Assessment of deep learning algorithms for 3D instance segmentation of confocal image datasets

Anuradha Kar[1]*, Manuel Petit[1], Yassin Refahi[2], Guillaume Cerutti[1], Christophe Godin[1]*, Jan Traas[1]*

1 Laboratoire RDP, Université de Lyon 1, ENS-Lyon INRAE , INRIA, CNRS, UCBL, 69364 Lyon, France

2 Université de Reims Champagne Ardenne, INRAE, FARE, UMR A 614, 51097 Reims, France

*Authors for correspondence:

Anuradha.Kar@ens-lyon.fr, Christophe.Godin@inria.fr, Jan.Traas@ens-lyon.fr

Abstract

Segmenting three dimensional microscopy images is essential for understanding phenomena like morphogenesis, cell division, cellular growth and genetic expression patterns. Recently, deep learning (DL) pipelines have been developed which claim to provide high accuracy segmentation of cellular images and are increasingly considered as the state-of-the-art for image segmentation problems. However, it remains difficult to define their relative performance as the concurrent diversity and lack of uniform evaluation strategies makes it difficult to know how their results compare. In this paper, we first made an inventory of the available DL methods for 3D segmentation. We next implemented and quantitatively compared a number of representative DL pipelines, alongside a highly efficient non-DL method named MARS. The DL methods were trained on a common dataset of 3D cellular confocal microscopy images. Their segmentation accuracies were also tested in the presence of different image artifacts. A new method for segmentation quality evaluation was adopted which isolates segmentation errors due to under/over segmentation. This is complemented with new visualization strategies that make interactive exploration of segmentation quality possible. Our analysis shows that the DL pipelines have very different levels of accuracy. Two of them show high performance, and offer clear advantages in terms of adaptability to new data.

Introduction

The use of 3 dimensional, quantitative (3D) microscopy has become essential for understanding morphogenesis, at cellular resolution, including cell division and growth as well as the regulation of gene expression (Thomas & John, 2017). In this context, image segmentation to identify individual cells in large datasets is a critical step. Segementation methods broadly belong to two types, namely ‘semantic segmentation’ in which each pixel within an image is associated with one of the predefined categories of objects present in the image. The other type, which is of interest in this paper, is ‘instance segmentation’ (Hafiz & Bhat, 2020). This type of method goes one step further by associating each pixel with an independent object within the image. Segmenting cells from microscopy images falls within this second type of problem. It involves locating the cell contours and cell interiors such that each cell within the image may be identified as an independent entity (Vicar et al., 2019). High accuracy cell instance segmentation is

essential to capture significant biological and morphological information such as cell volumes, shapes, growth rates and lineages (Lei et al., 2020).

A number of computational approaches have been developed for instance segmentation (e.g. (Thomas & John, 2017; Vu et al., 2019), (Cremers, Rousson, & Deriche, 2007; Gharipour & Liew, 2016) (Bailoni et al., 2019) (Kallasi, Rizzini, Oleari, & Aleotti, 2015; Pal & Pal, 1993)) such as for example the commonly used, watershed, graph partitioning and gradient based methods. In watershed approaches seed regions are first detected using criteria like local intensity minima or user provided markers . Starting from the seed locations, these techniques group neighboring pixels by imposing similarity measures until all the individual regions are identified. In graph partitioning, the image is treated as a graph, with the image pixels as its vertices. Subsequently pixels with similar characteristics are clustered into regions. Gradient based methods use edge or region descriptors to drive a predefined contour shape (usually rectangles or ellipses) and progressively fit them to accurate object boundaries, based on local intensity gradients (Ding, 2018; Zheng, Dong, Cao, Sun, & Li, 2014) .

Common challenges faced by these segmentation methods arise in low contrast images containing fuzzy cell boundaries. This might be due to the presence of nearby tissue structures as well as anisotropy of the microscope that perturb signal quality, poor intensity in deeper cell layers as well as blur and random intensity gradients arising from varied acquisition protocols (Van Valen et al., 2016) (W. Wang et al., 2019). Some errors can also be due to the fact that cell wall membrane markers are not homogenous at tissue and organ level: in some regions the cell membrane is very well marked resulting in an intense signal while in the other regions this may not be the case. These different problems lead to segmentation errors such as incorrect cell boundary estimation, single cell regions mistakenly split into multiple regions (over-segmentation), or multiple cell instances fusing to produce a condensed region (under-segmentation).

In recent years, a number of computational approaches based on large neural networks (commonly known as deep learning or DL) (LeCun, Bengio, & Hinton, 2015) have been developed for image segmentation (Zeng, Wu, & Ji, 2017) (Caicedo et al., 2019; Moen et al., 2019). These methods seem extremely promising. Indeed, the key advantages of DL based segmentation algorithms include automatic identification of image features, high segmentation accuracy, requirement of minimum human intervention, no need for manual parameter tuning and very fast inferential capabilities. These DL algorithms are made of computational units (‘neurons’), which are organised into multiple interconnected layers. For training a network, one needs to provide input training data (for example images) and corresponding target output (ground truth). Each network layer transforms the input data from the previous level into a more abstract feature map representation for the next level. For example the neurons from the first level would take the intensities of a subset of pixels as an input, apply a transformation function and transmit the resultant of thresholded values to the next layer. The second layer can then proceed on the information coming from several ‘neurons’ in the first layer. The final output of the network is compared with the ground truth using a loss (or cost) function. Learning in a neural network involves repeating this process and automated tuning of the network parameters multiple times. By passing the full set of training data through the DL network a number of times (also termed ‘epochs’) the network estimates the optimal mapping function between the input and the target or ground truth data. The number of epochs can be in the order of hundreds to thousands depending on the type of data and the network. The training will run until the training error is minimized. Then, in a second

'recognition' phase, the neural network with these learnt parameters can be used to identify patterns in previously unseen data.

Instance segmentation using deep learning is a challenging task especially for 3D data due to large computational time and memory requirements for extracting individual object instances from 3D data (Ren & Zemel, 2017) (Tokuoka et al., 2018) Therefore, the current trend in deep learning based segmentation methods is to proceed in two steps. Firstly deep networks are used to provide high quality semantic segmentation outputs. This involves the extraction of several classes of objects within an image such as cell boundaries, cell interiors and background. These DL outputs are then used with traditional segmentation methods to achieve the final high accuracy and automatic instance segmentation even in images with noise and poor signal quality. t (Eschweiler et al., 2019). A generic workflow of such a deep learning based instance segmentation process is shown in Figure 1.

In contemporary deep learning literature, two types of architecture for segmentation are commonly used: the ones based on the UNet/residual Unet network (Falk et al., 2019) (Zhu et al., 2020) and the approaches using the region proposal networks or RCNNs (Region Based Convolutional Neural Networks) (He, Gkioxari, Dollar, & Girshick, 2017). We will only briefly present both the general properties of both types of networks. The U-Net (Falk et al., 2019) has a so-called symmetric deep learning architecture. One part (called encoder) extracts the image features and the other part (named decoder) combines the features and spatial information to obtain the semantic segmentation, for example cell boundaries, cell body and image background. In order to obtain the instance segmentation, this is followed by methods such as watershed or graph partitioning. Examples of UNet based 2D and 3D segmentation algorithms include e.g. (Al-Kofahi, Zaltsman, Graves, Marshall, & Rusu, 2018) (Wolny et al., 2020) (Eschweiler et al., 2019).

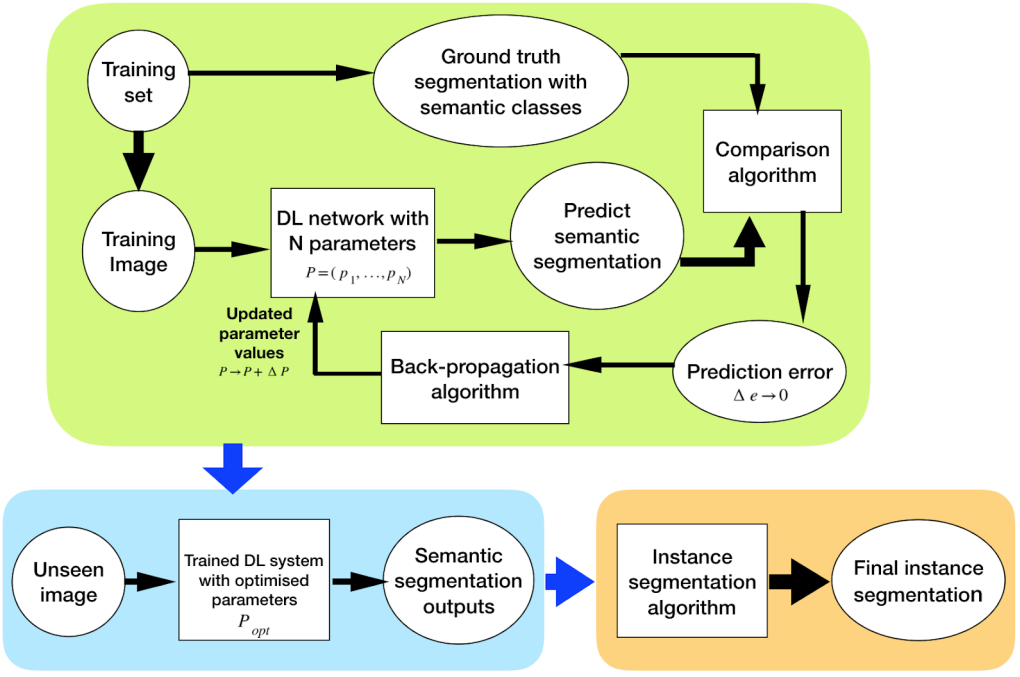

Figure 1 Generic workflow of a deep learning based image segmentation pipeline. The DL network is first trained to produce a semantic segmentation which corresponds as closely as possible to a given groundtruth. The trained network is then used to segment unseen images. The resulting semantic segmentation is then further processed to obtain the final instance segmentation.

Besides UNet, the other state-of-the-art deep learning architecture is a Region based convolutional neural network or RCNN such as Mask-RCNN or MRCNN (Zaki et al., 2020). MRCNN differs from UNet as the former includes modules for object detection and classification unlike the UNets. MRCNN has been used for high accuracy and automatic segmentation of microscopy images in several works(e.g. (Shu, Nian, Yu, & Li, 2020) (Linfeng Yang et al., 2020) (Johnson, 2018)).

There currently exists a large number of deep learning pipelines (we have identified and reviewed up to 35 works in the last 5 years in Appendix 4), where variants of both the above architectures are used to address specific challenges in segmentation such as sparse data sets, availability of partial ground truths, temporal information, etc (see Appendix 4 for a more extensive review). However the diversity of the currently available pipelines and inconsistent use of segmentation accuracy metrics makes it nearly impossible to characterise and evaluate their performance.

In this work, we compare representative segmentation pipelines which use one of the above two approaches. These pipelines have been identified from the literature and were selected based on the following criteria. (i) Firstly, as the focus of this work is on 3D confocal datasets the pipelines should be built for 3D instance segmentation of static images. The analysis of temporal information or specific architectures for cell or particle tracking are not included. (ii) Next, the pipeline implementations including pre- and postprocessing methods should be available in open-source repositories. (iii) To ensure that the pipeline is reproducible properly on other machines, the training dataset used originally by the authors should be available publicly. (iv) Lastly, the DL pipelines should be trainable with new datasets. Based on these criteria, we identified four pipelines ((Eschweiler et al., 2019), (Wolny et al., 2020) (Stringer, Wang, Michaelos, & Pachitariu, n.d.) (He et al., 2017)), which we further describe below.

The first pipeline is an adapted version of Plantseg (Wolny et al., 2020) which can be trained using 3D images composed of voxels. It uses a variant (see Materials and methods section) of 3D Unet called residual 3D-UNet (Zhu et al., 2020) for the prediction of cell boundaries in 3D, resulting in a semantic segmentation. These are then used in a post-processing step for estimating the final instance segmentation using graph partitioning. Examples of graph partitioning include GASP (Bailoni et al., 2019), Mutex (Wolf et al., 2018), Multicut (Kappes, Speth, Andres, Reinelt, & Schnörr, 2011).

The second deep learning pipeline (Eschweiler et al., 2019) comprises a 3D U-Net which can be trained using 3D confocal images (i.e. composed of voxels) for prediction of cell boundary, cell interior and image background regions (as 3D images). These semantic outputs of the 3D U-Net are then used to generate a seed image for watershed based post-processing. Seeds in watershed segmentation indicate locations within images from where growing of connected regions starts in the image watershed map. The seed images produced from the Unet outputs in this pipeline are therefore used to perform 3D watershed and obtain the final segmentation output.

The third pipeline is adapted from Cellpose (Stringer et al., n.d.). It uses a residual 2D-UNet architecture which should be trained using 2D images (composed of pixels). The 2D trained Unet predicts horizontal (X) and vertical (Y) vector gradients of pixel values, or flows, along with a pixel probability map (indicating whether pixels are inside or outside of the cell regions) for each 2D image. By following the vector fields the pixels corresponding to each cell region are clustered around the cell center. This is how 2D gradients in XY,

YZ and ZX planes are estimated. These 6 gradients are averaged together to find 3D vector gradients.These 3D gradients are used to estimate the cell regions in 3D.

The fourth deep learning pipeline is adapted by the authors of this paper from the well documented open Mask RCNN repository l (He et al., 2017) and the 3D segmentation concept using this model is inspired from (Linfeng Yang et al., 2020). For the Mask-RCNN based segmentation, a hybrid approach is adopted as shown in Figure 2. The pipeline uses a MRCNN algorithm which is trained using 2D image data to predict which pixels belong to cell areas and which do not in each Z slice of a 3D volume leading to a semantic segmentation. Then the Z slices containing the identified cell regions are stacked into a binary 3D seed image. The cell regions in this binary image are labelled using the connected component approach, where all voxels belonging to a cell are assigned a unique label . These labelled cell regions are used as seeds for watershed based processing to obtain the final 3D instance segmentation.

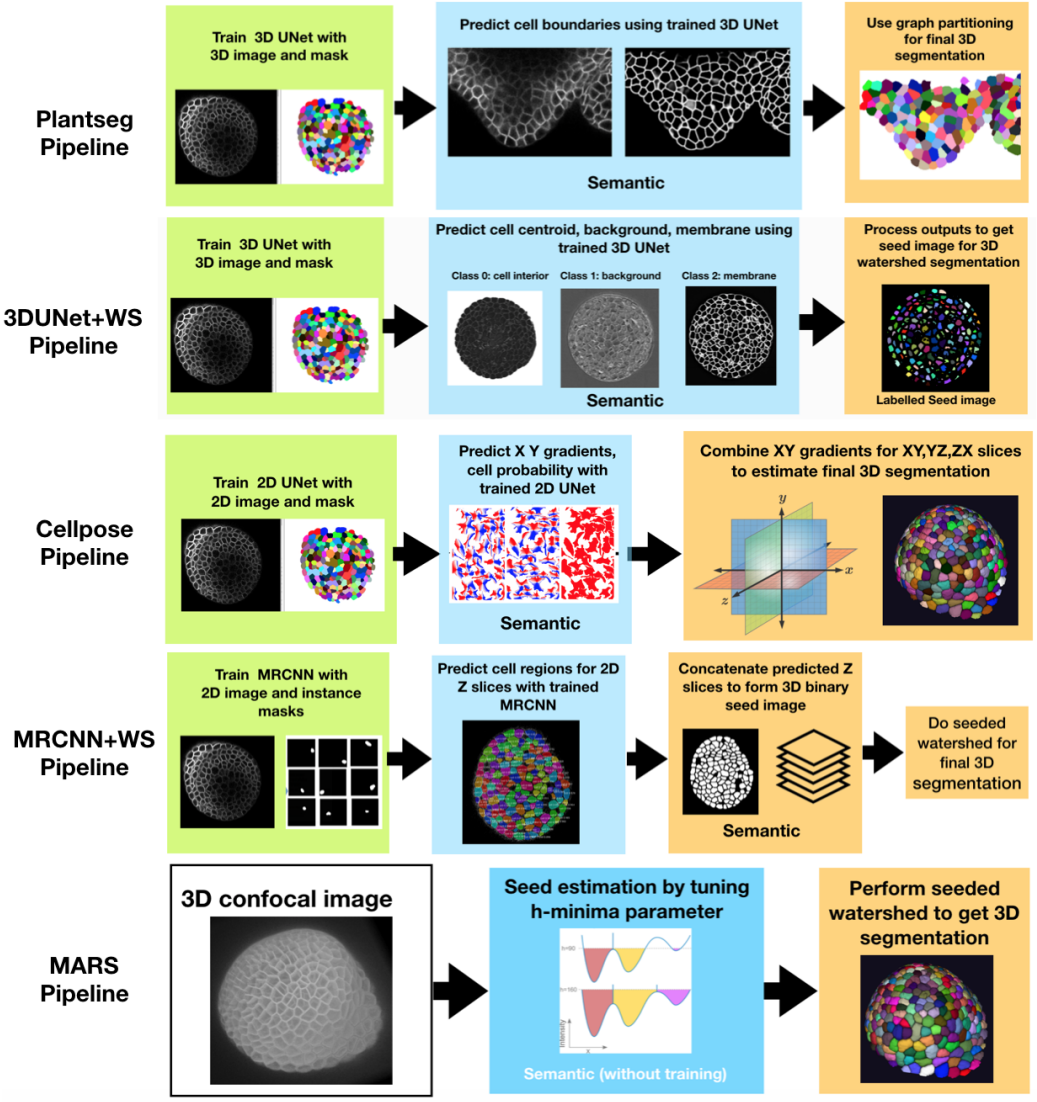

Figure 2 Displaying all the 3D segmentation pipelines together. The green colored boxes indicate the training process for the respective pipeline. The blue boxes indicate the predicted, semantic segmentations generated by the trained DL algorithms, the orange boxes indicate phases of post processing, leading to the final instance segmentation. The MARS pipeline doesn't include a training or post processing step, but parameter tuning is required.

A further aspect of investigation in this work is to observe how these deep learning pipelines compare to a classical non-deep learning pipeline in terms of segmentation accuracy. They were therefore compared with a watershed based segmentation pipeline named MARS (Fernandez et al., 2010), which uses automatic seed detection and watershed segmentation. In the MARS pipeline, the minima of local intensity of the image detected by a h-minima operator is used to initiate seeds in the image which are then used for 3D watershed segmentation of cells.

As in the original works, these five pipelines have been developed and tested on different datasets and use different evaluation metrics, it is difficult to directly compare their performance. The first motivation was, therefore, to test them on common 3D image datasets and estimate and compare their performance based on a common set of metrics. More in general, we aimed to develop an efficient strategy for quantitative and in depth comparison of any 3D segmentation pipeline that currently exists or is under development. Our results show clear differences in performance between the different pipelines and highlight the adaptability of the DL methods to unseen datasets.

