



HAL
open science

3D deep-learning based quantification of the nuclei distribution for light-sheet microscopy

Xareni Galindo

► **To cite this version:**

Xareni Galindo. 3D deep-learning based quantification of the nuclei distribution for light-sheet microscopy. Bioinformatics [q-bio.QM]. Université de Bordeaux, 2022. English. NNT : 2022BORD0275 . tel-03982040

HAL Id: tel-03982040

<https://theses.hal.science/tel-03982040>

Submitted on 10 Feb 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE PRÉSENTÉE
POUR OBTENIR LE GRADE DE

**DOCTEUR DE
L'UNIVERSITÉ DE BORDEAUX**

ÉCOLE DOCTORALE DES SCIENCES DE LA VIE ET DE LA SANTÉ
SPÉCIALITÉ : BIOINFORMATIQUE

Par : **Xareni GALINDO HERNÁNDEZ**

**3D deep-learning based quantification of the nuclei
distribution for light-sheet microscopy**

Sous la direction de : Florian LEVET

Soutenue le 20 Octobre 2022

Membres du jury :

M. LEVET, Florian
M. KERVRANN, Charles
Mme. RIVIERE, Charlotte
M. GRANIER, Xavier
Mme. BADOUAL, Anaïs

Ingénieur de recherche IINS
Directeur de recherche, INRIA
Maître de conférences, Université de Lyon 1
Professeur, Université de Bordeaux
Chargé de recherche, INRIA

Directeur de Thèse
Rapporteur
Rapporteur
Examineur
Invité

Titre: Quantification de la distribution de noyaux en 3D en microscopie à feuille de lumière en utilisant l'apprentissage profond

Résumé:

Ces dernières années, la popularité des cultures cellulaires en 3D, telles que les organoïdes ou les sphéroïdes, a explosé en raison de leur capacité à offrir des modèles précieux pour étudier la biologie humaine. Ces modèles sont en effet bien plus pertinents sur le plan physiologique que les cultures en 2D. Il est ainsi crucial de pouvoir segmenter leurs noyaux de manière précise et robuste étant donné que la quantification de la distribution de ces noyaux est au fondement d'une multitude de mesures quantitatives de fonctions cellulaires importantes. Or, aujourd'hui, l'acquisition automatique de centaines d'organoïdes est devenue une réalité grâce aux progrès des systèmes de microscopie récents. Néanmoins, cette grande quantité de données crée de nouveaux goulots d'étranglement. En effet, les méthodes traditionnelles de traitement d'images nécessitent d'adapter leurs paramètres en fonction des conditions d'imagerie, empêchant de facto une analyse automatique et sans biais. Cette situation a entraîné le développement rapide de méthodes basées sur l'apprentissage profond, telles que StarDist, qui ont démontré des performances impressionnantes pour la segmentation précise des noyaux. Un autre avantage de ces méthodes est leur capacité à réduire considérablement le temps de traitement des données. Cependant, les approches d'apprentissage profond nécessitent une grande quantité de données pour entraîner et générer des modèles de segmentation précis et généraux. Malheureusement, la plupart des données annotées disponibles sont en 2D, ce qui entrave l'utilisation de ces méthodes pour la segmentation des noyaux en 3D.

L'objectif de ce doctorat était d'implémenter un pipeline permettant la segmentation automatique et robuste des noyaux de cultures 3D, une tâche que j'ai réalisée en trois étapes. Tout d'abord, j'ai annoté manuellement 4657 noyaux 3D provenant de 6 organoïdes/sphéroïdes différents. Ce jeu de données annoté de noyaux 3D représente actuellement le seul jeu de données de vérité terrain disponible de cette taille pour les noyaux 3D, une ressource importante pour la communauté pour l'entraînement et l'évaluation de nouvelles méthodes. Deuxièmement, j'ai réalisé que l'annotation de structures 3D prend énormément de temps et qu'il y a un manque d'outil disponible pour aider les annotateurs à garder une trace des annotations effectuées. J'ai donc développé le Napari Annotation Helper (NAHP), un plugin Napari conçu pour aider les utilisateurs à garder une trace des annotations encours, pour réduire les erreurs d'annotation et pour extraire des informations statistiques des images d'intensité. Troisièmement, j'ai entraîné un modèle de segmentation 3D StarDist avec l'ensemble de données de vérité terrain que j'ai créé. Ce modèle a été utilisé pour segmenter automatiquement des centaines d'organoïdes et quantifier la distribution spatiale de leurs noyaux, une preuve de concept que mon pipeline peut être utilisé pour répondre à des questions biologiques spécifiques.

Mots clés: Apprentissage profond , Segmentation, Microscopie à feuille de lumière, Haute-résolution, StarDist, Traitement d'images.

Title: 3D deep-learning based quantification of the nuclei distribution for light-sheet microscopy

Abstract:

In recent years, the popularity of 3D cell cultures, such as organoids or spheroids, has exploded due to their ability to offer valuable models to study human biology, far more physiologically relevant than 2D cultures. Being able to segment nuclei accurately and robustly is crucial as quantifying the nuclei's distribution is the basis of a variety of quantitative measurements of important cell functions. Nowadays, automatically acquiring hundreds of organoids has become a reality thanks to the advances in microscopy systems. Nevertheless, this high amount of data creates new bottlenecks as traditional image processing methods require adapting their parameters depending on the imaging conditions, preventing automatic and bias-free analysis. This has resulted in the rapid development of deep learning-based methods, such as StarDist, that demonstrated amazing performances for accurately segmenting nuclei. An additional benefit is their ability to reduce data processing time significantly. Still, deep learning approaches require a large amount of ground truth data to generate accurate and general segmentation models. Unfortunately, most datasets currently available are 2D, hampering the use of these methods for segmenting nuclei in 3D.

The goal of this Ph.D. was to implement a pipeline enabling the automatic and robust segmentation of 3D cultures' nuclei, a task I achieved in three steps. First, and because of the lack of available ground truth in 3D, I manually annotated 4657 3D nuclei spanning from 6 different organoids/spheroids. This 3D nuclei annotated dataset currently represents the only available ground truth dataset of this size for 3D nuclei, a significant resource for the community for training methods and benchmarking their results. Second, I realized that labeling 3D structures are very time-consuming and that there is a lack of available tools to help annotators keep track of the annotations performed. Thus, I developed the Napari Annotation Helper (NAHP), a Napari plugin designed to help users keep track of annotations, reduce annotation errors and extract statistical information from the intensity images to annotate. Third, I trained a StarDist 3D segmentation model with the ground truth dataset I created. This model was used to automatically segment hundreds of organoids and quantify their nuclei spatial distribution, a proof of concept that my pipeline can be used to answer specific biological questions.

Keywords: Deep learning, Segmentation, Light-sheet microscopy, High-Resolution Microscopy, StarDist, Image Processing

Interdisciplinary Institute for Neuroscience (IINS)

CNRS UMR 5297

Université de Bordeaux Centre Broca Nouvelle-Aquitaine

146 Rue Léo Saignat

33076 Bordeaux (France)

Aquí está mi familia, plantando todas las semillas.
Aquí estoy yo, recibiendo todas las flores.

Rupi Kaur

Here is my family, planting all the seeds.
Here I am, receiving all the flowers

Rupi Kaur

Acknowledgments

This thesis is dedicated to Maxime Voisin; without your support from the first to the last this thesis would have been impossible. *Thank you for being my center when my world was spinning without control.* This achievement is as much yours as mine.

To the Voisin and Gordon family for welcoming me as one of you and being my family far away from home. Special thanks to Veronique, Christof, and Genevieve Voisin; thank you for showing me your love every time you can.

Thanks to my thesis advisor, Florian Levet, and our team leader, Jean-Baptiste Sibarita, for letting me be part of this incredible project and team. Thanks to Tom for sharing the pains and glories of doing a Ph.D. To Ihssane, Laetitia, and Federica for encouraging me during the writing of this thesis.

Last but not least, I would like to thank all the members of my thesis jury for the time spent evaluating my thesis dissertation.

Agradecimientos

A mi abuelito Arnulfo, tu muerte dejo un vacío en nuestra familia. Espero estes orgulloso de mi dónde sea qué estes. Tus enseñanzas nos acompañaran siempre.

A mi abuelita Emma, por ser el pilar de una familia única.

A Helena, que esta tesis sea prueba de que puedes lograr todo lo que te propongas, no importa lo imposible o lejano que parezca, lo importante es dar el primer paso. Siempre podrás contar conmigo, te quiero.

A mi mamá, papá y hermana, gracias por apoyarme en la distancia y por ayudarme a alcanzar mis metas, nada de esto hubiera sido posible sin ustedes. Separados tal vez en la distancia, pero nunca en el corazón.

A mi tía Emma por siempre hacerme saber lo orgullosa que esta de mí y lo feliz que la hace verme cumplir mis sueños.

A mis amigas Paola (Pausy) y Mariel y mi amigo Elias, gracias por estar a mi lado sin importar el tiempo y la distancia.

Table of Content

1. INTRODUCTION	17
1.1 IMPORTANCE OF FLUORESCENCE MICROSCOPY IN BIOLOGICAL RESEARCH	17
1.2 RESOLUTION IN FLUORESCENCE MICROSCOPY	17
1.3 HIGH-RESOLUTION MICROSCOPY	19
2. DEEP LEARNING FOR BIOLOGICAL RESEARCH	24
2.1 INTRODUCTION TO DEEP LEARNING MAIN CONCEPTS	24
2.1.1 <i>Types of neural networks</i>	25
2.1.2 <i>Convolutional neural networks</i>	26
2.1.3 <i>Training methods</i>	29
2.1.3.1 Supervised learning:	29
2.1.3.2 Unsupervised learning	30
2.1.3.3 Semi-Supervised learning	31
2.1.4 <i>Batch and epoch</i>	31
2.1.5 <i>Learning rate</i>	33
2.1.6 <i>Cost / Loss</i>	33
2.1.7 <i>Activation Functions</i>	34
2.1.8 <i>Fine-tuning</i>	35
2.1.9 <i>Evaluation methods</i>	35
2.1.10 <i>Deep Learning for nuclei segmentation</i>	36
2.1.10.1 Image segmentation	36
2.1.10.2 Deep learning algorithms for nuclei segmentation	37
2.1.10.2.1 U-net	37
2.1.10.2.2 ResNet	39
2.1.10.2.3 StarDist 2D and StarDist 3D	42
2.1.11 <i>Advantages and challenges of the use of Deep Learning algorithms for nuclei segmentation</i>	44
3. ANNOTATION PROCESS FOR GROUND TRUTH GENERATION	46
3.1.1 <i>High-content imaging of organoids with the soSPIM system</i>	46
3.2 ANNOTATION PROCESS	48
3.2.1 <i>From manual annotation</i>	49
3.2.2 <i>To semiautomatic annotation</i>	51
3.2.3 <i>Annotation results</i>	52
3.3 NAPARI ANNOTATION HELPER (NAHP)	52
3.3.1 <i>Labels layer features</i>	53
3.3.1.1 Color vision color palette	53
3.3.1.2 Color blind color palette	55
3.3.2 <i>Functions to facilitate annotation</i>	57
3.3.3 <i>Annotation helper extra functions</i>	58

3.4	DISCUSSION.....	61
4.	QUANTIFYING THE NUCLEI ORGANIZATION OF 3D CELL CULTURES WITH STARDIST	63
4.1	IMAGE PREPARATION.....	63
4.2	NUCLEI SEGMENTATION RESULTS	65
4.2.1	<i>StarDist 2D implementation</i>	65
4.2.1.1	StarDist 2D segmentation results using Data Science Bowl 2018.....	65
4.2.1.2	StarDist 2D segmentation results using the original -DAPI dataset.....	66
4.2.1.3	2D StarDist segmentation results using original-DAPI and DSB2018 datasets	68
4.2.2	<i>StarDist 3D implementation</i>	69
4.2.2.1	3D StarDist segmentation results using original-SOX dataset	69
4.2.2.2	3D StarDist segmentation results using original-DAPI, original-SOX and complementary datasets.....	71
4.2.3	<i>Quantification of the nuclei organization in 3D cultures using StarDist 3D</i>	73
4.3	THE WAVELET TRANSFORM (WT) AS A PREPROCESSING STEP TO IMPROVE STARDIST SEGMENTATION.....	75
4.3.1	<i>The Stationary Wavelet transform theory</i>	76
4.3.2	<i>Thresholding of the wavelet coefficients</i>	78
4.3.3	<i>StarDist 3D implementation using Stationary Wavelet Transform preprocessed images</i>	78
4.3.3.1	RGB wavelet reconstruction image segmentation	79
4.3.3.2	StarDist 2D results using thresholded wavelet image reconstruction.....	83
4.4	DISCUSSION.....	85
5.	CONCLUSIONS AND PERSPECTIVES	87
	BIBLIOGRAPHY	89
	APPENDIX 1 : SOFTWARE USED	97
	APPENDIX 2 : COLOR DIFFERENCE FORMULA	98
	APPENDIX 3 : PUBLICATION.....	100

Glossary

2D image: standard format in which images are acquired by a standard camera. In our case, the camera used is the microscope camera.

3D image: term typically used to indicate a z-stack of 2D images.

Image stack: see 3D image.

Data Science Bowl 2018 (DSB2018): the data science competition held in 2018 whose main task was to segment cell nuclei in microscopy images. The official data set is an open-source dataset usually referred to as the DSB2018 dataset and is often used to benchmark nucleus segmentation methods.

Convolutional neural network (CNN): a class of deep neural networks including convolutional layers based on blocks responsible for appropriate image feature retrieval (via convolutions) and scaling (with pooling blocks).

Deep neural network (DNN): is an artificial neural network machine learning architecture that includes several hidden layers and can be trained to solve more complex tasks on more complex data compared to shallow neural networks.

Ground truth (GT): is information known to be real or true, provided by direct observation and measurement.

Star-shaped polygon: is a polygon that contains a point from which the entire polygon boundary is visible.

Convex polygons: All interior angles are less than 180° , and all vertices point outwards, away from the interior. Convex polygons are star-shaped

Wavelet transformation (WT): Wavelet transforms are mathematical tools for analyzing data where features vary over different scales. For signals, features can be frequencies varying over time, transients, or slowly varying trends. For images, features include edges and textures. Wavelet transforms were primarily created to address the limitations of the Fourier transform.

Convolution: In image processing, convolution is the process of transforming an image by applying a kernel over each pixel and its local neighbors across the entire image.

Kernel: A kernel is a matrix of values whose size and values determine the transformation effect of the convolution process over an image.

Encoding: to convert data into a required format. For example, a color RGB image can be encoded into the HSV format (hue-saturation-value)

Decoding: to convert a coded message into intelligible language

Intersection over union (IoU): is a number that quantifies the degree of overlap between two bounding boxes. In the case of object segmentation, IoU evaluates the overlap between the ground truth and the segmentation.

Abbreviations

2D, 3D: Two, three-dimensional

CLAHE: Contrast limited adaptive histogram equalization

CNN: Convolutional neural network

CPU: Central processing unit

CWT: Continuous wavelet transformation

DL: Deep learning

DLNN: Deep learning neural network

DSB2018: Data Science Bowl 2018 data set

DWT: Discrete Wavelet Transform.

FN: False negative

FP: False positive

GUI: Graphical user Interface

GPU: Graphics processing unit

GT: Ground truth

IoU: Intersection over union

JND: Just noticeable difference

CVD: Color vision deficiency

LUT: Look up table

LSM: Light sheet microscopy

ML: Machine learning

NA: Numerical aperture

NAHP: Napari Annotation Helper

NN: Neural network

PSF: Point spread function

TIRF: Total Internal Reflection Fluorescence

TP: True positive

WLT: Wavelet Transformation

SPIM: Selective plane illumination microscopy (SPIM)

soSPIM: Single objective selective-plane illumination microscopy

SWT: Stationary wavelet transform

3D-XY: Images obtained after applying the À-Trous Wavelet transformation using a B3 Spline 5x5 filter in x-y directions.

3D-XYZ: Images obtained after applying the À-Trous Wavelet transformation using a B3 Spline 5x5 filter in x, y, and z directions.

List of Figures

- Figure 1: a) Jablonski Diagram representing the fluorescence timeline and the different energetic levels through which a molecule can transit. b) Diagram of the loss of energy of a fluorescent molecule and internal conversion that results in a wavelength shift of the emission spectrum (the Stokes Shift)..... 18
- Figure 2: Airy disk and resolution limit. The numerical aperture NA of an optical system determines the ability of an optical system to collect the emitted photons through a collection angle α . The image of point source at the focal plane of an optical system is a 2D pattern called the point spread function (PSF). The resolution of an optical system, i.e., the ability to distinguish two close emitting point sources, is defined by the Rayleigh criterion. 19
- Figure 3: The basic setup of a fluorescence microscope. Fluorescence microscopy uses a light to illuminate the sample (light source), this light excites fluorescent dye in the sample, which then emits light of a longer wavelength. The excitation filter filters out all wavelengths of the light source, except for the excitation range of the fluorophore under inspection. The dichroic filter or beamsplitter reflects the excitation signal towards the fluorophore and transmits the emission signal towards the detector. The emission filter is located within the imaging path of the microscope. It filters out the entire excitation range and transmits the emission range of the fluorophore under inspection. The objective transmits the excitation light from the sample to form the image. The light passes down through the dichroic mirror before reaching the detector. The emission filter is located within the imaging path of the microscope and its job is to filter out the entire excitation range and to transmit the emission range of the fluorophore under inspection (Sanderson 2018)..... 20
- Figure 4: Main illuminations techniques used in fluorescence microscopy. (a) Wide-Field microscopy (WF), (b) Total Internal Reflection Fluorescence (TIRF), (c) Confocal microscopy (CF), d) Multiphoton microscopy, (f) Light sheet, or selective plane illumination microscopy (SPIM). 22
- Figure 5: Schematic representation of the soSPIM system. The excitation light source is reflected onto a 45° mirror creating a thin light sheet illuminating the sample perpendicularly to the image plane. This geometry enables 3D-volume imaging by simply synchronizing the movements of the objective and the excitation beam..... 23
- Figure 6: Architecture of a single neuron or perceptron. x_1 , x_2 , and x_3 , represent the inputs, w_1 , w_2 and w_3 the weights, f represents the activation function and y the corresponding output. 25
- Figure 7: Diagram of a dense neural network with 3 fully connected layers, input layer, hidden layer and output layer. 26
- Figure 8: Schematic representation of a convolutional neural network with two hidden layers. 27
- Figure 9: The convolution between an input layer of size $32 \times 32 \times 3$ and a filter of size $5 \times 5 \times 3$ produces an output layer with spatial dimensions 28×28 . The depth of the resulting output depends on the number of distinct filters and not on the dimensions of the input layer or filter. Edited from (Aggarwal, 2018). 28

Figure 10: Example of same and valid padding process. This diagram shows the result of a convolution with a 3x3 kernel and an input after applying valid and same padding. 28

Figure 11: Stride diagram. S=1 means no gap; the filter is applied to all the pixels. S=2 means gap of 1, the filter is applied to alternative cells. This halves the dimension of the output vector. 29

Figure 12: Supervised training diagram. Supervised learning is a machine learning method in which models are trained using labeled data..... 30

Figure 13: Unsupervised learning diagram. Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data. 31

Figure 14: Semi-supervised learning. Semi supervised learning is a machine learning method in which models are trained using labeled data and unlabeled data..... 31

Figure 15:Early stopping principle. If the performance of the model on the validation dataset starts to degrade (e.g. loss begins to increase, or accuracy begins to decrease), then the training process is stopped. early stopping could potentially improve generalization when other regularizes are absent. 32

Figure 16: Dropout process. Dropout is a regularization method that approximates training a large neural network with different using different versions of the same neural network in parallel. 32

Figure 17: Learning rate diagram. The learning rate is a tuning parameter used in machine learning controls how quickly the model is adapted to the problem. 33

Figure 18: Activation functions diagrams. An activation function is a function that is added into an artificial neural network in order to help the network learn complex patterns in the data. ReLU is a widely used activation function, especially with Convolutional Neural networks. 34

Figure 19: Intersection over union (IOU) diagram. This metric is known to be good for measuring the overlap between two bounding boxes or masks. 36

Figure 20: Semantic segmentation (left) vs Instance segmentation (right) diagram (Varatharasan et al., 2019). Semantic segmentation treats multiple objects within a single category as one entity in contrast, instance segmentation identifies individual objects within these categories. 37

Figure 21:U-net architecture. U-Net is an architecture for semantic segmentation. U-net consists of a contracting path(green) and an expansive path(orange). Edited from (Ronneberger et al., 2015). 38

Figure 22: Residual learning building block. In this block a layer $\ell - 1$ is skipped over activation from $\ell - 2$. Edited from (K. He et al., 2016). 40

Figure 23: Example network architectures for ImageNet. Left: the VGG-19 model as a reference. Middle: a plain network with 34 parameter layers (3.6 billion FLOPs). Right: a residual network with 34 parameter layers. The dotted shortcuts increase dimensions. Taken from (K. He et al., 2016) 41

Figure 24: Examples of convex and star-shaped polygons. Convex polygons are star-shaped. 42

