

A. Further Related Works

In this section, we provide further discussion of the 2D segmentation techniques used in CellPose and CellStitch, as well as some alternative 3D segmentation approaches and their limitations when applying to our focused task.

2D Segmentation Methods. We introduce the following key 2D segmentation methods that facilitate the development of 3D cell segmentation methods:

U-Net [24] is a fully convolutional neural network specifically developed for biomedical image segmentation. It employs an encoder-decoder structure with skip connections that preserve spatial information, making it ideal for cell and tissue segmentation. Hover-Net [14] combines semantic segmentation with instance-level clustering via predicted horizontal and vertical distance maps (hover maps). Hover-Net is particularly effective for segmenting densely packed cell nuclei in pathology slides. Micro-Net [23] is a lightweight CNN architecture specifically tailored for segmenting cells in resource-limited environments. It leverages depth-wise separable convolutions and inverted residuals to achieve efficient segmentation with fewer parameters. DCAN [7] simultaneously predicts pixel-wise object masks and object contours using a dual-branch CNN architecture. Effective for precisely delineating individual cells by explicitly modeling boundaries. Omni-Seg [12] uses an ensemble of CNN models trained on multiple image types to generalize across a wide range of biological imaging modalities without retraining. It applies domain adaptation and multi-task learning techniques. NuSeT [40] is a CNN model tailored specifically for nuclear segmentation in fluorescence microscopy images, combining U-Net structure with a tailored loss function to enhance segmentation accuracy in noisy conditions.

3D Segmentation Methods. We provide an overview of existing 3D segmentation techniques, along with their limitations in the context of the current state-of-the-art methods.

3DCellSeg [34] is a two-stage deep-learning pipeline designed for dense 3D cell segmentation in membrane-stained images. The pipeline is robust with only one main hyperparameter, avoiding extensive manual tuning and generalizing across different datasets and cell shapes. But still, as a supervised 3D-based method, 3DCellSeg still requires annotated training data for new cell types or imaging conditions. Its performance can degrade if applied to modalities it wasn't trained on, necessitating retraining. Additionally, like many 3D-based methods, processing very large volumetric images can be computationally intensive comparing to 2D-based methods.

CellSeg3D [1] introduces a self-supervised 3D cell segmentation approach that eliminates the need for manual annotations. It employs a WNet3D architecture—comprising two 3D U-Nets connected sequentially—to perform segmentation by optimizing a soft normalized cuts loss directly on raw image volumes. However, as a self-supervised method, it relies on inherent image features, such as brightness differences, to delineate cells. This reliance may limit its ability to generalize effectively to end-user datasets, which often exhibit lower quality. While the performance is well on colorful fluorescent images, it may not be suitable for application in our generalized setting.

StarDist3D [37] is an instance segmentation method that represents cells (typically for nuclei) as star-convex shapes. A CNN predicts, for each pixel/voxel, the distances to the object boundary along a fixed set of radial rays, as well as an object probability score. StarDist assumes objects are approximately star-convex, so it may struggle with cells that have extremely irregular or concave shapes not well represented by a single star-shaped model. Another practical limitation is the need for training data for each new application. Due to the lack of large 3D training datasets, currently has no specific pretrained 3D StarDist models are publicly available. Still, StarDist3D achieves good performance on its specific cell types.

Go-Nuclear [33] is a recently introduced toolkit for 3D nucleus segmentation in tissue and organ datasets. Rather than a single algorithm, it encompasses a pipeline for generating training data and iteratively training models using multiple algorithms. The large curated dataset is used to train and fine-tune popular segmentation frameworks (CellPose, PlantSeg, StarDist) for nuclei in diverse organs. Yet, Go-Nuclear models are specialized for nuclear segmentation and rely on having a clearly labeled nucleus channel.

Due to the specificity of certain cell types and the lack of generalizability, as well as the absence of documentation on public open benchmarks, our work chooses to focus on CellPose, CellStitch, and PlantSeg, which have documented performance and are applicable to a wide range of cell types.

