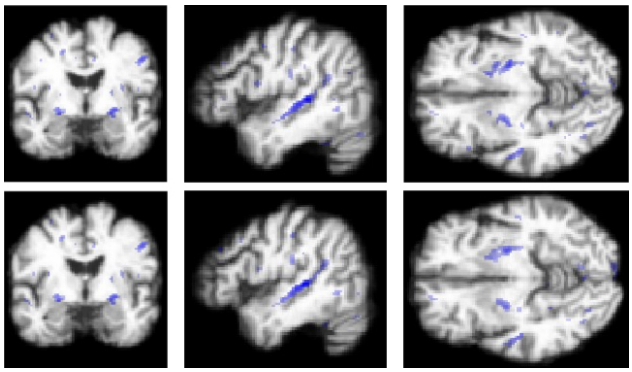


Figure 8. Top and Bottom rows: Overlay of filtered statistic image from group difference analysis of original (full resolution) and images generated via O-INR trained on sparsely sampled (30% slices) AD and CN images respectively. The above images indicate that O-INR is successful in preserving the group difference in 3D brain imaging data.

	Astronaut	Cat	Kodak05	Kodak19	Rocket
Size	512×512	300×451	512×768	256×171	427×640
SIREN	20.94	24.8	18.27	22.13	25.27
WIRE	24.48	27.4	20.47	24.6	26.27
MFN	25.22	24.8	23.6	20.96	25.78
iNGP	20.07	19.75	21.17	18.61	21.84
NFFB	19.83	21.14	23.45	21.39	26.18
O-INR	24.28	25.1	21.9	22.7	26.11

Table 2. Comparison of PSNR values (in dB) for O-INR and baselines for 2D image denoising.

Processing such scans with existing tools poses difficulties, and the results often need to be manually checked. Even partially mitigating this issue can radically increase sample sizes available for scientific analysis. We demonstrate the use of O-INR in representing such low resolution brain imaging data in (a) obtaining a faithful representation and (b) preserving statistical group differences.

Setup. We consider MRI data from Alzheimer's Disease Neuroimaging Initiative (ADNI) (Mueller et al., 2005; Jack Jr et al., 2008) and model 2D slices in a 3D brain scan via a sequence of functions as described in §4. Our data includes approximately 140 subjects each from cognitively normal (CN) and diseased (AD) individuals. O-INRs are trained for each image. For each 3D image, we progressively dropped more slices in one out of plane direction to simulate poor resolution, and use it for training.

Results summary. We observe that O-INR remains robust up to a high percentage of missing slices. The MSE of the reconstructed brain volumes reported in Tab. 3 show that O-INR is capable of learning the representation well even when more than 80% of the slices are dropped. More realistic downsampling, as in (Zhao et al., 2020), can also be adopted but is not used in our experiments. Finally, we use SnPM (Statistical NonParametric Mapping) toolbox

% missing	50	66	75	80	82
Train MSE	2.17e-5	3.10e-5	1.39e-6	1.58e-5	2.69e-6
Test MSE	8.33e-5	2.38e-4	4.70e-4	7.66e-4	1.08e-3

Table 3. Train/test MSE for 3D brain images with % of missing slices during training O-INR.

(Ashburner, 2010) to perform a statistical group difference analysis (voxel-wise t-test) on the real data (CN versus AD). Then, the same analysis was performed on O-INR derived data (CN versus AD). We find that voxels reported to be significant (uncorrected p-values) on the real data analysis overlap with the analysis on data based on O-INR slice imputation. Sizable clusters agree although the spatial extent is reduced (higher Type 2 error). Statistical analysis results with brain image underlay in Fig. 8 and additional results from group difference analysis are in Appendix §I.

7. Learning downstream tasks on O-INRs

Task. Using INRs for downstream tasks is an exciting emergent problem setting. Recently (Navon et al., 2023b) proposed an equivariant architecture for learning in the so-called "deep weight spaces" of INRs. However, in general, with standard INRs, signal processing operations on the latent space of MLPs remain difficult (Xu et al., 2022). Most methods must resort to discretization leading to loss in properties like continuity. The result in (Xu et al., 2022) explores the use of differential operators on INR: it is interesting but is not memory efficient (see pp 10 (Xu et al., 2022)). Since the O-INR operates on function spaces, many operations in signal processing (e.g., evaluating derivatives) are incredibly easy in principle. So, can these benefits be verified in practice?

Setup. When using O-INR to encode a signal e.g., an image, the signal is represented as the convolution of a known simple signal (e.g., a positional encoding) with a sequence of learned kernels. For ease of presentation, consider the domain to be one dimensional. Then our model is:

$$h(x) = f(x) * g(x) \tag{12}$$

where $h(x)$ is the true signal we want to represent, $f(x)$ is the positional encoding and $g(x)$ is the learned transform (convolution filters) between $f(x)$ and $h(x)$. When taking into account our multi-layer convolutional model with sine non-linearities, (12) can be written as (e.g., for 3 layers):

$$h(x) = (\sin(\sin(f(x) * g_1(x))) * g_2(x)) * g_3(x) \tag{13}$$

We make use of the property of computing derivatives over the convolution operation, namely:

$$h(x) = f(x) * g(x) \implies h'(x) = f'(x) * g(x) \tag{14}$$

Then, the derivative of our original signal is:

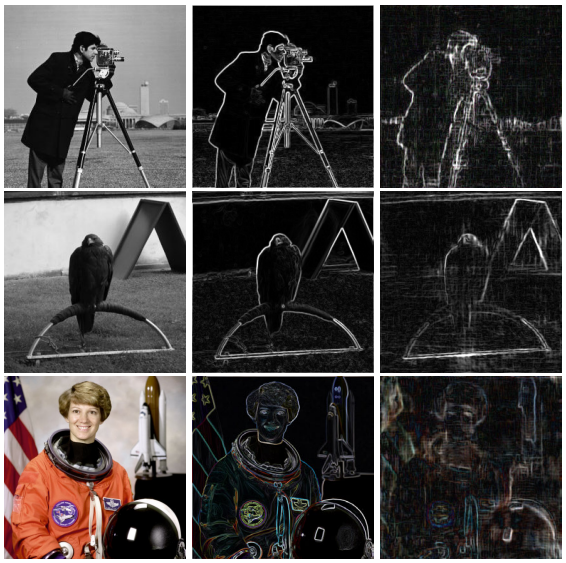


Figure 9. (L to R) Original image, true gradient of the image via Sobel filter and gradient obtained via O-INR (see §7). We see that the O-INR derivative closely matches the true derivative in all cases. Small discrepancies are due to the residual between the true image and its O-INR representation.

