Supplementary information.

1.1 Description of datasets.

Datasets 1 and 2: Cells and chloroplasts were plunge frozen using a Vitrobot Mark IV (FEI Thermo Fisher) and stored in liquid nitrogen. Cryo-FIB milling was performed using a Quanta dual-beam FIB/SEM (*Chlamydomonas*, Dataset 2) or Aquilos dual-beam FIB/SEM (Spinach, Dataset 1) instrument (FEI, Thermo Fisher Scientific), as previously described in [24]. Tomographic data was acquired on a 300 kV Titan Krios microscope (FEI, Thermo Fisher Scientific), equipped with a post-column energy filter (Quantum, Gatan) and a direct detector camera (K2 Summit, Gatan). Dose-symmetric (Dataset 1) or bidirectional tilt-series (Dataset 2) were acquired with SerialEM software [31] using 2° steps between -60° and +60°. Individual tilts were recorded in movie mode with 12 frames per second, at an image pixel size of 3.42Å (42000x magnification). The total accumulated dose for the tilt-series was kept around 100 $e^{-}/Å^{2}$. The *Chlamydomonas* tilt-series (Dataset 2) [12] were acquired with a Volta Phase Plate at -0.5 µm defocus [30], while the spinach tilt-series (Dataset 1) were acquired with a standard objective aperture and defocus values from -2.5 to -4.5 µm. Using IMOD software [32], tilt-series were aligned with patch tracking and reconstructed with weighted back projection. Additionally, Dataset 1 was dose weighted with TOMOMAN software [33] and denoised with cryo-CARE [25].

Dataset 3: Rod outer segments were isolated, vitrified on EM grids, thinned by cryo-FIB milling, and imaged by cryo-ET as described in [26]. Tilt-series were acquired on the same Titan Krios microscope as Datasets 1 and 2. Bidirectional tilt-series were collected between +50° and -50°, starting at 20° with a tilt increment of 2° and a total dose of 100 $e^{-}/Å^{2}$. The individual projection images were recorded at an image pixel size of 2.62 Å (53000x magnification) in movie mode with 6 frames per tilt at tilt angle α = 0°. The number of frames per tilt was increased at higher tilts according to the 1/cos(α) as implemented in SerialEM. Standard objective aperture imaging was used, with a target defocus of -3 µm. The acquired tilt-series were aligned using Platinum particles on the lamella surface as fiducial markers [26] and reconstructed in IMOD software. Furthermore, dose weighting was performed with TOMOMAN software and denoising was performed with cryo-CARE, similar to Dataset 1.

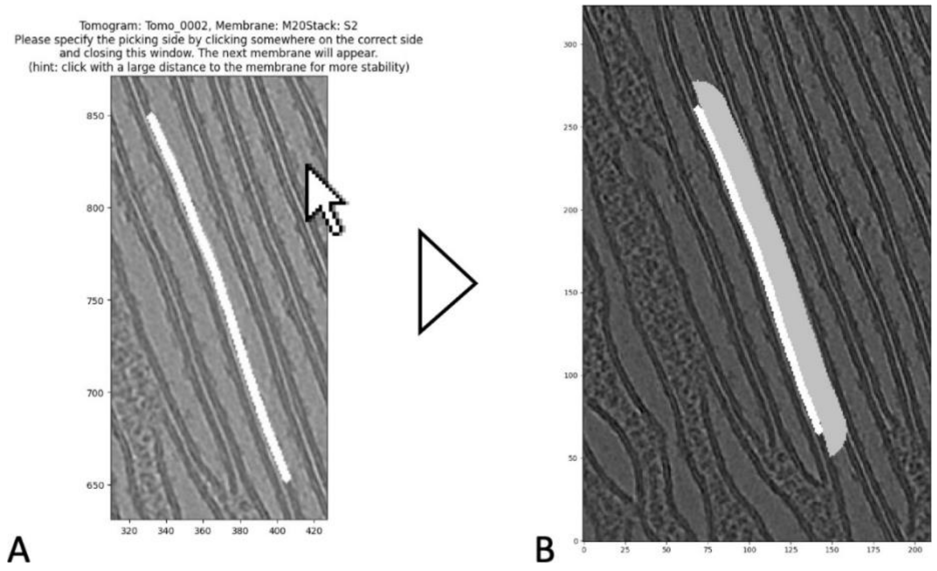
Tomogram: Tomo_0002, Membrane: M20Stack: S2
Please specify the picking side by clicking somewhere on the correct side
and closing this window. The next membrane will appear.
(hint: click with a large distance to the membrane for more stability)

Fig. S1. Graphical assistance provided by the MemBrain program. White segmentations represent given membrane segmentations. **A:** Membrane selection interface. The user should click on the side of the membrane where they want to detect proteins. **B:** Membrane segmentation and resulting selected side of the membrane (grey: thresholded neighborhood used to estimate normal vectors), visualized using MemBrain's membrane side inspection script, which can be used to verify that the correct sides were picked.