Geometry-aware Segmentation Methods. We further introduce several geometry- and topology-aware segmentation methods. Hu et al. [16, 17] propose loss functions and training schemes that encourage a segmentation network to preserve topological structures during end-to-end training from raw images. Lux et al. [19] define a topology-preserving loss framework based on a component graph for the joint ground-truth/prediction topology, again used during network training. Stucki et al. [32] introduce Betti matching, a spatially accurate loss for segmentation, by computing induced matchings between the persistence barcodes of prediction and ground truth. Shit et al. [28] propose centerline Dice, a similarity measure

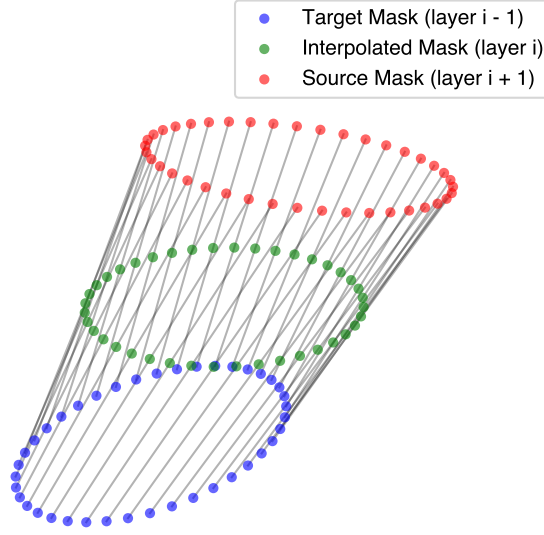


Figure 7. Example of interpolation between the source mask at layer $i + 1$ and the target mask at layer $i - 1$.

based on the intersection of segmentation masks with their skeletons, designed for tubular network-like structures, which is beyond the anisotropic 3D cellular segmentation studied in our work.

In other words, existing geometry-aware methods primarily address how to *train* a segmentation model, rather than how to perform a specific post-processing step on the output of such a model. These approaches are therefore complementary, not competing, to ours. All of them focus on loss functions for training a segmentation network from scratch or for fine-tuning it; their output is the segmentation itself. Moreover, these works mainly preserve **global topology** of one or a few foreground classes, whereas cell oversegmentation is an **instance-level** phenomenon. Frequently, Betti numbers and global connectivity are unchanged whether a single cell is split into two labels or kept as one, so these losses are blind to the specific oversegmentation errors that occur in a post-hoc manner. Adapting the ideas from the above methods to our setting would require substantial re-design to make them suitable as independent post-processing procedures for cellular segmentation. Therefore, we choose the most widely-used, state-of-the-art baselines and validate whether our method can further refine the segmentation quality over these methods. We hope that our framework will inspire follow-up work to adapt geometry-aware losses from prior segmentation methods into post-hoc processing methods.

B. Technical Details for Implementation

B.1. Cross-layer Interpolation to Recover 2D Mis-segmented Cell Masks

As our pre-trained classifier finds the oversegmented candidates, we then need to recover the missing mask between the two oversegmented cells. To achieve this, we adopt the interpolation method proposed in CellStitch [18], which builds upon the Wasserstein interpolation framework introduced by Solomon et al. [29].

We begin by treating each matched pair of cells in consecutive slices as source and target boundaries (i.e., the contours of each cell mask). Each boundary pixel is assigned a uniform mass, and an optimal transport (OT) plan is computed to map source pixels to target pixels. For any intermediate layer, we interpolate each matched pixel pair based on the transport weights, resulting in a geometry-aware interpolation of the boundary. The interpolated boundaries are then filled to reconstruct the full cell mask at that layer. An example is illustrated in Figure 7, where a mask at the intermediate layer i is generated using the cell contours from layers $i + 1$ and $i - 1$.

B.2. Computational Efficiency

We would like to further highlight our strong applicability in terms of computation efficiency. Since the number of cells, the average size of the cells, and the density of the cells varied from datasets to datasets, we believe providing a theoretical analysis towards the computational complexity of the algorithm is impractical. Therefore, we provide direct empirical results from real applications:

- Processing a large $300 \times 2008 \times 2008$ ($Z \times X \times Y$) animal stack takes approximately 1 hour in total on 2 NVIDIA A40 GPUs with 48GB storage.
- Running on a medium $2000 \times 224 \times 224$ ($Z \times X \times Y$) plant dataset takes around 7 minutes to identify oversegmentation candidates and around 15 minutes to stitch and recover all oversegmented cells—using CPUs only.

This speed is primarily due to:

- The use of sliced Wasserstein distance for EMD computation, which is well-suited to the uniform distribution of large cell masks, speeding up the calculation through random projections.
- The lightweight structure of our pre-trained MLP model.

MLP Hyperparameters and Structure. Our MLP is indeed lightweight and computationally efficient, which has 3 hidden layers with sizes 128, 64, and 32, uses ReLU activations, 0.3 dropout, a final sigmoid output.

