Ben-Gurion University of the Negev

Towards Croppable Implicit Neural Representations

Maor Ashkenazi, Eran Treister

Department of Computer Science, Ben-Gurion University of the Negev

Introduction

- Implicit Neural Representations (INRs) have gained popularity due to their ability to encode natural signals in neural networks weights
  - By modeling the signal as a prediction task from some coordinate system to the signal values
- A popular choice for the network is a fully connected network (MLP)

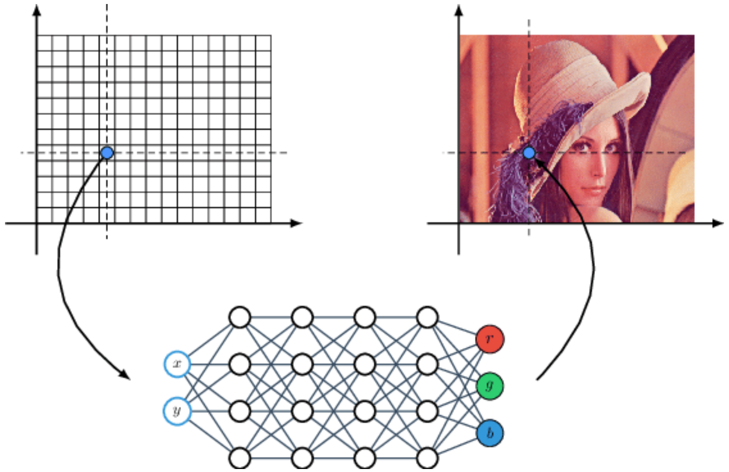
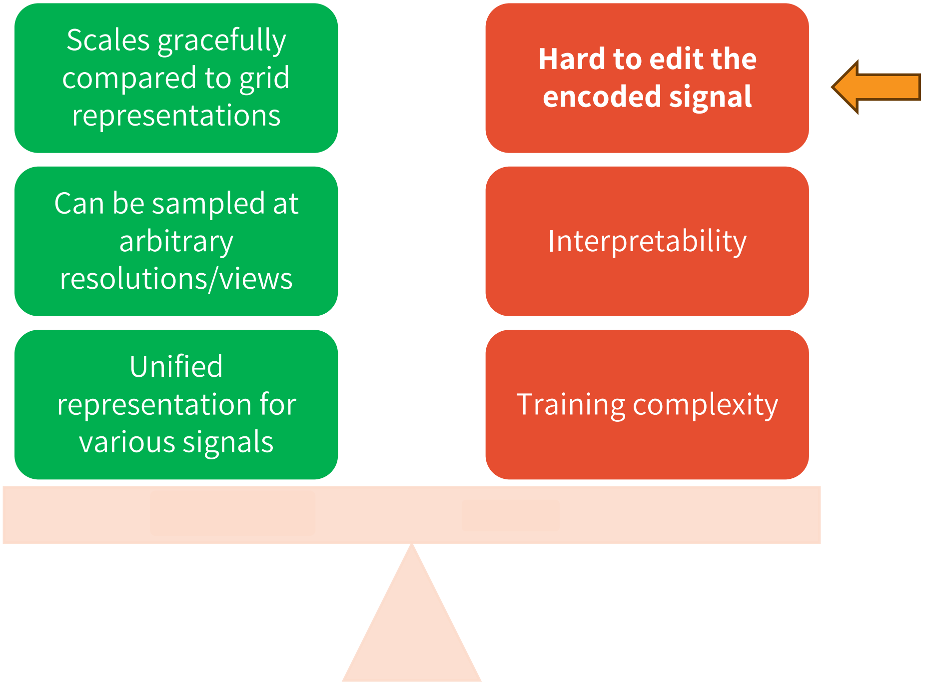


Image from “Adversarial Generation of Continuous Images“ (Skorokhodov et al., 2020)

Motivation

- INRs have many advantages over discrete representations, and allow for interesting applications
- However, their black-box nature presents disadvantages

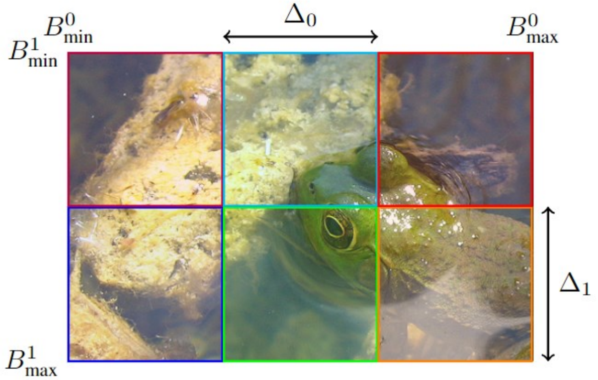


Goal

- We focus on the basic signal editing operation of **cropping**
- We wish to **remove parts of the encoded signal, with a proportional decrease of INR weights**
  - Without retraining/finetuning
  - Without compromising on encoding quality

