Supplementary for "Neuron Segmentation using 3D Wavelet Integrated Encoder-Decoder Network"

S.1 3D wavelet filters

Generally, the 3D wavelet filters are tensor products of the two filters of 1D wavelet, $\mathrm{f}_l, \mathrm{f}_h$, i.e.,

$$\mathrm{f}_{c_0c_1c_2} = \mathrm{f}_{c_0} \otimes \mathrm{f}_{c_1} \otimes \mathrm{f}_{c_2}, \quad c_0, c_1, c_2 \in \{l, h\}, \tag{S.1}$$

where $\otimes$ represents the tensor product. Take Haar wavelet for example, the low-pass and high-pass filters of 1D Haar wavelet are

$$\mathrm{f}_l^{\mathrm{H}} = \frac{1}{\sqrt{2}}(1,1)^T, \quad \mathrm{f}_h^{\mathrm{H}} = \frac{1}{\sqrt{2}}(1,-1)^T. \tag{S.2}$$

Then, via Eq. (S.1), we get the filters of the corresponding 3D Haar wavelet:

$$\mathrm{f}_{lll}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}; \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}\right], \tag{S.3}$$

$$\mathrm{f}_{llh}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}; \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}\right], \tag{S.4}$$

$$\mathrm{f}_{lhl}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}; \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}\right], \tag{S.5}$$

$$\mathrm{f}_{lhh}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}; \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\right], \tag{S.6}$$

$$\mathrm{f}_{hll}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}; \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix}\right], \tag{S.7}$$

$$\mathrm{f}_{hlh}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}; \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}\right], \tag{S.8}$$

$$\mathrm{f}_{hhl}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}; \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}\right], \tag{S.9}$$

$$\mathrm{f}_{hhh}^{\mathrm{H}} = \frac{1}{2\sqrt{2}}\left[\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}; \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}\right]. \tag{S.10}$$

We here introduce the commonly used orthogonal Daubechies wavelets and biorthogonal Cohen wavelets. Their corresponding 3D wavelet filters are designed according to Eq. (S.1).

Orthogonal wavelets Daubechies wavelet is orthogonal, a set of orthogonal basis for $L^2(x)$ could be derived from its scaling and wavelet functions. Daubechies wavelet has an approximation order parameter p, and the length of its filter is $2p$. Table S.1 shows the low-pass filter $\mathrm{f}_l = \{f_k^{(l)}\}$ of the wavelets with order $p, 1 \le p \le 6$, while the high-pass filter $\mathrm{f}_h = \{f_k^{(h)}\}$ can be deduced from

$$f_k^{(h)} = (-1)^k f_{\mathcal{N}-k}^{(l)}, \tag{S.11}$$

where $\mathcal{N}$ is an odd number. Daubechies(1) is Haar wavelet.

Biorthogonal wavelets Cohen wavelets are symmetric biorthogonal wavelets, and each of them is associated with scaling function ϕ, wavelet function ψ, and their dual functions $\tilde{\phi}, \tilde{\psi}$. Correspondingly, it has four filters f_l, f_h, $\tilde{\mathrm{f}}_l$, and $\tilde{\mathrm{f}}_h$. While a signal is decomposed using filters f_l and f_h with DWT, it can be reconstructed using the dual filters $\tilde{\mathrm{f}}_l$ and $\tilde{\mathrm{f}}_h$ with IDWT. Cohen wavelet is with two order parameters p and $\tilde{p}$. Table S.2 shows the low-pass filters with orders $2 \le p = \tilde{p} \le 5$. Their high-pass filters can be deduced from

$$f_k^{(h)} = (-1)^k \tilde{f}_{\mathcal{N}-k}^{(l)}, \tag{S.12}$$

$$\tilde{f}_k^{(h)} = (-1)^k f_{\mathcal{N}-k}^{(l)}, \tag{S.13}$$

where $\mathcal{N}$ is an odd number. Cohen$(1,1)$ is Haar wavelet.

Wavelet theory is valid for finite or infinite filters, but the infinite case is rarely covered in practical interest.