B.3. Dealing with Multi-Oversegmentations

Although the occurrence of multiple gaps within a single cell is rare, we want to highlight that our framework is capable of handling such cases.

For example, consider a long cell body that is broken into multiple parts—say A, B, and C from top to bottom. Since our algorithm processes gaps layer by layer from top to bottom, it will first determine whether to stitch A and B. If A and B are stitched into a new cell body D, the algorithm will then assess whether to stitch D with C. This iterative process allows the framework to handle multiple fragments in a structured and consistent manner.

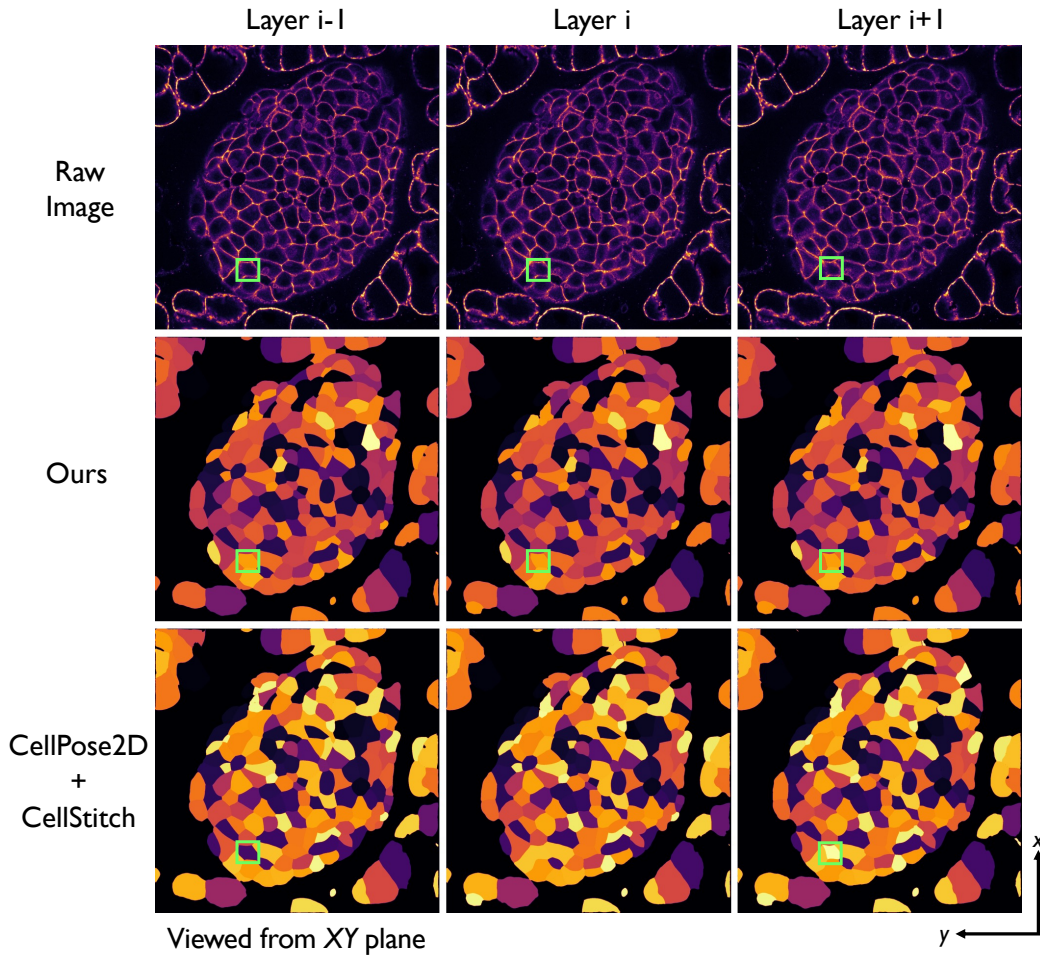


Figure 8. Successful correction for the undersegmented 2D mask from the animal dataset, viewed from the XY plane. The positions of the mis-segmented masks are highlighted with green boxes.