Partitioning the Signal

- We begin by dividing the input signal space



- Each dimension is split into C_i equally sized partitions, resulting in $\prod_{i=1}^{n} C_i$ partitions
- Separate weights will be dedicated for each partition
- The granularity of partitioning will determine the detail of which we crop the INR

A Straightforward Approach

- Assigning separate weights to different partitions by training a compact INR-per-partition
- Was done by KiloNeRF and related methods
    - Allowed for significant speed benefit, both in terms of training and inference

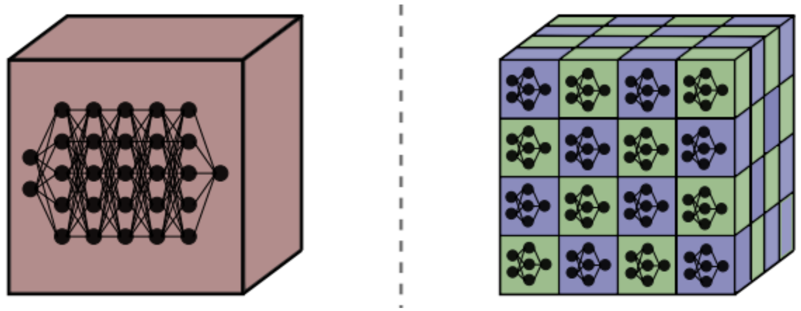


Image from: KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs (Reiser et al., 2021)

A Straightforward Approach

- Assigning separate weights by training a compact INR-per-partition
- Was done by KiloNeRF and related methods
  - Allowed for significant speed benefit, both in terms of training and inference

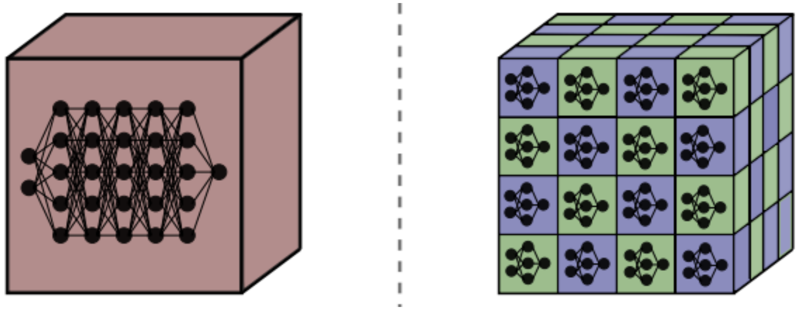
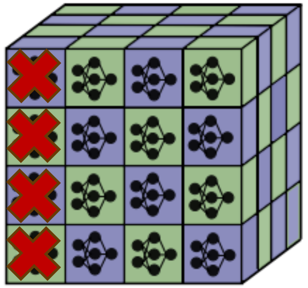


Image from: KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs (Reiser et al., 2021)

- Cropping can be achieved by removing specific INRs



Downsides to the Straightforward Approach

- Compact local INRs **lack global context**
- Can result in artifacts, especially noticeable along edges

