Máster Universitario en Análisis de Datos Ómicos
y Biología de Sistemas
Universidad de Sevilla
Universidad Internacional de Andalucía

Semi-automatic three-dimensional segmentation protocol of *Patiria miniata* epithelia during embryonic development

Miguel Ángel Spínola Tena[1], Jesús Ángel Andrés San Román, Antonio Jesús Tagua Jáñez, Pedro Gómez Gálvez[*1], Luis María Escudero Cuadrado*[1].

[1]Department of Cell Biology, University of Seville, 41013 Seville, Spain; migspiten@alum.us.es

*** Correspondence:**
[1]Pedro Gómez Gálvez
[1] pgomez-ibis@us.es

[1]Luis María Escudero Cuadrado
[1] lmescudero-ibis@us.es

Keywords: 3D image segmentation; Epithelial morphogenesis; Deep Learning.

Abstract

During embryonic development, epithelial cells grow, proliferate, rearrange and change shape to drive the initial events of morphogenesis. Therefore, the detailed analysis of epithelial tissues is especially interesting when studying the embryonic development of different animals. In this regard, image segmentation has gained prominence over the last few years in the attempt to elucidate the mechanisms that guide epithelial organization. In particular, the three-dimensional approaches which consider the epithelial cells' three-dimensional structure, as well as their contacting cells, are especially relevant. One of the main challenges when approaching a three-dimensional epithelia segmentation is to obtain an accurate model of the three-dimensional structure of such epithelia. Traditionally, epithelia segmentation methods entail segmenting each cell one by one and correcting the cellular contacts manually, which are both really time-consuming tasks. Nevertheless, over the past years the supervised Deep Learning algorithms have exceeded substantially the accuracy and the overall performance of traditional segmentation methods. In this work, we present an accurate and precise semi-automatic three-dimensional segmentation protocol for *Patiria miniata*'s (a sea star) embryos, based on the employment of a Convolutional Neural Network implemented within a reinforced Deep Learing model. The major biological relevance of our segmentation method lies in the fact that it enables to process live-imaging files automatically, significantly reducing the time required to process and segment these images. This represents a breakthrough and an advance that could be essential to understand the mechanisms that mediate growth, epithelial packing, and developmental processes, just like how they are coupled altogether.

1 Introduction

1.1 Epithelia structure, architecture, and three-dimensional packing

1.1.1 Epithelia

One of the major tissue types in animals and vertebrates is epithelial tissue, in which cells are tightly bounded together into sheets called epithelia (Alberts et al., 2002; Ganz, 2002) (**Fig. 1**). Epithelial cells have a structural polarity that produces the existence of three distinct domains (apical, basal, and lateral). The apical domain faces the lumen of an organ or the external environment surrounding it, whereas the basal domain interacts closely with the extracellular matrix (ECM) which underlies the epithelium and consists of a thin mesh called basal lamina (BL) (Gumbiner, 2005; Bays and DeMali, 2017) (**Fig. 1**). Lateral domain is where the structures responsible of keeping epithelial cells in close contact (tight and adherens junctions) are located (Coradini et al., 2011) (**Fig. 1**). These specialised junctions allow epithelia to form protective barriers preventing the loss of water and blocking the entrance of pathogens into the organism, among other functions (Duszyc et al., 2017). Epithelial sheets (**Fig. 2A**) can fold, bend, and reshape (**Fig. 2B**), as consequence of tissue morphogenesis (Heisenberg and Bellaïche, 2013).

1.1.2 Three-dimensional epithelial packing during development

Planar epithelial sheets (**Fig. 2A**) during morphogenesis undergo a reshaping process which consists in a severe reorganization of such plane epithelium, originating complex three-dimensional (3D) structures and curved epithelia (Wang et al., 2013) (**Fig. 2B**). This process is guided by cell-cell interactions, cell proliferation (Gibson et al., 2006; Farhadifar et al., 2007) and extrinsic mechanical forces (Heisenberg and Bellaïche, 2013). Interestingly, it is not a trivial process since cells will dispose themselves seeking energy minimization within the tissue (Gómez-Gálvez et al., 2018, 2021). It has been accepted that epithelial cells from a planar epithelial sheet are prism-shaped (**Fig. 2A**) acquiring frustrum-like shape when they bend (**Fig. 2B**). Then, most approaches to study epithelial development are 2D-based, mainly focused on their apical surface, due to it was thought apical and basal cell surfaces share topological properties. Nonetheless, it has been demonstrated that, under specific circumstances of curvature, epithelial cells adopt scutoid structure (Gómez-Gálvez et al., 2018) (**Fig. 2C**). Scutoid is a geometric solid between two parallel surfaces with at least a vertex along its apico-basal axis (**Fig. 2C**).

2D-based studies are unrealistic, which justify 3D approaches to understand epithelial development, which in turn is key for understanding embryogenesis (Lecuit and Lenne, 2007). On this matter, the characterization of the Scutoids was a milestone.

1.2 *Patiria miniata*

The organism selected to work with was *Patiria miniata* (**Fig. 3**), a species of sea star. These echinoids produce big transparent eggs which both significantly facilitates the analysis of their biological development and makes them ideal for microscopy (Nesbit et al., 2019). In general, all kind of echinoid embryos are characterized by their asymmetric cell division, originating a large cell (macromere) and a small cell (micromere). Conversely, these asymmetries are very subtle in *Patiria miniata* (Kominami, 1983; Barone et al., 2022). In addition, during the cleavage stage (**Fig. 4**) (when images are taken) there is an envelope around the embryos that protect and prevent them from moving, making manageable the image acquisition process. This cleavage stage is synchronous in

Patiria miniata, which means every cell divides simultaneously. As the embryo approaches the gastrulation stage (**Fig. 4**), this envelope breaks down leading to the so-called hatching stage. For this reason, we studied embryos at 256-cells (9 cell division cycles) and 512-cells (10 cell division cycles) stages. After 1024-cells stage hatching takes place and it would be extremely difficult to film the embryo (Byrne, 2011).

1.3 3D segmentation in epithelia

Image segmentation is a type of image processing (Uchida, 2013), a common technique applied in different fields, including developmental biology (Oliva et al., 2019). It is employed for identifying objects or regions of interest so they can be easily analysed (Roeder et al., 2012) (**Supplementary Fig. 1A-B**).

There are multiple approaches to obtain an accurate epithelial segmentation (Sridevi and Mala, 2012; Deshmukh and Mankar, 2014; Phonsa and Manu, 2019), including threshold-based (Saleh Al-amri and Kalyankar, 2010; Niu and Li, 2019) and mathematical morphology segmentation methods like Watershed algorithm (Lalitha et al., 2016) which is a simple and fast image segmentation method to detect, and split overlapped objects in an image (Grau et al., 2004; Chen and Zhang, 2010; Zheng et al., 2021) (**Supplementary Fig. 1C-D**). Performing an accurate 3D epithelium segmentation entails knowing how many cells are present within it, their contours and neighbours, and where they are precisely located. For this purpose, nuclei segmentation is a successfully implemented technique to get a readout of the nuclei information (Hollandi et al., 2020; Verma et al., 2021).

1.3.1 Deep Learning segmentation

Image segmentation techniques have evolved drastically thanks to the implementation of Deep Learning (DL) supervised algorithms employing Convolutional Neural Networks (CNN) (Sommer et al., 2011; Turkki et al., 2015) (**Supplementary Fig. 2**), using as inputs a training dataset composed by raw images coming from the microscope and its proper segmentation (ground truth). CNN's functioning is based on distinguishing the different objects within an image when input propagates forward and identifying such image's properties. Sometimes, the training dataset is limited because of the difficulty of image acquisition or annotation, hence, to increase the number of samples of the training dataset, data augmentation (DA) is usually applied. DA deals with transformation of the original training dataset through elastic transformations, rotation, or noise addition (Wang and Perez, 2017).

Different segmentation pipelines use these CNN to segment either 3D nuclei (StarDist (von Chamier et al., 2021)) or 3D cell instances (CellPose (Stringer et al., 2021), PlantSeg (Wolny et al., 2020)). Each pipeline is implemented with a particular architecture, being ResUNet (Zhang et al., 2018b) (**Supplementary Fig. 2**) one of the most widely employed in biology due to its great performance on semantic segmentation. Moreover, DL segmentation methods can be combined with morphology segmentation ones, like Watershed algorithm, to accomplish an accurate instance segmentation. The segmented instances could be then used to retrain such DL model to achieve better inferences, optimizing its capability to recognize independent objects and their boundaries. This is known as 'Reinforcement Learning' (RL) (Sutton and Barto, 2015) which is based on trying to enhance the model's decision-making to obtain a more accurate segmentation by training it with a new training dataset of higher quality.