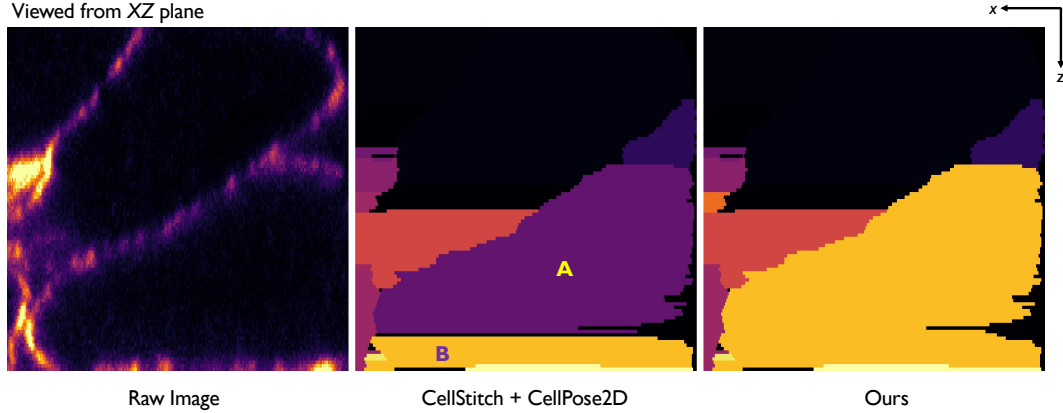


Figure 9. Example of a successful correction in animal cell datasets, viewed from the XZ plane. The left panel displays the raw microscopic image, the middle panel shows the 3D segmentation produced by CellStitch, and the right panel presents our results, where Cell A (purple) and Cell B (yellow) from the middle panel were stitched.

C. Visualization Examples with Further Analysis

In this section, we present several examples of correction results produced by our framework across various datasets and viewing planes to better illustrate its effectiveness. We begin by showcasing successful correction cases, followed by a closer examination of the unsure and incorrect cases. These examples also help us highlight the limitations of deterministic metrics such as mAP and Jaccard index, which may fail to fully capture the correctness of segmentation in complex scenarios.

C.1. More Successful Examples

Figure 8 illustrates an example of correcting a mis-segmentation that is not caused by an empty mask. In the CellStitch row, the purple cell at layer $i - 1$ and the bright yellow cell at layer $i + 1$ remain disconnected due to an undersegmentation error produced by CellPose2D at the same position in layer i . Our method successfully identifies this oversegmentation from a 3D perspective and corrects the 2D undersegmentation. The raw image clearly indicates that the three cell masks belong to the same 3D cell body. Moreover, in line 6 of Algorithm 1, we mean that there is no “complete” cell existing between the gap, with both its highest and lowest layers located between our candidates. However, we allow noisy cell masks to exist between the candidates, enabling us to also address 2D undersegmentation errors.

Figure 9 provides an example of correction viewed from the XZ plane, illustrating how two oversegmented cells (Cell A and Cell B) are stitched along the Z direction. Specifically, although the raw image shows that the cell’s 3D topological shape follows a strictly circular pattern (a gradual monotonic increase followed by a decrease), 2D segmentation inaccuracies produced by CellPose2D introduce **oscillations** that deviate its shape from the strictly monotonic standard. However, our relaxation approach using regression \mathcal{R}^2 alleviates this issue, as small oscillations still yield a high \mathcal{R}^2 value, distinguishing it from natural gap cases.

Figure 10 shows an example of accurate correction in the plant cell dataset by stitching the highlighted cell masks from layer $i - 1$ and layer $i + 1$. This correction is similar to the type shown in Figure 8, where we also addressed 2D undersegmentation. The true 2D segmentation in layer i is recovered through cross-layer interpolation:

C.2. Examples for Unsure Case

Figure 11 presents an example of an unsure case caused by hallucination masks produced by CellPose2D. In the first row, the green box in the ground truth highlights an area where no cells are present. However, in the same position in the CellStitch result, two hallucinated masks (purple and yellow) are generated at layer $i - 1$ and layer $i + 1$. These noisy masks lead our framework to stitch them together. While the correction appears accurate, there is no evidence from the ground truth segmentation to support our judgment. Therefore, we classify these cases as unsure, **pessimistically reporting only the absolutely correct cases**.

Figure 12 shows another uncertain case caused by noisy masks from CellPose2D. The green box in the ground truth marks an area without cells, but the CellStitch result includes two noisy masks (purple and orange) at layers $i - 1$ and $i + 1$, which are stitched together in our result. While the correction seems accurate, the ground truth provides no evidence to confirm this, so again, we conservatively report only the fully verified cases.

