Supplemental Document

Optics EXPRESS

Seeing through fibers: unsupervised image reconstruction in fiber bundle imaging systems: supplement

AMIR REZA VAZIFEH,[1] CONGLI WANG,[2] AMOGH JOSHI,[1] ILYA CHUGUNOV,[2] JIPENG SUN,[2] JIWOON YEOM,[2] JASON W. FLEISCHER,[1] JOSÉ S. PULIDO,[3] AND FELIX HEIDE[2,*]

[1] *Department of Electrical and Computer Engineering, Princeton University, Princeton, New Jersey 08544, USA*
[2] *Department of Computer Science, Princeton University, Princeton, New Jersey 08544, USA*
[3] *Wills Eye Hospital, Philadelphia, Pennsylvania 19107, USA*
[*] *fheide@princeton.edu*

Supplement DOI: https://doi.org/10.6084/m9.figshare.31323835

Parent Article DOI: https://doi.org/10.1364/OE.587675

Supplement 1: Seeing Through Fibers: Unsupervised Image Reconstruction in Fiber Bundle Imaging Systems

The Supplemental Document is organized as follows: Section 1 describes the simulation data generation and experimental data acquisition. Section 2 describes the MLP architectures and positional encoding used in our method, along with a hyperparameter analysis of positional encoding. Section 3 presents the performance of our method on additional simulated and experimental data. We also include an image processing pipeline for extracting fiber masks from the reconstructed videos. Section 4 provides further examples of the ablation study discussed in the paper. Section 5 provides details of the baseline methods used for comparison. Finally, Section 6 provides a description of the code to facilitate reproducibility of the results.

1. DATA CAPTURING

The overall data acquisition procedure is shown in Fig. S1. For the simulated data, a random Brownian motion was applied to a raw image to generate a sequence of 20 frames. Each frame was then multiplied by the fiber mask shown in Fig. S1 to simulate the imaging characteristics of the fiber bundle. For the 2D experimental scene, the same procedure was applied to produce 20 frames. The generated frames were displayed on a screen and subsequently captured using our imaging setup. For the 3D experimental scene, the object was manually moved relative to the imaging probe to acquire 20 frames from different viewpoints.

2. MLP ARCHITECTURE AND HYPERPARAMETERS

For all experiments in this paper, the coordinate-based MLP architectures used for homography estimation (g_ϕ) and scene reconstruction (f_θ) are illustrated in Fig. S2. The motion estimation network g_ϕ maps a time point t (corresponding to frame t in a burst) to its homography matrix $M_t \in \mathbb{R}^{3\times3}$, represented as an 8-dimensional vector, effectively defining a mapping $\mathbb{R} \to \mathbb{R}^8$. This network consists of three hidden layers with 256 neurons each, totaling 134,152 learnable parameters. After computing the homography and warping the coordinates, the warped coordinates are passed through a Fourier positional encoding [1] in Eq. (S1) to preserve fine details during reconstruction. Random Fourier features are sampled from a Gaussian distribution with standard deviation σ to construct the positional encoding as

$$\gamma(\hat{x},\hat{y}) = \Big[\hat{x},\hat{y},\sin\left(2\pi\,\mathbf{b}_1^\top \begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}\right),\ \cos\left(2\pi\,\mathbf{b}_1^\top \begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}\right),\ \ldots,\ \sin\left(2\pi\,\mathbf{b}_m^\top \begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}\right),\ \cos\left(2\pi\,\mathbf{b}_m^\top \begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}\right)\Big], \tag{S1}$$

where $\mathbf{b}_i \in \mathbb{R}^2$ is the i-th row of the random Gaussian matrix $B \in \mathbb{R}^{m\times2}$. In practice, the number of features m and the standard deviation σ are adjusted according to the complexity of the scene: smaller values capture smooth variations, while larger values encode finer details. For all experiments in this paper, we set $m = 256$. The standard deviation σ was fixed to 5 in most cases, but occasionally varied between 3 and 7. The scene reconstruction network f_θ maps encoded coordinates $\mathbb{R}^{514} \to \mathbb{R}^3$ to RGB values using four hidden layers of 256 neurons each, totaling 329,987 learnable parameters (total of 464,139 parameters when combined with g_ϕ). For better visualization, the predicted RGB values were multiplied by a constant factor (1.5 for all experimental data and 2 for all simulated data) to enhance brightness.

To study the effect of σ on reconstruction, we evaluated $\sigma \in \{1, 2, 5, 10, 50\}$ on five samples (Fig. S3). $\sigma = 50$ fails to reconstruct the scene, while $\sigma = 10$ exhibits a noticeable honeycomb pattern, and $\sigma = 1$ misses fine details. Overall, $\sigma = 5$ produces the most reliable reconstructions. We performed the same analysis on simulated data with ground truth, where $\sigma = 5$ achieves the highest PSNR and SSIM in most cases, with $\sigma = 2$ occasionally performing slightly better (Fig. S4, Table S1).