$$h'(x) = (\sin(\sin(f(x) * g_1(x))) * g_2(x))' * g_3(x) \quad (15)$$

which on repeated application of (14) and the chain rule leads to $\cos(\sin(f(x) * g_1(x))) * g_2(x)) \odot \cos(f(x) * g_1(x)) \odot f'(x) * g_1(x) * g_2(x) * g_3(x)$, where $\odot$ denotes point-wise multiplication and $*$ denotes convolution operation. Extension to higher order derivatives follows similarly.

Results summary. We show the effectiveness of this approach in computing derivatives in Fig. 9. We see that once an O-INR is trained, it can map different functions to their desired signals. Here, the input functions are the positional encodings of the grid and its derivative, which are then mapped via O-INR to their corresponding outputs: signal (image) and its first-order gradient. This shows that O-INR allows seamless calculus operations in the function space, a functionality difficult to achieve otherwise (Xu et al., 2022) thereby limiting the choice of baselines to the numerically obtained ground truth derivative.

We leverage this capability of computing gradients seamlessly in the recently proposed work (Chen et al., 2023) where INRs are used to solve time-dependent PDEs. The method involves computing the gradients of the network w.r.t. the inputs which is achieved via back-propagation. We simply replace this INR with our proposed O-INR and hence are able to compute such gradients simply via a forward pass as demonstrated above. It provides speeds up by 50% relative to (Chen et al., 2023) in iterations per sec.

We compared O-INR with Deep Weight-Space Networks (DWSNets) (Navon et al., 2023a). We evaluated classifying images represented as O-INR, i.e. we performed a head to head comparison of O-INR with DWSNets. For both MNIST and Fashion-MNIST, we trained O-INR and then used a NN-classifier (with 10 neighbors) on the trained model weights to measure test set accuracy. As can be seen from Table 4, O-INR achieves excellent performance in this task. This shows that the weights of O-INR correlate very well with the underlying data requiring minimal processing for downstream applications such as classification.

	MNIST	Fashion-MNIST
DWSNets	85.71	67.06
O-INR	**96.57**	**80.18**

Table 4. Test set classification accuracy obtained via NN-classifier on the weights of DWSNets and O-INR on MNIST and Fashion-MNIST datasets.

8. O-INR weight interpolation

Task. We verified above that operations like derivatives are possible, suggesting that the structure in $\mathcal{G}$ can be queried. So, does this ability allow other operations (interpolation)?

Setup. We investigate whether the convolutional weight space of O-INRs produces a more structured latent space than coordinate-based networks. One way to do this is by visualizing interpolations between O-INRs fit on different images from the CelebA dataset (Liu et al., 2015).

We should note that no generative model was trained on CelebA – the *only* two images that O-INR sees are the ones being interpolated.

Results summary. (Ainsworth et al., 2022) demonstrated the use of special weight-matching algorithms to align two models in the weight space. Using this idea, we permute the channel ordering of one O-INR's layers to minimize the total cosine distance between the activation statistics of the two O-INRs. Interpolation results presented in Fig. 10 use this strategy. Interestingly, even without an explicit weight matching, we find that all trends hold. We find that performing linear interpolation between the convolutional weights (in O-INR) corresponding to different images leads to reasonable and interpretable outputs, whereas interpolating individual layers in coordinate-based MLPs like SIREN does not yield coherent outputs Fig. 10. More examples of such interpolations obtained by manipulating individual convolutional layers are in Appendix §J. It is worthwhile to note that in (Skorokhodov et al., 2021), an explicit coupling variable was used to enforce continuity and hence interpolate between INRs. This is compute intensive as noted in (Skorokhodov et al., 2021). In our case, in Fig. 10, O-INRs are trained independently (no coupling was utilized). We simply move from one O-INR to the other.

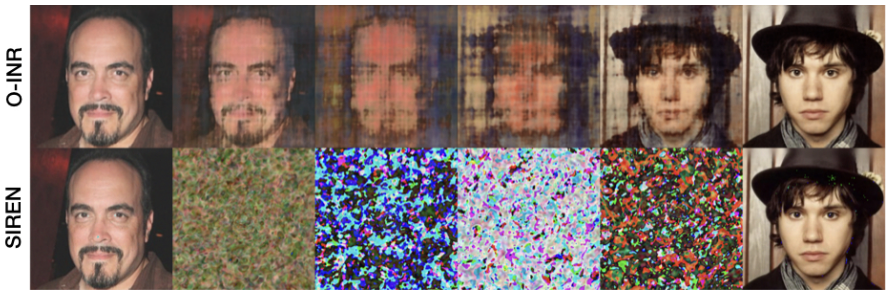


Figure 10. Interpolations between two CelebA images fit with SIREN versus O-INR. All layer weights are *linearly* interpolated.

9. Related Work

Implicit Neural Representations: INRs are useful in a wide range of tasks. By virtue of learning a continuous mapping, INRs can be sampled at any resolution thereby making them applicable in super-resolution and denoising (Saragadam et al., 2022; 2023; Peng et al., 2020). Other tasks including 3D rendering, boundary value problems, PDEs and generative modeling (Skorokhodov et al., 2021; Esmaeilzadeh et al., 2020; Schwarz et al., 2020) have also been studied using INRs. In (Shaham et al., 2021), the authors leveraged INRs for high resolution image to image translation. While the original development of INRs was intended for Euclidean data, more general non-Euclidean domains have been studied recently as in (Grattarola & Vandergheynst). Many works have also adapted INRs for scene representation (Niemeyer & Geiger, 2021; Guo et al., 2020; Yu et al.) and scene editing (Yuan et al., 2022; Feng et al., 2022; Fan et al., 2022; Gong et al., 2023).

