nature methods

Article

https://doi.org/10.1038/s41592-026-03178-8

MemBrain v2: an end-to-end tool for the analysis of membranes in cryo-electron tomography

Received: 22 April 2025

Accepted: 25 June 2026

Published online: 08 September 2026

Check for updates

Lorenz Lamm[1,2,3] ✉, Simon Zufferey[2], Hanyi Zhang[1,3], Ricardo D. Righetto[2], Florent Waltz[2], Wojciech Wietrzynski[2], Kevin A. Yamauchi[4,5], Alister Burt[6,7], Ye Liu[1,3], Antonio Martinez-Sanchez[8], Sebastian Ziegler[9,10], Fabian Isensee[9,10], Julia A. Schnabel[1,3,11,12], Benjamin D. Engel[2] ✉ & Tingying Peng[1,3] ✉

Cryo-electron tomography provides unique insights into macromolecular complexes in their native environments, yet membrane analysis remains a major bottleneck due to low signal-to-noise ratios, missing wedge artifacts and the complexity of membrane-associated particles. Existing tools often require extensive manual annotation, struggle with generalization across datasets and lack integrated solutions for segmentation, particle localization and quantitative analysis. We introduce MemBrain v2, a deep-learning-enabled framework that unifies these tasks into a streamlined pipeline. MemBrain-seg leverages a diverse, collaboratively generated training dataset and specialized model training strategies to achieve generalizable membrane segmentation across variable tomographic conditions. MemBrain-pick enables data-efficient localization of membrane-bound particles by integrating geometric constraints with deep learning, reducing the need for extensive manual annotation. MemBrain-stats provides quantitative insights into particle distributions, computing spatial metrics to analyze intramembrane particle organization. MemBrain v2 integrates seamlessly into cryo-electron tomography workflows, providing an accessible and structured approach to membrane analysis.

Cryo-electron tomography (cryo-ET) is a powerful technique for imaging the molecular environment inside native cells in three dimensions at sub-nanometer resolution[1]. By capturing all densities inside a cellular volume frozen in vitreous ice, cryo-ET provides detailed insights into the structures, interactions and spatial arrangements of diverse organellar and macromolecular components. However, the biological complexity captured by cryo-ET also presents substantial challenges for annotation and analysis[2]. Membranes and their embedded protein complexes play fundamental roles in numerous cellular processes, including the endoplasmic reticulum–Golgi secretory pathway[3–5], autophagy[6,7], vesicular transport[8], synaptic transmission[9–12], energy conversion in mitochondrial cristae and chloroplast thylakoids[13–22], organelle contact sites[23,24], cell membrane protrusions[25,26], viral replication[27–33] and bacterial cell division[34,35]. Accurate segmentation of membranes and precise localization of membrane-associated particles are essential for understanding how membrane architecture and molecular organization drive cellular functions.

Membrane analysis in cryo-ET is challenging due to the inherently low signal-to-noise ratio, which obscures structural details and complicates accurate interpretation. Additionally, the missing wedge effect—caused by limited angular sampling—introduces anisotropic distortions that particularly affect membrane structures oriented

A full list of affiliations appears at the end of the paper. ✉e-mail: lorenz.lamm@tum.de; ben.engel@unibas.ch; tingying.peng@helmholtz-munich.de

https://doi.org/10.1038/s41592-026-03178-8

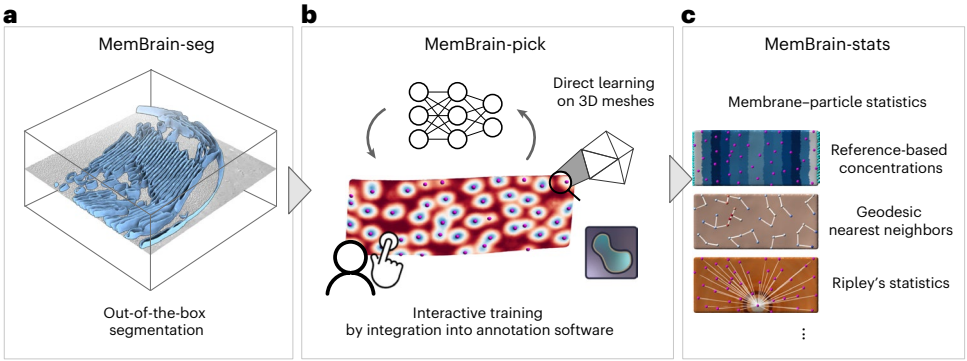


Fig. 1 | MemBrain v2 provides an end-to-end pipeline for analyzing membranes and membrane-associated particles in cryo-ET data. a, MemBrain-seg produces generalizable membrane segmentations out-of-the-box. **b**, MemBrain-pick efficiently localizes membrane-associated particles by iterative annotation and training directly on mesh surfaces. **c**, MemBrain-stats evaluates the outputs of MemBrain-seg and MemBrain-pick, providing particle-surface metrics.

perpendicular to the electron beam, making them difficult to resolve. While computational approaches for denoising[36–39] and missing wedge correction[40–42] attempt to mitigate these issues, their restorations are not always reliable and may introduce artifacts.

Several methods attempt to tackle membrane segmentation in cryo-ET, yet each comes with inherent limitations. Classically, TomoSegMemTV[43] incorporates local membrane curvature via tensor voting, but can struggle in areas with complex or rapidly varying membrane shapes. Deep-learning-based approaches, particularly U-Net-based[44–47] models, have driven major advancements. However, many implementations lack generalizability, because they have been trained on only a single cell or membrane type. Therefore, specialized models are often trained for single projects or datasets[48]. TARDIS[49] represents a promising step toward more generalizable segmentation by incorporating diverse datasets into training and offering pretrained models, making segmentation tools more accessible. Despite these advancements, it remains an open challenge to develop a widely applicable approach that ensures robust segmentation across diverse membrane architectures, tomographic conditions and biological contexts.

Beyond membrane segmentation, the automated localization of membrane-associated particles (for example, integral membrane proteins) in cryo-ET remains a major challenge. Because these particles are embedded in the lipid bilayer, they are difficult to distinguish from the surrounding membrane. Classical template-matching approaches[50–52] often fail in this context, as strong membrane contrast dominates cross-correlation scores, reducing detection accuracy. Deep-learning methods offer an alternative, with several convolutional neural network-based approaches recently introduced for particle localization in cryo-ET[53–58]. However, many of these heavily rely on extensively annotated volumes, which are labor intensive and often not technically feasible to generate. These models also struggle with sparsely annotated membranes, limiting their generalizability. The high variability in particle appearances further complicates training of generalist models, making each individual membrane annotation particularly valuable and motivating the need for specialized models that can operate with individual membrane annotations. To streamline this annotation process, MPicker[59] flattens membranes into two-dimensional (2D) stacks, but this transformation can introduce distortions and fail to preserve physical distances. To address these issues, annotation tools such as membranorama[17,60] and Surforama[61] project tomographic densities onto three-dimensional (3D) meshes, allowing for interactive exploration and particle localization. Among existing tools, MemBrain (v1)[62] was designed to work with limited annotations, but its performance remains constrained by its small receptive field. A more refined, interactive approach could better leverage sparse annotations to improve both the accuracy and efficiency of particle localization.

Quantitative tools for analysis of membrane segmentations include PyCurv[63], which estimates membrane curvature, and the Surface Morphometrics Pipeline[13], which extracts properties such as intermembrane distances. PyOrg[64] analyzes particle distributions but does not directly relate them to membrane features. Thus, these tools analyze either membrane geometry or particle distributions separately, lacking an integrated approach to quantitatively assess particle–membrane interactions.

To address the challenges described above, MemBrain v2 unifies cryo-ET membrane analysis in a single integrated framework: MemBrain-seg produces generalizable membrane segmentations using a collaboratively generated, diverse ground-truth (GT) dataset. MemBrain-pick efficiently localizes membrane-bound particles with interactive annotation support via Napari and Surforama. MemBrain-stats computes descriptive statistics such as particle concentrations and geodesic nearest-neighbor distances along the segmented membrane. Together, these modules enable end-to-end analysis of membranes from segmentation to particle localization and statistical analysis (Fig. 1). By combining versatility with usability, MemBrain v2 provides an intuitive solution that can be applied to diverse data sources to explore a broad spectrum of biological questions.

Results

Overview of MemBrain v2 modules

MemBrain v2 is a modular pipeline designed to streamline the analysis of membranes and their associated particles in cryo-ET datasets. Its three core modules—MemBrain-seg, MemBrain-pick and MemBrain-stats—work together to enable membrane segmentation, particle localization and quantification. MemBrain-seg (Fig. 1a) uses a U-Net-based approach to achieve robust membrane segmentation across a variety of experimental and tomography setups. It is trained on a diverse, iteratively refined dataset, which was generated using careful manual annotations and corrections in close collaboration with the community to ensure broad coverage of different membrane appearances. Together with our cryo-ET-specific data augmentations and membrane-focused loss functions in the network training, this approach ensures accurate and continuous membrane delineation, facilitating visualization and downstream analysis. MemBrain-pick (Fig. 1b) specializes in the efficient localization of membrane-bound particles (for example, membrane proteins). By training a neural network to operate directly on membrane surfaces, it incorporates the membrane geometry into its prediction and thus reduces the search space, enhancing both accuracy and data efficiency. Its integration with interactive Napari[65] tools, such as Surforama[61], allows for rapid annotation and refinement of particle positions, facilitating seamless transitions between GT generation and model training. MemBrain-stats (Fig. 1c) leverages the outputs of MemBrain-seg and MemBrain-pick to

provide quantitative insights into particle distributions on membranes. It computes key metrics such as particle concentrations, geodesic nearest-neighbor distances and Ripley's statistics. By linking membrane morphology with particle organization, MemBrain-stats enables investigation of the structural and functional relationships in cryo-ET datasets. MemBrain v2's modular design ensures both flexibility and accessibility, featuring straightforward command-line interfaces and seamless integration with Napari plugins for interactive annotation and visualization.

MemBrain-seg: a generalized approach for membrane segmentation

MemBrain-seg is a U-Net-based program that generates 3D membrane segmentations from input tomograms with a single command (Fig. 2a and Extended Data Fig. 1). It delivers robust segmentation performance across diverse tomograms containing a variety of organelles from various species, acquired with different microscope types, imaging settings and processing conditions, ranging from raw to denoised tomograms (Fig. 2e and Extended Data Fig. 2). While strong segmentation performance on thylakoid membranes was expected due to the high representation of spinach chloroplasts in our training data (Supplementary Fig. 1 and Supplementary Table 1), MemBrain-seg also achieves satisfactory results on vesicular structures and convoluted mitochondrial membranes. This adaptability is further demonstrated in the CZII data portal[66] (Supplementary Fig. 2), where MemBrain-seg was applied to all tomograms present in the portal, allowing users to conveniently inspect segmentations together with the corresponding tomograms in the browser via Neuroglancer[67].

To ensure consistent and robust model performance, MemBrain-seg was trained on a diverse dataset, iteratively refined through manual corrections (Supplementary Fig. 1). In each training round, segmentation predictions were reviewed and corrected to improve the dataset for subsequent iterations (Fig. 2b). The first two rounds (1 and 2) focused on training patches from our *Spinacia oleracea* ('Spinach', EMPIAR-12612) and *Chlamydomonas reinhardtii* ('Chlamy', EMPIAR-11830) datasets, followed by contributions from external collaborators ('Collaborators') to introduce additional diversity. To further enhance robustness, synthetic data from publicly available generators[39,68] ('Synthetic') and patches from the DeePiCt dataset[45] ('DeePiCt'; EMPIAR-10988) were integrated in the last two training rounds (4 and 5, respectively). Expanding the dataset incrementally across different tomography sources further enhanced the model's generalization, as demonstrated by its progressively improving performance across datasets with each additional training round (Extended Data Fig. 3). Here, we monitored performance by calculating the Dice score (a standard metric for segmentation accuracy) for different test datasets. The improved generalization is particularly evident in the 'DeePiCt' test dataset, where Dice scores improved from 39% to 66% (and up to 55% in rounds without any DeePiCt training data).

In addition to evaluating network performance across different training datasets, we also compared MemBrain-seg to DeePiCt[45], Ais[48] and TARDIS[49] (visual comparison in Extended Data Fig. 4); all programs providing pretrained models for membrane segmentation. In our test datasets, MemBrain-seg achieved the highest Dice scores compared to all methods, followed by TARDIS with the second-highest scores (Extended Data Fig. 3c). However, this comparison should be interpreted carefully because our GT annotations were generated by iteratively predicting with MemBrain-seg and manually correcting these predictions. As a result, a substantial portion of the GT voxels still originates directly from MemBrain-seg outputs, and may inherently favor its performance. To address this issue, we additionally assessed performance using fully synthetic datasets with absolute GT annotations, providing a more unbiased, but also less realistic, benchmark. Here, both MemBrain-seg and TARDIS achieved good values (66% in round 3 versus 59% Dice, respectively), hinting at strong generalization capabilities for both methods. To further mitigate potential bias—particularly differences in membrane thickness annotations, which may favor MemBrain-seg's predicted segmentation thickness—we introduced the Surface-Dice score (Fig. 2c). Unlike standard Dice scores, which compare segmentations at the voxel level, Surface-Dice evaluates the structural consistency of predicted and GT segmentations by comparing their skeletons (Supplementary Fig. 3). This metric better captures membrane continuity and is invariant to annotation style (for example, Supplementary Fig. 4). Evaluations with Surface-Dice were consistent with the standard Dice assessments (Extended Data Fig. 3a). By combining Dice and Surface-Dice metrics with our diverse dataset, we provide a publicly available benchmarking resource for membrane segmentation tools.

In addition to our base model, we provide versions trained with enhanced augmentations. These Fourier-based augmentations (Fig. 2d and Supplementary Fig. 5b,c) simulate tomographic style variations and missing wedge distortions, improving model robustness. Specifically, our missing wedge augmentation artificially introduces an additional missing wedge in training patches to mimic real-world artifacts, while Fourier amplitude augmentation randomly rescales Fourier frequency bands to simulate tomographic variability. These enhancements improved generalization across diverse tomographic conditions (Extended Data Fig. 3b). Notably, for round 4 data (that is, excluding DeePiCt training data), we evaluated models trained with either Fourier amplitude or missing wedge augmentation. Fourier amplitude increased Surface-Dice scores on the DeePiCt test set from 55% to 59%, while missing wedge augmentation improved scores to 57%, demonstrating their effectiveness in adapting to unseen datasets. We provide example effects of using these models in Supplementary Fig. 6.

In addition to augmentations during training, preprocessing tomograms before inference can also influence segmentation quality. In Supplementary Fig. 7, we assess how common denoising and missing wedge correction methods[36,40–42,69] affect MemBrain-seg's performance. Despite substantial differences among these preprocessing strategies, MemBrain-seg delivers consistent performance, highlighting its robustness to different tomogram appearances. However, in cases with strong missing wedge artifacts, like horizontal membranes or spherical vesicles, missing wedge correction may provide a notable improvement, as demonstrated for the membranes and vesicles of a mammalian cell in EMPIAR-13055 (ref. 70; Supplementary Fig. 7, center row).

Despite this general robustness, MemBrain-seg's performance may degrade when applied to datasets with substantial domain shifts, such as differences in microscopy conditions or sample preparation. In such cases, fine-tuning the model on a specific dataset (for example, with patches generated as shown in Supplementary Fig. 1a) can further optimize segmentation quality. We demonstrate this capability in Supplementary Fig. 8, where fine-tuning with only DeePiCt dataset patches in the training set improved Surface-Dice scores from 58% to 67%. To prevent overfitting, we continuously monitored performance on the full validation set depicting a broader range of membranes than the test set, using both the Dice score and Surface-Dice score. Notably, monitoring Surface-Dice during fine-tuning outperformed monitoring only Dice scores (67% versus 60%, respectively), avoiding premature early stopping due to the different membrane thicknesses in the fine-tuning dataset (Supplementary Fig. 4) and highlighting the effective adaptation of MemBrain-seg to domain shifts. A qualitative example of this fine-tuning process is provided in Extended Data Fig. 5, where the original MemBrain-seg model initially segmented the outer surface layer of *Thermoanaerobacter kivui* cells as membrane. Manual correction of several training patches followed by fine-tuning eliminated these false positives, resulting in a more focused segmentation. To guide users in selecting the most effective 3D patches for manual annotation, MemBrain-seg can also predict a

https://doi.org/10.1038/s41592-026-03178-8

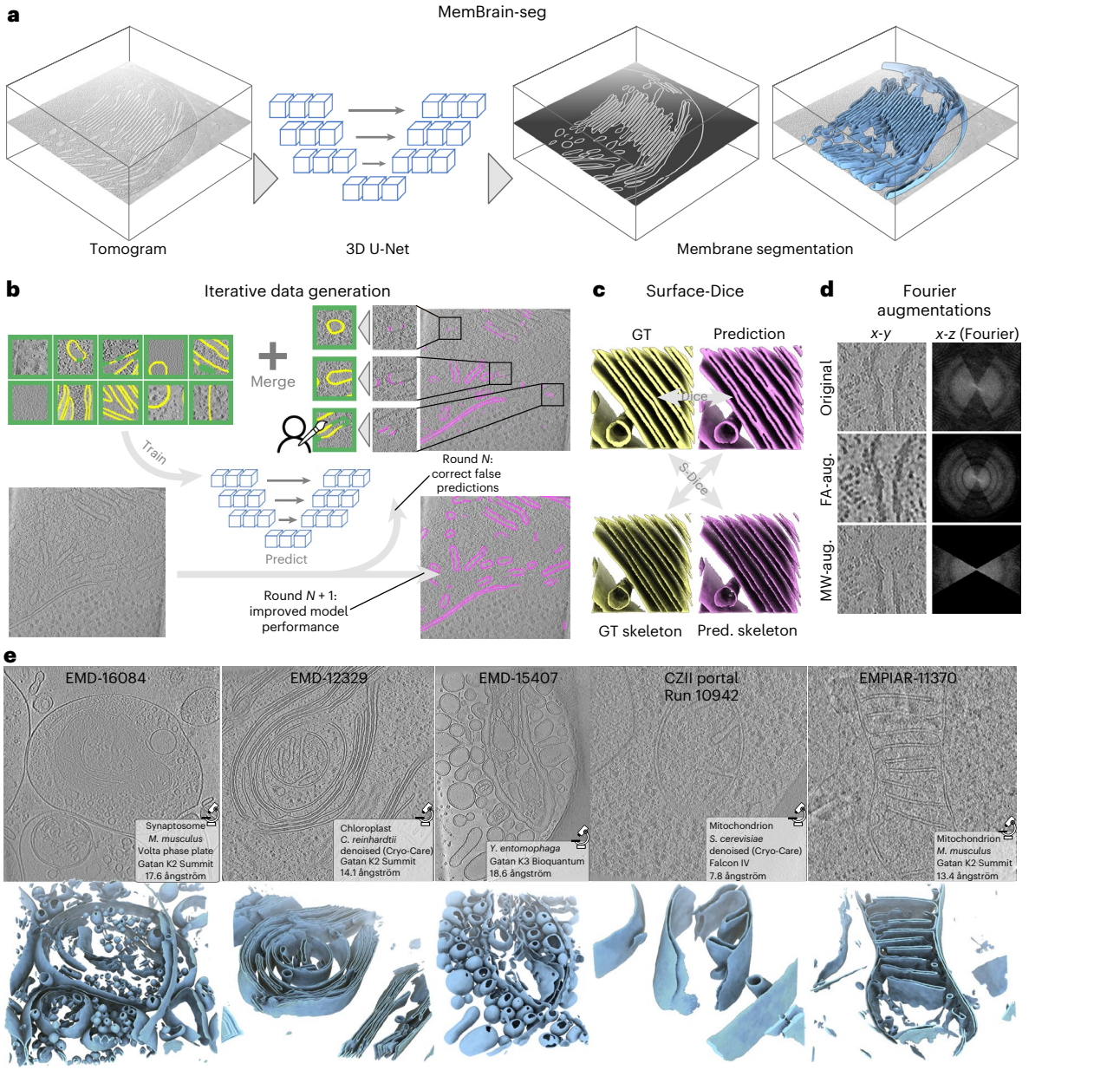


Fig. 2 | MemBrain-seg achieves accurate and generalizable membrane segmentation across diverse cryo-ET datasets through iterative active learning, membrane-specific loss function and Fourier-based augmentation. **a**, MemBrain-seg is based on a U-Net architecture and produces accurate 3D segmentations from input tomograms. **b**, Our diverse training dataset was generated iteratively via an active learning approach: For a model in training round *N*, we extracted patches in difficult regions, manually corrected the network prediction using membrane, background and ignore labels (Supplementary Fig. 1), and merged them into our training dataset. The next round (*N* + 1) shows improved segmentation performance. **c**, Surface-Dice evaluates segmentation quality by comparing the skeletons (Supplementary Fig. 11) of the predicted segmentation with the GT segmentation (yellow), and the GT skeleton with the predicted segmentation (pink). This emphasizes membrane continuity more efficiently than the conventional Dice metric, which directly compares voxel-wise overlap. **d**, MemBrain-seg's Fourier-based augmentations: Fourier amplitude (FA) augmentation randomly rescales intensities of Fourier frequency bands, altering image contrast. The missing wedge (MW) augmentation randomly applies an artificial missing wedge in Fourier space to imitate the effects of the real missing wedge. **e**, Examples of MemBrain-seg predictions on diverse public datasets, demonstrating robust out-of-the-box segmentation performance across a wide range of membrane architectures in tomograms acquired with diverse instrumentation. The top row shows slices through tomograms, and the bottom row shows corresponding MemBrain-seg predictions (light blue). For more examples, see Extended Data Fig. 2.

membrane uncertainty map (examples in Extended Data Fig. 6), which highlights areas where the model's predictions are less confident and may benefit from refinement.