Figure 25: Architecture of StarDist 2D (Schmidt et al., 2018).a) comparison between pixel classification and object detection using bounding boxes. b) StarDist segmentation methodology. c) StarDist neural network.	43
Figure 26: a) STARDIST-3D method is trained to densely predict object probabilities p and radial distances d_k to object boundaries. b) Schematic of STARDIST-3D architecture based on ResNet. c) During non-maximum suppression process (Weigert et al., 2020).....	43
Figure 27: Latitude–longitude lattice (top) and Fibonacci lattice (bottom). In the Fibonacci lattice, the points are much more evenly spaced, and the axial anisotropy is much smaller in comparison with the latitude-longitude lattice. Edited from (González, 2010).	44
Figure 28: a, JeWell chips in a six-well dish (left) and close-up image of the JeWells array with a density of 16 JeWells per mm ² (right). Inset shows a zoom on a JeWell inverted pyramidal microcavities flanked with four 45° mirroring surfaces. Scale bars, 3 cm (left), 500 μm (right) and 70 μm (inset). b, Schematic of the seeding procedure. c, Photograph of the imaging setup comprising a commercial inverted microscope, combined with JeWell chips, a laser scanning unit, and its custom-made control software. d, Principles of the soSPIM. Edited from (Beghin et al., 2022).....	47
Figure 29: Representative gallery of 96 neuroectoderm organoids from a library of >400 organoids (median plane of the 3D stacks) labeled with actin (gold), Sox2 (magenta) and DAPI (blue) acquired in an automated workflow in less than 1 hour. Edited from (Beghin et al., 2022).....	48
Figure 30: Examples of the images taken using SOX2 immunostaining and DAPI labeling by the team of Virgile Viasnoff at the Mechanobiology Institute of the National University of Singapore.	48
Figure 31: Examples of the images taken using DAPI labeling by the team of Virgile Viasnoff at the Mechanobiology Institute, National University of Singapore.	49
Figure 32: Results of the segmentation done using Ilastik. Ilastik software was used hoping to speed up the labeling process but due to its unsatisfactory performance this idea was discarded.....	50
Figure 33: Semi-annotation process diagram.2D labels are distinguish by different color. However, in the Fiji orthogonal view, some of the 3D nuclei annotation can be distinguishable by its color, meanwhile due to the bright-contrast configuration some are display in white although they have different values.	51
Figure 34: 3D Manual and Semi-automatic annotation results, where a) is a neuroectoderm organoid labeled manually, b) a cancer spheroid labeled using a semi-automatic process and c) is a neuroectoderm organoid labeled using a semi-automatic process.	52
Figure 35: Intensity and label file upload functions. Labels layer features : Intensity image loader (dark green), labels loader (light green), labels' color base dropdown menu (yellow), and look-up table or color palette options (orange).....	53
Figure 36: CIELAB color space. L axis describes the luminous intensity of the color, the coordinates a and b represent the main color axes, with red at positive a and green at negative a; yellow on positive b and blue on negative b. The C axis represents chroma or saturation, the H stand for hue, the hue coordinates move in a circle around the "equator" to describe the color family (red, yellow, green, and blue) and all colors in between (X-Rite, Incorporated, 2018, p.).....	54

Figure 37: The color difference between the default color base (Red) and other possible red colors to illustrate how the color difference influences the creation of the color vision color palette. 54

Figure 38: Normal color vision color palette colors calculated using a $\Delta E > 15$ in comparison with the default base color, red = (255, 0, 0). 55

Figure 39: How the rainbow colors may look to a color-blind person (Aytac, 2018). This image demonstrates the importance of implementing a color-blind palette in NAHP. 55

Figure 40: Colors used to create the color-blind color palette. This set of colors was chosen because it allows me to avoid multiple colors based on green and red, allowing color-blind users to distinguish them easily. 56

Figure 41: Color-vision color palette under deuteranopia y protanopia simulations done with Visolve. 56

Figure 42: Color-blind color palette under deuteranopia y protanopia simulations done with Visolve. 56

Figure 43: NAHP annotation functions. a) The Last label added function shows the value of the last label added to the annotated image, b) 2D label remained function shows the original 2D labels remaining in the corresponding frame, i.e., the 2D labels that need to be relabeled, c) label coordinates function shows the x and y coordinates of a certain label, and d) Label errors function shows the errors done during the annotation process. 57

Figure 44: Possible errors during annotation process. a) two different objects with the same label value, these objects can be located on the same frames or not, b) missing label along the z-axis. 58

Figure 45: Nuclei individualization example. The individualization function allows us to have an insight of the nuclei morphologies and to perform a nuclei profiling analysis. 59

Figure 46: Optimal bin packing problem diagram. a) image voxels represented by "carton boxes". b) image voxels arrange in the most efficient way. 59

Figure 47: Nuclei localization image example with centroids localization and properties table displayed. These functions allow us to have statistical information about the nuclei contained in the image. 60

Figure 48: Examples of new annotations generated using the area (magenta) and intensity (yellow) sliders vs. the original annotations done. 60

Figure 49: Napari Annotation Helper plugin full view. Functions to facilitate the annotation process: a) Fluorescence file upload, b) Labels file upload, c) Last label added, d) Original labels remained, e) Label coordinates, and f) Label error localization. Annotation helper extra functions: g) Nuclei individualization, h) Nuclei localization, i) Centroid localization, j) Properties table, k) Area threshold slider, and l) Intensity threshold slider. 62

Figure 50: Re-sizing process diagram. The resizing process allow us to avoid memory allocation problems and have patches inside the neural network flied of view. 64

Figure 51: Examples of images samples after cropping and preprocessing. Original data set sample 1 and 3.....	64
Figure 52: Prediction using the StarDist 2D model developed using DSB2018. In this zoom made in the different frames we can observe the over- and under-segmentation performed by this segmentation model.....	66
Figure 53: 3D view if the prediction using the StarDist 2D model developed using DSB2018.	70
Figure 54: Fluorescence image vs. StarDist 2D segmentation comparison using original-DAPI training dataset. In these images we can observe the poor performance achieved by this model.	67
Figure 55: StarDist 2D training performance using DAPI stained cell images. Although these graphs give the impression that the model works accurately due to the high values shown by the different metrics, a quick inspection of the segmentation obtained contradicts these conclusions.	67
Figure 56: Fluorescence image vs. StarDist 2D segmentation comparison using original-DAPI and the DSB2018 datasets.....	68
Figure 57: StarDist 2D training performance using original-DAPI and DSB2018 datasets.....	69
Figure 58: Fluorescence image stack vs. original-SOX training prediction.....	70
Figure 59: StarDist 3D training performance comparison between trainings performed with Sox2 stained cells.	71
Figure 60: Fluorescence image stack vs. original-DAPI training prediction.	72
Figure 61: Fluorescence image vs. SOX-DAPI training prediction.	72
Figure 62: StarDist 3D training performance comparison between DAPI and SOX-DAPI trainings. These figure shows clearly the difference between TP, FP and FN objects for each of the models for particular IoU values.	73
Figure 63: Examples of some of the 96 organoids acquired and segmented presented in (Beghin et al., 2022).....	74
Figure 64: Nuclei segmentation spatial bias analysis. a) 3D segmentation comparison between left-side and right-side illumination images. b) Quantification of the difference between the absolute number of nuclei detected in the two illumination directions and nuclei distribution. Edited from (Beghin et al., 2022).	75
Figure 65: Subcellular quantification of cell proliferation in oncospheres. a) StarDist segmentation contours and 3D reconstruction of segmented nuclei by surface rendering. b) detection of three different stages of cell division. c) 3D localization of individual proliferating nucleus represented in 3D. The colors code for the proliferation stages (blue for early G1, yellow for G1/S/G2, red for mitosis, middle and right panels) .Edited from (Beghin et al., 2022).....	75
Figure 66: Three levels of the λ -Trous wavelet transform. Arrows indicate pixels corresponding to non-zero entries in the filter h_i and are used to compute the center pixel at the next level. Orange	

dots are positions that full the undecimated wavelet transform would consider but that are skipped by the À-Trous algorithm. Edited from (Dammertz et al., n.d.). 77

Figure 67: a) Stationary wavelet transformation (SWT) decomposition diagram. Filters in each sample are the up-sampled versions of the previous b) Stationary wavelet filters. Edited from (Hurley, 2018). SWT is commonly used in signal processing and pattern recognition. 77

Figure 68: Image scales obtained using À-Trous Wavelet; adding the coefficient to the next image generated will give us the reconstructed version of the original image..... 77

Figure 69: Image reconstruction using thresholded coefficients. During this project we perform the reconstruction of the original image using coefficients 2, 3 and 4. 78

Figure 70: Example of wavelet scales computed using À-Trous algorithm. These images were obtained using the Jupyter Notebook available at (Zindy, 2019). 79

Figure 71: Fluorescence image stack (frame 35) and RGB image reconstructions using 3D-XY wavelet and 3D-XYZ wavelet scale 2, scale 3, and scale 4 as RGB color channels respectively. 80

Figure 72: Fluorescence image stack vs. 3D-XY wavelet image training prediction. 80

Figure 73: Fluorescence image stack vs. 3D-XYZ wavelet image training prediction. 81

Figure 74: StarDist 3D training performance comparison between trainings performed with RGB 3D-XY wavelet images (a), and RGB 3D-XYZ wavelet images(b)..... 81

Figure 75: Comparison between SOX-DAPI and RGB3D-XY segmentation results. 82

Figure 76: Comparison between SOX-DAPI (a) and RGB 3D-XY (b) performance plots..... 82

Figure 77: Example of image reconstruction using Bayes-Shrink and Visu-Srink and threshold with mode hard. a) image reconstruction performed using Bayes-Shrink method with model hard using scale 2, 3 and 4. b) image reconstruction performed using Visu-Shrink method with model hard using scale 2, 3 and 4..... 83

Figure 78: StarDist 2D segmentation results obtained using reconstructed images to perform the training and segmentation. a) segmentation results obtained when training StarDist with images reconstructed using Bayes-Shrink method with model hard using scale 2, 3 and 4. b) segmentation results obtained when training StarDist with images reconstructed using Visu-Shrink method with model hard using scale 2, 3 and 4. 84

Figure 79: StarDist 2D training performance comparison between trainings performed with images reconstructed using Bayer Shrink (a) and Visu Shrink (b) thresholding methods using hard mode. ... 84

1 | Introduction

1.1 Importance of fluorescence microscopy in biological research

To understand the nature and function of many biological processes and structures, it is necessary to identify and localize specific proteins, molecules, and macromolecules within the cellular environment. These cells, which help build up tissue and organs, are generally translucent in their natural state. Their immense number of constituents are optically indistinguishable from one another, making the identification of proteins and molecules a matter of high interest for the research community (Sanderson, 2018).

To achieve the identification of the different cell components, we need the ability to separate a specific protein or structure of interest from its surrounding environment. It also requires good contrast so that the fine structure details can be distinguished. By allowing the selective and specific detection of molecules at small concentrations with a good signal-to-background ratio, fluorescence microscopy has rapidly become a tool of choice for examining biological specimens, whether fixed or alive (Sanderson, 2018; Yuste, 2005). Besides, thanks to the large spectral range of fluorophores and fluorescence markers available on the market, it allows simultaneous imaging of different biological structures.

Fluorescence is a phenomenon occurring when a fluorescent molecule (or fluorophore) absorbs a photon, subsequently emitting a new photon with a different wavelength. The difference in energy between the emitted and absorbed photons is called *Stokes shift* (Figure 1), and its magnitude determines how easily these photons can be separated (Sanderson, 2018). This shift is critical as it provides contrast in optical microscopy: one has only to choose a specific spectral filter to filter out the excitation light, only detecting the fluorophores' emitted light. Nevertheless, this fluorescence lifecycle only occurs a limited number of times until photobleaching, a structural change of the molecule that prevents it from absorbing further photons, resulting in a limited observation time.

Through tagging with some of the many fluorophores that exist, single proteins become visible on the microscope. And since those available fluorophores cover a wide spectral range, simultaneous imaging of different biological samples in crowded environments with great accuracy has become a reality. In the end, contrast, specificity, and a simple implementation into an optical system have propelled fluorescence microscopy to a prominent position in life sciences.

1.2 Resolution in Fluorescence microscopy

The theoretical resolution achievable by fluorescence microscopy is limited to the micrometer scale as it is subject to the laws of light diffraction. In the microscopy context, resolution depicts the distance between two-point sources that can be recognized as distinct. Therefore, the spatial resolution determines the finest details a microscope can image (Jerome & Price, 2018; Sanderson, 2018).

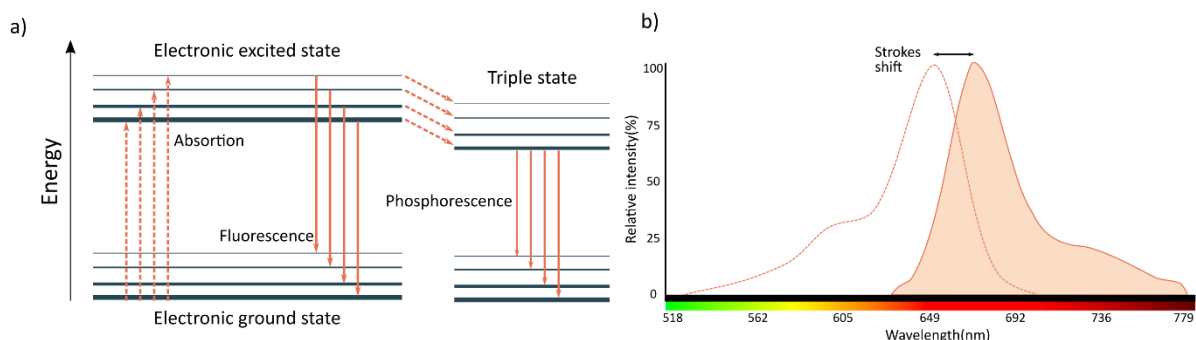


Figure 1: a) Jablonski Diagram representing the fluorescence timeline and the different energetic levels through which a molecule can transit. b) Diagram of the loss of energy of a fluorescent molecule and internal conversion that results in a wavelength shift of the emission spectrum (the Stokes Shift).

The primary cause of lateral image blurring in a microscope is light diffraction. When photons from an infinitesimally small point are focused through a lens, the resulting two-dimensional image is a central bright circle surrounded by a series of halos, known as the Airy pattern. The intensity distribution of the Airy pattern is called the point spread function, or PSF. The width of its main lobe depends critically on two factors: the wavelength of the emitted photon and the amount of light collected by the imaging system. Ultimately, the image of a biological sample is just the sum of the PSF of all its fluorescent molecules. This implies that, when a sample is illuminated, the localization of a molecule is possible when the main lobe of its Airy disk is distinguishable from those of neighboring molecules. However, when the separation between molecules is not enough, we are unable to locate them precisely (Figure 2)

In the nineteenth century, Lord Rayleigh determined the resolution limit of a microscope using the Airy disk. Rayleigh's resolution criterion states that two adjacent objects are just resolved when the maximum of the PSF of one object coincides with the first minimum of the other, in the image plane (Jerome & Price, 2018; Sanderson, 2018). This can be expressed as:

$$r = 0.61 \frac{\lambda}{NA} = 0.61 \frac{\lambda}{n(\sin \theta)} \quad (1)$$

Where:

r = minimum resolved distance,

λ = wavelength

NA = is the numerical aperture of the objective

n = refractive index of the medium

θ = half of the collecting angle of the lens

If we desire to image a 3D object, it is important to consider the axial resolution along the z-axis as well as the lateral spatial resolution. In this case, the ability of a microscope to separate two point sources having the same lateral x-y position but a different z axial position is critical, and the PSF also becomes a three-dimensional object. In the z-axial direction, the PSF is broader and more spread out, causing a resolution decrease along this axis to be three to four times worse than in x-y directions. This brings our attention to another important concept, the depth of field. The depth of field is defined as the thickness of the optical section along the z-axis within which objects in the specimen plane are in acceptably sharp focus. The larger the numerical aperture of the objective, the smaller the depth of field will be (Jerome & Price, 2018; Sanderson, 2018). The minimum distance resolved in the z-axis by an objective, assuming no spherical or chromatic aberration, can be given as:

$$d_z = \frac{2n\lambda}{NA^2} \tag{2}$$

Where:

- n = refractive index of the immersion medium
- λ = wavelength
- NA = is the numerical aperture of the objective

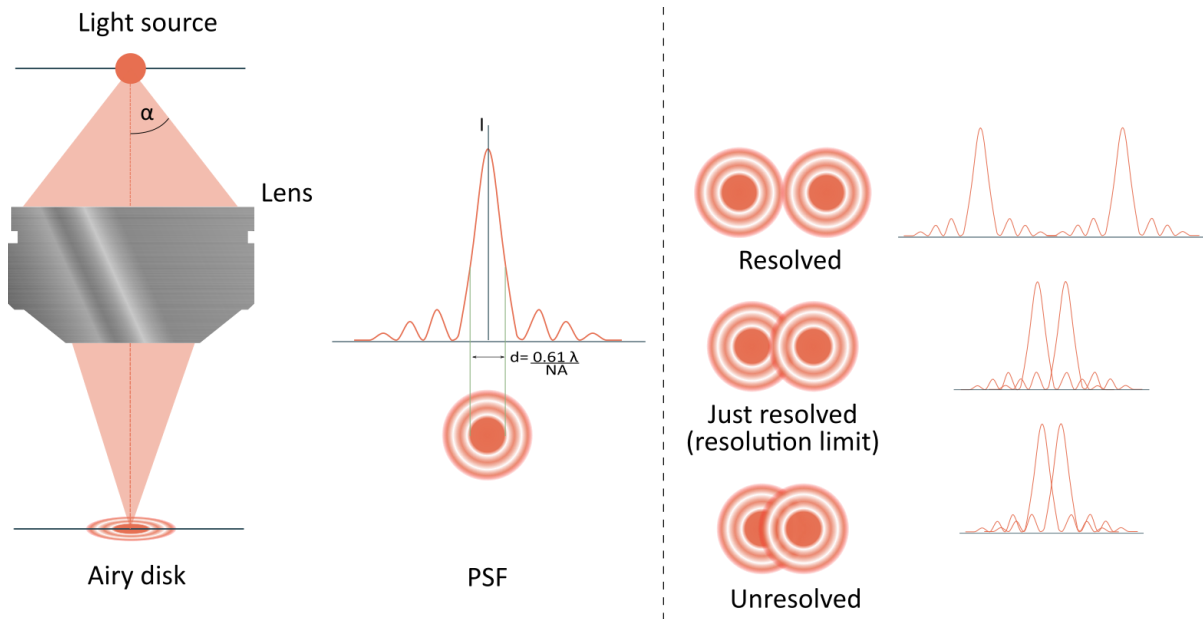


Figure 2: Airy disk and resolution limit. The numerical aperture NA of an optical system determines the ability of an optical system to collect the emitted photons through a collection angle α . The image of point source at the focal plane of an optical system is a 2D pattern called the point spread function (PSF). The resolution of an optical system, i.e., the ability to distinguish two close emitting point sources, is defined by the Rayleigh criterion.

It is worth reiterating that all criteria for resolution are intimately dependent upon the Airy disk size. We can circumvent some of its limitations, but if we use a lens for imaging, we cannot completely get rid of the consequences of the diffraction of light.

1.3 High-resolution microscopy

Fluorescence microscopy (Figure 3) is one of the most used imaging modalities to image living and fixed biological samples and has become the main tool for biology and biomedical science. Despite its inherent limitations caused by the diffraction limit, its importance has driven researchers to improve its effective resolution by optimizing its contrast. This has fueled the development of several types of fluorescence microscopes.

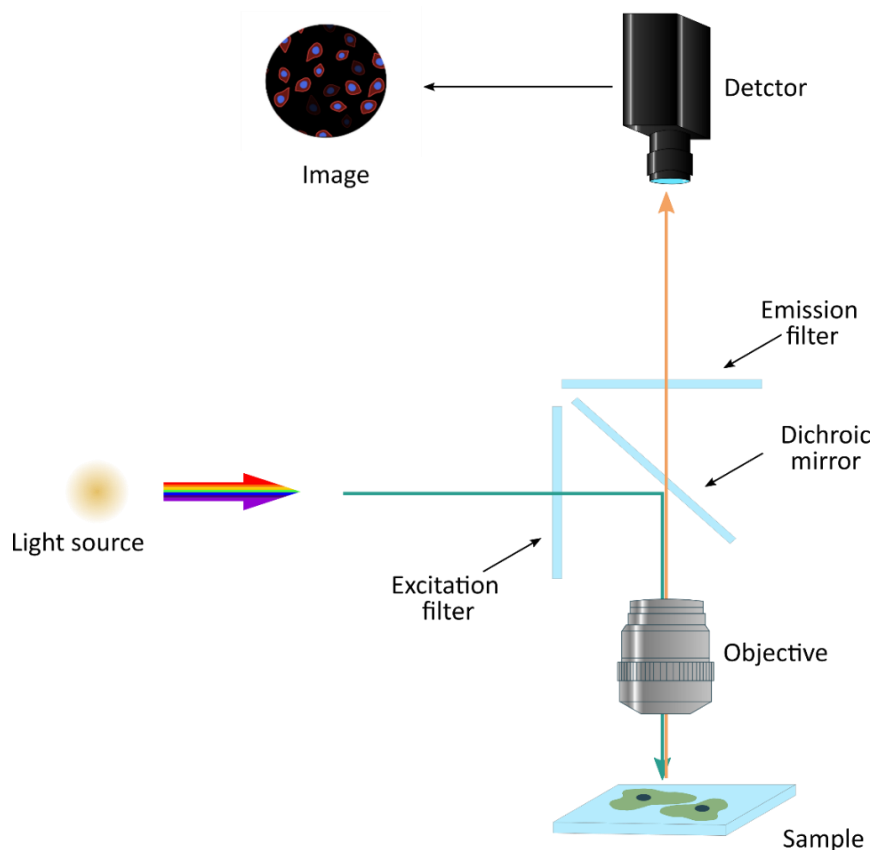


Figure 3: The basic setup of a fluorescence microscope. Fluorescence microscopy uses a light to illuminate the sample (light source), this light excites fluorescent dye in the sample, which then emits light of a longer wavelength. The excitation filter filters out all wavelengths of the light source, except for the excitation range of the fluorophore under inspection. The dichroic filter or beamsplitter reflects the excitation signal towards the fluorophore and transmits the emission signal towards the detector. The emission filter is located within the imaging path of the microscope. It filters out the entire excitation range and transmits the emission range of the fluorophore under inspection. The objective transmits the excitation light from the sample to form the image. The light passes down through the dichroic mirror before reaching the detector. The emission filter is located within the imaging path of the microscope and its job is to filter out the entire excitation range and to transmit the emission range of the fluorophore under inspection (Sanderson 2018).

Widefield microscope (Figure 4a) is one of the most basic and used fluorescence microscopes. It uses a parallel beam to illuminate the whole specimen at once to excite the fluorophores. While its way of illuminating the samples allows simple and fast imaging, it also means that the sensor collects out-of-focus light, which is detrimental to the image contrast and the microscope's effective resolution. This also gives widefield a poor z-axis resolution, making it unsuitable for 3D imaging (Jerome & Price, 2018; X. Wang & Lai, 2021).

Improving the effective resolution can be achieved by limiting the thickness on which the sample is imaged, de facto reducing out-of-focus blur. One example is Total Internal Reflection Fluorescence (TIRF) microscopy (Figure 4b), that confines the excitation light to a thin slice. Total internal reflection is an optical phenomenon occurring when a beam is totally internally reflected in the coverslip due to the refractive index difference between the glass and the imaging medium. This creates a standing wave propagating along the interface between the glass coverslip and the watery medium of the sample, creating an evanescent field whose amplitude decays exponentially. In a typical experiment setup, the fluorophores near the interface will be excited by the evanescent field (Sanderson, 2018; X. Wang & Lai, 2021, 2021). Due to the low penetration depth of the evanescent wave, there is almost no background fluorescence, resulting in an extremely high contrast signal. One major drawback of the

TIRF method is that the penetration depth of the evanescent field can only be approximately 60 to 200 nm beyond the upper surface of the coverslip, making it unsuitable for imaging structures deeper in the cell.