Recall that the original formulation of an INR is as a multi-layer perceptron (MLP). Various reparameterizations have been developed that seek to offload computation to other components to enable efficiency, especially for their use in NeRFs. For example, plenoxels (Fridovich-Keil et al., 2022) and plenoctrees (Yu et al., 2021a) represent a radiance field with explicit voxel or octree structures and DirectVoxGO (Sun et al., 2022) stores features on a voxel grid which is decoded into radiance values with a tiny MLP.

A few approaches are available for learning in the context of downstream tasks using INRs (Wang & Golland, 2022; Xu et al., 2022; Dupont et al., 2022a;b). The use of differential operators on INRs has been shown in (Xu et al., 2022) by treating INRs as functions which enables modification without explicit decoding. Further, in Wang & Golland (2022), the authors treat neural fields as integrable maps and propose discretization invariant layers that map elements of this function space for use in DNN models. Other works focus on learning in INR weight space (Dupont et al., 2022a; Navon et al., 2023a), or represent transformations of the underlying signal (e.g., modeling the evolution of a PDE) by modulating INR weights over time (Chen et al., 2023).

Continuous convolutions: Since discrete convolutions learn weights which are tied to the relative positions, continuous convolutions were initially designed to handle irregularly sampled data (Schütt et al., 2017; Simonovsky & Komodakis, 2017; Wu et al., 2019). Continuous time convolutions are well studied, but their recent use in deep learning applications includes modeling point clouds (Wang et al., 2021; Boulch, 2019), graphs (Fey et al., 2017), fluids (Ummenhofer et al., 2019), and even sequential data (Romero et al., 2021b;a). We note that the use of convolutions within INRs is rare (Peng et al., 2020). In most settings above, irregular sampling intervals can be handled while maintaining locality and translation invariance. CNNs for modeling long range dependencies in arbitrary number of dimensions have also been studied (Romero et al., 2022).

10. Conclusions

O-INR is a novel approach for fitting INRs that treats coordinate encodings as a function space, offering efficient and compact training on complex signals and particularly sequences of signals. O-INR also leverages the properties of convolutions and sinusoidal activations to produce fast closed form derivatives useful in downstream tasks as well as an interpretable latent weight space. O-INR is effective on a wide range of problems, and requires little to no hyper-parameter tuning. Future work will expand on some possibilities enabled by O-INR, such as fitting a radiance field with a single interpolation-free CNN whose input captures all relevant information about camera rays and their query points. We also hope to address some limitations, e.g., maintaining high performance and efficiency on arbitrary non-grid inputs, relevant in many 3D applications. Another direction is to use sparse grids to handle higher dimensions.

Acknowledgments

This work was supported by NIH RF1 AG059312, funding from the Vilas Board of Trustees and the MIT CSAIL-Wistron Program. The authors are grateful to the reviewers and the Area Chair for many helpful suggestions and to Alan McMillan, Ashish Raj and Lopa Mukherjee for feedback.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Agustsson, E. and Timofte, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 126–135, 2017.

Ainsworth, S., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*, 2022.

Ashburner, J. Vbm tutorial. *Tech. repWellcome Trust Centre for Neuroimaging, London, UK*, 2010.

Beylkin, G., Coifman, R., and Rokhlin, V. Fast wavelet transforms and numerical algorithms i. *Communications on pure and applied mathematics*, 44(2):141–183, 1991.

Boulch, A. Generalizing discrete convolutions for unstructured point clouds. *CoRR*, abs/1904.02375, 2019.

Chen, H., Wu, R., Grinspun, E., Zheng, C., and Chen, P. Y. Implicit neural spatial representations for time-dependent pdes. In *International Conference on Machine Learning*, 2023.

Chen, Y., Liu, S., and Wang, X. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8628–8638, 2021.

Dalca, A. V., Bouman, K. L., Freeman, W. T., Rost, N. S., Sabuncu, M. R., and Golland, P. Medical image imputation from image collections. *IEEE transactions on medical imaging*, 38(2):504–514, 2018.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Dupont, E., Kim, H., Eslami, S., Rezende, D., and Rosenbaum, D. From data to functa: Your data point is a function and you should treat it like one. *arXiv preprint arXiv:2201.12204*, 2022a.

Dupont, E., Loya, H., Alizadeh, M., Golinski, A., Teh, Y. W., and Doucet, A. Coin++: Neural compression across modalities. *Transactions on Machine Learning Research*, 2022(11), 2022b.

Esmaeilzadeh, S., Azizzadenesheli, K., Kashinath, K., Mustafa, M., Tchelepi, H. A., Marcus, P., Prabhat, M., Anandkumar, A., et al. Meshfreeflownet: A physics-constrained deep continuous space-time super-resolution framework. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15. IEEE, 2020.

Fan, Z., Jiang, Y., Wang, P., Gong, X., Xu, D., and Wang, Z. Unified implicit neural stylization. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XV*, pp. 636–654. Springer, 2022.

Fathony, R., Sahu, A. K., Willmott, D., and Kolter, J. Z. Multiplicative filter networks. In *International Conference on Learning Representations*, 2021.

Feng, B. Y., Jabbireddy, S., and Varshney, A. Viinter: View interpolation with implicit neural representations of images. In *SIGGRAPH Asia 2022 Conference Papers*, pp. 1–9, 2022.

Fey, M., Lenssen, J. E., Weichert, F., and Müller, H. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. *CoRR*, abs/1711.08920, 2017.

Fischl, B. Freesurfer. *Neuroimage*, 62(2):774–781, 2012.

Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5501–5510, 2022.

Gong, B., Wang, Y., Han, X., and Dou, Q. Recolornerf: Layer decomposed radiance field for efficient color editing of 3d scenes. *arXiv preprint arXiv:2301.07958*, 2023.

Grattarola, D. and Vandergheynst, P. Generalised implicit neural representations. In *Advances in Neural Information Processing Systems*.

Gropp, A., Yariv, L., Haim, N., Atzmon, M., and Lipman, Y. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pp. 3789–3799. PMLR, 2020.

Guo, M., Fathi, A., Wu, J., and Funkhouser, T. Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503*, 2020.