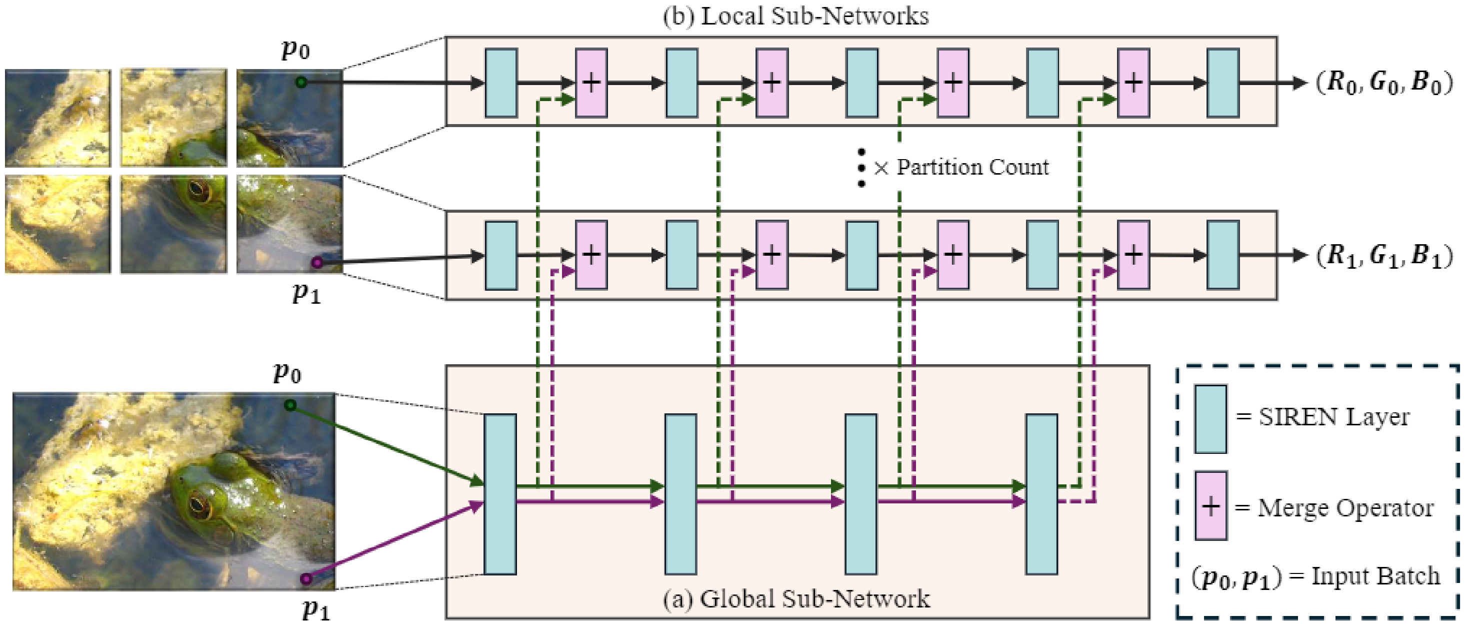
- In KiloNeRF, solved using knowledge distillation
  - By sampling novel viewpoints
  - **Requires training a full INR**

Image from: KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs (Reiser et al., 2021)

Our Approach

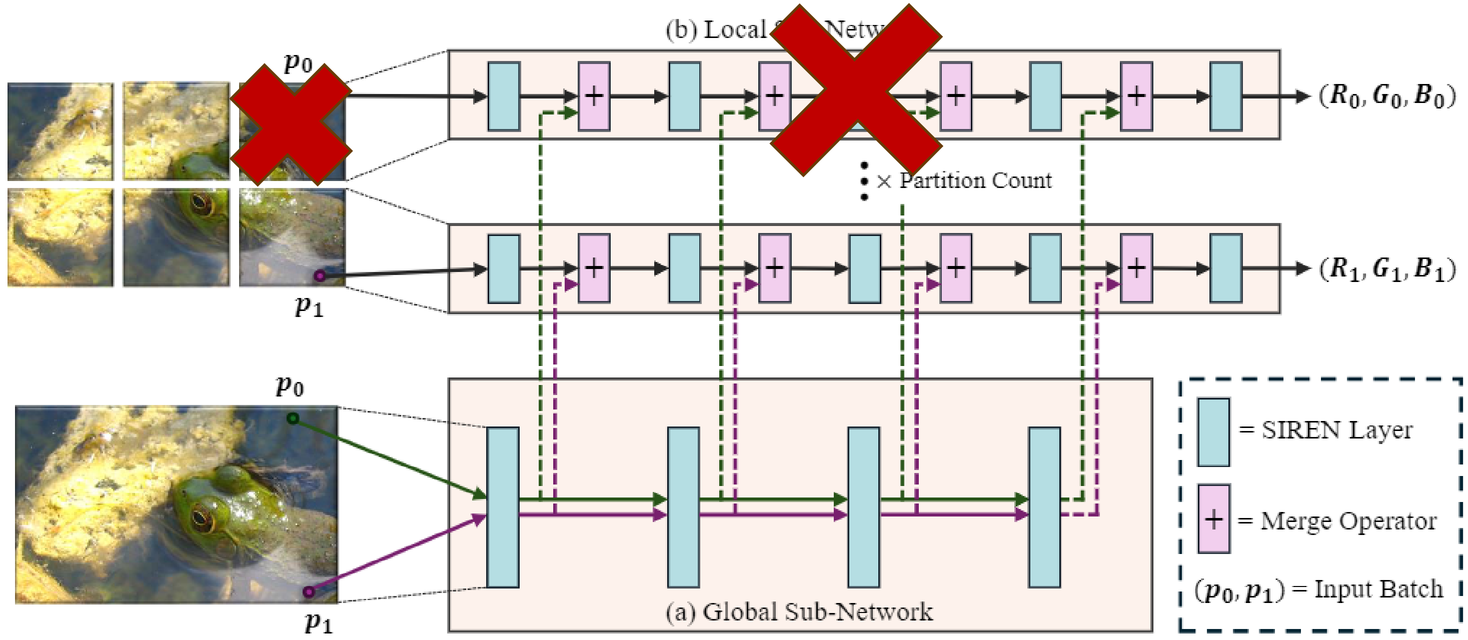
- A novel INR architecture, termed **Local-Global INRs**
- Based on combining both **local and global context learning**
  - A local sub-network for each partition
  - A global sub-network for the entire signal, used to augment the local features with global information
- The Local-Global architecture can be applied to most baseline MLP-based INRs
  - We focus on **SIREN** for its popularity
  - Additionally, we explore **INCODE**, a SOTA INR