1.3.2 Voronoi diagrams

Voronoi is a space-clustering and compartmentalization algorithm based on proximity. Voronoi diagrams are generated from a given set of seeds in space, originating the so called Voronoi cells, which are exclusive regions of the space formed by all points closer to a given seed than to other (Gómez-Gálvez et al., 2018) (**Fig. 5**). These diagrams have been applied as epithelial models, since it has been proven that they mirror epithelial cells' distribution (Honda, 1978), 3D packing and morphology (Gómez-Gálvez et al., 2018).

Here we present a semi-automatic methodology to generate a training dataset, employing 3D nuclei segmentation and Voronoi's algorithm, valid to train a CNN capable of accurately segmenting 3D epithelia images from *Patiria miniata* without manual proofreading.

2 Materials and Methods

In this project we followed a segmentation pipeline which will be presented in detail here in this section. Such pipeline is composed by different steps:

1. 14 nuclei and cells-contours stacks of images (**Fig. 6A-B**) from *20190806_pos2* collection of *Patiria miniata*'s embryo (14 different time points from a single embryo) were selected and pre-processed as described in section 2.1.

2. Afterwards, the previous pre-processed nuclei stacks of images (**Fig. 6A**) were 3D segmented (**Fig. 6C**, section 2.2), and from the cells-contours stacks of images (**Fig. 6B**), binary masks were manually drawn (**Fig. 6D**, section 2.3).

3. Segmented nuclei (**Fig. 6C**) and binary masks (**Fig. 6D**) stacks of images were pre-processed (section 2.1), being them the source to the obtaining of the Voronoi diagrams (**Fig. 6E**), as explained in section 2.4.

4. The previously obtained 14 Voronoi models (**Fig. 6E**) were posteriorly used as the "target" of our artificial training dataset, used to train the CNN of our DL model (section 2.5). In addition, 26 different cells-contours stacks of images and their corresponded accurate segmentation were provided to test the CNN performance. Then, we run the CNN inference over the 26 cell-contours images (**Fig. 6F**).

5. Using the previous 26 inferences (**Fig. 6F**) as input, we performed a Watershed segmentation at instance level (**Fig. 6G**) of each one of the 26 stacks of images (section 2.6).

6. The 26 segmentations (**Fig. 6G**) were employed as a new training dataset for such DL model as exposed in section 2.8 (reinforcement learning, RL). Our new DL model was used to infer over 66 new cells-contours stacks of images (**Fig. 6F**) from 3 different collections of embryos (*20210729_pos1*, *20200225_pos1* and *20190806_pos3*) that had not been previously introduced to our DL model.

7. Then a Watershed segmentation at instance level of each one of the 66 inferences generated in the previous step (**Fig. 6F**) was performed (section 2.6) (**Fig. 6G**).

8. At this point, the accuracy of the three segmentation methods employed (Voronoi, DL model and reinforced DL model) was measured following the protocol detailed in section 2.7, using our ground truth dataset to assess the performance of each model based on different parameters. Such ground truth dataset was composed by 35 manually labelled stacks of images (**Fig. 6H**) and their corresponded raw cells-contours stacks of images (**Fig. 6B**).

2.1 Images pre-processing

The stacks of images used in this work were obtained by confocal microscopy and provided by Deirdre Lyons and Vanessa Barone from the Scripps Institution of Oceanography at the University of California, San Diego (https://www.lyonslab.org). In this study we worked with stacks of embryos of *Patiria miniata* at different stages of development, mainly at 256 cells and 512 cells stages. Stained-nuclei stacks (**Fig. 3A**) and stained cells-contours stacks (**Fig. 3B**) of images were provided.

Before starting the image segmentation pipeline, stacks were pre-processed (**Fig. 6A-B**) using Fiji (Schindelin et al., 2012) (https://imagej.net/software/fiji/) to adjust their contrast and brightness. The dimensions of each one of these stacks were 1248x1248x70 which means each stack (three-dimensional image) is composed by 70 slices (two-dimensional images) of 1248x1248 each. The pixel width and height of these stacks were 0.1863522 micrometres, whereas their voxel depth was 1.1496555 micrometres. The depth of the stacks was too small compared with their height and width (1248x1248x70), hence nuclei (**Fig. 6A**), segmented nuclei (**Fig. 6C**), and binary masks (**Fig. 6D**) stacks of images were homogenized and reduced in order to expand their depth and make their resolution equal in X-axis, Y-axis, and Z-axis before proceeding with the Voronoi segmentation, acquiring new dimensions (375x375x130). Consequently, size, form, and other type of changes originated in the cells between slices of a stack could be noticed more easily.

```
%% Stacks Homogenization and Reduction

[originalImage, imgInfo] = readStackTif("stackImageName.tif");

xResolution = imgInfo(1).XResolution;
yResolution = imgInfo(1).YResolution;
spacingInfo = strsplit(imgInfo(1).ImageDescription, 'spacing=');
spacingInfo = strsplit(spacingInfo{2}, '\n');
z_pixel = str2num(spacingInfo{1});
x_pixel = 1/xResolution;
y_pixel = 1/yResolution;

shape = size(originalImage);
numRows = shape(1);
numCols = shape(2);
numSlices = round(shape(3)*(z_pixel/x_pixel));

originalImage = imresize3(originalImage, [numRows, numCols, numSlices]);

homoOriginalReduced = imresize3(originalImage, 0.3);

writeStackTif(homoOriginalReduced, "homogenized&ReducedImage.tif");
```

Previously to the training of the DL model, both the cells-contours stacks (**Fig. 6B**) and their corresponded Voronoi diagrams (**Fig. 6E**) which were used as training dataset were processed through a macros script from Fiji (described below) to normalize their saturation. This pre-processing was required to use these stacks as the training dataset for our DL model.

```
/* Macro template to normalize the saturation of multiple images in a folder:
 */

#@ File (label = "Input image directory", style = "directory") input
#@ File (label = "Output image directory", style = "directory") output
#@ String (label = "File suffix", value = ".tif") suffix

processFolder(input);

function processFolder(input) {
       list = getFileList(input);
       list = Array.sort(list);
       for (i = 0; i < list.length; i++) {
              if(File.isDirectory(input + File.separator + list[i]))
                     processFolder(input + File.separator + list[i]);
              if(endsWith(list[i], suffix))
                     processFile(input, output, list[i]);
       }
}

function processFile(input, output, file) {
       print("Processing: " + input + File.separator + file);
       open( input + File.separator + file );
       run("Enhance Contrast...", "saturated=0.3 process_all use");
       run("8-bit");
       run("8-bit");
       /*run("Size...", "width="+newWidth+" height="+newHeight+" depth="+slices+"
constrain average interpolation=Bilinear");
       */
       print("Saving to: " + output + File.separator + file);
       save(output + File.separator + file);
       close();
       /*selectWindow("Composite-1");
       close();*/
       /*open( inputDirLabels + File.separator + file );
       run("Size...", "width="+newWidth+" height="+newHeight+" depth="+slices+" constrain
interpolation=None");
       print("Saving to: " + outputDirLabels + File.separator + file);
       save(outputDirLabels + File.separator + file);
       close();*/
       //run("Make Binary");
}
```