Jack Jr, C. R., Bernstein, M. A., Fox, N. C., Thompson, P., Alexander, G., Harvey, D., Borowski, B., Britson, P. J., L. Whitwell, J., Ward, C., et al. The alzheimer's disease neuroimaging initiative (adni): Mri methods. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 27(4):685–691, 2008.

Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T., et al. Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6001–6010, 2020.

Journé, J.-L. *Calderón-Zygmund operators, pseudo-differential operators and the Cauchy integral of Calderón*, volume 994. Springer, 2006.

Li, Y., Song, Y., Cao, L., Tetreault, J., Goldberg, L., Jaimes, A., and Luo, J. Tgif: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4641–4650, 2016.

Lindstrom, M. J. and Bates, D. M. Nonlinear mixed effects models for repeated measures data. *Biometrics*, pp. 673–687, 1990.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Mueller, S. G., Weiner, M. W., Thal, L. J., Petersen, R. C., Jack, C. R., Jagust, W., Trojanowski, J. Q., Toga, A. W., and Beckett, L. Ways toward an early diagnosis in alzheimer's disease: the alzheimer's disease neuroimaging initiative (adni). *Alzheimer's & Dementia*, 1(1):55–66, 2005.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, January 2022.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4): 1–15, 2022.

Navon, A., Shamsian, A., Achituve, I., Fetaya, E., Chechik, G., and Maron, H. Equivariant architectures for learning in deep weight spaces, 2023a.

Navon, A., Shamsian, A., Achituve, I., Fetaya, E., Chechik, G., and Maron, H. Equivariant architectures for learning in deep weight spaces. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25790–25816. PMLR, 23–29 Jul 2023b.

Niemeyer, M. and Geiger, A. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

Niemeyer, M., Mescheder, L., Oechsle, M., and Geiger, A. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5379–5389, 2019.

Niemeyer, M., Mescheder, L., Oechsle, M., and Geiger, A. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3504–3515, 2020.

Oechsle, M., Mescheder, L., Niemeyer, M., Strauss, T., and Geiger, A. Texture fields: Learning texture representations in function space. In *Proceedings IEEE International Conf. on Computer Vision (ICCV)*, 2019.

Pal, S., Zeng, Z., Ravi, S. N., and Singh, V. Controlled differential equations on long sequences via non-standard wavelets. In *International Conference on Machine Learning*, pp. 26820–26836. PMLR, 2023.

Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., and Geiger, A. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pp. 523–540. Springer, 2020.

Que, Q., Belkin, M., and Wang, Y. Learning with fredholm kernels. *Advances in neural information processing systems*, 27, 2014.

Romero, D. W., Bruintjes, R., Tomczak, J. M., Bekkers, E. J., Hoogendoorn, M., and van Gemert, J. C. Flexconv: Continuous kernel convolutions with differentiable kernel sizes. *CoRR*, abs/2110.08059, 2021a.

Romero, D. W., Kuzina, A., Bekkers, E. J., Tomczak, J. M., and Hoogendoorn, M. Ckconv: Continuous kernel convolution for sequential data. *CoRR*, abs/2102.02611, 2021b.

Romero, D. W., Knigge, D. M., Gu, A., Bekkers, E. J., Gavves, E., Tomczak, J. M., and Hoogendoorn, M. Towards a general purpose cnn for long range dependencies in nd. *arXiv preprint arXiv:2206.03398*, 2022.

Rosasco, L., Belkin, M., and De Vito, E. On learning with integral operators. *Journal of Machine Learning Research*, 11(2), 2010.

Saragadam, V., Tan, J., Balakrishnan, G., Baraniuk, R. G., and Veeraraghavan, A. Miner: Multiscale implicit neural representation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27,*

2022, Proceedings, Part XXIII, pp. 318–333. Springer, 2022.

Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., and Baraniuk, R. G. Wire: Wavelet implicit neural representations. *arXiv preprint arXiv:2301.05187*, 2023.

Schütt, K., Kindermans, P.-J., Sauceda Felix, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

Schwarz, K., Liao, Y., Niemeyer, M., and Geiger, A. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33: 20154–20166, 2020.

Shaham, T. R., Gharbi, M., Zhang, R., Shechtman, E., and Michaeli, T. Spatially-adaptive pixelwise networks for fast image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14882–14891, 2021.

Simonovsky, M. and Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3693–3702, 2017.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

Skorokhodov, I., Ignatyev, S., and Elhoseiny, M. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10753–10764, 2021.

Soomro, K., Zamir, A. R., and Shah, M. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Srinivasan, P. P., Garbin, S. J., Verbin, D., Barron, J. T., and Mildenhall, B. Nuvo: Neural uv mapping for unruly 3d representations. *arXiv preprint arXiv:2312.05283*, 2023.

Sun, C., Sun, M., and Chen, H.-T. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5459–5469, 2022.

Sun, J., Wang, X., Wang, L., Li, X., Zhang, Y., Zhang, H., and Liu, Y. Next3d: Generative neural texture rasterization for 3d-aware head avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023.

Sun, Y., Liu, J., Xie, M., Wohlberg, B., and Kamilov, U. S. Coil: Coordinate-based internal learning for imaging inverse problems. *arXiv preprint arXiv:2102.05181*, 2021.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

Ummenhofer, B., Prantl, L., Thuerey, N., and Koltun, V. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2019.

Wang, C. J. and Golland, P. Deep learning on implicit neural datasets. *arXiv preprint arXiv:2206.01178*, 2022.

Wang, S., Suo, S., Ma, W., Pokrovsky, A., and Urtasun, R. Deep parametric continuous convolutional neural networks. *CoRR*, abs/2101.06742, 2021.

Wu, W., Qi, Z., and Fuxin, L. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pp. 9621–9630, 2019.

Wu, Z., Jin, Y., and Yi, K. M. Neural fourier filter bank. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14153–14163, June 2023.

Xie, S., Zhu, H., Liu, Z., Zhang, Q., Zhou, Y., Cao, X., and Ma, Z. Diner: Disorder-invariant implicit neural representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6143–6152, 2023.