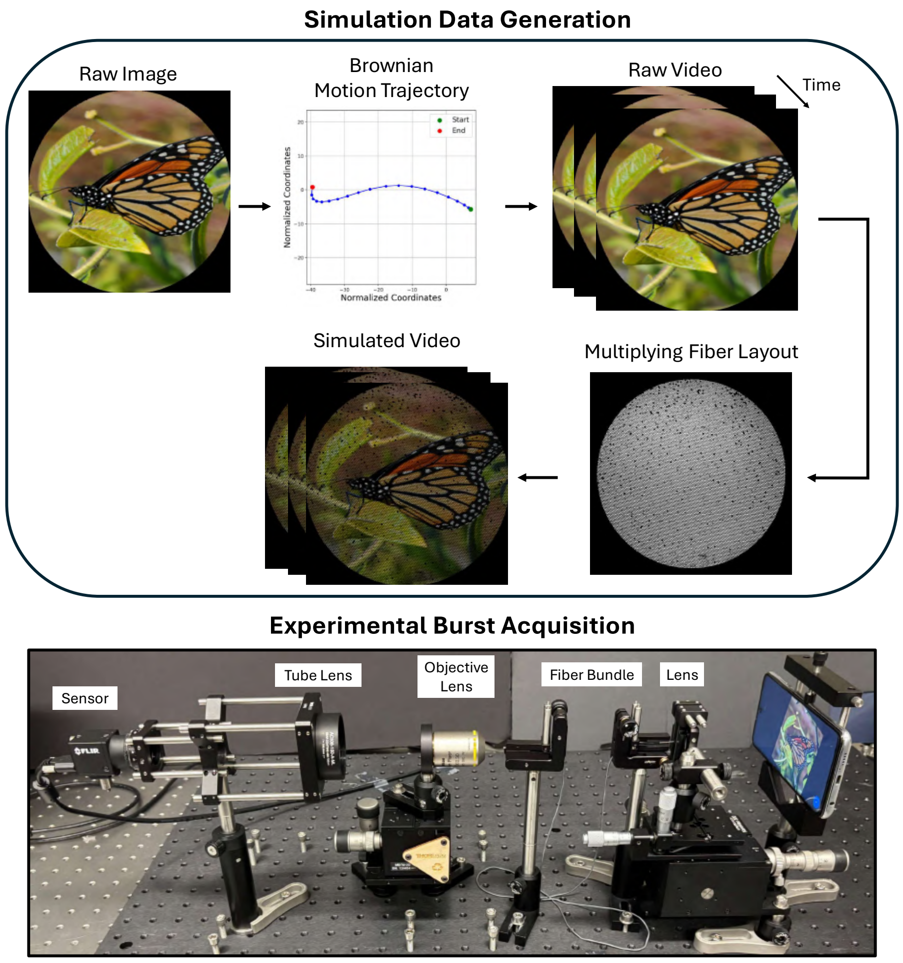

Fig. S1. Overview of the data capturing procedure. Top: Simulation data. Bottom: Experimental data.

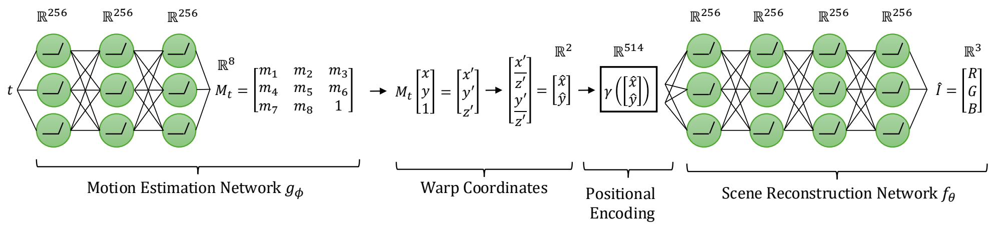
$\mathbb{R}^{256}$
$\mathbb{R}^{256}$
$\mathbb{R}^{256}$
t
$\mathbb{R}^{8}$
$M_t = \begin{bmatrix} m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 \\ m_7 & m_8 & 1 \end{bmatrix}$
$M_t \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \rightarrow \begin{bmatrix} x'/z' \\ y'/z' \end{bmatrix} = \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix}$
$\mathbb{R}^{2}$
$\mathbb{R}^{514}$
$\gamma\left(\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix}\right)$
$\mathbb{R}^{256}$
$\mathbb{R}^{256}$
$\mathbb{R}^{256}$
$\mathbb{R}^{256}$
$\mathbb{R}^{3}$
$\hat{I} = \begin{bmatrix} R \\ G \\ B \end{bmatrix}$
Motion Estimation Network g_ϕ
Warp Coordinates
Positional Encoding
Scene Reconstruction Network f_θ

Fig. S2. MLP architectures for motion estimation MLP g_ϕ and scene reconstruction MLP f_θ.

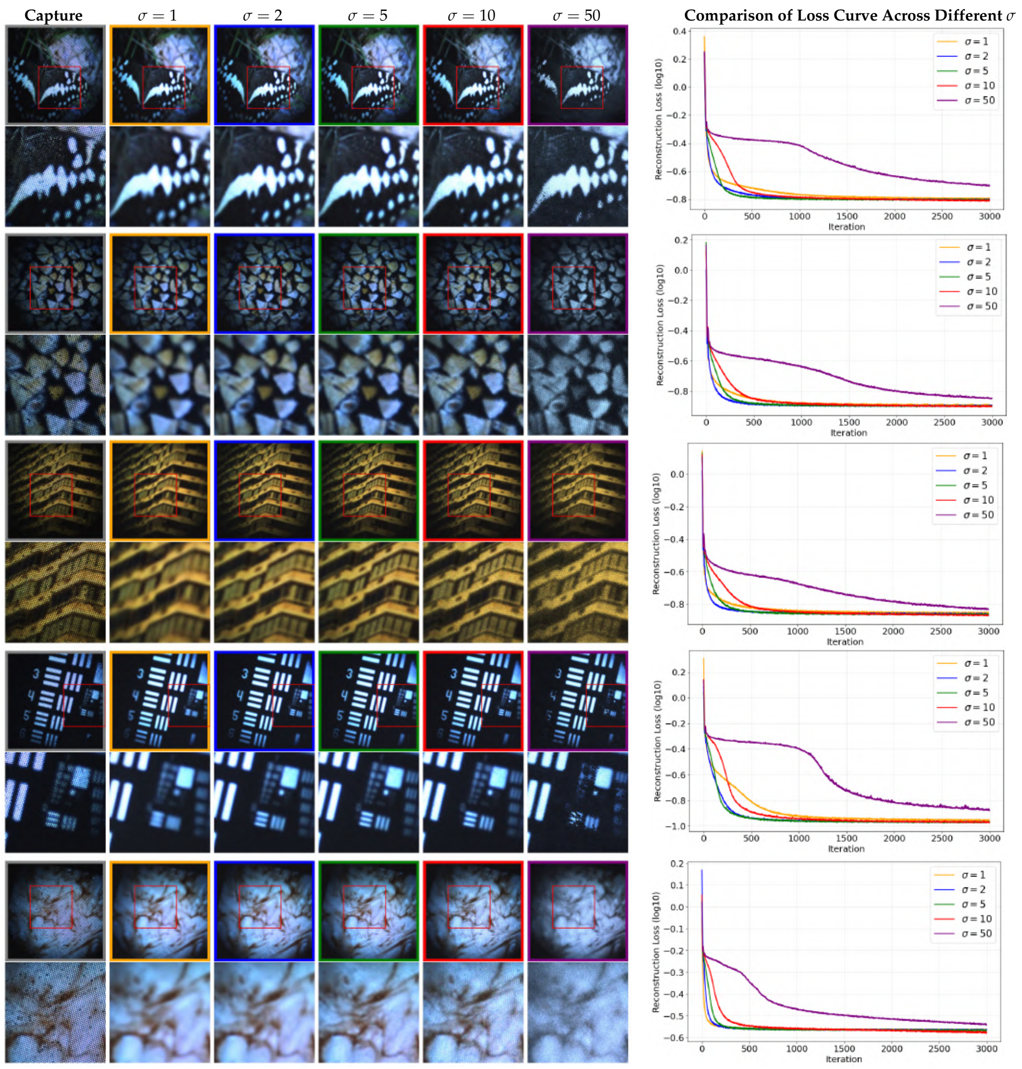

Fig. S3. Effect of varying the standard deviation σ in the Fourier positional encoding on experimental captures. Five examples are shown with $\sigma \in \{1, 2, 5, 10, 50\}$, all using homography as the motion model and Fourier positional encoding for scene reconstruction. For each video, only the first frame is shown.

Table S1. Effect of parameter σ in Fourier positional encoding on reconstruction quality for the simulated data in Fig. S4.