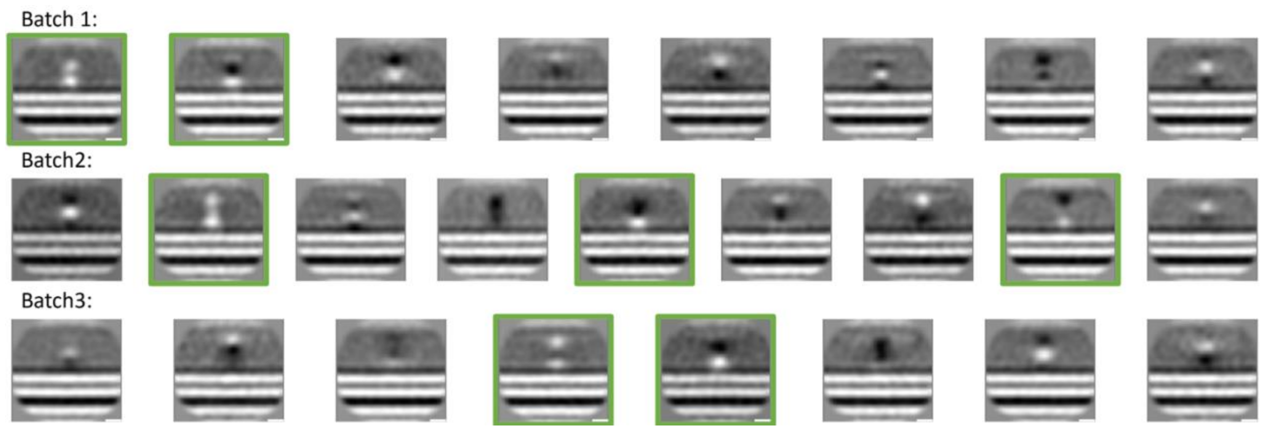

Fig. S2. Selection of 2D rotational average classes when using PySeg on Dataset 3. 270k particles were extracted, split into batches of 30k particles each, and clustered into several classes using PySeg's 2D rotational average clustering. From the resulting classes, those with a density in contact with the membrane were selected (green).

1.2 PySeg protein picking

Protein picking in PySeg was performed using the default parameters. The only exception was that the vertex density during the calculation of the graphs was set to 0.01 per nm, a value much higher than expected, to assure that all possible densities were picked. The minimum allowed distance between adjacent proteins was set to 4 nm. Refinement of the membrane normal vectors and the alignment of the membranes was done in STOPGAP (v.0.7.1) [29]. The star-file output by PySeg protein picking was converted into a STOPGAP motive list, subvolumes were extracted from 4x binned non-CTF-corrected tomograms (box size = 64 voxel, voxel size = 1.048 nm) and aligned with shifts only allowed perpendicular to the membrane. Subvolumes were re-extracted at the refined positions with a smaller box size of 32 voxels, and the refined motive list was converted into a star-file for PySeg. Afterwards, PySeg's plane_align_class.py script was used for two successive rounds of 2D classification (Affinity propagation, preference = -10, pp 3D = False). Here, a mollified cylinder mask (radius = 10 voxel, height = 10 voxel) focused the classification on the cytosol adjacent to the membrane without including the membrane signal itself. In the first round (referred to as $\text{PySeg}_{\text{clean}}$ in the manuscript), all picked proteins were processed in batches of 30k subvolumes, and those classes that indicated a density in contact with the membrane irrespective of its size were selected (see Figure S2). In a second round, all selected proteins were classified together and only those classes with reasonably large density were selected (these picks were not used for the evaluation in this paper).