As opposed to confining the excitation light, another solution is to provide widefield excitation while preventing out-of-focus light from being detected. In the case of confocal microscope (Figure 4c) a pinhole is placed in a plane optically conjugated to the image plane, preventing both the illumination of the entire field of view and out-of-focus light from being detected (Sanderson, 2018; X. Wang & Lai, 2021). This illumination method helps to overcome the background glare and provides good sectioning in the axial direction, making it possible to create blur-free images in three dimensions. Nevertheless, confocal microscopes are temporally limited since they use a point scanning excitation, possibly resulting in noisier images than widefield. Besides the lasers' high energy, they can easily cause photobleaching of fluorophores and phototoxicity of biological tissue, particularly in living cells (Jerome & Price, 2018; Sanderson, 2018). However, spinning disk confocal microscopy, that uses a multiple-point scanning system, has been developed to overcome this limitation. In this configuration, the sample is both illuminated, and the emitted light detected through a spinning disk with pinholes. This configuration allows faster acquisition and lower light requirements. However, some resolution is lost due to the crossing between multiple fluorescence points and fixed pinhole sizes (X. Wang & Lai, 2021).

Multi-photon microscopy (Figure 4d) employs femtosecond laser pulses to ensure that two photons (occasionally three) will hit a molecule within 1 femtosecond of difference. One key benefit of this process is its ability to restrict excitation to a tiny focal volume in thick samples ($\sim 0.1\mu\text{m}^2$). As a result, multiphoton microscopy offers low phototoxicity and higher spatial and temporal resolution in comparison with other *in vivo* imaging methods. This type of microscope is also suitable for imaging thick samples, offering an invaluable tool for studying cellular and subcellular mechanisms within the tissue (Dunn & Young, 2006; Sanderson, 2018). Nevertheless, it is also a laser-scanning technique, which brings some inherent temporal limitations.

Selective Plane Illumination Microscopy or better known as SPIM (Figure 4f) employs two objectives in a perpendicular geometry, resulting in a planar illumination of the sample from the side. In this configuration, only a well-defined volume around the focal plane of the detection optics is illuminated, providing good optical sectioning, and strongly reducing photo-bleaching. The thickness of the illuminating light sheet and the detection lens NA determine the axial resolution of the instrument (Cella Zanacchi et al., 2011; Engelbrecht & Stelzer, 2006; Huisken et al., 2004). Finally, the fluorescence signal emitted from the in-focus section is detected in parallel for the entire field of view, which provides high imaging speeds.

The main disadvantage of SPIM lies in its specific geometry, as extra optics are required to generate the light sheet. This induces significant constraints on both the imaging system and the sample mounting, preventing automatic acquisition of a vast amount of samples. In addition, SPIM microscopes, like any other fluorescence imaging system, suffer from scattering and absorption within the tissue. This issue can be mitigated by multi-view reconstruction, which involves collecting multiple datasets of the same object from different directions and combining their high-quality information into a single image in a postprocessing step (Huisken et al., 2004).

To circumvent this mechanical limitation, one can use a unique objective to create the light sheet and collect the fluorescence signal. The single objective selective plane illumination microscopy (Figure 5), soSPIM system (Galland et al., 2015), developed by the Sibarita team, takes benefit from this idea by using 45° mirrors to reflect an incoming laser beam in order to create a light sheet perpendicular to the detection objective optical axis. The first benefit of this technology is the possibility to use high numerical aperture objectives. Another benefit is related to the embedding of those 45° mirrors inside

dedicated microfabricated devices called Jewells. Their fabrication is standardized, and hundreds to thousands of them are accessible in one 96-well plate, significantly improving the imaging throughput of the system. It allows standardizing and parallelizing the culture and the imaging of 3D cell cultures (Beghin et al., 2022).

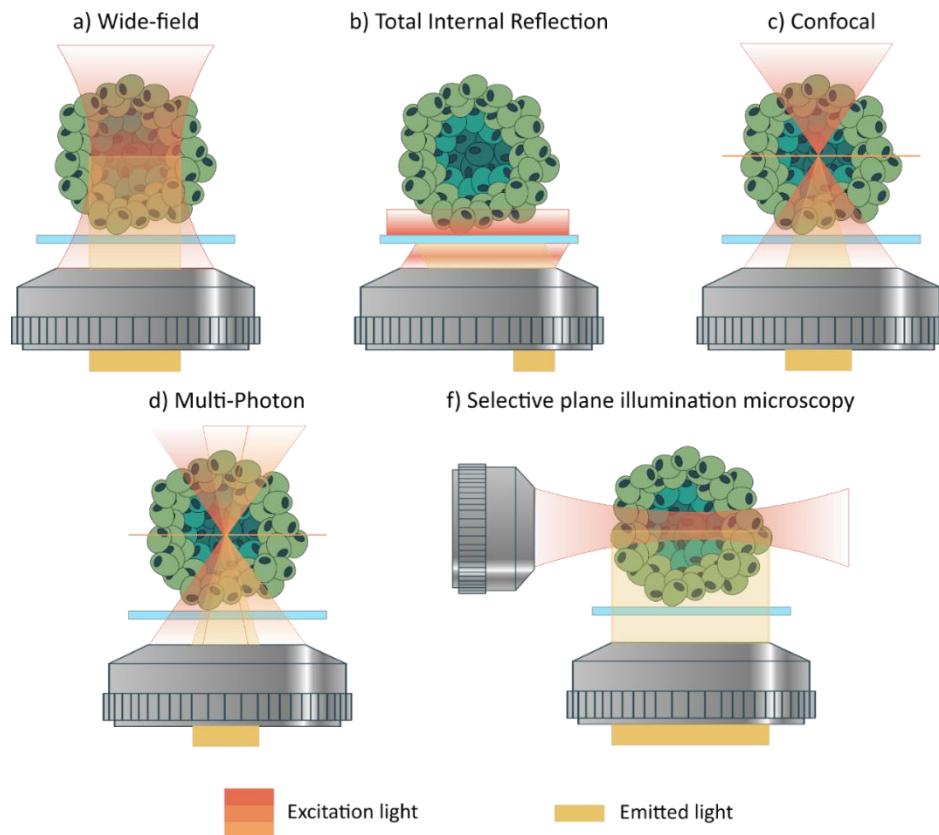
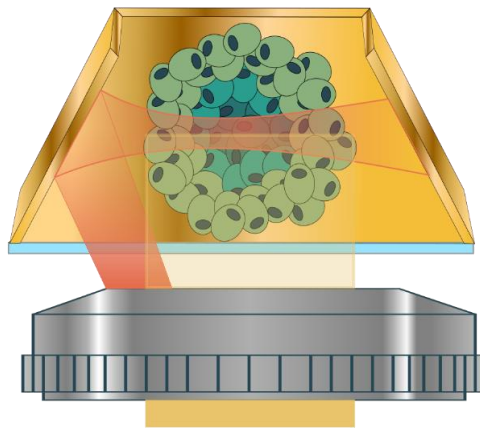


Figure 4: Main illuminations techniques used in fluorescence microscopy. (a) Wide-Field microscopy (WF), (b) Total Internal Reflection Fluorescence (TIRF), (c) Confocal microscopy (CF), (d) Multiphoton microscopy, (f) Light sheet, or selective plane illumination microscopy (SPIM).

During the elaboration of this project, the images we will use were taken using the soSPIM system at the Mechanobiology Institute of the National University of Singapore by the team of Virgile Viasnoff¹. The detail of the images will be discussed in further sections of this manuscript.

¹ <https://www.mbi.nus.edu.sg/science-features/so-spim-fcs/>



Excitation light Emitted light

Figure 5: Schematic representation of the soSPIM system. The excitation light source is reflected onto a 45° mirror creating a thin light sheet illuminating the sample perpendicularly to the image plane. This geometry enables 3D-volume imaging by simply synchronizing the movements of the objective and the excitation beam.

2 | Deep Learning for Biological Research

Microscopy imaging techniques are essential for studying fundamental biological processes. The improvement of microscopes' optic and electronic systems, analysis software, fluorophores, and computers have helped understand many fundamental biological principles (Sanderson, 2018). The new generations of microscopes allow researchers to acquire more images than ever before at a much higher speed with a higher spatial resolution and a larger field of view. However, as all these seem to be advantages, it is worth mentioning that the high production of image datasets makes it very challenging to analyze them using traditional approaches. Therefore, automatizing the analysis process has de facto become a priority. Deep neural networks have proven their ability to improve image analysis pipelines compared to traditional approaches (H. Wang et al., 2018). In addition, the combination of their ability to automatize the processing of a large number of microscopic images with their improved robustness and generalization have resulted in a fast-paced adoption in biology. However, limited training sets and low image quality are the main issues to overcome.

One of the most important tasks in biological research is segmenting cell nuclei, since identifying and quantifying their localization and organization are the basis of a variety of quantitative measurements of important cell functions. Until recently, the dominant approaches for this task have been based on classic image processing algorithms like Otsu thresholding and watershed. However, traditional methods also require expertise to properly adjust analysis parameters, a process usually highly dependent on the different experimental conditions. Deep learning neural networks have revolutionized image analysis techniques, proving to give an outstanding performance in nuclei segmentation. Besides, deep learning has helped to create fully automated and robust methods able to work with a wide variety of experimental conditions, different cell lines, and types of light microscopy (Caicedo et al., 2019). However, the robustness of most traditional machine learning methods is highly dependent on image quality, the extracted image features, and the performance of the feature classification methods. For this reason, it is essential to know the basic concepts of deep learning in order to get the most out of them.

2.1 Introduction to Deep Learning main concepts

Artificial neural networks (ANNs) are algorithm-based systems that try to mimic biological neural networks; i.e., these algorithms try to mimic the way human brains process information. This has made neural networks able to provide strong solutions to real-world challenges and help to address problems in several areas like classification, prediction, filtering, optimization, pattern recognition, among others. In contrast with biological neural networks, artificial neural networks pretend to abstract the complexity of the human brain focusing on what may theoretically matter most from an information-processing point of view. As well as their biological counterparts, ANNs seek to use the same information processing features like nonlinearity, high parallelism, resilience, fault tolerance, learning, the capacity to handle imprecise and fuzzy information and the ability to generalize (Thakur, 2021).

Artificial neural networks learn how to identify and classify objects in a similar way as biological neural networks. They execute tasks by analyzing samples and inferring rules without the need for direct interaction with a user. ANN-based models are empirical in nature; nonetheless, they may give practically correct answers to precisely or imprecisely defined issues, as well as phenomena that can only be understood through experimental data and field observations (Thakur, 2021).

Artificial neural networks are basically constructed following the next assumptions:

- Information is processed by a large number of basic components known as neurons.
- Signals are transferred from one neuron to the next via connecting linkages.
- Each connecting link has a weight associated with it, which transforms the signal conveyed in a conventional neural network.
- Each neuron determines its output signal by applying an activation function (typically nonlinear) to its net input (sum of weighted input signals) (Thakur, 2021).

Being artificial neural networks, deep learning neural networks are computational models that are composed of multiple processing layers and learn representations of data with multiple levels of abstraction (H. Wang et al., 2018). Deep learning architectures such as deep neural networks, deep belief networks and recurrent neural networks have been applied to improve the algorithms used in speech recognition, visual object recognition, object detection and many other domains. These algorithms discover intricate structure in large data sets by using a backpropagation algorithm. In consequence, the machine learns how it has to adjust its parameters to output an accurate result. These adjustable parameters, often called weights, are real numbers that can be seen as 'knobs' that define the input–output function of the machine (Aggarwal, 2018; Chollet, 2018; Thakur, 2021). Before talking about the basic concepts, we need to highlight two key aspects:

- Deep learning models consist of multiple layers or stages of nonlinear information processing.
- The deeper we go in a neural network, the more abstract the features extracted become.

2.1.1 Types of neural networks

To describe neural networks, we will begin by describing the simplest possible neural network, one which comprises of a single "neuron", this simple neural network being referred as perceptron (Figure 6). In this case, the input is mapped directly to the output by a function known as the "activation function." The edges from the input to the output contain the weights $w_1 . . . w_d$, with which the features are multiplied and added at the output node after the activation function converts the aggregated value into a class label. This type of neural network is known as single-layer neural network (Aggarwal, 2018; Chollet, 2018).

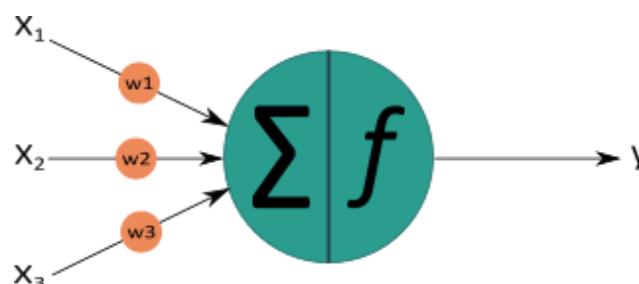


Figure 6: Architecture of a single neuron or perceptron. x_1 , x_2 , and x_3 , represent the inputs, w_1 , w_2 and w_3 the weights, f represents the activation function and y the corresponding output.

Neural networks are modeled as a collection of single neurons arranged in layers (Figure 7) to build a net, with cycles being forbidden. Neural networks are mainly composed of three layers, the input, the output, and one or more hidden layers. The input layer gets the data and passes it to the hidden layers. The hidden layers, in which weights are applied to the inputs, direct the inputs through an activation function and send their output to the output layer. This architecture is known as feed-forward because successive layers feed into one another in the forward direction from input to output. Unlike all layers, the output layer uses a converting function that converts values into probabilities. Those values can then be passed to a threshold function to determine the class scores (e.g., in classification) or a real-valued target (e.g., in regression) (Kotsiantis, 2017).

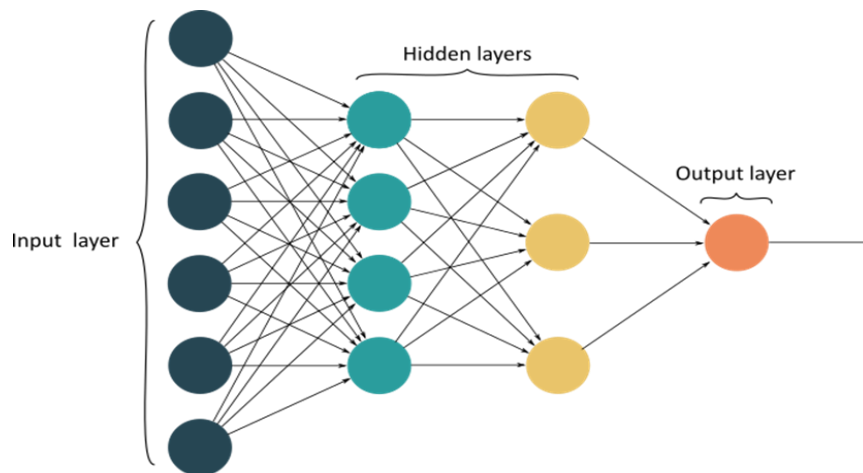


Figure 7: Diagram of a dense neural network with 3 fully connected layers, input layer, hidden layer and output layer.

The neural networks' learning ability strongly depends on the weight between neurons. For regular neural networks, the most common layer type is the fully connected layer in which neurons between two adjacent layers are fully pairwise connected. These types of networks are called fully connected neural networks. Therefore, the architecture of the neural network is almost fully defined, once the number of layers and the number/type of nodes in each layer have been defined (Aggarwal, 2018; Goodfellow et al., 2016).

Neural networks come in different varieties, and they are frequently employed in a variety of machine and deep learning applications. Convolutional neural networks are the most common networks used in computer vision and image processing.

2.1.2 Convolutional neural networks

For a convolutional neural network (CNN), an image is seen as a grid-structured composed of pixels, where each pixel has similar features to its neighbors. Therefore, each input image has a strong spatial dependency within its local regions. Convolutional neural networks (Figure 8) learn the spatial dependencies within the different image regions; this means that CNN tends to create similar feature values from local regions with similar patterns. Contrary to densely connected layers that learn global patterns, convolutional layers learn local patterns. This key characteristic gives convnets two interesting properties: translation invariance and the capacity to learn spatial hierarchies of patterns (Aggarwal, 2018; Chollet, 2018; Zuo et al., 2015).

The translation invariant property means that after learning a certain pattern, the network will be able to recognize it everywhere, needing fewer data to learn representations that have generalization power.

Besides, a CNN's ability to learn spatial hierarchies of patterns means that different layers will learn different spatial patterns. For example, the first convolutional layer will learn small local features like edges; the second convolution layer will learn larger patterns made of the features of the first layers, and so on. This allows convnets to efficiently learn increasingly complex and abstract visual concepts (Aggarwal, 2018; Chollet, 2018).

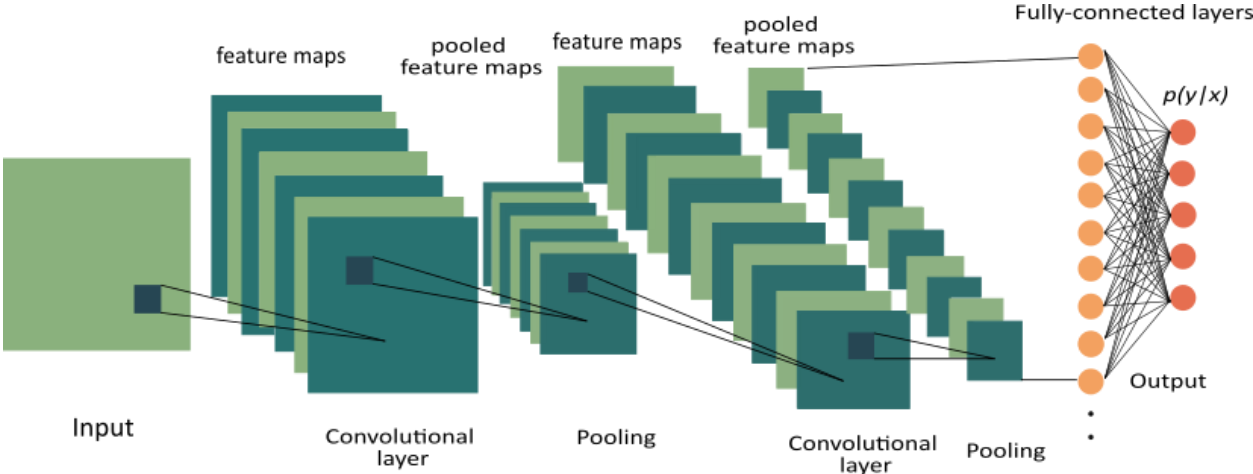


Figure 8: Schematic representation of a convolutional neural network with two hidden layers.

Convolutional neural networks are usually comprised of three types of layers, convolutional, pooling and fully connected layers. The convolution layers extract patches from the input image or feature maps and perform convolutional operations over all the patches. This means that a filter (also called kernel) will slide over the different patches to extract the different features contained in the input image or feature maps. The width and height of a features map are determined by the size of the filters used in the convolutional layer, while the depth is determined by the number of filters employed in the layer (Figure 9). Because it directly affects the number of parameters, the number of filters used in each layer regulates the model's capacity. Therefore, a considerable number of filters are required to capture a broad variety of possible shapes that combined generate the final image. The most commonly used kernels are those of size 3x3 or 5x5 (Aggarwal, 2018; Chollet, 2018).

After passing through a convolutional layer, the dimensionality of the input feature map can be reduced, increased, or remain constant. If we want the layer's feature map output to be of the same size as the input feature map, we need to apply an operation called padding. Padding consists in adding columns and rows on each side of the output feature map until its matches the input feature map size. This process is called "same padding," which means that the padding will be done such as the output has the same width and height as the input. When no zeros are added to the output feature map, the process is called "valid padding" (Aggarwal, 2018; Chollet, 2018). Padding types diagram can be seen in Figure 10.

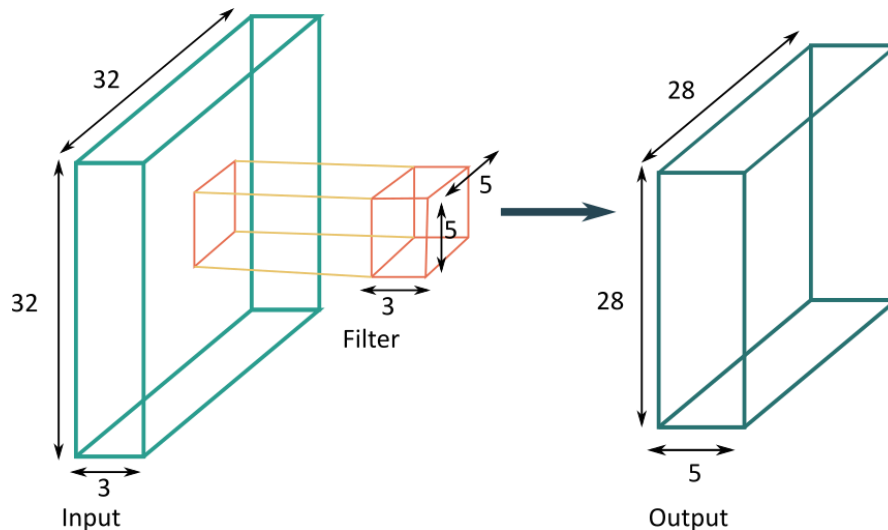


Figure 9: The convolution between an input layer of size $32 \times 32 \times 3$ and a filter of size $5 \times 5 \times 3$ produces an output layer with spatial dimensions 28×28 . The depth of the resulting output depends on the number of distinct filters and not on the dimensions of the input layer or filter. Edited from (Aggarwal, 2018).