Local-Global Architecture



$$\text{Merge}(\mathbf{L}, \mathbf{G}) = \sigma(\text{concat}([\mathbf{L}, \mathbf{G}]) \cdot \mathbf{W} + \text{b})$$

Cropping a Local-Global INR

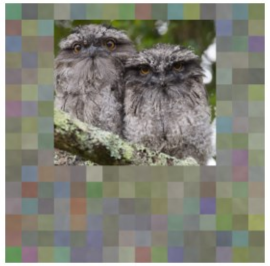


- Global sub-network parameters (+ merge operator) should encompass a small part of the overall architecture
  - 5-15% is sufficient to achieve good quality reconstruction

Cropping a Local-Global INR

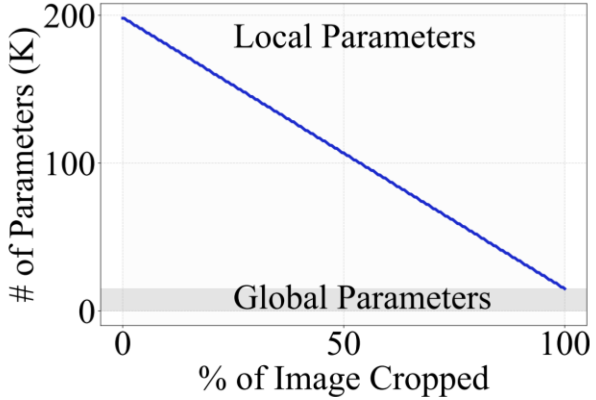
Images

199K Parameters

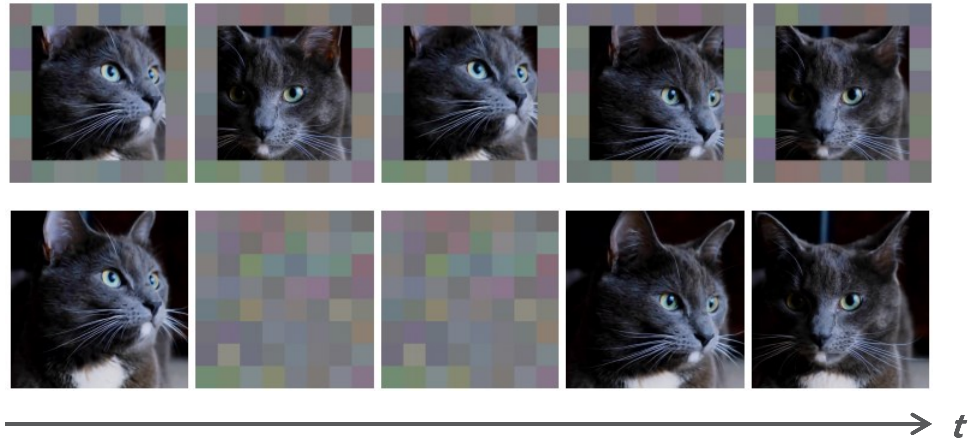


80K Parameters

47.7K Parameters



Videos



Faster and Better Convergence

- Local-Global INRs easily surpass an INR-per-Partition approach in terms of encoding quality
- **And even improve upon the baseline INR itself**
- Due to the local-subnetworks, **Local-Global INRs achieve faster training/inference**