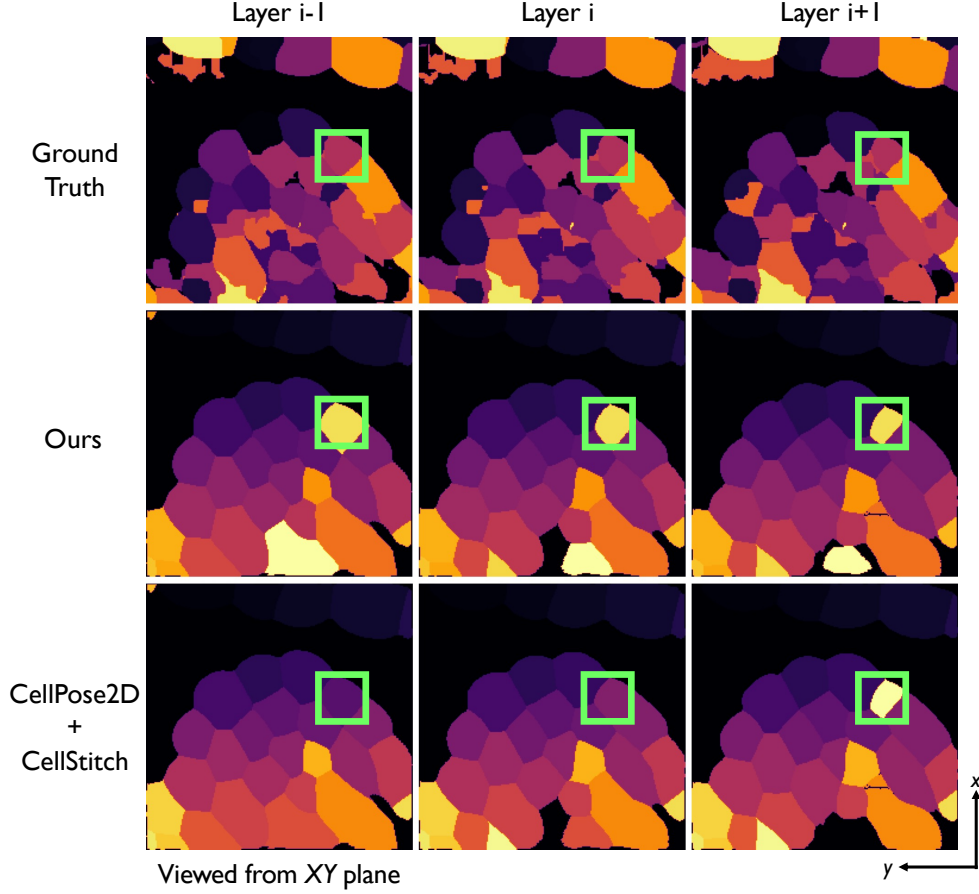


Figure 10. Accurate correction viewed from XY plane from the plant cell dataset. Position of the mis-segmented masks are highlighted.

C.3. Analysis on Incorrect Case

We also present example of incorrect cases to better understand the underlying issues in these situations and to provide clearer explanations for the causes of such incorrect results. As shown in Figure 14, in the row of CellStitch results, the dark blue mask at layer $i - 1$ and the yellow mask at layer $i + 1$ are stitched together in our results. Without the ground truth labels, this stitching appears correct, as both cells are similar in shape and located in the same position, with a missing mask in the intermediate layer. However, the ground truth row reveals that the dark blue mask is part of the large purple cell in layer $i - 1$, while the yellow mask belongs to the large orange cell in layer $i + 1$. Thus, the dark blue and yellow masks actually belong to two different cells, and their stitching is incorrect, making this an example of an incorrect case.

It is evident that both the dark blue and yellow masks only partially represent their corresponding ground truth cell masks. This highlights why we state that CellPose2D often “shrinks the cell masks smaller” than they are supposed to be. We illustrate this effect in Figure 13. When mask information is lost, both the geometric EMD measurement and the topological shape index are affected. For the EMD measurement, recall that EMD quantifies the effort required to transform one distribution into another. Losing geometric information alters the transformation, making the EMD between mask A and mask B differ from its original value. In the case shown in Figure 13, the EMD between the mis-segmented masks is smaller compared to the ground truth masks, suggesting a falsely higher similarity. For the topological shape index, which is calculated via changes in overlapping areas, as shown in Figure 14, the overlapping area between adjacent layers (e.g., layers $i + 1$ and $i - 1$) aligns more closely with the overlapping areas of preceding and succeeding layers (e.g., layers $i - 2$ and $i + 2$), as the masks shrink. These partial and altered representations lead our method to make incorrect judgments.