1.3 Configurations of comparison methods

DeepFinder:

Ground truth protein positions and orientations were used to map a thresholded volumetric model of PSII into the tomograms. This gave us an accurate segmentation mask, where, for all training protein positions, all voxels that were occupied by a protein density were labeled as 1. The remaining voxels were labeled as background (i.e., 0). We trained DeepFinder using the same data split as we did for MemBrain (28 membranes training set, 7 validation set). In addition to DeepFinder's default settings, we also performed hyper-parameter tuning for learning rate and random shifts range, and found the best performance on the validation set for a learning rate of 1×10^{-4}, and random shifts in the range of 11.
As preprocessing for our tomograms, we used the default preprocessing suggested in EMAN: low- and high-pass filtering, followed by normalization (to mean 0 with standard deviation 1) of the tomogram. Finally, values were clamped between -3 and 3 to remove outliers.

EMAN (v.2.91):

We used the same ground truth segmentation masks from DeepFinder to generate the ground truth for EMAN.

Because EMAN requires annotated 2D slices, we extracted 2D slices (size 64 x 64) from the 3D annotated volumes, each centered around a protein position to ensure that the patches contain proteins. We performed the default preprocessing (as described in the DeepFinder section), and also did hyperparameter tuning. EMAN's architecture is very shallow, which can avoid overfitting on the training data. Nevertheless, we wanted to rely on more information than only the training loss. Therefore, we trained several models to the end, and then performed tuning of the extraction parameters on the validation set to find the best models and setting: we trained models for varying numbers of epochs and different learning rates. Then, for each model, we tuned the protein extraction parameters for the density threshold, and the mass threshold (size of an object to be considered a true object). Here, we found the best model with a learning rate of 1×10^{-5}, trained for 5000 epochs. For the extraction, we used a mass threshold of 2.0 and a density threshold of 0.5.

crYOLO (v.1.8.0):
Instead of volumetric segmentation masks, crYOLO requires only the positions of the ground truth particles. Therefore, we converted our manually picked positions into the required format. crYOLO requires the user to label a protein in a 2D slice, even if only a small portion of it can be seen. Therefore, we experimented with transferring ground truth positions also to adjacent slices, but did not find an improvement in performance. Again, we performed hyper-parameter tuning. For training the neural network, we tuned the parameter "object_scale", determining the weight of true positive samples in the loss function (compared to the weight of false positives). Furthermore, we tuned parameters for the extraction of protein positions: the threshold for the network-assigned scores, the minimum distance between boxes, and the minimum number of boxes in one trace to be considered a protein. The best settings we found were an object scale of 50 (very high, probably necessary due to the sparse annotations), a score threshold of 0.05, a minimum distance of 5, and a minimum number of boxes in a trace of 2.

Template matching:
Template matching experiments were performed in STOPGAP [29]. A previously obtained average of PSII [12] was used as the template for detection of this protein in Datasets 1 and 2 using an angular search step of 20 degrees in bin4 tomograms (pixel size: 13.68 Å). Wedge lists containing information on electron exposure, defocus and tilt angle were obtained from the IMOD reconstruction metadata [32] for each tomogram. This information was used to properly weight the Fourier transform of the template rotated in each orientation and constrain the cross-correlation only to the areas where information is present. To avoid overfitting, the template was low-pass filtered to ~40 Å during search. For performance analysis, we visually tuned the number of extracted picks, as well as the minimum distance between two picks. We arrived at values of 400 picks per tomogram, and a minimum distance of 13 voxels, giving the best performance on the test tomograms.

1.4 Subtomogram averaging for Dataset 3.

To create the averages in Figure 5 and Figure S4 we utilized the following procedure: We extracted protein positions using MemBrain and compared them to the ones obtained by PySeg (see section S.1.2). We subdivided the picks into nine subsets, as can be seen in Table S1 For subsets 1, 3, 4, and 5, we used the normal voting procedure described in Section 2.2 to receive membrane normal vectors for each position. Then, from the normal vectors, we computed the Euler angles that describe the rotation to align the membrane horizontally. For subsets 2 and 6, we used the Euler angles that were computed by PySeg. Then, we used STOPGAP [29] to create averages of the picked positions in bin4 (pixel size: 10.48Å). First, we created an initial average. Subsequently, we refined this average in 3 rounds of subtomogram alignment, with the goal of refining the membrane alignment. For this, we used a cylindrical mask to focus the alignment only on the current membrane and the density on top of it. We allowed only small shifts in x-, y, and z-direction (2 voxels in each direction, bin4) and used a strong symmetry constraint (C20). For subset 6, we further performed subtomogram classification into 3 classes (classes 7, 8, 9) using STOPGAP's "ali_multiref" function. We performed this classification step for 10 rounds in order to investigate if there is a class corresponding mainly to membrane-embedded proteins.