Image Encoding

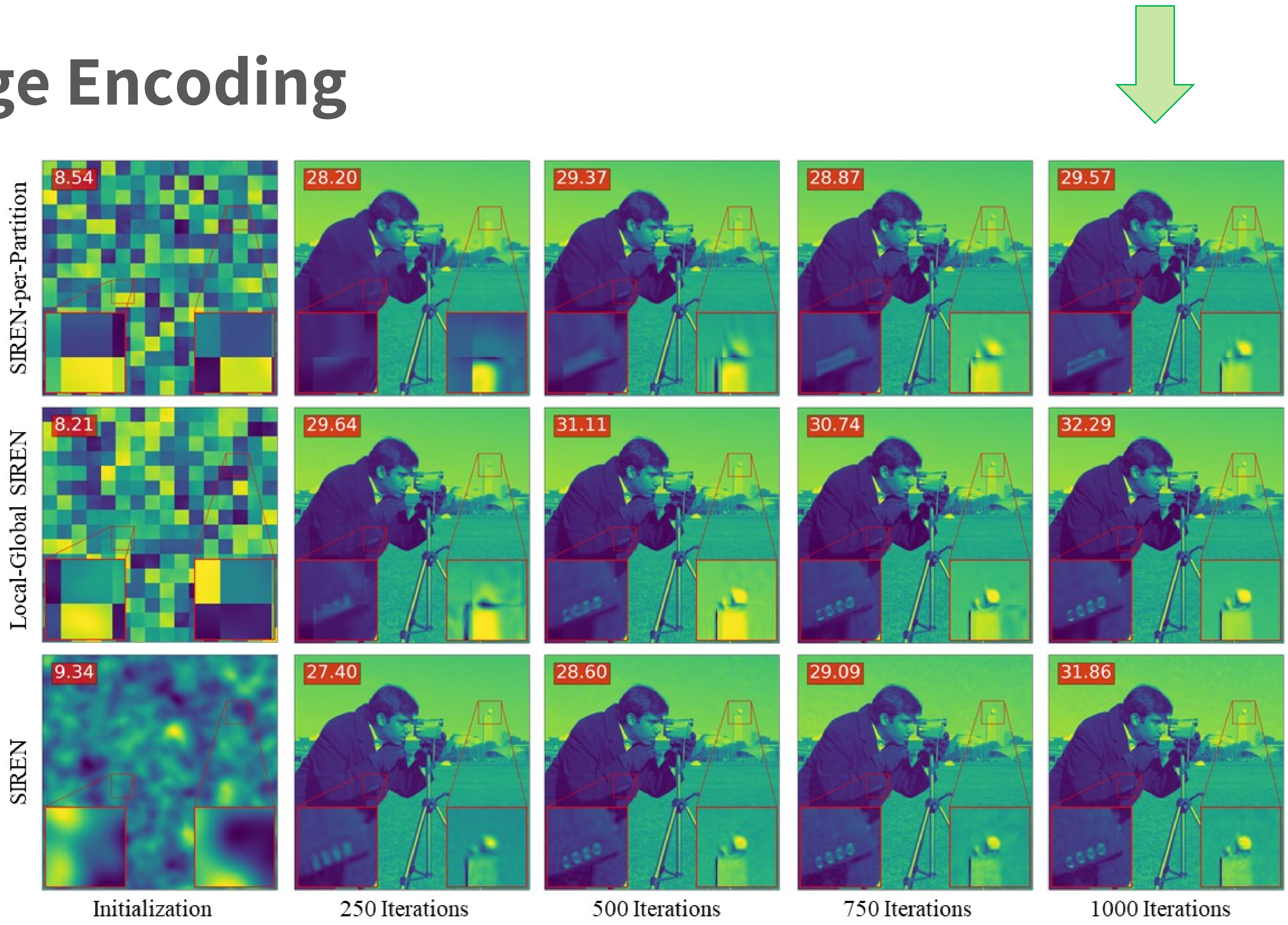


Image Encoding

Quantitative results on a DIV2K subset

Method	Partition Factors	SSIM ↑	PSNR (dB) ↑
SIREN-per-Partition	(16, 16)	0.957	31.73 ± 0.63
SIREN-per-Partition	Auto	0.955	31.90 ± 0.64
Local-Global SIREN	(16, 16)	0.968	33.94 ± 0.64
Local-Global SIREN	Auto	**0.971**	**34.13 ± 0.59**
SIREN	-	0.966	33.57 ± 0.65

Table 1. Mean encoding results on 25 DIV2K images using five random seeds per image. Automatic partitioning uses partition factors $11 \leq C_i \leq 16$ to ensure 32×32 pixel partitions.