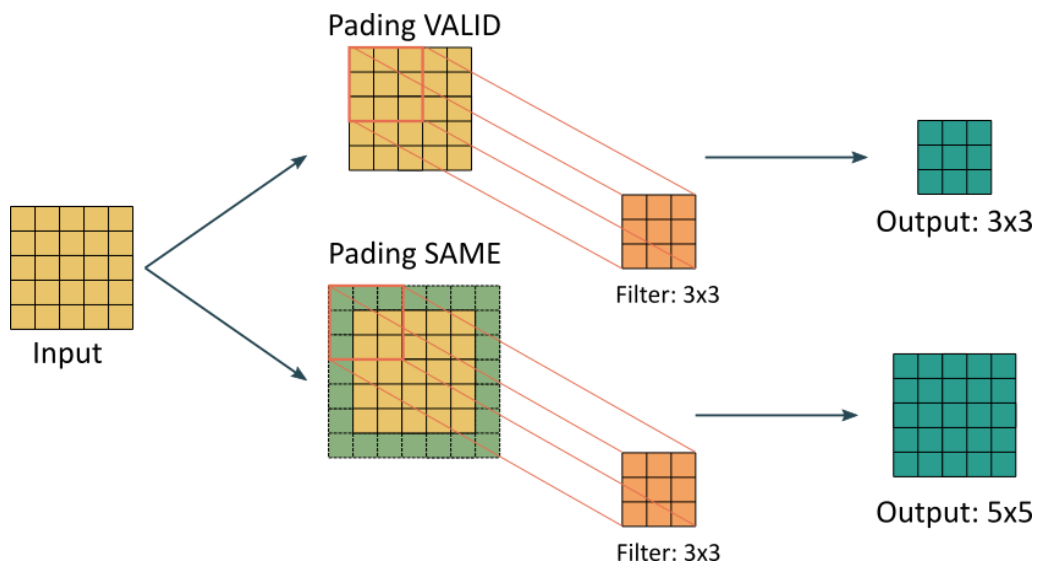


Figure 10: Example of same and valid padding process. This diagram shows the result of a convolution with a 3×3 kernel and an input after applying valid and same padding.

Another factor that influences the feature maps' output size is the kernel stride. The stride is the step size used to slide the filter over the input image or feature map. In general, the stride is equal to 1, but depending on the size of the feature maps desired, larger strides can be used (Figure 11) (Aggarwal, 2018).

The pooling layer gets the feature maps from the convolutional layers as input. It reduces the number of parameters and the computational complexity of the model, and it is typically achieved by using the "max" function. The max-pooling uses a 2×2 kernel with a stride value of 2 to extract the maximum value contained under this filter and downsample the feature maps by a factor of 2; this operation helps to reduce the number of features to process. Using a filter that extracts the maximum value of vicinity helps the network differentiate the input image's distinct features better than other downsampling methods. It is important to mention that the pooling operation does not change the

depth of the feature map on which the operation is applied (Aggarwal, 2018; Chollet, 2018; O’Shea & Nash, 2015).

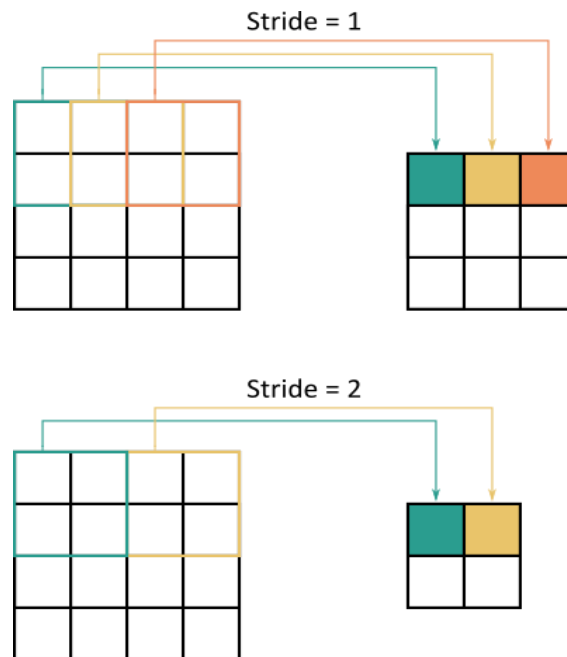


Figure 11: Stride diagram. $S=1$ means no gap; the filter is applied to all the pixels. $S=2$ means gap of 1, the filter is applied to alternative cells. This halves the dimension of the output vector.

Finally, the fully connected layers perform the same work as the layers on the artificial neural networks (ANNs) and attempt to reproduce class scores from the activations, i.e., this layer is used to perform the class classifications. Although all convnets have fully connected layers, they can differ in nature depending on the purpose of the neural network; for example, in classification or segmentation (Aggarwal, 2018; O’Shea & Nash, 2015).

In the following sections, we will provide examples of how convolutional neural networks are used to address the challenges that segmenting biological imaging can present. I will also talk about the CNNs used during the development of this project as U-Net and StarDist. It is important to mention that even though we did not work with all of these examples during this project, we think it is necessary to talk about them due to their importance in image analysis and biological research.

2.1.3 Training methods

2.1.3.1 Supervised learning:

Supervised learning algorithms (Figure 12) learn to associate some input with a certain output given a pairwise training data set of input x , or training data, and output y , also known as target. By learning to associate certain inputs with their corresponding expected output, supervised learning algorithms are able to produce a general hypothesis. This hypothesis is then used to make predictions about data never seen by the algorithm. The main goal of supervised training is to minimize the error between the target output and the output computed by the algorithm (Chollet, 2018; Goodfellow et al., 2016).

To perform supervised learning each instance is given with a known label to the neural network as input, and unlabeled data is given as test data, although to evaluate its performance, the test labels need to be known. The target and test labels can be continuous, categorical, or binary (Kotsiantis, 2017). Almost all popular deep learning applications today, including optical character recognition,

audio recognition, image classification, and language translation, fall into this group. In the case of image segmentation, the input data is an image, and the output or target is a pixel-level mask, also known as annotation (Chollet, 2018).

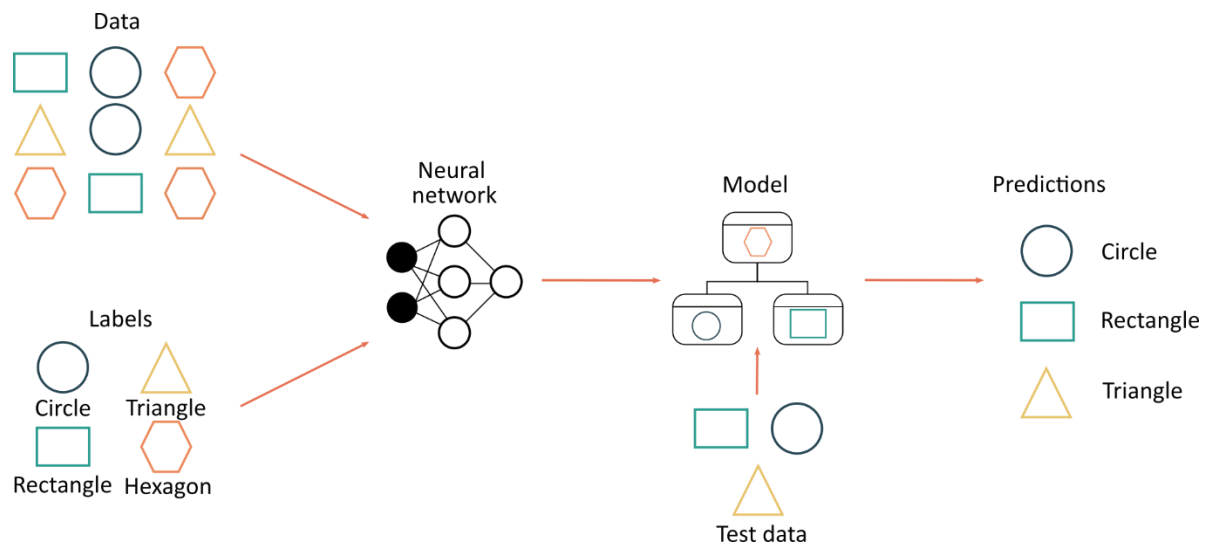


Figure 12: Supervised training diagram. Supervised learning is a machine learning method in which models are trained using labeled data.

2.1.3.2 Unsupervised learning

In contrast with supervised algorithms, unsupervised learning (Figure 13) makes no initial assumptions about how the data are related. Instead, they seek to discover and characterize the hidden distribution of data (Shariff et al., 2010).

In unsupervised learning, the data is given to the algorithm unlabeled. Meaning that algorithms learn data features and cluster the data according to those features by themselves. The few features the unsupervised algorithms learnt are used to recognize the class of the new data introduced. Contrary to supervised learning algorithms, which goal is to minimize the error or the misclassification between computed and target outputs, unsupervised learning algorithms aim to maximize the similarities between cluster prototypes and items from the dataset, meaning that unsupervised learning depends on the data similarities with the clusters prototypes. This way of learning algorithms can be extremely useful to distinguish images in which the general phenotype, or the effects of added drugs, are unknown. Examples of unsupervised methods can be found in High-content screening analysis (HCA), where unsupervised learning is used to build subcellular location trees for large numbers of randomly tagged proteins (Jo, 2021; Mahesh, 2018; Shariff et al., 2010).

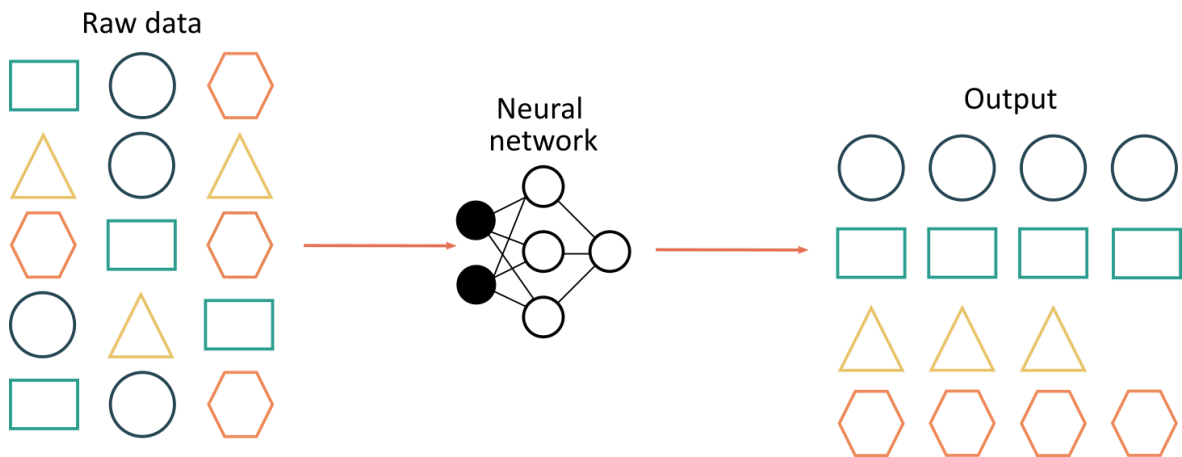


Figure 13: Unsupervised learning diagram. Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data.

2.1.3.3 Semi-Supervised learning

Semi-supervised machine learning (Figure 14) is a combination of supervised and unsupervised training methods; this means that it uses labeled as well as unlabeled data to perform certain learning tasks.

Semi-supervised classification methods are particularly relevant to scenarios where labeled data is scarce, as constructing a reliable supervised classifier would be uncanny. However, a better classifier can be achieved by adding a sufficient volume of unlabeled data to the training set. Nevertheless, those unlabeled data are required to properly sample the distribution of the raw data (Shariff et al., 2010; van Engelen & Hoos, 2020).

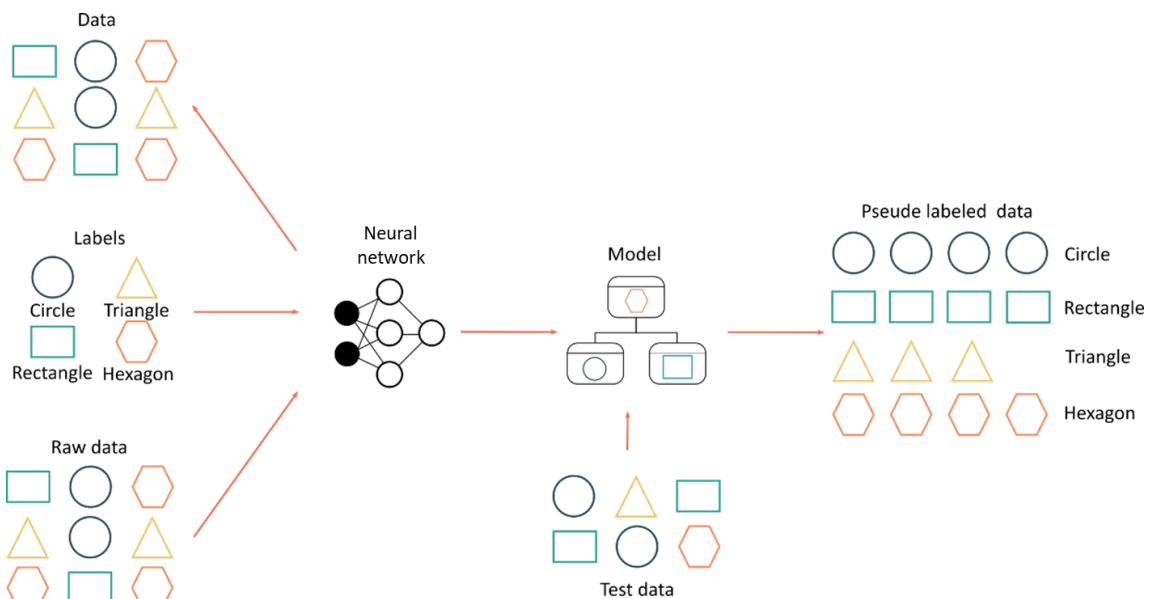


Figure 14: Semi-supervised learning. Semi supervised learning is a machine learning method in which models are trained using labeled data and unlabeled data.

2.1.4 Batch and epoch

While training a neural network, instead of sending the entire input in one go, we randomly divide the input into a pre-defined number of chunks of equal size called *batches*. Training data on batches makes the model more generalized because when using small batch sizes, the model tends to converge more rapidly and efficiently to a global minimum, thanks to this reduced size (Cole Matt R., 2019; Keskar et

al., 2017). One iteration of training all those batches in both forward and backpropagation is defined as an epoch. The number of epochs is critical: while one would expect that a higher number would result in a better network accuracy, the risk of overfitting is to be considered (Cole Matt R., 2019).

Early stopping (Figure 15) is a technique used to reduce the possibility of having an overfitting model without compromising its accuracy. This technique evaluates the network performance on the test set after each epoch (or every N epochs). If the network outperforms the previous best model saved, a copy of the network at the current epoch is saved. The final model is the model that has the best test set performance (“Early Stopping,” n.d.). Another regularization technique to prevent overfitting is the dropout. Dropout (Figure 16) prevents over-fitting of the network by randomly and temporarily removing some of the hidden layers’ neurons and their connections for each epoch of the training process (Srivastava et al., 2014).

Because large datasets can take a longer time to train, to not lose the training progress, it is advisable to implement checkpointing of the model’s parameters (weights) at every certain number of epochs; but only if it is the best weights at that point in time. This practice is called checkpointing and is the term used to describe when a “snapshot” of the model’s parameters after a certain number of training epochs is saved. Saving the best weights allows us to keep a copy of the progress at a given epoch in case you want to tune your hyperparameters at any given epoch or resume the training process from any epoch with a checkpoint performance (Ku Wee Kiat, 2018).

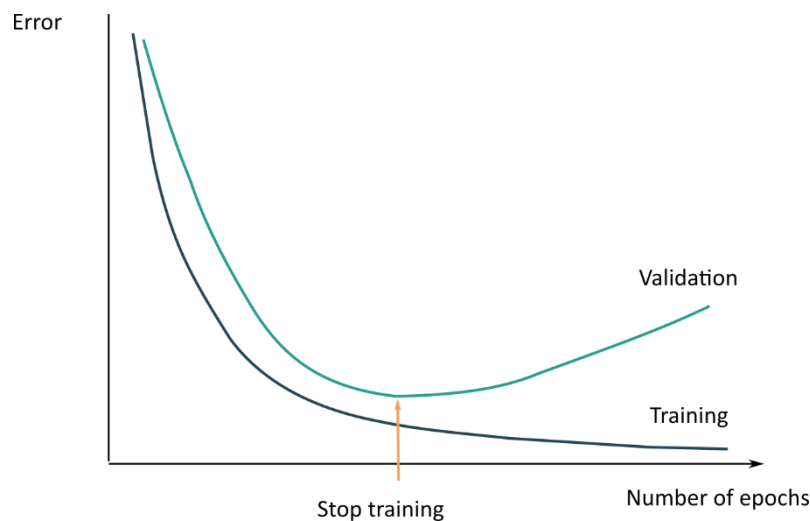


Figure 15: Early stopping principle. If the performance of the model on the validation dataset starts to degrade (e.g. loss begins to increase, or accuracy begins to decrease), then the training process is stopped. Early stopping could potentially improve generalization when other regularizers are absent.

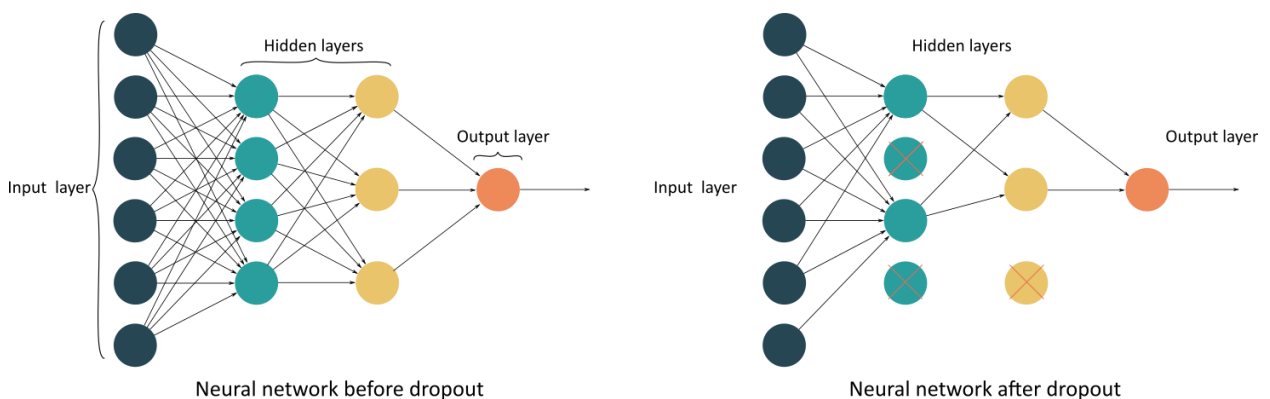


Figure 16: Dropout process. Dropout is a regularization method that approximates training a large neural network with different using different versions of the same neural network in parallel.

2.1.5 Learning rate

Learning rate (Figure 17) is a hyper-parameter that controls how much we are adjusting the weights of our network with respect to the gradient loss. We can also define it as the rate at which we descend towards the minima of the cost function. The lower the value, the slower we travel along the downward slope. We should choose the learning rate very carefully as very large value increase risks of missing the optimal solution, and very low ones result in time-consuming network convergence (Aggarwal, 2018).

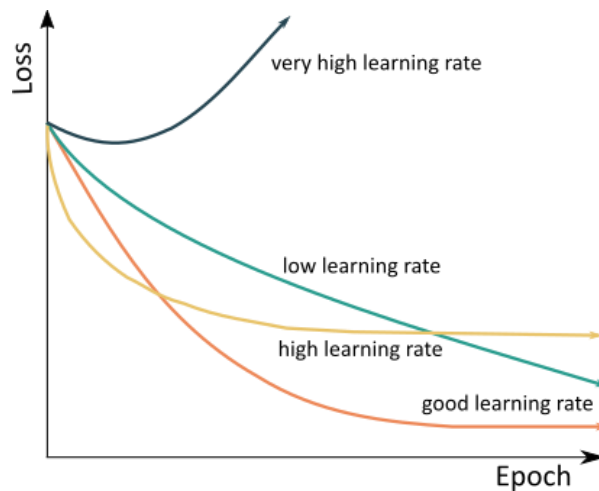


Figure 17: Learning rate diagram. The learning rate is a tuning parameter used in machine learning controls how quickly the model is adapted to the problem.

2.1.6 Cost / Loss

Loss functions are used to measure how well a network models the data. The goal is therefore to minimize its value as it computes the distance between the network output and the expected output derived from the training data (Cole Matt R., 2019). This means that if the loss function has a very high value, the predictions made by our model will be very different from the actual results (Aggarwal, 2018; Chollet, 2018). Loss functions are divided into two categories, regression loss like mean absolute error and classification loss like binary cross entropy. These loss functions are defined as follows:

- **Mean absolute error:** calculates the average of the absolute difference between the actual and predicted value.

$$L(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

- **Binary cross entropy:** It gives the probability value between 0 and 1 for a classification task. Cross-Entropy calculates the average difference between the predicted and actual probabilities.

$$L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^N y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

Some other examples of loss functions are Mean absolute errors (MAE) and categorical cross entropy.

2.1.7 Activation Functions

The activation functions (Figure 18), unique for each neuron, are used to normalize the neurons' output. These functions translate the input signals to output signals and constrain the range the output can take by applying a non-linear operation. For this, an artificial neuron calculates a "weighted sum" of its input by multiplying the inputs' features by the weights assigned to them and sends it to the output node. In many settings, the artificial neuron includes a variable called bias, which is usually added when the binary class distribution predicted by the neurons is highly imbalanced, and an invariant part of the prediction exists. This bias is incorporated using a bias neuron and allows to shift the activation function by adding a constant (i.e., the given bias) to the input (Aggarwal, 2018; Avinash Sharma V, 2017). The following equation describes the artificial neuron's output:

$$u(x) = \sum w * x + b$$

The most basic activation function is the identity or linear function, which is often used in the output node when the target is a real value. Sigmoid and hyperbolic activation functions were used early in the development of neural networks. The sigmoid function's output is a value between (0, 1). This activation function is usually used in computations that should be interpreted as probabilities. Although similar in shape to the sigmoid function, the hyperbolic tangent function is used when the outputs of the computations are desired to be both positive and negative; its output is a value between [-1, +1] (Aggarwal, 2018). Although sigmoid and tanh functions have been used to introduce non-linearity to the neural networks, in recent years ReLU (Rectified Linear Units) activation function, a piecewise linear function, has replaced them. Rectified linear units preserve many of the properties that make linear models easy to optimize with gradient-based methods and also preserve many properties that make linear models generalize well by learning complex relationships from the data. Also, ReLU adds more sensitivity to the weighted sum preventing neurons from getting saturated (Abien Fred Agarap, 2018; Aggarwal, 2018; Goodfellow et al., 2016).