Results

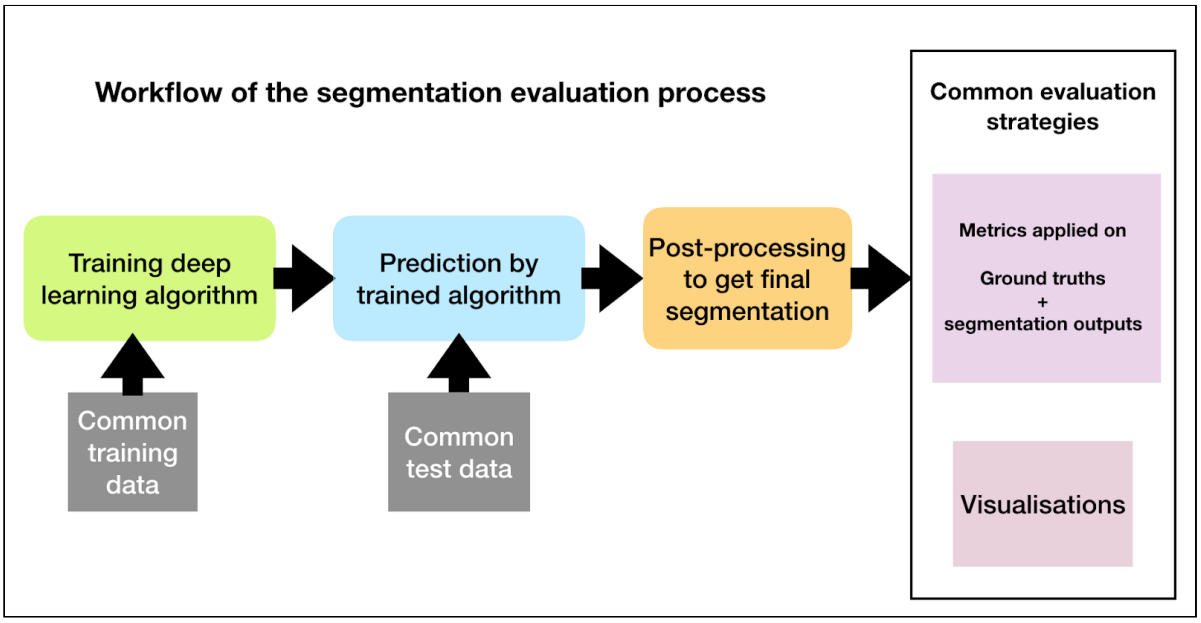

Figure 3 Schematic workflow of the segmentation evaluation process. The evaluation of segmentation pipelines begins with the training of the deep learning models on a common training dataset (confocal images and ground truth). The training and post processing steps for each pipeline are reproduced in the exact way as defined in the respective papers or their repositories. Then the five pipelines are tested on a common test set of images. The test dataset (Figure 4) contains both raw confocal images and their corresponding expert annotated ground truths, and therefore it is possible to assess the segmentation accuracy of the 5 pipelines by comparing segmentation output of each pipeline with the respective ground truth data. Finally the relative accuracy of each method is evaluated using multiple strategies.

Segmentation of test data using the 5 pipelines

The workflow for quantitative performance evaluation of the segmentation pipelines adapted in this paper is shown in the schematic diagram of Figure 3. All 4 deep learning pipelines were trained following the specifications of respective pipelines (more details in Materials and Methods section) as given in their

repositories. For training we used a common set composed of 124 3D original stacks of confocal images from Arabidopsis shoot apical meristems and their ground truth segmentations, which are publicly available, as described in (PNAS). This is one of the more extensive 3D confocal sets with ground truth publicly available. The trained networks were used to segment two test datasets of floral meristem images (Figure 4) described in (Refahi et al 2021, see Materials and methods section) and for which ground-truths were available as well. Sample results from these pipelines on one test stack (TS1-00h) are shown in Figure 4.

For MARS, a manual tuning of three parameter values is generally required to obtain optimal segmentation (h-minima, Gaussian smoothing sigma for image and that for seeds, see Materials and Methods section for details on MARS parameters to tune). This can involve many trials before optimal segmentation is obtained and needs expert supervision. Therefore, two different approaches were tested. First the optimal MARS parameters were found for one 3D image and these were then kept constant for the remaining images in the test set. Alternatively tuning was repeated for every image in the test set (results referred to as MARS*).

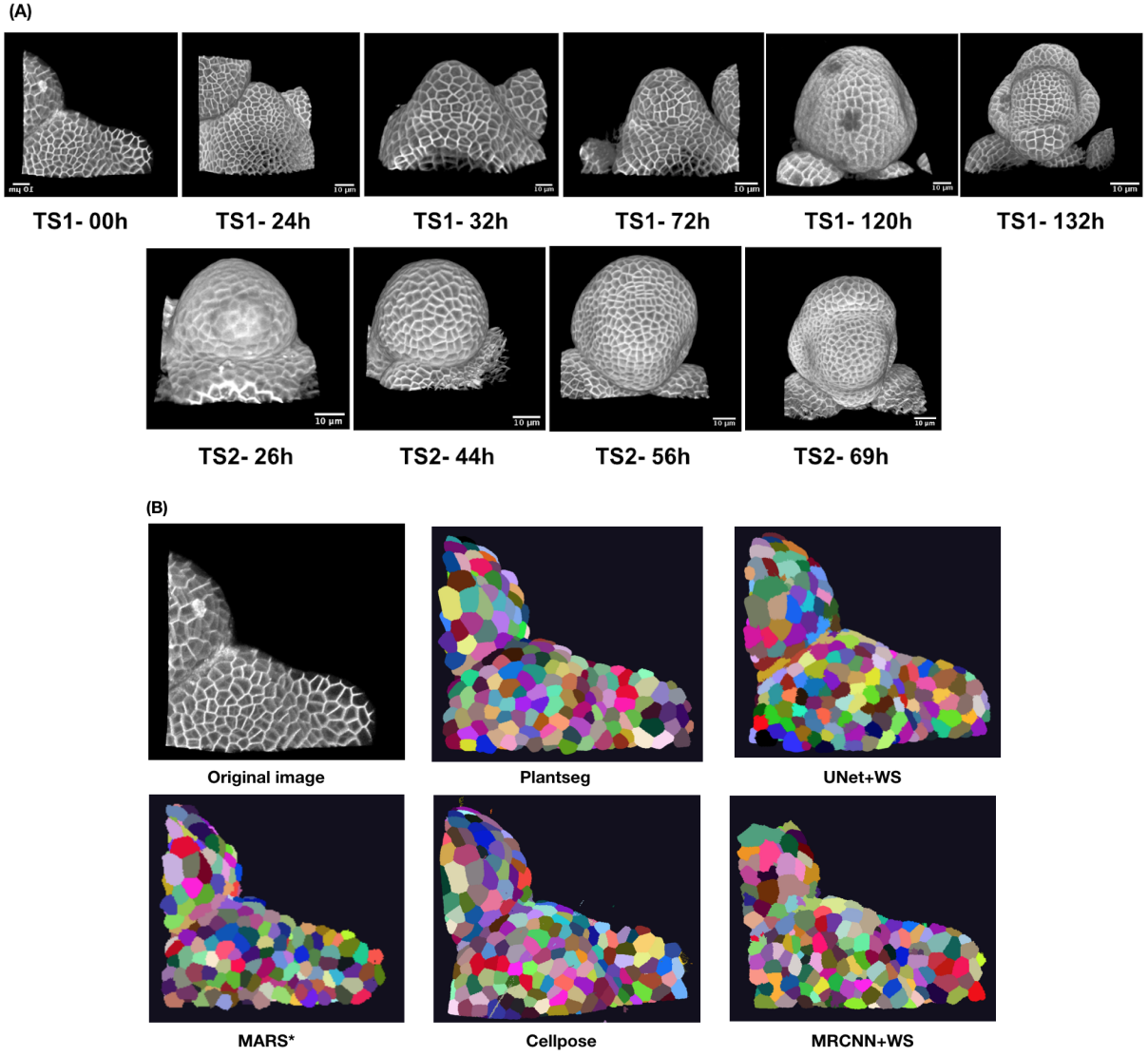

Figure 4 (A) The two test datasets containing a total of 10 confocal image stacks of two different Arabidopsis floral meristems (B) A sample test stack (TS1-00H) and its segmentation by 5 segmentation pipelines.

Comparing the segmentation pipelines

We next compared the quality of the segmentations produced by the 5 pipelines. For this purpose we adopted three different strategies (see also Materials and methods for details). First, we analyzed the quality of segmentation on overall stacks, including all cell layers. Next, the confocal stacks were split into different cellular layers (L1, L2, inner) and the segmentation quality of the 5 pipelines for each layer was studied. Finally, we evaluated the segmentation quality on images with commonly occuring (artificially generated) aberrations. In each strategy, several metrics were used for quantitative assessment.

Strategy 1: Evaluating segmentation quality for entire image stacks

To estimate the segmentation quality of the outputs from the 5 pipelines, we used their segmented results and the corresponding ground truths of the test datasets. A Volume averaged Jaccard Index (VJI) metric was used to estimate overlap between the predicted segmentations and ground-truths. The VJI used here measures the degree of overlap averaged over the cell volume. In the VJI metric the averaging over cell volume is done to avoid biases arising from the cell sizes on the standard JI. Also, two additional metrics that identify the rates of over and under segmentation were applied (details of these two metrics are in the Materials and Methods section). The rate of over-segmentation is the % of cells in the ground truth associated with multiple regions in the predicted segmentation. Conversely the rate of under-segmentation is the percentage of cases where several regions in the groundtruth are associated with a single cell in the predicted segmentation.

Algorithms	Vol. Avg. Jaccard index (VJI)	Rate of over-segmentation	Rate of under-segmentation
Plantseg	0.840	10.30%	2.43%
3D Unet+ WS	0.768	19.20%	2.40%
Cellpose	0.741	21.38%	7.12%
MaskRCNN+WS	0.615	49.10%	5.05%
MARS	0.610	20.18%	7.64%
MARS*	0.826	12.25%	2.86%

Table 1: Mean values (average over the two test datasets) of segmentation evaluation metrics.

The results, summarized in Table 1 and Figure 5A-D, reveal a number of differences between the pipelines. Firstly, comparing the results of MARS and MARS* the importance of tuning the MARS parameter values to obtain good segmentation accuracy is observed. The MARS results are obtained by using a constant set of values for the MARS parameters (which are h-minima= 2, Gaussian smoothing sigma for image =0.5 and that for seeds= 0.5). These parameters were found to be best for one image in the test set, but keeping these parameters fixed yields non-optimal accuracy for other images in the set as may be observed in Figure 5 A and B from the VJI values for MARS results. On the other hand, by tuning the h-minima and smoothing parameters for each image, MARS is seen to be capable of yielding high accuracies as seen from MARS*

results. However, as mentioned above, this parameter tuning process is much time and effort consuming and requires multiple trials with the 3 parameters for each image in the set. Note that these images were taken using the same experimental protocol.

Among all the pipelines, Plantseg performs best as measured using the VJI metric values, closely followed bv the MARS* and then UNet+Watershed. Cellpose and in particular MRCNN+Watershed perform less well.

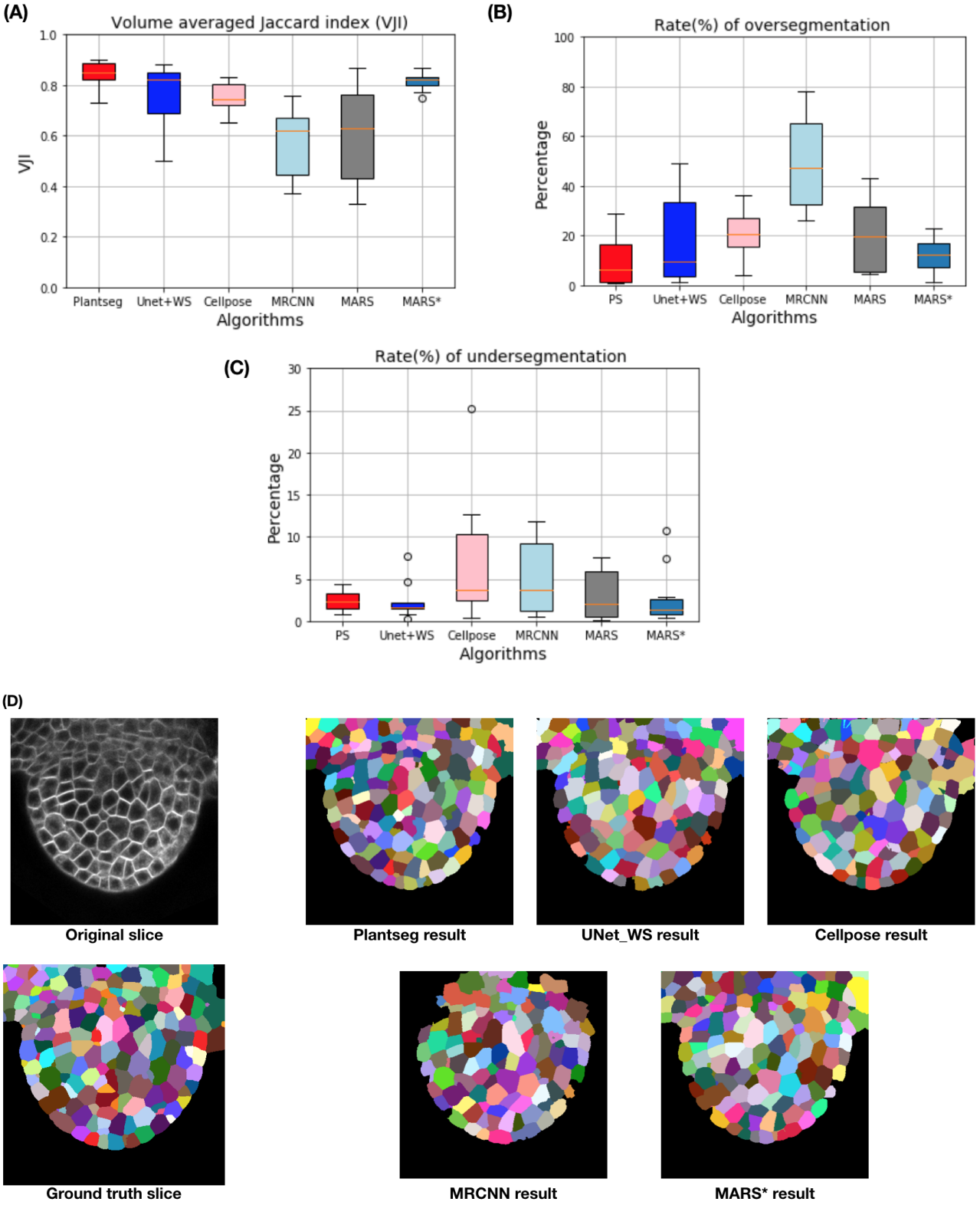

Figure 5 (A) Results of VJI metric from the five segmentation pipelines. (B) and (C) shows rates of over and under segmentation. The distributions shown here are estimated over the results from the two test datasets TS1 and TS2. (D) Example segmentation results by 5 pipelines on a test image slice.

With respect to rates of over-segmentation, Plantseg and the MARS* pipelines have lowest rates followed by Unet+Watershed. MARS results (i.e without parameter tuning) yields higher rates of both over and under-segmentation errors compared to MARS*. The two hybrid pipelines Cellpose and MRCNN+Watershed have much higher rates of over-segmentation. The rate of under-segmentation is much lower for all the pipelines than the rate of over-segmentation, but again the 3D Unet, Plantseg and MARS* perform better than the other two. Overall, we can conclude that the errors in the pipelines are mostly due to over-segmentation and therefore to improve the segmentation quality one needs to address this point first.

From the above results, it is observed that results from the MARS pipeline without individual tuning of the parameters are not optimal. Therefore for the rest of the paper, MARS* results (i.e optimized MARS) are used to compare with the deep learning pipelines

Strategy 2: Segmentation quality evaluation for different cell layers

3D confocal stacks often show different levels of intensity and contrast in different cell depths. In particular in the inner layers, the cell segmentation can be challenging. We therefore tested the performance of the pipelines for their capacity to segment the different cell layers.

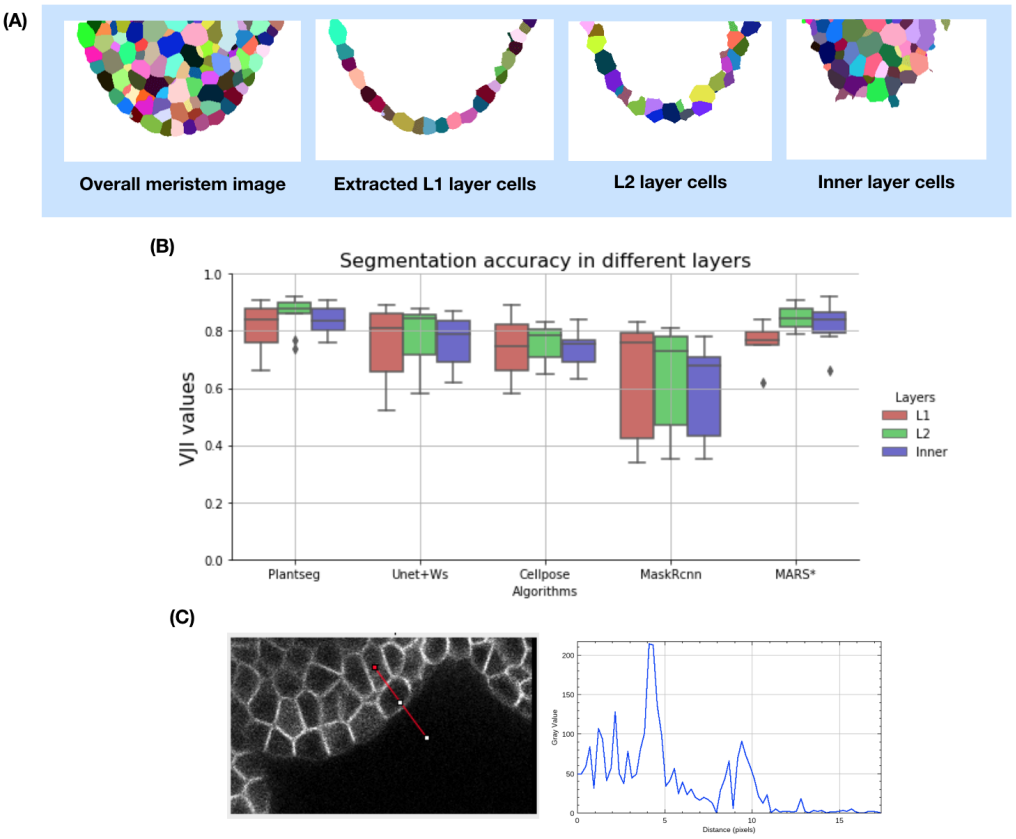

Figure 6 (A) Extracting L1, L2 and inner layers from an input segmented meristem image (B) Estimating segmentation accuracy (VJI) for different cell layers. All stacks from the test dataset are used for this evaluation. (C)Boundary Intensities profile plot for outer and inner layer cells

For identifying the layers from the segmented stacks, we used a neighborhood connectivity criterion for labeled images. Within a segmented image, the L1 cells are identified as the cells having the background as their neighbor. The L2 layer cells are defined as the cells with L1 cells as their neighbors and the inner layer

cells are obtained as the remaining cells after removing the L1, L2 cells and background from the image (Figure 6A). The variation of segmentation quality of the different pipelines was studied on the basis of the VJI values in the three cellular layers (Figure 6B). Plantseg produces the most accurate segmentations in all the layers out of the 5 pipelines and its accuracy doesn't degrade significantly in the outer or inner layers. The VJI index of the UNet+Watershed pipeline is only slightly lower and nearly the same for all the layers. MARS* performs only slightly better than UNet in inner layers, but has lower accuracy in the L1 layer. Cellpose and MRCNN+Watershed perform less well than the others, in all three layers and MRCNN accuracy drops in the innermost layer. As could be expected the inner cells are less well segmented, as the signals are weaker and the signal/noise ratio is lower. However, as seen in Figure 6, the segmentation of the L2 layer is slightly better than that of the L1. This is actually observed for all tested pipelines.This might be linked to the weak labelling of the outer membranes (Figure 6C) which is often observed.