MemBrain-pick: an interactive tool to efficiently localize membrane-associated particles

Identification of membrane-associated particles (for example, integral membrane proteins or membrane-bound ribosomes) in cryo-ET data is a challenging task. In MemBrain-pick, we aim to facilitate this process by enabling a smooth transition from MemBrain-seg outputs to the area of interest for particle localization and training an automated model. With our MemBrain Napari plugin and its integrated 3D lasso functionality, users can isolate single-membrane instances from full-tomogram segmentations (Fig. 3a and Extended Data Fig. 1b), which can subsequently be visualized and annotated in Surforama[61,65] (Fig. 3b) for manual annotation of particle locations.

https://doi.org/10.1038/s41592-026-03178-8

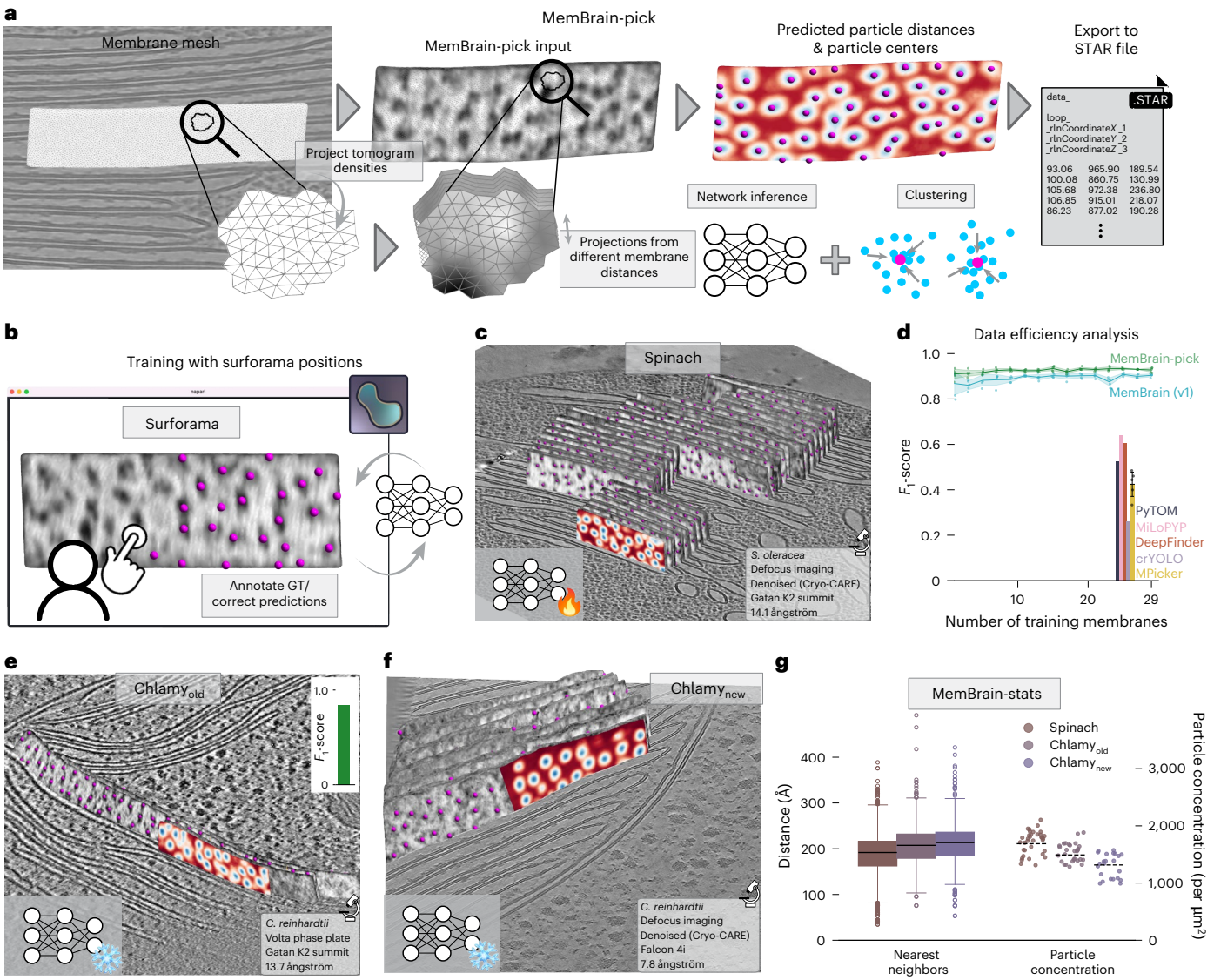


Fig. 3 | MemBrain-pick accurately localizes membrane-associated particles through efficient learning on mesh surfaces. a, MemBrain-pick projects tomographic densities onto a triangular mesh surface at multiple distances along the membrane-normal vector. The DiffusionNet[71]-based architecture operates directly on the surface and predicts a heat map depicting the distance to the nearest particle position (magenta). Particle positions are subsequently identified using Mean Shift clustering on the heat map and exported as .star files. For a detailed workflow, see Extended Data Fig. 7. **b**, Surforama interface enables interactive annotation and correction of MemBrain-pick outputs to generate or refine GT particle positions. **c**, MemBrain-pick predictions on the Spinach test dataset after training with the Spinach training dataset. **d**, MemBrain-pick training results with incremental numbers of training membranes and comparison with other methods (see also Supplementary Fig. 10). Lines indicate mean F_1-scores across runs, shaded areas represent the standard deviation, and dots show individual runs. MiLoPYP, crYOLO and DeepFinder were only trained with 25 membranes (validation split based on tomogram), and MPicker with 29 membranes. Template matching was performed in PyTOM. Bars indicate mean F_1-scores, error bars (for MPicker) represent the standard deviation, and dots show individual runs. **e**, Predictions on the $\text{Chlamy}_{\text{old}}$ dataset with the same model as in **c**. Inset shows the F_1-score performance evaluated with GT particle positions. **f**, Predictions on the $\text{Chlamy}_{\text{new}}$ dataset with the same model as in **c**. **g**, MemBrain-stats quantifies nearest-neighbor distances and particle concentrations for the predictions in **c**, **e** and **f** (see also Extended Data Fig. 10). Box plots show medians (center lines), interquartile ranges (boxes) and whiskers extending to 1.5 times the interquartile range; outliers are shown as points. Particle concentrations are shown per membrane (dots), with dashed lines indicating global concentration (total particles divided by total membrane area). Sample sizes: Spinach ($N = 4{,}322$ positions; 35 membranes), Chlamy_old ($N = 1{,}091$; 28), Chlamy_new ($N = 1{,}164$; 23).

These annotations allow the training of a specialized particle localization model: MemBrain-pick enables efficient and accurate localization of membrane-bound particles by integrating membrane geometry into the particle detection pipeline. It uses a DiffusionNet-based[71] neural network to directly operate on membrane meshes with projected tomographic densities (Fig. 3a and Extended Data Fig. 7), which reduces the search space to membrane-associated regions and improves detection efficiency. Additionally, this network design is aligned with the visualizations in Surforama, making the output more interpretable. The network predicts a heat map representing the distance to the nearest particle center, which is then further processed with our score-guided mean shift clustering[72] to predict precise particle positions (see examples in Extended Data Figs. 8 and 9).

We tested MemBrain-pick on stacked Spinach thylakoid membranes, demonstrating exceptional performance for Photosystem II (PSII) localization (Fig. 3c,d). Even with only a single annotated membrane, our model achieved robust performance with an F_1-score of 91%, highlighting its ability to operate with minimal training data. In contrast, other deep-learning approaches such as DeepFinder and crYOLO require extensive volumetric annotations

https://doi.org/10.1038/s41592-026-03178-8

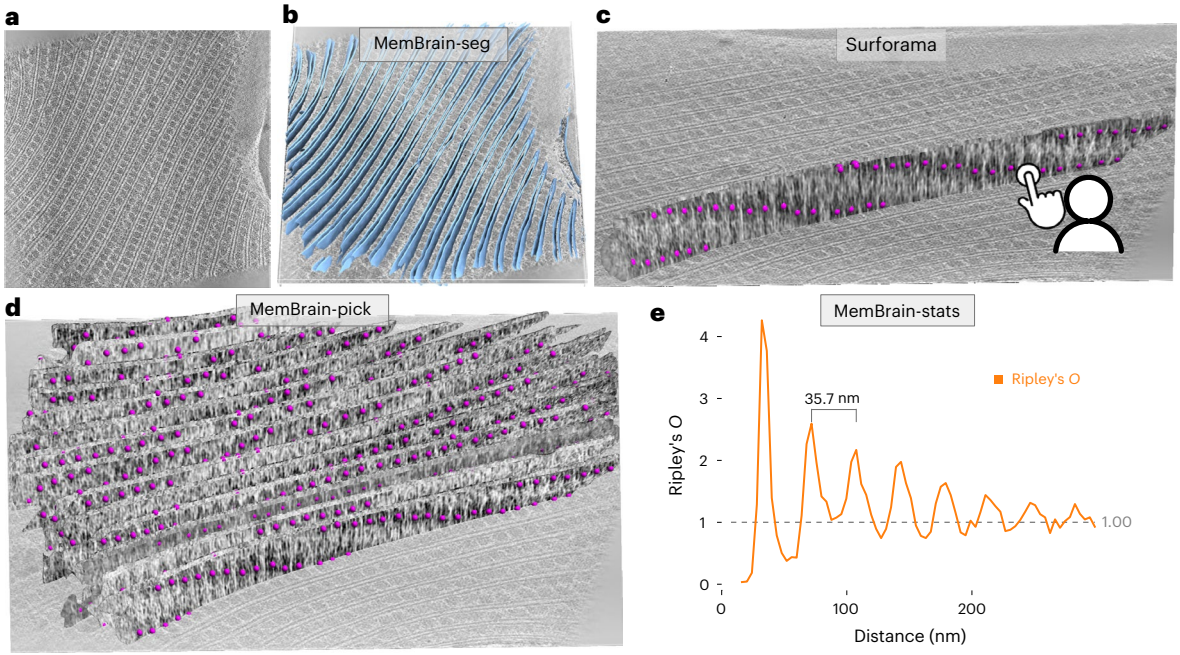


Fig. 4 | MemBrain v2 end-to-end workflow detects periodic phycobilisome organization. a, Raw tomogram slice of EMD-31244. **b**, Out-of-the-box MemBrain-seg segmentation (light blue). **c**, A single-membrane instance can be visualized in Surforama and manually annotated with GT phycobilisome positions (magenta). **d**, MemBrain-pick localizes particles (trained with data from **c**) on all membranes in the tomogram. **e**, MemBrain-stats computes Ripley's *O* statistic using the positions from **d** with a bin size of 5 nm. The distance between peaks (35 nm) was measured to estimate phycobilisome chain unit spacing.

(Supplementary Fig. 9) and struggle with sparsely annotated training data (DeepFinder 60%, crYOLO 26% F_1-score when trained with 25 membranes; Supplementary Fig. 10b,c). To most effectively make use of the sparse annotations, MiLoPYP relies on self-supervised and few-shot learning for particle localization, but it still does not match MemBrain-pick's performance in this task (MiLoPYP 64% F_1-score). Similarly, specialized approaches like MPicker (combined with EPicker, 42% F_1-score; Supplementary Fig. 10a) and MemBrain v1 (90% F_1-score) also fail to match MemBrain-pick's performance (93% F_1-score when trained on 29 membranes). Template matching (performed in PyTOM[50,51]) achieved an F_1-score of 52%.

To evaluate MemBrain-pick's generalizability across datasets, we applied one of these Spinach-trained models (that is, trained with 29 membranes) to stacked thylakoid membranes in two *Chlamydomonas* datasets ($\text{Chlamy}_{\text{old}}$ and $\text{Chlamy}_{\text{new}}$). Despite differences in species and imaging setups, MemBrain-pick maintained robust localization with high F_1-scores (84%) in the $\text{Chlamy}_{\text{old}}$ dataset (Fig. 3e). Predictions on the $\text{Chlamy}_{\text{new}}$ dataset (which lacked GT annotations) also appeared clean and plausible (Fig. 3f).

MemBrain-stats: quantitative analysis of particle distributions

MemBrain-stats complements MemBrain-seg and MemBrain-pick by providing quantitative analysis of particle distributions on membrane surfaces. The module computes particle-surface concentrations, geodesic nearest-neighbor distances and spatial distribution statistics such as the Ripley's functions (Extended Data Fig. 10). Additionally, MemBrain-stats is compatible with outputs from the Surface Morphometrics Pipeline[13], allowing users to load these results into our membrane containers and jointly analyze them with particle concentrations (Extended Data Fig. 9). These metrics allow users to quantitatively interpret membrane particle organization within cryo-ET data, enabling deeper biological insights into membrane-associated processes.

Using MemBrain-stats, we quantified the distribution of the above-predicted thylakoid membrane particles and compared them to previously published values for total particles in stacked thylakoids (Fig. 3g). For Spinach, our results closely align with the published particle concentration[22] (Spinach versus published: 1,688 versus 1,714 particles per μm^2). For *Chlamydomonas*, our predicted values are slightly higher than the reported concentration[17] ($\text{Chlamy}_{\text{old}}$ and $\text{Chlamy}_{\text{new}}$ versus published: 1,492 and 1,316 versus 1,292 particles per μm^2). Because the previous studies only measured PSII–PSII nearest-neighbor distances and neglected some densities marked as 'unknown', our computed nearest-neighbor distances between all particles (corresponding to PSII + 'unknown') are slightly shorter (Spinach versus published: 18.8 versus 21.2 nm; $\text{Chlamy}_{\text{old}}$ and $\text{Chlamy}_{\text{new}}$ versus published: 20.7 nm and 21.2 nm versus 24.4 nm).

In summary, this streamlined MemBrain v2 workflow achieved comparable results to previously published manual analysis of stacked thylakoids, but with greatly accelerated speed and throughput that opens the door to studies with more biological conditions and reproducibility. Below, we apply this pipeline to analyze the higher-order organization of additional types of membrane-bound particles—phycobilisomes and ribosomes.

Test application: phycobilisome organization on thylakoid membranes

Phycobilisomes are large light-harvesting antennae, found in cyanobacteria and some species of eukaryotic algae, that capture light and transfer the excitation energy to thylakoid membrane-embedded photosystems. Phycobilisomes from different species were previously shown by cryo-ET to assemble into rows[73,74], and the function of this higher-order organization in photosynthesis remains an open question. Previous studies of phycobilisome organization in cryo-ET relied heavily on manual annotations, making a large-scale analysis extremely time-consuming[75,76]. By applying MemBrain v2 to a high-resolution cryo-ET dataset of red algae chloroplasts (EMD-31244)[75], we demonstrate how our approach can efficiently extract particle positions and spatial patterns of phycobilisome chains (Fig. 4).

In the first processing step, MemBrain-seg consistently produced clearly separated, high-precision segmentations of all thylakoid membranes (Fig. 4b). For further processing, we utilized the MemBrain lasso tool (Extended Data Fig. 1b) to perform connected component

https://doi.org/10.1038/s41592-026-03178-8

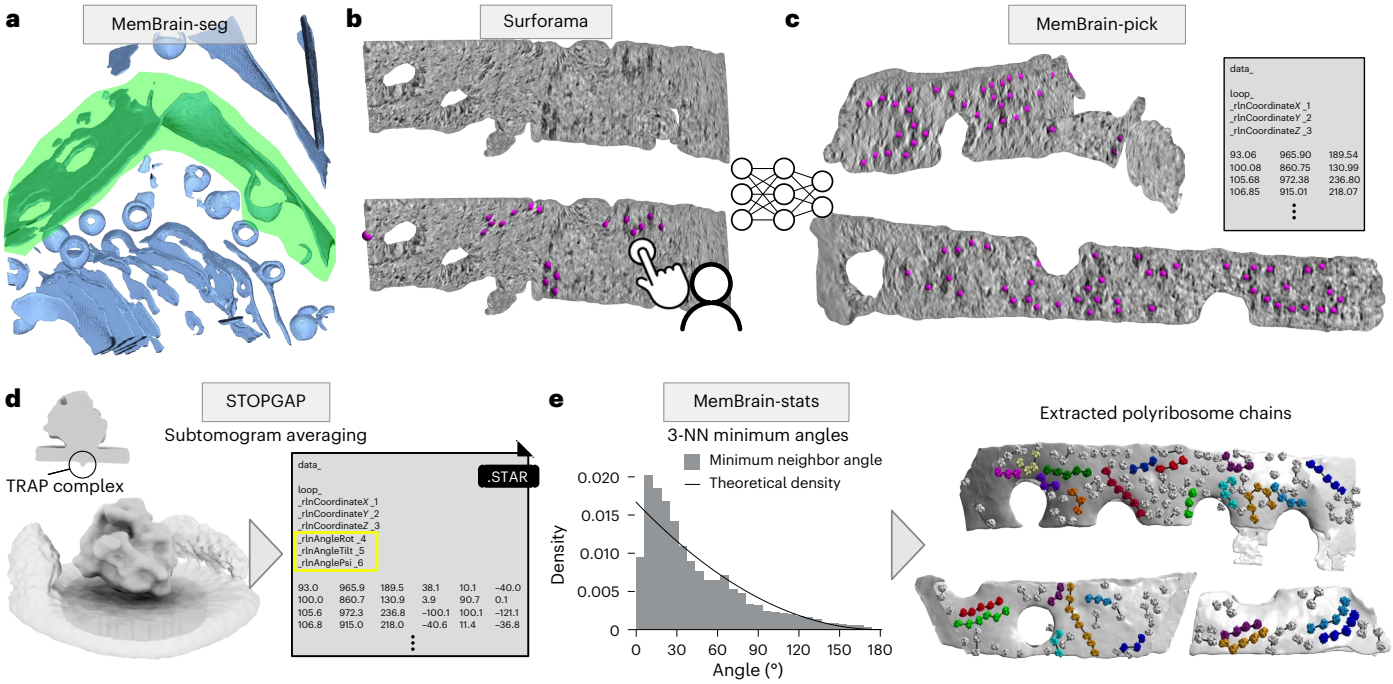


Fig. 5 | MemBrain v2 end-to-end workflow analyzes nearest-neighbor orientations and finds poly-ribosome chains on outer membranes of the nuclear envelope. a, MemBrain-seg prediction (light blue) on an entire tomogram from EMPIAR-11830 depicting a nuclear envelope and surrounding membranes. The nuclear envelope is isolated using the MemBrain lasso tool in Napari (green indicates selected volume). **b**, Top, visualization of the extracted membrane from **a** in Surforama. Bottom, GT ribosome positions (magenta) are manually annotated to train MemBrain-pick. **c**, MemBrain-pick predictions (magenta) on another nuclear envelope from the same dataset after training with data from **b**. **d**, STA results. Left, subtomogram average depicting a membrane-bound ribosome (22.7-Å resolution from all 4,515 particles identified by MemBrain-pick), with a clipped view highlighting the transmembrane TRAP complex. Right, .star file as output giving particle orientations in addition to positions. **e**, Left, histogram of three-nearest-neighbor (3-NN) minimum angles: For each position, we plotted the minimum angle between the corresponding orientation and the three nearest-neighbors' orientations. The theoretical density depicts the behavior for uniformly random angles between 0° and 180°. Right, extracted poly-ribosome chains (different colors) using a combination of in-plane angles and distances between nearest neighbors. Ribosomes not part of a chain of at least three units are shown in gray.

analysis, allowing us to extract individual membrane instances and visualize them in Surforama[61]. This interactive visualization enabled efficient manual annotation of phycobilisome chain unit positions in six selected membrane instances (Fig. 4c). These manually determined positions served as GT data to train a MemBrain-pick model. Once trained, MemBrain-pick was applied to the remaining 23 membranes in the tomogram, successfully identifying phycobilisome positions across the entire tomogram. The predicted positions exhibited a clear periodic arrangement of phycobilisomes along the membranes (Fig. 4d). To quantify this spatial organization, we applied MemBrain-stats to compute the Ripley's *O* function, which confirmed the regular spacing of phycobilisomes (~35 nm; Fig. 4e). This periodicity is consistent with previous manually determined values (34.5 nm[75]), demonstrating the reliability of our automated approach. This end-to-end MemBrain analysis from raw tomograms to quantified phycobilisome chains exemplifies how to rapidly extract spatial information from large datasets to investigate native membrane organization in many cells and experimental conditions.

Test application: poly-ribosome chain organization on the nuclear envelope

In another end-to-end application of MemBrain v2, we analyzed ribosome distributions on the outer membrane of the nuclear envelope, aiming to identify patterns in their localization and orientation. The EMPIAR-11830 (ref. 77) dataset contains over 1,800 tomograms capturing diverse organelles in *C. reinhardtii*. When analyzing tomograms containing the nuclear periphery, MemBrain-seg accurately segmented the nuclear envelope as well as all cytoplasmic membrane structures around it (Fig. 5a). Using the MemBrain lasso tool, we isolated the nuclear envelope membranes of interest and visualized them in Surforama, enabling efficient annotation of GT ribosome positions (Fig. 5b).

To automate ribosome localization, we manually annotated 13 membranes, trained a MemBrain-pick model and predicted ribosome positions across the remaining 79 membranes, detecting a total of 4,515 positions (example predictions in Fig. 5c). To validate these predictions, we performed subtomogram averaging (STA) in STOPGAP[52], which resulted in a map clearly depicting a membrane-bound ribosome (Fig. 5d), including the density for the transmembrane TRAP complex[78]. The average also contains fuzzy peripheral densities likely corresponding to neighboring ribosomes, which can also be inferred by inspecting the distribution of the picks (Fig. 5e). In addition to structural validation, STA provided particle orientation estimates, enabling further spatial analysis with MemBrain-stats.

To further quantify ribosome organization, we analyzed nearest-neighbor orientation patterns. The distribution of minimum angles among the three geodesic nearest neighbors revealed a shift toward smaller angles compared to a uniformly random orientation model (Fig. 5e). This result aligns with previous studies reporting ribosome clustering in chains along the nuclear envelope[3,79,80]. Leveraging MemBrain-stats output, we extracted poly-ribosome chains by integrating constraints on nearest-neighbor angles, distances and chain continuity (Fig. 5e). This end-to-end MemBrain analysis from raw tomograms to identified poly-ribosome chains can facilitate the study of ribosome biology in diverse organisms.

Discussion

The analysis of membranes by cryo-ET is beneficial for many biological studies, as membranes and their associated particles play central roles

in cellular processes. However, the 3D nature of cryo-ET data introduces substantial challenges. Manual segmentation of membranes and annotation of their embedded particles is tedious and time-consuming, particularly because 3D structures require inspection from many orientations and on different planes. Additionally, a fundamental challenge in cryo-ET remains the lack of publicly available GT annotations, limiting the development of robust machine-learning models.

