A fast algorithm for 3D volume reconstruction from light field microscopy datasets

Jonathan M. Taylor[1,✉]

[1]School of Physics and Astronomy, University of Glasgow, Glasgow G12 8QQ, UK.

Light field microscopy can capture 3D volume datasets in a snapshot, making it a valuable tool for high-speed 3D imaging of dynamic biological events. However, subsequent computational reconstruction of the raw data into a human-interpretable 3D+time image is very time-consuming, limiting the technique's utility as a routine imaging tool. Here we derive improved equations for 3D volume reconstruction from light field microscopy datasets, leading to dramatic speedups. We characterise our open-source Python implementation of these algorithms, and demonstrate real-world reconstruction speedups of more than an order of magnitude compared to established approaches. The scale of this performance improvement opens up new possibilities for studying large timelapse datasets in light field microscopy.

Correspondence: *jonathan.taylor@glasgow.ac.uk*

Light field microscopy provides snapshot 3D imaging of dynamic scenes *via* a lenslet array placed in the image plane of the microscope, which casts a multi-apertured intensity pattern onto a camera sensor. The mix of spatial and angular information about the target sample emission in this single raw 2D snapshot image allows 3D image reconstruction across an extended depth-of-field (Fig. 1a), but only after intensive computational processing (1). Solving this inverse problem is extremely computationally demanding, traditionally requiring the computation of thousands of Fourier transforms per iteration. This requirement for high levels of computational resource is particularly problematic given the attractiveness of light field imaging for 3D time series (2–5), where a large number of different timepoints must all be reconstructed.

Here we will demonstrate how to mathematically simplify the light field reconstruction process, speeding up real-world computation times by more than an order of magnitude, while delivering identical output volume results.

The image reconstruction process is an inverse problem that can be cast as a deconvolution. The spatially-variant point spread function (PSF) of the light field microscope is typically computed theoretically from wave-optics calculations (1, 2), and Richardson-Lucy deconvolution is then used to estimate the 3D volume that gives rise to the measured 2D intensity pattern observed on the camera sensor. The standard Richardson-Lucy algorithm is described by the following iterative formula:

$$O_{\text{est}}^{(i+1)} = O_{\text{est}}^{(i)} \times H^T \left(\frac{I}{H(O_{\text{est}}^{(i)})} \right), \qquad \textbf{(1)}$$

although the widely-used light field implementation in (2) uses an alternative variation (see footnote (6) for further discussion) where the error term is computed in object space:

$$O_{\text{est}}^{(i+1)} = O_{\text{est}}^{(i)} \times \frac{H^T I}{H^T H(O_{\text{est}}^{(i)})}. \qquad \textbf{(2)}$$

In either form, $\times$ denotes elementwise multiplication and the fraction implies elementwise division; $O_{\text{est}}^{(i)}$ is the estimation of the 3D object to be reconstructed, at iteration i; I is the 2D light field image recorded on the camera; H is the "forward projection" operator mapping from the object O to the resultant camera image I; and H^T is the matrix transpose of the operator H. The optical interpretation of H^T leads to it being termed the backward-projection operator. A typical starting condition would be $O_{est}^{(0)} = H^T I$, and O_{est} converges to an estimate of the true object O over $N_{\text{iter}} \sim 10$ iterations.

The basic building blocks of the deconvolution problem are therefore the forward- and backward-projection operators, H and H^T, which model the image formation process. In light field microscopy these projection operators are expressed as the sum of many separate convolution operations. Given a three-dimensional object O consisting of voxels indexed O_{xyz}, each pixel value I_{mn} of a forward-projected image I can be computed as

$$I_{mn} = \sum_{xyz} O_{xyz} \times H_{mnxyz}, \qquad \textbf{(3)}$$

where H_{mnxyz} are the matrix elements of the PSF applicable to voxel x, y, z. To render the reconstruction problem tractable, raw images are resampled such that footprint of each lenslet in the lenslet array spans an exact odd integer number of camera pixels, A. Thus any subpixel a, b at the same relative position within any lenslet footprint will have the same the point spread function (Fig. 1b). This simplifies the problem, enabling Equation 3 to be rewritten as:

$$I = \sum_{abz} M_{ab}\{O_z\} \otimes H_{abz} \qquad \textbf{(4)}$$

where M_{ab} is a masking operator which zeroes all pixels except those in the image pixel group satisfying $x \bmod A = a$ and $y \bmod A = b$, and $\otimes$ is the convolution operator. H_{abz} represents the complete PSF for a voxel at coordinate a, b, z. Thus the forward-projected image from an object consisting of Z individual z-planes can be computed using a total of $A^2 Z$ convolution operations.