Strategy 3: Evaluating pipelines on synthetically modified images

Confocal images are often affected by effects such as noise, shadows and motion blur which tend to perturb the image signal. Especially in the inner layers, due to loss of optical signal and scattering, there occur regions of very poor signal and distortions. In order to study the impact of these variations on the segmentation quality, the effects of noise, blur and intensity variations are simulated on the test set of confocal images. The five segmentation algorithms are then applied to the modified images and the VJI values and rates of under/over segmentations are estimated to observe and compare their robustness against these conditions.

Effect of image noise: In confocal images, noise can be contributed by the imaging apparatus such as the electronic detector or amplifier and this noise can be modeled using Gaussian statistics (Haider et al., 2016). An image to which two different Gaussian noise levels corresponding to noise variances of 0.04 and 0.08 are added is shown in Fig.7A. Higher the variance, more is the noise effect as seen in the PSNR values.

The five pipelines behaved very differently under the impact of image noise as shown in Figure 7B. In particular CellPose and MRCNN are very sensitive to Gaussian noise as their accuracy drops sharply when Gaussian noise variance is increased. At a noise variance of 0.08, Cellpose shows no detection. For MRCNN, higher noise leads to loss in identified cell regions, which results in large blob-like regions after watershed based post-processing leading to higher under-segmentation (Figure 7D). The difference with Plantseg and UNet + WS may be explained by differences in the instance segmentation components. As explained above, Plantseg uses graph partitioning. UNet +WS like the MRCNN pipeline uses 3D watershed although the seed identification criteria are different for the two pipelines.

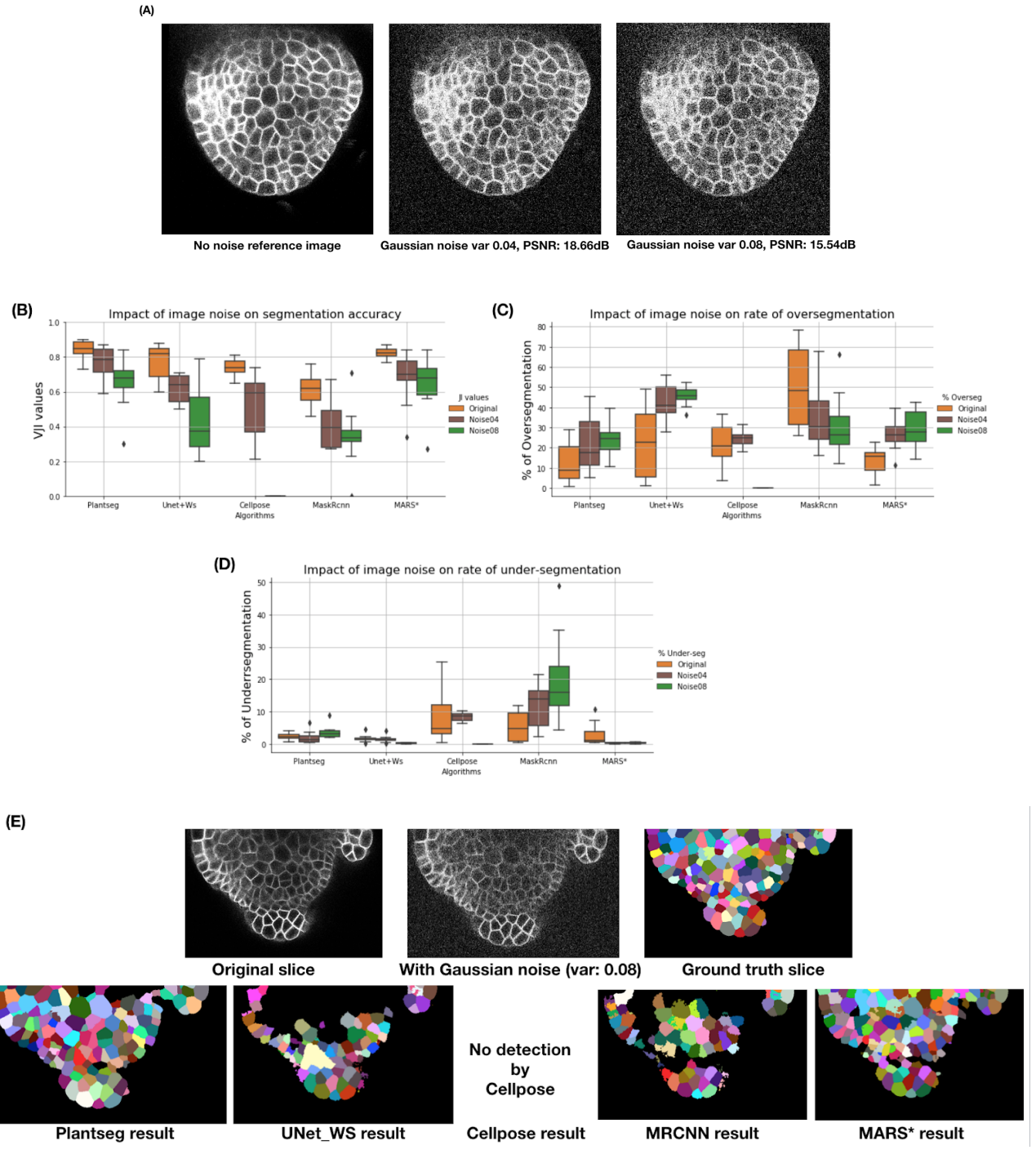

Figure 7 (A) A test image after applying Gaussian noise (var 0.04, 0.08) (B) Variation of segmentation accuracy (VJI) with 3 Gaussian noise variances (C) variation in rates of over-segmentation (D) variation in rates of under-segmentation. Note that for noise variance of 0.08, Cellpose is unable to identify cells. (E) Example results from the five pipelines under the impact of image noise (Gaussian noise variance 0.08)

Effect of image blur. Blurring commonly occurs in microscopy images due to artefacts like lens aberrations or due to optical diffraction in the imaging setup (Hadj, Blanc-Féraud, Aubert, & Engler, 2013) **.** It can also be caused by motion of the objects in the microscope To simulate blur, the test confocal image is convolved with a horizontal motion blur kernel (material and methods). A sample image before and after blurring is shown in Figure 8A.

The 10 test stacks were subjected to the blurring function and were segmented using the 5 pipelines. VJI values were then computed for each of the results and plotted (Figure 8B). It is seen that the Plantseg and

MARS pipelines are least affected by blurring, whereas this effect produces large variability in results in the other three (Figure 8B). Apart from Plantseg and MARS, the other pipelines suffer higher rates of over-segmentation (Fig8C) under the impacts of blurring. The rates of under-segmentations are below 5% .

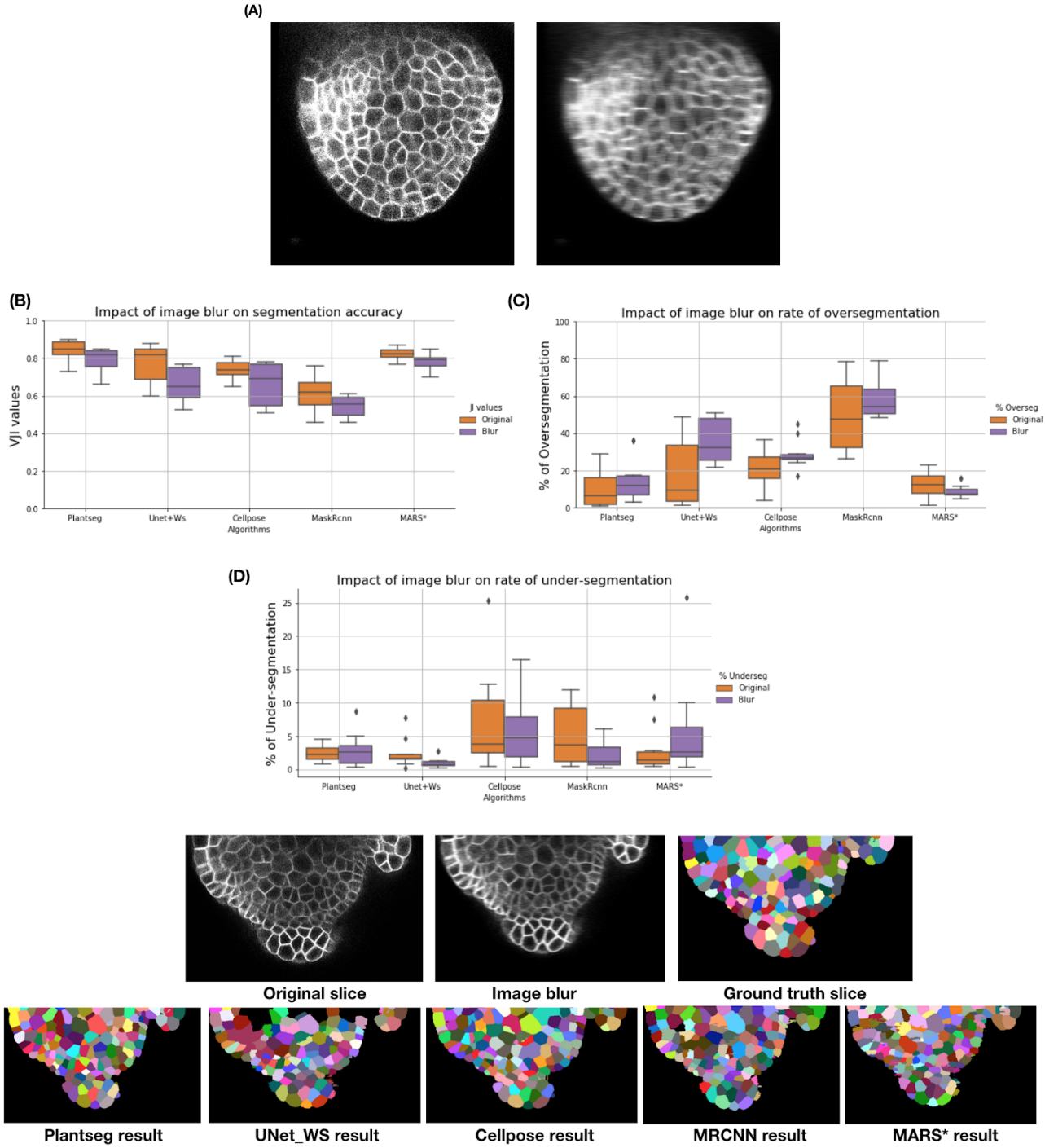

Figure 8 (A) Effect of blurring on an image (B) Comparing segmentation accuracies of pipelines under the effect of image blur (b) Comparing rates of over-segmentation (D) Under-segmentations due to image blur. (E) Results from the five pipelines under the impact of image blur

Image intensity variations. Partially bright regions in microscopy images may be caused by inhomogeneous illumination sources and shadow effects are mostly caused by presence of light absorbing objects or obstructions (Cheng & Kriete, 1990; Ricci et al., 2020) or due to a non-homogenous cell membrane marker. To emulate the effect of such intensity variations within an image, partial overexposure or underexposure regions (Figure 9A) were imposed on the test images, which were then segmented using the

5 pipelines. The VJI values and rates of over- and under-segmentations for all the results were computed (Figure 9). The box plots of VJI results for all stacks with partial overexposure effect are shown Figure 9A-D.

Overexposure had a strong negative impact on the VJI of Plantseg, UNet+Watershed and Cellpose. The MARS* and MaskRCNN pipelines were not affected appreciably. Likewise, over-segmentation increased significantly in the Plantseg, Unet+Watershed and Cellpose pipelines (Figure 9C) while the other two were barely affected. Under-segmentations remained at low levels i.e. below 10% for all pipelines (not shown).

Underexposure was strongly reflected in the VJI results from Plantseg. The accuracy of Cellpose, followed by Unet+Watershed and MARS* was also clearly affected (Figure 9D). MRCNN was less sensitive.
Partial underexposure induced a high degree of over-segmentation in MARS* and Unet+WS while errors in Plantseg and Cellpose were mostly induced due to under-segmentation as seen in Figure 9F. The MRCNN based pipeline, although having low overall accuracy, is found to be less sensitive to underexposure.

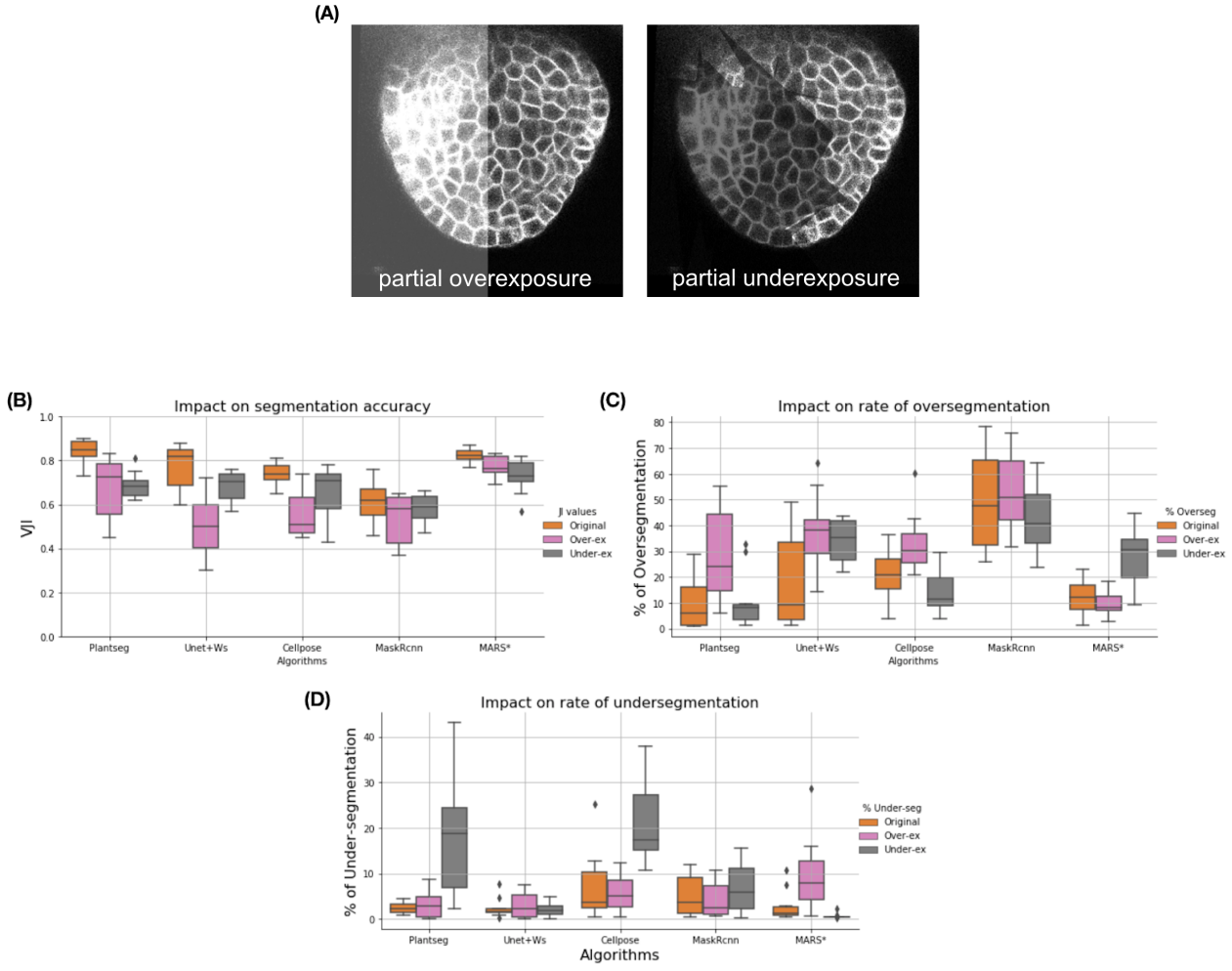

Figure 9 Impact of image exposure levels on segmentation quality of 5 pipelines. (A) examples of partial over and underexposure. In (B) the VJI values for over and under-exposure are plotted together with the original VJI values for unmodified stacks. Similarly in (C) and (D) the rates of over and under-segmentation are plotted for the impacts of over and underexposure alongside those for the unmodified stacks.