2.2 3D nuclei segmentation

Using as source the stacks of images relative to the nuclei staining (**Fig. 6A**), a 3D nuclei segmentation algorithm was applied to them. Such nuclei segmentation was performed for every nucleus from each one of the 70 slices of the 14 stacks selected. In this regard, MATLAB software (https://es.mathworks.com/products/matlab.html) version R2022a was employed. The code used for this purpose is specified below and was based in a previously developed open source two-dimensions image segmentation tutorial: https://es.mathworks.com/matlabcentral/fileexchange/25157-image-segmentation-tutorial?s_tid=srchtitle.

```
%% 3D Nuclei Segmentation

function SeaStar3DNucleiSegmentation(inPath, imageName, outputPath, outputName, scale)

    if nargin < 4
        scale = 1;
    end
    originalImage = readStackTif(strcat(inPath, imageName));
    originalImage = imresize3(originalImage, "Scale", scale, "Method", "nearest");

    BW = imbinarize(originalImage, "global");
    binaryImage = imfill(BW, 'holes');

    binaryImage = bwareaopen(binaryImage, 30, 26);

    se = strel('sphere', 15);
    labeledImage = bwlabeln(binaryImage, 26);
    labeledImage2 = labeledImage;

    uniqueLables = unique(labeledImage);

    for cellIx = 2:length(uniqueLables)
        cellId = uniqueLables(cellIx);
        if size(unique(labeledImage2 == cellId), 1) == 2
            cellProps = regionprops3(labeledImage2 == cellId, "Volume", "Centroid");
            labeledImage2(labeledImage2 == cellId) = 0;
            if cellProps.Volume < 150
                continue
            end
            y = round(cellProps.Centroid(1));
            x = round(cellProps.Centroid(2));
            z = round(cellProps.Centroid(3));
            labeledImage2(x, y, z) = cellId;
        end
    end

    dilatedImage2 = imdilate(labeledImage2, se);
    writeStackTif(double(dilatedImage2/255), strcat(outputPath, outputName));

end
```

As a result of this second step of the pipeline, stacks with each nucleus individually segmented were obtained (**Fig. 6C**). These stacks were then saved since they will serve as seeds (each segmented nucleus from each resultant stack of images) for the Voronoi diagrams. Comparing original nuclei (**Fig. 6A**) and segmented nuclei (**Fig. 6C**) locations within the stack of images, we could confirm the segmentation was correctly performed.

2.3 Masks drawing of the corresponded cells-contours stacks

Once we obtained a proper nuclei segmentation (**Fig. 6C**), the next step was to draw the masks (**Fig. 6D**) for the 14 cells-contours stacks (**Fig. 6B**) for our training dataset. We later used these to delimit the plane in which cells are distributed. For this purpose, we used the Volume Segmenter application (https://es.mathworks.com/help/images/ref/volumesegmenter-app.html) which was already implemented within MATLAB software.

A mask is a binarization of the cells-contours stacks of images to distinguish cells from background. These masks were drawn manually over the already pre-processed cells-contours stacks (**Fig. 6B**). Even though each one of these stacks is composed of 70 slices, there were some empty slices (only background). We are interested in the plane occupied by cells in each slice thus it is why we only drew a mask (**Fig. 6D**) over the slices in which we could find any cell of interest. We focused only on central cells of the embryos since they provide information of the complete 3D structure of the cell. On the contrary, peripheral cells are not completely surrounded by neighbour cells, hence they are not as interesting for studying 3D packing in epithelia.

2.4 Voronoi algorithm

Subsequently, using as inputs the 3D nuclei segmentation (**Fig. 6C**) and as delimiting space the drawn cell mask (**Fig. 6D**), we modelled an artificial 3D epithelium using a 3D Voronoi diagram (**Fig. 7B**), using MATLAB software for this purpose. Following the protocol detailed in section 2.1, the input images were pre-processed to adjust their resolution (nuclei, segmented nuclei and binary masks stacks of images). After such pre-processing, we accomplished a homogenized and reduced (same resolution and dimensions as the input stacks of images) 3D Voronoi segmentation employing the code described below:

```
%% Homogenized and Reduced 3D Voronoi Segmentation

labelledImage = nuclei;
labelledImage2 = labelledImage;

uniqueLables = unique(labelledImage);

for cellIx = 2:length(uniqueLables)
    cellId = uniqueLables(cellIx);
    if size(unique(labelledImage2 == cellId), 1) == 2
        cellProps = regionprops3(labelledImage2 == cellId, "Volume", "Centroid");
        labelledImage2(labelledImage2 == cellId) = 0;

        y = round(cellProps.Centroid(1));
        x = round(cellProps.Centroid(2));
        z = round(cellProps.Centroid(3));
        labelledImage2(x, y, z) = cellId;
    end
end

se = strel('sphere', 10);
dilatedImage2 = imdilate(labelledImage2, se);

homoNucleiReduced = imresize3(dilatedImage2, 0.3, 'nearest');
homoMaskReduced = imresize3(mask, 0.3, 'nearest');

homoVoronoi = VoronoizateCells(homoMaskReduced, homoNucleiReduced); %%Voronoi segmentation
writeStackTif(homoVoronoi, "resultantHomogenized&ReducedVoronoi.tif");
```

The implemented function for the Voronoi segmentation (VoronoizateCells) was adapted from https://github.com/ComplexOrganizationOfLivingMatter/NaturalVariation/blob/main/Code/src/VoronoizateCells.m an already existing function previously developed by Pedro Gómez Gálvez for the same purpose on Cyst stacks of images. In **Fig. 7B** is depicted an example of a resultant 3D Voronoi diagram, accompanied by the 3D projection of its original cells-contours stack of image (**Fig. 7A**).

```
%% 3D Voronoi Segmentation

function [voronoiSeaStar] = VoronoizateCells(binaryMask, segNuclei)

    binaryMask = binaryMask/max(binaryMask(:));
    voronoiSeaStar = segNuclei.*cast(binaryMask, class(segNuclei));
    perimCells = bwperim(voronoiSeaStar > 0);

    idsToFill = find(binaryMask == 1 & segNuclei == 0);
    [row, col, z] = ind2sub(size(binaryMask), idsToFill);
    labelPerId = zeros(size(idsToFill));
    idsPerim = find(perimCells == 1);
    [rowPer, colPer, zPer] = ind2sub(size(binaryMask), idsPerim);
    labelsPerimIds = voronoiSeaStar(perimCells);

    if isempty(gcp('nocreate'))
        parfor nId = 1:length(idsToFill)
            distCoord = pdist2([col(nId), row(nId), z(nId)], [colPer, rowPer, zPer]);
            [~,idSeedMin] = min(distCoord);
            labelPerId(nId) = labelsPerimIds(idSeedMin);
        end
        poolobj = gcp('nocreate');
        delete(poolobj);
    else
        for nId = 1:length(idsToFill)
            distCoord = pdist2([col(nId), row(nId), z(nId)], [colPer, rowPer, zPer]);
            [~,idSeedMin] = min(distCoord);
            labelPerId(nId) = labelsPerimIds(idSeedMin);
            disp([num2str(nId) '/' num2str(length(idsToFill))])
        end
    end
    voronoiSeaStar(idsToFill) = labelPerId;
end
```

2.5 Deep Learning model – Training

Employing a ResUNet architecture (**Supplementary Fig. 2**), we trained a DL model using as training dataset the Voronoi segmentation accomplished before (**Fig. 6E**). Such training was accomplished using an Anaconda (https://www.anaconda.com) environment which was already implemented by Daniel Franco (https://github.com/danifranco/EM_Image_Segmentation/tree/master/utils/env).

Our training dataset was constituted only by 14 stacks in addition to DA during training. Some of the most important training parameters, including DA specifications, are detailed below:

```
AUGMENTOR:
        ENABLE: True
        VFLIP: True
        HFLIP: True
        ELASTIC: True
        BRIGHTNESS: True
        BRIGTHNESS_FACTOR: (0.1, 0.3)
        CONTRAST: True
MODEL:
        ARCHITECTURE: resunet
        FEATURE_MAPS: [16, 32, 48]
        DROPOUT_VALUES: [0.1, 0.1, 0.1]
TRAIN:
        ENABLE: True
        OPTIMIZER: ADAM
        LR: 1.E-5
        BATCH_SIZE: 5
        EPOCHS: 1300
        PATIENCE: 50
```

With regard to the training parameters detailed above, the augmentor was enabled to increase the number of samples of the training dataset. In this sense, the vertical flip (VFLIP), horizontal flip (HFLIP), elastic, brightness and contrast transformations were activated, which means that the data augmentation was performed by modifying the stacks of images from the original training dataset by flipping them vertically and horizontally, applying elastic distortion and changing the brightness and the contrast of such images. Besides, the brightness factor was employed to delimit the range of values in which the brightness values would oscillate.