According to the convolution theorem, each convolution consists of two Fourier transforms and one inverse, each com-

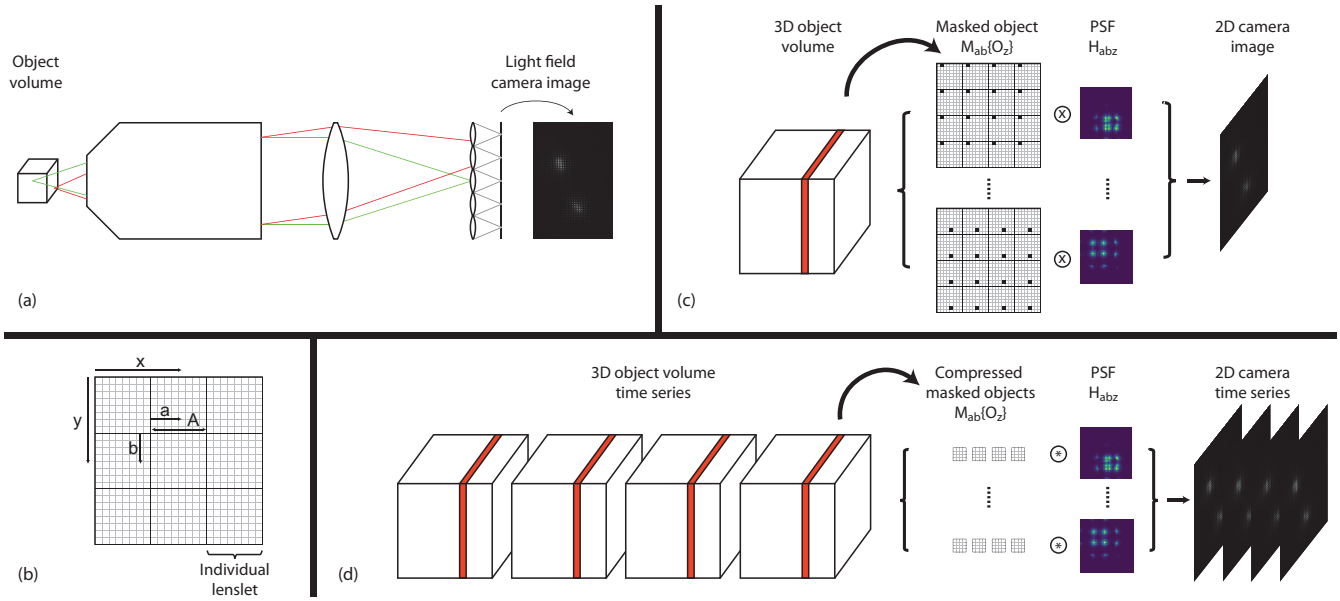

Fig. 1. (a) Optical schematic of light field microscopy. (b) Pixel indexing relative to lenslet footprints. (c) Conventional projection operation for deconvolution ($\otimes$ represents convolution). (d) New fast projection strategy ($\circledast$ represents a new custom operation - see main text).

puted using the Fast Fourier Transform (FFT) algorithm:

$$M_{ab}\{O_z\} \otimes H_{abz} = \mathcal{F}^{-1}\left(\mathcal{F}(M_{ab}\{O_z\}) \times \mathcal{F}(H_{abz})\right). \quad \textbf{(5)}$$

Any strategy aiming for more than merely incremental performance speedup of the overall calculation must speed up all three of the FFT operations in this equation. Speeding up any one of these on its own would be insufficient: even if the run time for one of them on its own could be reduced to zero, the overall run time would still only be improved by a factor of $\sim 30\%$. In what follows we will demonstrate how to improve the run time of each of these three operations in turn, to achieve an order of magnitude speedup in computation time. Our strategy is illustrated schematically in Fig. 1c-d.