Sample	PSNR [dB] ↑						SSIM ↑					
	Captured	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 50$	Captured	$\sigma = 1$	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$	$\sigma = 50$
Liver	8.66	22.67	**23.09**	21.89	20.64	16.23	0.149	0.644	**0.678**	0.545	0.433	0.273
Kidney	7.92	21.72	**22.41**	20.99	20.04	16.06	0.099	0.755	**0.781**	0.596	0.427	0.399
Lens Tissue	14.89	22.89	23.09	**23.40**	23.16	17.67	0.463	0.759	0.788	**0.803**	0.757	0.208
Resolution Chart	13.02	17.87	20.99	**23.08**	22.89	17.44	0.632	0.692	0.806	**0.843**	0.821	0.473
Tiger	13.17	21.45	21.67	**21.69**	21.44	18.08	0.393	0.720	**0.732**	0.723	0.646	0.272
Orange Butterfly	13.89	20.64	21.69	**22.40**	22.13	16.39	0.424	0.752	0.777	**0.777**	0.679	0.221
Black Butterfly	13.90	19.13	20.57	**21.43**	20.81	17.43	0.497	0.620	0.709	**0.785**	0.703	0.339
Tower	14.23	19.85	21.06	**22.06**	22.04	17.99	0.522	0.625	0.700	**0.729**	0.723	0.302

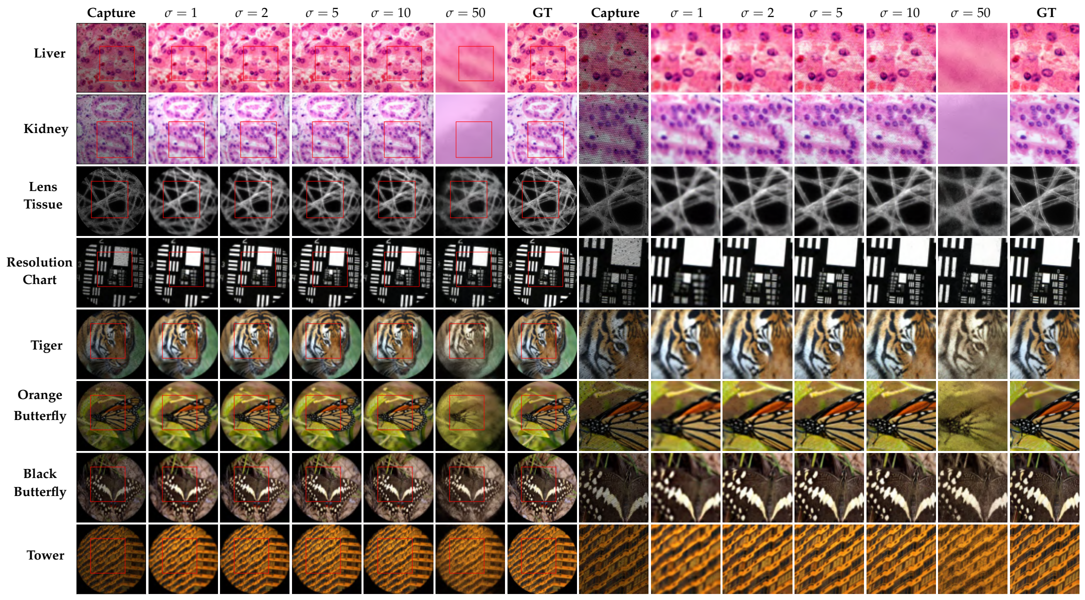

Fig. S4. Effect of varying the standard deviation σ in the Fourier positional encoding on simulation data. Five examples are shown with $\sigma \in \{1, 2, 5, 10, 50\}$, all using homography as the motion model and Fourier positional encoding for scene reconstruction. For each video, the first frame is shown.

3. ADDITIONAL RESULTS

A. Simulation Data

To further validate our approach, we present the results of our algorithm on three additional simulation examples. As before, random Brownian motion was applied to generate 20-frame videos for reconstruction. The first frame of each video and the corresponding results are shown in Fig. S5. On average, our method achieved a PSNR improvement of 8.03 dB and an SSIM improvement of 0.30 across the three samples. Beyond superresolution, our method also provides a compressed representation of videos. Storing 20 RGB frames of size $(1000, 1000)$ requires 60,000,000 uint8 values (57.22 MB), whereas the MLPs contain 464,139 float32 parameters (1.77 MB), yielding a 96.91% compression ratio. Reconstruction for all three examples runs in under two minutes on an A100 GPU. Detailed statistics for these additional simulations are reported in Table S2.

B. Experimental Data (2D Scene)

To complement the main experimental results, we provide reconstructions from eight additional examples. The reconstruction results for these additional cases are shown in Fig. S6.

C. Fiber Mask Extraction

The fiber mask can be extracted by comparing the reconstructed and captured videos. Specifically, each reconstructed frame is subtracted from its captured counterpart, and the results are averaged. A binary mask is then obtained using automated thresholding, such as Otsu's method [3]. Averaging across frames improves robustness: when dark regions of the scene overlap with the fiber mask, distinguishing between the mask and the scene becomes difficult. However, since the fiber mask is fixed while the scene moves, a given fiber location can appear over brighter regions in different frames. This variation allows the averaging process to enhance the mask contrast and yield a more accurate estimation (Fig. S7). Unlike prior work that requires explicit calibration with a white reference image [2, 4–8], our burst-based reconstruction method enables the computational extraction of the fiber mask. The extracted mask can further be used for single-image superresolution through techniques such as inpainting. Specifically, an MLP is trained on the intensities at known core positions and used to reconstruct the unknown ones. The results are shown in Fig. S8, where our method outperforms inpainting, highlighting the advantage of using multiple frames for reconstruction.

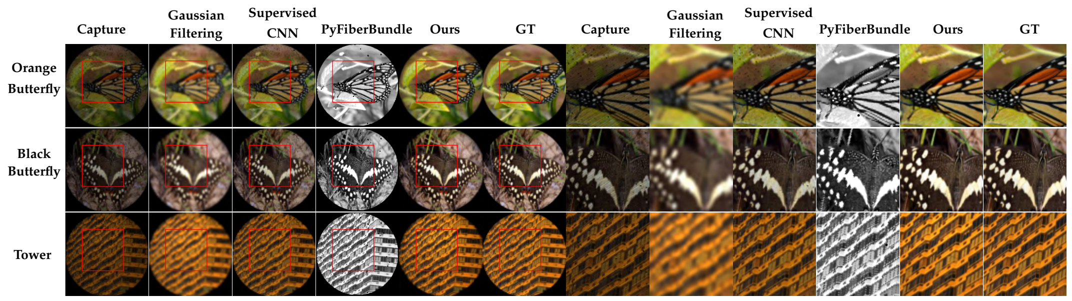

Fig. S5. Reconstruction results on simulated data. For each video, only the first frame is shown.