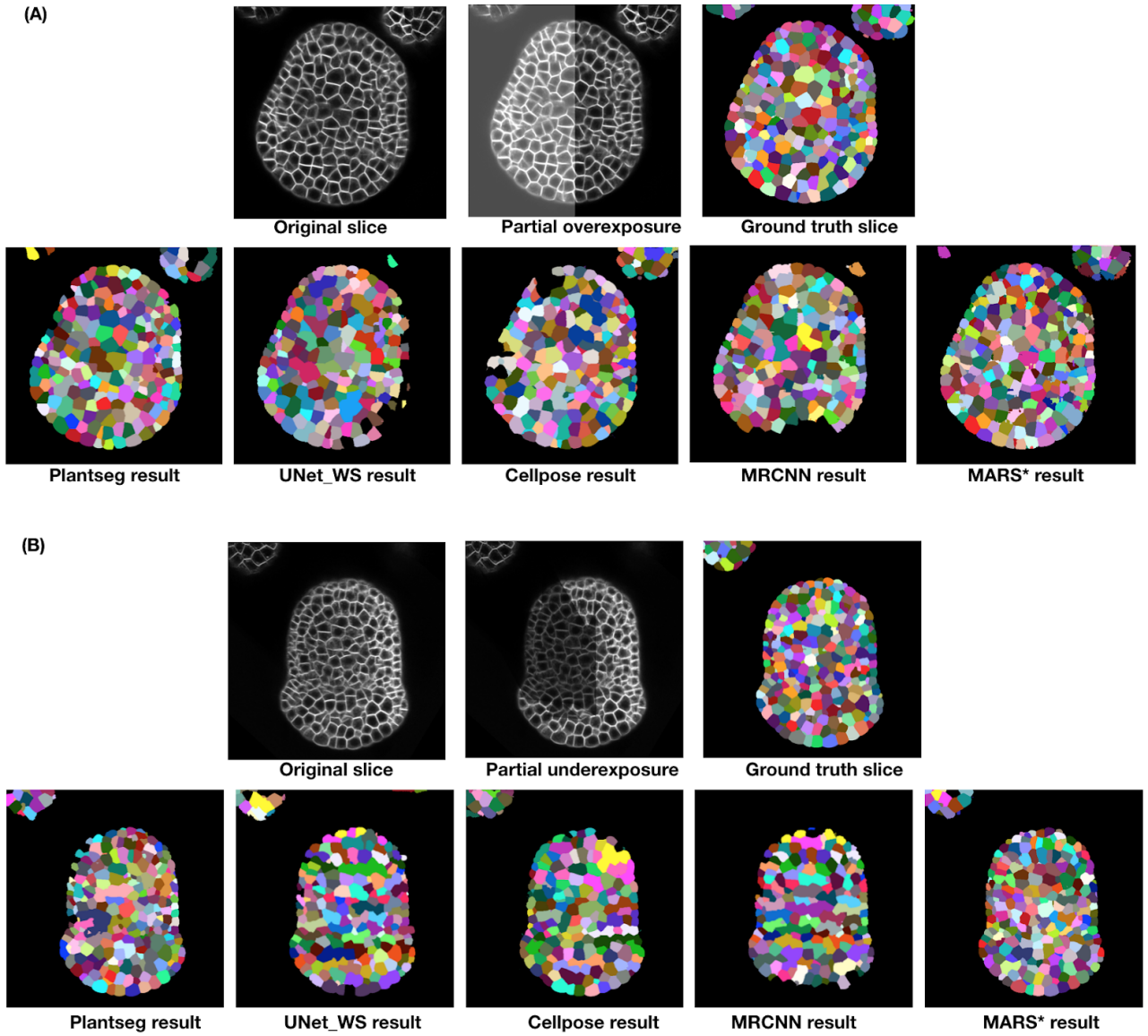

Figure 10 (A) Sample results from the five pipelines under the impact of image over-exposure (B) Results from the five pipelines under the impact of partial underexposure

In conclusion, overall for the Plantseg, Unet+Watershed and Cellpose (or the Unet based) pipelines the effect of image intensity variations appears to be much stronger than image noise or blur effects. MARS* and MaskRCNN on the other hand were relatively stable. Plantseg nevertheless retains a high accuracy under most of the conditions. MARS* out of all the pipelines is found to be the most stable under the effect of image artefacts, although it leads to increased over-segmentation in partially underexposed samples.

3D visualization of segmentation quality

At present, visualization and exploration of 3D image data is possible by software packages such as ImageJ, Paraview or MorphographX (Barbier de Reuille et al., 2015). However, only a few tools allow users to project any extrinsic property over an image and to interact with a 3D image dataset at cellular resolution. One such platform is Morphonet (Leggio et al., 2019) (see Appendix 2) which is an open source and web-based platform for interactive visualization of 3D morphodynamic datasets by converting image datasets (3D

stacks) into corresponding 3D meshes. So far, Morphonet has been used to project a variety of genetic and morphological information such as gene expression patterns, cell growth rates and anisotropy values on plant and animal tissue images.

In this work, we have developed a MophoNet based method for interactive 3D visualization of segmentation quality measures, by projecting VJI values on 3D meshes created from 3D confocal images (Figure 11). The 3D meshes were created by the marching cubes algorithm (Feltell & Bai, 2010) and uploaded on MorphoNet. The VJI values for the image were computed and uploaded as a quantitative property of each individual 3D cell of the image for each segmentation pipeline. The VJI values can then be viewed using colormaps on the Morphonet browser.

The steps of the Morphonet based visualization pipeline are provided as open resources in the SegCompare Gitlab repository (described in Appendix 1) so that users can upload their own segmentation results on our datasets from any segmentation method or pipeline and benchmark study their accuracy characteristics. The only requirement is that these segmentations should be 16bit labelled .tif files and must have their ground truth segmentations for evaluation. With segmented and ground truth .tif stacks, the steps for Morphonet based segmentation quality visualization includes (Figure 11): a) estimation of cell-by-cell VJI and converting this information to Morphonet compatible format (b) mesh calculation from the ground truth segmented image and converting it to Morphonet compatible .obj format and finally (c) logging in to Morphonet, creating a dataset and uploading the mesh and VJI information. All these steps are documented and implemented in an easy to use Python notebook as described in Appendix 1 and 3.

The benefits of the Morphonet based visualization is that a user can: a) check the VJI value for each cell by clicking on it b) impose a color mapping of the Jaccard index values so that users can at a glance observe the overall distribution of the VJI values on the cells of the 3D stack (Figure 12) as well as locate cell regions where poor Jaccard values are found or concentrated c) interact with segmented data in 3D, e.g rotate 360 degrees or zoom into cells or slice through the segmented data in XYZ directions and inspect segmentation quality in any inner cell layers d) requires no special software or hardware installation. Also data uploaded on Morphonet can be shared with multiple users who can directly access it on the Morphonet platform.

Examples of viewing segmentation quality on Morphonet is illustrated in Figure 12 where mesh belonging to a test stack is uploaded on Morphonet and the VJI values for all the segmentation pipelines are projected on it individually to observe the cell-by-cell variation in segmentation quality.

Sample segmentation accuracy data and 3D meshes in Morphonet compatible format are provided in our repository : https://figshare.com/projects/3D_segmentation_and_evaluation/101120

Sample videos demonstrating various aspects of the 3D visualization process on Morphonet may also be found under “Videos” in our repository described in Appendix 3.:

https://doi.org/10.6084/m9.figshare.14686872.v1

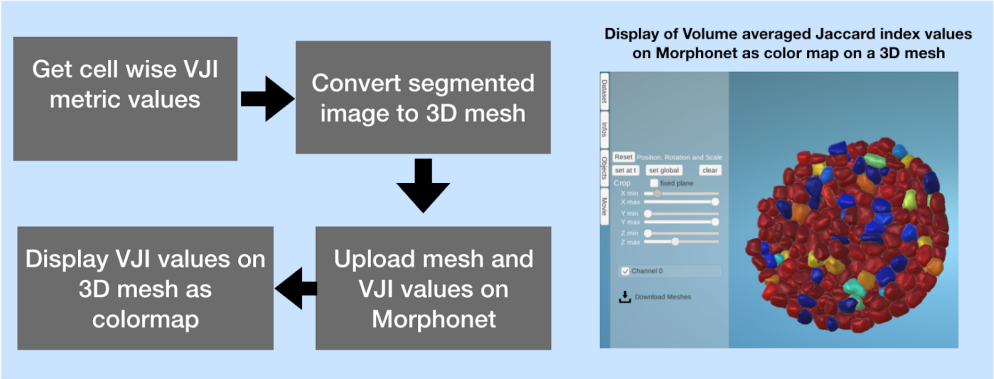

Figure 11 Process to view segmentation quality in 3D on Morphonet.

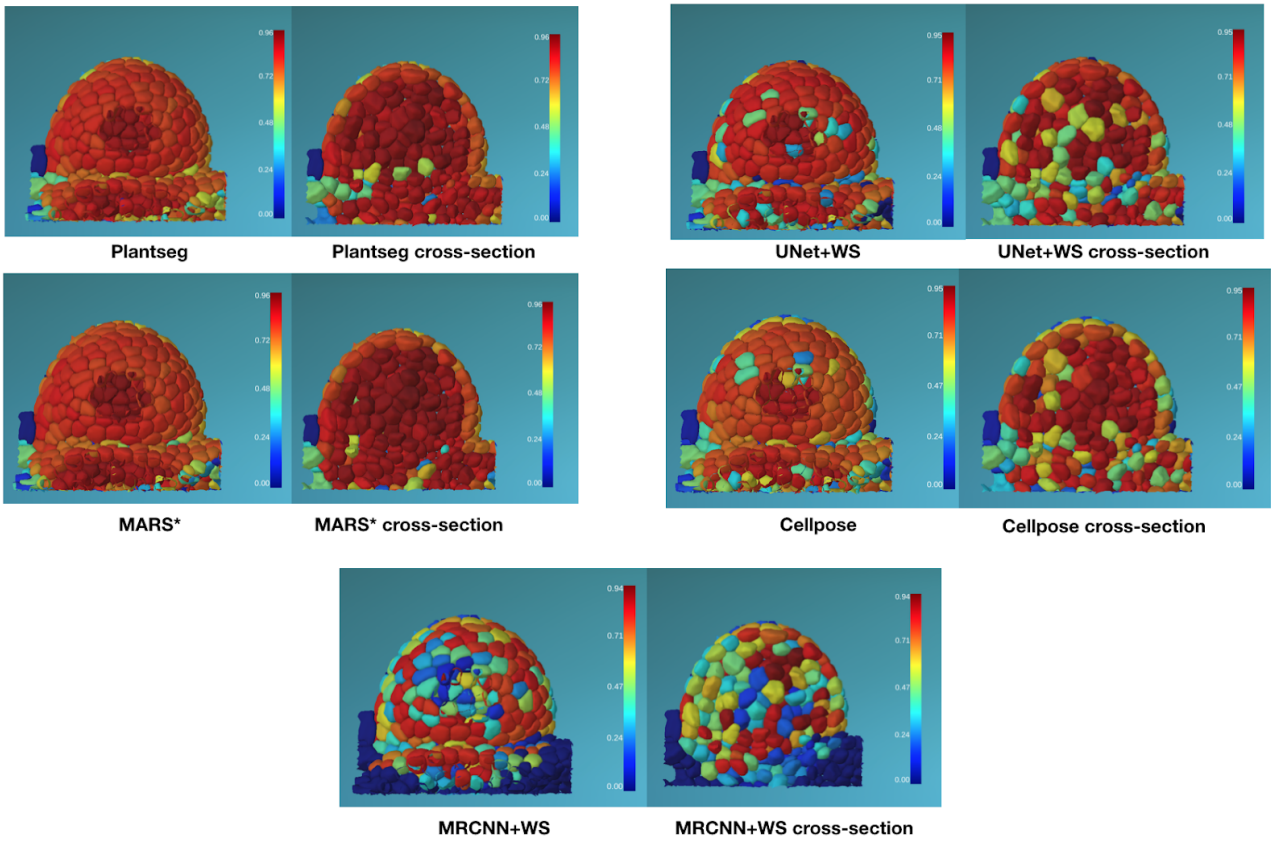

Figure 12. Segmentation quality results (VJI values) for a test stack (TS2-26h) from 5 pipelines displayed on Morphonet. Users can slice through each 3D stack in XYZ directions and check the property (here VJI values) for each cell in the interior layers of the tissue structure. For example, for each pipeline in the above figure the left image shows the full 3D stack and the right image shows the cross section of the same stack after slicing 50% in the Z direction. VJI values are projected as a "property" or colormap on the cells. In this figure a "jet" colormap is used where red represents high and blue represents low VJI values as shown in the color bars alongside.

Discussion

In this study we have analysed 4 deep learning segmentation pipelines and a non-deep learning one. Initially it was difficult to predict the relative performance of the individual pipelines, because they were applied on diverse datasets and used different evaluation metrics. Most common evaluation strategies consist of estimating simple numerical metric values, which also vary widely between the methods. For example in the Plantseg original paper (Wolny et al., 2020), the variation of information metric is used, for the original

Unet+WS paper, the aggregated Jaccard metric is used, whereas for Cellpose (Stringer et al., n.d.) an average precision metric is used. Also, besides the conventional Jaccard Index, the aggregated Jaccard index metric (AJI) is used in papers like (Eschweiler et al., 2019) and (Kumar et al., 2017), in which the AJI not only estimates the overlap of the true positives but also accounts for false positives and false negatives present in the segmentations. The numerical results from these metrics are difficult to compare since they are based on different concepts. Another problem is that these estimates do not provide a detailed view about the types of segmentation errors present in the segmentation results or how the errors are spatially distributed. Therefore, we have provided full details and coding implementations of the metrics used in this study so that they can be re-used by others. In addition to the VJI metric we evaluated the segmentations by measuring the rates of over- and under-segmentations which indicates the strengths and weaknesses of specific segmentation procedures. This spatial distribution of segmentation quality was further addressed through a layerwise evaluation in this paper. The range of metrics presented here provided an exhaustive analysis of the type and magnitude of the errors as well as their spatial distributions produced. Each of the metrics gives its own specific information which finally allowed us to define the advantages and shortcomings of each segmentation pipeline.

The MorphoNet based interactive evaluation is a fast and efficient way to visualize segmentation quality superposed on 3D representations. It comprises 3 steps - calculation of cell by cell VJI values for a pair of stacks, creating a 3D mesh and uploading these on Morphonet using their Python API (Appendix 2). All these steps are covered in two Python notebooks provided in the SegCompare Gitlab repository (Appendix 1). With this visualization, users can navigate through 3D objects in an image and study the segmentation quality for different 3D sections by slicing through the image data (converted to 3D meshes). Since the Morphonet visualization is browser based, it does not require any software installation or special hardware for 3D data visualization. This technique may therefore be added as a final step to any segmentation pipeline- deep learning or non deep learning, for a one-step analysis of the 3D segmentation performance.

We also tried to evaluate how much better the deep learning algorithms performed compared to the non-deep learning algorithms which is a recurrent question in deep learning and computer vision research. Results of PlantSeg, the best performing deep learning pipeline of those tested here, were matched by those of the non-deep learning MARS algorithm. However, the issue with MARS is that getting optimal results is time and effort consuming since re-tuning is necessary for each dataset and tissue image. Without tuning, the MARS accuracy may degrade significantly over a dataset as observed from our optimized (MARS*)and non-optimized MARS results. By contrast, deep learning models, especially PlantSeg, once trained, performed well throughout. A comparison of performance of the pipelines on a completely different dataset (Phalusia mamaliata embryo 3D images captured using light sheet microscopy) are presented in Appendix 5 which shows that Plantseg produces high accuracy results without retraining on other tissues as well. Results of Plantseg and MARS optimised for each Phallusia stack (called here MARS^{*PM}) are found to be best for the embryo images closely followed by the Unet+WS pipeline.

The quantitative impact of image artefacts on the accuracies of segmentation pipelines are not always analysed in literature. This is why we dedicated part of our study to estimate the impact of image artefacts like blur, over and under exposure and 3 levels of Gaussian noise on the VJI values and rates of over and under-segmentation of the 5 segmentation pipelines. The size of our test dataset therefore consisted of a

total of 60 3D stacks with 10 original and 50 synthetically modified ones. In addition, completely unseen 3D images from plant and animal datasets were also segmented using our pipelines as described in Appendix 5.

Several features regarding deep learning based segmentation pipelines were observed during their testing in this study. Although 2D segmentation pipelines may be adapted to perform 3D segmentations (e.g. as done in Cellpose and MRCNN+WS), the end to end straight 3D segmentation pipelines (such as Plantseg, Unet+WS pipelines) achieved higher accuracies. This trend is observed in the segmentation accuracy results of the pipelines for 10 original stacks as well as with the stacks having artefacts. We did, however, also observe a difference in performance between the two end-to-end 3D pipelines (Plantseg and 3DUnet+WS)., Plantseg provides two semantic output classes (background and boundaries) while 3DUnet+WS produces three classes (background, boundary and cell interiors). For instance segmentation, Plantseg relies on graph partitioning while Unet_WS uses watershed. Thus, the question arises as to whether the accuracy is affected by the deep learning architecture itself or by the way the full segmentation pipeline is constructed.

Do deep learning networks have a performance limit or can they be further improved? New artificial intelligence concepts such as Single Shot MultiBox Detectors (SSD) or context encoder networks (Yi, Wu, Hoeppner, & Metaxas, 2018) are currently emerging. At this stage they still need further development to improve their technical capabilities for 3D analysis. This clearly illustrates, however, the need for reliable approaches to quantitatively test their accuracy and understand their qualities and characteristics.

Data and code availability

All the data and coding implementations from this work are available as open resources. The methods for reproduction of the segmentation pipelines and the segmentation evaluation techniques are documented in the Gitlab repository named SegCompare (https://mosaic.gitlabpages.inria.fr/publications/seg_compare) along with relevant resources in Jupyter notebooks. The 3D confocal training and test datasets used in this work are provided in open data repositories. These are described in detail in Appendix 1 and 3.

Materials and Methods

Training data

The training data for all the deep learning algorithms comprise 3D confocal image stacks and their corresponding ground truth segmentations. The structure of a 3D confocal image is shown in Figure 13.

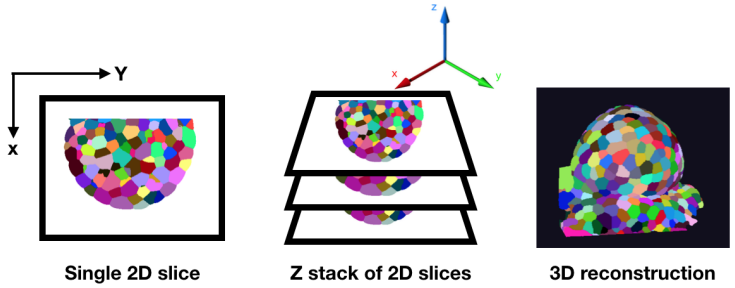

Figure 13 A confocal image is made up by scanning through each point on a 2D plane of an object. The 3D confocal image is made up of such 2D frames stacked along the Z axis. Using the 2D Z slices a full 3D view of the object can be reconstructed.

The source of the training data is Arabidopsis thaliana shoot apical meristems (SAM) which is a multicellular tissue where cells at the surface grow radially outward from the central to the peripheral zone. This data has been previously used and published in (Willis et al., 2016). The SAM 3D stacks were captured using a time-lapse confocal microscope for every 4 h for 0 to~80 h and were passed through a pipeline of image correction steps. The mean size of the stacks is 150x512x512 pixels (zyx dimensions).