Table S.3. Deep network configurations.

data size	channel number		3D U-Nets[a]	3D WaveUNets[b]
	encoder	decoder		
$32 \times 128 \times 128$	1, 4	4, 4	DS-x	WADS-y
$16 \times 64 \times 64$	4, 8	8, 4	DS-x	WADS-y
$8 \times 32 \times 32$	8, 16	16, 8	DS-x	WADS-y
$4 \times 16 \times 16$	16, 32	32, 16	DS-x	WADS-y
$4 \times 16 \times 16$	32, 32		bottom block	

[a] The three 3D U-Nets are named as 3D U-Net(x), $x \in$ {PU, PDc, ScIn}.
[b] The four 3D WaveUNets designed in this paper are named as 3D WaveUNet(y), $y \in$ {DDc, DIn, DI, DIDn}.

S.2 The naive down-sampling and up-sampling

In Eqs. (1) and (3), 3D DWT and IDWT are implemented using 3D naive down-sampling and up-sampling. For a tensor $\boldsymbol{X} = \{X_{i,j,k}\} \in \mathbb{R}^{d\times m\times n}$,

$$(\downarrow 2)\boldsymbol{X} = \{((\downarrow 2)\boldsymbol{X})_{i,j,k}\} \in \mathbb{R}^{\lfloor \frac{d}{2} \rfloor \times \lfloor \frac{m}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor}, \tag{S.14}$$

and

$$((\downarrow 2)\boldsymbol{X})_{i,j,k} = X_{2i,2j,2k}. \tag{S.15}$$

For a 3D tensor $\boldsymbol{X} = \{X_{i,j,k}\} \in \mathbb{R}^{d\times m\times n}$,

$$(\uparrow 2)\boldsymbol{X} = \{((\uparrow 2)\boldsymbol{X})_{i,j,k}\} \in \mathbb{R}^{2d\times 2m\times 2n}, \tag{S.16}$$

and

$$((\uparrow 2)\boldsymbol{X})_{i,j,k} = \begin{cases} X_{\frac{i}{2},\frac{j}{2},\frac{k}{2}} & \text{if } \frac{i}{2}, \frac{j}{2}, \frac{k}{2} \in \mathbb{Z}, \\ 0 & \text{else.} \end{cases} \tag{S.17}$$

S.3 The configuration for 3D WaveUNet

In Sec. 2.2, using the seven dual structures, we design seven 3D encoder-decoder networks for neuron segmentation. While the first three networks are variants of 3D U-Net, the last four are 3D WaveUNets designed in this paper. Each of them contains four nested dual structures with one bottom block containing two convolutions. Table S.3 illustrates their configurations. In Table S.3, the first column shows the input size. Every number in the table corresponds to a convolutional layer with batch normalization (BN) and ReLU. While the number in column "encoder" is the number of input channels of the convolution, the number in column "decoder" is that of output channels. At the end of networks, a convolution with kernel size of 1×1 converts the output of decoder into the predicted segmentation result.

Table S.1. Low-pass filters of the Daubechies wavelets. The high-pass filters of Daubechies wavelets could be deduced from the low-pass filters via Eq. (S.11). Daubechies wavelets are orthogonal.

p	1	2	3	4	5	6
$f_k^{(l)}$	1	$1+\sqrt{3}$	0.332670552950	0.230377813309	0.160102397974	0.111540743350
	1	$3+\sqrt{3}$	0.806891509311	0.714846570553	0.603829269797	0.494623890398
		$3-\sqrt{3}$	0.459877502118	0.630880767930	0.724308528438	0.751133908021
		$1-\sqrt{3}$	−0.135011020010	−0.027983769417	0.138428145901	0.315250351709
			−0.085441273882	−0.187034811719	−0.242294887066	−0.226264693965
			0.035226291886	0.030841381836	−0.032244869585	−0.129766867567
				0.032883011667	0.077571493840	0.097501605587
				−0.010597401785	−0.006241490213	0.027522865530
					−0.012580751999	−0.031582039317
					0.003335725285	0.000553842201
						0.004777257511
						−0.001077301085
factor	$1/\sqrt{2}$	$1/(4\sqrt{2})$	1	1	1	1

Table S.2. Low-pass filters of the Cohen wavelets. The high-pass filters of Cohen wavelets could be deduced via Eqs. (S.12)-(S.13). The filters and dual filters of biorthogonal Cohen wavelets are applied to decompose and reconstruct image, respectively.