MemBrain v2 addresses these challenges by providing an integrated solution for robust membrane segmentation, membrane-associated particle localization and quantitative spatial analysis. The MemBrain-seg module generates generalizable membrane segmentations using a diverse, well-annotated training dataset and specialized augmentations. By making this dataset publicly available, we provide a foundation for further development and benchmarking of different segmentation methods. To facilitate benchmarking, we include the Surface-Dice metric, which is more sensitive to topological continuity than the traditional Dice score. The MemBrain-pick module offers interactive and data-efficient model training for the localization of membrane-bound particles, substantially reducing the effort required for large-scale analysis. Finally, MemBrain-stats provides quantitative tools for analyzing membrane-bound particle distributions, enabling users to extract meaningful biological insights from native membranes. Together, these components form an end-to-end pipeline for membrane analysis in cryo-ET.

Despite these advancements, certain limitations remain. MemBrain-seg, like other segmentation approaches, is limited by the diversity of available training data and does not explicitly avoid all membrane-like non-target structures. In practice, such cases can be addressed through lightweight fine-tuning and are often reflected in our prediction uncertainty. Moreover, while MemBrain-seg robustly captures membrane morphology, the resulting segmentation thickness does not necessarily reflect the true biological membrane thickness. Variations in segmentation thickness may stem from annotator-dependent biases, such as differing brush sizes (Supplementary Fig. 4). Another challenge is to maintain MemBrain-pick's performance when trying to localize diverse membrane protein complexes, as the heterogeneity of targets can complicate universal model performance. Future improvements could involve integrating MemBrain v2 with complementary tools like TomoTwin[57], to improve versatility across various membrane and protein complex types. For MemBrain-pick, this could lead to more expressive network input features, while MemBrain-seg could also potentially benefit in that this would enable it to distinguish different membrane types in a self-supervised way. Another challenge stems from the inherent constraints of cryo-ET data—feature visibility depends on contrast and noise levels of the tomograms, meaning faint or occluded particles may be overlooked or poorly annotated. Additionally, strong membrane density signals may dominate and obscure weaker signals from small integral membrane proteins, complicating their accurate detection. These challenges point toward opportunities for future innovation.

MemBrain v2 is built with accessibility and practicality in mind. Its command-line interface for MemBrain-seg simplifies segmentation workflows, making advanced membrane analysis accessible even to users without programming expertise. The seamless integration of MemBrain-pick with Napari plugins fosters an intuitive and interactive experience for particle localization, enabling rapid model refinement and validation. MemBrain-pick and MemBrain-stats can operate on segmentations from MemBrain-seg or other membrane segmentation tools, outputting STAR files for compatibility with STA pipelines and enabling joint analysis with Surface Morphometrics Pipeline results. Importantly, MemBrain-seg has already gained considerable traction within the cryo-ET community. Many groups have adopted it for their membrane segmentation tasks[12,15,20,21,31,33,81–95], and it has been applied by the Chan Zuckerberg Imaging Institute to their extensive public cryo-ET dataset[66]. By making all components and datasets open source, MemBrain v2 aims to support collaboration and transparency within the scientific community, offering a versatile and reliable tool for advancing cryo-ET research.

Online content

Any methods, additional references, Nature Portfolio reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41592-026-03178-8.

References

1. Turk, M. & Baumeister, W. The promise and the challenges of cryo-electron tomography. *FEBS Lett.* **594**, 3243–3261 (2020).
2. Young, L. N. & Villa, E. Bringing structure to cell biology with cryo-electron tomography. *Annu. Rev. Biophys.* **52**, 573–595 (2023).
3. Gemmer, M. et al. Visualization of translation and protein biogenesis at the ER membrane. *Nature* **614**, 160–167 (2023).
4. Albert, S. et al. Direct visualization of degradation microcompartments at the ER membrane. *Proc. Natl Acad. Sci. USA* **117**, 1069–1080 (2020).
5. Bykov, Y. S. et al. The structure of the COPI coat determined within the cell. *eLife* **6**, e32493 (2017).
6. Li, M., Tripathi-Giesgen, I., Schulman, B. A., Baumeister, W. & Wilfling, F. In situ snapshots along a mammalian selective autophagy pathway. *Proc. Natl Acad. Sci. USA* **120**, e2221712120 (2023).
7. Bieber, A. et al. In situ structural analysis reveals membrane shape transitions during autophagosome formation. *Proc. Natl Acad. Sci. USA* **119**, e2209823119 (2022).
8. Foster, H. E., Ventura Santos, C. & Carter, A. P. A cryo-ET survey of microtubules and intracellular compartments in mammalian axons. *J. Cell Biol.* **221**, e202103154 (2022).
9. Radecke, J. et al. Morphofunctional changes at the active zone during synaptic vesicle exocytosis. *EMBO Rep.* **24**, e55719 (2023).
10. Radhakrishnan, A. et al. Symmetrical arrangement of proteins under release-ready vesicles in presynaptic terminals. *Proc. Natl Acad. Sci. USA* **118**, e2024029118 (2021).
11. Liu, Y. -T. et al. Mesophasic organization of GABAA receptors in hippocampal inhibitory synapses. *Nat. Neurosci.* **23**, 1589–1596 (2020).
12. Matsui, A. et al. Cryo-electron tomographic investigation of native hippocampal glutamatergic synapses. *eLife* **13**, RP98458 (2024).
13. Barad, B. A., Medina, M., Fuentes, D., Wiseman, R. L. & Grotjahn, D. A. Quantifying organellar ultrastructure in cryo-electron tomography using a Surface Morphometrics Pipeline. *J. Cell Biol.* **222**, e202204093 (2023).
14. Waltz, F. et al. How to build a ribosome from RNA fragments in *Chlamydomonas mitochondria*. *Nat. Commun.* **12**, 7176 (2021).
15. Gemin, O. et al. Ribosomes hibernate on mitochondria during cellular stress. *Nat. Commun.* **15**, 8666 (2024).
16. Huokko, T. et al. Probing the biogenesis pathway and dynamics of thylakoid membranes. *Nat. Commun.* **12**, 3475 (2021).
17. Wietrzynski, W. et al. Charting the native architecture of Chlamydomonas thylakoid membranes with single-molecule precision. *eLife* **9**, e53740 (2020).
18. Gupta, T. K. et al. Structural basis for VIPP1 oligomerization and maintenance of thylakoid membrane integrity. *Cell* **184**, 3643–3659 (2021).
19. Waltz, F. et al. In-cell architecture of the mitochondrial respiratory chain. *Science* **387**, 1296–1301 (2025).
20. Chang, Y. -T. et al. Cytoplasmic ribosomes on mitochondria alter the local membrane environment for protein import. *J. Cell Biol.* **224**, e202407110 (2025).

21. Dietrich, L., Agip, A. -N. A., Kunz, C., Schwarz, A. & Kühlbrandt, W. In situ structure and rotary states of mitochondrial ATP synthase in whole Polytomella cells. *Science* **385**, 1086–1090 (2024).
22. Wietrzynski, W. et al. Molecular architecture of thylakoid membranes within intact spinach chloroplasts. *eLife* **14**, RP105496 (2025).
23. Wozny, M. R. et al. In situ architecture of the ER-mitochondria encounter structure. *Nature* **618**, 188–192 (2023).
24. Collado, J. et al. Tricalbin-mediated contact sites control ER curvature to maintain plasma membrane integrity. *Dev. Cell* **51**, 476–487.e7 (2019).
25. Rodrigues-Oliveira, T. et al. Actin cytoskeleton and complex cell architecture in an Asgard archaeon. *Nature* **613**, 332–339 (2023).
26. Sartori-Rupp, A. et al. Correlative cryo-electron microscopy reveals the structure of TNTs in neuronal cells. *Nat. Commun.* **10**, 342 (2019).
27. Mendonça, L. et al. Correlative multi-scale cryo-imaging unveils SARS-CoV-2 assembly and egress. *Nat. Commun.* **12**, 4629 (2021).
28. Klein, S. et al. SARS-CoV-2 structure and replication characterized by in situ cryo-electron tomography. *Nat. Commun.* **11**, 5885 (2020).
29. Wolff, G. et al. A molecular pore spans the double membrane of the coronavirus replication organelle. *Science* **369**, 1395–1398 (2020).
30. Klein, S. et al. IFITM3 blocks influenza virus entry by sorting lipids and stabilizing hemifusion. *Cell Host Microbe* **31**, 616–633 (2023).
31. Armbruster, E. G. et al. Sequential membrane- and protein-bound organelles compartmentalize genomes during phage infection. *Cell Host Microbe* **33**, 484–497.e6 (2025).
32. Dahmane, S. et al. Membrane-assisted assembly and selective secretory autophagy of enteroviruses. *Nat. Commun.* **13**, 5986 (2022).
33. Dahmane, S. et al. Cryo-electron tomography reveals coupled flavivirus replication, budding and maturation. *Nat. Commun.* **17**, 828 (2026).
34. Navarro, P. P. et al. Cell wall synthesis and remodelling dynamics determine division site architecture and cell shape in *Escherichia coli*. *Nat. Microbiol.* **7**, 1621–1634 (2022).
35. Khanna, K., Lopez-Garrido, J., Sugie, J., Pogliano, K. & Villa, E. Asymmetric localization of the cell division machinery during Bacillus subtilis sporulation. *eLife* **10**, e62204 (2021).
36. Buchholz, T.-O., Jordan, M., Pigino, G. & Jug, F. Cryo-CARE: content-aware image restoration for cryo-transmission electron microscopy data. In *IEEE 16th International Symposium on Biomedical Imaging (ISBI)* (eds Ledesma Carbayo, M. J. & González Ballester, M. Á.) 502–506 https://doi.org/10.1109/ISBI.2019.8759519 (IEEE, 2019).
37. Bepler, T., Kelley, K., Noble, A. J. & Berger, B. Topaz-Denoise: general deep denoising models for cryoEM and cryoET. *Nat. Commun.* **11**, 5208 (2020).
38. Maldonado, J. C. et al. F2FD: fourier perturbations for denoising cryo-electron tomograms and comparison to established approaches. In *IEEE 20th International Symposium on Biomedical Imaging (ISBI)* 1–5 https://doi.org/10.1109/isbi53787.2023.10230476 (2023).
39. Purnell, C., Heebner, J. & Swulius, M. Training neural networks with simulated CryoET data. *Microsc. Microanal.* **29**, 967–968 (2023).
40. Liu, Y. -T. et al. Isotropic reconstruction for electron tomography with deep learning. *Nat. Commun.* **13**, 6482 (2022).
41. Wiedemann, S. & Heckel, R. A deep learning method for simultaneous denoising and missing wedge reconstruction in cryogenic electron tomography. *Nat. Commun.* **15**, 8255 (2024).
42. Kishore, V., Debarnot, V., Righetto, R. D., Engel, B. D. & Dokmanić, I. ICECREAM: high-fidelity equivariant cryo-electron tomography. *Acta Crystallogr. D Struct. Biol.* **82**, 295–313 (2026).
43. Martinez-Sanchez, A., Garcia, I., Asano, S., Lucic, V. & Fernandez, J. -J. Robust membrane detection based on tensor voting for electron tomography. *J. Struct. Biol.* **186**, 49–61 (2014).
44. Ronneberger, O., Fischer, P. & Brox, T. U-Net: convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (eds Navab, N. et al.) 234–241 https://doi.org/10.1007/978-3-319-24574-4_28 (Springer, 2015).
45. de Teresa-Trueba, I. et al. Convolutional networks for supervised mining of molecular patterns within cellular context. *Nat. Methods* **20**, 284–294 (2023).
46. Muth, S. et al. SynapseNet: deep learning for automatic synapse reconstruction. *Mol. Biol. Cell* **36**, ar127 (2025).
47. Khosrozadeh, A. et al. CryoVesNet: a dedicated framework for synaptic vesicle segmentation in cryo-electron tomograms. *J. Cell Biol.* **224**, e202402169 (2025).
48. Last, M. G. F., Abendstein, L., Voortman, L. M. & Sharp, T. H. Streamlining segmentation of cryo-electron tomography datasets with Ais. *eLife* **13**, RP98552 (2024).
49. Kiewisz, R. et al. Accurate and fast segmentation of filaments and membranes in micrographs and tomograms with TARDIS. Preprint at *bioRxiv* https://doi.org/10.1101/2024.12.19.629196 (2024).
50. Hrabe, T. et al. PyTom: a python-based toolbox for localization of macromolecules in cryo-electron tomograms and subtomogram analysis. *J. Struct. Biol.* **178**, 177–188 (2012).
51. Chaillet, M. L. et al. Extensive angular sampling enables the sensitive localization of macromolecules in electron tomograms. *Int. J. Mol. Sci.* **24**, 13375 (2023).
52. Wan, W., Khavnekar, S. & Wagner, J. STOPGAP: an open-source package for template matching, subtomogram alignment and classification. *Acta Crystallogr. D Struct. Biol.* **80**, 336–349 (2024).
53. Wagner, T. et al. SPHIRE-crYOLO is a fast and accurate fully automated particle picker for cryo-EM. *Commun. Biol.* **2**, 218 (2019).
54. Genthe, E. et al. PickYOLO: fast deep learning particle detector for annotation of cryo electron tomograms. *J. Struct. Biol.* **215**, 107990 (2023).
55. Moebel, E. et al. Deep learning improves macromolecule identification in 3D cellular cryo-electron tomograms. *Nat. Methods* **18**, 1386–1394 (2021).
56. Liu, G. et al. DeepETPicker: fast and accurate 3D particle picking for cryo-electron tomography using weakly supervised deep learning. *Nat. Commun.* **15**, 2090 (2024).
57. Rice, G. et al. TomoTwin: generalized 3D localization of macromolecules in cryo-electron tomograms with structural data mining. *Nat. Methods* **20**, 871–880 (2023).
58. Huang, Q., Zhou, Y. & Bartesaghi, A. MiLoPYP: self-supervised molecular pattern mining and particle localization in situ. *Nat. Methods* **21**, 1863–1872 (2024).
59. Yan, X. et al. MPicker: visualizing and picking membrane proteins for cryo-electron tomography. *Nat. Commun.* **16**, 472 (2025).
60. Tegunov, D. Membranorama, version cdc9287. *GitHub* https://github.com/dtegunov/membranorama (2023).
61. Yamauchi, K. A. et al. Surforama: interactive exploration of volumetric data by leveraging 3D surfaces. Preprint at *bioRxiv* https://doi.org/10.1101/2024.05.30.596601 (2024).
62. Lamm, L. et al. MemBrain: a deep learning-aided pipeline for detection of membrane proteins in Cryo-electron tomograms. *Comput. Methods Programs Biomed.* **224**, 106990 (2022).
63. Salfer, M., Collado, J. F., Baumeister, W., Fernández-Busnadiego, R. & Martínez-Sánchez, A. Reliable estimation of membrane curvature for cryo-electron tomography. *PLoS Comput. Biol.* **16**, e1007962 (2020).
64. Martinez-Sanchez, A., Baumeister, W. & Lučić, V. Statistical spatial analysis for cryo-electron tomography. *Comput. Methods Programs Biomed.* **218**, 106693 (2022).

65. Chiu, C. -L. & Clack, N. Napari: a Python multi-dimensional image viewer platform for the research community. *Microsc. Microanal.* **28**, 1576–1577 (2022).
66. Ermel, U. et al. A data portal for providing standardized annotations for cryo-electron tomography. *Nat. Methods* **21**, 2200–2202 (2024).
67. Maitin-Shepard, J. et al. Google/Neuroglancer. *Zenodo* https://doi.org/10.5281/zenodo.5573293 (2021).
68. Martinez-Sanchez, A., Lamm, L., Jasnin, M. & Phelippeau, H. Simulating the cellular context in synthetic datasets for cryo-electron tomography. *IEEE Trans. Med. Imaging* **43**, 3742–3754 (2024).
69. Tegunov, D. & Cramer, P. Real-time cryo-electron microscopy data preprocessing with Warp. *Nat. Methods* **16**, 1146–1152 (2019).
70. Bertiaux, E. et al. The luminal ring protein C2CD3 acts as a radial in-to-out organizer of the distal centriole and appendages. *PLoS Biol.* **23**, e3003519 (2025).
71. Sharp, N., Attaiki, S., Crane, K. & Ovsjanikov, M. DiffusionNet: discretization agnostic learning on surfaces. *ACM Trans. Graph.* **41**, 1–16 (2022).
72. Derpanis, K. G. Mean shift clustering. *Lecture Notes* 32 (2005).
73. Rast, A. et al. Biogenic regions of cyanobacterial thylakoids form contact sites with the plasma membrane. *Nat. Plants* **5**, 436–446 (2019).
74. Shen, L. et al. Structure of a unique PSII-Pcb tetrameric mega-complex in a chlorophyll d-containing cyanobacterium. *Sci. Adv.* **10**, eadk7140 (2024).
75. Li, M., Ma, J., Li, X. & Sui, S.-F. In situ cryo-ET structure of phycobilisome-photosystem II supercomplex from red alga. *eLife* **10**, e69635 (2021).
76. You, X. et al. In situ structure of the red algal phycobilisome-PSII-PSI-LHC megacomplex. *Nature* **616**, 199–206 (2023).
77. Kelley, R. et al. Toward community-driven visual proteomics with large-scale cryo-electron tomography of Chlamydomonas reinhardtii. *Mol. Cell* **86**, 213–230.e7 (2026).
78. Pfeffer, S. et al. Dissecting the molecular organization of the translocon-associated protein complex. *Nat. Commun.* **8**, 14516 (2017).
79. Afonina, Z. A. & Shirokov, V. A. Three-dimensional organization of polyribosomes–A modern approach. *Biochemistry* **83**, S48–S55 (2018).
80. Pfeffer, S. et al. Structure and 3D arrangement of endoplasmic reticulum membrane-associated ribosomes. *Structure* **20**, 1508–1518 (2012).
81. Taniguchi, R. et al. Nuclear pores safeguard the integrity of the nuclear envelope. *Nat. Cell Biol.* **27**, 762–775 (2025).
82. Klumpe, S. et al. In-cell structure and snapshots of copia retrotransposons in intact tissue by cryoelectron tomography. *Cell* https://doi.org/10.1016/j.cell.2025.02.003 (2025).
83. Tavakoli, A., Hu, S., Ebrahim, S. & Kachar, B. Hemifusomes and interacting proteolipid nanodroplets mediate multi-vesicular body formation. *Nat. Commun.* **16**, 4609 (2025).
84. Swistak, L. et al. The bacterial type three secretion system induces mechanoporation of vacuolar membranes. *PLoS Biol.* **23**, e3003135 (2025).
85. Akıl, C., Xu, J., Shen, J. & Zhang, P. Unveiling the structural spectrum of SARS-CoV-2 fusion by in situ cryo-ET. *Nat. Commun.* **16**, 5150 (2025).
86. Glynn, C. et al. A generalizable and targeted molecular biopsy approach for in situ cryogenic electron tomography of vitreous brain tissue. *Cell Rep. Methods* **5**, 101080 (2025).
87. Sachweh, J. et al. The small GTPase Ran defines nuclear pore complex asymmetry. *Cell* **188**, 5931–5946.e16 (2025).
88. Glushkova, D., Böhm, S. & Beck, M. Systematic membrane thickness variation across cellular organelles revealed by cryo-ET. *J. Cell Biol.* **225**, e202504053 (2026).
89. Zhao, X.-T. et al. The Myo2 adaptor Ldm1 and its receptor Ldo16 mediate actin-dependent lipid droplet motility. *Cell Rep.* **44**, 116475 (2025).
90. Erwin, A. L. et al. Molecular visualization of neuronal TDP43 pathology in situ. Preprint at *bioRxiv* https://doi.org/10.1101/2024.08.19.608477 (2024).
91. Lu, M.-A. et al. Molecular mechanisms of flotillin complexes in organizing membrane microdomains. *Nat. Commun.* **17**, 2541 (2026).
92. Rangan, R. et al. CryoDRGN-ET: deep reconstructing generative networks for visualizing dynamic biomolecules inside cells. *Nat. Methods* **21**, 1537–1545 (2024).
93. Lascaux, P. et al. TEX264 drives selective autophagy of DNA lesions to promote DNA repair and cell survival. *Cell* **187**, 5698–5718 (2024).
94. Ding, C. et al. SARM1 loss protects retinal ganglion cells in a mouse model of autosomal dominant optic atrophy. *J. Clin. Invest.* **135**, e191315 (2025).
95. Rose, K. et al. In situ cryo-ET visualization of mitochondrial depolarization and mitophagic engulfment. *Proc. Natl Acad. Sci. USA* **122**, e2511890122 (2025).

[1]Helmholtz Munich, German Research Center for Environment and Health, Munich, Germany. [2]Biozentrum, University of Basel, Basel, Switzerland. [3]School of Computation, Information and Technology, Technical University of Munich, Munich, Germany. [4]Department of Biosystems Science and Engineering, ETH Zürich, Basel, Switzerland. [5]Swiss Institute of Bioinformatics, Basel, Switzerland. [6]MRC Laboratory of Molecular Biology, Cambridge Biomedical Campus, Cambridge, UK. [7]Department of Structural Biology, Genentech, South San Francisco, CA, USA. [8]Department of Information and Communications Engineering, Faculty of Computers Sciences, University of Murcia, Murcia, Spain. [9]German Cancer Research Center (DKFZ), Division of Medical Image Computing, Heidelberg, Germany. [10]Helmholtz Imaging, German Cancer Research Center (DKFZ), Heidelberg, Germany. [11]School of Biomedical Engineering and Imaging Sciences, King's College London, London, UK. [12]Munich Center for Machine Learning, Munich, Germany.
✉e-mail: lorenz.lamm@tum.de; ben.engel@unibas.ch; tingying.peng@helmholtz-munich.de

Methods

MemBrain-seg

MemBrain-seg is a 3D U-Net-based tool designed for generalizable membrane segmentation in cryo-ET data. It incorporates an iteratively generated training set, specialized augmentations and a Surface-Dice loss function to enhance segmentation accuracy.