Table S2. Quantitative reconstruction results for the samples in Fig. S5.

Video Name	Video Dimension	PSNR [dB]/SSIM ↑						Runtime (s) ↓	Compression ↑
		Captured	Gaussian Filtering	Supervised CNN	PyFiberBundle [2]	Ours	Improvement		
Orange Butterfly	(20,1000,1000,3)	13.89/0.42	20.06/0.71	20.80/0.74	13.08/0.68	**22.44/0.77**	+8.55/+0.35	100.61	96.91%
Black Butterfly	(20,1000,1000,3)	13.90/0.50	18.77/0.56	20.22/0.75	14.89/0.61	**21.53/0.79**	+7.63/+0.30	104.21	96.91%
Tower	(20,1000,1000,3)	14.23/0.52	19.30/0.58	20.48/0.74	11.81/0.50	**22.13/0.77**	+7.90/+0.25	103.02	96.91%

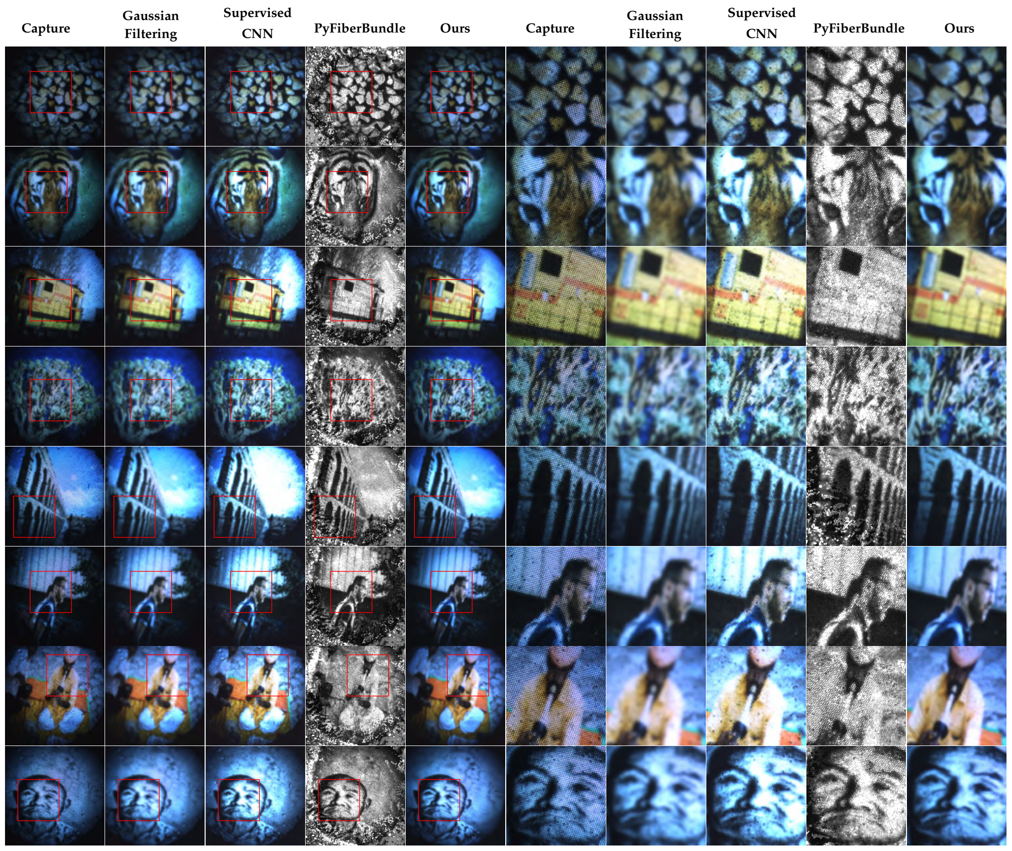

Fig. S6. Reconstruction results on samples displayed on a screen and captured with our imaging setup. For each video, the first frame is shown.

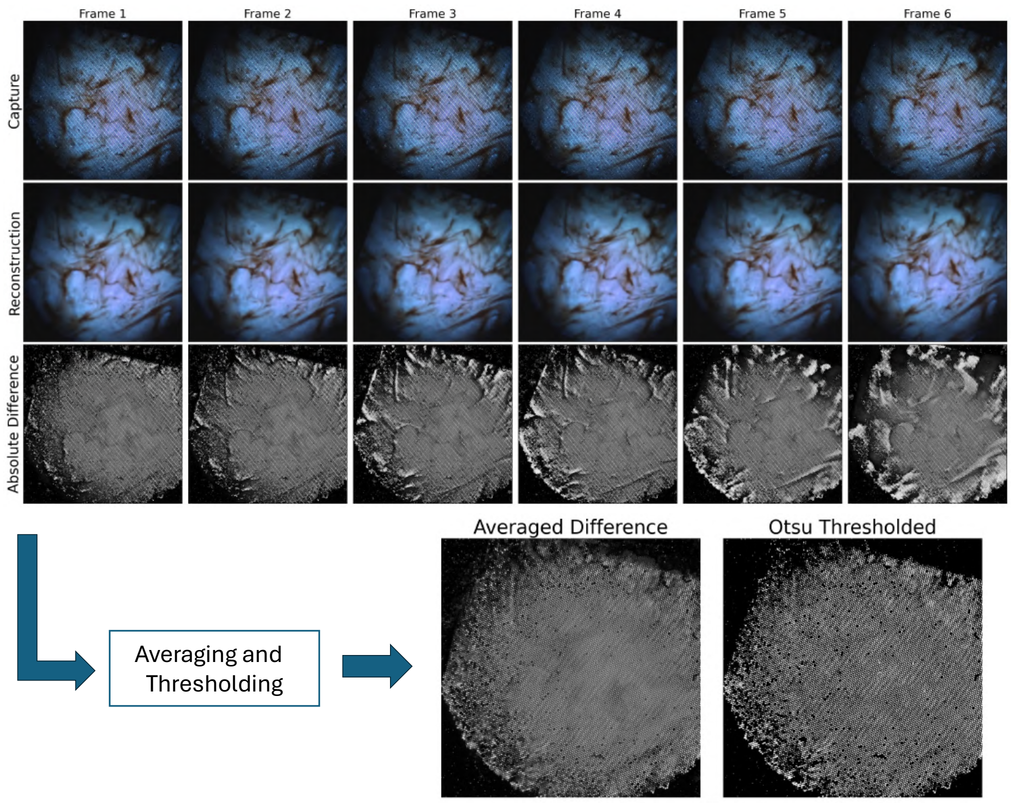

Fig. S7. Image processing pipeline for computationally extracting the fiber layout.