Xu, D., Wang, P., Jiang, Y., Fan, Z., and Wang, Z. Signal processing for implicit neural representations. In *Advances in Neural Information Processing Systems*, 2022.

Yu, A., Li, R., Tancik, M., Li, H., Ng, R., and Kanazawa, A. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5752–5761, 2021a.

Yu, A., Ye, V., Tancik, M., and Kanazawa, A. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4578–4587, 2021b.

Yu, H.-X., Guibas, L., and Wu, J. Unsupervised discovery of object radiance fields. In *International Conference on Learning Representations*.

Yuan, Y.-J., Sun, Y.-T., Lai, Y.-K., Ma, Y., Jia, R., and Gao, L. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18353–18364, 2022.

Zhao, C., Dewey, B. E., Pham, D. L., Calabresi, P. A., Reich, D. S., and Prince, J. L. Smore: a self-supervised anti-aliasing and super-resolution algorithm for mri using deep learning. *IEEE transactions on medical imaging*, 40(3): 805–817, 2020.

Zhou, P., Xie, L., Ni, B., and Tian, Q. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv preprint arXiv:2110.09788*, 2021.

Appendix

We present additional experimental details and empirical results for various experiments presented in the main paper and some ablation studies, pertaining to the use of noise as a channel for positional encoding. We start with a short discussion of the Calderon-Zygmund operator.

A. Calderon-Zygmund (CZ) Operator

Calderon-Zygmund (CZ) operators (Journé, 2006) are singular integrals with the special property that the kernel of the operator, $a(t, s)$ is smooth away from the diagonal and satisfies:

$$\begin{aligned} |a(t,s)| &\leq \frac{1}{|t-s|} \\ |\partial_t^M a(t,s)| + |\partial_s^M a(t,s)| &\leq \frac{C_0}{|t-s|^{1+M}} \end{aligned} \tag{16}$$

where $C_0 > 0$ is a constant and $M \geq 1$ is an integer which corresponds to the M-th partial derivative.

B. Multi-resolution training for 2D images

For each image we trained the O-INR model on a set of lower resolution images. For example: the cameraman image was originally of size 256×256, and hence O-INR was trained using images in the range 120×120 to 224×224. Similarly, for the human face image, O-INR was trained on images with size in the range 256×256 to 360×360 and final performance was evaluated on the original image of size 512×512. Since, baselines such as SIREN (Sitzmann et al., 2020), WIRE (Saragadam et al., 2023) and MFN (Fathony et al., 2021) can only be trained on an image of a specific resolution, we first trained them on the lowest, highest and average resolution of image used for training O-INR, however, as expected the performance of baseline models was the best when they were trained using the highest resolution of image shown to O-INR. Hence, we report PSNR for all baseline methods trained on the highest resolution of image used to train O-INR. The performance is measured on the original (higher) resolution image in all cases.

For training O-INR, we used a learning rate of 0.0005 for 1000 epochs and the number of sinusoidal frequencies used for each dimension was 20, 10 coming from $\sin$ and 10 coming from $\cos$. The number of parameters for O-INR model to achieve comparable performance was $\approx$ 100k whereas baseline methods required $\approx$ 130k parameters. Additional results are presented in Fig. 11

C. 2D Image Representation

For comparing representation capability of 2D images, O-INR and baseline methods were trained on a single image (fixed resolution). We report the representation capability in terms of PSNR. All models had ≈ 172k trainable parameters and trained until convergence with learning rate in the order of 0.001. Here, the number of sinusoidal frequencies used for each dimension was 20, 10 coming from $\sin$ and 10 from $\cos$. In Fig. 12, we present additional results for 2D image representation.

Ground truth | O-INR (26.05) | SIREN (28.56) | WIRE (26.44) | MFN (29.36)

Figure 11. Performance comparisons of O-INR in multi-resolution training setting. The ground-truth together with images from O-INR and other baselines (L to R), with the PSNR value in dB. O-INR achieves comparable performance

Figure 12. Performance comparison of O-INR for 2D image representation. Each row displays the ground-truth together with images from O-INR and other baselines (L to R), with the PSNR value in dB.

D. 3D Volume Representation

For 3D volume representation both O-INR and baseline models were trained with $\approx$ 1M parameters. We used a learning rate of 0.001. In this case, the number of sinusoidal frequencies used for each dimension was 16, 8 coming from $\sin$ and 8 from $\cos$. Additional results in Fig. 13 show that in terms of IoU, O-INR performs competitive with alternatives.

E. 2D Image denoising

In recovering an image from a noisy version, we trained O-INR and all baseline models on noisy variants obtained by following the noise addition procedure in (Saragadam et al., 2023). All models had roughly 130k trainable parameters. We used a learning rate of 0.003 for O-INR. Here, the number of sinusoidal frequencies used for each dimension was 20, 10 coming from $\sin$ and 10 from $\cos$. In Fig. 14 we present additional results.

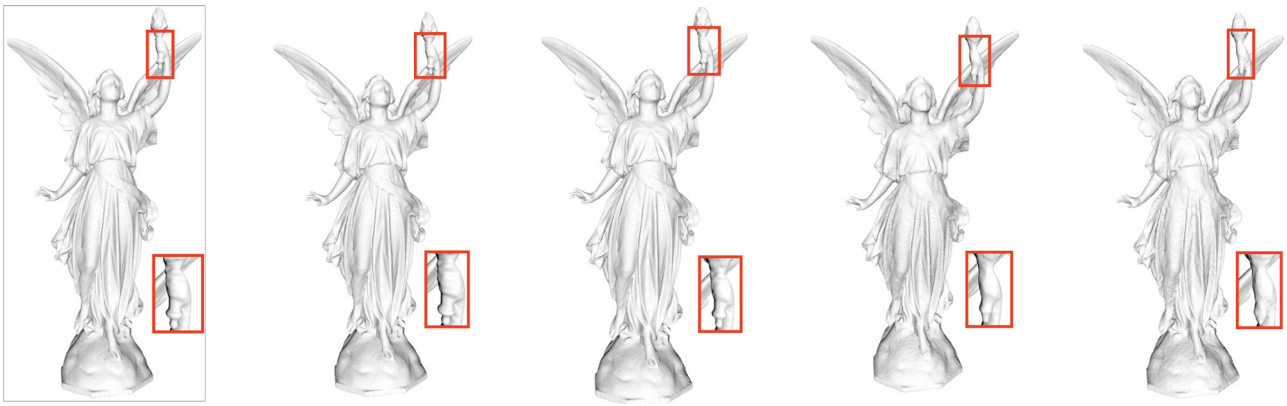


Figure 13. We report IoU achieved for each method after training converged. O-INR achieves best performance among all baselines. Zoomed in parts in each case show the that minute details are captured better by O-INR.