U-Net. *Architecture*. MemBrain-seg's U-Net architecture is based on design choices from nnU-Net[96], a leading biomedical image segmentation framework, and is implemented using MONAI[97]. It consists of five downsampling and upsampling blocks, with deep supervision[98] to improve gradient flow and convergence.

Loss function. We use a composite loss function combining binary cross-entropy, Dice loss and Surface-Dice loss (see 'Surface-Dice'). To handle uncertain image regions, an ignore feature excludes certain voxels from loss calculations (see 'Iterative dataset generation' and Supplementary Fig. 1).

Training setup. All models were trained for 1,000 epochs using stochastic gradient descent, a batch size of 2, and 160^3-shaped training patches on RTX-4090 GPUs. We apply a polynomial learning rate scheduler decreasing from 0.01 to 0.0. Before training, intensities are normalized per patch by subtracting the mean and dividing by the standard deviation.

Inference. For inference, tomograms are processed using a sliding-window approach with Gaussian-weighted patch aggregation of scores. Each tomogram is divided into overlapping 160^3 patches, and we apply eightfold test-time augmentation by flipping along all axis combinations, averaging predictions before thresholding. MemBrain-seg also supports internal rescaling to a specified pixel size (default 10 Å). In this mode, patches are extracted at the original resolution, rescaled before passing through the network, then restored to the original scale before aggregation. As rescaling occurs entirely on the GPU, this approach substantially speeds up processing compared to full-tomogram rescaling. Inference on typical tomograms requires approximately a few minutes on a single GPU (see Supplementary Table 2 for a runtime comparison).

Data augmentations. Data augmentation is critical for improving generalization by simulating the diverse appearances of tomograms and membranes[99]. Below, we describe the geometric, intensity-based and Fourier-based augmentations applied during training.

Conventional data augmentations. We apply a mix of classically used geometric and intensity transformations. These augmentations, applied randomly and on-the-fly, introduce strong variation in training samples, enhancing the network's adaptability (Supplementary Fig. 5a).

Geometric transforms. These transformations aim to show the image from different viewing points, and include rotation at arbitrary angles, zooming within a range of 0.7 to 1.4, and both shuffling and flipping of axes.

Intensity transforms. Applied to alter image characteristics, these encompass median filtering, Gaussian blurring, Gaussian noise addition, brightness and contrast adjustments, low-resolution simulation, random erasing, additive brightness gradient, local Gamma transform and sharpening.

Fourier amplitude augmentation. Fourier amplitude spectrum matching, as introduced in DeePiCt[45], adjusts tomogram styles by matching their frequency-domain amplitudes. It does so by rotationally averaging Fourier intensities per frequency band, generating a one-dimensional (1D) sequence for both input and target tomograms. The ratio of these sequences is used to create a 3D radial filter, which is then multiplied in Fourier space to transform the input tomogram.

In MemBrain-seg, however, our goal is to improve generalization without requiring explicit style normalization before inference. Previous studies have shown that training with style augmentations is more effective for generalization than pre-prediction normalization[100,101]. Therefore, we introduce Fourier amplitude augmentation, which randomly modifies tomogram styles during training. Rather than deriving transformations from a limited set of reference tomograms, we generate random normalization sequences to simulate diverse styles dynamically.

To achieve this, we create a 1D sequence, x, of length 80 (that is, half the patch size) using a random walk as shown in equation (1):

$$\begin{aligned} x_0 &= 1.0 \\ x_{n+1} &= |x_n + \Delta_n| \\ \Delta_n &\sim N(0, \sigma^2) \end{aligned} \tag{1}$$

where σ = 0.2, |.| is the absolute value, and N is a normal distribution. This sequence is expanded into a 3D radial kernel and multiplied with the Fourier transform of the input volume, introducing random style variations. The effects on image appearance are illustrated in Supplementary Fig. 5b: Upscaling of higher frequencies results in noisier images with weaker membrane contrast (second row), whereas downscaling of high frequencies enhances coarse structures, making membranes more prominent (third row).

Missing wedge augmentation. Segmenting membranes in regions affected by the missing wedge presents a substantial challenge due to the anisotropic distortions it introduces. To mitigate this, we adopt an approach inspired by IsoNet[40], in which we rotate input subvolumes randomly before applying an artificial missing wedge by masking Fourier coefficients in a wedge-like shape. This controlled simulation (Supplementary Fig. 5c) replicates the characteristic loss of densities caused by incomplete angular sampling in cryo-ET. In experimental data, the information lost due to the missing wedge is not directly visible, yet in some cases, the image context provides hints about where membranes should be. By artificially replicating these effects, we train the network to infer missing structures based on contextual information. Training pairs are generated by applying this transformation to subvolumes while preserving their unaltered GT labels, pushing the model to learn to accurately segment membranes even in wedge-impacted regions.

Surface-Dice. Surface-Dice extends Centerline-Dice[102] to 3D membrane segmentation, serving as both a metric for evaluating binarized membrane segmentation outputs and a loss function during network training. By leveraging a differentiable skeletonization approach, Surface-Dice better reflects the continuity and consistency of membrane structures in 3D than normal Dice scores.

Computation. Similarly to Centerline-Dice, Surface-Dice relies on skeletonizations of both the predicted segmentation (M_{pred}) and the ground truth (M_{GT}), denoted as S_{pred} and S_{GT}, respectively. Skeletonization in Surface-Dice reduces the membrane segmentation to a one-voxel-thick surface, forming a 2D manifold in 3D space (illustrated in Supplementary Fig. 3a,b). To compute Surface-Dice ($\text{Dice}_{\text{surf}}$), we define Surface-precision ($\text{Prec}_{\text{surf}}$) and Surface-recall ($\text{Rec}_{\text{surf}}$) as shown in equation (2):

$$\begin{aligned} \text{Prec}_{\text{surf}}(M_{\text{pred}}, M_{\text{GT}}) &= |S_{\text{pred}} \cap M_{\text{GT}}|/|S_{\text{pred}}| \\ \text{Rec}_{\text{surf}}(M_{\text{pred}}, M_{\text{GT}}) &= |S_{\text{GT}} \cap M_{\text{pred}}|/|S_{\text{GT}}| \end{aligned} \tag{2}$$

The |·| operator sums up all positive voxels in the respective set. Surface-Dice ($\mathrm{Dice}_{\mathrm{surf}}$) is then computed as the harmonic mean of these two terms as shown in equation (3):

$$\mathrm{Dice}_{\mathrm{surf}}(M_{\mathrm{pred}}, M_{\mathrm{GT}}) = 2 \times \frac{\mathrm{Prec}_{\mathrm{surf}}(M_{\mathrm{pred}}, M_{\mathrm{GT}}) \times \mathrm{Rec}_{\mathrm{surf}}(M_{\mathrm{pred}}, M_{\mathrm{GT}})}{\mathrm{Prec}_{\mathrm{surf}}(M_{\mathrm{pred}}, M_{\mathrm{GT}}) + \mathrm{Rec}_{\mathrm{surf}}(M_{\mathrm{pred}}, M_{\mathrm{GT}})} \quad (3)$$

Surface-dice as a loss function. To use Surface-Dice as a loss function during training, we avoid thresholding network outputs, as this operation is non-differentiable. Instead, we use a differentiable skeletonization approach, similar to the method used in Centerline-Dice. This process involves iterative membrane erosion to progressively thin the segmentation. At each iteration, an additional erosion and dilation step is applied, and differences between the segmentation before and after these operations are evaluated. The presence of differences indicates that erosion has removed portions of the membrane, meaning the current iteration already represents a thin surface in those regions. Both erosion and dilation are implemented via minimum and maximum pooling, ensuring that all operations remain differentiable and efficient. For further details, refer to the Centerline-Dice publication[102].

Iterative dataset generation. To efficiently build a well-annotated membrane segmentation dataset, we used an iterative approach inspired by active learning principles[103]. This strategy minimized manual annotation efforts by focusing on areas where the network struggled with segmentation. The schematic correction workflow and different iterations are visualized in Fig. 2b and Supplementary Fig. 1. During the annotation process, we pay particular attention to the quality of the training dataset, that is, the accuracy of the segmentation. The ignore label is particularly helpful here, as it prevents the model from learning uncertain regions simply as non-membrane background. This annotation quality is a key factor for the generalization capability of our network. Detailed instructions on how users can annotate training patches to customize the model can be found on GitHub via https://github.com/teamtomo/membrain-seg/blob/main/docs/Usage/Annotations.md/.

Initial dataset and first iteration. We initiated our project with membrane segmentation patches from the *S. oleracea* dataset, initially annotated using TomoSegMemTV[43]. Using MITK Workbench[104], we manually refined these patches, each sized 160^3 voxels, assigning to every voxel the background, membrane or ignore classes—the latter not being evaluated by the loss function during training as it marks uncertain membrane regions, where exact delineation of membranes was challenging. The process of correcting predicted membrane segmentations is depicted in Supplementary Fig. 1b, where we remove false-positive voxels from the segmentation, add falsely negative voxels to it and assign the ignore class in uncertain regions. Using these corrected annotations, we trained an initial U-Net[44,96] model for membrane segmentation. While this model outperformed TomoSegMemTV, its segmentations remained suboptimal. We therefore conducted a second round of annotations, targeting patches where the model's predictions were weak. This refinement process resulted in 69 accurately annotated spinach chloroplast patches (Supplementary Table 1, column 'Spinach'; Supplementary Fig. 1b, 'round 1').

Expansion to further datasets. To improve generalization across different imaging setups, we extended our approach to *C. reinhardtii* tomograms from EMPIAR-11830 (ref. 77). Due to differences in imaging conditions, initial segmentations performed worse than in the Spinach dataset. By iteratively re-annotating 33 patches (covering thylakoid membranes, mitochondria and the Golgi apparatus), we aimed to improve performance on this dataset (Supplementary Table 1, column 'Chlamy'; Supplementary Fig. 1b, 'round 2'). To further expand the model's applicability beyond our own data, we collaborated with other research groups who tested MemBrain-seg on their own datasets and provided re-annotated patches from regions where MemBrain-seg struggled. After careful quality assessment, these contributed 27 additional patches, substantially increasing dataset diversity (Supplementary Table 1, column 'Community'; Supplementary Fig. 1b, 'round 3'). This ongoing collaboration continues to improve MemBrain-seg's performance across a wide range of tomograms. Instructions on how to annotate and share patches can be found on the MemBrain-seg GitHub repository.

Adding publicly available sources. We further leveraged the few existing publicly available segmentation sources to improve our model even further. To achieve this, we generated 40 membrane patches using the open-source tomography simulators PolNet[68] and CryoTomoSim[39] (20 patches each; Supplementary Table 1, column 'Synthetic'; Supplementary Fig. 1b, 'round 4'). Additionally, we integrated an externally annotated dataset by extracting 15 reliable segmentation patches from the publicly available DeePiCt dataset[45] (EMPIAR-10988). These regions were selected from areas where MemBrain-seg initially struggled, enhancing dataset diversity (Supplementary Table 1, column 'DeePiCt'; Supplementary Fig. 1b, 'round 5').

Skeletonization. MemBrain-seg provides a functionality to convert binary segmentations into one-voxel-thick membrane sheets, which is essential for several downstream applications and the computation of Surface-Dice. We solve this task similarly to TomoSegMemTV[43]: First, we compute a distance transform (DT), converting the segmentation into a distance map relative to membrane boundaries. The goal is to identify the center sheet, which corresponds to the regions with the highest distance values. To extract this center sheet, we approximate membrane-normal vectors by calculating the Hessian matrix of the distance transform and extracting its eigenvectors at each voxel p. The eigenvector n associated with the largest intensity curvature provides the direction pointing outward from the membrane. We then apply non-maximum suppression (NMS) along these normal vectors: a voxel is retained if it has the highest DT value along its normal vector; otherwise, it is suppressed, according to equation (4):

$$\mathrm{NMS} = \begin{cases} 1, \text{if } \mathrm{DT}(p) \geq \mathrm{DT}(p + d \times n) \forall d \in [-1, 1] \\ 0, \text{otherwise} \end{cases} \quad (4)$$

This process yields a one-voxel-thick sheet, representing the center sheet of the membrane segmentation.

In addition to this 3D membrane skeletonization approach, we also provide the option to use scikit-learn's[105] 2D skeletonization method, which is then applied for each z-slice separately.

Uncertainty quantification. To estimate the uncertainty of MemBrain-seg predictions, we leverage its test-time augmentation functionality[106]. When enabled (by default), MemBrain-seg's test-time augmentation applies all eight possible axis flips of the input tomogram along the x, y and z axes and performs inference for each orientation. This results in eight independent membrane segmentation predictions per tomogram. While primarily intended to improve segmentation robustness, we further utilize these predictions to compute a per-voxel variance map of the sigmoid-activated outputs, providing a quantitative estimate of model uncertainty across all voxels.

However, we observed that variance values are consistently high close to membrane boundaries, likely because of the ambiguity in defining the actual thickness of the membrane. Therefore, we process the variance maps further to highlight uncertain regions instead of membrane outlines. To this end, we follow two distinct approaches to compute the membrane uncertainty for foreground (that is, membrane) segmentation and background: (1) For the foreground uncertainty,

we first erode the membrane segmentation to exclude the typically uncertain boundary voxels. For the remaining (eroded) membrane regions, the voxel-wise variance values directly define the membrane uncertainty. Boundary voxels removed by erosion are then assigned the nearest-neighbor uncertainty value from the eroded region. (2) For the background uncertainty, we use the per-voxel variance values without further processing. After combining foreground and background uncertainties, we perform median filtering with a 5×5 filter to smooth the uncertainty map. Example results are visualized in Supplementary Fig. 20.

Model fine-tuning. MemBrain-seg's segmentation performance can be affected by batch shifts, leading to suboptimal results when tomogram or membrane appearances deviate substantially from the training domain. To mitigate this, we implemented a fine-tuning strategy that allows users to quickly adapt the model to their specific datasets. Fine-tuning requires a small number of annotated patches (typically 10–20, each 160^3 voxels at 10-Å pixel size) and follows a transfer learning approach[107], where pretrained model weights serve as initialization. To prevent overfitting and reduce computational costs, we limit training to 100 epochs (instead of 1,000) and lower the learning rate to 10^{-5} from 10^{-2}. An early stopping criterion halts training if validation performance deviates too far from the original model's performance, ensuring stability. To enable this, the original, full validation dataset also serves as the validation dataset in this task.

For fine-tuning, incorporating the Surface-Dice loss can be beneficial, as it disregards variations in annotation thickness (for example, see Supplementary Fig. 4), preventing premature stopping due to segmentation thickness differences. This strategy enables MemBrain-seg to efficiently adapt to new datasets, enhancing performance across diverse tomographic conditions. The workflow is illustrated in Supplementary Fig. 8.

MemBrain-seg usability. MemBrain-seg is designed to be as easy to use as possible. The default way of using it is by a command-line interface. Via its API, membrain-seg is also easy to integrate into other software packages and existing pipelines, allowing subsequent analysis of membrane segmentations. Additionally, MemBrain-seg is already integrated into other packages with a graphical user interface, like ColabSeg[108] and Scipion[109].

MemBrain-pick

MemBrain-pick enables the automated localization of membrane-associated particles in cryo-electron tomograms. Unlike other deep-learning approaches, it operates directly on membrane surfaces rather than voxel-based representations, allowing for a more specialized and focused detection of membrane-bound particles.

Workflow. The MemBrain-pick workflow consists of three key steps: data preparation, model training and prediction. Data preparation consists of extracting relevant membrane areas from MemBrain-seg segmentations, and conversion to mesh representations. The model is then trained with membrane meshes as input, where we project tomographic densities onto the mesh triangles (Fig. 3 and Extended Data Fig. 7). We train the DiffusionNet-model[71] to predict a membrane particle distance map. During inference, we extract particle positions using Mean Shift clustering[110].

Data preparation. *Mesh generation and projection*. To represent membranes as analyzable surfaces, we first extract membrane segmentations from MemBrain-seg (for example, crop using our Lasso tool; see 'Napari tools') and convert them into triangular mesh representations. This is achieved using the Marching Cubes algorithm[111]. To ensure a uniform mesh resolution, we resample triangle sizes through Voronoi clustering[112], which redistributes vertices to achieve evenly sized barycentric areas per vertex. Once the mesh is generated, we leverage its per-vertex normal vectors to extract relevant tomographic information. For each vertex, we sample intensity values along its normal vector in both directions, generating a feature vector of N sampled density values per vertex.

Training data generation. MemBrain-pick requires GT annotations for training. To facilitate accurate annotation, MemBrain-pick is compatible with Surforama[61], a Napari-based[65] annotation tool that operates similarly to MemBrain-pick by displaying tomographic densities projected onto a membrane surface. Using Surforama, users can efficiently annotate particle center positions directly on membrane meshes. These annotations are exported as STAR files[113], ensuring seamless integration with MemBrain-pick's training pipeline, as well as other processing software.

Surface partitioning. To optimize model efficiency and prevent overfitting to global membrane geometries, we partition each membrane into smaller overlapping surface crops before feeding them into MemBrain-pick. Each partition consists of 2,000 triangles, selected by initializing at a random seed triangle and iteratively expanding to include the 2,000 nearest neighbors based on geodesic distance. To ensure comprehensive coverage, partitions are generated with sufficient overlap so that each mesh vertex has a neighboring seed within close proximity. This procedure maintains enough in-plane context per patch for accurate particle localization while reducing computational complexity and avoiding learning global geometries.

Training. *Training objective from GT*. Rather than directly predicting particle center locations, MemBrain-pick is trained to estimate a continuous distance field relative to the nearest GT particle position. For each mesh vertex, we compute the Euclidean distance to its closest annotated particle center, generating a smooth distance map that serves as the training target. The network is optimized using a mean-squared error loss, minimizing the discrepancy between predicted and GT distance values.

Data augmentations. To enhance generalization and prevent overfitting, we developed a suite of point cloud augmentations to be applied during training. These augmentations introduce controlled variations in spatial and feature domains, mimicking fluctuations in tomographic appearance:

- Spatial Gaussian smoothing: Smooths point cloud features by computing a weighted average of neighboring points, with Gaussian weights based on spatial distance and a randomly drawn sigma.
- Feature Gaussian smoothing: Applies a Gaussian filter along the normal vector direction to reduce noise in per-vertex features.
- Feature dropout: Randomly sets a fraction of features to zero.
- Feature noise: Adds Gaussian noise to each feature to simulate signal variability in tomographic data.
- Feature shift: Offsets feature values by adding a random constant, introducing brightness shifts.
- Feature scale: Scales feature intensities by a random factor to account for contrast variations.
- Random erasing: Selects small patches in the sample and zeroes out all features within them.
- Random brightness gradient: Scales feature intensities proportionally to the distance from a randomly chosen point within or near the sample.
- Random brightness gamma: Applies a gamma correction, where the gamma value is determined based on distance from a randomly sampled point.

Effects of combinations of these augmentations during training are visualized in Supplementary Fig. 12.

Network architecture. We use DiffusionNet[71], a deep-learning framework designed for learning on mesh surfaces. DiffusionNet propagates per-vertex features using a learned diffusion process, incorporating both geometric and contextual information. We apply it to membrane meshes, where input features are derived from tomographic intensity projections, and training targets are distance maps representing particle localization. To enhance performance, we integrate a learnable 1D convolutional layer that processes features along the membrane normal before diffusion. This is inspired by the concept of separable convolution[114]. In our implementation, this 1D convolutional module consists of two sequential 1D convolution layers (kernel size 3), which we kept fixed for all experiments. Additionally, we stack multiple DiffusionNet blocks to capture both local curvature and broader membrane context.

Inference. *Merging overlapping partitions*. Each partition is processed independently, producing per-vertex particle distance maps. To ensure seamless predictions, we assign each vertex a Gaussian-decaying weight from the partition seed and merge overlapping predictions using a weighted average, similar to a sliding-window approach. This reduces edge artifacts and ensures continuity across partitions.

Membrane-normal distance shifts. To compensate for segmentation inaccuracies, we perform inference multiple times with shifted feature extraction windows. Instead of using a single fixed range along the membrane normal, we iteratively shift the sampled density range in small steps. The final prediction is obtained by averaging across shifts, improving robustness to variations in membrane thickness and segmentation offsets.

Mean shift clustering. After generating particle center distance maps with DiffusionNet, we apply Mean Shift clustering[110,115] to extract particle center locations. First, we threshold predicted distances to remove clear background regions. For the remaining vertices, we refine cluster centers using a score-guided adaptation of Mean Shift, similar to ref. 63, where the position of each vertex p is updated based on a weighted average of its neighbors, as shown in equation (5):

$$m(p) = \frac{\sum_{q\in N_p} k(q-p)\times(1-s_q)\times q}{\sum_{q\in N_p} k(q-p)\times(1-s_q)} \tag{5}$$

Here, N_p includes all points within a radius b around p, excluding p itself, and s_q represents the normalized predicted distance score. The bandwidth b, defining the neighborhood size, is set to match the maximum allowed distance score in our training data. We use Euclidean distance as the kernel function k, and the additional weighting by predicted scores to improve convergence toward high-confidence particle centers.

To accelerate clustering, we use a PyTorch-based GPU implementation[110], substantially reducing inference time. Once mean shift has converged, we apply density-based spatial clustering (DBSCAN[116]) to merge redundant cluster centers and eliminate outliers, refining the final particle localizations.

MemBrain-stats

Particle concentration. Particle concentration is computed as the number of detected particles divided by the total surface area of the membrane mesh. For analyses focused on a single-membrane side, we provide two approaches to ensure accurate estimation: either by dividing the total area by two, assuming particles appear only on one side and both sides are of roughly equal size (that is, not much membrane curvature), or by isolating a single-membrane side surface through edge exclusion (see 'Edge exclusion for robust analysis'), followed by connected component filtering.

Geodesic distance calculation. Geodesic nearest neighbors are computed using either an exact shortest-path algorithm for triangular meshes[117] or an approximate method based on the heat diffusion equation[118]. The latter provides a computationally efficient alternative by leveraging the heat method to estimate geodesic distances. In practice, both approaches yield nearly identical results, with the heat method offering a substantial speed advantage.

Geodesic nearest neighbors. To compute geodesic nearest-neighbor distances, each detected particle center is first projected onto its nearest mesh vertex. Accurate distance estimation requires a sufficiently fine-grained mesh to minimize quantization errors. Once projected, the N nearest geodesic neighbors are determined for each particle center using one of the above methods.