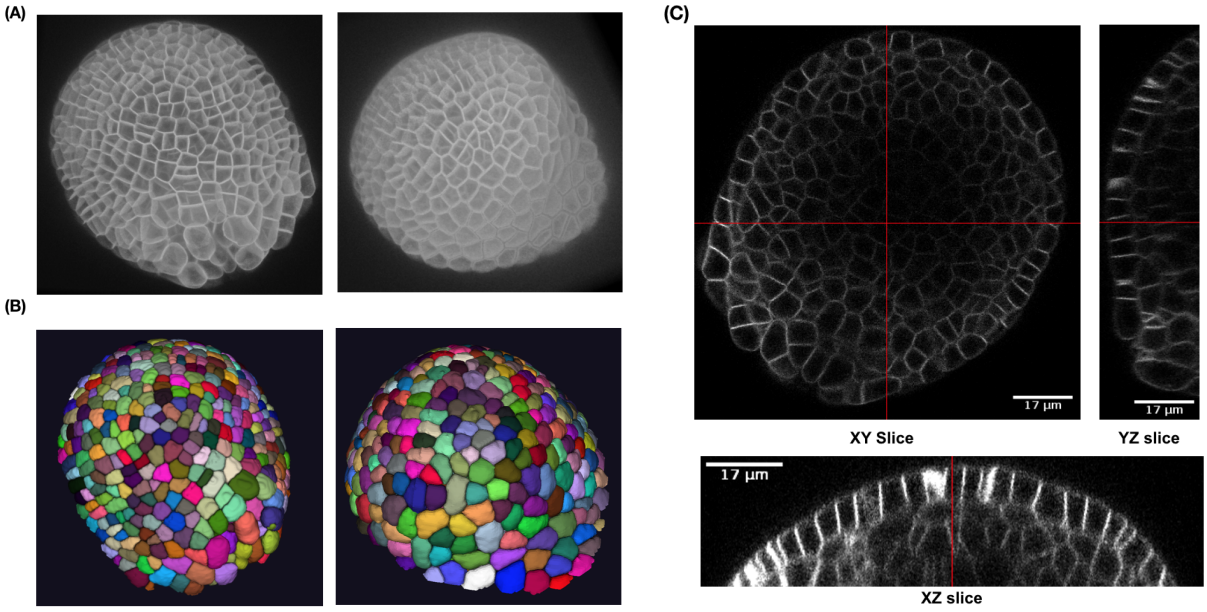

Figure 14 (A) 3D projection of two training images and (B) corresponding ground truth segmentations (C) Lateral (XY) and axial slices (XZ, YZ) of a sample confocal training image

For each stack, slice mis-alignments due to vibrations or microscope stage movements were corrected via translation transformations using the StackReg module of ImageJ. Also, z-slices with horizontal shifts were replaced with the closest z-slice with no shifts. Imaging errors during vertical movement of the plant due to growth were also compensated for by estimating stretching constants for each stack by comparing rapidly acquired low-z-resolution stacks with slowly acquired high-z-resolution stacks [(Willis et al., 2016),S1,Table 10]. In addition, Gaussian and an alternative-sequential filtering was done for noise removal.

The ground truth data for training consists of the 3D segmentations of the above image stacks done by 3D watershed followed by extensive manual corrections on a slice by slice basis. For corrections of segmentation errors, cell boundaries were estimated from the segmented images superposed on the original images for visual inspection. For errors due to over or under-segmentation or missing cells, the noise filter and watershed parameters were adjusted until satisfactory segmentations were obtained for peripheral as well as central zones of the SAM images. In the ground truth images, voxels which belong to the same cell, have the same label.

Test dataset

For testing the segmentation algorithms, a dataset of ten 3D confocal stacks was used (described in (Refahi et al., 2021)). First 6 stacks (0h, 24h, 32h, 72h, 120h, 132h timepoints) are from one meristem (FM1 in

Refahi et al., 2021) while the time points 26h, 44h, 56h and 69h are from another (FM6). Images of the stacks are shown in Figure 4. These images are from the floral meristem of Arabidopsis from initiation to stage 4. Corrections for alignment and vertical movements are described in Refahi et al. (2021). Expert ground truth segmentations of these test stacks were also available to perform the numerical comparisons.

Training and segmentation details for deep learning pipelines

The four DL algorithms were first trained using the common meristem dataset. The details on how these pipelines could be reproduced are described in the Gitlab repository i(Appendix 1). For training the deep learning networks and testing, a CUDA enabled NVIDIA Quadro P5000 GPU was used with a Intel Xeon 3 GHz processor. Python 3.x is the default programming language for training/testing all the segmentation pipelines and implementing the evaluation metrics.

Plantseg: The residual 3D UNet as described in (Wolny et al., 2020) was used for training on our data. The Residual 3D U-Net is a variant of the 3D U-Net and comprises a contracting encoder and an expanding decoder part but uses residual skip connections in each convolutional block in the U-Net. For final segmentation of our test data, all three of the graph partitioning post-processing strategies (GASP, Mutex, Multicut) were found to provide similar results, so GASP was used to obtain all the final segmentations from Plantseg that are evaluated in this work (Figure 15).

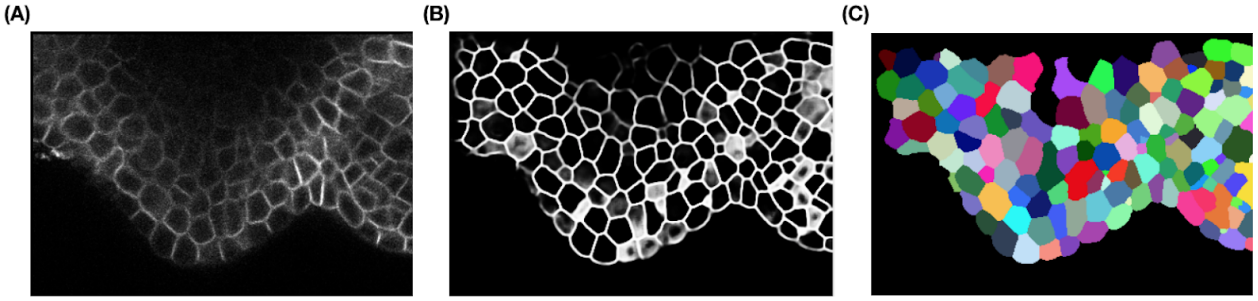

Figure 15 Plantseg workflow: (A) Input image (B) Boundary prediction (C) Final segmentation

Unet+Watershed: The 3D UNet module of the pipeline was trained using the custom training dataset described above. The 3D Unet predicts 3 classes of output images from an input 3D confocal stack. These classes are: cell centroids, cell membranes and background maps (Fig. 16(b-d)). Dimensions of these 3 output images are the same as that of the input stack.

Using these 3 predicted image classes a seed image is obtained by first thresholding the centroid maps (0.8 times the max intensity), followed by its morphological opening (circular kernel, size 5x5x5) and subtracting the membrane and background maps from the resultant image. An example seed image is shown in Figure 16(F) and the corresponding seeded watershed segmentation output is shown in Figure 16G.

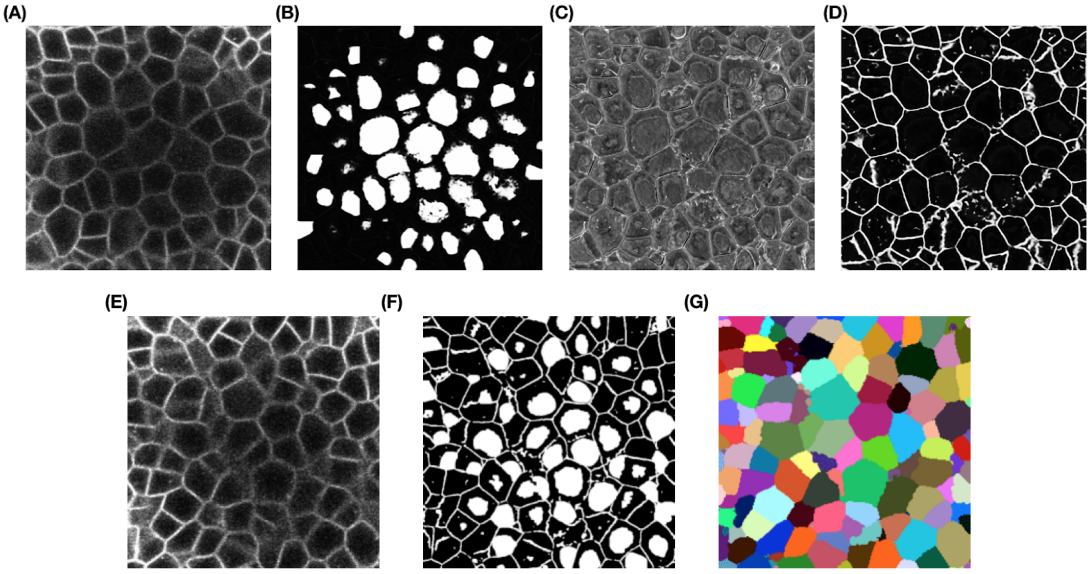

Figure 16 3D Unet+ WS workflow (A) An input confocal image (xy slice) (B) Class 0 prediction- centroids (C) Class 1 prediction- background (D) Class 2 output- cell membranes. (E) An input confocal image (xy slice) (F) Seed image slice (G) Final segmented slice using watershed on (F)

Cellpose: The Cellpose pipeline uses a 2D Unet to predict horizontal (X) and vertical (Y) flows along with a pixel probability map for each 2D test image. Using the XY intensity gradients or vector fields, pixels belonging to each object to be segmented can be aggregated around the centroid region for that object. For the final segmentation in 3D of the test sets, Cellpose uses the 2D trained model to predict the horizontal and vertical gradients for each of the XY, XZ and YZ sections of a 3D volume. These six predicted gradients are then averaged pairwise to obtain the final XYZ vector map in 3D.

The 2DUnet module was trained using 2D slices (512x512 pixels) from the custom training dataset along with their corresponding 2D masks. The tunable parameters of the method such as flow_threshold, cell probability threshold and cell diameter were set to the default values of 0.4, 0.0 and 30 pixels respectively.

Mask RCNN: Mask RCNN uses a backbone network such as Resnet-50 or Resnet-101 to extract image features followed by generating region proposals of objects (to be segmented) using a region proposal network (RPN). These region proposals are refined and fed to a fully convolutional classifier to identify object classes. The final output of MRCNN includes 1) boundary box for each object instance 2) pixel level mask for each object identified 3) class predictions for each object instance 4) confidence score of each prediction.

In the MRCNN with watershed pipeline, using a 2D trained Mask-R CNN model the cell regions in each Z slice of a 3D volume are predicted (Fig. 17 C). The predicted Z slices with the identified cell regions are stacked together to produce a 3D binary seed image which is then labelled using a 26-neighbor connected components labelling method . The labelled seed image is then used for watershed based post-processing to obtain the final 3D instance segmentation. The Mask-RCNN algorithm with a Resnet 101 backbone network was trained using 2D images along with instance masks for each object (to be segmented) in the image. Using Resnet 50 as a backbone network did not give satisfactory results. Training images (2D slices) from the custom training dataset were used and the instance masks were generated from the corresponding

ground truth 2D masks (each labeled cell region forms an instance mask) as shown in Figure 17(A). Both the raw image and the set of instance masks for each image are then used for training the MaskRCNN network.

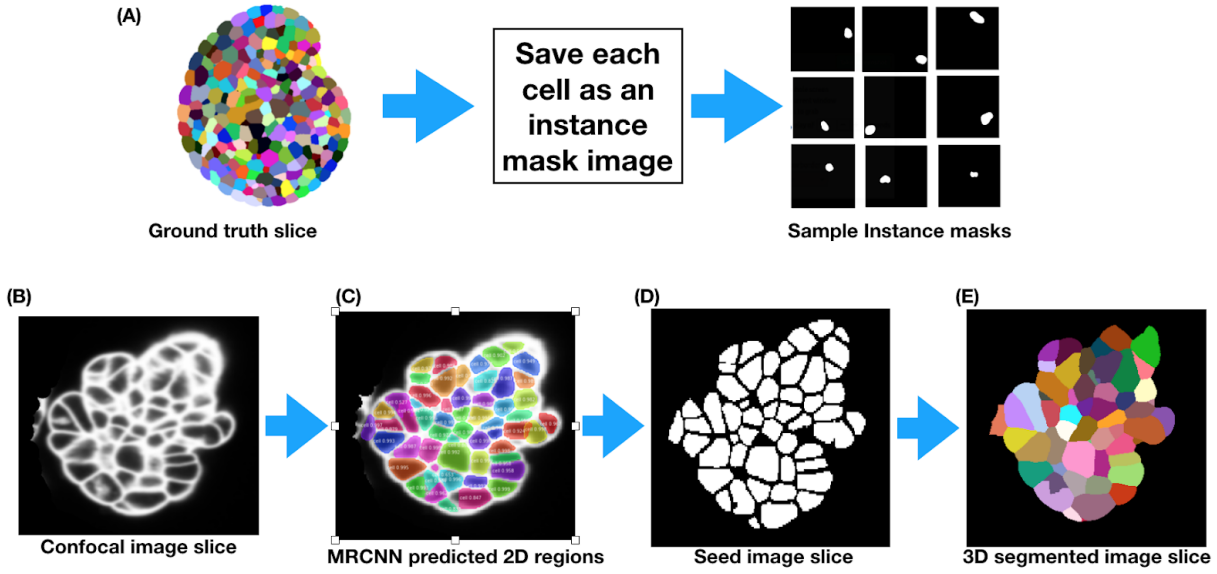

Figure 17 MRCNN+Watershed workflow: (A) Creation of instance masks for training MRCNN (B) Example confocal slice (C) 2D predictions by MRCNN (D) Binary seed image created from identified cell regions in (C).(E) Same slice after 3D segmentation using watershed on the binary seed image.

Pipelines	Parameters to tune
Plantseg	Pre-processing: Interpolation (upto degree 2), Gaussian or median filter Post-processing: CNN predictions threshold, watershed seeds sigma, watershed boundary sigma, superpixels minimum size (voxels), cell minimum size (voxels)
Unet+Watershed	Pre-processing: None Post-processing: Morphological opening/erosion kernel sizes for estimating watershed seeds from Unet outputs.
Cellpose	Pre-processing: None Post-processing: Cell diameter, cell probability threshold, flow error threshold
MRCNN+ Watershed	Pre-processing: None Post-processing: Morphological opening/erosion kernel sizes for estimating watershed seeds from MRCNN outputs.
MARS	h-minima, Gaussian smoothing sigma for image and seeds

Table 2: Parameters to tune for each segmentation pipeline

Evaluation metrics

Volume averaged Jaccard index. The Jaccard index is a metric which estimates the similarity between two regions of labelled images, G and P, in terms of the intersection between them divided by their union (Taha & Hanbury, 2015). Let us denote G_i and P_j these two overlapping regions, their Jaccard index is defined as:

$$JI(G_i, P_j) = \frac{|G_i \cap P_j|}{|G_i \cup P_j|} \quad (1)$$

where |X| denotes the volume of region X. We also define an asymmetric inclusion index metric between two regions defined as:

$$I\left(G_{i,}\ P_{j}\right) = \frac{\left|G_{i} \cap P_{j}\right|}{\left|G_{i}\right|} \tag{2}$$

Given a set of labelled regions G_i in the image G, we associate with each cell region index i a cell region index $A(i) = j$ in the image P containing the predicted segmentations P_j, using the Jaccard index:

$$A(i) = \underset{j\in[2,N]}{\arg\max}\ JI(G_i, P_j) \tag{3}$$

Note that if the cell G_i has no intersection with any cell of P, we set $A(i) = 0$. Similarly, using the asymmetric index each region G_i of the image G is associated with one region index $B(i) = j$ in the image P according to:

$$B(i) = \underset{j\in[1,N]}{\arg\max}\ I(G_i, P_j) \tag{4}$$

Reciprocally, each region P_j of the image P is associated with one region index $B'(j) = i$ in the image G:

$$B'(j) = \underset{i\in[1,M]}{\arg\max}\ I(P_j, G_i) \tag{5}$$

We then define an average metric between two images P and G, called Volume averaged Jaccard, that assesses how well the regions of two images overlap Index (VJI):

$$VJI(G,P) = \frac{\sum_{i=2}^{M} |G_i| JI(G_i, P_{A(i)})}{\sum_{i=2}^{M} |G_i|} \tag{6}$$

where G_i and P_j represent regions in respectively images G and P corresponding to either ground truth or predicted labeled cells. In this equation, it is assumed that $JI\left(G_{i}, P_{0}\right) = 0$ for any i. Background regions are not included in this estimation and in both the segmented and ground truth images, the background label is set as 1.

Rates of over and under segmentation to evaluate the quality of 3D segmentation methods. The Volume averaged Jaccard index is a measure that quantifies the degree of overlap between two segmentations although it does not indicate whether the cell segmentation errors are due to over or under-segmentation. In order to detect these different type of errors, we use the asymmetric index (2) to automatically determine a region-to-region correspondence map (Michelin, 2016).

Using the previous inclusion index metrics, a correspondence map is obtained by : 1) making region-to-region association from the segmentation G to segmentation P then 2) repeating the procedure in the opposite direction ie. from image P to image G before 3) building the resulting reciprocal associations between sets of regions.

The two first steps consist of computing two values $B(i)$ and $B'(j)$ for each region index i of the first image and each region index j of the second image using (4) and (5) respectively. Using the previous computed indexes, set of pair of associated regions index between image P and G can be defined by:

$$\mathcal{A} = \{(i,j) \mid B(i) = j \;\; or \;\; B'(j) = i)\} \tag{7}$$

Let us define the two subsets $\mathcal{A}(i) = \{j \mid \; (i,j) \in \mathcal{A}\}$ and $\mathcal{A}'(j) = \{i \mid \; (i,j) \in \mathcal{A}\}$ corresponding to the region indexes j associated with a given region index i and the region indexes i associated with a region index j respectively. We then consider the different cases of resulting reciprocal mapping:

- one-to-one (exact match between G_i and P_j) if $\mathcal{A}(i) = \{j\} \;\; and \;\; \mathcal{A}'(j) = \{i\}$.
- one-to-many (over-segmentation of G_i) if $|\mathcal{A}(i)| > 1 \;\; and \;\; \forall j \in \mathcal{A}(i), \; \mathcal{A}'(j) = \{i\} \; or \; \mathcal{A}'(j) = \emptyset$.
- many-to-one (under-segmentation of G_i) if $|\mathcal{A}'(j)| > 1 \;\; and \;\; \forall i \in \mathcal{A}'(j), \; \mathcal{A}(i) = \{j\} \; or \; \mathcal{A}(i) = \emptyset$.
- many-to-many otherwise

It has to be noted that the correspondence involving the image background is treated separately, ie. without considering a reciprocal association: the regions of segmentation G that maximize their inclusion with the background of the image P are associated and vice versa. From the resulting reciprocal mapping, a global rate of over and under segmentation can be calculated by counting the number of voxels of the over or under correspondence regions in the image P.