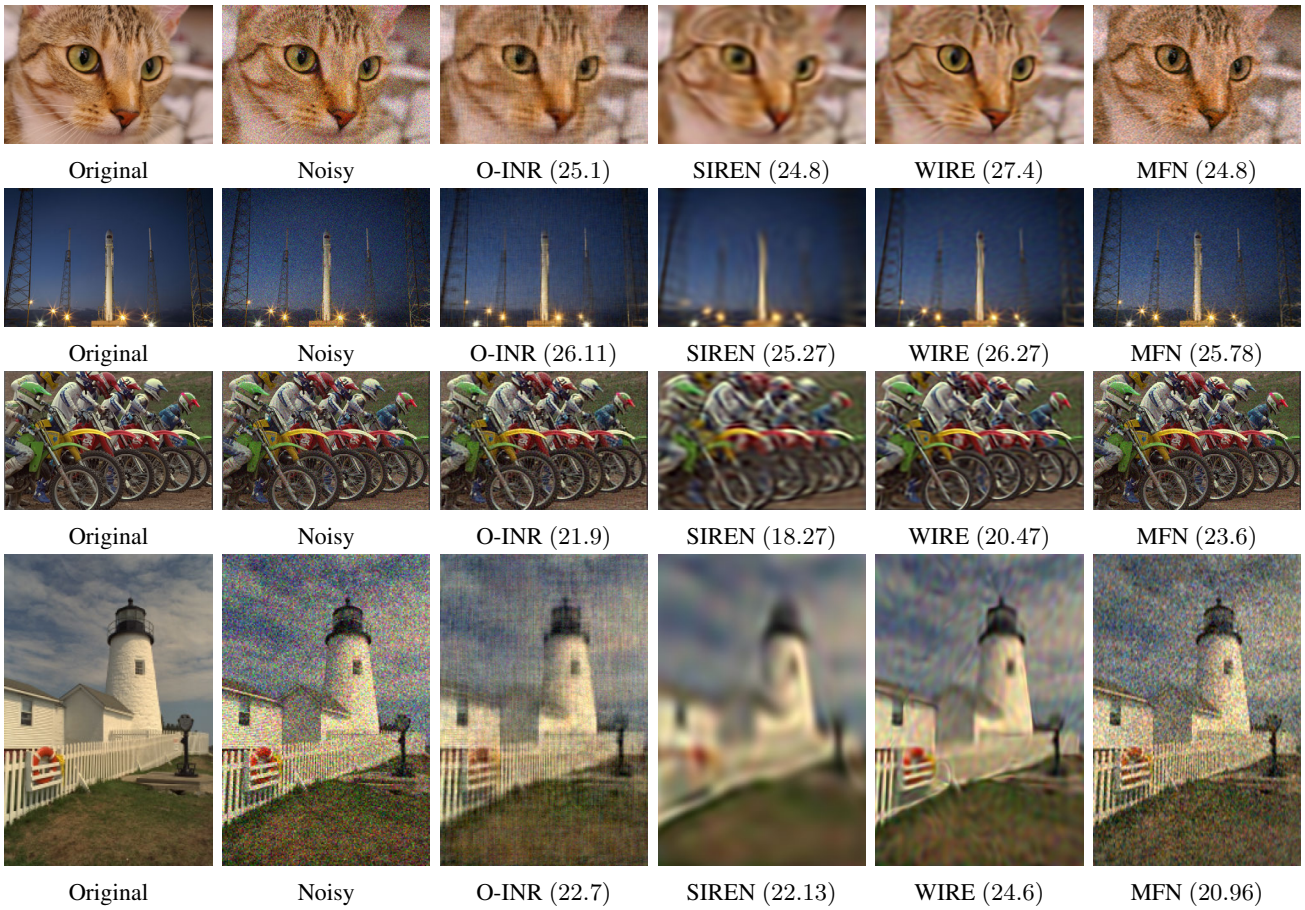


Figure 14. Performance comparisons of O-INR for representing noisy images. For each method, we note the PSNR it achieves on the image in dB. O-INR and other baselines perform similarly.

F. Effectiveness of Continuous Convolution in O-INR

We present results for 2D image representation and 2D image denoising where continuous convolution was used in O-INR. As described in the main paper, continuous convolutions are not strictly necessary for these tasks. Here, we demonstrate that performance of O-INR is not impacted by this choice as the PSNR is comparable whether one uses continuous or discrete convolutions. For example, in Fig. 15, the coffee-mug image attains a PSNR of 31.18 with continuous convolution O-INR whereas with discrete convolution O-INR, PSNR on the same image is 32.04 as reported in Fig. 3.

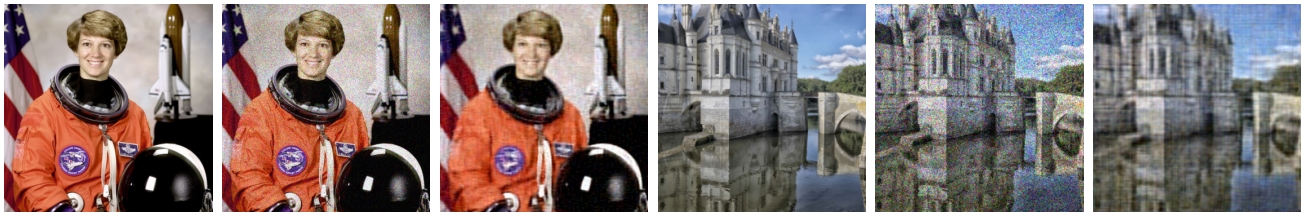
Similarly, in the case of representing noisy images, as can be seen in Fig. 16, for the image of an astronaut, continuous convolution based O-INR achieves PSNR of 23.5. On the other hand, as shown in the main paper, with discrete convolutions, we can achieve a PSNR of 24.48 on the same image. Hence, it is clear that the performance of O-INR appears to be not heavily dependent on the choice of continuous or discrete convolution and one can decide based on the representation task at hand.