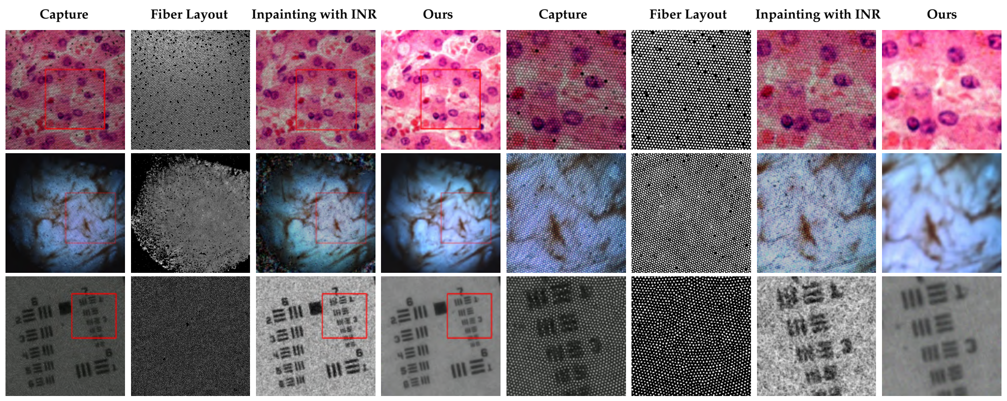

Fig. S8. Comparison of our method with inpainting based on implicit neural representations. In the third column, an INR is trained on known core positions to interpolate the unknown ones.

4. ADDITIONAL EXAMPLES FOR ABLATION STUDY

A. Impact of Motion Model

We evaluated several additional simulation and experimental captures to examine the effect of the motion model. By fixing the scene reconstruction network f_θ and the positional encoding $\gamma(\cdot)$, we tested different motion model strategies for the motion estimator MLP g_ϕ, including a no-motion baseline, optical flow (with/without total variation (TV) loss), and homography. In Fig. S9, Fourier positional encoding [1] with a ReLU MLP is used for f_θ, while in Fig. S10, a SIREN MLP [9] without positional encoding is used for f_θ. In both cases, similar patterns were observed. When motion is not explicitly modeled and the MLP f_θ is only used to memorize and reconstruct the video, the results are poor, underscoring the importance of motion modeling. Optical flow without total variation loss also fails to estimate motion reliably. Even with total variation loss, optical flow struggles for self-similar structures, as seen in the third example in Fig. S10. In contrast, homography consistently yields the best results across all examples. This advantage arises because homography estimates a single global transformation for the entire frame, whereas optical flow attempts to predict pixel-wise motion, making it less robust in challenging scenarios. The simulated results in Fig. S11 show a consistent trend, with homography outperforming other motion models both visually and quantitatively (Table S3).

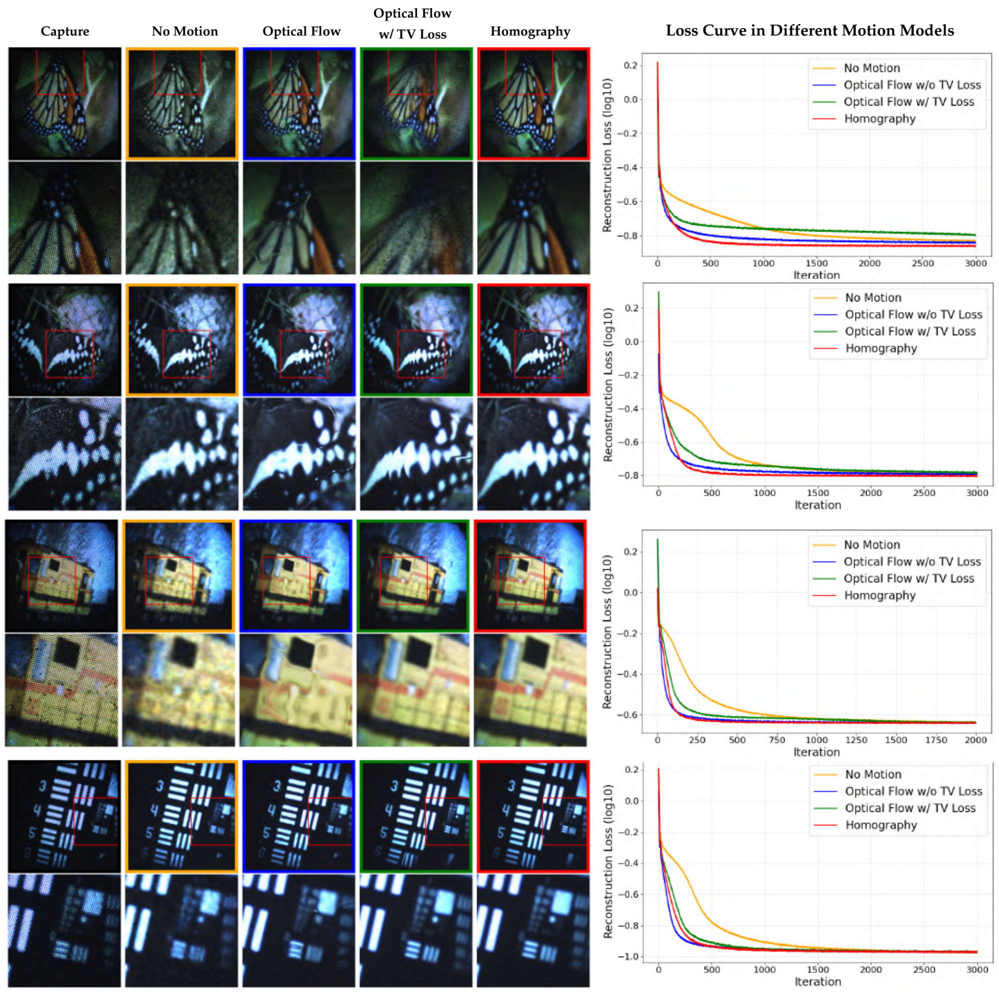

Fig. S9. Comparison of motion models for reconstruction quality in experimental captures. The scene reconstruction MLP f_θ is set to ReLU with Fourier positional encoding [1] $\gamma(.)$ in all cases, while different motion models are tested. For each video, the first frame is shown in the figure.