Finally, a tolerance on the region border position can be introduced by considering an eroded region G_i^r in $I(G_i^r, P_j)$ and reciprocally, an eroded region P_j^r in $I(P_j^r, G_i)$ where r is the radius of a sphere (structuring element). The higher the parameter, the more one-to-one correspondences will be found by the method. In the present study, the parameter r was fixed to 2 for all the comparisons.

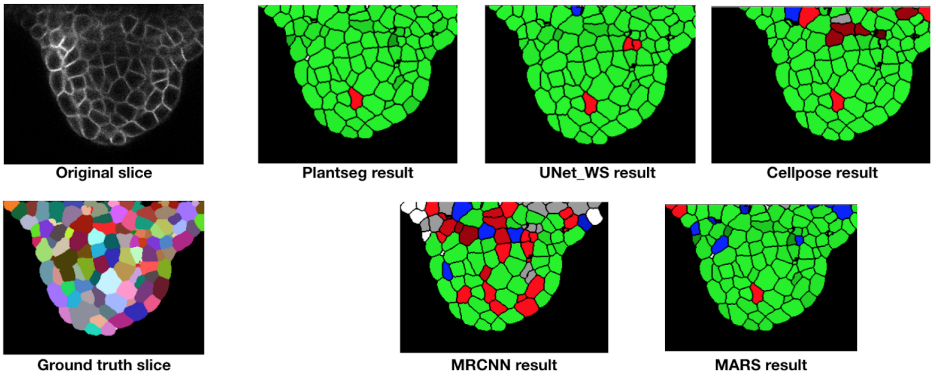

Figure 18 Segmentation quality metric (Michelin, 2016) applied to outputs from 5 segmentation pipelines and types of errors displayed as a colormap (on a common Z slice). The green cell regions represent regions of complete overlap between ground truth and predicted segmentations (i.e regions of fully correct segmentation). Red regions represent over and blue regions represent under-segmentation errors. White regions are regions where cells were mistaken for background. The benefit of this metric is that it helps to estimate the rate of over and under-segmentations as a volumetric statistics and as spatial distributions.

Simulation of image artefacts

The effects of noise, blur and intensity variations are simulated on the test set of 10 confocal images to evaluate their impact on the segmentation quality of the pipelines. The procedure for simulation of these artefacts are described below.

Image noise:, The Gaussian noise was added to the images. The noise variable z is represented as the probability density function (PDF) P(z) and given by:

$$P(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(z-\mu)^2}{2\sigma^2}} \quad (5)$$

where μ is the mean and σ is the standard deviation (or the square of the variance). The Gaussian distributions for different variance values are plotted in Figure 17A. Gaussian noise was generated with different values of noise variance ([0.00, 0.04, 0.08]) and added to an image as shown in Figure 7A.

Image blur: To simulate motion blur, the test confocal image i(x,y) is convolved with a horizontal motion blur kernel w(dx, dy) of size 4x4 to get the final blurred image b(x,y) .In frequency domain this is given by the following relation:

$$b(x,y) = w * i(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} w(dx,dy)\, i(x+dx,\, y+dy) \quad (6)$$

Image intensity variations: Partially bright regions in microscopy images may be caused by inhomogeneous illumination sources and shadow effects are mostly caused by presence of light absorbing objects or obstructions(Cheng & Kriete, 1990; Ricci et al., 2020). To emulate the effect of intensity variations within an image, partial overexposure (Figure 19A) and random shadow regions (Figure 19B) are imposed (individually) on the test images. In order to impose the partial overexposure effect, for each Z slice of a given 3D test stack, a brightness mask is created, which is a 2D array having the same size as the x, y dimensions of the 3D stack. This 2D brightness mask array is filled with gray integer values of 255 for the left half of the mask array (Figure 19A). This brightness mask is then numerically added to each Z slice array with 30% transparency (using OpenCV function cv2.addweighted) to obtain the partially brightened image array as shown in Figure 19A.

For creating the randomly shadowed or under-exposed regions, a shadow mask is created with random dark geometrical patch areas for each 2D Z slice of a 3D test stack. The dark patches are created by reducing image pixel intensities within the patch areas by a factor of 30% of the original intensity values. Figure 19 A and B shows the intensity profiles within the yellow boundary boxes impacted due to the over and under-exposure effects respectively. Within the over-exposed regions, image intensities are higher than original values and vice-versa for under-exposed regions.

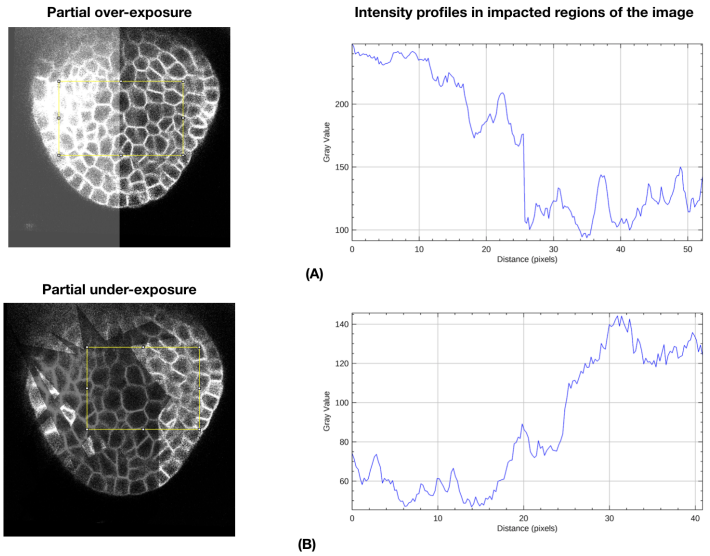

Figure 19 Modification of image intensity (inside selected area within the yellow box) (a) image intensity transition under partial overexposure (b) image intensity variations due to imposition of underexposure.

Acknowledgement

We would like to thank Grégoire Malandain for critical reading of the text, Alexandre Cunha, Johannes Stegmaier for advice on the UNet + WS implementation. Johathan Legrand is thanked for help with the implementation of MARS and building the Gitlab repository. We would also like to thank Emmanuel Faure for his guidance on the use of the Morphonet platform. This study was supported by the Agence Nationale de la Recherche-ERA-CAPS grant, Gene2Shape (17-CAPS-0006-01).

Competing interests:

The authors declare no competing interests

Bibliography

Al-Kofahi, Y., Zaltsman, A., Graves, R., Marshall, W., & Rusu, M. (2018). A deep learning-based algorithm for 2-D cell segmentation in microscopy images. *BMC Bioinformatics*, *19*(1), 365.

Arbelle, A., & Raviv, T. R. (2018). Microscopy cell segmentation via adversarial neural networks. *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)* (pp. 645–648). Presented at the 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), IEEE.

Bailoni, A., Pape, C., Wolf, S., Beier, T., Kreshuk, A., & Hamprecht, F. (2019). A

Generalized Framework for Agglomerative Clustering of Signed Graphs applied to Instance Segmentation. *ArXiv*, *abs/1906.11713*.

Barbier de Reuille, P., Routier-Kierzkowska, A.-L., Kierzkowski, D., Bassel, G. W., Schüpbach, T., Tauriello, G., Bajpai, N., et al. (2015). MorphoGraphX: A platform for quantifying morphogenesis in 4D. *eLife*, *4*, 05864.

Caicedo, J. C., Roth, J., Goodman, A., Becker, T., Karhohs, K. W., Broisin, M., Molnar, C., et al. (2019). Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *Cytometry. Part A: the Journal of the International Society for Analytical Cytology*, *95*(9), 952–965.

Cheng, P., & Kriete, A. (1990). Image Contrast in Confocal Light Microscopy.

Cremers, D., Rousson, M., & Deriche, R. (2007). A Review of Statistical Approaches to Level Set Segmentation: Integrating Color, Texture, Motion and Shape. *International journal of computer vision*, *72*(2), 195–215.

Dawoud, Y., Hornauer, J., Carneiro, G., & Belagiannis, V. (2020). Few-Shot Microscopy Image Cell Segmentation. *ArXiv*, *abs/2007.01671*.

Ding, K. (2018). A Simple Method to improve Initialization Robustness for Active Contours driven by Local Region Fitting Energy. *CoRR*, *abs/1802.10437*.

Eschweiler, D., Spina, T. V., Choudhury, R. C., Meyerowitz, E., Cunha, A., & Stegmaier, J. (2019). CNN-based preprocessing to optimize watershed-based cell segmentation in 3D confocal microscopy images. *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)* (pp. 223–227). IEEE.

Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., et al. (2019). U-Net: deep learning for cell counting, detection, and morphometry. *Nature Methods*, *16*(1), 67–70.

Feltell, D., & Bai, L. (2010). A New Marching Cubes Algorithm for Interactive Level Set with Application to MR Image Segmentation. *ISVC*.

Fernandez, R., Das, P., Mirabet, V., Moscardi, E., Traas, J., Verdeil, J.-L., Malandain, G., et al. (2010). Imaging plant growth in 4D: robust tissue reconstruction and lineaging at cell resolution. *Nature Methods*, *7*(7), 547–553.

Gharipour, A., & Liew, A. W.-C. (2016). Segmentation of cell nuclei in fluorescence microscopy images: An integrated framework using level set segmentation and touching-cell splitting. *Pattern recognition*, *58*, 1–11.

Graham, S., Vu, Q. D., Raza, S. E. A., Azam, A., Tsang, Y. W., Kwak, J. T., & Rajpoot, N. (2019). Hover-Net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. *Medical Image Analysis*, *58*, 101563.

Gu, Z., Cheng, J., Fu, H., Zhou, K., Hao, H., Zhao, Y., Zhang, T., et al. (2019). CE-Net: Context Encoder Network for 2D Medical Image Segmentation. *IEEE transactions on medical imaging*, *38*(10), 2281–2292.

Guerrero-Peña, F. A., Marrero-Fernández, P. D., Tsang, I. R., & Cunha, A. (2019). A Weakly Supervised Method for Instance Segmentation of Biological Cells. *ArXiv*, *abs/1908.09891*.

Guignard, L., Fiúza, U.-M., Leggio, B., Laussu, J., Faure, E., Michelin, G., Biasuz, K., et al. (2020). Contact area-dependent cell communication and the morphological invariance of ascidian embryogenesis. *Science*, *369*(6500).

Hadj, S. B., Blanc-Féraud, L., Aubert, G., & Engler, G. (2013). Blind restoration of confocal microscopy images in presence of a depth-variant blur and Poisson noise. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 915–919.

Hafiz, A. M., & Bhat, G. M. (2020). A survey on instance segmentation: state of the art. *International journal of multimedia information retrieval*, *9*(3), 171–189.

Haider, S. A., Cameron, A., Siva, P., Lui, D., Shafiee, M. J., Boroomand, A., Haider, N., et al. (2016). Fluorescence microscopy image noise reduction using a stochastically-connected random field model. *Scientific Reports*, *6*, 20640.

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)* (pp. 2980–2988). Presented at the IEEE International Conference on Computer Vision (ICCV), IEEE.

Jiang, J., Kao, P.-Y., Belteton, S. A., Szymanski, D. B., & Manjunath, B. S. (2019). Accurate 3D cell segmentation using deep features and CRF refinement. *2019 IEEE International Conference on Image Processing (ICIP)* (pp. 1555–1559). Presented at the 2019 IEEE International Conference on Image Processing (ICIP), IEEE.

Johnson, J. (2018). Adapting Mask-RCNN for Automatic Nucleus Segmentation. *ArXiv*, *abs/1805.00500*.

Kallasi, F., Rizzini, D. L., Oleari, F., & Aleotti, J. (2015). Computer vision in underwater environments: A multiscale graph segmentation approach. *OCEANS 2015 - Genova* (pp. 1–6). Presented at the OCEANS 2015 - Genova, IEEE.

Kappes, J. H., Speth, M., Andres, B., Reinelt, G., & Schnörr, C. (2011). Globally Optimal Image Partitioning by Multicuts. *EMMCVPR*.

Kirschbaum, E., Bailoni, A., & Hamprecht, F. A. (2020). Disco: deep learning, instance segmentation, and correlations for cell segmentation in calcium imaging. In A. L. Martel, P. Abolmaesumi, D. Stoyanov, D. Mateus, M. A. Zuluaga, S. K. Zhou, D. Racoceanu, et al. (Eds.), *Medical image computing and computer assisted intervention – MICCAI 2020: 23rd international conference, lima, peru, october 4–8, 2020, proceedings, part V*, Lecture notes in computer science (Vol. 12265, pp. 151–162). Cham: Springer International Publishing.

Korfhage, N., Mühling, M., Ringshandl, S., Becker, A., Schmeck, B., & Freisleben, B. (2020). Detection and segmentation of morphologically complex eukaryotic cells in fluorescence microscopy images via feature pyramid fusion. *PLoS Computational Biology*, *16*(9), e1008179.

Kromp, F., Fischer, L., Bozsaky, E., Ambros, I., Doerr, W., Taschner-Mandl, S., Ambros, P.,

et al. (2019). Deep Learning architectures for generalized immunofluorescence based nuclear image segmentation. *ArXiv*, *abs/1907.12975*.

Kumar, N., Verma, R., Anand, D., Zhou, Y., Onder, O. F., Tsougenis, E., Chen, H., et al. (2020). A Multi-Organ Nucleus Segmentation Challenge. *IEEE transactions on medical imaging*, *39*(5), 1380–1391.

Kumar, N., Verma, R., Sharma, S., Bhargava, S., Vahadane, A., & Sethi, A. (2017). A dataset and a technique for generalized nuclear segmentation for computational pathology. *IEEE transactions on medical imaging*, *36*(7), 1550–1560.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Leggio, B., Laussu, J., Carlier, A., Godin, C., Lemaire, P., & Faure, E. (2019). MorphoNet: an interactive online morphological browser to explore complex multi-scale data. *Nature Communications*, *10*(1), 2812.

Lei, T., Wang, R., Wan, Y., Du, X., Meng, H., & Nandi, A. (2020). Medical Image Segmentation Using Deep Learning: A Survey. *ArXiv*, *abs/2009.13120*.

Liu, D., Zhang, D., Song, Y., Zhang, F., O'Donnell, L., Huang, H., Chen, M., et al. (2020). Unsupervised Instance Segmentation in Microscopy Images via Panoptic Domain Adaptation and Task Re-Weighting. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4242–4251.

Lugagne, J.-B., Lin, H., & Dunlop, M. J. (2020). DeLTA: Automated cell segmentation, tracking, and lineage reconstruction using deep learning. *PLoS Computational Biology*, *16*(4), e1007673.

Lux, F., & Matula, P. (2020). Cell Segmentation by Combining Marker-Controlled Watershed and Deep Learning. *ArXiv*, *abs/2004.01607*.

Michelin, G. (2016, October). *Outils d'analyse d'images et recalage d'individus pour l'étude de la morphogenèse animale et végétale* (Doctoral dissertation).

Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., & Van Valen, D. (2019). Deep

learning for cellular image analysis. *Nature Methods*, *16*(12), 1233–1246.

Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern recognition*, *26*(9), 1277–1294.

Payer, C., Štern, D., Feiner, M., Bischof, H., & Urschler, M. (2019). Segmenting and tracking cell instances with cosine embeddings and recurrent hourglass networks. *Medical Image Analysis*, *57*, 106–119.

Refahi, Y., Zardilis, A., Michelin, G., Wightman, R., Leggio, B., Legrand, J., Faure, E., et al. (2021). A multiscale analysis of early flower development in Arabidopsis provides an integrated view of molecular regulation and growth control. *Developmental Cell*, *56*(4), 540-556.e8.

Ren, M., & Zemel, R. S. (2017). End-to-End Instance Segmentation with Recurrent Attention. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 293–301). Presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE.

Ricci, P., Sancataldo, G., Gavryusev, V., Franceschini, A., Müllenbroich, M. C., Silvestri, L., & Pavone, F. S. (2020). Fast multi-directional DSLM for confocal detection without striping artifacts. *BioRxiv*.

Scherr, T., Löffler, K., Böhland, M., & Mikut, R. (2020). Cell segmentation and tracking using CNN-based distance predictions and a graph-based matching strategy. *Plos One*, *15*(12), e0243219.

Shu, J.-H., Nian, F.-D., Yu, M.-H., & Li, X. (2020). An Improved Mask R-CNN Model for Multiorgan Segmentation. *Mathematical Problems in Engineering*, *2020*, 1–11.

Stringer, C., Wang, T., Michaelos, M., & Pachitariu, M. (n.d.). Cellpose: a generalist algorithm for cellular segmentation. *Nature Methods*, *18*(1), 100–106.

Taha, A. A., & Hanbury, A. (2015). Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC medical imaging*, *15*(1), 29.

Thomas, R. M., & John, J. (2017). A review on cell detection and segmentation in microscopic images. *2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT)* (pp. 1–5). Presented at the 2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT), IEEE.

Tokuoka, Y., Yamada, T. G., Hiroi, N., Kobayashi, T. J., Yamagata, K., & Funahashi, A. (2018). Convolutional Neural Network-Based Instance Segmentation Algorithm to Acquire Quantitative Criteria of Early Mouse Development. *BioRxiv*.

Van Valen, D. A., Kudo, T., Lane, K. M., Macklin, D. N., Quach, N. T., DeFelice, M. M., Maayan, I., et al. (2016). Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. *PLoS Computational Biology*, *12*(11), e1005177.

Vicar, T., Balvan, J., Jaros, J., Jug, F., Kolar, R., Masarik, M., & Gumulec, J. (2019). Cell segmentation methods for label-free contrast microscopy: review and comprehensive comparison. *BMC Bioinformatics*, *20*(1), 360.

Vu, Q. D., Graham, S., Kurc, T., To, M. N. N., Shaban, M., Qaiser, T., Koohbanani, N. A., et al. (2019). Methods for segmentation and classification of digital microscopy tissue images. *Frontiers in bioengineering and biotechnology*, *7*, 53.

Wang, C., Zhang, X., Choi, H. J., Lin, B., Yu, Y., Whittle, C., Ryan, M., et al. (2019). Deep learning pipeline for cell edge segmentation of time-lapse live cell images. *bioRxiv*.

Wang, W., Taft, D. A., Chen, Y.-J., Zhang, J., Wallace, C. T., Xu, M., Watkins, S. C., et al. (2019). Learn to segment single cells with deep distance estimator and deep cell detector. *Computers in biology and medicine*, *108*, 133–141.

Willis, L., Refahi, Y., Wightman, R., Landrein, B., Teles, J., Huang, K. C., Meyerowitz, E. M., et al. (2016). Cell size and growth regulation in the Arabidopsis thaliana apical stem cell niche. *Proceedings of the National Academy of Sciences of the United States of America*, *113*(51), E8238–E8246.