The discrete Fourier transform of the masked object $M_{ab}\{O_z\}$ involves a high degree of redundancy, since most elements of this masked object array are zero. We observe that this problem can be simplified by generalising the Danielson-Lanczos lemma (7, 8) to an A-way result. In one dimension, the k^{th} element F_k of the discrete Fourier transform of a function f can be expressed in terms of A smaller Fourier transforms:

$$F_k = \tilde{F}_k^{(0)} + (W)^k \tilde{F}_k^{(1)} + (W)^{2k} \tilde{F}_k^{(2)} + \ldots + (W)^{(A-1)k} \tilde{F}_k^{(A-1)} \quad \textbf{(6)}$$

where $W = \exp(-2\pi i/X)$ and $\tilde{F}_k^{(a)}$ is the (reduced-size) discrete Fourier transform of $\tilde{f}$, where $\tilde{f}$ is a vector consisting of only the nonzero elements of $M_a\{f\}$. Note that, in the notation of Equation 6, k indexes the reduced-size discrete Fourier transform $F_k^{(a)}$ beyond its normal domain of $k \in [0, X/A)$, exploiting the periodic boundary conditions implicit in the Fourier transform.

In our case the M operator ensures that only one of the A distinct terms on the right-hand size of Equation 6 is nonzero. Therefore, eliminating all the zero terms, generalising to two dimensions, and summing over all image pixel groups a, b:

$$F_{mnz} = \sum_{ab} (W)^{abmn} \tilde{F}_{mnz}^{(a,b)}. \quad \textbf{(7)}$$

Consequently, instead of requiring A^2 FFTs that each operate over the full image size XY, we have reduced these to operating on arrays of size XY/A^2 (compressed arrays representing image pixel groups each containing only those pixels retained by the $M_{ab}\{O_z\}$ operator). We have therefore reduced the computational requirements of these FFTs by a factor of A^2. After computation of the FFTs, the weighting multiplications in Equation 7 must still be applied, albeit in an operation of reduced computational complexity $\mathcal{O}(XY)$, but overall the computational demands of computing F_{nm} are dramatically reduced.

We now move on to consider the Fourier transform of the point spread function H_{abz}. H and its Fourier transforms are invariant properties of the imaging system. If sufficient memory storage is available, the Fourier transforms can be precomputed once and the computation of $\mathcal{F}(H_{abz})$ eliminated completely from Equation 4. However, given typical values of $A \geq 15$, $Z \sim 50$ and megapixel images, tens or hundreds of GB of memory would be required to cache all the precomputed values. That may be feasible on some high-end CPU-based platforms, but exceeds the capacity of most GPU platforms. Nevertheless, even when precalculation is not possible, our algorithm amortises the computation of each Fourier transform across batches of multiple timepoints in a time series dataset, reconstructing each batch concurrently. We also exploit symmetry relationships in the PSFs, to permit rapid computation of e.g. $\mathcal{F}(H_{(A-a)bz})$ once $\mathcal{F}(H_{abz})$ has been computed on-the-fly.

Finally, by explicitly substituting Equation 5 into Equation 4 and exploiting the distributive property of the Fourier transform, we can dramatically reduce the number of inverse

Fourier transforms required:

$$I_{mn} = \sum_z \mathcal{F}^{-1} \sum_{ab} \mathcal{F}(M_{ab}\{O_z\}) \times \mathcal{F}(H_{abz}). \quad \textbf{(8)}$$

We note that in a practical implementation using a non-circulant Fourier transform it is not feasible to further promote the inverse FFT outside the summation over z, because the size of the intermediate arrays will vary according to the lateral extent of H_{abz}, and hence vary with z.

The backward-projection operator required for Richardson-Lucy deconvolution is traditionally denoted H^T in recognition that it is the transpose of the matrix operator H. However, in practice H and H^T are both implemented as convolutions (as per Equations 3-4). Hence we follow an almost identical approach for the backward projection, starting from the following equation in place of Equation 4:

$$O_{mnz} = \sum_{ab} M_{ab}\{I\} \otimes H^T_{abz}. \quad \textbf{(9)}$$

We note that during the one-time computation of H and H^T during initial optical modelling of the microscope PSF, computing H^T does not require additional convolution operations as used in (2), and can instead be populated near-instantaneously by pixelwise reshuffling of elements of H.