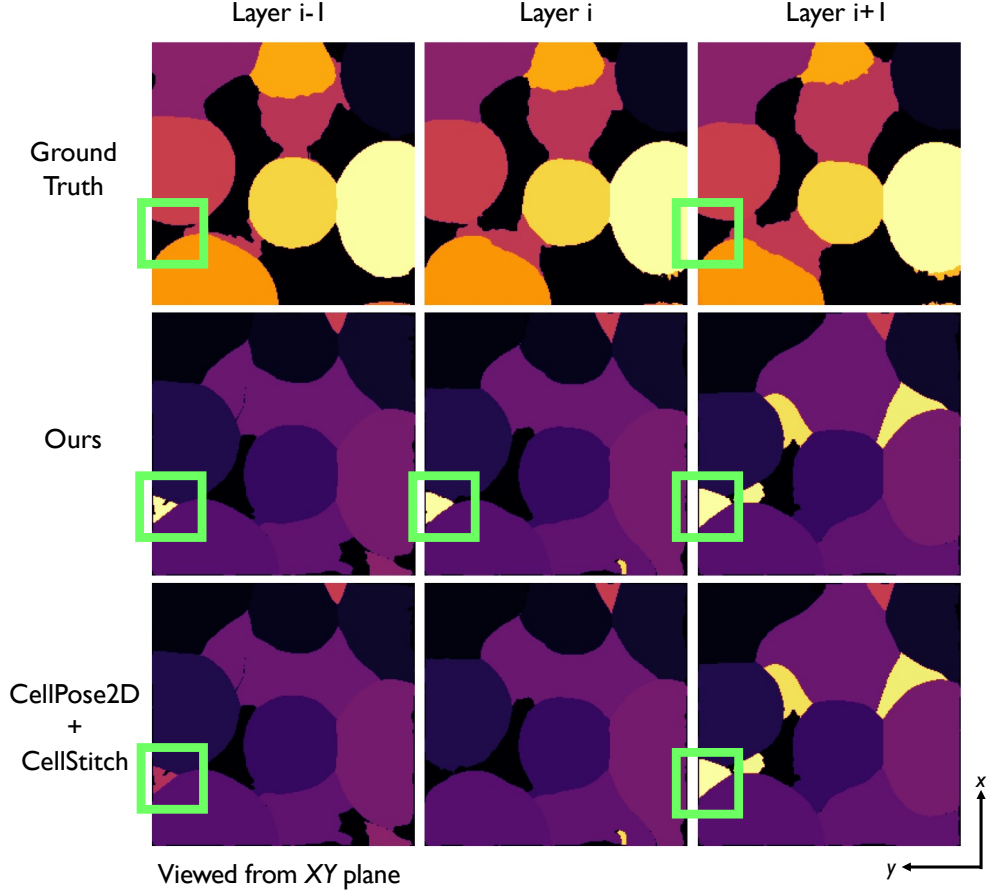


Figure 11. Example of the unsure case from plant dataset.

C.4. Comparison between PlantSeg and CellStitch on Pancreas-B Dataset

In Figure 15, we show a representative YZ-plane slice from the final 3D segmentation results. Most of the hallucinated noisy masks generated by CellPose2D are propagated into the final CellStitch output. Although PlantSeg also exhibits oversegmentation artifacts, its overall segmentation quality is better suited for end-users’ downstream analysis.

D. Dataset Information

Dataset	# Stacks	Type	Labeled	Anisotropy ($z:y:x$)
Anther	100	Plant / Public	✓	4:1:1
Filament	100	Plant / Public	✓	4:1:1
Leaf	100	Plant / Public	✓	4:1:1
Pedicle	100	Plant / Public	✓	4:1:1
Sepal	100	Plant / Public	✓	4:1:1
Valve	100	Plant / Public	✓	4:1:1
Pancreas-A	11	Animal / Private	×	4:1:1
Pancreas-B	1	Animal / Private	×	4:1:1

Table 5. Overview of datasets.

Table 5 shows the information for both plant and animal dataset. All the plant-type datasets are publicly available at