G. Noise as Positional Encoding for O-INR

As mentioned in **Remark 2** of the main paper, O-INR is capable of simply using noise as a proxy for the high frequency positional encoding term due to the use of convolutional layers. But this is a poor choice for standard INRs with MLP layers due to the lack of location bias. Here, we present empirical evidence for the effectiveness of using noise sampled from standard normal distribution as a means of providing high frequency component for the positional encoding. We see results in Fig. 17.

Ground truth | **O-INR** (31.18) | Ground truth | **O-INR** (32.46)

Figure 15. Performance of O-INR for 2D image representation using continuous convolution. The ground-truth together with images from O-INR with the PSNR value in dB. Performance is comparable to O-INR using discrete convolution



Original | Noisy | **O-INR** (23.5) | Original | Noisy | **O-INR** (24.14)

Figure 16. Performance of O-INR with continuous convolution for representing noisy images. We note the PSNR it achieves on the image in dB. Performance is comparable to O-INR using discrete convolution

Ground truth | **O-INR** (26.68) | Ground truth | **O-INR** (29.77)

Figure 17. Performance of O-INR for 2D image representation using standard normal noise for high frequency positional encoding. The ground-truth together with images from O-INR with the PSNR value in dB.

H. O-INR for sequence data

We used a learning rate of 0.001 along with 20 positional encodings for each spatial dimension, 10 for $\sin$ and 10 for $\cos$ for training an O-INR model. When trained on the first 16 frames of a cat video (Sitzmann et al., 2020), O-INR can achieve an average PSNR of 35.68 (or MSE of 0.00027). Additionally, we also trained O-INR on frames obtained from sub-sampling a video, demonstrating a capability to recover the original sequence despite only seeing a sparse version. In the supplementary material, please refer to videos in the "Result" folder for original and videos recovered from trained O-INR, corresponding to both experimental settings: "consecutive" and "sparse". The "consecutive" sub-folder contains results for the scenario where O-INR was trained on consecutive frames. The folder names therein indicate the video, either of a cat or a road scene and the number denotes the value of 'n' for the first 'n' consecutive frames in the video. The "sparse" sub-folder has results for the scenario, where O-INR was trained on a sparse subset of sub-sampled frames from the video. Folder names therein indicate the dataset, the number of frames used to train the model and the final number of frames present in the video recovered via O-INR.

I. Applications to Brain Imaging

For the 3D brain imaging data, we chose approximately 140 subjects each from the cognitively normal (CN) and diseased (AD) groups and trained O-INR on the T1 MRI scans as mentioned in §6. For performing the group difference analysis using O-INR we use only 34% of available slices, by sampling every third slice along the Coronal direction for training

Figure 18. Top/Bottom: Rows show frames from a video from the TGIF dataset: original and the ones from O-INR trained on sparsely sampled frames of a long video sequence. O-INR represents the scenes in the sequence well.

Figure 19. Top to Bottom: Rows represent frames from cat video (Sitzmann et al., 2020) original and ones obtained from O-INR representation. In this case the model was trained on consecutive frames of the video.

Figure 20. Top/Bottom: Rows show frames from a bike video: original and the ones from O-INR trained on sparsely sampled frames of a long video sequence. O-INR represents the scenes in the sequence well.

the O-INRs. We used 20 positional encodings for each spatial dimension: 10 for $\sin$ and 10 for $\cos$. An initial learning rate of 0.001 was used alongside a Cosine Annealing scheduler with a minimum learning rate of 5×10^{-4} and maximum steps of 10000. The whole brain image was generated at the original resolution using the trained model. The models trained achieved an MSE of 2.67×10^{-4} at the original resolution, indicating that O-INR is able to represent the 3D volume well.

In order to perform statistical analysis, we used the Statistical non-Parametric Mapping (SnPM) toolbox. We performed statistical group difference (voxel-wise t-test) on the real data (CN versus AD) with 10000 permutations. Then, the same analysis process was repeated on O-INR derived data (CN versus AD). Note that the O-INR's were trained only a fraction of the original resolution. We find that voxels reported to be significant (uncorrected p-values) on the real (non-sampled) data agree fully with the analysis results on data based on O-INR slice imputation. Sizable clusters agree although the spatial

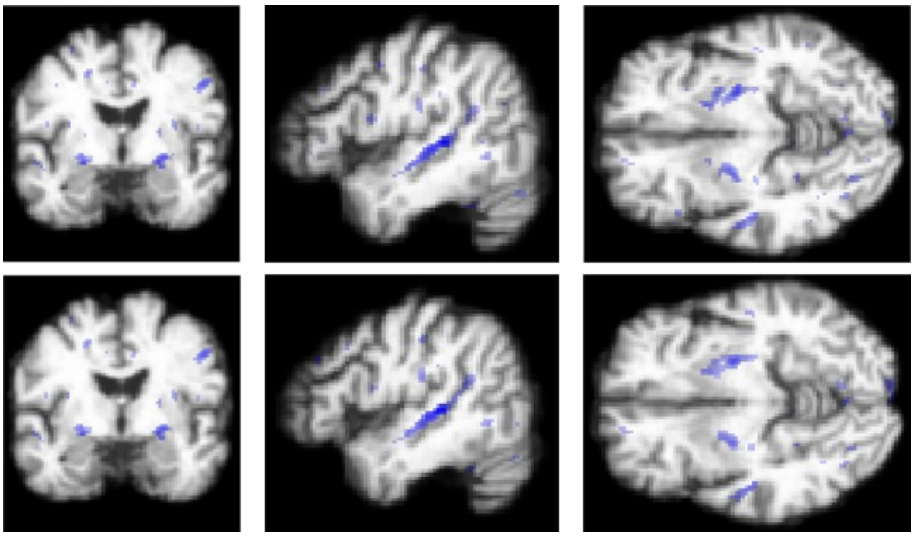


Figure 21. Top and Bottom rows: Overlay of filtered statistic image from group difference analysis of original (full resolution) and images generated via O-INR trained on sparsely sampled (30% slices) AD and CN images respectively. The above images indicate that O-INR is successful in preserving the group difference in 3D brain imaging data.

extent is reduced (higher Type 2 error). This is evident in both the overlay diagram in Fig. 21 as well as the T-statistic and uncorrected p-values in Fig. 22.