Video Encoding

- A 12 second RGB cat video (512×512, 300 frames)
- Using partition factors $C_0 = 5, C_1 = 8, C_2 = 8$
- Sub-sampling a part of the pixels in each training iteration
  - Our method requires less memory, allowing us to increase the number of sampled pixels

Table 2: Mean video encoding results, using 10 random seeds. (*) next to method stands for sampling $2 \cdot 10^{-2}\%$ of pixels in each iteration. SPP, LGS stand for SIREN-per-Parition and Local-Global SIREN, respectively.

Method	SSIM ↑	PSNR (dB) ↑
SPP	0.826	29.58 ± 0.02
LGS (ours)	**0.854**	**30.28 ± 0.05**
SIREN	0.815	29.71 ± 0.09
SPP (*)	0.854	30.83 ± 0.01
LGS (*) (ours)	**0.888**	**31.91 ± 0.02**

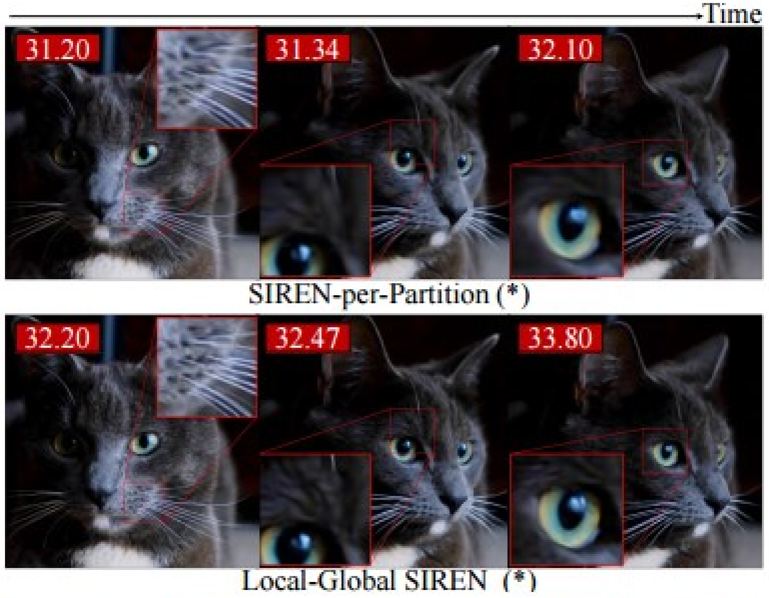


Figure 6: Three frames of an encoded video. PSNR is at the top left of each frame.

Audio Encoding

- Using partition factor $C_0 = 32$

Table 11: Audio encoding results after 1k training iterations. Averaged on 10 seeds.

Audio Clip	Method	C_0	MSE ($\cdot 10^{-5}$) $\downarrow$	PSNR (dB) $\uparrow$
Bach (7s)	SIREN-per-Partition	32	12	39.26 ± 0.30
	Local-Global SIREN (ours)	32	**3**	**45.18 ± 0.99**
	SIREN	-	10	39.94 ± 0.75
Counting (12s)	SIREN-per-Partition	32	75	31.24 ± 0.19
	Local-Global SIREN (ours)	32	**48**	**33.18 ± 0.34**
	SIREN	-	62	32.07 ± 0.32