In relation to the model specifications, the employed architecture was ResUNet (**Supplementary Fig. 2**), and the sizes of the feature maps was [16, 32, 48]. Feature maps are generated by applying filters or feature detectors to the input image (or the output of the previous layer) and the parameter "FEATURE_MAPS" allows us to customize their size. In addition, the dropout is a regularization method for preventing the DL model from overfitting (Srivastava et al., 2014) which consists in ignoring (or dropping out) some of the layer outputs and takes values between 0 and 1.

In relation with the training specifications, the selected optimizer was Adam, which is an efficient method for the optimization of complex problems (Kingma and Ba, 2015). Other essential parameters are the learning rate (LR), the batch size, the number of epochs and the patience of the CNN. LR determines the step size at each iteration while moving toward a minimum of a loss function. Sometimes, a small LR improves the training speed and accuracy (Wilson and Martinez, 2001). The batch size is defined as the number of samples (stacks of images) that will be propagated through the network at each iteration. In some cases, the higher the batch size the better the accuracy, although more time would be required to train the network (Radiuk, 2018). Considering that our training dataset was not comprised of a high number of samples, we stablished a batch size of 5. The number of epochs stablishes how many times the CNN from our DL model will work through the complete training dataset. A high number of epochs may improve the accuracy of the model, despite it could also lead to over training of the dataset and just the opposite result (Dhande and Shaikh, 2019; Afaq and Rao, 2020). Lastly, the patience sets the maximum number of epochs that the model must run in case that the loss will not get lower (which means the accuracy of the model is not improving). If the loss during training does not improve after 50 consecutive iterations, such training will early stop, which could save us a lot of time. Complete training parameters, specifications and DA details employed can be found in GitHub: (https://github.com/Spinola1014/3D-Epithelia-Segmentation).

As a result, we obtained the probability maps (**Fig. 6F**) of each one of the stacks included within the test dataset. Probability maps were applied as a measure of the reliability of these predictions (Joksic and Bajat, 2005). Once the model's training had finished, we got in return some metrics that enabled us to measure and analyze the training process to see if it was successful. On one hand, the **Jaccard Index** (Jaccard and Zurich, 1901), also known as the Jaccard similarity coefficient, of two sets is defined as the overlapping items in both sets divided by the total number of unique items, which ranges between 0 and 1, hence the bigger the Jaccard index the more the inferences produced by the model will resemble to the ground truth. On the other hand, the **Loss** is a metric employed to assess how a DL model fits the training set (training loss) and how well is the performance of this model using new data different from training set (validation loss), also called validation set. The loss function computes the distance between the current output of the algorithm from the model and the expected output, so it is a method to evaluate how the algorithm models the data (Boser et al., 1992), which means the lower the loss the smaller the difference between expected and current outputs and the better the performance of the model (it is calculated from a sum of the errors for each example in the dataset considered).

After its training, this DL model was used to infer the probability maps (**Fig. 6F**) of the 35 cells-contours stacks of images (**Fig. 6B**) corresponded to the ground truth labels (**Fig. 6H**). The reason why we used different images from the training dataset to test the performance of the model is because it enables us to know if the model is really learning, if it is overfitted and to understand its functioning.

2.6 Deep Learning model – Watershed segmentation

Probability maps (**Fig. 6F**) generated by the model were processed in order to obtain a segmentation at instance level. Out of the 26 inferences obtained, 5 of them were discard due to their low quality. In this regard, we applied a Watershed segmentation algorithm to these 19 remaining probability maps to try and obtain a better identification and separation of the independent cells and their boundaries. For this purpose, we used PlantSeg application (https://github.com/hci-unihd/plant-seg), which is a tool for cell instance aware segmentation in densely packed 3D images. It is a two-steps tool which includes a CNN predictions phase and a segmentation phase. Nonetheless, we only performed a Watershed segmentation because we already used a CNN for predictions. Without any pre-processing, we stablished the following parameters for the Watershed segmentation (enabling the option to convert the results to tiff):

- **Algorithm =** SimpleITK. *This segmentation algorithm is a simplified interface to the methods and data structures of the Insight Toolkit (ITK). This toolkit support more than 15 different image file formats, provides more than 280 filters for image analysis and implements a unified interface to the ITK intensity-based registration pipeline.* (*https://buildmedia.readthedocs.org/media/pdf/simpleitk/master/simpleitk.pdf*).

- **Under-/Over-segmentation fact =** 0.72 *This variable ranges from 0 to 1. High values bias the segmentation towards the over-segmentation. Since de cells contours are not very well defined in original stacks, it seems appropriate to use high values for this parameter.*

- **Run Watershed in 2D =** False. *Superpixels are created in the whole 3D volume, which make the segmentation process slower and more demanding, but can improve the segmentation accuracy.*

- **CNN Predictions Threshold** = 0.026 *Used for the superpixels extraction and Distance Transform Watershed. Low values indicate more seeds are placed resulting in over-segmentation.*

- **Watershed Seeds Sigma =** 4.0; **Watershed Boundary Sigma =** 0.0 *Useful to apply a Gaussian smoothing on the input before the segmentation operations. This is mainly helpful for seeds computation.*
- **Superpixels Minimum Size (voxels) =** 50 *Size filter to the initial superpixels over segmentation so segments smaller than the threshold will be merged with the nearest neighbor segment.*

- **Cell Minimum Size (voxels) =** 100 *Final size processing filter to merge cells smaller than the threshold with the nearest neighbor cell.*

2.7 Deep Learning model – Reinforcement Learning

Afterwards, the output of the Watershed segmentation (**Fig. 6G**) was used to build a new training dataset. In this case, the Watershed segmentation results (**Fig. 6G**) from de DL model inferences were used as training labels instead of the results from the Voronoi segmentation (reinforcement learning). This RL was performed using the same CNN and the parameters employed were also equal, including DA specifications in this step compared to the previous training. Sometimes, the Deep Reinforcement Learning process could lead to an overfitted model (Zhang et al., 2018a), hence, in order to be able to detect whereas our reinforced DL model was actually overfitted or not we tested its accuracy and performance using a large dataset (66 new cells-contours stacks of images (**Fig. 6B**)) to infer their 3D segmentation. Lastly, we employed such reinforced DL model to predict the segmentation of the 35 cells-contours stacks corresponded to the ground truth labels (**Fig. 6H**), just as the Voronoi segmentation model and the first trained DL model, and mathematically compared to the aforementioned labels (**Fig. 6H**) as described in section 2.8.

2.8 Validation assessment of the different 3D segmentation methods

In order to evaluate the quality of our method we assessed the segmentation accuracy at different points:

1. Voronoi training dataset generation.
2. Segmentation using the first trained DL model.
3. Segmentation after applying RL to our previous DL model.

These cell instances segmentations were measured based on the SegCompare metric, which in turn considered four different parameters (the number of cells, the correct segmentation rate, the over-segmentation rate and the under-segmentation rate):

- **SegCompare:** SegCompare is a tool for segmentation and segmentation-evaluation methods (Kar et al., 2022) which is based on the segmentation comparison method developed by Gäel Michelin (https://gitlab.inria.fr/mosaic/publications/seg_compare) (Michelin, 2016). This application enables us to assess different segmentation rates, which are a measure of segmentation quality. The most important of them are the following:
    - **Number of cells:** Number of objects detected in each stack of image.
    - **Correct segmentation rate:** It is defined as the proportion of cells that have been correctly segmented with respect to the ground truth segmentation.
    - **Over-segmentation rate:** The over-segmentation is known as the process by which the objects being segmented (cells) from the background are themselves segmented into subcomponents. This parameter assesses the proportion of over-segmented objects within each stack of image evaluated.
    - **Under-segmentation rate:** Conversely to the over-segmentation rate, the under-segmentation rate calculates the proportion of different objects within an image that have been considered as a single object during the segmentation process (two different cells that have been identified as a single cell by the algorithm of the model).

To assess the accuracy of each model employed, these parameters were calculated for each one of the 35 cells-contours stacks of images corresponded to the ground truth labels (**Fig. 6H**) (from *20200114_pos4* collection) and for each one of the three segmentation methods that were employed in this work: Voronoi segmentation, first DL model and reinforced DL model. These metrics are based in the degree of similarity between the ground truth image (**Supplementary Fig. 3A**) and the predicted segmentation (**Supplementary Fig. 3B**). In **Supplementary Fig. 3A-B** are depicted a ground truth image and its corresponded predicted segmentation performed by the reinforced DL model, respectively. Both type of images, the ground truth and the predicted segmentations, were mathematically compared to assess their degree of similarity (**Supplementary Fig. 3C**) and the accuracy of predicted objects (giving each object equal importance). In this regard, the more differences between the two images (**Supplementary Fig. 3C**), the lower the accuracy of the model in question.