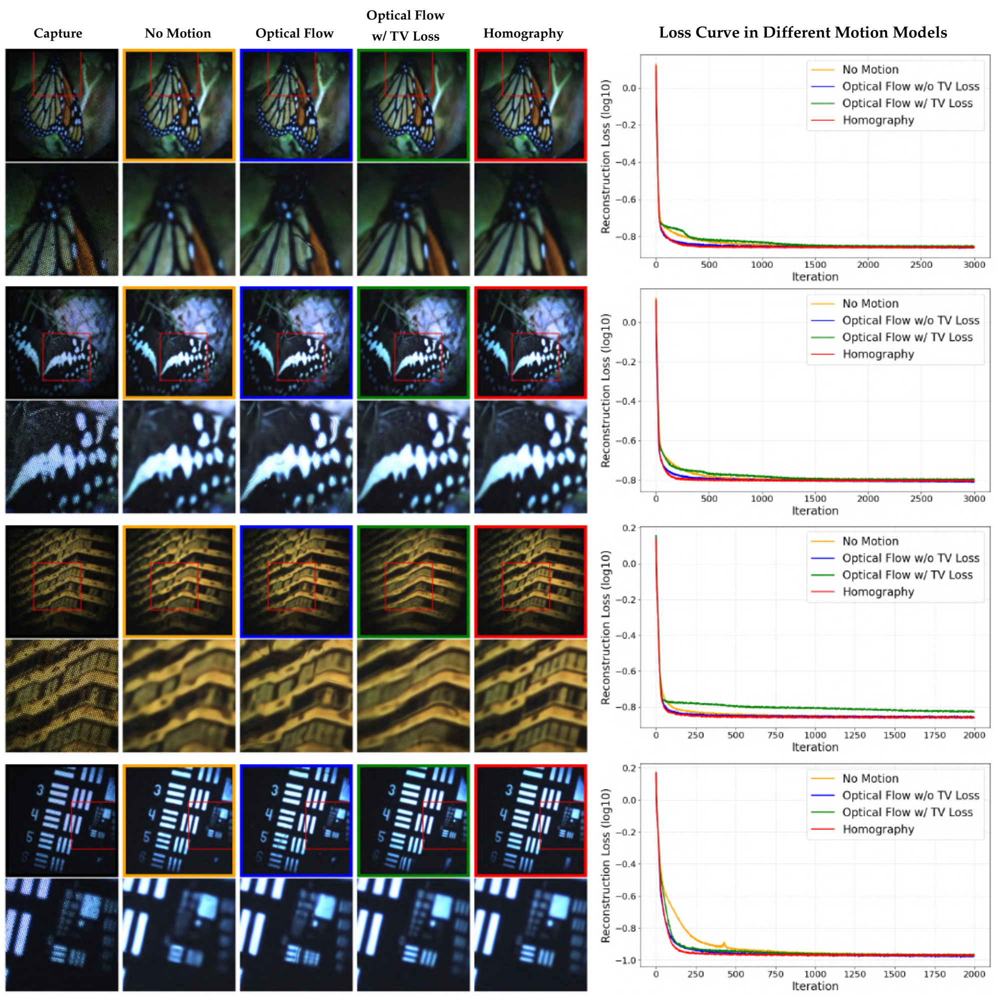

Fig. S10. Comparison of motion models for reconstruction quality in experimental captures. The scene reconstruction MLP f_θ is set to SIREN [9] without any positional encoding $\gamma(.)$ in all cases, while different motion models are tested. For each video, the first frame is shown.

Table S3. Comparison of motion models on reconstruction quality for simulated data shown in Fig. S11.

Sample	PSNR [dB] ↑					SSIM ↑				
	Captured	No motion	Flow w/o TV	Flow w/ TV	Homography	Captured	No motion	Flow w/o TV	Flow w/ TV	Homography
Liver	8.66	19.14	12.19	19.05	**22.63**	0.149	0.429	0.047	0.390	**0.672**
Kidney	7.92	18.85	12.14	18.86	**22.26**	0.099	0.436	0.041	0.484	**0.752**
Lens Tissue	14.89	23.24	18.21	22.99	**23.54**	0.463	0.676	0.600	0.733	**0.810**
Resolution Chart	13.02	20.57	19.24	19.50	**23.13**	0.632	0.684	0.779	0.758	**0.851**
Tiger	13.17	20.48	18.45	21.10	**21.83**	0.393	0.563	0.508	0.670	**0.721**
Orange Butterfly	13.89	19.71	17.18	20.70	**22.44**	0.424	0.525	0.504	0.707	**0.770**
Black Butterfly	13.90	20.07	15.91	19.97	**21.53**	0.497	0.550	0.473	0.630	**0.794**
Tower	14.23	20.08	15.91	20.10	**22.13**	0.522	0.529	0.506	0.618	**0.772**

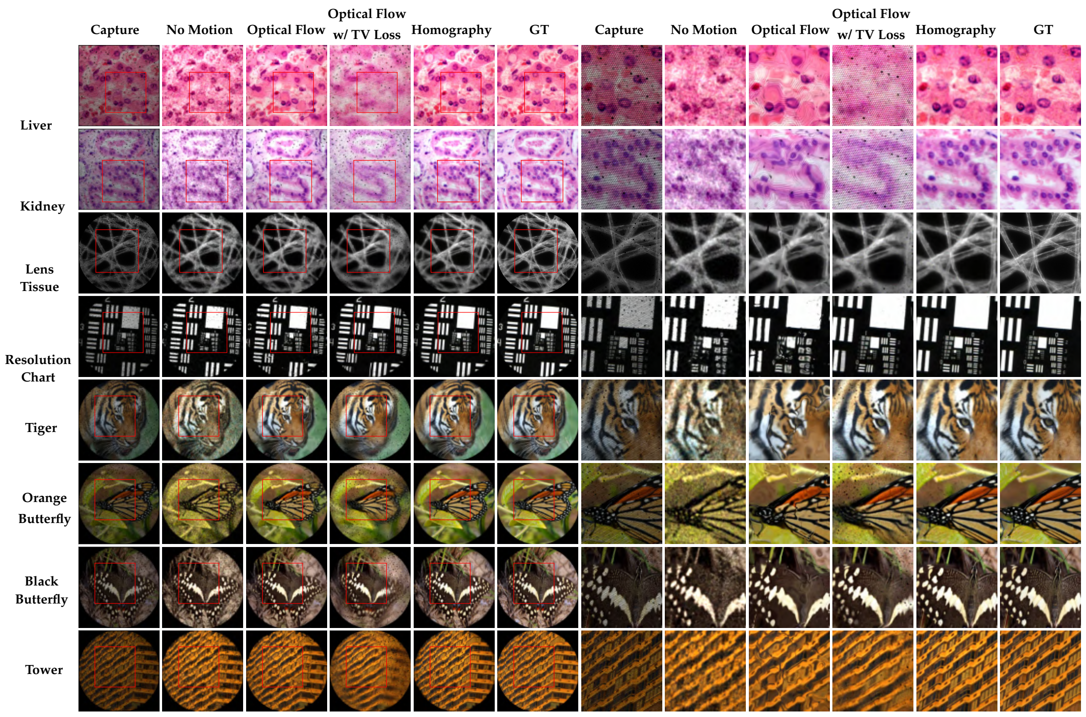

Fig. S11. Comparison of motion models on reconstruction quality on simulation data. The scene reconstruction MLP f_θ is set to ReLU with Fourier positional encoding [1] $\gamma(.)$ in all cases, while different motion models are tested. For each video, the first frame is shown.