$(p,\tilde{p})$	$(2,2)$		$(3,3)$		$(4,4)$		$(5,5)$	
filter	f_l	$\tilde{\mathrm{f}}_l$	f_l	$\tilde{\mathrm{f}}_l$	f_l	$\tilde{\mathrm{f}}_l$	f_l	$\tilde{\mathrm{f}}_l$
$f_k^{(l)}$	0	0	0	0.06629126	0	0	0.01345671	0
	0.35355339	−0.17677670	0	−0.19887378	−0.06453888	0.03782846	−0.00269497	0
	0.70710678	0.35355339	0.17677670	−0.15467961	−0.04068942	−0.02384947	−0.13670658	0.03968709
	0.35355339	1.06066017	0.53033009	0.99436891	0.41809227	−0.11062440	−0.09350470	0.00794811
	0	0.35355339	0.53033009	0.99436891	0.78848562	0.37740286	0.47680327	−0.05446379
	0	−0.17677670	0.17677670	−0.15467961	0.41809227	0.85269868	0.89950611	0.34560528
			0	−0.19887378	−0.04068942	0.37740286	0.47680327	0.73666018
			0	0.06629126	−0.06453888	−0.11062440	−0.09350470	0.34560528
					0	−0.02384947	−0.13670658	−0.05446379
					0	0.03782846	−0.00269497	0.00794811
							0.01345671	0.03968709
							0	0

S.3.1 The denoising block used in 3D WaveUNet(DIDn)

The denoising block, used in 3D WaveUNet(DIDn), is implemented by hard shrinkage with threshold $\lambda = 0.25$:

$$\text{HardShrink}(x) = \begin{cases} x, & \text{if } x > \lambda, \\ x, & \text{if } x < -\lambda, \\ 0, & \text{otherwise.} \end{cases} \tag{S.18}$$

In the denoising block, we filter every coefficient x in the seven high-frequency components

$$\boldsymbol{X}_c,\ c \in \{llh, lhl, lhh, hll, hlh, hhl, hhh\},$$

according to Eq. (S.18).

S.4 3D neuron reconstruction using various tracing algorithms

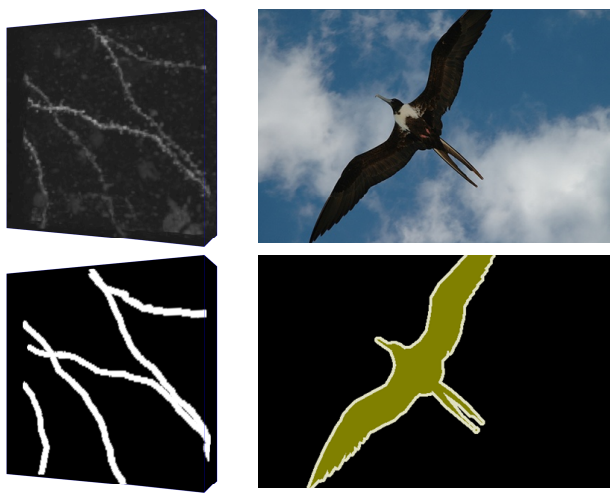
To better illustrate the effectiveness of 3D wavelet integrated deep networks on 3D neuron reconstruction, we apply various automatic tracing approaches to reconstruct the test images segmented by the different 3D deep networks integrated with or without 3D wavelets. Table S.4 shows the reconstruction performances of five reconstruction approaches, including APP2 (Xiao and Peng, 2013), Mean-shift Spanning Tree (MST) tracer (Peng et al., 2010), NeuroGPS (Quan et al., 2016), Snake tracer (Narayanaswamy et al., 2011), and TReMAP (Zhou et al., 2016), on the 28 test neuronal images segmented by various 3D deep networks. From Table S.4, one can find that, although these automatic tracing approaches perform diversely on the segmented neuronal images, 3D wavelet could consistently improve the reconstruction performance of these tracing approaches.