Ripley's statistics. To compute Ripley's statistics, we follow the framework described in PyOrg[64], adapting the formulation to 2D geodesic distances on membrane surfaces, which are modeled as 2D manifolds in 3D space. For a set of membrane particle locations S and a set of triangles T forming the membrane mesh, we define equation (6):

$$\begin{aligned} K(r) &= \pi r^2 \times \frac{\sum_{p\in S} |S\cap B_r(p)|}{\sum_{p\in S} A(T\cap B_r(p))} \\ L(r) &= \sqrt{\frac{K(r)}{\pi}} - r \\ O(r) &= \frac{\sum_{p\in S} |S\cap B_{r,\Delta r}(p)|}{\sum_{p\in S} A(T\cap B_{r,\Delta r}(p))} \end{aligned} \tag{6}$$

where $|.|$ denotes the number of particles in a given region, A(.) computes the area of triangles in a set, $B_r(p)$ is the geodesic ball of radius r around particle p, and $B_{r,\Delta r}(p)$ is the geodesic ring around p, extending from radius r to $r+\Delta r$, as shown in equation (7):

$$\begin{aligned} B_r(p) &= \{q, |, d(q,p) \le r\} \\ B_{r,\Delta r}(p) &= \{q, |, r \le d(q,p) \le r+\Delta r\} \end{aligned} \tag{7}$$

where $d(.,.)$ represents the geodesic distance between two points on the membrane surface.

By restricting area computations to the triangular mesh T, we automatically normalize particle concentration based on the connected membrane surface rather than the entire tomographic volume. As in PyOrg, the functions K,$L$and O are closely related: L stabilizes the variance of K, and O can be understood as the derivative of K, providing information on local particle clustering patterns.

Edge exclusion for robust analysis. Particle localization can become unreliable near the boundaries of membrane segmentations due to the lack of in-plane context. Additionally, segmented membranes are often truncated at arbitrary heights, meaning that segmentation edges do not necessarily correspond to actual membrane boundaries. This can distort spatial metrics such as nearest-neighbor distances. To mitigate these issues, we aim to automatically exclude edge regions from certain analyses.

The Marching Cubes algorithm gives us triangular meshes depicting the segmentations' hulls. As membranes are thin sheets, their edges are represented by regions with high curvature. We detect these areas by computing the discrete mean curvature at each vertex, defined as the average angle between all neighboring triangles within a geodesic sphere of radius R. Membrane edge regions are then approximated by selecting the top 5% of vertices with the highest curvature values. Finally, all points within a distance d from these edge vertices are excluded from the analysis to prevent artifacts in localization and spatial measurements. An example of the edge exclusion is visualized in Extended Data Fig. 10I.

In-plane orientation comparison. To analyze the relative in-plane orientations of nearest-neighbor particles, we first compute geodesic nearest-neighbor pairs as described above. We assume that particle orientations are known, for instance, from STA, where each particle is aligned such that its *z*axis is parallel to the local membrane normal.

To express the particle's orientation in the membrane plane, we construct a rotation matrix that aligns the standard unit vector $(0, 0, 1)^T$ with the membrane normal at the particle's location. This rotation matrix transforms the basis vectors, mapping $(0, 1, 0)^T$ into the membrane plane, providing a reference direction for in-plane orientation.

For a pair of particles, we compare their in-plane orientations by computing the angle θ between their respective in-plane vectors as shown in equation (8):

$$\theta = \cos^{-1}\left(\frac{v_1 \bullet v_2}{||v_1|| \times ||v_2||}\right) \quad (8)$$

where v_1 and v_2 are the in-plane orientation vectors of the two particles, $\bullet$ represents the dot product, and $||.||$ is the vector's L_2-norm. This provides a quantitative measure of orientation similarity within the membrane plane.

Integration of Surface Morphometrics outputs into MemBrain containers. MemBrain-stats enables the joint analysis of membrane geometry and particle localization by integrating outputs from the Surface Morphometrics Pipeline[13] with MemBrain-pick data structures. Because the two tools may represent surface properties on different mesh elements, MemBrain-stats provides the property_from_morphometrics command to transfer Morphometrics-derived values (for example, curvature or membrane distance) onto the vertices of MemBrain meshes. This mapping ensures compatibility between both formats and allows interactive visualization of geometric properties alongside predicted particle positions in Napari. The resulting enriched meshes can be further analyzed using MemBrain-stats's function protein_concentration_wrt_property, which quantifies particle concentration as a function of selected geometric features, linking membrane morphology to the organization of membrane-bound molecular complexes.

Napari tools

Compatibility with Surforama. MemBrain-pick is fully compatible with Surforama, a Napari-based[65] tool for annotating membrane particles in cryo-ET. Similarly to MemBrain-pick's network input, Surforama projects tomographic densities onto the membrane mesh, allowing users to navigate along the membrane-normal vector to visually inspect the in-plane densities of embedded particles. Surforama exports annotations in STAR format[113], which also serves as the preferred input format for training MemBrain-pick. To ensure seamless integration, MemBrain-pick outputs are made compatible with Surforama by generating both STAR files and .h5 containers, which can be directly visualized in Surforama with a single MemBrain-pick command.

Lasso functionality (Lasso, masking, connected components). We developed a MemBrain-pick Napari[65] plugin (napari-lasso-3d) to facilitate the transition from MemBrain-seg outputs (full-tomogram segmentations) to MemBrain-pick inputs (segmented single-membrane instances). The plugin includes a lasso functionality, allowing users to interactively select and extract 3D regions of interest. Following selection, a connected component analysis is performed to isolate individual membrane segments, which can then be saved for further processing. Additionally, users can refine segmentations using standard Napari annotation tools, enabling cleaner and more precise membrane instance selection. For example use cases, see Extended Data Fig. 1b–e.

Applications

MemBrain-seg experiments. All membrane segmentations shown in Fig. 2 and Extended Data Fig. 2 were generated using MemBrain-seg's 'membrain segment' command using the MemBrain_seg_v10_alpha model. Visualizations were generated using ChimeraX[119].

MemBrain-seg ablation studies. To assess the impact of different dataset components, we conducted a fivefold cross-validation experiment (Supplementary Table 1). Each fold consists of patches from distinct tomograms, ensuring no tomogram appears in both the training and test sets at the same time. Given the limited size of our dataset, this cross-validation strategy provides a more robust evaluation of model performance.

For each iteration, the model was trained using four folds and evaluated on the remaining one, cycling through all folds so that each served as the test set once. Performance was measured using the Surface-Dice and standard Dice score, offering a comprehensive assessment of segmentation quality. In the plots (Extended Data Fig. 3), we report the mean and standard deviation across all folds, along with individual run results.

Incremental training. To evaluate how increasing dataset diversity enhances MemBrain-seg's performance and generalization, we incrementally trained models by sequentially incorporating the five rounds of annotations as detailed in 'Iterative dataset generation'. Each step followed the same cross-validation setup described above. The results of this incremental training approach are presented in Extended Data Fig. 3a.

Data augmentations. To analyze the effect of Fourier-based data augmentations, we trained models using the dataset from incremental training round 4, that is, excluding the DeePiCt data. By training on this subset, we specifically examined the impact of augmentations on generalization to the DeePiCt test set, serving as an external generalization dataset. The results, shown in Extended Data Fig. 3b, highlight the improved performance with augmentations.

Fine-tuning with Surface-Dice. To evaluate the performance of fine-tuning, we initialized models using those trained in incremental training round 4 and fine-tuned them with round 5 data (DeePiCt) while ensuring fold separation remained consistent. We tested two fine-tuning strategies: one using Dice and binary cross-entropy loss for optimization and validation tracking, and another incorporating Surface-Dice into the loss function. The impact of these strategies is shown in Supplementary Fig. 8b,c.

TARDIS comparison. To compare MemBrain-seg with TARDIS, we downloaded the latest available TARDIS model for 3D membrane segmentation (downloaded on 6 February 2025). For our quantitative comparison, we applied the model to all our patches using the tardis_mem command to obtain binary segmentations for each patch. For the evaluation presented in Extended Data Fig. 3c, we computed Dice and Surface-Dice scores relative to our GT, excluding regions labeled as 'ignore'.

It is important to note that our GT dataset was generated iteratively, incorporating incremental versions of MemBrain-seg. Although we manually corrected segmentation errors, the dataset still exhibits some bias toward MemBrain-seg predictions. As such, direct comparisons with other programs should be interpreted with caution. For the qualitative comparison presented in Supplementary Fig. 19, we performed the tardis_mem command for the test tomograms presented in Fig. 2.

Ais comparison. The Ais platform provides several downloadable models for segmentation of various structures and facilitates sharing of

models within the community. For the comparison, we used the model #31 (https://aiscryoet.org/models/31/), which was trained for membrane segmentation using tomograms from EMPIAR-11830 (ref. 77).

For test tomograms and test patches, we first rescaled the images to the training pixel size of 15.68 Å. We then padded the volumes with zeros so that all axis lengths were multiples of 16, as required for network inference. After running the model, we removed the padded borders from the predicted segmentations and rescaled the outputs back to the original voxel spacing of 10 Å to ensure comparability with our GT annotations. The evaluation of test patches was performed analogously to the comparison to TARDIS. Results of these comparisons are presented in Extended Data Fig. 3c and Supplementary Fig. 19.

DeePiCt comparison. For comparison with the DeePiCt model, we used the pretrained membrane segmentation model from the DeePiCt GitHub repository (https://github.com/ZauggGroup/DeePiCt/). Following the provided instructions, we first rescaled evaluation patches and test tomograms to a uniform pixel size of 13.48 Å, matching the pixel size used during DeePiCt training. We initially tested DeePiCt's optional Fourier spectrum matching filter by aligning our data to the spectrum of one of the provided DeePiCt tomograms. However, we observed improved results without this preprocessing step and therefore omitted it from our final evaluation. We configured the given config.yml and metadata.csv files, leaving the default parameters and only adjusting file paths. Then, we ran model inference on the evaluation patches and test tomograms using the 'memb' class as semantic prediction class with the 'deploy_local.sh' script. As in the Ais and TARDIS comparisons, predicted segmentations were rescaled back to the original voxel spacing of 10 Å for quantitative evaluation. Results of these comparisons are presented in Extended Data Fig. 3c and Supplementary Fig. 19.

MemBrain-seg fine-tuning example. For the qualitative demonstration of MemBrain-seg's fine-tuning workflow (Extended Data Fig. 5), we first applied the pretrained MemBrain_seg_v10_alpha model to a *T. kivui* tomogram. The initial segmentation erroneously included the outer surface layer in addition to the membrane regions. To correct this, we selected ten representative 3D patches from tomograms of the same dataset and manually corrected their annotations, following the process described in Supplementary Fig. 1. These corrected patches were then used for fine-tuning the model according to the procedure described in 'Model fine-tuning'. After fine-tuning, MemBrain-seg accurately segmented the membrane structures while omitting most of the surface layer, effectively eliminating the false positives observed in the initial prediction. The complete workflow, from initial segmentation to fine-tuned output, is illustrated in Supplementary Fig. 21.

MemBrain-seg preprocessing experiments. For assessing the robustness of MemBrain-seg's results on tomograms preprocessed by different denoising methods as shown in Supplementary Fig. 7, we chose one *C. reinhardtii* tomogram from EMPIAR-11830 (ref. 77; 01122021_BrnoKrios_arctis_lam1_pos4), one *S. oleracea* tomogram from EMPIAR-12612 (ref. 22; batch_tomo_002) and one *Mus musculus* tomogram from EMPIAR-13055 (ref. 70; MTEC_tomo020). All tilt series were processed from raw movie frames automatically with AreTomo3 (ref. 120) using standard parameters to generate CTF-corrected tomograms, including reconstructions pairs from even or odd raw frames. Lamella slab masks were created using Slabify[121]. Deconvolved tomograms using Warp's Wiener-like filter[69] were generated using the implementation from IMOD[122]. IsoNet[40] was run on deconvolved tomograms using the slab masks for 30 iterations and a sub-volume size of 72^3 voxels. Cryo-CARE[36] was trained on each pair of tomograms from even/odd raw frames for 100 epochs with 200 steps each, using subvolumes of 72^3 voxels. DeepDeWedge[41] was trained on each pair of tomograms from even/odd raw frames for 1,000 epochs using subvolumes of 72^3 voxels, also using the lamella slab masks. Finally, MemBrain-seg predictions were made using the MemBrain_seg_v10.2 model for all versions of each tomogram using eightfold test-time augmentation.

MemBrain-pick quantitative evaluation. *Evaluation metric.* To make results comparable with metrics reported in MemBrain (v1)[62], we also adopted the same calculation of F_1-scores to quantify the MemBrain-pick performance: Given a radius r, for a predicted position, we define it a true positive (TP_P) if it is within distance r of a GT position, and a false positive (FP_P) otherwise. Similarly, for a GT position, a (TP_R) is within distance r or a predicted position, and a (FN_R) is not. With this, we can define recall, precision and the F_1-score (harmonic mean of precision and recall), according to equation (9):

$$R = \frac{TP_R}{TP_R + FN_R}$$
$$P = \frac{TP_P}{TP_P + FP_P} \quad (9)$$
$$F_1 = \frac{2 \times R \times P}{R + P}$$

In our experiments, we chose the same radius also used in ref. 63, which is 4.5 bin-4 voxels, corresponding to $4.5 \times 14.08\text{Å} = 63.36\text{Å}$.

Spinach data. The Spinach dataset used to evaluate MemBrain-pick's performance is described in more detail in a separate publication[22]. To train the MemBrain-pick model used in Fig. 3c, we used the same dataset as in MemBrain (v1)[62], consisting of 45 membrane segmentations from 9 tomograms of *S. oleracea* thylakoid membranes. Previously existing GT annotations used in MemBrain v1 were manually refined to improve accuracy and to classify particles into three categories: PSII, cytochrome b6f (b6f) and unidentifiable densities (UK).

Following the protocol in MemBrain v1, we designated all 10 membranes from 2 tomograms as the test dataset. For the incremental training performance plot in Fig. 3d, 5 membranes from the remaining 35 (fixed across runs) were assigned to the validation set. Training sets were constructed by randomly sampling N membranes from the remaining 30, where for each value of N, 5 independent subsets were drawn, and a separate model was trained for each subset. The same dataset splits were used for training both MemBrain-pick and MemBrain v1 models.

Chlamy generalization

To evaluate the generalization performance of MemBrain-pick, we applied a model trained exclusively on *S. oleracea* data to two independent *C. reinhardtii* datasets, acquired under different imaging conditions but both depicting thylakoid membranes with embedded PSII and b6f complexes (Fig. 3e,f). Details on the acquisition and setup of the $Chlamy_{old}$[17] and $Chlamy_{new}$[77] datasets can be found in their respective publications.

For $Chlamy_{old}$, we used 28 segmented membranes extracted from 4 tomograms, along with GT annotations from the original publication[17], enabling the computation of F_1-scores (Fig. 3e). The Spinach-trained MemBrain-pick model was applied directly to these membranes to predict particle positions.

For $Chlamy_{new}$, we first segmented all membranes in four tomograms by applying MemBrain-seg to Cryo-CARE-denoised[36] tomograms. Using the MemBrain Napari plugin, we extracted 23 thylakoid membranes from these full-tomogram segmentations. To further enhance image interpretability, we applied IsoNet[40] before predicting particle positions with the Spinach-trained MemBrain-pick model (Fig. 3f).

Phycobilisomes

For the analysis presented in Fig. 4, we used data from red algae[75], which depicts phycobilisome–PSII supercomplexes (EMD-31244).

Membrane segmentations were generated using MemBrain-seg (MemBrain_seg_v10_alpha model), followed by connected component splitting via the MemBrain Napari plugin, yielding 29 individual membrane segmentations. To provide GT annotations, we manually labeled phycobilisome positions on six membranes using Surforama and trained MemBrain-pick with its default parameters. The trained model was then applied to the remaining 23 membranes in the tomogram, enabling us to compute Ripley's statistics on this dataset. For computing Ripley's *O* statistic, we divided all distances into bin sizes of 5 nm.

Ribosome analysis in EMPIAR-11830

MemBrain-pick prediction. For the ribosome analysis shown in Fig. 5, we extracted particle positions from the *C. reinhardtii* dataset (EMPIAR-11830)[77], focusing on the outer membrane of the nuclear envelope and the endoplasmic reticulum. To scale up the analysis, MemBrain-seg was applied to 140 tomograms (MemBrain_seg_v10_alpha model), extracting the largest connected component from each segmentation. After removing segmentations containing membrane structures of other classes or excessive noise, 97 tomograms remained for further processing. Ribosome GT annotations were manually assigned to 13 membranes using Surforama, and MemBrain-pick was trained with its default parameters. The trained model was then used to predict ribosome positions across the remaining membranes, yielding 4,515 predicted positions. Both MemBrain-seg and MemBrain-pick were applied to Cryo-CARE-denoised[36] tomograms to enhance contrast and improve localization accuracy.

STA. The endoplasmic reticulum-bound and nuclear envelope-bound ribosomes from the MemBrain-pick predictions were further processed in STA. Averaging was performed in STOPGAP[52]. Bin-4 3D CTF-corrected tomograms were generated as implemented in IMOD[122], from which subtomograms for all ribosome positions were extracted with a box size of 80 pixels. Because all particles were pre-aligned normal to the membrane from MemBrain-pick, initial alignment was performed using a 'cone search', which corresponds to a global search around one axis (perpendicular to the membrane) while restricting the search for the other two angles. After initial alignment, all angular searches were iteratively reduced as resolution increased, reaching 22.7-Å resolution after 14 iterations. The Fourier shell correlation plot using a soft spherical mask, with correction for mask-induced correlations[123], is visualized in Supplementary Fig. 13.

Ribosome chains. For the analysis shown in Fig. 5e, we first computed the three nearest neighbors for each ribosome and determined the in-plane angles between them using MemBrain-stats, based on orientation estimates from STA. We plotted the smallest of these three angles for each ribosome position and compared it to a theoretical distribution. The theoretical distribution was generated by randomly sampling 100,000 triplets of uniformly distributed angles between 0° and 180° and plotting their distribution. Next, we extracted ribosome chains by iteratively adding chain units based on three spatial constraints: (1) a center-to-center distance below 40 nm, (2) an in-plane angle below 60°, and (3) an angle between the current chain end-connection and a potential new end-connection below 80°. Using these criteria, we identified and reconstructed the ribosome chains shown in Fig. 5e. An example Jupyter notebook showing the extraction of these ribosome chains can be accessed on the MemBrain-stats GitHub repository.

Note that we used three nearest neighbors instead of only the first nearest neighbor, as the closest ribosomes often did not appear to be part of the same chain (Fig. 5e). This approach allowed for a more reliable identification of polysome-like arrangements.

Ribosomes in EMD-10409

To generate the visualization of endoplasmic reticulum-bound ribosomes in Extended Data Fig. 8b, we processed the tomogram from EMD-10409 (ref. 4) using MemBrain-seg and MemBrain-pick. All membranes in the tomogram were first segmented using MemBrain-seg (MemBrain_seg_v10_alpha model). The segmented membrane was then divided into six nonoverlapping regions using the MemBrain Napari tool. Ribosome GT annotations were manually assigned to three of these regions in Surforama, which were subsequently used for training MemBrain-pick. The trained model was then applied to the remaining three regions to predict ribosome positions, as shown in Extended Data Fig. 8b. For MemBrain-seg, the raw tomogram was used directly, whereas for MemBrain-pick, we applied IsoNet[40] to enhance contrast and improve particle localization.

Respirasomes

To evaluate MemBrain-pick's performance on mitochondrial membrane-associated complexes, we analyzed respirasomes in *C. reinhardtii* mitochondria in tomograms from EMPIAR-11830 (ref. 77). Membrane segmentations were generated from four tomograms using MemBrain-seg (MemBrain_seg_v10_alpha model), and 45 individual cristae segmentations were extracted using the MemBrain Napari plugin. To provide GT particle positions, we used previously generated template-matching-based coordinates[19], projected them onto the membrane surfaces, and used these as training data for MemBrain-pick. The model was trained on 35 membranes from 3 tomograms and then applied to the remaining 10 membranes from the fourth tomogram (Extended Data Fig. 8a). All tomograms were denoised using Cryo-CARE[36] before membrane segmentation and particle localization.

SARS-CoV-2 spike protein localization

For the localization of severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) spike proteins (Extended Data Fig. 9c), we segmented tomograms EMD-11863 and EMD-11865 using the MemBrain_seg_v10_alpha model. We then cropped out fully formed viral particles that were detached from the endoplasmic reticulum membrane using our napari-lasso-3d tool (see 'Napari tools'). To facilitate manual spike protein annotation in Surforama, we dilated the viral membrane segmentation by 10 voxels. Using Surforama, we manually annotated the surface protein positions of 10 segmented virions from tomogram EMD-11865, which served as GT for training MemBrain-pick. The trained model was subsequently applied to five segmented virions from EMD-11863, producing the predictions shown in Extended Data Fig. 9c. Both tomograms were preprocessed with IsoNet to enhance contrast and mitigate missing wedge artifacts before particle localization.

Surface morphometrics application

In Extended Data Fig. 9, we demonstrate the compatibility of MemBrain-seg outputs with downstream quantitative analysis tools such as the Surface Morphometrics Pipeline[13]. For all three tomograms shown (EMD-10766, EMD-12329 and EMD-11863), membranes were segmented using MemBrain_seg_v10_alpha and cropped to the regions of interest using our napari-lasso-3d tool (see 'Napari tools'). The cropped segmentations were then processed in the Morphometrics pipeline with minimal adjustments to the default configuration (max_triangles = 100,000, extrapolation_distance = 2 nm, neighbor_count = 100, radius_hit = 15 nm, min_component = 1,000, exclude_borders = 0). The resulting outputs provided curvature values (via PyCurv) and intermembrane distances, which were visualized directly in Extended Data Fig. 9a,b.