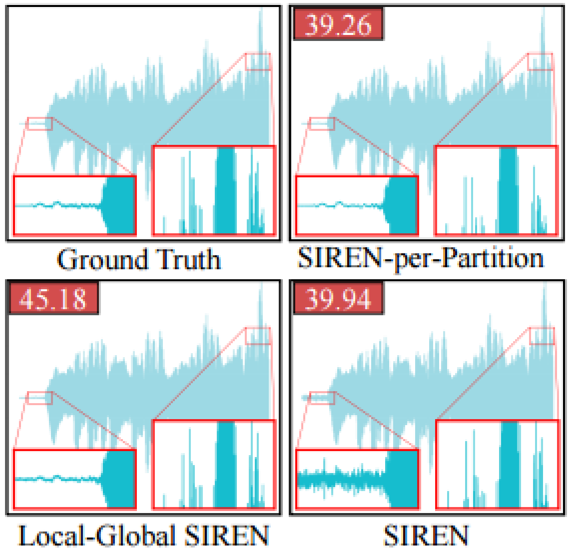
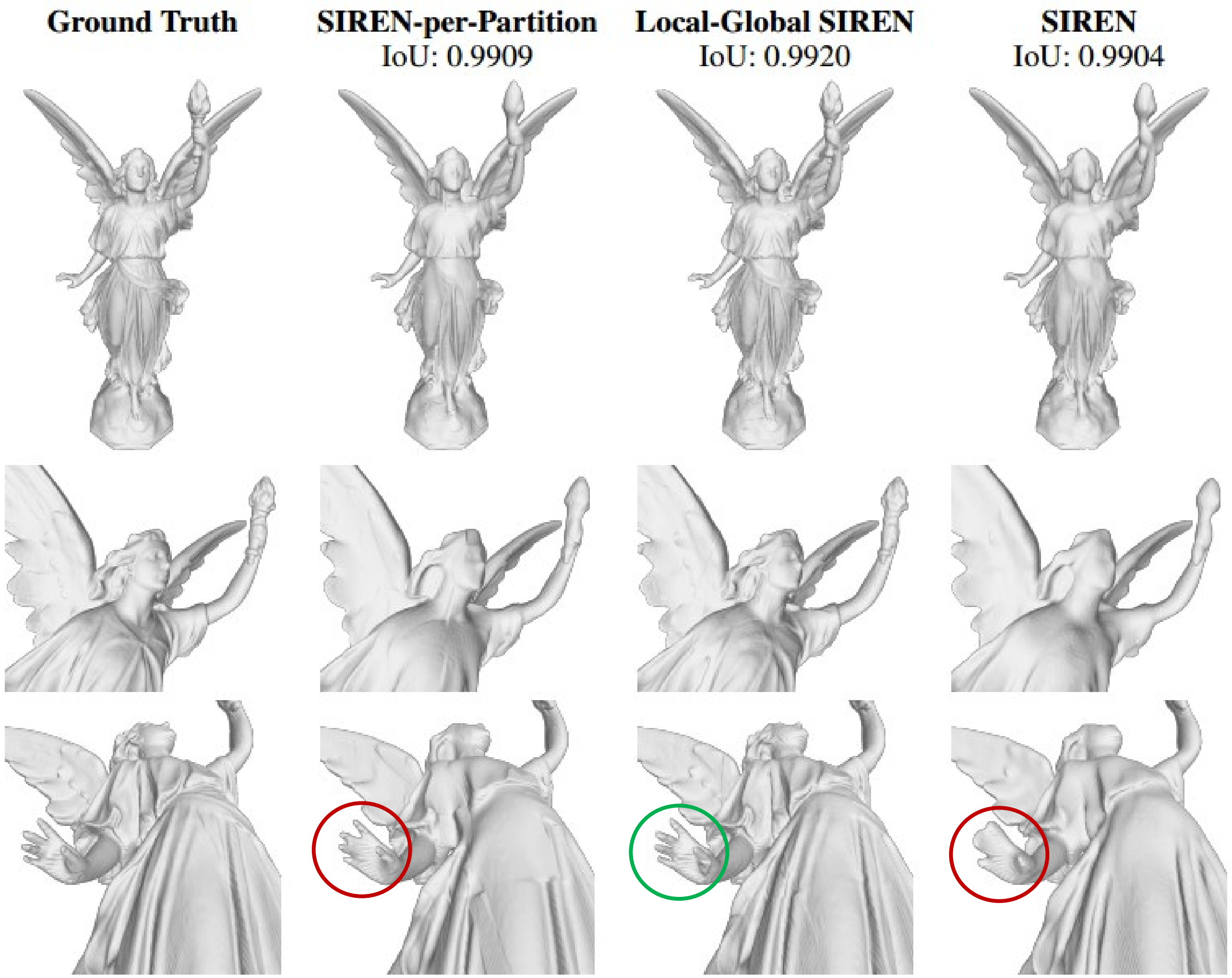


Figure 5: Encoded *Bach* audio clips. Mean PSNR values using 10 random seeds are on the top left of each figure.

3D Shape Encoding

- A voxel grid of size $512 \times 512 \times 512$
- Using partition factors $C_0 = 5, C_1 = 8, C_2 = 8$



Partitioning Effects

Larger Partition Factors → Smaller Partitions → More Local Sub-networks → Less Neuron Interconnectivity

Table 3: Effect of partitioning on latency and accuracy. Results averaged on 10 seeds.