The 35 cells-contours stacks of images from the ground truth dataset (**Fig. 6B**) were previously segmented by Pedro Gómez Gálvez (semi-automatic segmentation plus manual proof-reading) to obtain our ground truth dataset of labels (**Fig. 6H**). In order to evaluate the accuracy of the three models implemented in this work (Voronoi model, DL model and reinforced DL model), the 35 nuclei (**Fig. 6A**) and their corresponded 35 cells-contours stacks of images (**Fig. 6B**) were processed through the Voronoi methodology (steps 1 to 3 from the pipeline described at the beginning of Materials and Methods section), as well as the 35 cells-contours stacks of images (**Fig. 6B**) were introduced to both the DL model and the reinforced DL model to infer their segmentations.

3 Results and Discussions

In this work, we have defined a novel DL based segmentation pipeline, able to segment with accuracy the epithelium of *Patiria miniata* taking as source an artificial training generated after using a Voronoi model. To proof the quality of our method, we have evaluated out predictions at every step of the pipeline.

3.1 Assessment of the Voronoi based training dataset

The first step of our segmentation pipeline consisted in the generation of a 3D Voronoi model (**Fig. 7B**) of *Patiria miniata* epithelium (**Fig. 7A**) in order to be used as training dataset for DL segmentation (Materials and Methods). Obtaining an actual 3D epithelia completely segmented in order to be used for training dataset is highly time consuming. For this reason, we have designed a method to alleviate this critical step for any segmentation project, i.e., getting a proper training dataset. Then, as it is defined in the **Fig. 6A-E**, after automatically capturing the epithelial nuclei and drawing manually the binary mask, we executed the Voronoi algorithm, getting 3D simulations of epithelial cells. As a result, we obtained images (Voronoi diagrams) that mirror the epithelial cells' shape and connectivity (**Fig. 9B**). Nonetheless, as can be seen in **Fig. 9B**, the predicted cellular contours and boundaries are unrealistic (**Fig. 9A**), which means these diagrams are far from perfection.

Subsequently, we evaluated the segmentation quality of our artificial training dataset after comparing it with the ground truth (**Table 1**). According to these results (**Table 1**), the correct segmentation rate for this Voronoi model was over 85% (85.34%) and the under-segmentation rate was below 1% (0.52%), which were pretty satisfactory results. This support the idea that our training dataset is really acceptable. Conversely, the over-segmentation rate was significantly higher than desired: 13.64%. Overall, these results suggest that the Voronoi segmentation works positively. Nevertheless, there was margin for improvement in the 3D segmentation of these stacks.

Biologically speaking, the Voronoi segmentation method offers acceptable results that could be manually curated to improve them, enabling us to be able to undoubtedly identify each independent cell and extract its information (size, volume, centroid, morphology) in order to study in detail, the morphogenesis and embryonic development.

3.2 Assessment of the Deep Learning model

In pursuance of the aforementioned DL model, we had to create a training dataset which served us to teach the CNN how to perform a good 3D segmentation. As it was shown before, the Voronoi segmentation method provided positive results, thus we built a training dataset with them. The aim was to spend the shortest time possible, hence we selected 14 stacks of stained-nuclei stacks of images and their corresponded cells-contours stacks as well and performed a Voronoi segmentation just as like the one described in 'Materials and Methods' section. At this point we built the training dataset for the DL model training using both the homogenized and reduced cells-contours stacks and their corresponded Voronoi segmentation. One key aspect of this training was the DA. As our training dataset was not very large (only included 14 stacks of images), we needed to employ DA in order to enlarge the input size and variability within it. The main purpose of this technique is to guarantee that the model receives enough data, just as diverse enough, so it can learn at the same time we keep the model from overfitting. As mentioned before in section 2.5, the parameters used for this training and DA were uploaded to a GitHub repository (https://github.com/Spinola1014/3D-Epithelia-Segmentation).

In **Fig. 8A-B** are depicted the values of Jaccard index as well as the loss values obtained during the model training (along epochs) and validation phases. Training Jaccard index (**Fig. 8A**) got over 0.65 which is a correct result, however, validation Jaccard index (**Fig. 8A**) stabilized right under 0.60, hence overall these Jaccard coefficient results are modest. Regarding the training loss (**Fig. 8B**), it is highly remarkable that it got under 0.06 and stabilized around 0.07 during validation phase. Nonetheless, these values could be misleading and at the middle of the training (epoch 100 approximately) training loss and validation loss curves tend to separate (**Fig. 8B**). It does not look like there is a big gap between them and there is no reason to mistrust these results despite it is common that both curves separate too much when the model is too complex and overfitted. On the other hand, the fact that both curves seem to be stable instead of continuing to decrease indicates that the training has finished, or in other words, that the model did not require further training because results were not expected to improve. Altogether, these results suggest that the CNN of the model is learning indeed, and it could work with our current data (stacks of images).

After training our built DL model with the aforementioned dataset, we inferred the segmentation of 26 completely new cells-contours stacks of images (**Fig. 6B**) to analyse the output and the performance by the model itself. As a result, we acquired the corresponded probability map for membrane detection (**Fig. 6F**) of each one of the 26 cells-contours stacks of images introduced to the DL model (**Fig. 6B**). In regard to these probability maps, they were slightly blurred as can be seen in **Fig. 6F**. Such probability maps generated were then used as input to perform a 3D segmentation at instance level (**Fig. 6G**) employing PlantSeg software and the Watershed algorithm (as explained before in section 2.6), and then mathematically evaluated and compared to the ground truth labels (**Fig. 6H**) as described in section 2.8. In relation to the previous results obtained from Voronoi segmentation (**Table 1**), the average number of cells detected in each stack of image is practically the same as the average number of cells detected by the DL model (**Table 1**). Considering that we employed the prior Voronoi diagrams (**Fig. 6E**) to constitute the training dataset for our DL model, it makes sense that both the Voronoi model and the DL model identify approximately the same number of objects (cells) (**Table 1**). In terms of the correct segmentation rate of our DL model (74.03%) we can conclude that it is not a bad outcome, however, it was not enough to fulfil the purpose of this work. Although the under-segmentation rate did not exceed the 1% (0.93%), which is a positive outcome, the over-segmentation rate of our DL model was excessively high (25.04%, **Table 1**). This might be due to the fact that the CNN predicts non-trimmed stacks of images, but the ground truth labels stacks of images (**Fig. 6H**) were trimmed indeed, including only a specific group of cells. For this reason, when comparing the predictions from our DL model with the ground truth labels, the over-segmentation rate reaches such a high value (**Table 1**).

Despite these results seemed discouraging, our DL model was capable to perform an acceptable 3D segmentation semi-automatically and could be improved in order to achieve a higher accuracy.

3.3 Reinforcement Learning of the previous model and assessment

The goal of this last step of our protocol was to improve the accuracy of our previous DL model as much as possible in a fully automatic way. In this regard, we took the output of the previous DL model (the probability maps generated), processed it through the Watershed algorithm (as explained in section 2.6) and used the resultant segmentations (**Fig. 6G**) as a new training dataset to carry out a reinforcement learning on the prior DL model.

In **Supplementary Fig. 4** are depicted the probability maps generated by the final reinforced DL model for the images of the three different collections used to test it. Comparing these probability maps with the ones generated by the first DL model (**Fig. 6F**), the reinforced DL model's were more defined and not as blurry. Moreover, looking at the metrics of this retrained DL model (**Fig. 8C-D**) we can see that both the training and validation Jaccard indexes are above 0.6 (**Fig. 8C**), which is a similar result that the one we got after training our first model (**Fig. 8A**), although now it took 100 more epochs to train the CNN and curves look smoother. Regarding the loss functions, it is highly noticeable that both the training and the validation loss have the same shape and similar behaviour (**Fig. 8D**), getting both of them under 0.20 loss. These results are not excellent, but they are positive and satisfying since they may suggest that the CNN and the DL model were learning appropriately.