For Extended Data Fig. 9c,d, we further combined Surface Morphometrics outputs with MemBrain-pick predictions to enable joint geometric and protein-density analyses. Using MemBrain-stats, the Surface Morphometrics properties were mapped onto the MemBrain-pick mesh vertices via the property_from_morphometrics command (see 'Integration of morphometrics outputs into MemBrain containers'). The resulting enriched containers were visualized in surforama (Extended Data Fig. 9c) and analyzed using the distance to the nearest

surrounding endoplasmic reticulum membrane as the morphometrics property of interest, which revealed the protein complex distribution shown in Extended Data Fig. 9d.

MemBrain-pick comparisons to other methods

In Fig. 3 and Supplementary Fig. 10, we compare the performance of MemBrain-pick with other particle-picking methods commonly used in cryo-ET. Below, we provide details on the applications of these methods using our Spinach dataset (see 'Spinach data').

Template matching (PyTOM). Template matching was performed using pytom-match-pick[51] on the two test-set Spinach tomograms. The single-particle cryo-EM map of spinach PSII (EMD-6617)[124] was chosen as a template. The template was resampled to a pixel size of 14.08 Å to match that of the tomogram, and cropped to a box size of 32 pixels. Finally, the contrast was inverted to match that of the tomograms (protein density being black). A slab mask for the tomogram was created using Slabify[121]. The program pytom_match_template.py was run using a fan (per-tilt) wedge model, a particle diameter of 310 Å (corresponding to the longest axis of PSII), a high-pass filter of 400 Å, C2 symmetry and random phase correction. The angular sampling was calculated automatically from the particle diameter and pixel size, corresponding to 4.89° along $Z1$ and X, and 5.14° along $Z2$ (Euler angles in PyTOM conventions).

For extraction, the score maps were masked with individual single-membrane masks of the test membranes. Automated extraction using pytom_extract_candidates.py with the --number-of-false-positives option failed to provide meaningful results, so a custom procedure was adopted. For each membrane, the threshold was calculated by finding the 99th percentile of the scores within the mask. This roughly corresponds to the noise threshold when sorting all the scores within the mask (Supplementary Fig. 10d). A script for this calculation (tm_find_thresh) is provided at https://github.com/CellArchLab/cryoet-scripts/. Then pytom_extract_candidates.py was used to extract up to 100 candidates above this threshold per membrane, with an exclusion radius of 6 pixels (84.48 Å) around each peak, corresponding to the shortest axis of PSII.

DeepFinder. To train DeepFinder[55], we performed a training-validation split at the tomogram level to prevent overlapping patches within a single tomogram. Of 9 tomograms, 5 were used for training, 2 for validation, and the remaining 2 were reserved for testing (same as in the split described for the quantitative MemBrain-pick evaluation), resulting in a training set of 25 membranes and a validation set of 11 membranes.

GT segmentation masks were generated using the spheres approach, where spheres with a radius of 4 voxels were placed at the 3D GT positions. DeepFinder was trained using a patch size of 40 voxels for 100 epochs with 20 steps per epoch and a step size of 9 voxels for random shifts. After training, segmentation masks were predicted for the test tomograms, and particle positions were extracted using DeepFinder's clustering module with a bandwidth parameter of 3 voxels.

All mentioned hyperparameters were optimized through grid search using the validation tomograms. To associate predicted particle positions with membranes, we considered only positions within a maximum distance of 4 voxels from the membrane surface.

CrYOLO. For CrYOLO[53], we used the same data split as for DeepFinder, ensuring nonoverlapping training and validation patches per tomogram. GT positions were converted into CrYOLO-compatible format by generating .cbox files containing bounding box coordinates. A box size of 10 voxels was chosen, with bounding boxes propagated across three slices above and below each GT position.

Hyperparameter tuning was performed on the validation dataset to determine the optimal settings, resulting in the following parameters: threshold = 0.05, tracing_min_length = 3, distance = 6, min_num_boxes = 5 and positive_weight = 50.

Membrane–position associations were performed following the approach used for DeepFinder, considering only predicted positions within 4 voxels of the membrane surface.

MPicker/EPicker. For the comparison with MPicker[59] and EPicker[125], we first flattened our membranes using the MPicker GUI. This involved manually selecting several points per membrane on the luminal side of the segmentation to define a reference surface on the correct membrane side. MPicker's built-in functionality was then used to flatten the membrane, generating a 2D image stack representation. Next, we utilized MPicker's 'Load raw coordinates' function to import a text file containing our GT positions for PSII and UK particles. MPicker transformed these 3D coordinates to their corresponding positions on the flattened membrane stack (Supplementary Fig. 10). Finally, for each membrane, we saved the flattened membrane stack as a .mrc file and the transformed GT positions as a .txt file.

To train EPicker, we needed pairs of 2D images (.mrc files) and their corresponding 2D coordinates (.thi files). These were prepared by extracting the 2D slice from the flattened membrane array that contained the most particle center positions. Additionally, we included the coordinates from the adjacent slices in both directions along the z axis, as the center slice still retained enough particle density to confirm the presence of these particles.

EPicker training was conducted using a learning rate of 0.0001, a batch size of 4 and 140 epochs, executed via the script epicker_train.sh. We trained separate models for each of our four training folds, matching the same folds used in the MemBrain-pick and MemBrain v1 analyses. After training, we predicted positions on the validation membranes and optimized the parameters --thres (score thresholding) and --dist (minimum distance between detected particles) using the Mpicker epicker_batch.py script. These parameters were fine-tuned separately for each data fold.

Once the optimal parameters were determined, we applied EPicker to our test set. MPicker then converted the predicted 2D coordinates back into 3D tomogram coordinates, enabling direct comparison with the GT positions. This allowed us to compute performance metrics and assess the accuracy of the MPicker–EPicker pipeline relative to our GT annotations.

MiLoPYP. For MiLoPYP, we used the same data split as for DeepFinder and CrYOLO. We converted our PSII GT positions into .txt center coordinate files to match the MiLoPYP format. During training, we performed a grid search over the bbox_size parameter, varying it from 12 to 20. After training, we performed an additional grid search on the validation set to find the best score threshold value thres. These searches resulted in the parameters of bbox_size = 16 and thres = 0.8. However, when applying the trained model to the test tomograms, we observed that the score distribution differed substantially from that of the validation set. Therefore, we selected the final threshold based on qualitative inspection of the test predictions, yielding thres = 0.45.

To compare MemBrain-pick with MiLoPYP, we made minor adjustments to the code: As processing our full tomograms (bin-4 shape: (928, 928, 464)) in a single step was not feasible due to GPU memory constraints, we implemented a sliding-window approach, which divided the tomogram into patches of size 232^3 with a step size of 116 in all axes, allowing for largely overlapping patches.

Membrane–position associations were performed following the approach used for DeepFinder, considering only predicted positions within 4 voxels of the membrane surface.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

All tomograms used in this study are publicly available. The latest version of our MemBrain-seg training dataset is accessible on Zenodo via https://doi.org/10.5281/zenodo.15089685 (ref. 126). For MemBrain-seg demonstrations, we used tomograms from: CZII-DS-10007, CZII-DS-10224, CZII-DS-10442, CZII-DS-10443, CZII-DS-10444, EMD-3977, EMD-4869, EMD-12329, EMD-12727, EMD-12749, EMD-15407, EMD-16084, EMD-18306, EMD-18748, EMD-30364, EMD-35019, EMD-43050, EMD-44176, EMD-50605, EMPIAR-10988, EMPIAR-11058, EMPIAR-11370, EMPIAR-11830, EMPIAR-12612 and EMPIAR-13055. For MemBrain-pick training and evaluation, we used tomograms from EMPIAR-12612, EMD-10780–EMD-10783, EMPIAR-11830 and EMD-10409. To showcase the entire MemBrain v2 workflow, we used data from EMPIAR-11830 and EMD-31244. The GT annotations used for performing the efficiency analysis in MemBrain-pick are available on Zenodo via https://doi.org/10.5281/zenodo.15090084 (ref. 127). The GT annotations for performing the fine-tuning on *T. kivui* tomograms are available on Zenodo via https://doi.org/10.5281/zenodo.17737066 (ref. 128). The subtomogram average of a *Chlamydomonas* nuclear envelope-bound ribosome has been deposited under EMD-54837. Source data are provided with this paper.

Code availability

The reviewed code associated with this manuscript has been deposited on Zenodo via https://doi.org/10.5281/zenodo.20870721 (ref. 129). The full MemBrain v2 program is also pip-installable via PyPI (pip install membrain) and via https://github.com/CellArchLab/MemBrain-v2. Individual modules can be accessed via the following repositories, each of which contains detailed documentation: https://github.com/teamtomo/membrain-seg/, https://github.com/CellArchLab/membrain-pick/, https://github.com/CellArchLab/membrain-stats/ and https://github.com/CellArchLab/napari-lasso-3d/.

References

96. Isensee, F., Jaeger, P. F., Kohl, S. A. A., Petersen, J. & Maier-Hein, K. H. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nat. Methods* **18**, 203–211 (2021).
97. Cardoso, M. J. et al. MONAI: an open-source framework for deep learning in healthcare. Preprint at https://doi.org/10.48550/arXiv.2211.02701 (2022).
98. Wang, L., Lee, C. -Y., Tu, Z. & Lazebnik, S. Training deeper convolutional networks with deep supervision. Preprint at https://doi.org/10.48550/arXiv.1505.02496 (2015).
99. Shorten, C. & Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *J. Big Data* **6**, 60 (2019).
100. Tellez, D. et al. Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Med. Image Anal.* **58**, 101544 (2019).
101. Wagner, S. J. et al. Structure-preserving multi-domain stain color augmentation using style-transfer with disentangled representations. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021* 257–266 https://doi.org/10.1007/978-3-030-87237-3_25 (Springer, 2021).
102. Shit, S. et al. ClDice - a novel topology-preserving loss function for tubular structure segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* https://doi.org/10.1109/cvpr46437.2021.01629 (IEEE, 2021).
103. Settles, B. *Active Learning Literature Survey* (Tech Rep, 2009).
104. Wolf, I. et al. The medical imaging interaction toolkit (MITK): a toolkit facilitating the creation of interactive software by extending VTK and ITK. In *Medical Imaging 2004: Visualization, Image-Guided Procedures, and Display* (ed. Galloway, R. L., Jr) https://doi.org/10.1117/12.535112 (SPIE, 2004).
105. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
106. Wang, G. et al. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing* **335**, 34–45 (2019).
107. Zhuang, F. et al. A comprehensive survey on transfer learning. *Proc. IEEE Inst. Electr. Electron. Eng.* **109**, 43–76 (2021).
108. Siggel, M., Jensen, R. K., Maurer, V. J., Mahamid, J. & Kosinski, J. ColabSeg: An interactive tool for editing, processing, and visualizing membrane segmentations from cryo-ET data. *J. Struct. Biol.* **216**, 108067 (2024).
109. Jiménez de la Morena, J. et al. ScipionTomo: towards cryo-electron tomography software integration, reproducibility, and validation. *J. Struct. Biol.* **214**, 107872 (2022).
110. Zhao, M. et al. Faster Mean-shift: GPU-accelerated clustering for cosine embedding-based cell segmentation and tracking. *Med. Image Anal.* **71**, 102048 (2021).
111. Lorensen, W. E. & Cline, H. E. Marching Cubes: a high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* https://doi.org/10.1145/37401.37422 (ACM, 1987).
112. Sullivan, C. & Kaszynski, A. PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). *J. Open Source Softw.* **4**, 1450 (2019).
113. Hall, S. R. The STAR file: a new format for electronic data transfer and archiving. *J. Chem. Inf. Comput. Sci.* **31**, 326–333 (1991).
114. Mamalet, F. & Garcia, C. Simplifying ConvNets for fast learning. In *Artificial Neural Networks and Machine Learning – ICANN 2012* 58–65 https://doi.org/10.1007/978-3-642-33266-1_8 (Springer, 2012).
115. Cheng, Y. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **17**, 790–799 (1995).
116. Ester, M., Kriegel, H., Sander, J. & Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD* **96**, 226–231 (1996).
117. Mitchell, J. S. B., Mount, D. M. & Papadimitriou, C. H. The discrete geodesic problem. *SIAM J. Comput.* **16**, 647–668 (1987).
118. Sharp, N. potpourri3d https://github.com/nmwsharp/potpourri3d (2024).
119. Pettersen, E. F. et al. UCSF ChimeraX: structure visualization for researchers, educators, and developers. *Protein Sci.* **30**, 70–82 (2021).
120. Peck, A. et al. AreTomoLive: automated reconstruction of comprehensively corrected and denoised cryo-electron tomograms in real time and at high throughput. *Nat. Methods* **23**, 1121–1125 (2026).
121. Righetto, R. & Lamm, L. CellArchLab/Slabify-et: v0.3.0. *Zenodo* https://doi.org/10.5281/ZENODO.13941081 (2024).
122. Mastronarde, D. N. Accurate, automatic determination of astigmatism and phase with Ctfplotter in IMOD. *J. Struct. Biol.* **216**, 108057 (2024).
123. Chen, S. et al. High-resolution noise substitution to measure overfitting and validate resolution in 3D structure determination by single particle electron cryomicroscopy. *Ultramicroscopy* **135**, 24–35 (2013).
124. Wei, X. et al. Structure of spinach photosystem II–LHCII supercomplex at 3.2 Å resolution. *Nature* **534**, 69–74 (2016).
125. Zhang, X., Zhao, T., Chen, J., Shen, Y. & Li, X. EPicker is an exemplar-based continual learning approach for knowledge accumulation in cryoEM particle picking. *Nat. Commun.* **13**, 2468 (2022).
126. Lamm, L., Zufferey, S., Wietrzynski, W., Righetto, R. D. & Engel, B. MemBrain-seg training dataset (v0.0.1). *Zenodo* https://doi.org/10.5281/zenodo.15089686 (2025).

127. Lamm, L., Wietrzynski, W., Righetto, R. D. & Engel, B. MemBrain-pick Spinach Thylakoid Evaluation Data. *Zenodo* https://doi.org/10.5281/zenodo.15090084 (2025).
128. Lamm, L., Zufferey, S., Wietrzynski, W., Righetto, R. D. & Engel, B. MemBrain-seg fine-tuning dataset. *Zenodo* https://doi.org/10.5281/zenodo.17737066 (2025).
129. Lamm, L., Righetto, R. D. & Engel, B. MemBrain v2 codebase (v1.0). *Zenodo* https://doi.org/10.5281/zenodo.20870721 (2026).

Acknowledgements

We are grateful to everyone who tested and provided feedback on the beta version of MemBrain v2, including M. Pöge, R. Brandt, P. Dutka, W. (Seven) Guo, B. Gallet, T. O' Sullivan, X. Zhu, Z. Hou, P. Zhang, V. Salo, T. Hoffmann, G. Zanetti, J. Anstatt, J. Sachweh, A. (Bram) Koster, Y. Chang, Y. Bykov, P. V. Jadhav, D. Shepherd and C. Papantoniou. We thank A. Kotecha and team at Thermo Fisher Scientific for leading the *Chlamydomonas* visual proteomics project that generated much of the raw data (EMPIAR-11830) used for initial training of MemBrain-seg. We thank the users who contributed training patches and annotations to improve MemBrain-seg's generalizability: V. Ananda, P. P. Navarro, B. Wimmer, O. Medalia, Y. -T. (Atty) Chang, M. Medina, B. Barad, D. Grotjahn, M. J. J. Limlingan, M. Pilhofer, L. Thärichen, L. de Jager, M. Chaillet, F. Förster, B. Silva, D. 'Raphael' Park, R. H. James, T. Marlovits, S. Klumpe, J. Redzovic, F. Koch, G. Tagiltsev and J. Briggs. We acknowledge D. Litvinov for helpful contributions during MemBrain's development. These contributions from the cryo-ET community have substantially improved MemBrain-seg's capabilities. Calculations were performed at sciCORE (https://scicore.unibas.ch/) scientific computing center at the University of Basel.

Author contributions

Conceptualization by T.P., B.D.E. and L.L, with support from A.M.-S. in the design of Surface-Dice and skeletonization. Development of main functionalities by L.L. Software implementations by L.L., R.D.R. and H.Z., with support from K.Y., A.B., Y.L., S. Ziegler and F.I. Data acquisition and initial membrane annotation by W.W. Further patch annotations by S. Zufferey, L.L. and S. Ziegler. Testing and suggestion of features by W.W., R.D.R., F.W. and S. Zufferey. STA by F.W., tomogram preprocessing and template matching by R.D.R. Supervision by T.P., B.D.E., J.A.S. and A.M.-S. Original draft writing by L.L. Review and editing of the paper by T.P., B.D.E. and R.D.R., with feedback from all authors.

Funding

L.L. discloses support from the Munich School for Data Science (MUDS) and a fellowship from Boehringer Ingelheim Fonds. Parts of this work were funded by Helmholtz Imaging (HI), a platform of the Helmholtz Incubator on Information and Data Science. A.M-S. discloses support from grants RYC2021-032626-I and CNS2023-144921 funded by MICIU/AEI/10.13039/501100011033 and the European Union by NextGenerationEU/PRTR, and grant PID2023-151075OA-I00 funded by MICIU/AEI/10.13039/501100011033 and FEDER, EU. A.M-S. is also supported by grant FSRM/10.13039/100007801 (22686/PI/24) funded by Fundación Séneca and the Universidad de Murcia through its program AttractRyC 2023. B.D.E. discloses support from an HFSP Research Grant (RGP0005/2021), a Swiss National Science Foundation project grant (310030E_217510, in collaboration with DFG FOR 5573/1) and ERC consolidator grant 'cryOcean' (fulfilled by the Swiss State Secretariat for Education, Research and Innovation, M822.00045). A.B. discloses support from the Medical Research Council (MC_UP_1201/6). Open access funding provided by Helmholtz Zentrum München - Deutsches Forschungszentrum für Gesundheit und Umwelt (GmbH).

Competing interests

The authors declare no competing interests.

Additional information

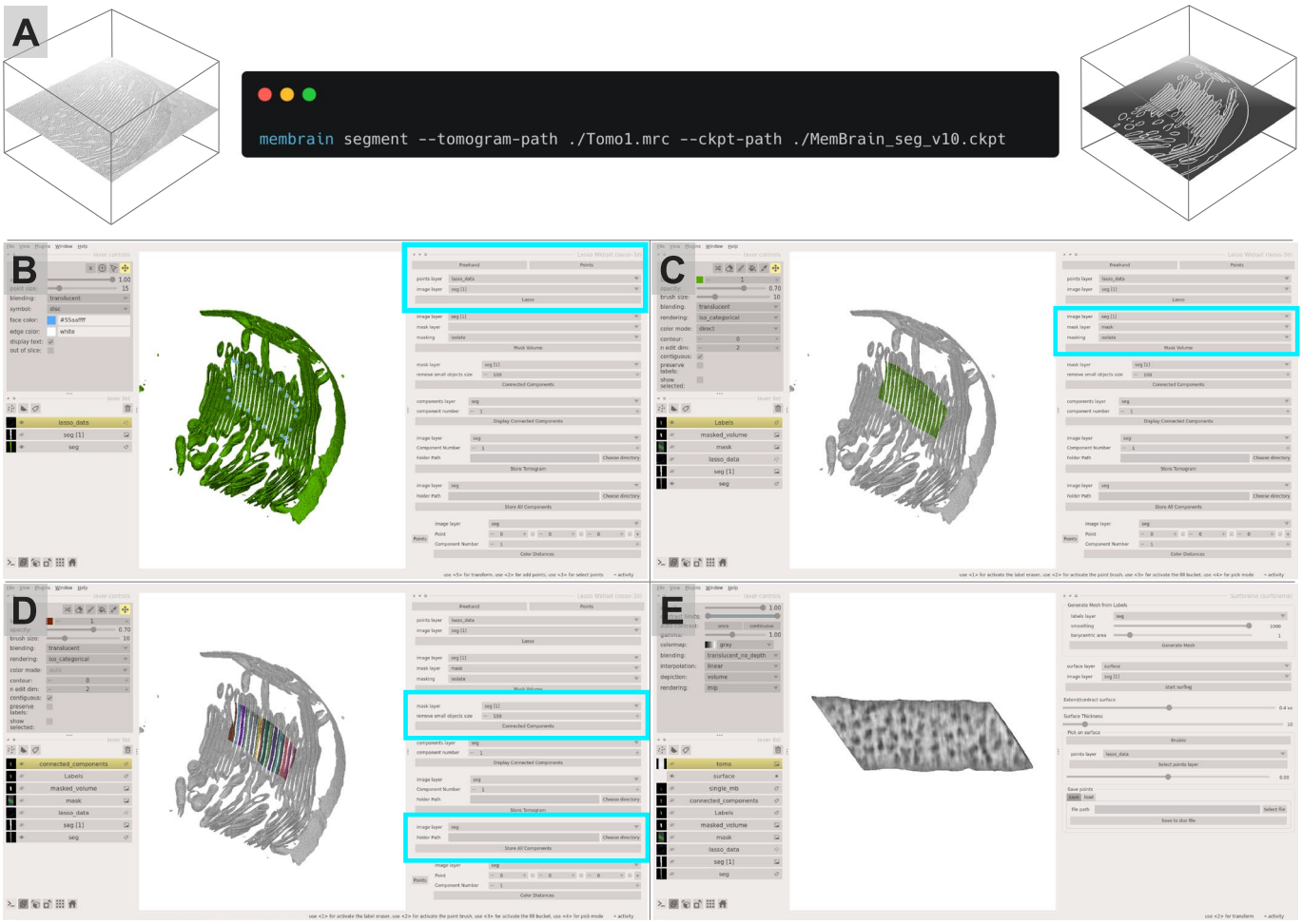
Extended data is available for this paper at https://doi.org/10.1038/s41592-026-03178-8.

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41592-026-03178-8.

Correspondence and requests for materials should be addressed to Lorenz Lamm, Benjamin D. Engel or Tingying Peng.