To achieve our anticipated order-of-magnitude speedup by using Equation 7 to compute $\mathcal{F}(M_{ab}\{O_z\})$ in Equation 5, it was necessary to write carefully-optimized custom computer code, since the specific multiplication and tiling process underpinning a practical implementation of Equation 7 is a performance bottleneck, and it is a highly bespoke operation that is not available in standard numerical libraries. We have made available a high-performance open-source reference implementation of our complete light field reconstruction algorithm (9). It is written primarily in the Python programming language, with performance-critical code written in C++, cython (10) and CUDA, drawing on the FFTW library (11) to compute Fourier transforms in our C++ code. To ensure maximum performance our code incorporates elements of dynamic load-balancing between CPU cores, and machine-adaptive runtime optimisations. We also provide a demonstration of how it can be integrated as-is into existing Matlab workflows (12).

Table 1 presents performance benchmarks for our light field reconstruction code. For batch reconstruction of a light field timelapse series, our new approach delivers a performance gain of $8-35\times$ compared to the widely-used MATLAB implementation (2) based on Equations 2, 4 and 9, and for maximum speed computation can be offloaded to a consumer GPU unit.

As explained above, our implementation performs optimally when batch-reconstructing multiple timepoints in a time series dataset simultaneously. Nevertheless, even with a non-optimal batch size of 1 we note that our implementation already outperforms existing implementations (Table 1). Fig. 2 explores the relationship between overall run time and batch size for our code, revealing a clear linear-plus-baseline scaling law. Regardless of batch size, a fixed amount of computational work must be performed to compute $\mathcal{F}(H)$. This

System	CPU/ GPU	Batch size	This work	(2)	Speedup
A	8xCPU	32	**78.4**	1075.9	14×
A	GPU	24	**5.6**	194.5	35×
B	16xCPU	32	**30.34**	-	
C	4xCPU	32	**140.0**	1095.6	7.8×
A	8xCPU	1	628.8	1075.9	(1.7×)
A	GPU	1	19.0	194.5	(10×)

Table 1. Measured run times (in seconds per frame) for reconstructing a light field timelapse series, showing speedups of $8-35\times$ compared to the existing state-of-the-art (Ref. (2)). The test parameters, representative of a typical light field imaging scenario, are specified in (9). Run times are also shown for a batch size of 1, as discussed in the main text, although that scenario does not allow our algorithms to perform at their fastest.
System A: 8 core Intel Xeon, 3GHz, 32GB RAM.
System A GPU: PNY Quadro RTX A4000, 1.56GHz, 16GB RAM, 224.03GB/s.
System B: 16 core Intel Xeon, 3.2GHz, 256GB RAM.
System C: 4 core Intel Core i5, 2.5GHz, 16GB RAM.

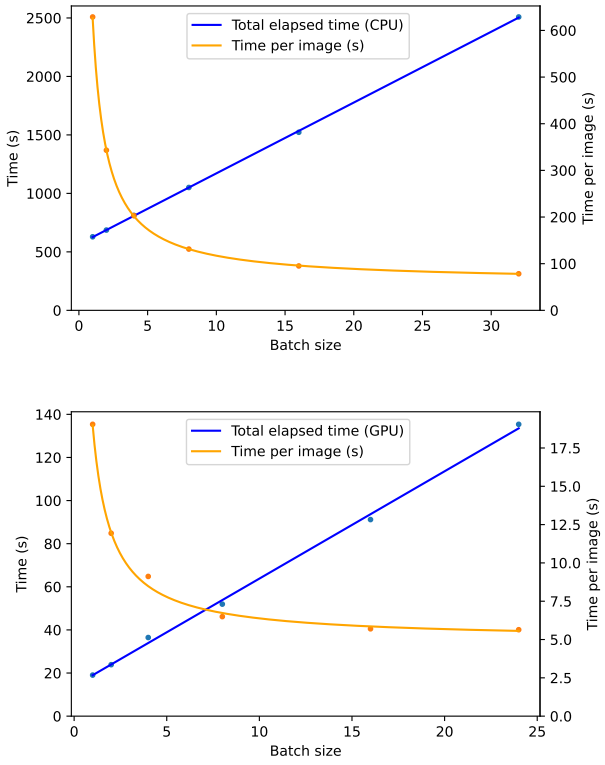

Fig. 2. Performance scaling with batch size, showing trend of constant baseline work (computing $\mathcal{F}(H)$) plus linear scaling of additional work per batch item. Datapoints are measured run times (System A as specified in Table 1) and trendlines are a fit to a linear scaling model.

baseline work can only be eliminated completely if sufficient RAM is available to cache pre-calculated values for all $\mathcal{F}(H)$, which can extend to hundreds of GB. The total additional work (computing the other elements of Equation 4 via Equations 7 and 8) scales linearly with the batch size. Hence there is a clear advantage to our amortising the constant work across simultaneous deconvolutions of multiple timepoints.