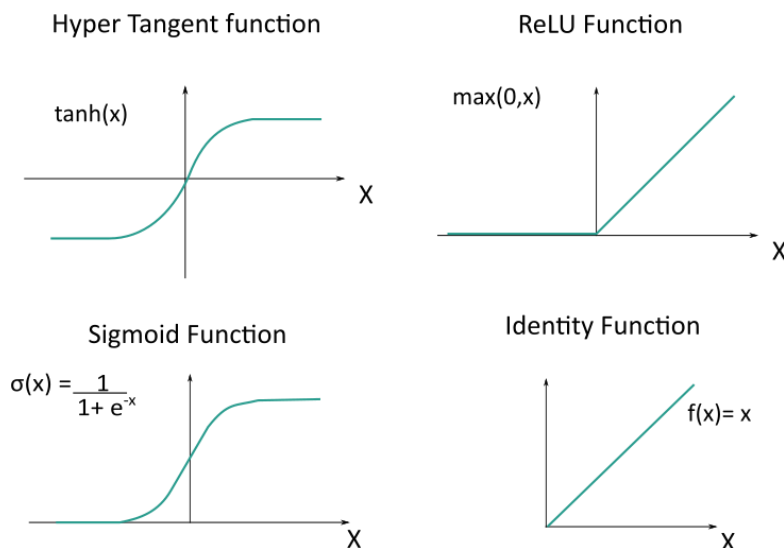


Figure 18: Activation functions diagrams. An activation function is a function that is added into an artificial neural network in order to help the network learn complex patterns in the data. ReLU is a widely used activation function, especially with Convolutional Neural networks.

2.1.8 Fine-tuning

Utilizing a pretrained network is an efficient technique for applying deep learning to small image datasets. A pretrained network is a network that was previously trained on a large dataset, typically on a large-scale image-classification task. Higher layers in the convolutional base encode more specialized characteristics, while lower layers encode more generic, reusable features. Because more specific features need to be applied to the new situation, it is more beneficial to fine-tune the more specific features. Thus, a good strategy to fine-tune only the top two or three layers in the convolutional base (Aggarwal, 2018; Chollet, 2018; Thakur, 2021). Fine-tuning process consists of the following steps:

- Pretrain a neural network model on a big dataset
- Create a new neural network model, i.e., the target model. This copies all model designs and their parameters on the source model except the output layer. It is assumed this model parameters contain the knowledge learned from the source dataset and this knowledge will also be applicable to the target dataset. Besides, is important to use a pretrained network which output is closely related to the labels of the source dataset.
- Add an output layer to the target model, whose number of outputs is the number of categories in the target dataset. Then randomly initialize the model parameters of this layer.
- Train the target model on the target dataset. The output layer will be trained from scratch, while the parameters of all the other layers are fine-tuned based on the parameters of the source model.

2.1.9 Evaluation methods

Evaluating the performance of deep learning models is essential to determine their accuracy. Some of the metrics commonly used are based on the number of true positives (TP), false positives (FP), and false negatives (FN). A true positive (TP) element represents an object correctly predicted to belong to a certain class (according to the target mask). A false negative (FN) is an object wrongly classified as not belonging to the given class, while a false positive (FP) is an object wrongly classified as belonging to a class. In the case of image segmentation, a TP is a predicted mask that overlaps a mask in the GT with an amount above a certain threshold; a FP is a predicted mask that have no correspondence in the GT or a correspondence that is below the set threshold; finally, FN are GT objects that do not have a corresponding object in the predicted mask.

The intersection over union, IoU (Figure 19), is a standard way to define if a detection has been well done. IoU is a coefficient of similarity between two datasets and measures the overlapping between the predicted objects and the ground truth objects (Padilla et al., 2020), classifying each object as TP, FP, or FN according to a previously set IoU threshold t . If $\text{IoU} \geq t$, then the prediction is considered as correct. On the contrary, if $\text{IoU} < t$, prediction is considered incorrect (Ong et al., 2019). Evaluating the performance of deep learning models is essential to determine their accuracy.

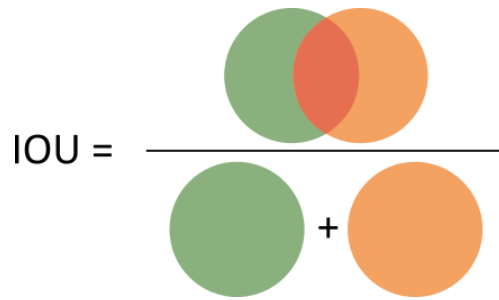


Figure 19: Intersection over union (IOU) diagram. This metric is known to be good for measuring the overlap between two bounding boxes or objects on a masks.

The assessment of many object detections and image segmentation models is mostly based on the precision P and recall R. Precision reflects the proportion of predicted objects correctly matching the ground truth. Recall, on the other hand, computes the proportion of ground truth being correctly identified in the predicted objects. Both range from 0 to 1, with higher scores indicating better results of the model (Manal El Aidouni, 2019; Padilla et al., 2020).

$$Precision = \frac{TP}{TP + FP} = \frac{True\ positives}{All\ objects\ detected}$$

$$Recall = \frac{TP}{TP + FN} = \frac{True\ positives}{All\ ground\ truth\ objects}$$

Another important metric is the F-measure better known as F-score. It is calculated from the recall and precision; it computes the overlap between predicted segmentation and ground truth. This metric is one of the most used to evaluate a neural network performance (Müller et al., 2022). The F-score is given by the following formula:

$$F_1 = 2 \frac{precision \cdot recall}{precision + recall} = \frac{tp}{tp + \frac{1}{2}(fp + fn)}$$

2.1.10 Deep Learning for nuclei segmentation

2.1.10.1 Image segmentation

In computer vision, image segmentation is the process of dividing a digital image into multiple regions based on the pixels' different properties to extract meaningful information for easy analysis (Sultana et al., 2020). Traditional segmentation methods frequently rely on hand-design chosen features, where researchers or users of such tools need to be actively involved in selecting those features, i.e., each method needs to be fine-tuned for each experiment. In contrast, deep learning-based methods require minimal input parameters from the user and do not require fine-tuning between experiments, which makes them more straightforward to apply than classical approaches (Hollandi et al., 2022). Furthermore, deep neural networks have proven to be capable of learning a hierarchy of increasingly complex features directly from in-domain data and perform pixel classification to assigning each pixel on the image a class, being able to distinguish objects from the background or one object from another one of the same kind (Havaei et al., 2017). These two different ways to classify pixels on an image are called semantic segmentation and instance segmentation; these concepts can be defined as:

- **Semantic segmentation:** is called semantic segmentation when each pixel is labeled as belonging to a class, for example, distinguishing between the background and cells' nucleus but without making any distinction among nuclei. In this case, a label will be assigned to the

background, usually zero, and another one to all the nuclei, usually one. Then, when our neural network performs the segmentation, it will assign the same labels to pixels belonging to the background and nuclei, respectively, giving, as a result an image with the same number labels as the ground truth images (Figure 20a).

- **Instance segmentation:** instance segmentation is one step ahead of semantic segmentation. In this case, instead of assigning the same label to all pixels belonging to the same type of objects, every object of the same class will receive a different label assigned. Therefore, each class instance on the image has a unique ID, and the resulting image will be an image where pixel boundaries separate all the objects. As a result, there will be a clear distinction between which pixels belong to an object and which pixels belong to the image's background but also, it will be clear boundaries between objects (Figure 20b).

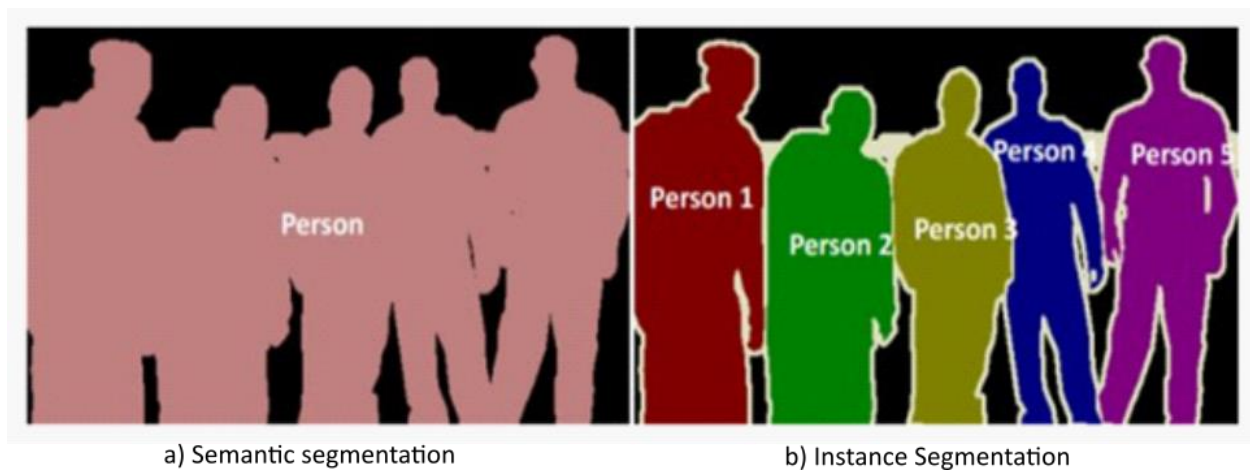


Figure 20: Semantic segmentation (left) vs Instance segmentation (right) diagram (Varatharasan et al., 2019). Semantic segmentation treats multiple objects within a single category as one entity in contrast, instance segmentation identifies individual objects within these categories.

Some of the most well know convolutional neural networks used for nuclei segmentation are Mask-RCNN (Kaiming He et al., 2017), U-Net (Ronneberger et al., 2015), and StarDist (Schmidt et al., 2018; Weigert et al., 2020). During the development of this project, I focused my attention on developing nuclei instance segmentation using StarDist 2D and StarDist 3D. Because of this, I will mostly describe these neural networks and their corresponding backbones, U-net and ResNet (K. He et al., 2016).

2.1.10.2 Deep learning algorithms for nuclei segmentation

2.1.10.2.1 U-net

U-net is one of the most used CNN to perform image segmentation and is focused on segmenting biomedical and biological microscopy images. U-Net has been applied to general pixel-classification tasks in 2D images and 3D image stacks with one or multiple channels. In addition, U-Net has been adapted to detect and segment arbitrary structures in biological tissue using the corresponding training data with precise segmentation results, showing its capabilities to outperform previous segmentation algorithms. Some of the tasks where U-net has been applied successfully include prediction of a single reference point per cell, cell nuclei segmentation, cell segmentation, and medical image segmentation (Falk et al., 2019; Siddique et al., 2021).

U-net takes its name from the shape of its architecture (Figure 21), consisting of a contracting path, encoder, and an expansive path, decoder. The contracting path (Figure 21, green side) follows the architecture of a traditional convolutional neural network. The U-Net encoder consists of the repeated

application of 3x3 unpadded convolutions, followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. Each of these downsampling steps will double the number of feature channels. On the other hand, the expansive path (Figure 21, orange side) consists of an upsampling of the feature map followed by a 2x2 convolution (“up-convolution”) that halves the number of feature channels, at the same time a concatenation with the correspondingly cropped feature map from the contracting path is performed. This concatenation is followed by two 3x3 convolutions, each followed by a ReLU, the result passes to the next decoder layer, and the process is repeated. At the final layer, a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. The upsampling path helps U-Net to propagate context information to higher resolution layers by combining high-resolution features from the decoding and the output of the encoder layers, creating a high-resolution segmentation map. It is important to mention that U-net neural network is a supervised deep learning algorithm whose output is smaller than its input due to the unpadded convolutions. It has also been proven that when basic data augmentations are implemented on the dataset, U-Net can be used with very few images and still have very high accuracy (Ronneberger et al., 2015).

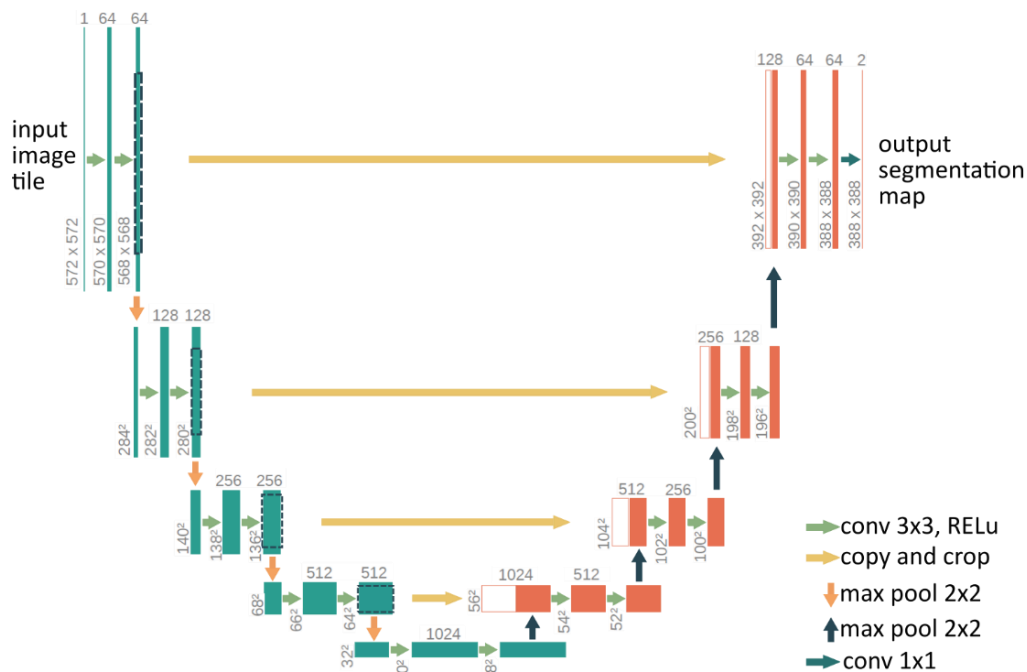


Figure 21:U-net architecture. U-Net is an architecture for semantic segmentation. U-net consists of a contracting path(green) and an expansive path(orange). Edited from (Ronneberger et al., 2015).

U-Net results can be extended from semantic to instance segmentation with some post-processing or using annotated images with three classes, background, edges, and nuclei, instead of two, background and nuclei. U-net is also known for performing very accurately even when using just a few images to train it (Ronneberger et al., 2015). Although U-net already offers good results, several papers have been published where U-net is the base of new segmentation algorithms. Some of the architectures developed using U-Net as a backbone are 3D U-Net (Çiçek et al., 2016), U-Net ++ (Zhou et al., 2018), U-NetPlus (Hasan & Linte, 2019), W-Net (Xia & Kulis, 2017), Cellpose (Stringer et al., 2020), and StarDist 2D (Schmidt et al., 2018) among others. Most of these new architectures are used to analyze and segment biomedical and biological images as U-Net (Siddique et al., 2021).

3D U-Net is an augmentation of the basic U-net framework that enables 3D volumetric segmentation, where all the 2D operations are replaced by 3D operations (Çiçek et al., 2016). U-net++ consists of an encoder and decoder connected through a series of nested dense convolutional blocks. The main idea

is to bridge the semantic gap between the feature maps of the encoder and decoder prior to fusion to propagate more semantic information between the two paths (Zhou et al., 2018). U-NetPlus uses a pre-trained encoder for the downsampling paths and a re-design of the U-net upsampling path where transposed convolutions are replaced by an upsampling operation based on nearest-neighbor (NN) interpolation (Hasan & Linte, 2019). The W-Net architecture consists of a U-Net-base encoder and U-Net-base decoder that reconstructs original input images and predicts the segmentation maps without labeling information (Xia & Kulis, 2017). Cellpose architecture replaces the feature concatenation on the upsampling path in U-net architecture with direct summation to reduce the number of parameters. Also, it replaces the standard building blocks of a U-net with residual blocks. Cellpose architecture also doubles the depth of the network as typically done for residual networks. In addition, it uses global average pooling on the smallest convolutional maps to obtain a representation of the “style” of the image. (Stringer et al., 2020). StarDist neural network is the algorithm I used to develop this project, and it will be described in detail in the next section.

2.1.10.2.2 ResNet

Residual Network (ResNet) is one of the most popular deep learning algorithms; it is used in several computer vision tasks because it is easy to train, although it can be a very deep neural network. ResNet addresses the degradation converging problem caused when the depth of a neuronal network increases. This means that: with the network depth increasing, accuracy gets saturated and degrades rapidly. This neural network introduces deep residual learning framework; this type of neural network uses shortcut connections that enable the flow of information across layers without attenuation that would be caused by multiple stacked non-linear transformations, resulting in improved optimization (K. He et al., 2016; Targ et al., 2016).

To clearly define ResNet first, we need to define residual learning. Consider $H(x)$ as an underlying mapping to be fit by a few stacked layers, where x represents the inputs to the first of these layers. We can say that multiple nonlinear layers can asymptotically approximate complicated functions, which is equivalent to saying that these layers can also asymptotically approximate a residual function, $H(x)-x$. Meaning that, rather than using stacked layers to approximate $H(x)$, we explicitly let these layers approximate a residual function $F(x): = H(x)-x$, causing $H(x)$ to become $F(x) + x$. Residuals connect layers that are not neighbored in chain-like neural networks. These new constructions break the convention of stacking layers to build a chain-like neural network. They introduce loops into neural networks, which were previously chain-like (F. He et al., 2019; K. He et al., 2016). The reformulation of this convolutional neural network architecture was done to assess the degradation problem presented when the depth of a neural network increases. The residual learning function is presented in every few layers of the neural network, which building block can be seen in Figure 22; this building block can be represented by the equation $y = F(x, \{W_i\}) + x$, where $F(x, \{W_i\})$ represents the residual mapping to be learned and can represent multiple convolutional layers. The operation $F + x$ is performed by a shortcut connection and element-wise addition. The shortcut connections perform identity mapping, and their outputs are added to the outputs of the stacked layers. This shortcut does not add any extra parameter or computation complexity to the neural network, which makes ResNet stand out from other algorithms (K. He et al., 2016).

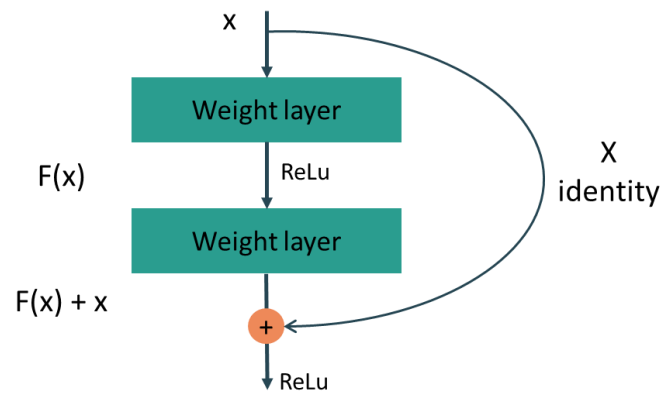


Figure 22: Residual learning building block. In this block a layer $\ell - 1$ is skipped over activation from $\ell - 2$. Edited from (K. He et al., 2016).

ResNet base-line is inspired by VGG nets (Simonyan, K., et al 2014) where the convolutional layers mostly have 3×3 filters and follow the next design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer. The downsampling is performed by convolutional layers that have a stride =2. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. By inserting shortcut connections into the VGG net we turn the network into its counterpart residual version, i.e., ResNet (K. He et al., 2016). ResNet architecture can be seen in Figure 23

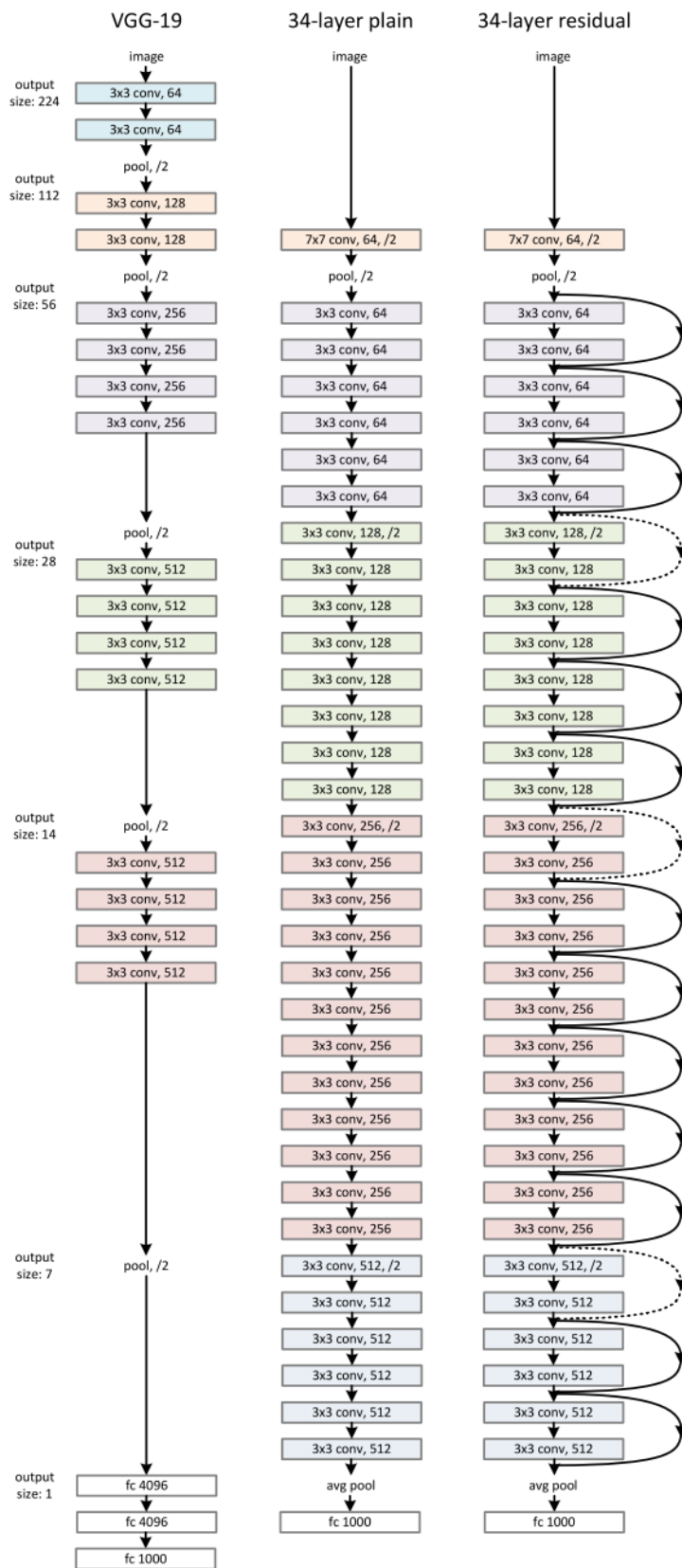


Figure 23: Example network architectures for ImageNet. Left: the VGG-19 model as a reference. Middle: a plain network with 34 parameter layers (3.6 billion FLOPs). Right: a residual network with 34 parameter layers. The dotted shortcuts increase dimensions. Taken from (K. He et al., 2016) .