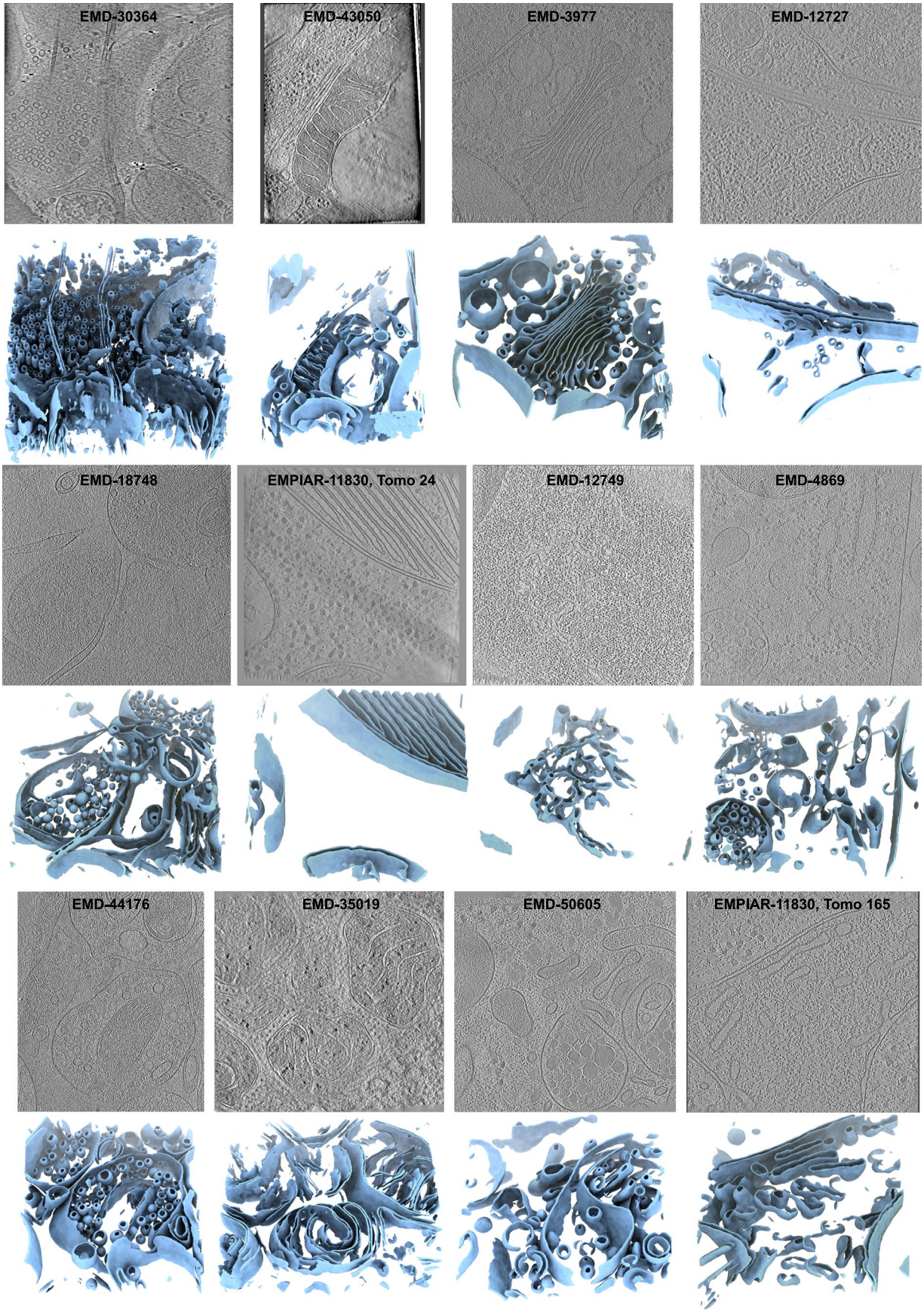
Peer review information *Nature Methods* thanks Benjamin Barad and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Primary Handling Editor: Arunima Singh, in collaboration with the *Nature Methods* team.

https://doi.org/10.1038/s41592-026-03178-8



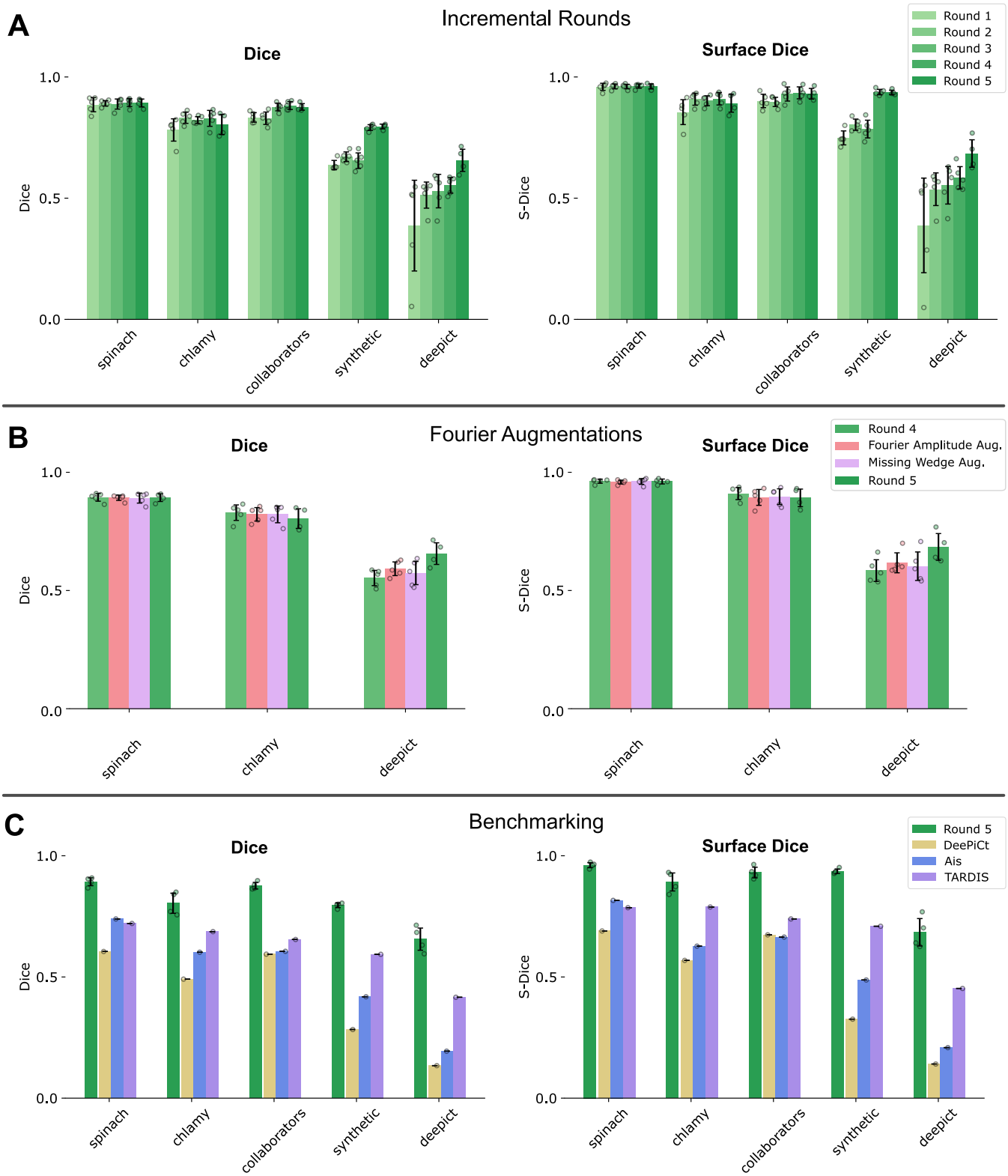
Extended Data Fig. 1 | Napari plugins. A: MemBrain-seg can be used with a single command line membrain segment and outputs binary segmentations from a tomogram. **B-D:** MemBrain lasso plugin. **B:** Click and drag to draw a 3D lasso to select an area of interest. **C:** Either isolate or remove the selected area from the segmentation. **D:** Extract connected components of the remaining segmentation to find the membrane instances of interest, and save them out as .mrc files. **E:** Extracted membranes can be visualized in Surforama.

https://doi.org/10.1038/s41592-026-03178-8



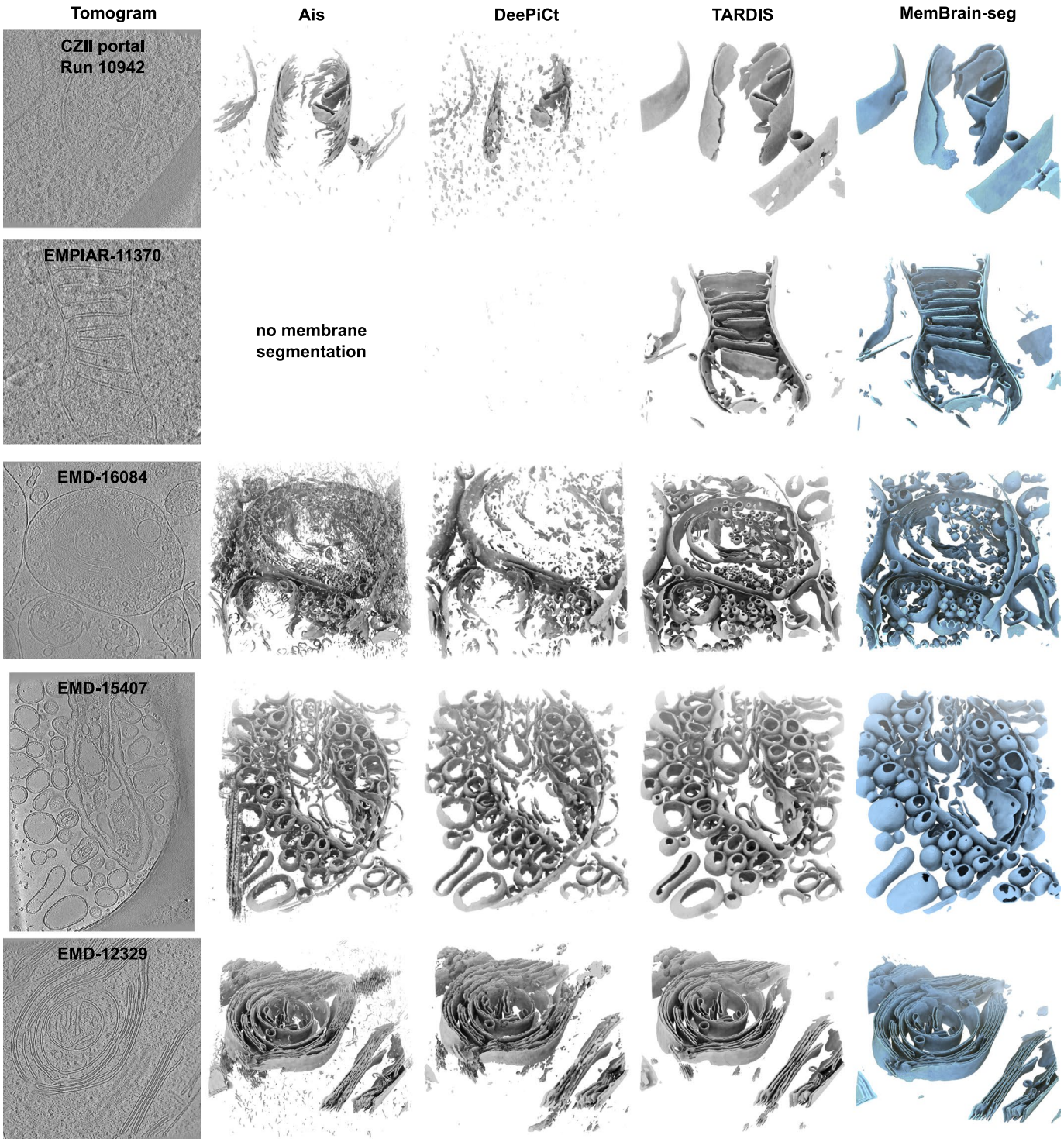
Extended Data Fig. 2 | More examples of MemBrain-seg segmentations from different datasets and data sources. For each EMPIAR or EMDB dataset shown, top row: slice through tomogram, bottom row: corresponding MemBrain-seg prediction (light blue).

https://doi.org/10.1038/s41592-026-03178-8



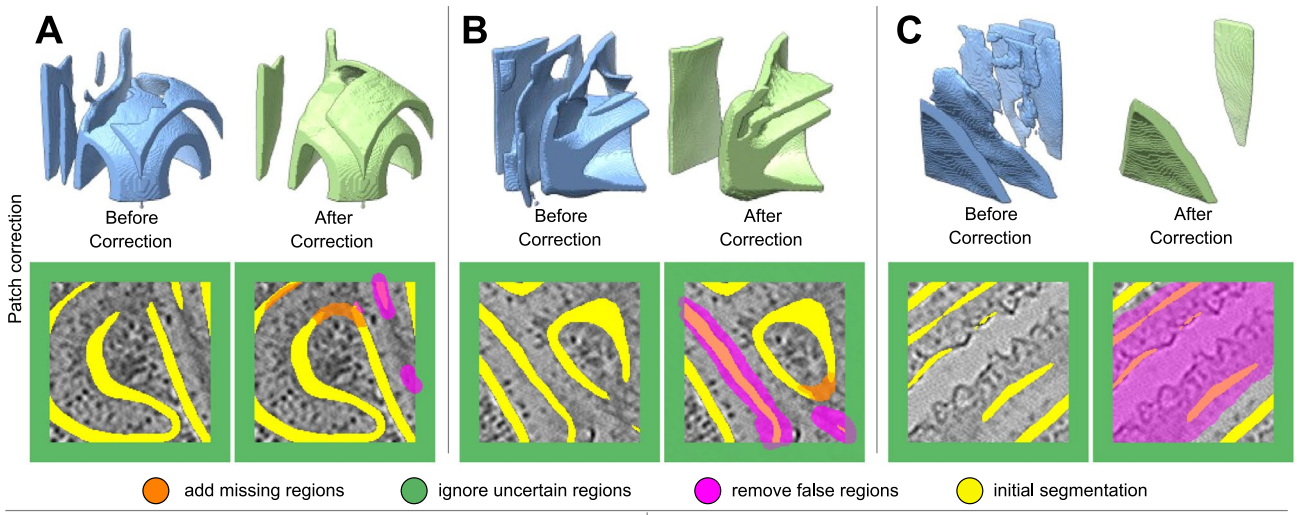
Extended Data Fig. 3 | MemBrain-seg Evaluations. A: Evaluation of incremental training performance on different datasets with Dice and Surface-Dice scores. MemBrain-seg models were trained with increasingly bigger training sets from rounds 1 to 5. **B:** Evaluation of the effect of our Fourier-based augmentations on generalization performance. Both models for Fourier Amplitude and Missing Wedge augmentations were trained with data from incremental training round 4, and compared to plain incremental training with rounds 4 and 5. **C:** Benchmarking of MemBrain-seg with other pretrained models, evaluated on our entire dataset. All means (bar height), standard deviations (error bars) and single points correspond to runs of the 5-fold cross-validation per experiment for the MemBrain-seg bars. TARDIS, Ais, and DeePiCt were evaluated only once on the entire dataset.

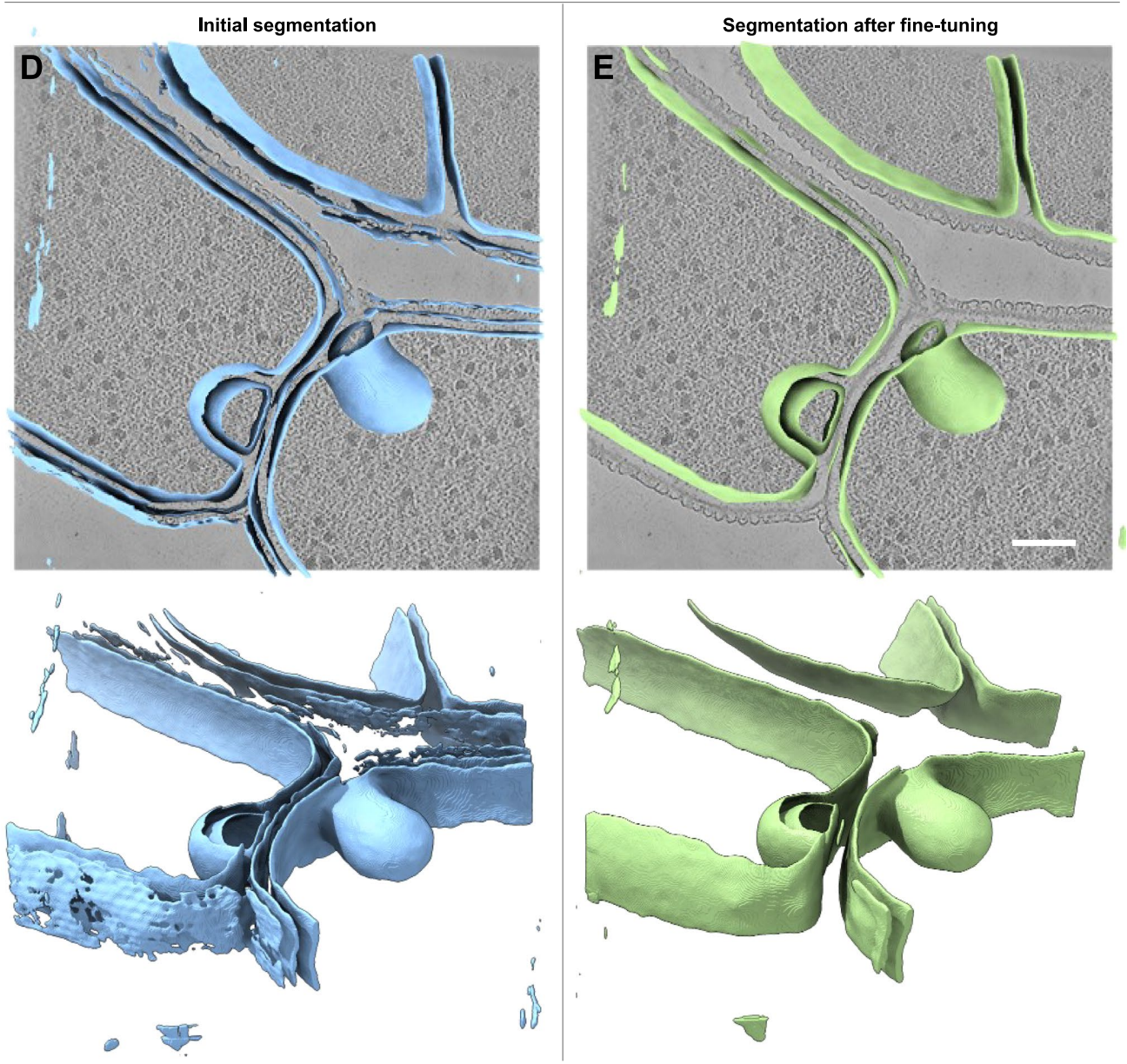
https://doi.org/10.1038/s41592-026-03178-8



Extended Data Fig. 4 | Qualitative Comparison of MemBrain-seg with other pretrained membrane segmentation models. Depicted are the same tomograms as in Fig. 2, segmented using Ais, DeePiCt, TARDIS, and MemBrain-seg Shown are the same tomograms as in Fig. 2, together with membrane segmentations generated using Ais[48], DeePiCt[45], TARDIS[49], and MemBrain-seg. For each tomogram, the raw slice is displayed on the left, followed by the corresponding 3D membrane renderings for each method. Ais did not provide membrane segmentations for EMPIAR-11370.