Signal	Model	Partition Factors	MSE ↓ ($\cdot 10^{-4}$)	SSIM ↑	PSNR (dB) ↑	Train ↓ Time (s)
Image 512 × 512	Local-Global SIREN (ours)	(2, 2)	**11.2**	**0.946**	**32.59 ± 0.52**	74
		(4, 4)	12.0	**0.946**	32.10 ± 0.47	40
		(8, 8)	13.5	0.942	32.29 ± 0.42	26
		(16, 16)	15.3	0.934	32.00 ± 0.39	22
		(32, 32)	19.0	0.917	31.51 ± 0.28	**15**
	SIREN	-	18.4	0.914	31.17 ± 0.68	34
Video 300 × 512 × 512	Local-Global SIREN (ours)	(5, 4, 4)	**32.8**	**0.862**	**30.95 ± 0.07**	386
		(5, 8, 8)	34.7	0.854	30.28 ± 0.05	284
		(5, 16, 16)	41.8	0.834	29.52 ± 0.03	**244**
	SIREN	-	43.4	0.815	29.71 ± 0.09	2354

Local-Global INCODE

- INCODE is a SOTA INR, which demonstrated improved performance on various downstream tasks
- We apply our method to INCODE, resulting in a Local-Global INCODE
- We recreate three downstream tasks for the original paper, and show how **Local-Global INCODE improved downstream performance**

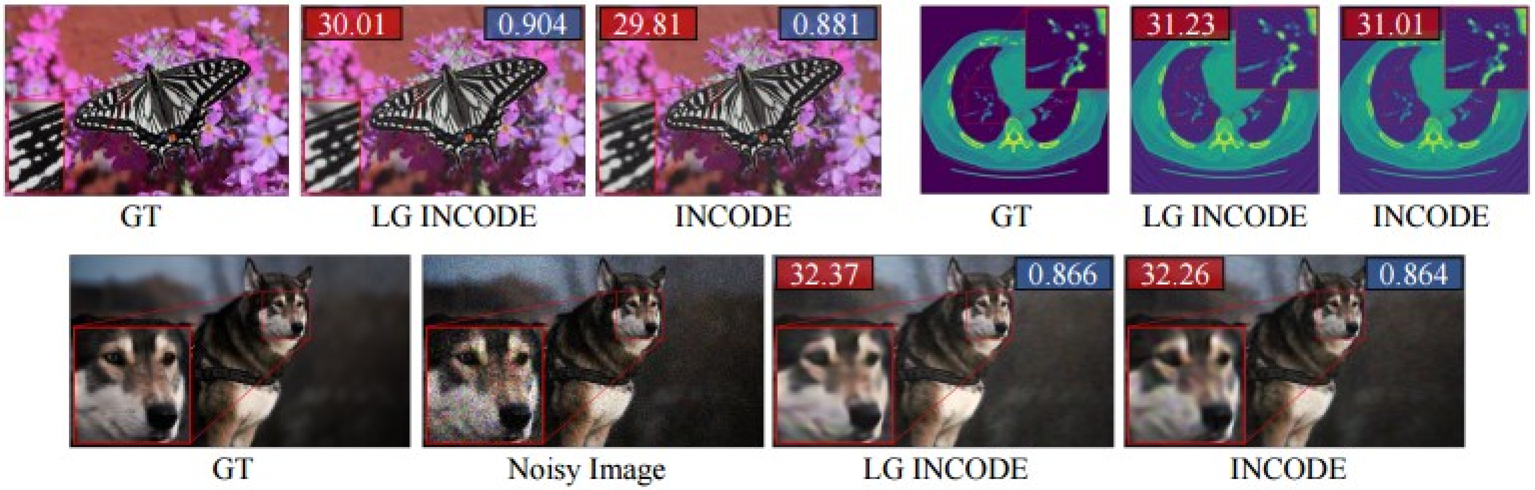


Figure 8: Local-Global (LG) INCODE applied to downstream tasks. Top-left: 4x image super-resolution, top-right: CT reconstruction, bottom: image denoising. Mean PSNR and SSIM values across 10 seeds are displayed in the top-left and top-right corners of each frame, respectively.

Conclusions

Paper

Code

- **Local-Global INRs seamlessly support cropping with a proportional weight decrease**
  - No retraining needed
  - Eliminating the need for a pretraining step as in other methods
- Superior encoding quality and training speeds
  - Adjustable partitioning allows for a balance between latency and accuracy

- More experiments in the paper, including extending a previously encoded signal by adding novel local sub-networks

Thank You

Ben-Gurion University
of the Negev