T	p_{uncorr}	x,y,z mm		
6.00	0.0001	**21**	**48**	**37**
2.97	0.0017	22	42	44
1.53	0.0650	20	61	28
4.23	0.0001	**57**	**59**	**32**
3.92	0.0001	55	69	35
3.02	0.0016	60	68	42
4.06	0.0002	**72**	**48**	**36**
3.92	0.0002	**65**	**45**	**19**
2.62	0.0055	55	45	15
1.96	0.0250	63	37	17
3.88	0.0002	**61**	**52**	**42**
3.82	0.0002	**33**	**66**	**40**
2.78	0.0033	35	71	34
3.68	0.0002	**33**	**61**	**33**
3.48	0.0004	**67**	**70**	**58**
2.11	0.0185	60	72	54
3.48	0.0006	**40**	**20**	**24**
3.38	0.0006	**71**	**60**	**56**

T	p_{uncorr}	x,y,z mm		
6.00	0.0001	**21**	**48**	**37**
3.29	0.0002	22	41	44
2.76	0.0027	20	57	31
4.06	0.0002	**72**	**48**	**36**
1.63	0.0542	71	55	32
3.97	0.0001	**55**	**69**	**35**
3.64	0.0002	57	59	32
3.09	0.0012	59	74	41
3.92	0.0001	**65**	**45**	**19**
2.78	0.0030	65	37	17
2.55	0.0057	55	45	15
3.88	0.0001	**33**	**66**	**40**
1.80	0.0388	37	67	33
3.69	0.0002	**29**	**92**	**49**
2.89	0.0026	33	85	45
2.24	0.0126	35	97	46
3.42	0.0003	**61**	**51**	**43**
1.87	0.0315	62	58	37

Figure 22. Statistical analysis results from SnPM showing T-statistics, uncorrected p-values and the cluster center location of sizable clusters. Table on the left summarizes the analysis results on the real data whereas the table on the right summarizes the analysis on the data from O-INR slice imputation. Nearly all sizable clusters (in bold) on the left have a corresponding cluster on the right (the rank may be slightly up or down) indicating strong agreement between the results.

J. Weight space interpolations of O-INR

In addition to performing interpolation between two different O-INRs, we also manipulated convolutional layers in individual O-INRs. We find that manipulating individual convolutional layers by interpolating its weights while holding others fixed yields structurally coherent changes to the image as shown in Fig. 23. In particular, early layers in the O-INR capture large-scale features of the image (Conv1 perturbs the shape of the head, Conv3 perturbs the eyes and nose) while later layers reflect local properties such as color and texture (Conv4 and Conv5). In Fig. 24 we present additional results for weight interpolation between O-INRs trained on images from CelebA (Liu et al., 2015).

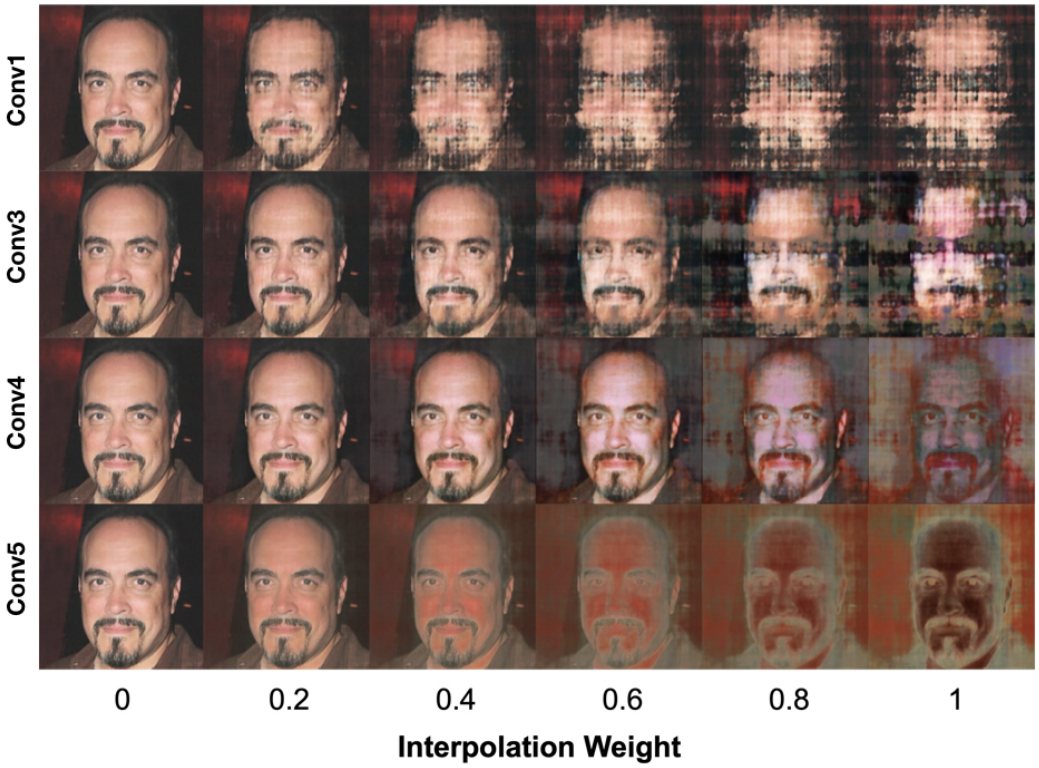


Figure 23. Images produced by interpolating the weights of a single convolutional layer between two O-INRs fit on different CelebA images. Other weights are held fixed.

Figure 24. Randomly selected examples of images produced by interpolating the weights of O-INRs fit on different images from the CelebA dataset.