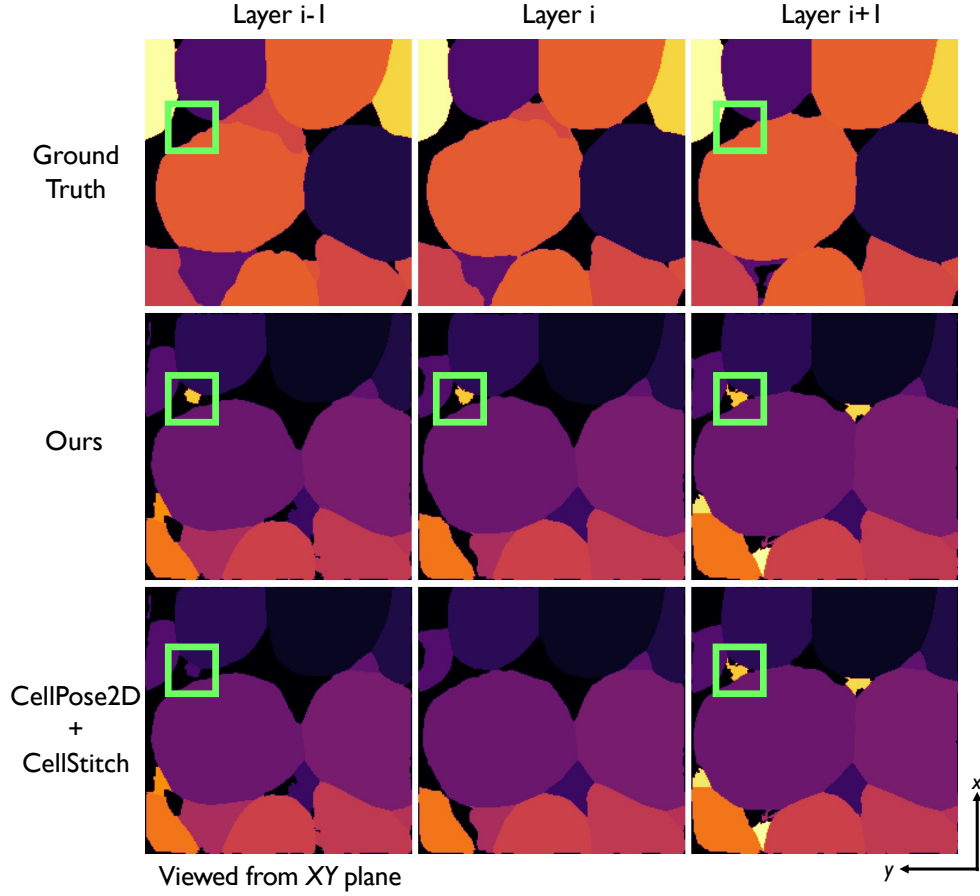


Figure 12. Extra example of unsure case led by the noisy masks.

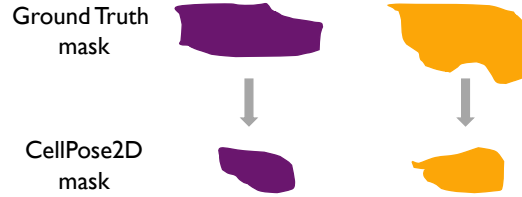


Figure 13. Analysis towards the misleading shrunk masks shown in Figure 14, where the cell masks are viewed from the XY plane.

Bassel [2]. For plant-type data, the voxel resolution is unknown. For animal-type data, the voxel resolution ($Z \times X \times Y$) is $0.4 \times 0.1 \times 0.1 \mu\text{m}$.

Pre-training Setup. For CellPose, we use the pre-trained `cyto2` model to generate 2D pre-segmented results. We also tune the diameter case-by-case, from 60 to 100, based on the cells in each datasets. For PlantSeg, we use the pre-trained `generic_confocal_3D_unet` model to generate 3D segmentation results.