2.1.10.2.3 StarDist 2D and StarDist 3D

StarDist 2D and StarDist 3D are instance segmentation methods based on U-Net. StarDist is a top-down segmentation method in which individual cell instances are localized first using a rough shape representation, and then the initial shape is refined in an additional step. Contrary to other image segmentation methods that use axis-aligned bounding boxes as nuclei shape representation, StarDist predicts cell nuclei shape using star convex polygons and star-convex polyhedral for 2D and 3D objects, respectively (Figure 25 a) (Schmidt et al., 2018). A *star-shaped polygon* (Figure 24) is a polygon that contains a point z from which all points of the polygon boundary is visible (Ghosh, 2007) A convex polygon is a polygon in which all interior angles are less than 180° and all vertices 'point outwards'; all convex polygons are star-shaped (Wolfram Research, Inc, 2022; Wolfram Research, Inc., 2022). Star-convex polyhedrons are the 3D version of the star-convex polygons. This method to predict a cell nuclei shape allows StarDist to predict the shape representation of cell nuclei without the need to perform a refinement since star-convex polygons are well-suited to approximate the typically roundish shapes of cell nuclei in microscopy images.

To predict the shapes of the nuclei, StarDist 2D predicts a star-convex polygon for every pixel. Then, along a set of n predefined radial directions with equidistant angles, it regresses the distances, $r_{i,j}^k$, of each pixel belonging to an object to its boundary (Figure 25 b). Separately, it predicts the probability of every pixel to belong to an object to only consider polygon proposals from pixels with sufficiently high probabilities (Figure 25 b). Then, the pixel's object probability, $d_{i,j}$, is computed using the normalized Euclidian distance to the nearest background pixel. After, Non-maximum suppression (NMS) is used to output the final set of polygons, each representing an individual object instance (Schmidt et al., 2018). Non-maximum suppression (NMS) is a post-processing algorithm that merges all detections belonging to the same object. NMS algorithm selects high-scoring detections and deletes close-by less confident neighbors since they are likely to cover the same object (Hosang et al., 2017).

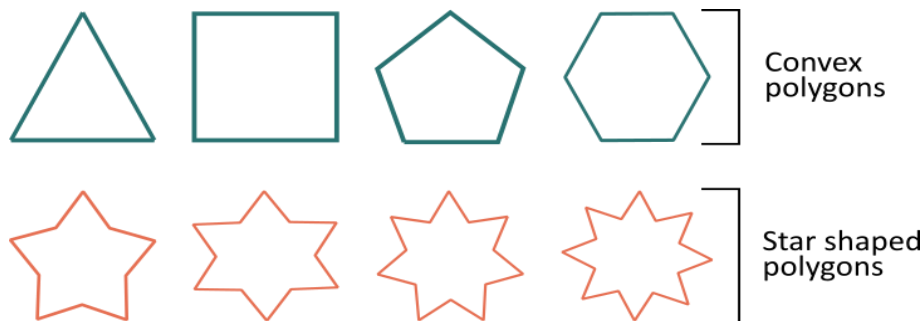


Figure 24: Examples of convex and star-shaped polygons. Convex polygons are star-shaped.

The main advantages of using StarDist 2D algorithms to perform cell nuclei segmentation are its capacity to accurately segment merged cells and its ability to perform well with very crowded nuclei. Besides, StarDist 2D can predict a reasonable complete shape from only partially visible nuclei at the image boundary.

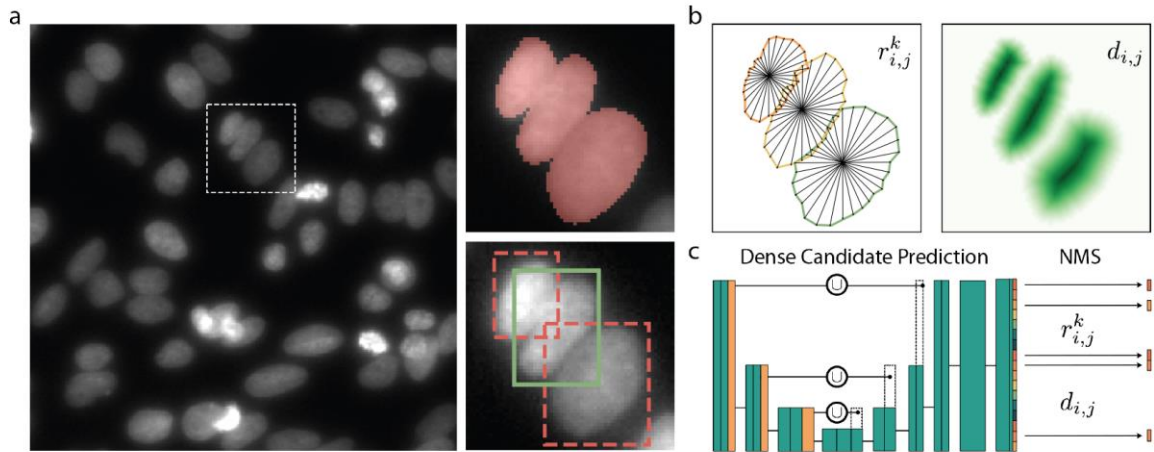


Figure 25: Architecture of StarDist 2D (Schmidt et al., 2018). a) comparison between pixel classification and object detection using bounding boxes. b) StarDist segmentation methodology. c) StarDist neural network.

In the case of StarDist 3D, the shape of cell nuclei is described using a star-convex polyhedron, which computation is similar to the star-shape polygons of the 2D case. Noticeably, an ellipsoid representing the nuclei shape is computed using an evenly distributed set of n predefined unit rays (Weigert et al., 2020). To take into consideration the anisotropy of the images, StarDist 3D generates an anisotropically scaled vector, $\vec{u}_k = \left(\frac{x_k}{s_x}, \frac{y_k}{s_y}, \frac{z_k}{s_k} \right)$, where $\vec{s} = (s_x, s_y, s_k)$ is the anisotropy factor calculated as the median bounding box size of all objects in the training images, and x_k, y_k , and z_k are points over a spherical Fibonacci lattice (Figure 27). Therefore, the ellipsoids unit rays used to define the shape of the polyhedrons are $\vec{r}_k = \frac{\vec{u}_k}{|\vec{u}_k|}$. Finally, the surface of the star-convex polyhedrons represented by the distances $d_{i,j}$ is then given by its vertices $d_k \cdot \vec{r}_k$ and the final convex hull determine during the NMS process (Figure 26c) (Weigert et al., 2020).

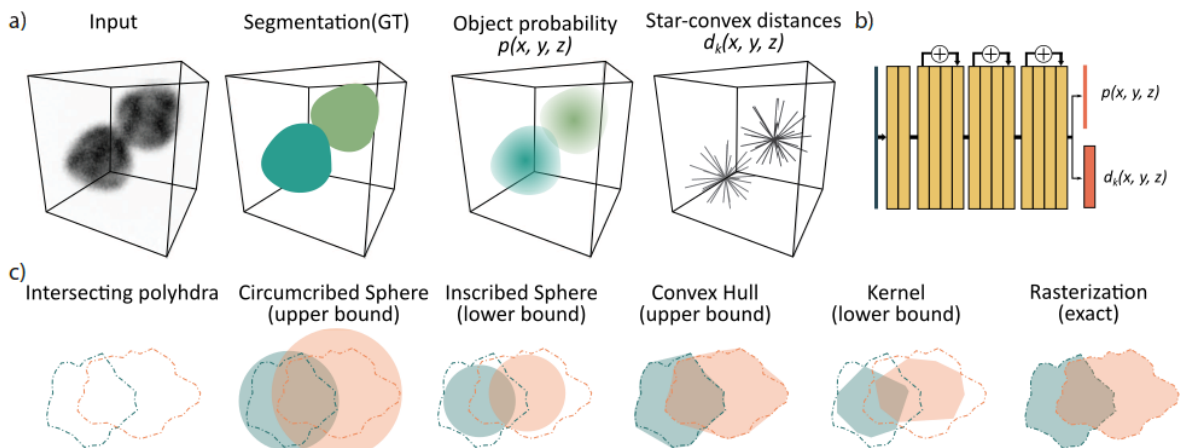


Figure 26: a) STARDIST-3D method is trained to densely predict object probabilities p and radial distances d_k to object boundaries. b) Schematic of STARDIST-3D architecture based on ResNet. c) During non-maximum suppression process (Weigert et al., 2020).

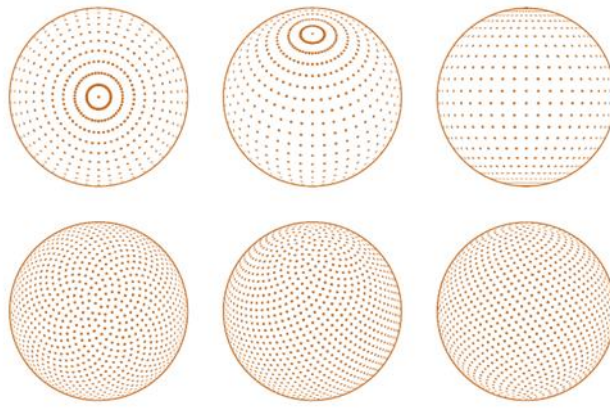


Figure 27: Latitude–longitude lattice (top) and Fibonacci lattice (bottom). In the Fibonacci lattice, the points are much more evenly spaced, and the axial anisotropy is much smaller in comparison with the latitude–longitude lattice. Edited from (González, 2010).

2.1.11 Advantages and challenges of the use of Deep Learning algorithms for nuclei segmentation

Many image analysis workflows like counting cells, tracking moving populations, localizing proteins, and classifying phenotypes involve the segmentation of cell nuclei as a first step to extract meaningful biological information. Because of this, researchers have focused their efforts on designing and improving nuclear and cellular segmentation methods. The use of deep learning in the nuclei segmentation task has proven to overcome traditional image segmentation methods and has shown an accuracy closer to human expertise. In addition, these approaches can be more generalized, robust, and fully automated, allowing the analysis of a high number of images in a shorter period of time (Caicedo et al., 2019).

The most widely used deep learning algorithm in nuclei image segmentation is U-Net, as it is also used in various U-net-based architecture methods. Similarly, other neural networks such as Mask-RCNN and ResNet have been adapted to perform this task.

Although U-net and Mask-RCNN are the main benchmarking neural networks, there exist other algorithms developed to perform nuclei segmentation. For example, R2U-Net, a recurrent residual U-net, uses recurrent residual modules instead of feedforward convolutional layers resulting in a network that combines ResNet and U-net (Alom et al., 2018). Methods like the one presented in (Narotamo, 2019), where U-net is combined with YOLO (Redmon et al., 2016), combines the computational efficiency of YOLO's object detection with U-net's powerful segmentation generating a nuclei segmentation method powerful and computationally efficient. Furthermore, region proposal-based neural networks like Mask-RCNN and YOLO, when combined with U-net, help to remove duplicate bounding box proposals while the segmentation module learns contextual information from the predictions to improve instance differentiation (Mela & Liu, 2021). (Vuola et al., 2019) present a segmentation method that combines Mask-RCNN and U-net, and instead of using weighted border pixels as the original U-net, it adds extra output channels that predict borders between neighboring nuclei. Other nuclei segmentation methods like the one presented in (Guerrero-Pena et al., 2018) and (Van Valen et al., 2016) take advantage of multiclass weighted loss function for segmenting individual touching cells in cluttered regions, where the latter uses the segmentation of the cell nuclei to seed a refinement of the cellular interior segmentation using active contours. Finally, in (Naylor et al., 2019), they focus the attention of the neural network on the center of the nuclei and generate an erode segmentation using distance maps.

As we can see, several methods exist to perform nuclei segmentation, each of which approaches the problem differently and offers different advantages such as more computational efficiency or better capability of segmenting touching objects. By taking account of the axial anisotropy and by being capable of segmenting images with low SNR, I considered that StarDist 3D was the best choice to achieve the objectives proposed at the beginning of this project. However, and as it was emphasized in most of the presented papers, there is a lack of ground truth data for training 3D deep learning-based methods. I tackled this problem in the next section of this manuscript.

3 | Annotation process for ground truth generation

Main contributions presented in this Section:

- A **ground truth dataset** of 7 organoids manually annotated representing more than 4,500 individual 3D nuclei.
- A **Napari plugin** designed to facilitate the 3D annotation of objects starting from 2D masks.

Nuclei segmentation is an important task in cell analysis that requires accurate and reliable segmentation methods. One of its biggest challenges is segmenting densely packed cell nuclei when many nuclei are touching each other in the x, y and z direction. Unfortunately, this step is still a bottleneck nowadays, mainly caused by its complexity and the speed difference between the data generation and analysis. Indeed, model-based methods struggle both in analyzing very complex data and in generalizing for different conditions. On the contrary, deep learning approaches, and especially the supervised ones, can succeed in both venues at the expense of having enough data to train with. As a result, availability of ground truth data has become the main bottleneck for data-based methods (Behl et al., 2020; Chen et al., 2018).

A high-quality ground truth dataset is crucial to the success of any supervised deep learning algorithm. Consequently, its creation is time-consuming, challenging, and error-prone because it relies on human efforts. Hopefully, there exist open-source datasets that can be used to train and test segmentation algorithms (Broad Institute, 2021; Caicedo et al., 2019; Kai & Kaizhu, 2021). While these datasets are an invaluable resource for the community, with tens of thousands of manually annotated nuclei across different samples, conditions, and microscopes, they are mostly limited to 2D. Of all those, only one stack is fully annotated in 3D, roughly corresponding to 800 nuclei (Kai & Kaizhu, 2021). Unfortunately, this one stack is not sufficient to robustly train a segmentation network, resulting in a dire need from the community for more high-quality annotations of nuclei in 3D.

In this section, I will explain how I improved my 3D annotation pipeline from a basic manual process to a semi-automatic one by way of developing an annotation plugin for Napari called **Napari annotation helper (NAHP)**.

3.1.1 High-content imaging of organoids with the soSPIM system

In recent years, the popularity of 3D cell cultures has exploded due to their ability to offer valuable models to study human biology, far more physiologically relevant than 2D cultures (Jensen & Teng, 2020; Kapałczyńska et al., 2016). As a result, an increasing number of 3D organotypic cultures (organoids) are being developed to mimic intestines, liver, brain, kidney, lung, and many other organs. In addition, 3D cell cultures allow the design of tissues and organs with essential structural and physiological features in a controlled manner, helping to reduce the time and cost of drug development (Sakalem et al., 2021; Shao et al., 2020). Nevertheless, using 3D cultures to their full extent has

remained elusive, as it demands new approaches such as fast, automated multi-scale imaging to quantify and control organoid diversity.

In collaboration with the team of V. Viasnoff at the Mechanobiology Institute of the National University of Singapore, our team has expanded the soSPIM to tackle this issue. This new imaging platform is capable of streamlining organoid culture with fast light-sheet 3D imaging and can perform 3D imaging of up to 300 organoids in 1 hour with subcellular resolution (Beghin et al., 2022). To briefly summarize, we have designed new miniaturized 3D cell culture devices, called JeWell chips, that comprise thousands of well-arrayed microwells with pyramidal shapes flanked with 45° mirrors (JeWells) to enable soSPIM imaging. An insight into the image acquisition process can be seen in Figure 28.

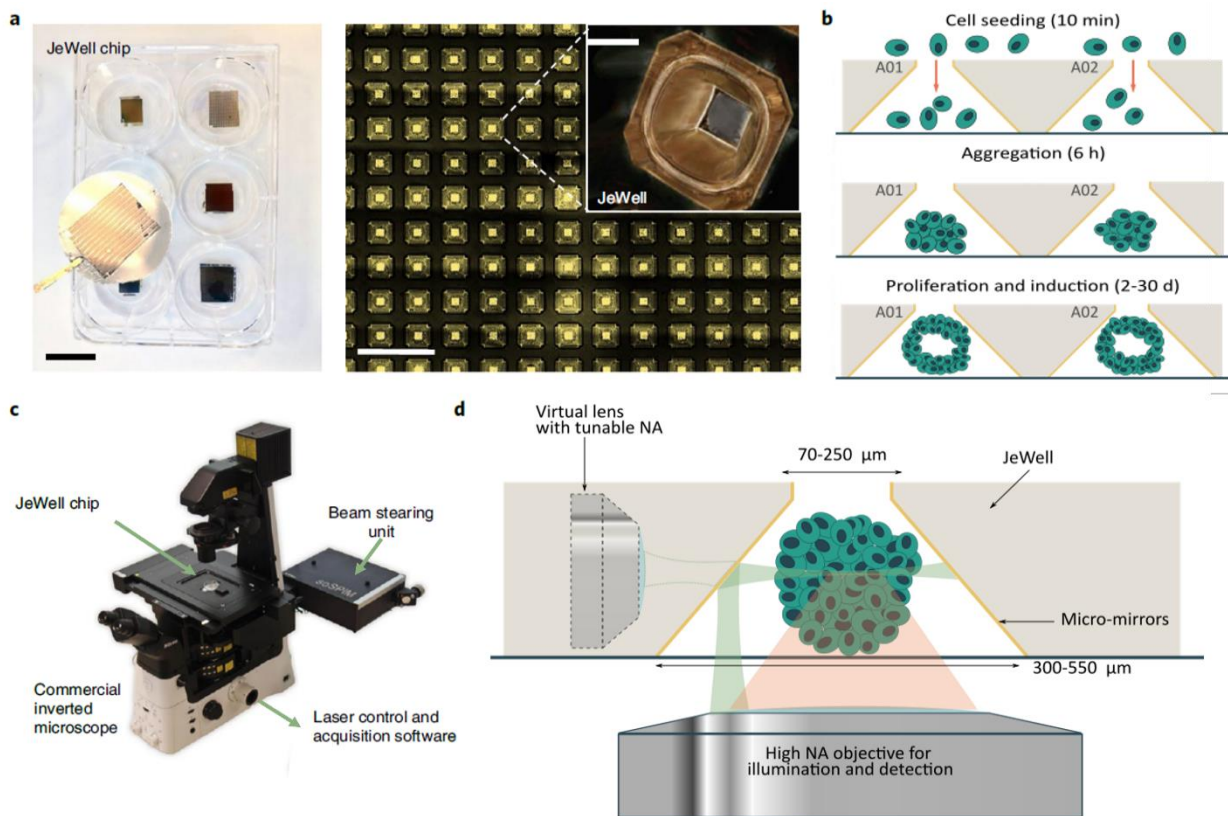
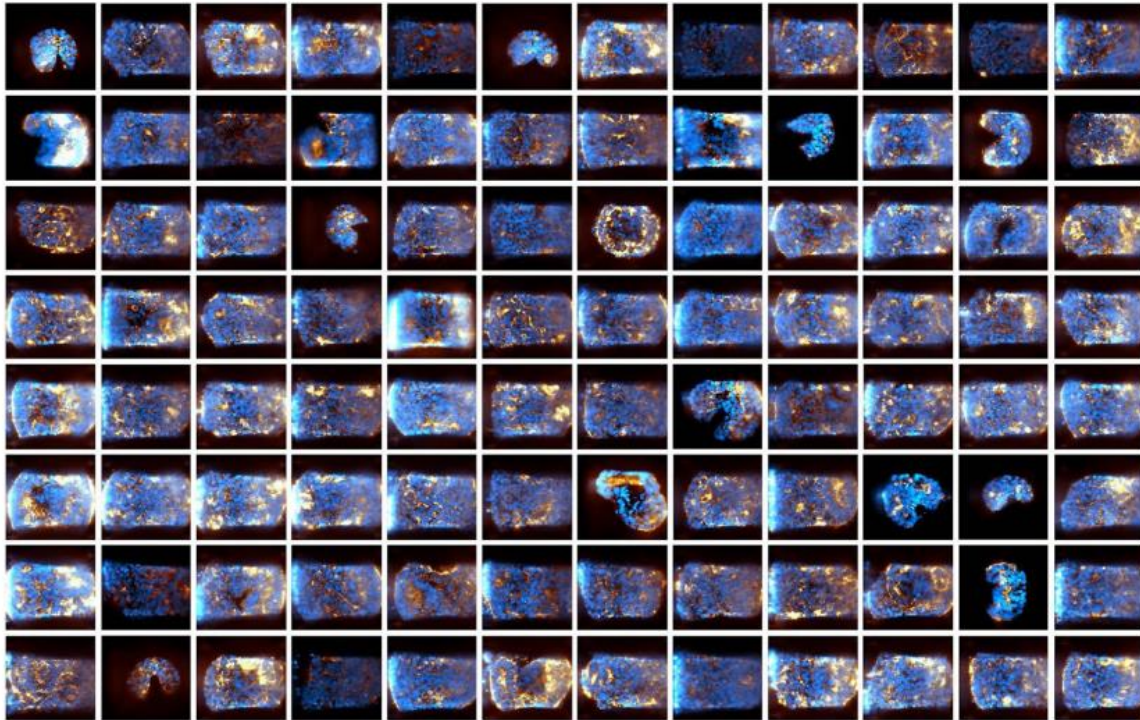


Figure 28: a, JeWell chips in a six-well dish (left) and close-up image of the JeWells array with a density of 16 JeWells per mm² (right). Inset shows a zoom on a JeWell inverted pyramidal microcavities flanked with four 45° mirroring surfaces. Scale bars, 3 cm (left), 500 μm (right) and 70 μm (inset). b, Schematic of the seeding procedure. c, Photograph of the imaging setup comprising a commercial inverted microscope, combined with JeWell chips, a laser scanning unit, and its custom-made control software. d, Principles of the soSPIM. Edited from (Beghin et al., 2022).

The JeWell's geometry offers several advantages: (1) it ensures that only one single organoid grows in each well, (2) it standardizes the organoid culture and, (3) it can be used as a proxy to automatically align and calibrate the light sheet and to correct for drifts, automatizing imaging of the samples. We tested our platform on various organoids and tumoroids (hepatocytes, neuroectoderm, intestinal, HCT116). As an illustration, Figure 29 depicts a collection of 96 neuroectoderm organoids differentiated from hESCs, fixed at day 8 and immunostained for nuclei (DAPI, 405 nm), Sox2 (488 nm) and N-cadherin (actin, 647 nm) inside a JeWells chip. We acquire 96 organoids with three wavelengths in 1 hour, representing 173 GB of raw data, an amount of data inconceivable to manually analyze. In this context, my work was to develop a robust and automatic pipeline for the 3D segmentation of the organoids' nuclei.