Subset name	**Description**	**# positions**
1 $\text{MemBrain}_{\text{rod\_outer}}$	All MemBrain picks	27906
2 $\text{PySeg}_{\text{clean}}$	All PySeg picks after round 1	65614
3 $\text{MemBrain} \setminus \text{PySeg}_{\text{all}}$	MemBrain picks, missed by PySeg before round 1	2163
4 $\text{MemBrain} \setminus \text{PySeg}_{\text{clean}}$	MemBrain picks, missed by PySeg after round 1	15899
5 $\text{MemBrain} \cap \text{PySeg}_{\text{clean}}$	MemBrain picks, coinciding with PySeg picks after round 1	12007
6 $\text{PySeg}_{\text{clean}} \setminus \text{MemBrain}$	PySeg picks after round 1, missed by MemBrain	49562
7 $\text{PySeg}_{\text{clean}} \setminus \text{MemBrain}$ Class 1	Classified using subtomogram classification in STOPGAP	9665
8 $\text{PySeg}_{\text{clean}} \setminus \text{MemBrain}$ Class 2	Classified using subtomogram classification in STOPGAP	21678
9 $\text{PySeg}_{\text{clean}} \setminus \text{MemBrain}$ Class 3	Classified using subtomogram classification in STOPGAP	18219

Table S1. The number of particles in different subsets of Dataset 3 picks used for averaging comparison. “Round 1” corresponds to the first PySeg pick cleaning step, where only averages containing membranes with densities were selected.

Layer	Type	# Filters	Size
1	Conv (valid)	32	3x3x3
	BN		
	ReLU		
2	Conv (same)	32	3x3x3
	BN		
	ReLU		
3	Conv (valid)	32	3x3x3
	BN		
	ReLU		
4	Conv (same)	32	3x3x3
	BN		
	ReLU		
5	Linear	1	(box_size – 4)^3 * 32

Table S2. Overview of our MemBrain network architecture. Conv=Convolutional layer; BN=Batch normalization

Method	**Required inputs**	**Preprocessing time**	**Training time**	**Inference time**	**Particle extraction time**
MemBrain 1mb	Particle positions (training) Membrane segmentations (training + inference)	2-5 min / mb	~20 min	1.5s / mb*	1.5s / mb*
MemBrain 8mb		2-5 min / mb	~30 min	1.5s / mb*	1.5s / mb*
MemBrain 28mb		2-5 min / mb	~2h	1.5s / mb*	1.5s / mb*
DeepFinder	Particle positions (training)	**	~2h	5 min / tomo	1-2h / tomo
CrYOLO	Particle positions (training)	**	4-5h	1 min / tomo	
EMAN2	Particle positions (training)	**	1h 20min	15 min / tomo	1 min / tomo
Template matching (STOPGAP)	Template structure	-	-	45 min / tomo***	<1 min/tomo***

Table S3. Comparison of required preprocessing, training, inference and extraction times. Training and prediction of neural networks was performed using a Nvidia Quadro RTX 5000 GPU. Preprocessing time for MemBrain includes sampling of points, extraction of subvolumes and normalization of subvolumes. Inference time for all methods is the time needed to apply the trained network. Particle extraction time gives the time required for extracting the final particle positions from the neural network output. Note that crYOLO already outputs particle positions.
For the required inputs, we did not give any times, as it is very difficult to estimate them: they highly depend on which software is used, how accurate it is, and how skilled the manual annotator is.
* Time depends largely on the density of sampled points on the membranes. The times for standard settings are shown.
** DeepFinder, CrYOLO and EMAN2 don't require preprocessing by default. However, it is recommended to normalize tomograms and optionally also band-pass filter them.
*** Timing provided for the following settings: template size 16 x 16 x 16, tomogram size 928 x 928 x 464, angular search step 20 degrees (number of orientations: 2052), CPU cores: 80. Extraction time depends on CC threshold selected (here: 0.15) and minimum distance between points (here: 4 pixels).