In our GPU implementation we measure that the linear-scaling work is more substantial compared to the baseline work of computing $\mathcal{F}(H)$, probably due to the challenges of developing optimized CUDA kernels to implement the highly

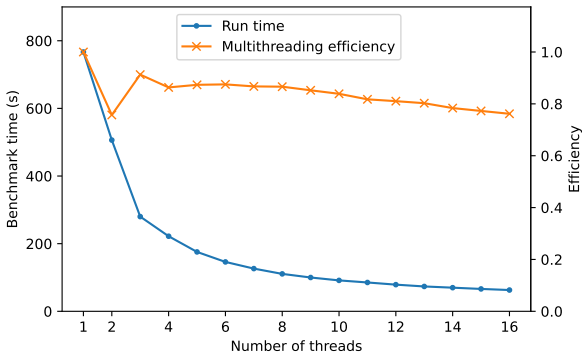

Fig. 3. Performance scaling with number of parallel CPU threads, showing run-time improvement with increasing number of parallel threads (System B as specified in Table 1). Multithreading efficiency is defined as 1.0 for a case where the measured time for an n-thread scenario is n times as fast as the 1-thread time.

specific custom operations embodied in Equation 7. This means that there are diminishing additional benefits to increasing the batch size beyond 8 on a GPU, which happily in turn means the RAM requirements are lower on a GPU, where RAM is typically scarcer.

Fig. 3 confirms that our implementation scales well when the work is parallelised across multiple CPU cores: the benchmark run time with 16 parallel threads was over 12 times faster than the single-threaded run time. The small decrease in efficiency (i.e. speedup being slightly less than n times when using n threads) can be explained by the increased pressure that multithreaded code imposes on the system's memory bandwidth. The slight anomaly in efficiency for 2 threads is likely related to the dual-CPU architecture of the testbed system.

While many researchers continue to prefer the mathematically precise reconstruction afforded by the classical approach of Equation 2, others are researching the use of machine-learning to estimate the object from the raw camera image (5, 13–15). The aim of such research is to compute an object estimate that is as faithful as possible to the true object, while using less computational time than required to explicitly compute Equation 2. Some approaches commence with a limited number of Richardson-Lucy iterations (13) or use direct optical modelling during the training phase (5); computational performance in both these scenarios will benefit directly from our new results. Pure machine-learning based approaches also stand to benefit from our results: machine-learning architectures in imaging are commonly based around convolutional neural networks, and it is increasingly recognised that networks perform better and can be trained faster when the network structure encapsulates physical insights into the image formation process (14, 15). Thus the mathematical insights behind our results hold promise for improving performance and effectiveness of future machine-learning architectures for light field microscopy reconstruction.

In summary, we have presented mathematical results enabling a dramatic reduction in computational work for 3D image reconstruction in light field microscopy. The results and approach we have presented here are potentially applicable to any multi-aperture computational imaging system with a space-variant PSF endowed with translational symmetry properties. We have made available an open-source implementation of our algorithms (9), with performance measured to be more than an order of magnitude faster than previously-available codes. We anticipate that our algorithms and open-source implementation will lead to an expansion in the uptake of light field microscopy as a tool for 3D+time imaging of rapid biological processes, now that the excessive computational demands of the reconstruction process have been brought under control.

Funding

Royal Society of Edinburgh (sabbatical grant).

Data availability

The computer code and test datasets underlying the results presented in this paper are available at Refs. (9, 12).