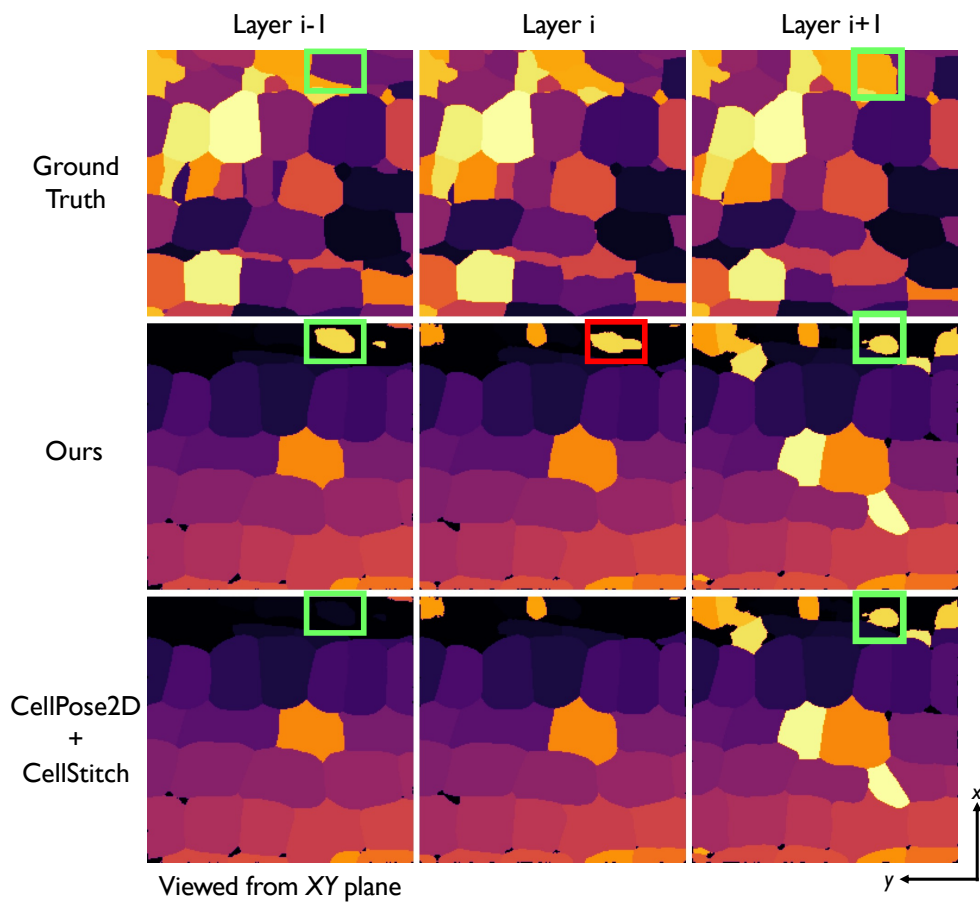
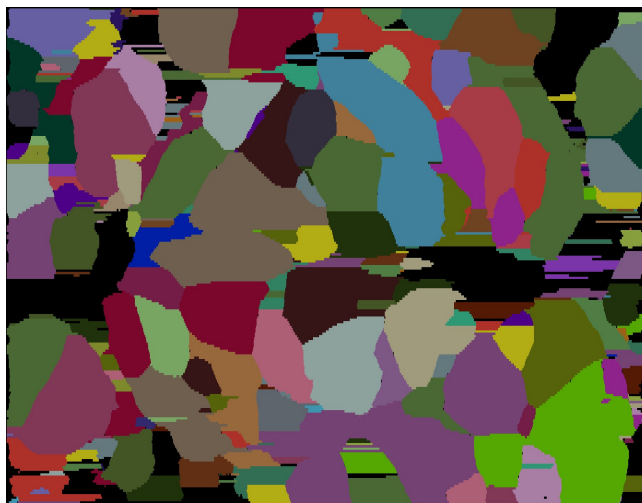


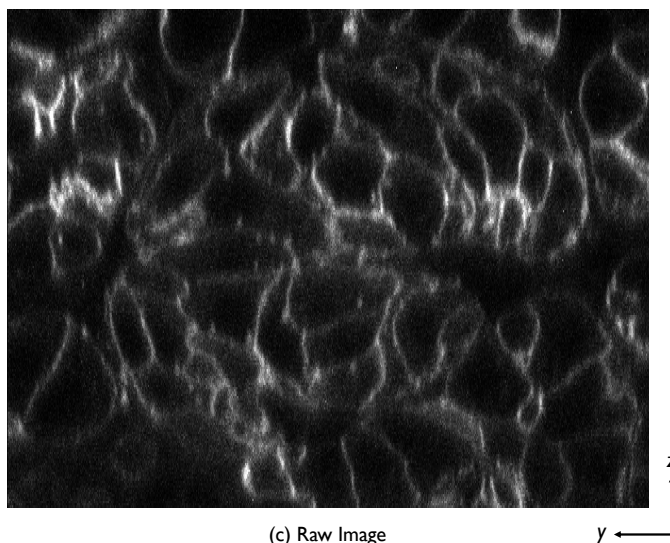
Figure 14. An example of incorrect stitching, where two masks from different cells are erroneously stitched together. The mask in red box is highlighted as the incorrect recovered mask. The incorrect stitching is partially influenced by misleading information from the 2D segmentation results.



(a) CellPose2D + CellStitch results



(b) PlantSeg results



(c) Raw Image

Figure 15. Example from Pancreas-B dataset. (a) Final 3D segmentation results produced by CellStitch; (b) Final 3D segmentation results produced by PlantSeg; (c) Raw fluorescence image for cell membranes. A specific layer of 2D segmentation is selected and viewed from the YZ plane. Note that the images have been adjusted for z -anisotropy.