96 organoids, 70 z planes each, 3 wavelengths, 173 GB of (2,048 × 2,048 × 70) stacks, 1 h acquisition time

Figure 29: Representative gallery of 96 neuroectoderm organoids from a library of >400 organoids (median plane of the 3D stacks) labeled with actin (gold), Sox2 (magenta) and DAPI (blue) acquired in an automated workflow in less than 1 hour. Edited from (Beghin et al., 2022).

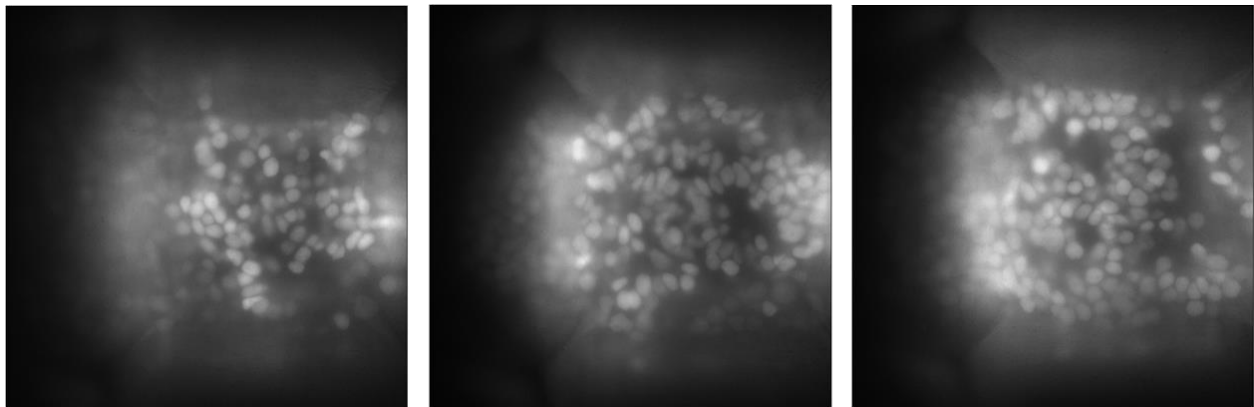


Figure 30: Examples of the images taken using SOX2 immunostaining and DAPI labeling by the team of Virgile Viasnoff at the Mechanobiology Institute of the National University of Singapore.

3.2 Annotation process

A large amount of data needs to be collected and labeled to train and test supervised neural networks. The first step of my work was to generate a corpus of ground truth data necessary to train a high-quality StarDist (Schmidt et al., 2018) model as we can observe in 2D StarDist segmentation results using original-DAPI and DSB2018 datasets section. Therefore, I had to select a subset of representative image stacks from the dozens that were already acquired. From all the conditions and 3D culture models that were imaged, I determined that I should only focus on organoids labeled with DAPI and Sox2, as DAPI stains nuclei and Sox2 is an early development marker present in differentiated nuclei (Kapuscinski, 1995; Sarkar & Hochedlinger, 2013). I also decided that different types of 3D cell cultures,

i.e., neuroectoderm organoids and cancer spheroids, had to be included in the corpus for a better model generalization. I first selected 3 image stacks from a batch of neuroectoderm organoids that exhibited interesting morphologies, as they were all labeled with both DAPI and Sox2 as can be seen in Figure 30 and the ones label with just DAPI can be seen in Figure 31. Depending on the samples, these images may have been cropped in the x, y and z directions to avoid mirror reflections and frames in which the signal was insufficient to distinguish any nuclei. I then started to annotate them manually.

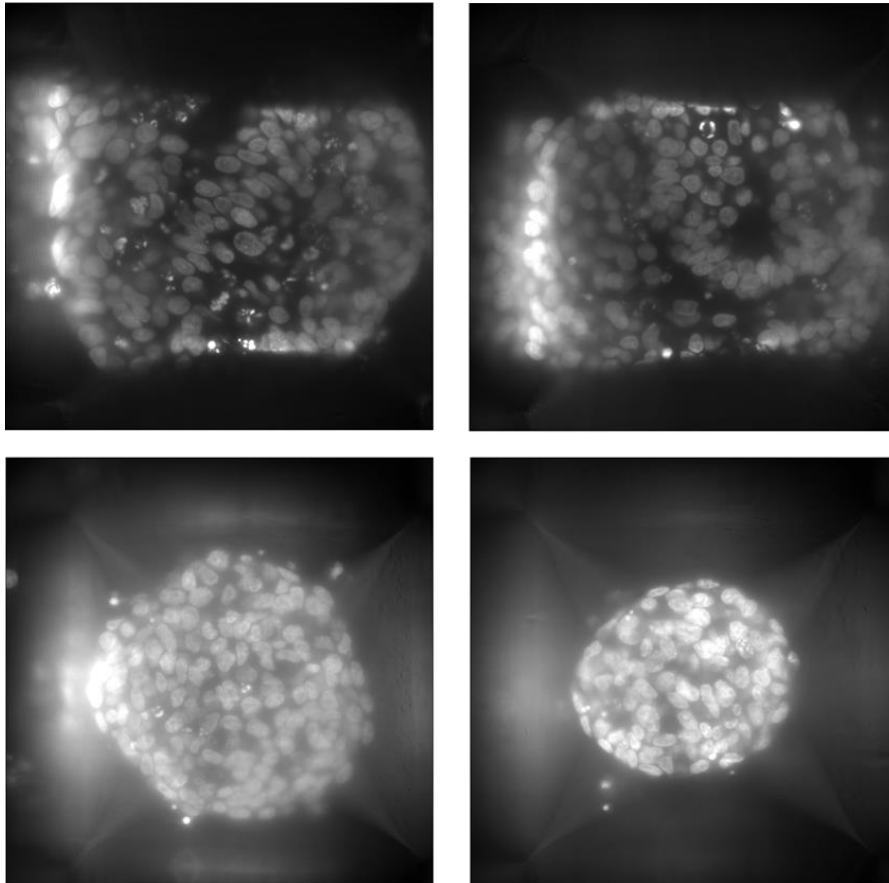


Figure 31: Examples of the images taken using DAPI labeling by the team of Virgile Viasnoff at the Mechanobiology Institute, National University of Singapore.

3.2.1 From manual annotation

Manual annotation only requires 3 components: (1) an image to annotate, (2) a software that can open the image and handle the creation and transfer of regions of interest (ROIs) into another image, and (3) a user that will delineate the different objects with ROIs. As I wanted to annotate nuclei in 3D, one obvious improvement to prevent a full manual annotation would have been to pre-process the image stacks to segment the nuclei in 2D, and then connect the 2D labels to create 3D labels.

There exists software that performs semi-automatic annotation using machine learning with a widely varying spectrum of functionalities, such as Ilastik (Berg et al., 2019), QuPath (Bankhead et al., 2017), LabKit (Arzt et al., 2022), and DeepCell Label (*DeepCell Label*, 2016). Ilastik offers well-established machine learning-based image processing tasks, allowing for a wide range of applicability. Ilastik gives people with no prior knowledge of machine learning the ability to apply any of the workflows available, such as pixel or object classification workflows (Berg et al., 2019). On the other hand, QuPath can be used for a variety of image analysis applications. It incorporates extensive annotation and visualization

tools using building blocks to create custom workflows, linking these together for batch processing with powerful scripting functionality (Bankhead et al., 2017). LabKit is a Fiji plugin for the segmentation of microscopy images offering easy to use manual and automated image segmentation routines that can be rapidly applied to single- and multi-channel images. As well as Ilastik, LabKit allows the user to segment their images by manually drawing dense labels on the entire image using random forest to perform pixel classification (Arzt et al., 2022). DeepCell Label is an instance labels annotation tool able to annotate 2D, 3D, and timelapse images (*DeepCell Label*, 2016). Other annotation tools can be found in (Hollandi et al., 2022).

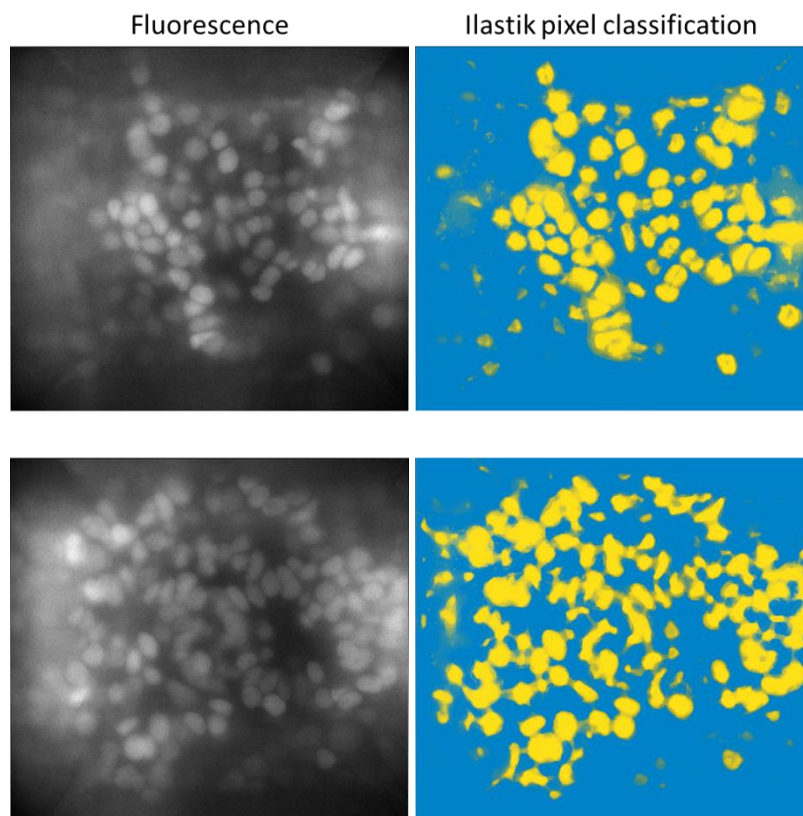


Figure 32: Results of the segmentation done using Ilastik. Ilastik software was used hoping to speed up the labeling process but due to its unsatisfactory performance this idea was discarded.

Even though a lot of those software allows semi-automatic annotation, it is not always straightforward to apply them to the images used during this project. The combination of uneven illumination, similarity between nuclei and background intensity values, high nuclei density, and presence of both in-focus and out-of-focus objects in the same images makes the annotation tools unable to properly segment them (Figure 32). Besides, the size of the images can also generate memory allocation problems, as when using Ilastik and QuPath, further delaying the task. Unfortunately, using the pre-trained StarDist 2D model also resulted in very poor segmentations (see StarDist 2D segmentation results using the original -DAPI dataset section for a detailed explanation). Because of this, it was necessary to resort to manual annotation to create a ground truth to train and test StarDist and obtain accurate results.

While FIJI (Schindelin et al., 2012) seemed to be an ideal choice for annotating images, I rapidly realized it was more adapted for creating semantic segmentation annotations. Indeed, modifying the ROIs' label on the fly is complex, an important limitation when annotating touching and overlapping objects. For this reason, I decided to use Napari (Sofroniew & Lambert, 2019), where I could directly create instance annotation. Napari is designed for browsing, annotating, and analyzing large multi-

dimensional images (Sofroniew & Lambert, 2019). Thanks to its label layer, I could easily switch between different types of annotated regions, such as polygons, dots, or free-hand drawings. In the end, creating a label falls down to selecting a label value and painting the shape of the nucleus directly on the image. My annotation process was thus twofold.

First, I manually labelled all the nuclei in 2D on all the frames of the image stack. Then, I reconnected all the 2D labels of each individual nucleus to create 3D labels. This means that all these 2D labels, for an individual nucleus, were given the same unique value. It is essential to mention that relabeling does not need to be sequential, as it does not affect the segmentation result. However, for convenience, I started with 1 and made it sequential.

Even if close to optimal from a quality standpoint, manual annotation has more downsides than upsides. Among its limitations, the most noticeable are that it is very time-consuming and that it requires significant effort to produce only a limited amount of data when images are too crowded or have very challenging lighting or SNR conditions. In my case, one stack's frame took around 1 hour to be fully annotated, equivalent to approximately two full working weeks to annotate a 70-frame stack. Importantly, I had access to a WACOM CINTIQ model DTH-W1310 tablet to accelerate the annotation process. Annotating with a simple mouse would have been even longer and more tiring.

Unfortunately, two weeks per stack adds up to several months for a ground truth corpus of critical size. Being able to speed-up this process was therefore critical. Because of this, searching for methods to speed up the process was necessary.

3.2.2 To semiautomatic annotation

While still insufficient to properly train StarDist in 3D, these 3 annotated stacks represented thousands of 2D annotations. I realized that I could thus resort back to pre-processing my image stacks to segment the nuclei in 2D. Combined with the open-source ground truth datasets presented above, I was able to train a high-quality model for StarDist in 2D (see Section StarDist 2D segmentation results using the original -DAPI and DSB2018 for more information). I then chose 4 new stacks labeled with DAPI (Figure 31), 1 neuroectoderm organoid and 3 cancer spheroids, and run the trained StarDist 2D model to segment all the nuclei in 2D. After that, I uploaded these images to Napari to transform the 2D segmentation into 3D annotated images. The use of the graphic tablet again facilitated this process. In Figure 33 we can observe the difference between labels done in 2D and 3D, where each color represents a different label number.

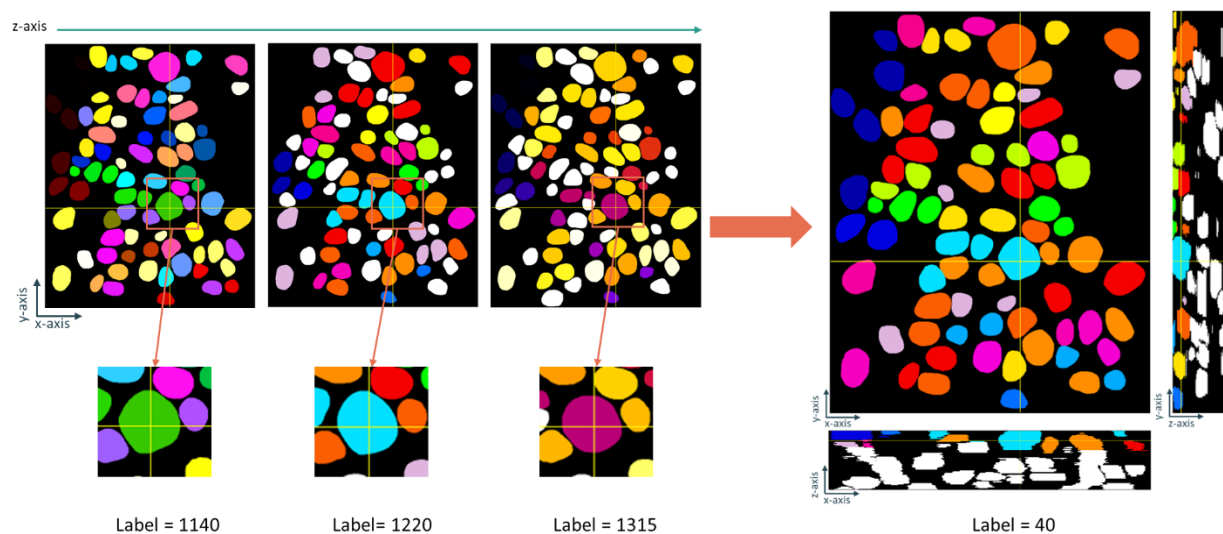


Figure 33: Semi-annotation process diagram. 2D labels are distinguish by different color. However, in the Fiji orthogonal view, some of the 3D nuclei annotation can be distinguishable by its color, meanwhile due to the bright-contrast configuration some are display in white although they have different values.

While simple, this pipeline modification helped diminish the time spent for annotating by an impressive half as I only needed 1 week to annotate an image stack versus 2 weeks with a full manual process. Nevertheless, the 2D segmentations still required some corrections, whether it be annotating missing nuclei, erasing segmentation errors or correcting the object boundaries if necessary. Unfortunately, Napari was far from perfect for this task, and the gain in time could therefore have been even higher with a dedicated tool.

3.2.3 Annotation results

At the end of this process, I fully annotated 7 stacks, roughly summing up to around 4,700 3D nuclei. Examples of those annotations are shown in Figure 34. At that time, this amount was sufficient to train a high-quality StarDist model in 3D. In the next chapter, I will present the resulting quantifications in the context of the collaboration with Singapore.

While I was finished with the annotations for the Singapore collaboration, I realized that the team was on the verge of acquiring new organoids and spheroids models. As it meant the necessity to perform some new annotations, I decided to use my experience to develop a Napari plugin designed to help with reconnecting 2D labels to 3D ones.

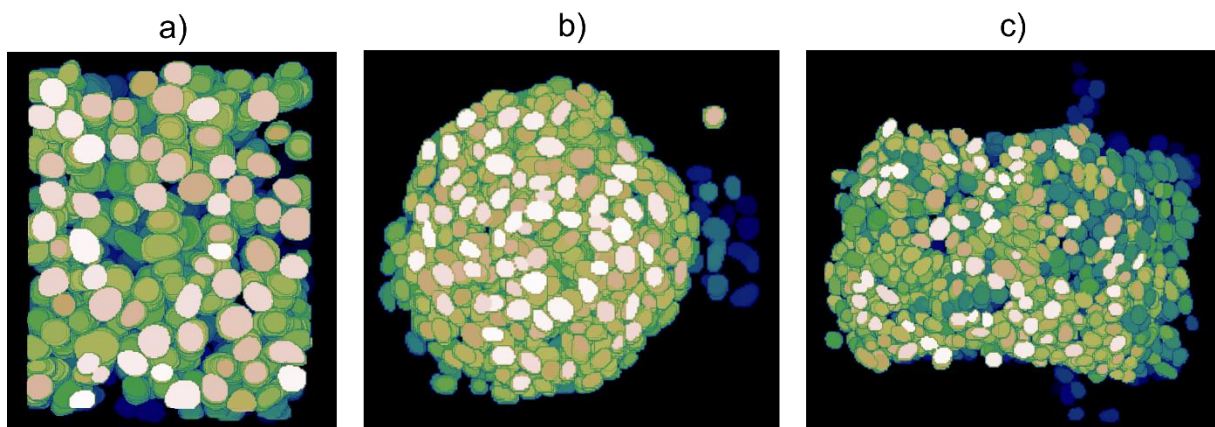


Figure 34: 3D Manual and Semi-automatic annotation results, where a) is a neuroectoderm organoid labeled manually, b) a cancer spheroid labeled using a semi-automatic process and c) is a neuroectoderm organoid labeled using a semi-automatic process.

3.3 Napari annotation helper (NAHP)

Napari Annotation Helper or NAHP is a python-developed Napari plugin using magicgui (Magicgui, 2020) and PyQt5 (Riverbank Computing, n.d.) python packages. NAHP's objective is to assist users in performing annotation faster and more easily by providing a set of functions that will help them avoid annotation errors, object-label mismatch, label repetition, and confusion when tracking the labels added to an image.

Noticeably, NAHP uses 2D segmented image stacks (preferably with instance labels) as input to generate 3D annotations. As this 2D segmentation step is not incorporated into NAHP, users are free to use any segmentation method they want as a starting point. Typically, I used my trained StarDist 2D model to get these 2D segmented image stacks.

NAHP consists of three main sections, first, I will address the labels loading widget that contains color-blind inclusive features helping users to distinguish 3D labels from previously segmented 2D labels.

Then, I will explain the functions developed to help with the annotation process. Finally, I will present another set of functions designed to extract new quantitative measurements from the labels.

3.3.1 Labels layer features

NAHP requires to load two images, the one to annotate and the one containing the previously 2D segmentations. The image loading functions are available through two widgets called "Fluorescence" and "Labels".

As both 2D and 3D labels are identified and differentiated by integer values, it is critical to be able to distinguish them. Based on the safe assumption that there will not be more than a few thousands individual nuclei in the organoids we are acquiring, I decided to add 10,000 to each label value of the 2D segmentations. This addition results in having labels greater than 10,000 for all 2D labels, while the 3D labels will begin at 1.

Using this trick also allowed me to visually separate 2D and 3D labels. Accessible through a drop-down button, the user can choose one base color (red, blue, magenta, orange, yellow, or white) that will be used for all the 2D labels. Simultaneously, a look-up table (LUT) is created which purpose is to generate colors that are visually as far away from this chosen base color. Thanks to it, every new 3D label will have a different color that will be distinguishable from the 2D segmentation. In spirit of inclusion, I also added the possibility to generate a color-blind LUT. Figure 35 shows the fluorescence image and labels widgets. In the following sections, I will discuss in depth how the different LUTs are generated.

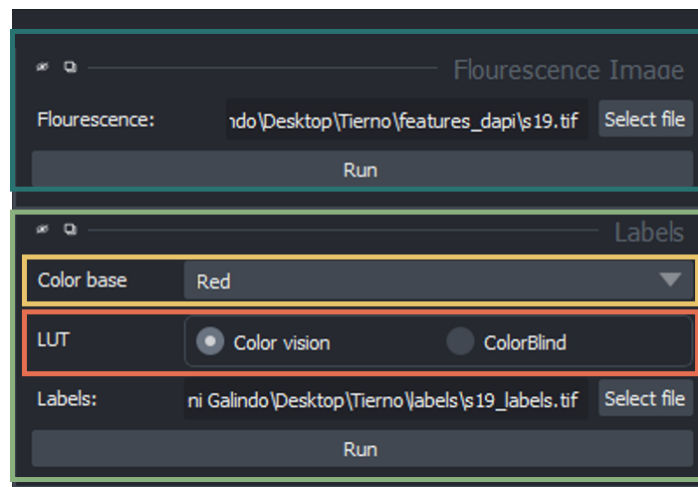


Figure 35: Intensity and label file upload functions. Labels layer features : Intensity image loader (dark green), labels loader (light green), labels' color base dropdown menu (yellow), and look-up table or color palette options (orange).

3.3.1.1 Color vision color palette

Ensuring that two colors are distinguishable is done by computing the color difference, which is their distance in a device-independent color space. A device-independent color space is a space in which the color coordinates used to specify the color produce the same color regardless of the device on which they are applied. The computation of the color difference allows quantifying the features used to describe color, such as hue, chroma, lightness, and brightness. Color coordinates, like RGB coordinates, can be located in a color-geometrical space like the Euclidian geometrical space. Nevertheless, as most color spaces are not perceptually linear, Euclidian distance is not well adapted. As a result, better perceptually uniform color spaces have been developed, particularly, the CIELAB

color space (Figure 36). The CIELAB color difference formula, ΔE_{00} or CIEDE2000, is accepted as the more accurate color difference formula (Lindbloom, 2017; Sharma et al., 2005).