Subsequently, we proceeded to assess this retrained model performance by comparing its predictions with our ground truth labels. The results were compiled in **Table 1** and the first thing that we noticed was that the SegCompare results were remarkable considering that this retrained model achieved the highest correct segmentation rate (93.02%) and the lowest over-segmentation rate (6.32%) of all three models (Voronoi segmentation method, the first DL model and this reinforced model). It is worth to mention that this reinforced DL model exhibited a nearly 20% higher correct segmentation rate than the prior DL model. In addition, this reinforced DL model achieved a substantially lower over-segmentation rate (6.32%, **Table 1**), as well as an actually lower under-segmentation rate (0.66%, **Table 1**), in comparison with the previous DL model. These results in general represented an outstanding improvement of such DL model. Regarding the average number of cells identified by this reinforced DL model, it is slightly lower (18.03) than the number of cells recognised by the DL model (**Table 1**). This event can be explained by the low over-segmentation rate achieved by this reinforced DL model (6.32%, **Table 1**). Since the DL model showed an elevated over-segmentation rate (25.04%, **Table 1**), it was considering a larger number of cells than were actually present in the stacks of images.

Depicted in **Fig. 9D** are three examples of the 3D segmentations executed by this reinforced DL model on three different cells-contours stacks of images (**Fig. 9A**). Unlike the Voronoi diagrams (**Fig. 9B**), both the DL model's resultant segmentation (**Fig. 9C**) and the reinforced DL model's resultant segmentation (**Fig. 9D**) are capable of imitate the curvature of the cells' boundaries, which is really helpful and useful when pursuing a good 3D epithelia segmentation. When visually compared the three reinforced DL model's segmentations (**Fig. 9D**) with the corresponded segmentations obtained from the first DL model's prediction (**Fig. 9C**), their corresponded Voronoi diagrams (**Fig. 9B**) and their corresponded ground truth labels (**Fig. 9D**), it becomes apparent that the reinforced DL model performs more accurate 3D segmentations. Voronoi diagrams (**Fig. 9B**) respect the spaces and holes between cells due to the fact that these diagrams are based on manually drawn masks (**Fig. 6D**) which surfaces are identical to the ground truth labels'. Moreover, in spite of the reinforced DL model's predictions (**Fig. 9D**) seem to unrecognize some cells which are present in the ground truth (**Fig. 9E**), they do appear within the next slices of the stack indeed. As mentioned before, this model is not perfect and still requires some human supervision and correction, which is deduced from the comparison between **Fig. 9D** and **Fig. 9E**. However, it performs more accurate segmentations than the prior DL model (**Fig. 9C**) and even considering the need for supervision and corrections, such retrained DL model represents a great advance since it could save a lot of time and a lot of effort to the current researchers who are trying to decipher those mechanisms that guide morphogenesis and embryonic development.

In summary, this reinforced DL model is characterized by working fully automatically, hence although it is not perfect and still makes some mistakes during the segmentation, the results that it provided were truly promising and extraordinary. Besides, in spite of metrics and mathematical evaluation provide some insightful information, they do not reveal the whole reality that lays behind and they do not completely reflect the degree of improvement accomplished with this reinforced DL model. In this regard, **Fig. 9** is of great relevance to acknowledge and highlight the results obtained. Employing the Voronoi algorithm we managed to develop a semi-automatic protocol to obtain a training dataset of enough quality and accuracy (**Fig. 9B**) to train a CNN and, eventually, obtain a DL model capable of performing a precise 3D segmentation (**Fig. 9D**) on a fully automatic manner, just introducing the original stained cells-contours stacks of images (**Fig. 6B**) to the network.

4 Conclusions

When applying an image segmentation method, segmenting each cell from each stack one by one and correcting the cellular contacts manually are common practices, which are both time-consuming. Here we are proposing a novel segmentation protocol able to segment with high accuracy *Patiria miniata* epithelia, combining a Voronoi computational model and Deep Learning. This pipeline validates the use of an artificial training dataset to train a CNN in order to segment actual epithelia. This approach could be used to save time in the critical point of 3D segmentation, the creation of a new training dataset.

Based on the results compiled in **Table 1** it is evident that the training dataset built in the first place applying the Voronoi algorithm was considerably good, specifically according to the correct segmentation rate, justifying the development of an automatic DL model and the use of these Voronoi diagrams (**Fig. 9B**) as training dataset. Despite the segmentation results obtained with this DL model were not exceptional (**Fig. 9C**, **Table 1**), the pipeline was completely automatic, which entails a great advantage compared to the Voronoi algorithm. Nonetheless, it has been shown that there was a gap of opportunity to improve (**Table 1**), hence we decided to retrain this previous DL model (RL). Finally, we were able to enhance the segmentation quality of the output from the DL model and use it as a new training dataset. As a result, we present a reinforced DL model capable of performing a 3D epithelia segmentation from *Patiria miniata*'s embryos with an adequate accuracy and a noticeable correct segmentation rate while keeping a reasonably low over-segmentation and under-segmentation rates (**Table 1**). On one hand, some of the mathematical assessment performed over this reinforced DL model, as well as the generated learning metrics (**Fig. 8C-D**), and the resultant segmentations (**Fig. 9D**), support this work and the usage of this model for future 3D epithelia segmentation inferences over *Patiria miniata*'s embryos. On the other hand, it would be necessary to repeat the validation assessment of the three different models employed in this work, using non-trimmed ground truth labels (**Fig. 9E**) or previously excluding all the cells that we assume are irrelevant to consider.

In summary, our segmentation method paves the way to automatically process live-imaging files of the whole development of *Patiria miniata*'s embryos. This advance can be essential to analyse in detail how growth, epithelial packing and developmental processes are coupled.

5 Conflict of Interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

6 Author Contributions

L.M.E.C. and P.G.G. designed the study, A.J.T.J. provided all the collections of stacks of images used in this work, M.A.S.T. processed the images and implemented the code employed, with help from J.A.A.S.R. and P.G.G., and J.A.A.S.R. performed the training of the models and obtained the inferences made by them, as well as the metrics later used to assess their functioning. M.A.S.T. wrote the manuscript and interpreted and discussed the results with the assistance from J.A.A.S.R., L.M.E.C. and P.G.G.

7 Funding

This work is funded by the Spanish Ministry of Science and Innovation through grant PID2019-103900GB-I00.

8 Acknowledgments

We are thankful to Dr. Deirdre Lyons and Dr. Vanessa Barone for all the stacks of images from *Patiria miniata* provided for this work.

9 Data and Code Availability Statement

All the code employed for the different steps of the protocol described in this work can be found in GitHub (https://github.com/Spinola1014/3D-Epithelia-Segmentation). In this repository, a README document has been placed in order to explain carefully and clearly the whole content of the repository.

10 References

Afaq, S., and Rao, S. (2020). Significance Of Epochs On Training A Neural Network. *International Journal of Scientific & Technology Research* 9, 485–488. Available at: www.ijstr.org.

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. (2002). *The Molecular Biology of the Cell, 4th edition*. 4th ed. New York: Garland Science Available at: http://www.geocities.com/zrnet76/.

Barone, V., Byrne, M., and Lyons, D. C. (2022). Cell size asymmetries in the sea star embryo Running title. doi: 10.1101/2022.03.05.483134.

Bays, J. L., and DeMali, K. A. (2017). Vinculin in cell–cell and cell–matrix adhesions. *Cellular and Molecular Life Sciences* 74, 2999–3009. doi: 10.1007/s00018-017-2511-3.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A Training Algorithm for Optimal Margin Classifiers. *Association for Computing Machinery Press*, 144–152.

Byrne, M. (2011). Impact of ocean warming and ocean acidification on marine invertebrate life history stages: Vulnerabilities and potential for persistence in a changing ocean. *Oceanography and Marine Biology: An Annual Review* 49, 1–42. Available at: https://www.researchgate.net/publication/228949747.

Chen, W. bin, and Zhang, X. (2010). A new watershed algorithm for cellular image segmentation based on mathematical morphology. in *2010 International Conference on Machine Vision and Human-Machine Interface, MVHI 2010*, 653–656. doi: 10.1109/MVHI.2010.55.

Coradini, D., Casarsa, C., and Oriana, S. (2011). Epithelial cell polarity and tumorigenesis: New perspectives for cancer detection and treatment. *Acta Pharmacol Sin* 32, 552–564. doi: 10.1038/aps.2011.20.