B. Impact of MLP Architecture and Positional Encoding

We evaluated additional experimental and simulation captures to study the effect of positional encoding. By fixing the motion network g_ϕ to a homography, we tested different encoding strategies for the scene reconstruction network f_θ. As shown in Fig. S12, ReLU without positional encoding fails to reconstruct the scene, highlighting the necessity of encoding. The closest baseline in terms of the loss curve is ReLU with NeRF positional encoding [10], which improves reconstruction but still struggles with high-frequency details. Both methods show similar loss behavior during early iterations, consistent with the observation that networks first learn low-frequency components before high-frequency ones [11]. Hash encoding [12] further improves reconstruction compared to NeRF, although honeycomb artifacts remain. The best reconstructions are achieved with ReLU combined with Fourier encoding [1] and with SIREN [9], which uses sinusoidal activations without positional encoding. Both methods effectively preserve high-frequency content, although SIREN occasionally shows unstable loss curves, as seen in the fourth example of Fig. S12. Quantitative evaluation on simulated data (Table S4 and Fig. S13) confirms that ReLU with Fourier positional encoding [1] achieves the best performance.

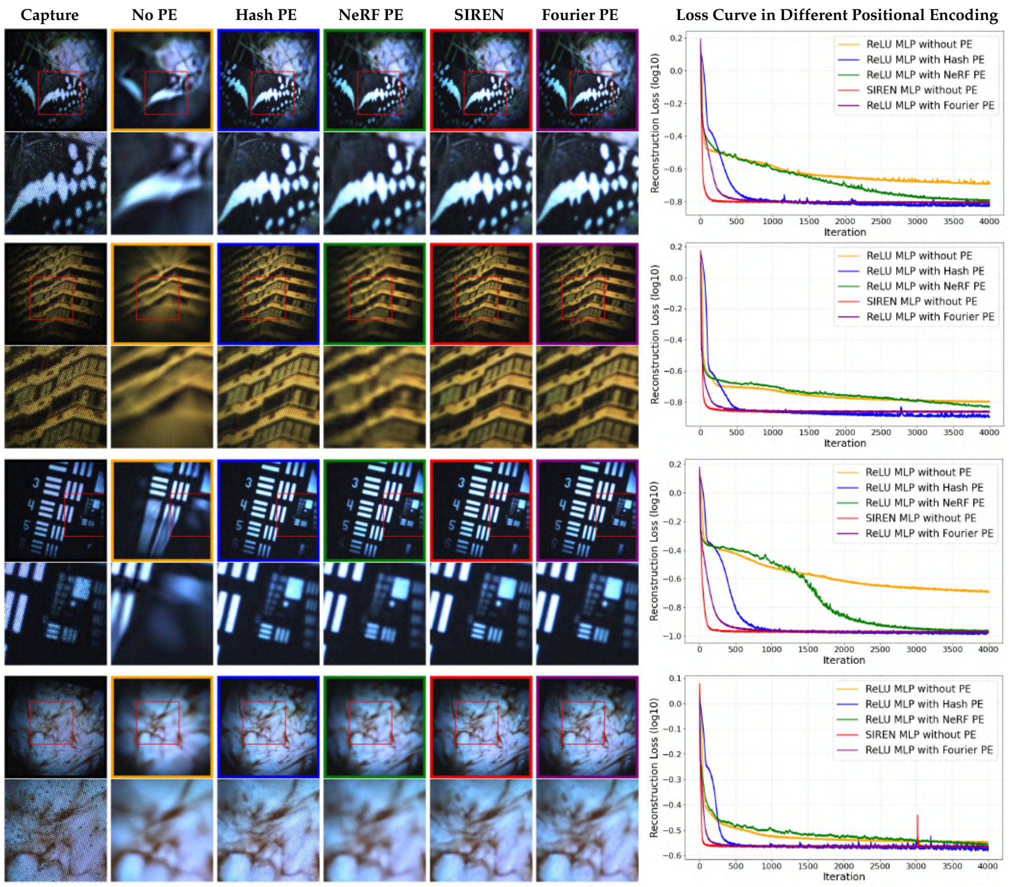

Fig. S12. Comparison of MLP architectures and positional encodings for the scene reconstruction MLP f_θ on experimental captures. The motion estimation MLP g_ϕ is fixed to a homography model for all cases. For each video, the first frame is shown in the figure.

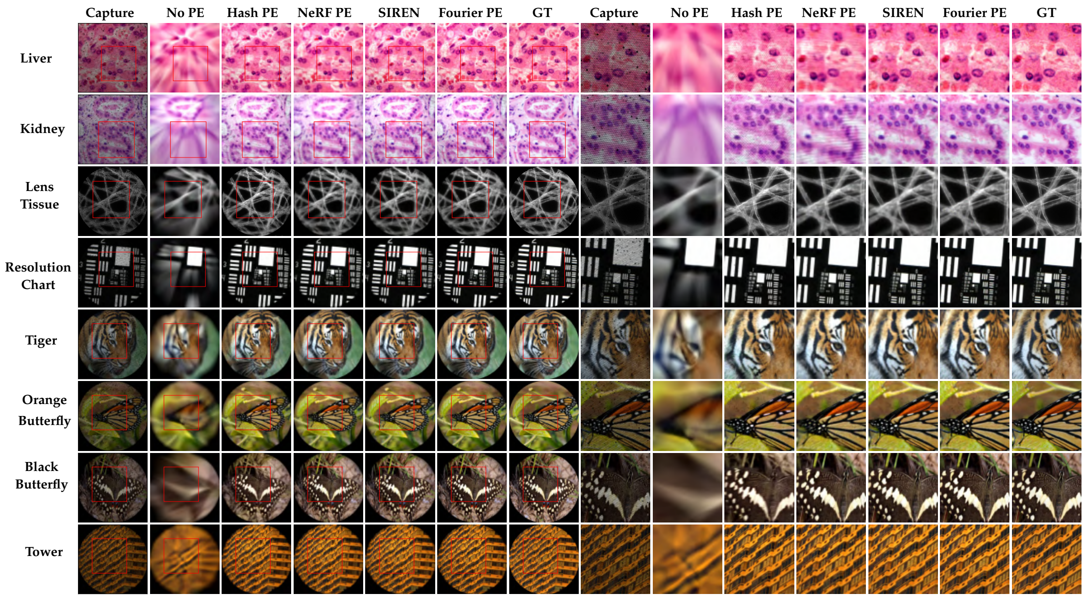

Fig. S13. Comparison of MLP architectures and positional encodings for the scene reconstruction MLP f_θ on simulation data. The motion estimation MLP g_ϕ is fixed to a homography model for all cases. For each video, the first frame is shown in the figure.

Table S4. Comparison of MLP architectures and positional encodings on reconstruction quality for simulated data shown in Fig. S13.