Wolf, S., Pape, C., Bailoni, A., Rahaman, N., Kreshuk, A., Köthe, U., & Hamprecht, F. A. (2018). The Mutex Watershed: Efficient, Parameter-Free Image Partitioning. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV*, Lecture notes in computer science (Vol. 11208, pp. 571–587). Cham: Springer International Publishing.

Wolny, A., Cerrone, L., Vijayan, A., Tofanelli, R., Barro, A. V., Louveaux, M., Wenzl, C., et al. (2020). Accurate and versatile 3D segmentation of plant tissues at cellular resolution. *eLife*, *9*.

Yang, Lin, Zhang, Y., Guldner, I. H., Zhang, S., & Chen, D. Z. (2016). 3D Segmentation of Glial Cells Using Fully Convolutional Networks and k-Terminal Cut. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, & W. Wells (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, Lecture notes in computer science (Vol. 9901, pp. 658–666). Cham: Springer International Publishing.

Yang, Linfeng, Ghosh, R. P., Franklin, J. M., Chen, S., You, C., Narayan, R. R., Melcher, M. L., et al. (2020). NuSeT: A deep learning tool for reliably separating and analyzing crowded cells. *PLoS Computational Biology*, *16*(9), e1008193.

Yi, J., Wu, P., Hoeppner, D. J., & Metaxas, D. (2018). Pixel-wise neural cell instance segmentation. *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)* (pp. 373–377). Presented at the 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), IEEE.

Yi, J., Wu, P., Jiang, M., Huang, Q., Hoeppner, D. J., & Metaxas, D. N. (2019). Attentive neural cell instance segmentation. *Medical Image Analysis*, *55*, 228–240.

Zaki, G., Gudla, P. R., Lee, K., Kim, J., Ozbun, L., Shachar, S., Gadkari, M., et al. (2020). A deep learning pipeline for nucleus segmentation. *BioRxiv*.

Zeng, T., Wu, B., & Ji, S. (2017). DeepEM3D: approaching human-level performance on

3D anisotropic EM image segmentation. *Bioinformatics*, *33*(16), 2555–2562.

Zhao, Z., Yang, L., Zheng, H., Guldner, I. H., Zhang, S., & Chen, D. Z. (2018). Deep learning based instance segmentation in 3D biomedical images using weak annotation. In A. F. Frangi, J. A. Schnabel, C. Davatzikos, C. Alberola-López, & G. Fichtinger (Eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part IV*, Lecture notes in computer science (Vol. 11073, pp. 352–360). Cham: Springer International Publishing.

Zheng, Q., Dong, E., Cao, Z., Sun, W., & Li, Z. (2014). Active contour model driven by linear speed function for local segmentation with robust initialization and applications in MR brain images. *Signal processing*, *97*, 117–133.

Zhu, N., Liu, C., Singer, Z. S., Danino, T., Laine, A., & Guo, J. (2020). Segmentation with Residual Attention U-Net and an Edge-Enhancement Approach Preserves Cell Shape Features. *ArXiv*, *abs/2001.05548*.

Supplementary information

Appendix 1

Gitlab repository SegCompare

SegCompare is an open repository of resources to train, test and evaluate multiple deep learning or non-deep learning algorithms (that are described in this article) for instance segmentation of 3D confocal images and compare their segmentation quality both in a quantitative and visual manner. All the segmentation pipelines and evaluation methods in the repository are implemented using Python programming language. SegCompare resources are user friendly and contain:

a) Steps for replicating deep learning and non-deep learning segmentation pipelines on custom user data . This includes steps for retraining the deep learning models used in this article as well as using them for directly segmenting user data.

b) Methods for quantitative evaluation of segmentation quality for a given segmented image and its ground truth image

c) Methods for 3D visualization of segmentation quality on the Morphological browser interface named Morphonet.

Link to the Gitlab page:https://mosaic.gitlabpages.inria.fr/publications/seg_compare

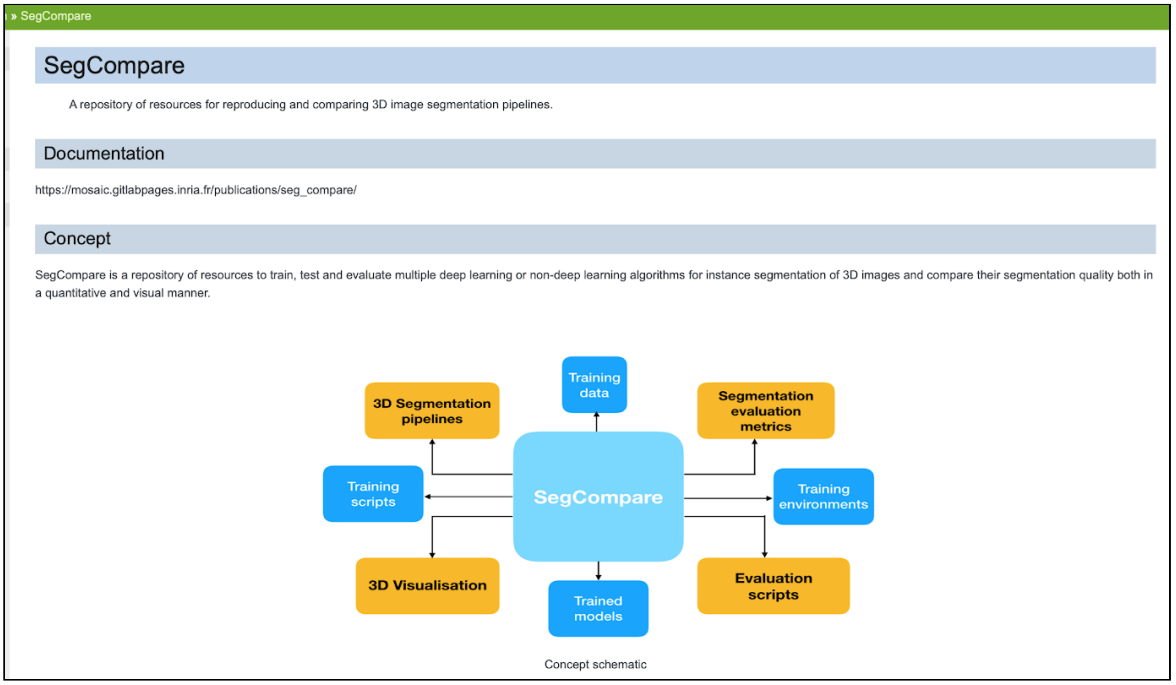

Figure A1.1 The SegCompare repository on Gitlab hosting the resources for training and evaluation of segmentation pipelines described in this paper

The utilities of this repository are the following:

For users wanting to segment their data: They can directly use one of the trained models and pipelines described here to segment their data.

For users designing a new 3D segmentation pipeline: They can use the fully annotated image datasets and evaluation methods to estimate the quality of their method.

For users having segmented data and expert ground truth: They can use this repository to evaluate the quality of their segmentations with quantitative metrics and 3D visualizations.

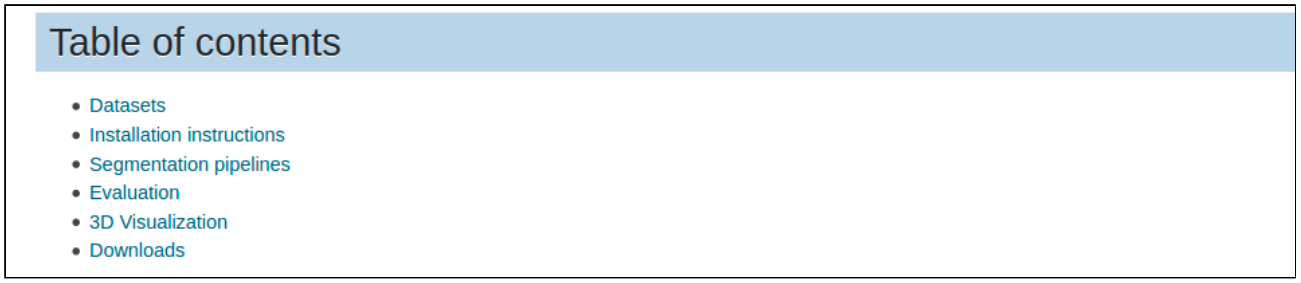

Figure A1.2 Contents of the SegCompare repository

The contents of SegCompare are as follows:

Datasets: A description of the confocal image datasets used for training and testing the segmentation pipelines described here. The links to the actual images are provided in Appendix 3.

Installation instructions: Each pipeline has its own library dependencies and therefore needs dedicated environments for running. For this, Python environments for each pipeline may be set up using yaml files, which are also provided in the SegCompare repository. The installation Instructions section provides detailed instructions to install environments for the segmentation pipelines, evaluation and visualization methods. After installing the environments, users can run the segmentation pipelines (for training or testing) or the evaluation and visualization methods.

Segmentation pipelines: Brief descriptions of each pipeline along with details of their pre- and post processing steps are presented here. Steps for dataset preparation for training the pipeline and links to training dataset are provided.

Evaluation: This section provides details of the segmentation evaluation metrics (Volume averaged Jaccard Index, Rates of over and under segmentation) and Jupyter (Python) notebooks for implementing them on a pair of segmented images. For evaluating segmentations a user must have a segmented image and corresponding ground truth segmentation (currently .tif format is supported for images). Sample segmented and ground truth data are in the data repository described in Appendix 3.

3D Visualization: This section describes how the browser based Morphological data visualization platform Morphonet may be used for visualizing segmentation quality. A Jupyter notebook is provided which contains the full implementation of the visualization pipeline starting from a segmented image. For this visualization, results from the Jaccarding Index evaluation notebook are required and the full workflow is documented in the repository. Links to sample meshes and datasets for uploading to Morphonet are provided along with demo videos (described in Appendix 3).

Downloads: This section contains links to Jupynter notebooks for implementations of the MARS pipeline, segmentation evaluation metrics and 3D visualizations. Sample CSV files containing Jaccard index estimates and corresponding mesh files for uploading to Morphonet are in the data repository (see Appendix 3) .

Appendix 2

Morphonet based visualization of segmentation quality

To visualize quality of the five segmentation pipelines on Morphonet, users can click the link below which will directly take them to an uploaded dataset with segmentation quality information from the 5 pipelines (MorphoNet works best with Chrome or Firefox) :
https://morphonet.org/icRos2mO

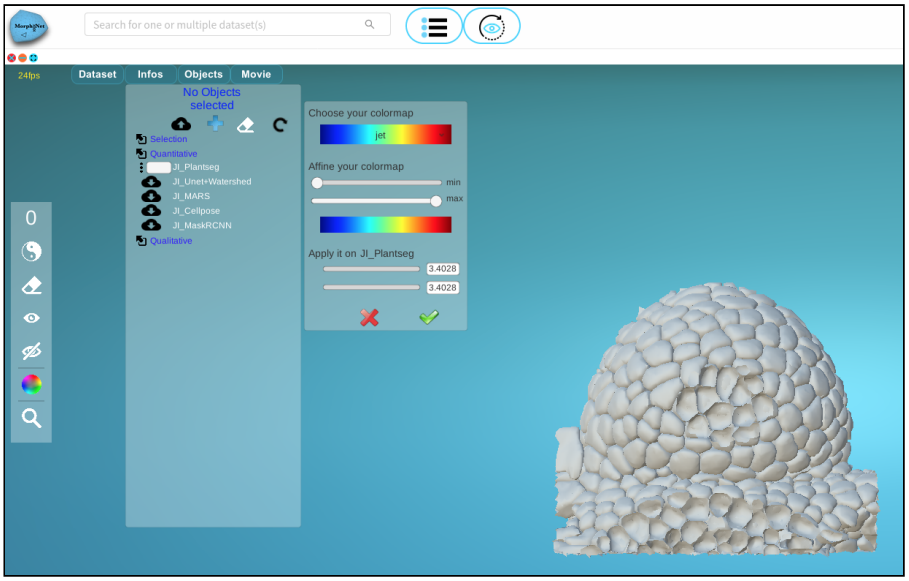

Figure A2.1 A mesh uploaded on Morphonet (right side of window). Information to be displayed is found by clicking the “Infos” tab on top. By clicking on the specific info name under the “Quantitative” section the small panel opens up on the right which can be used to select the colormap (e.g Jet). Clicking the green ✓ button on this panel sets the colormap representing Segmentation Quality information on the 3D mesh. More information on displaying multiple meshes may be found in : https://morphonet.org/help_api

At first only a 3D mesh of the image is displayed. Next, to visualize Jaccard Index info for the pipelines, click on Info-> Click on Info name-> Set colormap. This superposes the VJI values as color-mapped information on the mesh. Multiple meshes and multiple information may be uploaded in this manner.

To upload and visualize information on Morphonet, users need to create an account on Morphonet.org by clicking the “Signup” option on the page or use the guest account provided with this paper. To use this guest account, users may login using : **username:** guest, **password:** guest2021.

To use the Morphonet platform for uploading new data, after logging in to Morphonet, the users may upload meshes for multiple time points and corresponding information to superpose on the meshes. For visualizing segmentation quality on a cell by cell basis, users need to first compute the Volume -averaged Jaccard Index metric using the Jaccard_Index.ipynb (may be found under Downloads section of the SegCompare repository), using as input a segmented image and corresponding ground truth segmentation. The output of the Jupyter notebook is a CSV file containing Volume averaged Jaccard Index measure for each cell. This CSV file along with the ground truth segmented image may be used with the 3D_visualization.ipynb notebook (also under Downloads/Notebooks in the SegCompare repository) to do a one step uploading of mesh and numerical information on Morphonet. After running this notebook, users can go to the Morphonet.org page, navigate to their dataset to visualize it. Sample videos demonstrating these operations maybe found in our figshare repository as described below under “Videos”.

Appendix 3

Data and model repositories

The training and test datasets used in this work are available online. The training dataset of shoot apical meristems may be found under: https://www.repository.cam.ac.uk/handle/1810/262530

The test datasets of floral meristems may be found at:
https://www.repository.cam.ac.uk/handle/1810/318119

A repository of materials generated as part of this study may be found at
LINK: https://figshare.com/projects/3D_segmentation_and_evaluation/101120

This repository (3D Segmentation and evaluation) contains the trained models for each of the pipelines, meshes for uploading to Morphonet and corresponding segmentation accuracy files in CSV format. The structure of this repository is as below:

Trained deep learning models: The models trained in the four deep learning pipelines are provided.Instructions for running them are in the Gitlab repository (Appendix 1).
Original stacks and segmented data: Segmented confocal stacks by each of the five pipelines are provided along with ground truth stacks for each. Users may test the segmentation evaluation methods using them. Details of using the evaluation function are in the Gitlab (Appendix 1).
Meshes for Morphonet: Example meshes that might be uploaded to Morphonet are included. Users may test the Morphonet visualisation using these and the cellwise VJI values (saved in CSV files). Procedure for the visualization is provided in Gitlab.
Accuracy results: Cellwise VJI values saved in CSV files are provided for each pipeline. These may be used for projection on Morphonet for 3D visualization of segmentation quality.
Videos: Videos (.mp4 format) showing examples on how to use the Morphonet based 3D visualization method on a sample test image, videos showing sample training and test data.

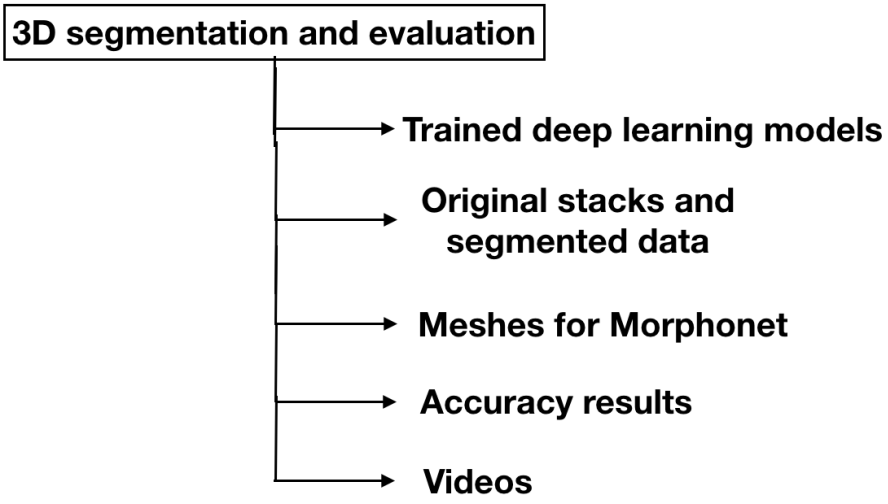

Figure A3.1 Contents of the data and model repository

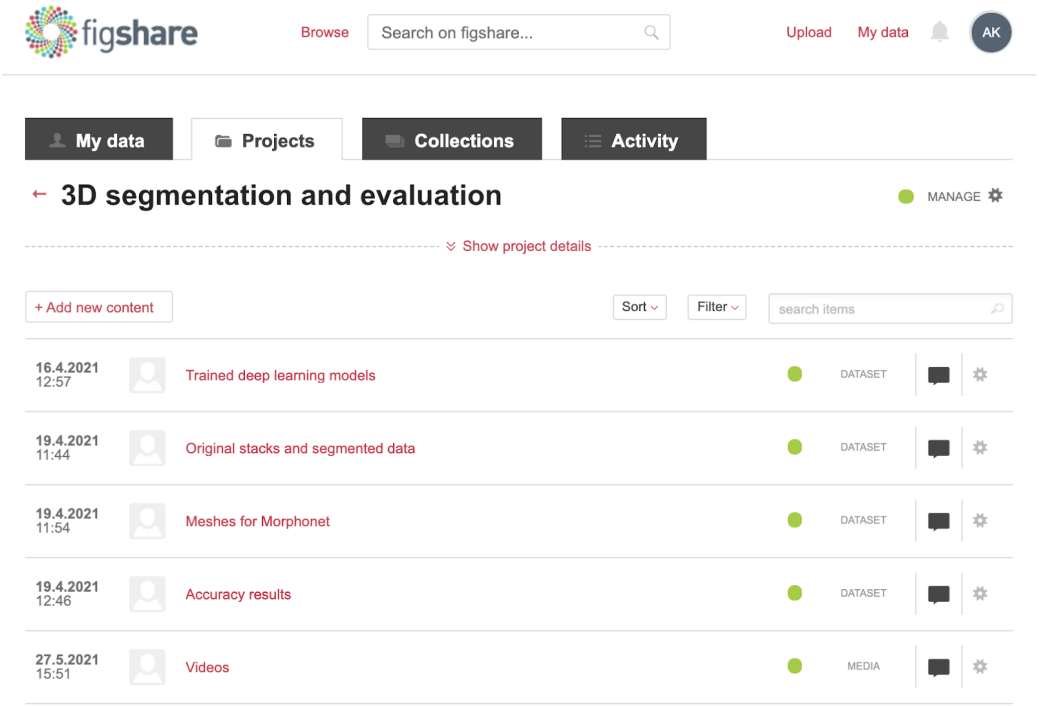

Figure A3.2 Organization of the 3D segmentation and evaluation repository on Figshare

Appendix 4 - Literature overview

Current research on deep learning based instance segmentation techniques

In complement to the survey given in the introduction, we provide here a more extensive overview of the existing deep learning based segmentation methods identifying the major trends of research in this rapidly evolving field. The focus of the survey is on methods that are developed for instance segmentation of images and papers for non-image datasets are excluded. We identified different categories of pipelines, which have been developed to address specific challenges. The research works belonging to each category are discussed below. The main purpose is to illustrate the existing diversity, rather than giving all the details of the individual methods, which is out of the scope of this article.