Deshmukh, B. S., and Mankar, V. H. (2014). Segmentation of microscopic images: A survey. in *Proceedings - International Conference on Electronic Systems, Signal Processing, and Computing Technologies, ICESC 2014*, 362–364. doi: 10.1109/ICESC.2014.68.

Dhande, G., and Shaikh, Z. (2019). Analysis of Epochs in Environment based Neural Networks Speech Recognition System. in *Proceedings of the International Conference on Trends in Electronics and Informatics (ICOEI 2019)*, 605–608.

Duszyc, K., Gomez, G. A., Schroder, K., Sweet, M. J., and Yap, A. S. (2017). In life there is death: How epithelial tissue barriers are preserved despite the challenge of apoptosis. *Tissue Barriers* 5. doi: 10.1080/21688370.2017.1345353.

Farhadifar, R., Röper, J. C., Aigouy, B., Eaton, S., and Jülicher, F. (2007). The Influence of Cell Mechanics, Cell-Cell Interactions, and Proliferation on Epithelial Packing. *Current Biology* 17, 2095–2104. doi: 10.1016/j.cub.2007.11.049.

Ganz, T. (2002). Epithelia: Not just physical barriers. *Proceedings of the National Academy of Sciences of the USA* 99, 3357–3358. Available at: www.pnas.orgcgidoi10.1073pnas.072073199.

Gibson, M. C., Patel, A. B., Nagpal, R., and Perrimon, N. (2006). The emergence of geometric order in proliferating metazoan epithelia. *Nature* 442, 1038–1041. doi: 10.1038/nature05014.

Gómez-Gálvez, P., Vicente-Munuera, P., Anbari, S., Buceta, J., and Escudero, L. M. (2021). The complex three-dimensional organization of epithelial tissues. *Development (Cambridge)* 148. doi: 10.1242/dev.195669.

Gómez-Gálvez, P., Vicente-Munuera, P., Tagua, A., Forja, C., Castro, A. M., Letrán, M., et al. (2018). Scutoids are a geometrical solution to three-dimensional packing of epithelia. *Nat Commun* 9. doi: 10.1038/s41467-018-05376-1.

Grau, V., Mewes, A. U. J., Alcañiz, M., Kikinis, R., and Warfield, S. K. (2004). Improved watershed transform for medical image segmentation using prior information. *IEEE Trans Med Imaging* 23, 447–458. doi: 10.1109/TMI.2004.824224.

Gumbiner, B. M. (2005). Regulation of Cadherin-Mediated Adhesion in Morphogenesis. *Nature Reviews* 6, 622–634. doi: 10.1038/nrm1699.

Heisenberg, C. P., and Bellaïche, Y. (2013). Forces in tissue morphogenesis and patterning. *Cell* 153, 948. doi: 10.1016/j.cell.2013.05.008.

Hollandi, R., Szkalisity, A., Toth, T., Tasnadi, E., Molnar, C., Mathe, B., et al. (2020). nucleAIzer: A Parameter-free Deep Learning Framework for Nucleus Segmentation Using Image Style Transfer. *Cell Syst* 10, 453-458.e6. doi: 10.1016/j.cels.2020.04.003.

Honda, H. (1978). Description of Cellular Patterns by Diricblet Domains : The Two-Dimensional Case.

Jaccard, P., and Zurich, E. (1901). Étude Comparative de la Distribution Florale dans une Portion des Alpes et du Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles* 142, 547–579. doi: 10.5169/seals-266450.

Joksic, D., and Bajat, B. (2005). Probability maps as a measure of reliability for intervisibility analysis. *Spatium* 12, 22–27.

Kar, A., Petit, M., Refahi, Y., Cerutti, G., Godin, C., and Traas, J. (2022). Benchmarking of deep learning algorithms for 3D instance segmentation of confocal image datasets. *PLoS Comput Biol* 18, e1009879. doi: 10.1371/journal.pcbi.1009879.

Kingma, D. P., and Ba, J. (2015). Adam: A Method for Stochastic Optimization. in *International Conference on Learning Representations* Available at: http://arxiv.org/abs/1412.6980.

Kominami, T. (1983). Establishment of embryonic axes in larvae of the starfish, Asterina pectinifera. *J Embryol Exp Morphol* 75, 87–100.

Lalitha, K., Amrutha, R., Stafford, M., and Shivakumar, M. (2016). Implementation of Watershed Segmentation. *IJARCCE* 5, 196–199. doi: 10.17148/ijarcce.2016.51243.

Lecuit, T., and Lenne, P. F. (2007). Cell surface mechanics and the control of cell shape, tissue patterns and morphogenesis. *Nat Rev Mol Cell Biol* 8, 633–644. doi: 10.1038/nrm2222.

Michelin, G. (2016). Outils d'analyse d'images et recalage d'individus pour l'étude de la morphogenèse animale et végétale. Available at: https://tel.archives-ouvertes.fr/tel-01451608.

Nesbit, K. T., Fleming, T., Batzel, G., Pouv, A., Rosenblatt, H. D., Pace, D. A., et al. (2019). "The painted sea urchin, Lytechinus pictus, as a genetically-enabled developmental model," in *Methods in Cell Biology* (Academic Press Inc.), 105–123. doi: 10.1016/bs.mcb.2018.11.010.

Niu, Z., and Li, H. (2019). Research and analysis of threshold segmentation algorithms in image processing. in *Journal of Physics: Conference Series* (Institute of Physics Publishing). doi: 10.1088/1742-6596/1237/2/022122.

Oliva, D., Elaziz, M. A., and Hinojosa, S. (2019). *Studies in Computational Intelligence. Metaheuristic Algorithms for Image Segmentation: Theory and Applications*. , ed. J. Kacprzyk Cham, Switzerland: Springer Nature Switzerland AG Available at: http://www.springer.com/series/7092.

Phonsa, G., and Manu, K. (2019). A survey: Image segmentation techniques. in *Advances in Intelligent Systems and Computing* (Springer Verlag), 1123–1140. doi: 10.1007/978-981-13-0761-4_105.

Radiuk, P. M. (2018). Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets. *Information Technology and Management Science* 20. doi: 10.1515/itms-2017-0003.

Roeder, A. H. K., Cunha, A., Burl, M. C., and Meyerowitz, E. M. (2012). A computational image analysis glossary for biologists. *Development (Cambridge)* 139, 3071–3080. doi: 10.1242/dev.076414.

Saleh Al-amri, S., and Kalyankar, N. (2010). Image Segmentation by Using Threshold Techniques. *J Comput* 2, 83–86.

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., et al. (2012). Fiji: An open-source platform for biological-image analysis. *Nat Methods* 9, 676–682. doi: 10.1038/nmeth.2019.

Sommer, C., Straehle, C., Kothe, U., and Hamprecht, F. A. (2011). Ilastik: Interactive learning and segmentation toolkit. in *Proceedings - International Symposium on Biomedical Imaging*, 230–233. doi: 10.1109/ISBI.2011.5872394.

Sridevi, M., and Mala, C. (2012). A Survey on Monochrome Image Segmentation Methods. *Procedia Technology* 6, 548–555. doi: 10.1016/j.protcy.2012.10.066.

Srivastava, N., Hinton, G., Krizhevsky, A., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

Stringer, C., Wang, T., Michaelos, M., and Pachitariu, M. (2021). Cellpose: a generalist algorithm for cellular segmentation. *Nat Methods* 18, 100–106. doi: 10.1038/s41592-020-01018-x.

Sutton, R. S., and Barto, A. G. (2015). Reinforcement Learning: An Introduction Second edition, in progress. *The MIT Press*.

Turkki, R., Linder, N., Holopainen, T., Wang, Y., Grote, A., Lundin, M., et al. (2015). Assessment of tumour viability in human lung cancer xenografts with texture-based image analysis. *J Clin Pathol* 68, 614–621. doi: 10.1136/jclinpath-2015-202888.

Uchida, S. (2013). Image processing and recognition for biological images. *Dev Growth Differ* 55, 523–549. doi: 10.1111/dgd.12054.

Verma, R., Kumar, N., Patil, A., Kurian, N. C., Rane, S., Graham, S., et al. (2021). MoNuSAC2020: A Multi-Organ Nuclei Segmentation and Classification Challenge. *IEEE Trans Med Imaging* 40, 3413–3423. doi: 10.1109/TMI.2021.3085712.

von Chamier, L., Laine, R. F., Jukkala, J., Spahn, C., Krentzel, D., Nehme, E., et al. (2021). Democratising deep learning for microscopy with ZeroCostDL4Mic. *Nat Commun* 12. doi: 10.1038/s41467-021-22518-0.