Method	F1 Score 1 mb	8 mbs	28 mbs
Default MemBrain	0.88	**0.91**	**0.92**
More augmentation angles	0.88	0.92	**0.91**
No augmentation	0.89	0.89	0.92
Volume size 8	0.88	0.90	0.90
Volume size 16	0.83	0.91	0.90
Uniform Sampling	0.87	0.90	0.90

Table S4. Extended ablation study. We tested our default MemBrain settings against several other design choices. More augmentation angles: Instead of rotating our subvolumes only for 90°, we allow for all angles (computationally more expensive). No augmentation: No augmentation (rotation, flipping or random noise) was used. Note that, although results are comparable to the default model, training is more stable when using augmentation; see Figure S5. Volume size 8 / Volume size 16: Instead of the default subvolume size of 12, we used sizes of 8 and 16. Uniform Sampling: In order to account for unbalances in our dataset with respect to GT distances, we split all distances into evenly sized bins, and sampled subvolumes from all bins with equal frequency during training.

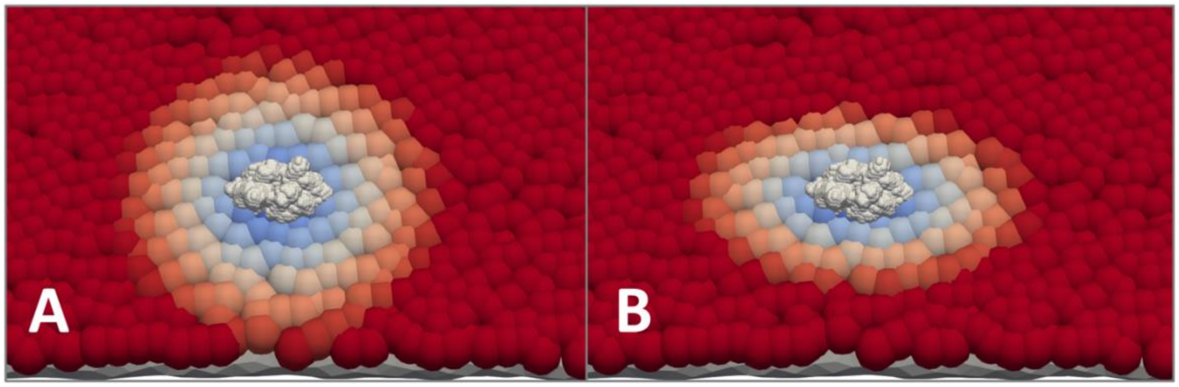

Fig. S3. Visualization of the score assignment using Euclidean vs. Mahalanobis distance. The white density represents a PSII particle, mapped into its ground truth position and orientation. Distance values are represented by colors: blue points correspond to points with low distances. Red points have a higher assigned distance. **A:** Visualization of the Euclidean distance. **B:** Visualization of the Mahalanobis distance.

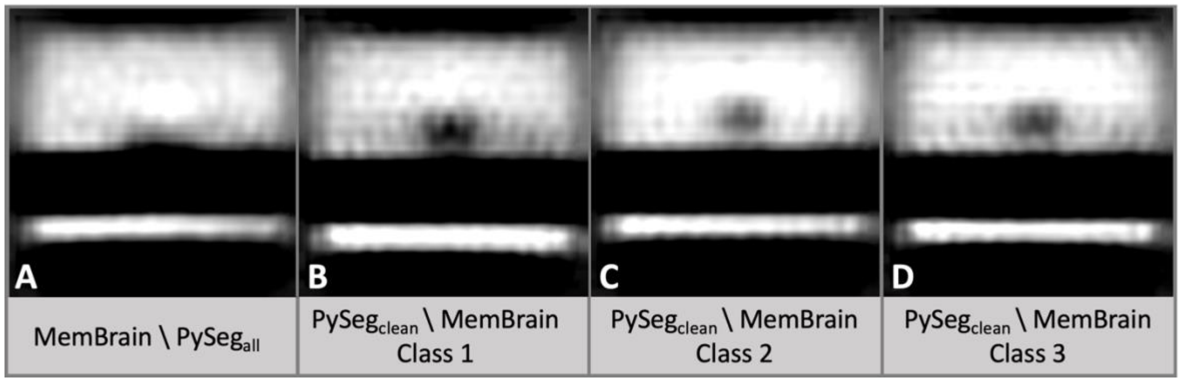
A
B
C
D
MemBrain \ PySeg$_{all}$
PySeg$_{clean}$ \ MemBrain Class 1
PySeg$_{clean}$ \ MemBrain Class 2
PySeg$_{clean}$ \ MemBrain Class 3

Fig. S4. Rotationally symmetrized subtomogram averages generated from different subsets of picked positions from Dataset 3. **A:** MemBrain positions that were missed by PySeg (before cleaning steps). **B-D:** Subtomogram classification results for subdividing (PySeg$_{clean}$ \ MemBrain) into 3 classes. For more details on how the averages were created, see S.1.4. and Table S1.