Pipelines for end to end 3D instance segmentation. As discussed in the introduction of the main text, end to end 3D (3D input, 3D output) segmentation pipelines have been implemented using either Unet, residual UNet or Mask RCNN architectures. For more details on the Unet and Residual UNet architectures see (Falk et al., 2019) (Zhu et al., 2020) and (He et al., 2017) for MaskRCNN. Besides the pipelines used here (Plantseg (Wolny et al., 2020), Unet_WS (Eschweiler et al., 2019) and Cellpose (Stringer et al., n.d.)), ((Jiang, Kao, Belteton, Szymanski, & Manjunath, 2019)) proposed a method which uses 3D images of A. Thaliana and time lapse images of leaf epidermal tissue for training a 3D Unet. This Unet extracts cell boundaries that are processed using 3D watershed along with conditional random fields (a prediction concept which uses contextual information from previous labels).

Deep learning algorithms for 2D instance segmentation: Mask RCNN is widely used for highly accurate 2D instance segmentation. **(Zaki et al., 2020)** tested two deep learning architectures for 2D nucleus segmentation i.e. a feature pyramid network (FPN) and a Mask RCNN. This study indicates that Mask RCNN gave superior results. (Shu et al., 2020) used a modification of the basic MRCNN to perform multi-organ segmentation of human esophageal cancer CT images and mitigate effects of fuzzy organ boundaries and diverse organ shapes in the images. The additional features in the proposed algorithm include a pre-background classification step to improve boundary predictions and use of a custom loss function. In (Liu et al., 2020) a modified Mask RCNN architecture termed as Panoptic Domain Adaptive Mask R-CNN is developed to achieve unsupervised segmentation of nuclei from histopathology images.

UNets are also used for 2D image segmentation. In (Al-Kofahi et al., 2018) a UNet based module followed by post processing steps of thresholding and watershed is used to predict locations of the cells and their nuclei. Multiple deep learning architectures based on UNet, modified UNet and Mask RCNN are tested for 2D nuclear image segmentation in (Kromp et al., 2019) and the Mask RCNN architecture was found to outperform the UNet based models in the 2D segmentation task. It may be noted that for evaluation (Zaki et al., 2020) use F1-score, while (Kromp et al., 2019) use under/oversegmentation and aggregated Jaccard index.

Deep learning based segmentation pipelines for specific purposes in bioimaging

a. Deep learning for cell segmentation and tracking: Deep learning pipelines for instance segmentation coupled with cell tracking have been proposed in several works, such as (Payer, Štern, Feiner, Bischof, & Urschler, 2019) where the pipeline makes predictions for every cell instance in videos, as well produces temporally connected instance segmentations. Cell instance segmentation in calcium imaging videos is described in (Kirschbaum, Bailoni, & Hamprecht, 2020) which uses temporal information to estimate pixel-wise correlation and shape information to identify cells and classify active and non-active cells. (Scherr, Löffler, Böhland, & Mikut, 2020) also proposes a modified Unet based model which can be used for tracking cells while dealing with challenging conditions such as crowded cell regions, poor image quality and on data with missing annotations. The method is tested on cell images from mouse muscle stem cells, HeLa cells, and images from developing embryos. Other approaches for implementation of cell segmentation and tracking include (C. Wang et al., 2019) and (Lugagne, Lin, & Dunlop, 2020). The latter uses two Unet models to create an architecture named DELTA to first segment the cells followed by tracking lineage reconstruction from time lapse videos of E. coli cells in fluidic medium.

b. Pipelines for addressing sparse annotations and small training datasets: For training of deep learning based segmentation models, annotated ground truth data is essential. However, expert annotation of biomedical images (especially 3D datasets) is a highly labour intensive and time consuming process. For this reason, several deep learning pipelines have been developed which can work with sparsely annotated data. These include the method described by (Zhao et al., 2018) which uses only a few fully annotated voxel instances to segment a full 3D stack. In (Guerrero-Peña, Marrero-Fernández, Tsang, & Cunha, 2019) it is demonstrated how varying the contrasts of cell boundaries and a new loss function (weighted cross entropy) could be useful to obtain high accuracy segmentations when a 3D Unet model trained with a small and sparsely annotated training dataset. The issue of sparse annotations is also addressed in works like (Dawoud, Hornauer, Carneiro, & Belagiannis, 2020) and (Arbelle & Raviv, 2018).

c. Pipelines for segmenting images with densely packed cells/tissues : Cell instance segmentation in images where cells appear in dense clusters or in overlapping manner is a common research problem. It is quite challenging as there are high chances of errors in separating each cell. Specially designed deep learning models for segmenting densely packed cell regions are reported in works like (Korfhage et al., 2020). It uses an object detection module called a feature pyramid network, which apparently outperforms MRCNN in this task. The feature pyramid network extracts information of the same image at different scales, in this case the cells and the subcellular nuclear scale. In (Linfeng Yang et al., 2020) a hybrid architecture combining UNet and MRCNN is proposed to address effects of crowded and variable sized objects, named Nuclei Segmentation Toolset or NuSeT for nuclei segmentation. The U-Net here is used for semantic level segmentation, the modified MRCNN predicts the instance bounding boxes based on the UNet outputs which are then finally used as seeds for watershed segmentation. (Vu et al., 2019) uses a combination of two CNNs, that provide a semantic segmentation of nuclear material, followed by a final instance level segmentation of the individual nuclei. The research in (Kumar et al., 2020) performs 3 class classifications on a human tissue image dataset to distinguish between cell boundaries, inside and outside of dense nuclei regions. A new CNN architecture named HoVer-Net is presented in (Graham et al., 2019) for instance level segmentation of nuclei from histological images where they appear overlapped with each other. This method

can further classify the type of nuclei, e.g.the type of cells-such as between tumour and lymphocyte cells from which the nuclei are obtained. Another method specially designed for dense cell clusters in images is (Lux & Matula, 2020) which uses two Unet based deep models to predict pixels belonging to the cell regions. Using this output, a final watershed based segmentation is implemented.

d. Pipelines for segmenting special cell shapes: In many biological datasets, cells or tissues could have morphologically complex, e.g. very thin or elongated shapes that are difficult to segment by generic pipelines built for regular spherical cell shapes. Deep learning pipelines, with specific DL architectures for segmenting this kind of data have been developed. This includes the method described by **(Yi et al., 2019)** which can perform precise segmentation of neural cells which have unconventional structures while also countering challenges like cell division and unclear cell boundaries. An approach combining object detection and segmentation is described in (Yi et al., 2018) which is successful in high precision detection of small scale and narrow structures of neural cells. Other works include (Lin Yang, Zhang, Guldner, Zhang, & Chen, 2016) for the segmentation of glial cells, a deep learning model named DeepEM3D-Net in (Zeng et al., 2017) for segmenting 3D neurite images and (Gu et al., 2019) for segmentation of diversely shaped human organs in images.

Conclusions.

From this literature survey, the following aspects are observed 1) UNet and Mask RCNN are two most common deep learning architectures that are currently used for instance segmentation of biological images 2) The number of works on end to end 3D deep learning for instance segmentation is much lower than that for 2D image datasets. 3) the existing segmentation pipelines have been trained on a wide variety of datasets (plant and animal tissue, cell and nuclei; 2D and 3D still images, videos, time-lapse images) and therefore it is not possible to determine their relative performance levels. 4) For many of the methods surveyed it is not possible to reproduce the pipeline as they are not open source or do not allow retraining. 5) There exists a shortage of large 3D annotated image datasets on plant and animal tissues which are publicly available.

Appendix 5

Cross dataset validation of the segmentation pipelines

We tested the performance of our trained deep learning pipelines and MARS on data different from confocal images of floral meristems, for example on data from other microscopes and tissue types to observe the adaptability of our methods to new and unseen data (the deep learning pipelines were trained on shoot apical meristem images). Images from two datasets are used for this and are described below.

a) Ascidian Phalusia mamaliata (PM) embryo images: The 5 pipelines are used for segmenting 3 images from the PM embryo dataset which is described in (Guignard et al., 2020). These are captured using multi-view Light-sheet (MuVi-SPIM) microscopes from fluorescently labelled cell membranes. Ground truth segmentations for these images are also provided in the dataset.

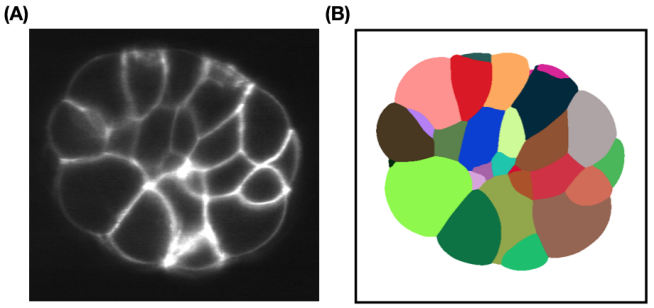

Figure A5.1 Slice view of a sample (A) Ascidian embryo image and its (B) ground truth segmentation

Three test images and their ground truths are taken from each of the above datasets and all five pipelines are used to segment this data. Then the volume averaged Jaccard Index metric is used to estimate the segmentation quality as done with the floral meristem test dataset and the results are in Table A5.1.

Sample name	Plantseg	Unet+WS	MARS*$_{PM}$	Cellpose	MRCNN+WS
PM 01	0.90	0.84	0.91	0.40	0.51
PM 02	0.83	0.82	0.87	0.34	0.66
PM 03	0.89	0.80	0.86	0.32	0.67

Table A5.1: VJI values for segmentation results using Ascidian PM data and 5 pipelines

It is seen that the deep learning pipelines Plantseg, Unet+WS provide high accuracy results on completely unseen data without requirement of re-training. MARS*$_{PM}$ indicates the MARS algorithm results after tuning the parameters for this dataset and it also provides high accuracy segmentations on this dataset. The MRCNN+WS results are similar to what we got previously from this pipeline on floral meristem data. The Cellpose accuracy however falls from their average values observed for floral meristem data.

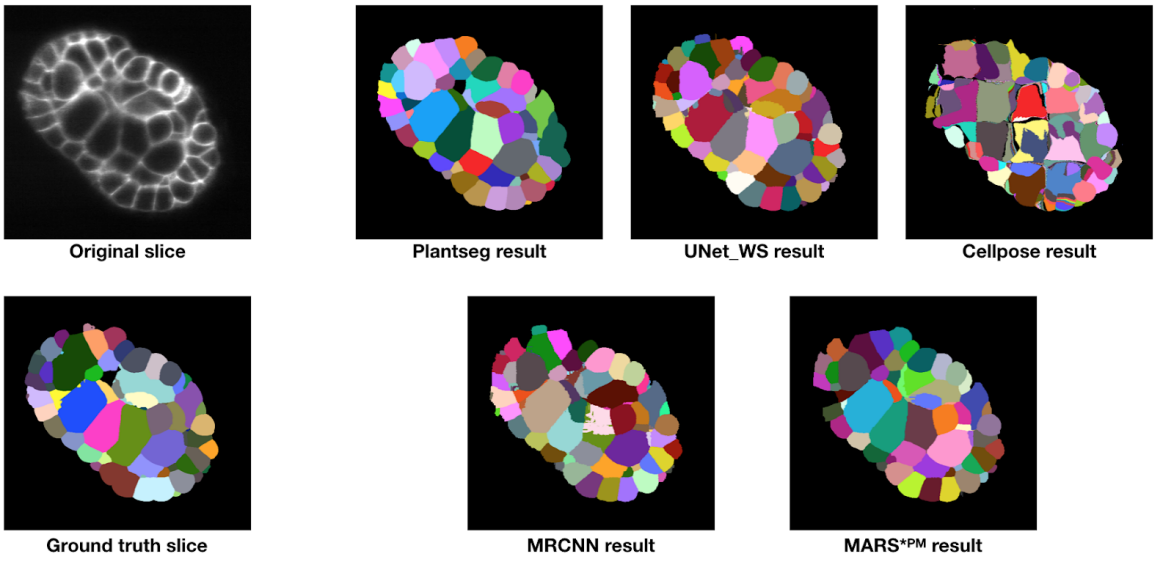

Figure A5.2 Ascidian embryo image (PM03), ground truth and segmentations by 5 pipelines

b) Arabidopsis ovule images: 3D confocal Images of Arabidopsis thaliana ovules at various developmental stages are provided by the authors of the Plantseg pipeline (Wolny et al., 2020). Three stacks from this dataset along with their ground truths are used for evaluating our 5 pipelines. Structurally these are quite different from Arabidopsis floral meristems and the segmentation results from the five pipelines along with the original and ground truth images are shown in Figure A5.3 below.

Sample name	Plantseg	Unet+WS	MARS*$_{ov}$	Cellpose	MRCNN+WS
Ov 01	0.82	0.80	0.85	0.68	0.64
Ov 02	0.87	0.89	0.91	0.50	0.62
Ov 03	0.77	0.78	0.83	0.69	0.55

Table A5.2: VJI values for segmentation results using ovule data and 5 pipelines

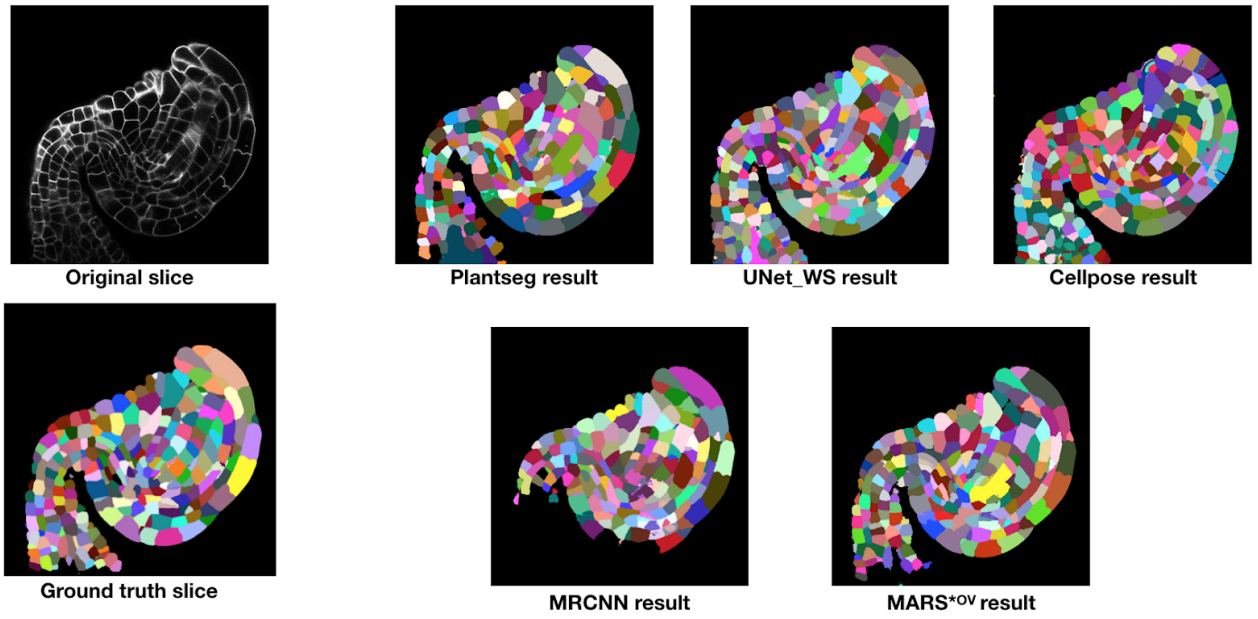

Figure A5.3 Ovule image (Ov 03), ground truth and segmentations by 5 pipelines

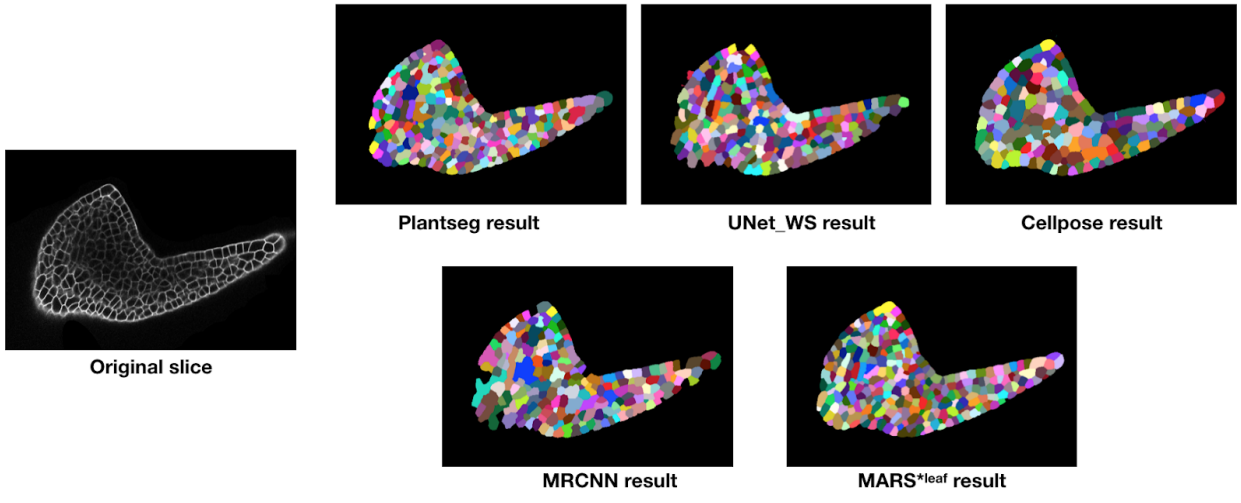

Figure A5.4 Leaf image and segmentations by 5 pipelines

c) Leaf images: A 3D confocal image of Arabidopsis thaliana leaf obtained from the dataset provided by (Wolny et al., 2020) is used for testing the 5 pipelines and the results are in Figure A5.3. This tissue shape is not present in the training data but still the segmentation results look satisfactory. The Plantseg repository does not provide a volume segmented ground truth for this dataset so VJI values could not be computed.