Wang, J., and Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning.

Wang, Y. C., Khan, Z., and Wieschaus, E. F. (2013). Distinct Rap1 Activity States Control the Extent of Epithelial Invagination via α-Catenin. *Dev Cell* 25, 299–309. doi: 10.1016/j.devcel.2013.04.002.

Wilson, D. R., and Martinez, T. R. (2001). The need for small learning rates on large problems. in *Proceedings of the International Joint Conference on Neural Networks*, 115–119. doi: 10.1109/ijcnn.2001.939002.

Wolny, A., Cerrone, L., Vijayan, A., Tofanelli, R., Barro, A. V., Louveaux, M., et al. (2020). Accurate and versatile 3D segmentation of plant tissues at cellular resolution. *Elife* 9, 1–34. doi: 10.7554/eLife.57613.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. (2018a). A Study on Overfitting in Deep Reinforcement Learning. Available at: http://arxiv.org/abs/1804.06893.

Zhang, Z., Liu, Q., and Wang, Y. (2018b). Road Extraction by Deep Residual U-Net. *IEEE Geoscience and Remote Sensing Letters* 15, 749–753. doi: 10.1109/LGRS.2018.2802944.

Zheng, T., Duan, Z., Wang, J., Lu, G., Li, S., Yu, Z., et al. (2021). Research on Distance Transform and Neural Network Lidar Information Sampling Classification-Based Semantic Segmentation of 2D Indoor Room Maps. *Sensors* 21. doi: 10.3390/s21041365.

Figures Legends

Fig. 1. Epithelium structure and epithelial cells organization. Epithelial cells are distributed forming layers (sheets) in which every cell is physically coupled to one another through specialized cell-cell adhesion structures: tight junctions (TJ) and adherens junctions (AJ). These are polarized cells, and both the TJ and AJ structures are located on their lateral surface. In addition, they possess an apical surface, which is in contact with the lumen of an organ or the external environment, and a basal surface as well, being this basal pole where epithelial cells establish their adhesion with the basal lamina (BL) of the extracellular matrix. This structure (BL) support epithelial cells and helps them attach to and communicate with nearby cells and the connective tissue below the epithelium (**Coradini et al., 2011**).

Fig. 2. Representation of epithelia and epithelial cells. (**A**) Schematic representation of a planar monolayer epithelial sheet composed by prism-shaped epithelial cells. (**B**) Schematic representation of a curved monolayer epithelia due to an invagination or folding event. In this case, cells reshape acquiring new three-dimensional structure (frustrum). (**C**) Modelling figures of two scutoids which are participating in a transition and depiction of a scutoid structure, characterized by at least a vertex in its apico-basal axis (**Gómez-Gálvez et al., 2018**).

Fig. 3. Examples of the different stacks of images employed in this study. (**A**) Confocal-microscopy slice from a stack of images of fluorescently stained nuclei (H2B-RFP). (**B**) Confocal-microscopy slice from a stack of images of fluorescently stained cellular membranes or contours (ras-GFP). The scale of these images is specified in their corresponded low-right corner.

Fig. 4. Schematic representation of the development stages of *Patiriella regularis*. *Patiria miniata* development stages are represented in this figure through the development cycle of *Patiriella regularis*. Both organisms show similar embryonic development (**Byrne, 2011**).

Fig. 5. Schematic representation of Voronoi diagrams. (**A**) Two-dimnesional Voronoi diagram. Starting from an accurate two-dimensional nuclei segmentation and space delimitation, a two-dimensional Voronoi diagram can be obtained. (**B**) Three-dimensional Voronoi diagram from a previous three-dimensional nuclei segmentation. Voronoi diagrams are a partition of a plane into regions close to each of a given set of objects or seeds (nuclei), which means the space occupied is divided into different instances in such a way that the seed which identifies each one of these instances represents the closest seed for every spatial point located within such instance.

Fig. 6. Schematic pipeline for the elaboration of the training dataset, inference and instance segmentation. Voronoi segmentation was employed to build a training dataset for the convolutional neural network of our Deep Learning model. (**A**) Original nuclei stacks of images and (**B**) cells-contours stacks were preprocessed to adjust their contrast and brightness. Once processed, original nuclei images were three-dimensionally segmented (**C**) to obtain their centroids as seeds for posterior Voronoi diagrams. From processed cells-contours stacks we manually drew binary masks (**D**) to delimit the plane in which are located the cells we are interested on. Finally, combining seeds and masks, we obtained the corresponded (**E**) Voronoi diagrams. These Voronoi diagrams were employed for training de DL model, which then was used to infer (**F**) the probability maps of a set of stacks of images. Afterwards, (**G**) a Watershed segmentation was performed generating a 3D epithelia segmentation at instance level. Lastly, accuracy and precision of acquired results were assessed by comparing them to (**H**) the ground truth labels based on different metrics.

Fig 7. Schematic representation of a three-dimensional Voronoi diagram. (**A**) Three-dimensional projection of an original cells-contours stack of images and (**B**) its corresponded three-dimensional Voronoi diagram obtained.

Fig 8. Jaccard Index and Loss values of the Deep Learning model and the reinforced Deep Learning model. As a result of the training of our Deep Learning model we obtained its (**A**) training and validation Jaccard indexes, as well as its (**B**) training and validation Loss curves. These metrics allows as to measure and analyze the learning process of the convolutional neural network implemented within our Deep Learning model. After the reinforcement learning process, we obtained (**C**) the training and validation Jaccard indexes of such reinforced Deep Learning model and (**D**) its training and validation Loss curves.

Fig 9. Comparison between the Voronoi diagrams, Deep Learning model's segmentation outputs, reinforced Deep Learning model's segmentation outputs and the ground truth labels of three different given original images. Depiction of a (**A**) specific slice of three different cells-contours stacks of images from the ground truth dataset, (**B**) the exact same slice of its corresponded Voronoi diagram, (**C**) its corresponded three-dimensional segmentation inference made by the Deep Learning model and (**D**) the corresponded segmentation inference made by the reinforced Deep Learning model. (**E**) The expected resultant segmentation (ground truth labels) is also represented.

Supplementary Fig 1. Schematic representation of the image segmentation process and the Watershed segmentation algorithm. (**A**) Original gray-scale image and (**B**) the resultant segmented image after applying a threshold-based segmentation to extract details of interest from the original image (to differentiate the coins from the background). (**C**) A gray-scale image is conceived as a topographic relief where the height of each pixel is related to its gray level. Considering rain falling, the watersheds would be the lines that delimit each pond formed, represented by labels 1 to 3 (**Zheng et al., 2021**). (**D**) The watershed transform enables to identify each independent object within an image through a distance map, which is estimated by the watershed algorithm.

Supplementary Fig 2. Schematic representation of ResUNet architecture. Schematic representation of the kind of architecture (ResUNet) implemented to build the convolutional neural network for our Deep Learning model (**Xiang et al., 2018**).

Supplementary Fig 3. Schematic depiction of the mathematical comparison of the performance of a model. The accuracy of each one of the three methods employed in this work (Voronoi, Deep Learning model and reinforced Deep Learning model) was assessed based on the similarity between (**A**) the ground truth labels and (**B**) the predicted 3D segmentation. The second image (**B**) is a prediction made by the reinforced Deep Learning model and the first image (**A**) is its corresponded ground truth. In the third image (**C**) is shown the overlap between both images, and in the last image a specific region is amplified.

Supplementary Fig 4. Probability maps from the inferences of the reinforced Deep Learning model. (**A**) Original cells-contours image from collection 20210729_pos1 and its corresponded resultant probability map. (**B**) Original cells-contours image from collection 20200225_pos1 and its corresponded resultant probability map. (**C**) Original cells-contours image from collection 20190806_pos3 and its corresponded resultant probability map.

Tables Headers

Table 1. Validation analysis of the performance of the Voronoi model, the Deep Learning model and the reinforced Deep Learning model three-dimensional segmentations. Comparison, based on different mathematical metrics, between the segmentation provided by the Voronoi model, the segmentations provided by the Deep Learning model and the reinforced Deep Learning model compared to the ground truth labels.