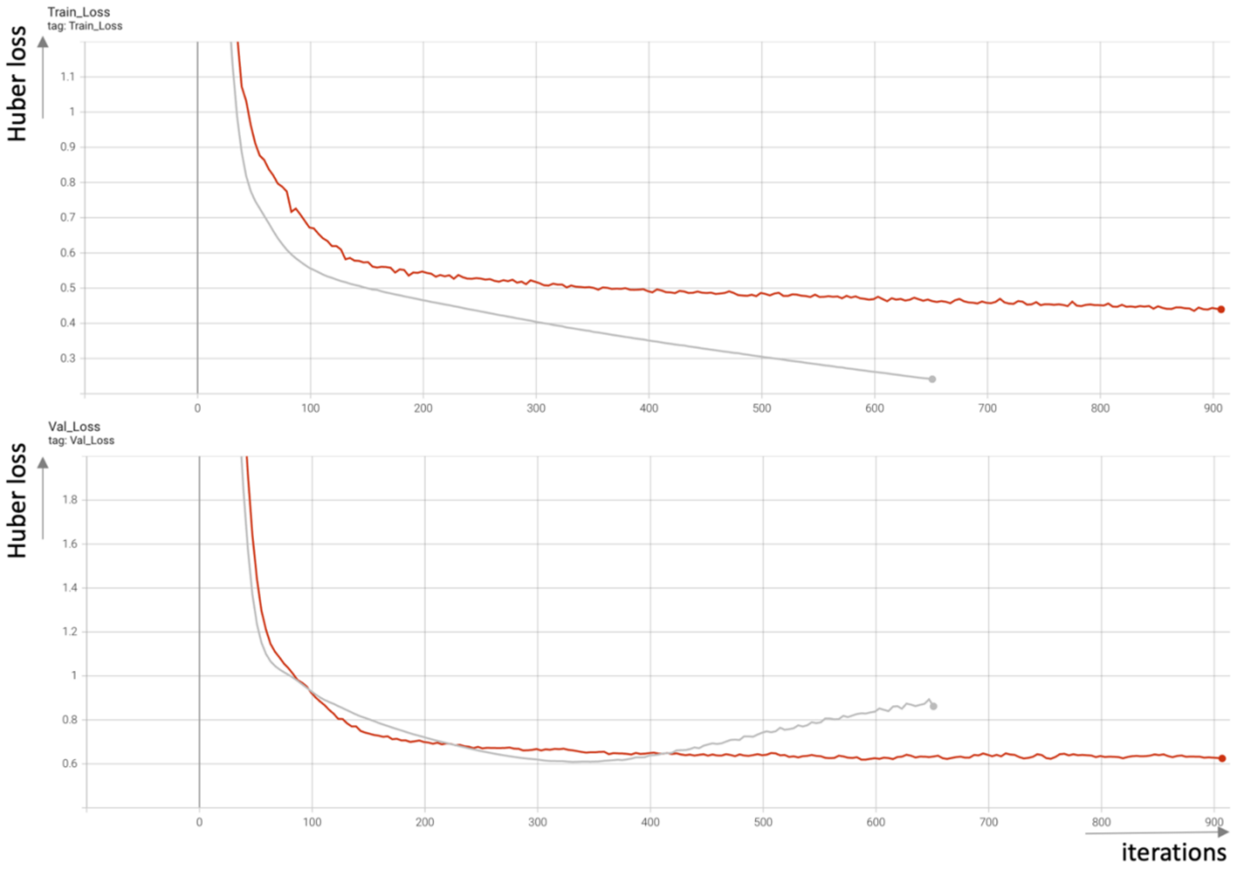

Fig. S5. Training and validation curves of example training runs using one single membrane for training. The red curve corresponds to a training run with data augmentation, the gray curve is a training run without data augmentation. From iteration ~330 on, the gray curve starts indicating overfitting, as the validation loss increases while the training loss further decreases.

[31] D.N. Mastronarde, Automated electron microscope tomography using robust prediction of specimen movements, J. Struct. Biol. 152 (2005) 36–51.

[32] D.N. Mastronarde, S.R. Held, Automated tilt series alignment and tomographic reconstruction in IMOD, J. Struct. Biol. 197 (2017) 102–113.

[33] W. Wan, williamnwan/TOMOMAN: TOMOMAN 08042020, (2020). https://doi.org/10.5281/zenodo.4110737.