Sample	PSNR [dB] ↑						SSIM ↑					
	Captured	No PE	Hash PE[12]	NeRF PE[10]	SIREN[9]	Fourier PE[1]	Captured	No PE	Hash PE[12]	NeRF PE[10]	SIREN[9]	Fourier PE[1]
Liver	8.66	18.32	20.32	22.11	22.13	**22.63**	0.149	0.458	0.420	0.565	0.600	**0.672**
Kidney	7.92	17.69	19.12	16.85	21.99	**22.26**	0.099	0.583	0.336	0.391	0.739	**0.752**
Lens Tissue	14.89	18.31	22.54	23.16	23.20	**23.54**	0.463	0.563	0.703	0.763	0.806	**0.810**
Resolution Chart	13.02	11.00	22.14	22.46	22.36	**23.13**	0.632	0.528	0.800	0.853	**0.869**	0.851
Tiger	13.17	16.87	20.61	21.39	21.81	**21.83**	0.393	0.571	0.555	0.706	**0.767**	0.721
Orange Butterfly	13.89	17.30	21.47	22.10	22.44	**22.44**	0.424	0.625	0.594	0.781	**0.851**	0.770
Black Butterfly	13.90	14.55	17.99	21.23	21.35	**21.53**	0.497	0.435	0.567	0.754	**0.799**	0.794
Tower	14.23	16.09	21.36	21.51	**22.15**	22.13	0.522	0.490	0.670	0.742	**0.813**	0.772

5. DETAILS OF BASELINE COMPARISONS

For PyFiberBundle [2], we employed the `pybundle.calib_tri_interp` function, which is based on triangular linear interpolation. While the package also provides a function for processing a sequence of images, this approach yielded inferior results. Therefore, only the results obtained using `calib_tri_interp` are reported in this paper.

We trained a supervised CNN on 4,100 simulated images using the architecture shown in Fig. S14, with mean squared error (MSE) as the loss function. The model was trained for 20 epochs using the Adam optimizer [13], and the corresponding training loss curve is also shown in Fig. S14. Although the model performs well on images from the training set, it fails to generalize to unseen images and experimental data, suggesting that supervised learning does not generalize well across different sample types.

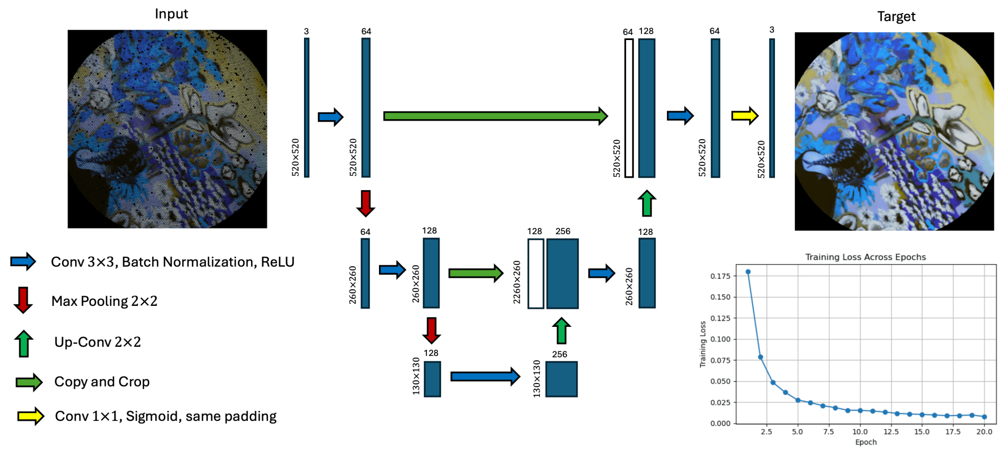

Fig. S14. Architecture of the supervised CNN trained on simulated data, along with the corresponding training loss curve.

6. CODE AVAILABILITY

The code is publicly available at https://github.com/amirrezavazifeh/Neural-Field-Fiber-Bundle-Imaging. The repository includes:

- `utils.py:` functions and classes for motion modeling and positional encoding.
- `main.ipynb:` Jupyter notebook for running the method on new image bursts.

To run reconstruction on a new burst, frames should be named sequentially as `0_000.png`, `1_000.png`, . . . , `{N-1}_000.png`, where N is the number of frames. Users can select the motion model (e.g., homography), MLP type (e.g., ReLU with Fourier positional encoding), and key hyperparameters (e.g., σ for positional encoding). The repository includes scripts for ablation studies on motion models, positional encoding, and hyperparameter analysis. Each scene may require hyperparameter tuning for optimal results. All experiments can be reproduced using A100 GPUs on Google Colab.

REFERENCES

1. M. Tancik, P. Srinivasan, B. Mildenhall, *et al.*, "Fourier features let networks learn high frequency functions in low dimensional domains," in *Advances in Neural Information Processing Systems*, (2020), pp. 7537–7547.
2. M. R. Hughes, "Real-timing processing of fiber bundle endomicroscopy images in Python using PyFibreBundle," Appl. Opt. **62**, 9041–9050 (2023).
3. N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," IEEE Trans. on Syst. Man, Cybern. **9**, 62–66 (1979).
4. P. Wang, G. Turcatel, C. Arnesano, *et al.*, "Fiber pattern removal and image reconstruction method for snapshot mosaic hyperspectral endoscopic images," Biomed. Opt. Express **9**, 780–790 (2018).
5. K. Vyas, M. Hughes, B. G. Rosa, and G.-Z. Yang, "Fiber bundle shifting endomicroscopy for high-resolution imaging," Biomed. Opt. Express **9**, 4649–4664 (2018).
6. J.-H. Han and S. M. Yoon, "Depixelation of coherent fiber bundle endoscopy based on learning patterns of image prior," Opt. Lett. **36**, 3212–3214 (2011).
7. S. P. Mekhail, N. Abudukeyoumu, J. Ward, *et al.*, "Fiber-bundle-basis sparse reconstruction for high resolution wide-field microendoscopy," Biomed. Opt. Express **9**, 1843–1851 (2018).
8. X. Liu, L. Zhang, M. Kirby, *et al.*, "Iterative ℓ_1-min algorithm for fixed pattern noise removal in fiber-bundle-based endoscopic imaging," J. Opt. Soc. Am. A, Opt. Image Sci. Vis. **33**, 630–636 (2016).
9. V. Sitzmann, J. N. P. Martel, A. W. Bergman, *et al.*, "Implicit neural representations with periodic activation functions," in *Advances in Neural Information Processing Systems*, (2020).
10. B. Mildenhall, P. P. Srinivasan, M. Tancik, *et al.*, "NeRF: representing scenes as neural radiance fields for view synthesis," Commun. ACM **65**, 99–106 (2022).

11. N. Rahaman, A. Baratin, D. Arpit, *et al.*, "On the spectral bias of neural networks," in *International Conference on Machine Learning*, (2019).
12. T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," ACM Trans. on Graph. **41**, 1–15 (2022).
13. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv [cs.LG] (2014).