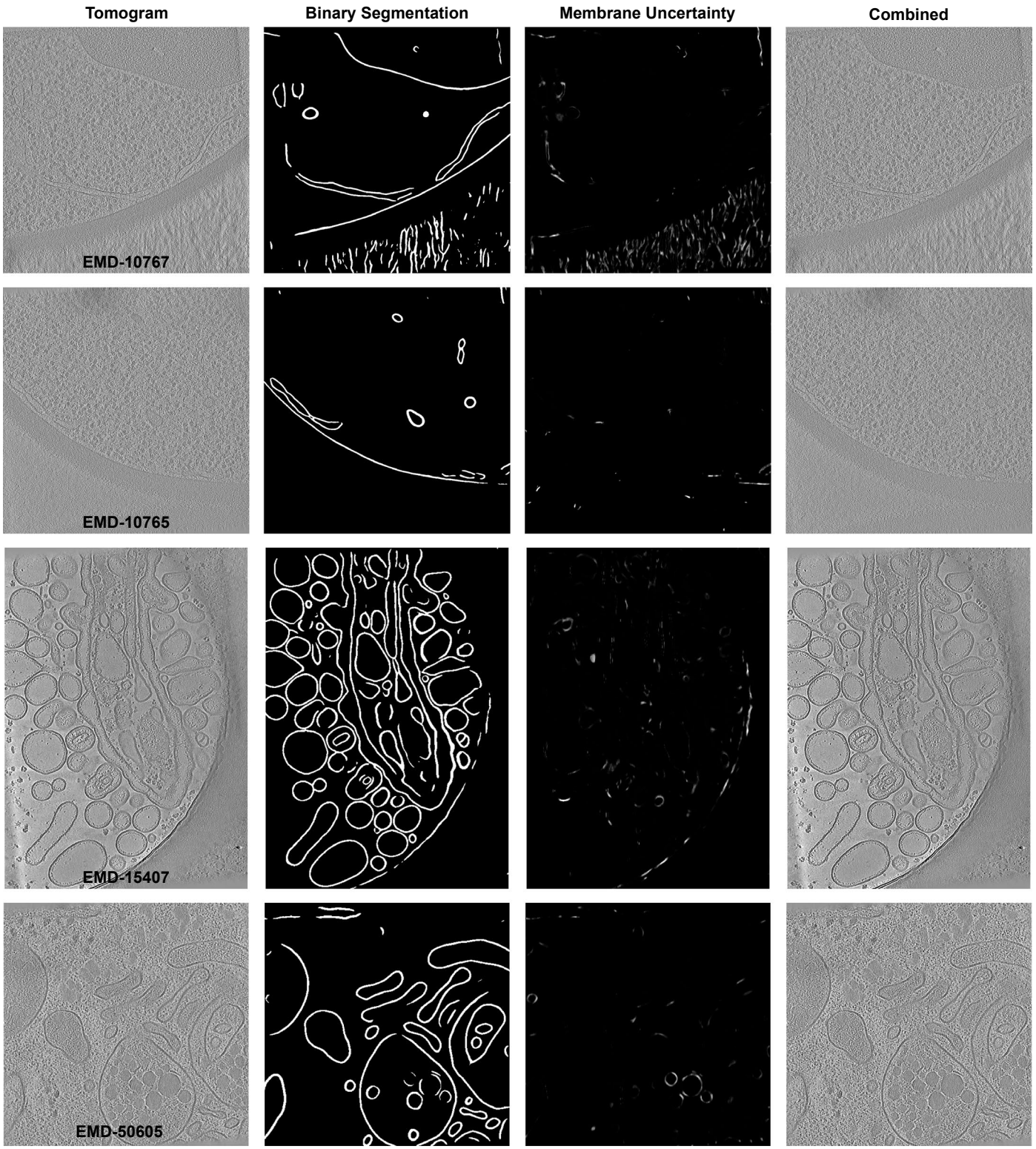
https://doi.org/10.1038/s41592-026-03178-8





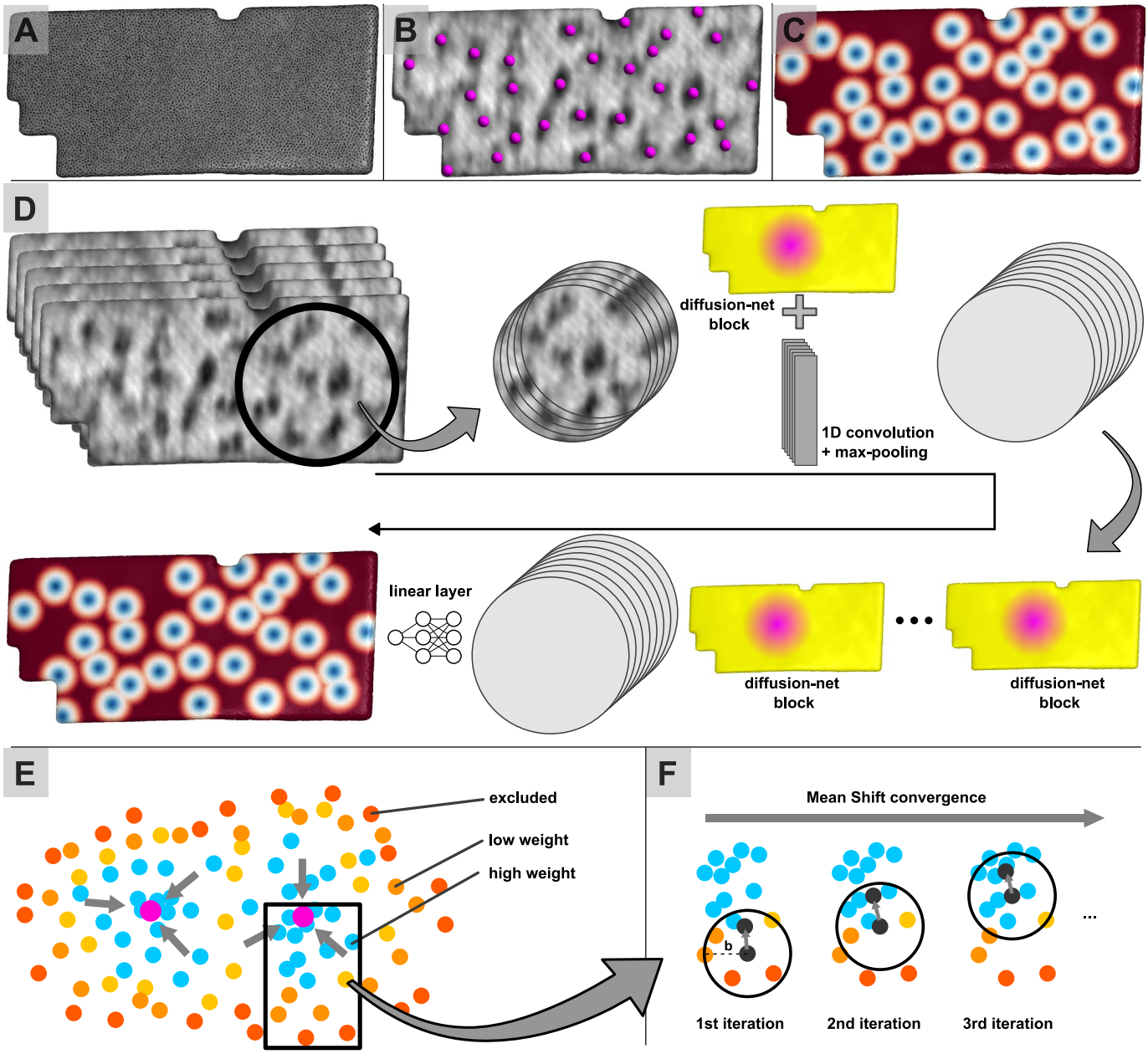
Extended Data Fig. 5 | Improvement of MemBrain-seg segmentations. **A-C:** Selected 3D patches with false segmentations (mainly false-positive surface layer), before and after manual correction. The lower panels illustrate the corresponding 2D tomogram slices, overlayed with the predicted and manual annotations **D:** Initial segmentation of the *Thermoanaerobacter kivui* tomogram predicted by MemBrain-seg's model MemBrain_seg_v10_alpha. **E:** Segmentation of the same tomogram after fine-tuning using the new corrected 3D patches.

https://doi.org/10.1038/s41592-026-03178-8



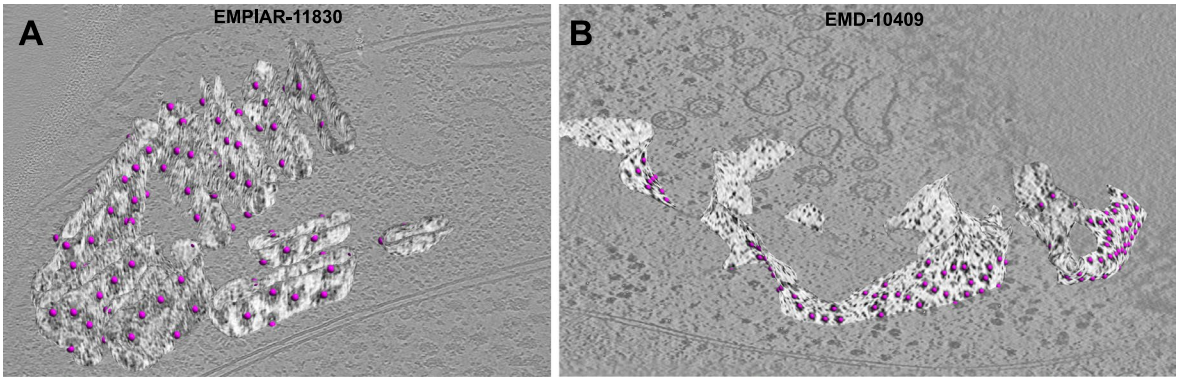
Extended Data Fig. 6 | MemBrain-seg's uncertainty quantification. Example tomogram slices (column 1), together with their binary segmentation predicted by MemBrain-seg (column 2). The third column depicts MemBrain-seg's predicted membrane uncertainty, and the fourth column overlays the uncertainty (blue) on top of the tomogram and segmentation.

https://doi.org/10.1038/s41592-026-03178-8



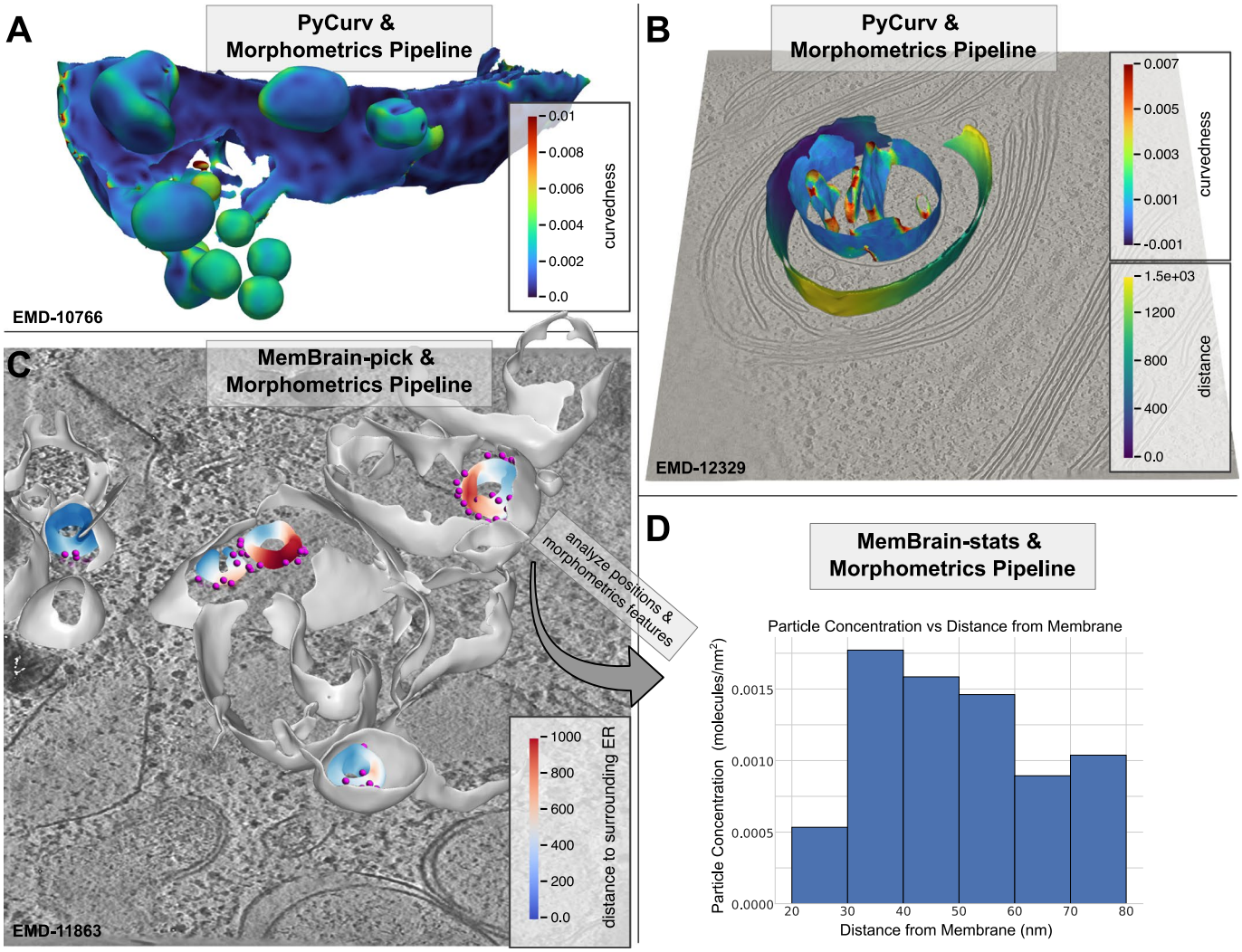
Extended Data Fig. 7 | MemBrain-pick processing steps. A: Triangular mesh representation of a segmented membrane. **B:** Tomographic densities projected onto the mesh with GT particle positions (magenta). **C:** Training target: Geodesic distance map to the nearest particle center. **D:** MemBrain-pick architecture: Tomographic densities are projected onto the mesh from multiple distances from the membrane surface, creating *N* feature channels per vertex. The mesh is partitioned into overlapping, evenly sized regions to ensure shape consistency and prevent overfitting to global structures. Each partition is first processed by a 1D convolution along the channel dimension, followed by four DiffusionNet blocks. Finally, an MLP predicts the geodesic distance map to the nearest particle center. **E:** Score-guided mean shift clustering refines particle localization. **F:** Example iterations of a clustering seed (black) converging toward a predicted particle center.

https://doi.org/10.1038/s41592-026-03178-8



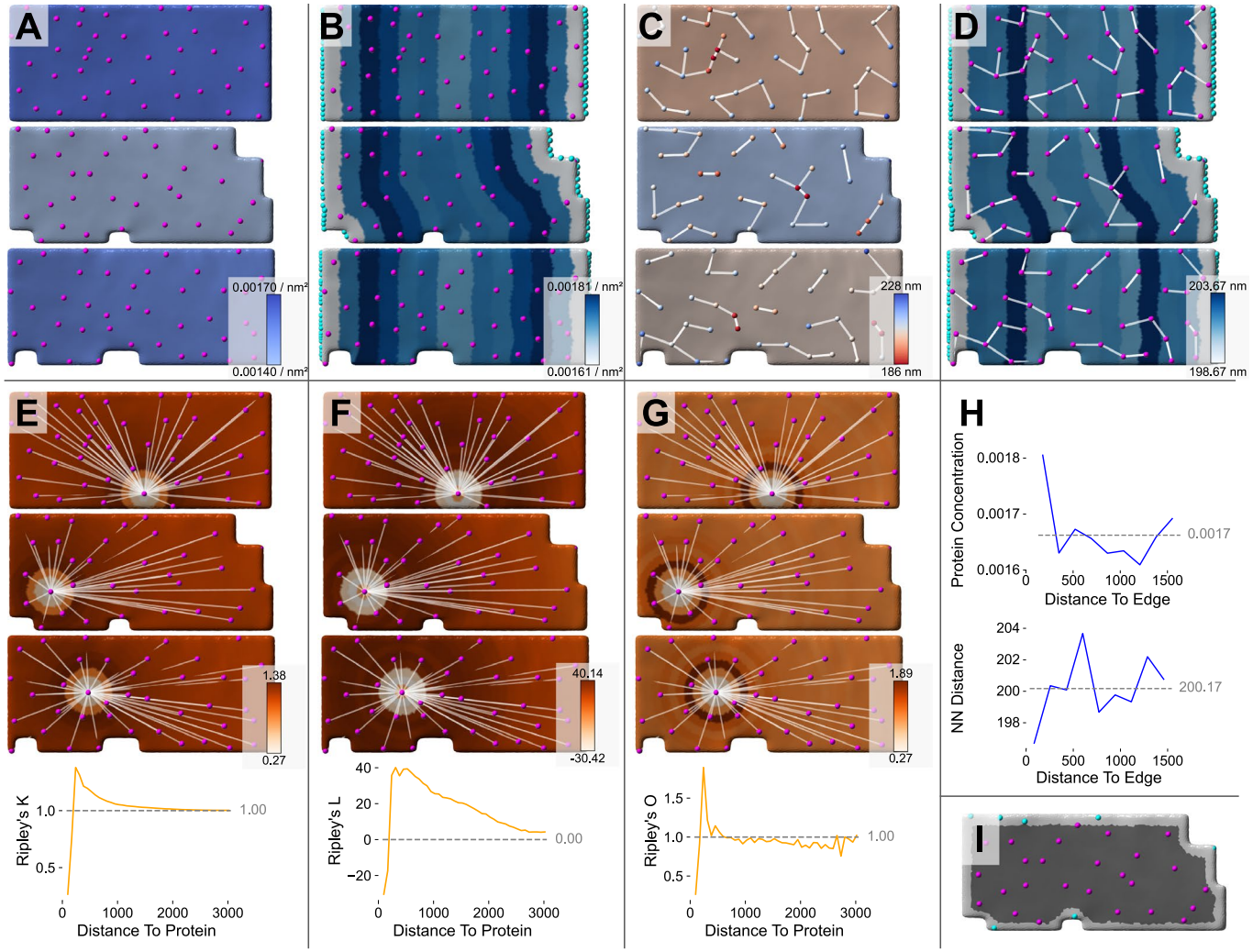
Extended Data Fig. 8 | MemBrain-pick additional applications. A: MemBrain-pick predictions of respirasome particles on mitochondrial crista membranes in EMPIAR-11830 refs. 19,77. **B:** MemBrain-pick predictions of ribosome positions on ER membranes in EMD-10409.

https://doi.org/10.1038/s41592-026-03178-8



Extended Data Fig. 9 | Compatibility with other morphometrics programs. **A:** Visualization of curvature computed using PyCurv, as implemented in the Surface Morphometrics Pipeline. **B:** Visualization of curvature (inner membrane), and inter-membrane distance (outer membrane), as implemented in the Surface Morphometrics Pipeline. **C:** Napari-based visualization of our MemBrain-pick prediction containers after merging outputs of the Surface Morphometrics Pipeline. MemBrain-pick's predictions (magenta) resemble SARS-CoV-2 spike proteins. Mesh colors represent the virion's distance to the surrounding ER membrane. **D:** Analysis of the merged containers in MemBrain-seg: Combining information from MemBrain-pick and the Surface Morphometrics Pipeline allows to compute particle concentrations with respect to different morphometric features.

https://doi.org/10.1038/s41592-026-03178-8



Extended Data Fig. 10 | MemBrain-stats visualizations. A: Particle (magenta) concentrations for different membranes. Membrane surfaces are colored according to their concentrations. **B**: Particle concentrations with respect to distance to the membrane borders (cyan). Distances are divided in equally-spaced bins, and each bin is colored according to its concentration. Bin concentrations are computed from all membranes in one tomogram. **C:** Geodesic nearest neighbor distances: Membranes are colored according to their average nearest neighbor distances. Single particles are colored according to their respective nearest neighbor distances. White lines represent nearest neighbor connections. **D:** Membrane is divided into bins and colored according to average bin nearest neighbor distances, similar to B. **E:** Ripley's K plot is visualized for a single starting particle. Membranes are colored according to their Ripley's K value for the vertices' geodesic distances to the starting particles. **F:** Same as E for Ripley's L. **G:** Same as E for Ripley's O. **H:** Particle concentrations / Geodesic nearest neighbor distances per binned distances to the membrane edge, corresponding to the membranes shown in B and D. **I:** Segmentation edge exclusion: Particle positions close to the segmentation edge (cyan) can be excluded from some analyses.

nature portfolio

Corresponding author(s): Lorenz Lamm, Benjamin D. Engel, Tingying Peng

Last updated by author(s): Apr 18, 2026

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed
☐	☒ The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐ A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐ The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐ A description of all covariates tested
☒	☐ A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒ A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐ For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐ For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐ For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐ Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	Manual membrane segmentations were performed in MITKWorkbench, particle posititions for evaluating MemBrain-pick were generated in MemBrain (v1) and manually corrected in membranorama. Other particle locations were generated with surforama. All tomograms used in this study are publicly available.
Data analysis	Subtomogram averaging was performed in STOPGAP. All other analyses in our study were performed with our custom code, available on our Github repository (https://github.com/CellArchLab/MemBrain-v2)

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

All tomograms used in this study are publicly available. The latest version of our MemBrain-seg training dataset is accessible via Zenodo: https://zenodo.org/

records/15089686
For MemBrain-seg demonstrations, we used tomograms from: CZII-DS-10007, CZII-DS-10224, CZII-DS-10442, CZII-DS-10443, CZII-DS-10444, EMD-3977, EMD-4869, EMD-12329, EMD-12727, EMD-12749, EMD-15407, EMD-16084, EMD-18306, EMD-18748, EMD-30364, EMD-35019, EMD-43050, EMD-44176, EMD-50605, EMPIAR-10988, EMPIAR-11058, EMPIAR-11370, EMPIAR-11830, EMPIAR-12612, and EMPIAR-13055.
For MemBrain-pick training and evaluation, we used tomograms from EMPIAR-12612, EMD-10780-10783, EMPIAR-11830, and EMD-10409. To showcase the entire MemBrain v2 workflow, we utilized data from EMPIAR-11830 and EMD-31244.
The GT annotations used for performing the efficiency analysis in MemBrain-pick are available via Zenodo: https://zenodo.org/records/15090084. The GT annotations for performing the finetuning on T. kivui tomograms is available at via https://zenodo.org/records/17737066.
The subtomogram average of a Chlamydomonas nuclear envelope-bound ribosome has been deposited under EMD-54837.

Research involving human participants, their data, or biological material

Policy information about studies with human participants or human data. See also policy information about sex, gender (identity/presentation), and sexual orientation and race, ethnicity and racism.

Reporting on sex and gender	*Use the terms sex (biological attribute) and gender (shaped by social and cultural circumstances) carefully in order to avoid confusing both terms. Indicate if findings apply to only one sex or gender; describe whether sex and gender were considered in study design; whether sex and/or gender was determined based on self-reporting or assigned and methods used. Provide in the source data disaggregated sex and gender data, where this information has been collected, and if consent has been obtained for sharing of individual-level data; provide overall numbers in this Reporting Summary. Please state if this information has not been collected. Report sex- and gender-based analyses where performed, justify reasons for lack of sex- and gender-based analysis.*
Reporting on race, ethnicity, or other socially relevant groupings	*Please specify the socially constructed or socially relevant categorization variable(s) used in your manuscript and explain why they were used. Please note that such variables should not be used as proxies for other socially constructed/relevant variables (for example, race or ethnicity should not be used as a proxy for socioeconomic status). Provide clear definitions of the relevant terms used, how they were provided (by the participants/respondents, the researchers, or third parties), and the method(s) used to classify people into the different categories (e.g. self-report, census or administrative data, social media data, etc.) Please provide details about how you controlled for confounding variables in your analyses.*
Population characteristics	*Describe the covariate-relevant population characteristics of the human research participants (e.g. age, genotypic information, past and current diagnosis and treatment categories). If you filled out the behavioural & social sciences study design questions and have nothing to add here, write "See above."*
Recruitment	*Describe how participants were recruited. Outline any potential self-selection bias or other biases that may be present and how these are likely to impact results.*
Ethics oversight	*Identify the organization(s) that approved the study protocol.*

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	not applicable in this study
Data exclusions	not applicable in this study
Replication	not applicable in this study
Randomization	not applicable in this study
Blinding	not applicable in this study

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Clinical data
☒	☐ Dual use research of concern
☒	☐ Plants

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Plants

Seed stocks

Report on the source of all seed stocks or other plant material used. If applicable, state the seed stock centre and catalogue number. If plant specimens were collected from the field, describe the collection location, date and sampling procedures.

Novel plant genotypes

Describe the methods by which all novel plant genotypes were produced. This includes those generated by transgenic approaches, gene editing, chemical/radiation-based mutagenesis and hybridization. For transgenic lines, describe the transformation method, the number of independent lines analyzed and the generation upon which experiments were performed. For gene-edited lines, describe the editor used, the endogenous sequence targeted for editing, the targeting guide RNA sequence (if applicable) and how the editor was applied.

Authentication

Describe any authentication procedures for each seed stock used or novel genotype generated. Describe any experiments used to assess the effect of a mutation and, where applicable, how potential secondary effects (e.g. second site T-DNA insertions, mosiacism, off-target gene editing) were examined.