S.5 Comparison of neuronal image and natural image

Different from the common objects in natural images, the nerve fibers in neuronal cubes are line-shaped, with zigzag edges. Fig. S.1 shows an example neural cube from NeuCuDa and an example image from Pascal VOC (Everingham et al., 2015). During the annotation of nerve fiber in the cube, some label noises are introduced into the label matrix, because of the zigzag fiber edges. The label noises occupy a significant proportion in the voxels labeled as "nerve fiber"; in the natural images, although some label noises also appear near the object edges, they only occupy a small proportion. Therefore, the experiences sourced from natural image might not apply to the neuronal images, due to the gap between two domains.

Table S.4. Reconstruction results of various automatic tracing approaches on neuronal images segmented by 3D deep networks integrated with or without 3D wavelets.

3D Network		Reconstruction														
Architecture	Dual Structure	APP2			MST			NeuroGPSTree			Snake tracer			TReMap		
		ESA	DSA	PDS	ESA	DSA	PDS	ESA	DSA	PDS	ESA	DSA	PDS	ESA	DSA	PDS
3D U-Net	DS-PU	2.5444	7.3450	0.2200	5.2071	13.6657	0.2840	3.3983	7.4945	0.3475	5.4563	13.5233	0.2703	4.9636	13.5156	0.2909
	DS-PDc	2.4046	7.1799	0.1950	4.9122	13.8708	0.2674	3.3042	7.5735	0.3300	5.3888	13.6766	0.2644	4.6019	13.5037	0.2627
	DS-ScIn	2.3614	6.9439	0.1968	4.8481	13.5585	0.2650	3.3046	7.7271	0.3328	6.4323	14.8387	0.2620	4.7483	13.5524	0.2688
3D WaveUNet	WADS-DDc	2.2495	6.5712	0.1956	**4.5830**	**12.9024**	**0.2578**	3.1746	7.3695	**0.3287**	5.3404	13.5490	0.2645	4.3416	12.7909	0.2650
	WADS-DIn	2.0288	6.2569	0.1922	4.8316	13.3179	0.2668	3.2134	7.4323	0.3350	**5.2720**	**13.3878**	0.2644	4.2825	12.4185	0.2658
	WADS-DI	2.0676	6.2566	**0.1857**	4.8255	13.6180	0.2612	**3.1214**	**7.3349**	0.3358	5.3402	13.5475	**0.2605**	**4.0762**	**12.1576**	**0.2599**
	WADS-DIDn	**1.9973**	**6.0173**	0.1897	4.8720	13.5247	0.2675	3.2096	7.4646	0.3384	5.4028	13.5505	0.2638	4.2252	12.5162	0.2624



(a) neuronal cube. (b) natural image.

Fig. S.1. Comparison of neuronal image and natural image. (a) A neuronal cube (top) with its label matrix (bottom). (b) An example image (top) with its manual segmentation result (bottom).

References

Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision 111, 98–136.

Narayanaswamy, A., Wang, Y., Roysam, B., 2011. 3-d image preprocessing algorithms for improved automated tracing of neuronal arbors. Neuroinformatics 9, 219–231.

Peng, H., Ruan, Z., Long, F., Simpson, J.H., Myers, E.W., 2010. V3d enables real-time 3d visualization and quantitative analysis of large-scale biological image data sets. Nature biotechnology 28, 348.

Quan, T., Zhou, H., Li, J., Li, S., Li, A., Li, Y., Lv, X., Luo, Q., Gong, H., Zeng, S., 2016. Neurogps-tree: automatic reconstruction of large-scale neuronal populations with dense neurites. Nature methods 13, 51.

Xiao, H., Peng, H., 2013. App2: automatic tracing of 3d neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree. Bioinformatics 29, 1448–1454.

Zhou, Z., Liu, X., Long, B., Peng, H., 2016. Tremap: automatic 3d neuron reconstruction based on tracing, reverse mapping and assembling of 2d projections. Neuroinformatics 14, 41–50.