References

1. Michael Broxton, Logan Grosenick, Samuel Yang, Noy Cohen, Aaron Andalman, Karl Deisseroth, and Marc Levoy. Wave optics theory and 3-D deconvolution for the light field microscope. *Optics Express*, 21(21):25418, 2013. doi: 10.1364/oe.21.025418.
2. Robert Prevedel, Young Gyu Yoon, Maximilian Hoffmann, Nikita Pak, Gordon Wetzstein, Saul Kato, Tina Schrödel, Ramesh Raskar, Manuel Zimmer, Edward S. Boyden, and Alipasha Vaziri. Simultaneous whole-animal 3D imaging of neuronal activity using light-field microscopy. *Nature Methods*, 11(7):727–730, 2014. ISSN 15487105. doi: 10.1038/nmeth.2964.
3. Oliver Skocek, Tobias Nöbauer, Lukas Weilguny, Francisca Martínez Traub, Chuying Naomi Xia, Maxim I. Molodtsov, Abhinav Grama, Masahito Yamagata, Daniel Aharoni, David D. Cox, Peyman Golshani, and Alipasha Vaziri. High-speed volumetric imaging of neuronal activity in freely moving rodents. *Nature Methods*, 15(6):429–432, 2018. ISSN 15487105. doi: 10.1038/s41592-018-0008-0.
4. Nils Wagner, Nils Norlin, Jakob Gierten, Gustavo De Medeiros, Bálint Balázs, Joachim Wittbrodt, Lars Hufnagel, and Robert Prevedel. Instantaneous isotropic volumetric imaging of fast biological processes. *Nature Methods*, 16:497–500, 2019. doi: 10.1038/s41592-019-0393-z.
5. Zhaoqiang Wang, Lanxin Zhu, Hao Zhang, Guo Li, Chengqiang Yi, Yi Li, Yicong Yang, Yichen Ding, Mei Zhen, Shangbang Gao, Tzung K. Hsiai, and Peng Fei. Real-time volumetric reconstruction of biological dynamics with light-field microscopy and deep learning. *Nature Methods*, 18(May), 2021. ISSN 15487105. doi: 10.1038/s41592-021-01058-x.
6. Note1. In light field microscopy, native focal plane artefacts arise from missing structural information in the captured raw image. However, these are further exacerbated by the presence of background or scattered light outside the footprint of circular lenslets. Empirically the performance of the alternative form (Equation 2) degrades less severely in this scenario, although we are not aware of any explicit mention or derivation of this property in the literature.
7. G. C. Danielson and C. Lanczos. Some improvements in practical fourier analysis and their application to X-ray scattering from liquids. *Journal of the Franklin Institute*, 233(4):365–380, 1942. ISSN 00160032. doi: 10.1016/s0016-0032(42)90624-0.
8. William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C*. Cambridge University Press, 1992.
9. Jonathan M. Taylor. Python code for fast light field reconstruction. DOI: 10.5281/zenodo.7736455, 2023.
10. Stefan Behnel, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith. Cython: The Best of Both Worlds. *Computing in Science & Engineering*, 13:31–39, 2011.
11. Matteo Frigo. A fast Fourier transform compiler. *Proc. ACM SIGPLAN*, pages 169–180, 1999. ISSN 03621340. doi: 10.1145/989393.989457.
12. Jonathan M. Taylor. Matlab/Python integration for fast light field reconstruction. DOI: 10.5281/zenodo.7736536, 2023.
13. Yuanlong Zhang, Bo Xiong, Yi Zhang, Zhi Lu, Jiamin Wu, and Qionghai Dai. DiLFM: an artifact-suppressed and noise-robust light-field microscopy through dictionary learning. *Light: Science and Applications*, 10:152, 2021. ISSN 20477538. doi: 10.1038/s41377-021-00587-6.
14. Diptodip Deb, Zhenfei Jiao, Alex B. Chen, Michael Broxton, Misha B. Ahrens, Kaspar Podgorski, and Srinivas C. Turaga. FourierNets enable the design of highly non-local optical encoders for computational imaging. *arxiv preprint arXiv:2104.10611*, 2022.
15. Yue Li, Yijun Su, Min Guo, Xiaofei Han, Jiamin Liu, Harshad D. Vishwasrao, Xuesong Li, Ryan Christensen, Titas Sengupta, Mark W. Moyle, Ivan Rey-Suarez, Jiji Chen, Arpita Upadhyaya, Ted B. Usdin, Daniel Alfonso Colón-Ramos, Huafeng Liu, Yicong Wu, and

Hari Shroff. Incorporating the image formation process into deep learning improves network performance. *Nature Methods*, 19(11):1427–1437, 2022. ISSN 15487105. doi: 10.1038/s41592-022-01652-7.