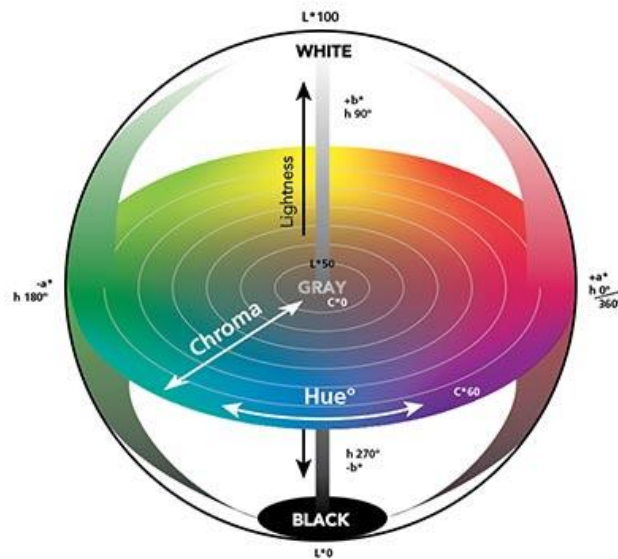


Figure 36: CIELAB color space. L axis describes the luminous intensity of the color, the coordinates a and b represent the main color axes, with red at positive a and green at negative a; yellow on positive b and blue on negative b. The C axis represents chroma or saturation, the H stand for hue, the hue coordinates move in a circle around the "equator" to describe the color family (red, yellow, green, and blue) and all colors in between (X-Rite, Incorporated, 2018, p.).

The minimum color difference value to define two colors as perceptually different is called "the just noticeable difference" or JND and is equivalent to $\Delta E_{00} = 1$. Two colors having a color difference $\Delta E_{00} > 1$ are thus distinguishable. While it is known that having a $\Delta E_{00} = 7$ already corresponds to an obvious perceptual difference, I still chose to only add colors to the LUT that would have a $\Delta E_{00} > 15$. Figure 37 shows an example of the different color difference values mentioned between the default color base and another red colors. An example of color-vision LUT compared with the default color base can be seen in Figure 38.




Color	Color coordinates	Color difference
	R = 255, G = 0, B = 0 R = 250, G = 0, B = 0	$\Delta E_{00} = 1.0$
	R = 255, G = 0, B = 0 R = 255, G = 30, B = 35	$\Delta E_{00} = 7.6$
	R = 255, G = 0, B = 0 R = 210, G = 65, B = 60	$\Delta E_{00} = 15.34$

Figure 37: The color difference between the default color base (Red) and other possible red colors to illustrate how the color difference influences the creation of the color vision color palette.

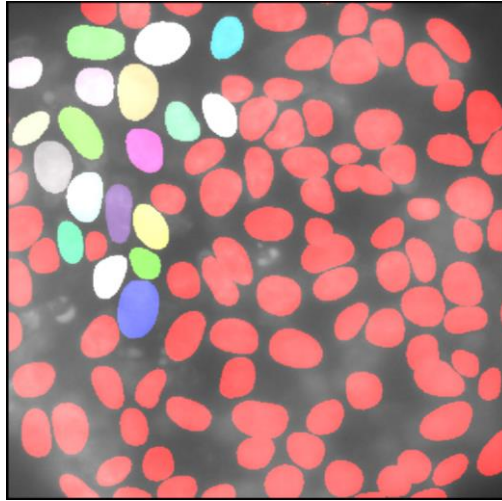


Figure 38: Normal color vision color palette colors calculated using a $\Delta E > 15$ in comparison with the default base color, red = (255, 0, 0).

3.3.1.2 Color blind color palette

Color blindness or color vision deficiency (CVD) is the inability of a person to perceive differences between certain colors. The most common types of color blindness are deuteranomaly (green-weakness), deuteranopia (green-blindness), protanomaly (red-weakness), and protanopia (red-blindness). People suffering from any of those CVD forms have issues distinguishing red or green, even when these colors are just a part of others, resulting in the whole spectrum of colors being affected (Aytac, 2018; SHAFFER, 2016).

People who have red-green color blindness may be able to perceive intense red and green. However, other shades of red and green are perceived as yellow-brown, and brown and orange as shades of brown. A safe way to generate a CVD-friendly color palette is using a combination of colors where at least one of them is not associated with red-green color blindness, such as blue with orange, red, or brown (SHAFFER, 2016). Figure 39 shows how rainbow colors can be perceived by a person that suffers from CVD.

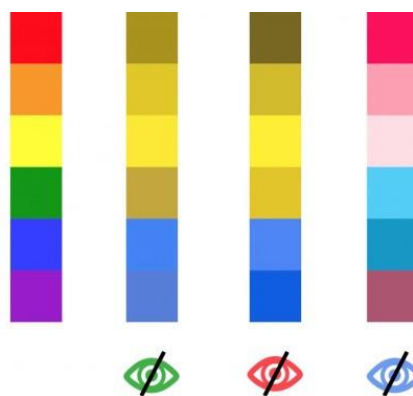


Figure 39: How the rainbow colors may look to a color-blind person (Aytac, 2018). This image demonstrates the importance of implementing a color-blind palette in NAHP.

In contrast with the color vision LUT, I decided that the color-blind LUT would use a limited number of colors, as more colors would equal to more risks of having indistinguishable colors. I therefore chose six colors that appear alternately during the annotation process, with them ensuring high contrast and no bad combinations from a perception standpoint (Figure 40). Figure 41 and Figure 42 show how the

color vision LUT and the color blind LUT are perceived by a person with deuteranopia or protanopia, as simulated by the Visolve (Ryobi Systems Co., 2020) software.

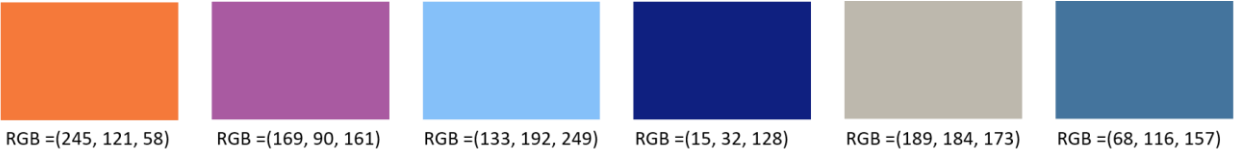


Figure 40: Colors used to create the color-blind color palette. This set of colors was chosen because it allows me to avoid multiple colors based on green and red, allowing color-blind users to distinguish them easily.

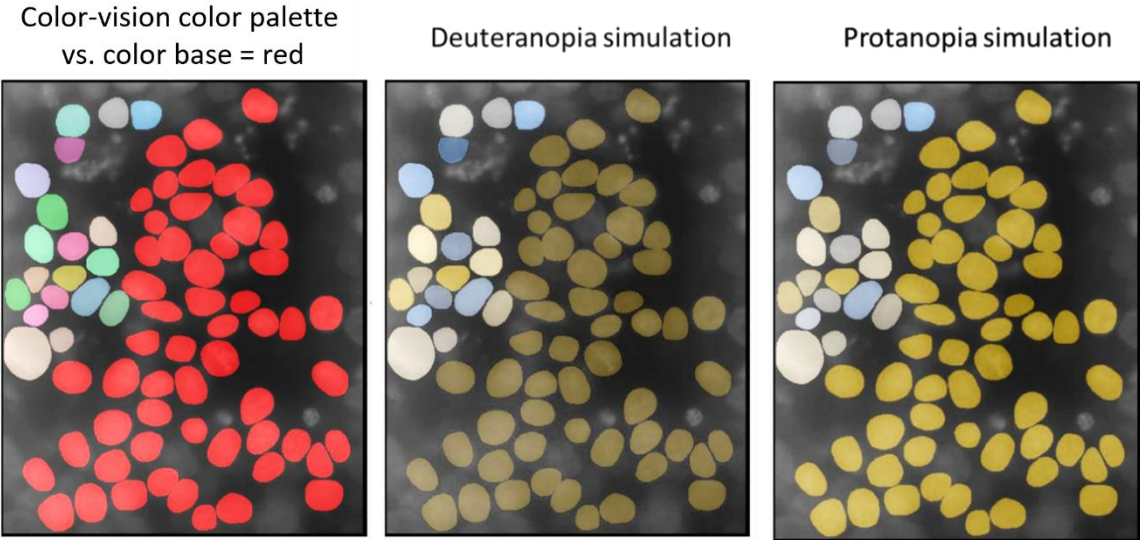


Figure 41: Color-vision color palette under deuteranopia y protanopia simulations done with Visolve. These images show how colors containing red and green can be perceived by a color-blind person when using the color-vision color palette during the annotation process.

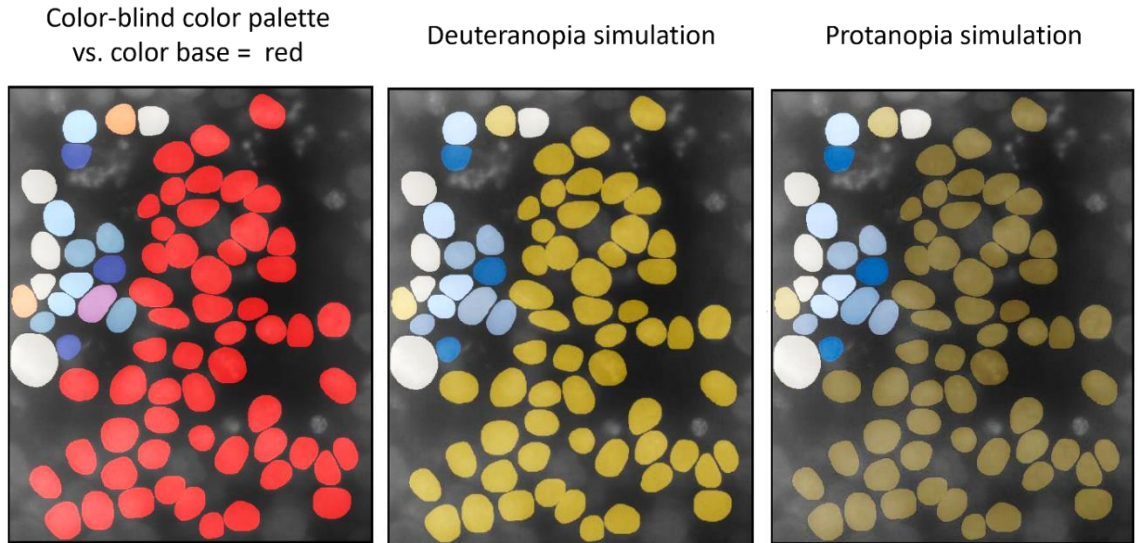


Figure 42: Color-blind color palette under deuteranopia y protanopia simulations done with Visolve. These images show how colors containing red and green can be perceived by a color-blind person when using the color-blind color palette during the annotation process.

3.3.2 Functions to facilitate annotation

To develop a plugin that makes annotating easier, I wanted to address practical situations that frequently resulted in errors. I mainly identified three situations: label tracking, label localization and error localization. Next, I will describe how I tackled each situation.

Label tracking. During annotation, I kept track of all the added labels by writing their values in a classical notebook. In addition to the time spent registering all the labels, this solution was problematic because even minimal distractions could lead to confusion about what was the current label value, and what should be the next one. Consequently, I added a widget that keeps track of all the 3D labels already added and that only displays the last one (Figure 43a). In addition, I added a second widget which purpose is to identify 2D labels that still must be re-labeled as 3D labels in the current frame. The idea is to help identifying labels in crowded regions or very small labels (a few pixels) that are otherwise very hard to locate. Similar to the other widget, it keeps track of all the remaining 2D labels and only displays the first one (Figure 43b).

Label localization. During annotation, it is often necessary to find a specific annotated item, whether to correct it, delete it, or continue with its annotation. The “*Label coordinates*” widget prints the 5 first coordinates of any label chosen by the user, whether it be 2D or 3D (Figure 43c).

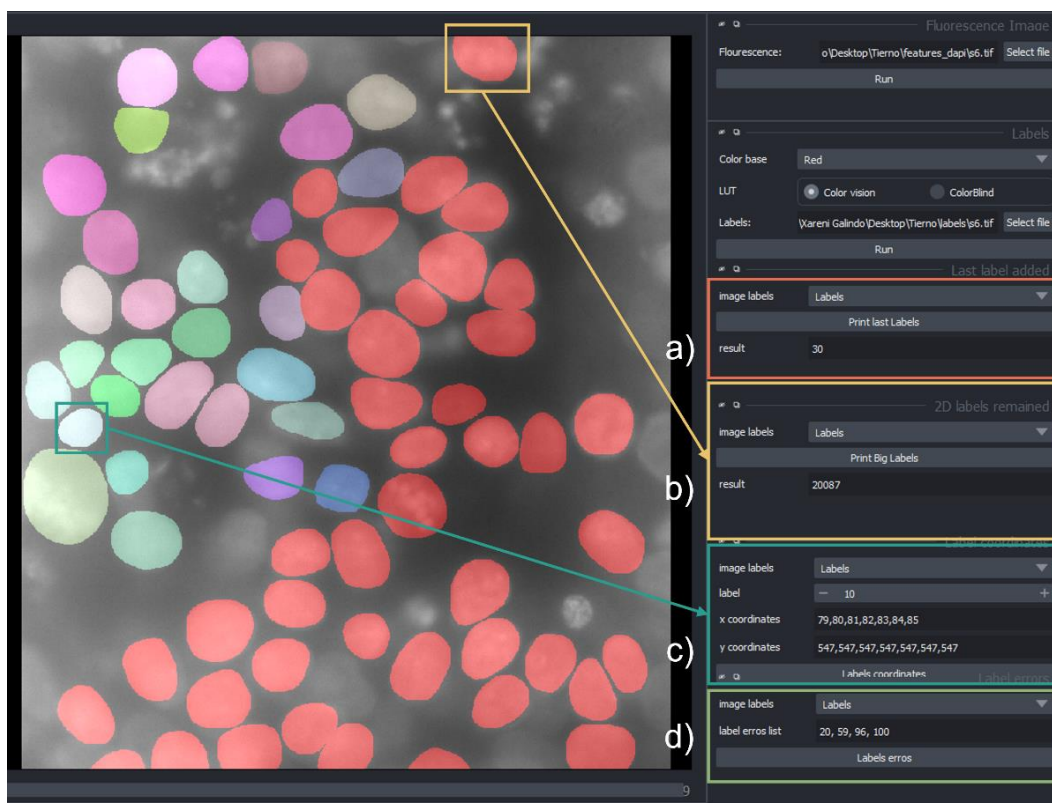


Figure 43: NAHP annotation functions. a) The Last label added function shows the value of the last label added to the annotated image, b) 2D label remained function shows the original 2D labels remaining in the corresponding frame, i.e., the 2D labels that need to be re-labeled, c) label coordinates function shows the x and y coordinates of a certain label, and d) Label errors function shows the errors done during the annotation process.

Error localization. Juggling with labels makes it very easy for human errors. The two most common errors are two objects sharing the same label or a missing label within a labeled object, as illustrated in Figure 44. Hopefully, those errors are easy to identify as one would expect one connected component per object. I therefore implemented a function that screens all the 3D labels, identifies the ones that are not contiguous in the x, y, or z direction, and prints them (Figure 43d).

I expect that these 4 functions will help reducing both the number of errors and the time spent annotating.

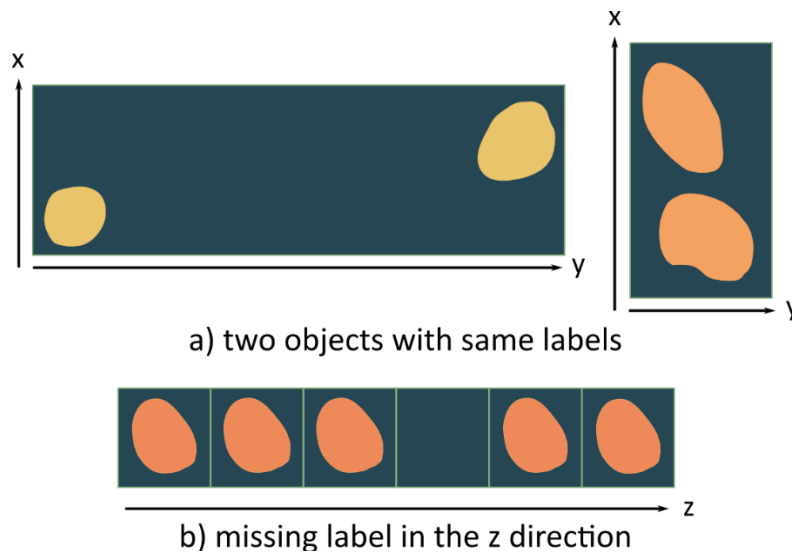


Figure 44: Possible errors during annotation process. a) two different objects with the same label value, these objects can be located on the same frames or not, b) missing label along the z-axis.

3.3.3 Annotation helper extra functions

Finally, I decided to include a set of functions designed to obtain extra information from the annotated images.

Nuclei individualization. Being able to individualize the labeled nuclei is important as it can help with several tasks such as examining the labeling quality or generating simulations. Therefore, I decided to add a function that individualizes the nuclei and cleverly organizes them for visualization. This function requires the number of labels to individualize and generates 3 layers: one for the label itself, one for the corresponding part of the fluorescence images and one for the bounding box (Figure 45). Because the nuclei voxels have different sizes, concatenating all the voxels together was not computationally efficient and led to images with huge dimensions. As the image voxels extracted can be seen as carton boxes, I addressed the concatenation problem as an "optimal bin packing problem", i.e., how to pack a set of n 3D boxes with minimal space waste (Figure 46) (Maarouf et al., 2008). Specifically, I relied on a Python package called `3dbinpacking` (*3D Bin Packing*, 2020; Dube et al., n.d.).

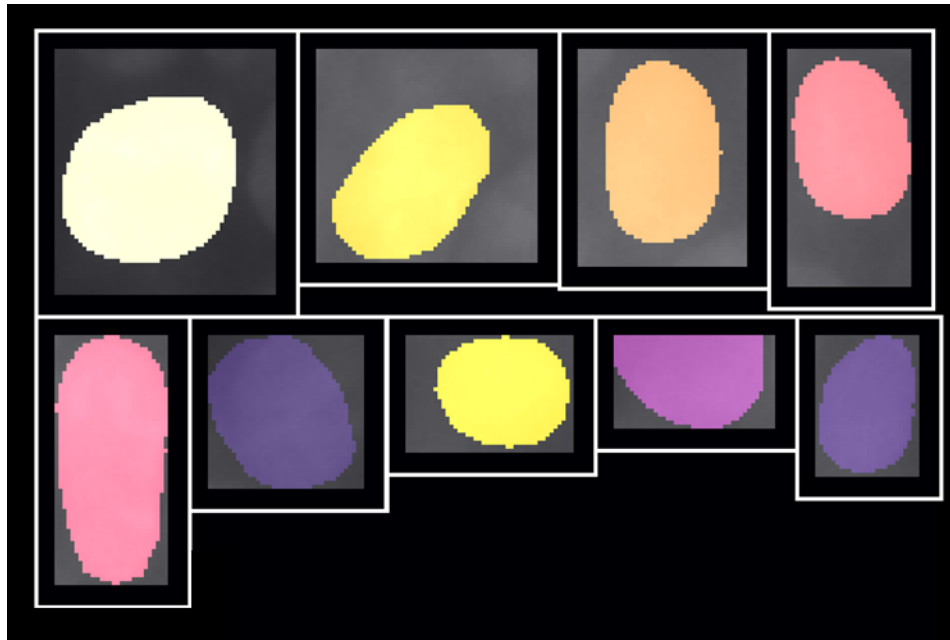


Figure 45: Nuclei individualization example. The individualization function allows us to have an insight of the nuclei morphologies and to perform a nuclei profiling analysis.

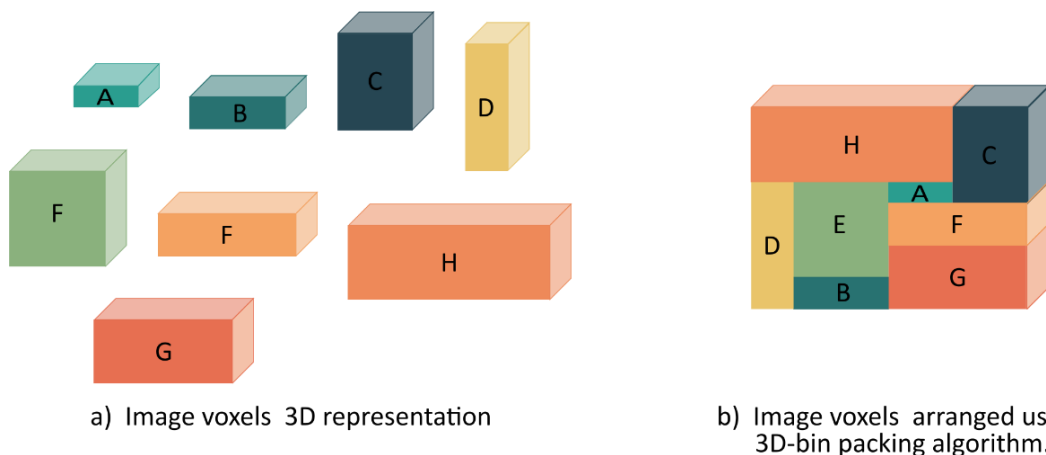


Figure 46: Optimal bin packing problem diagram. a) image voxels represented by “carton boxes”. b) image voxels arrange in the most efficient way.

Nuclei localization. Jointly to the individualization process, I added the possibility of localizing the individualized nuclei directly in the original fluorescence image stack. This is done by duplicating the label layer and setting all the non-individualized labels to 0, making them invisible (Figure 47).

Properties table. This function computes, for each 3D label, several quantitative measurements. It uses both the labeled and fluorescence image stacks and display in a panda-compatible table several features: label id, area, centroid, bbox (bounding box), intensity maximum, intensity mean, and intensity minimum (Figure 47).

Centroid localization. Finally, this function uses the centroid column computed in the “Properties table” and create a new point layer, allowing displaying the centroids directly on the fluorescence image stack (Figure 47).

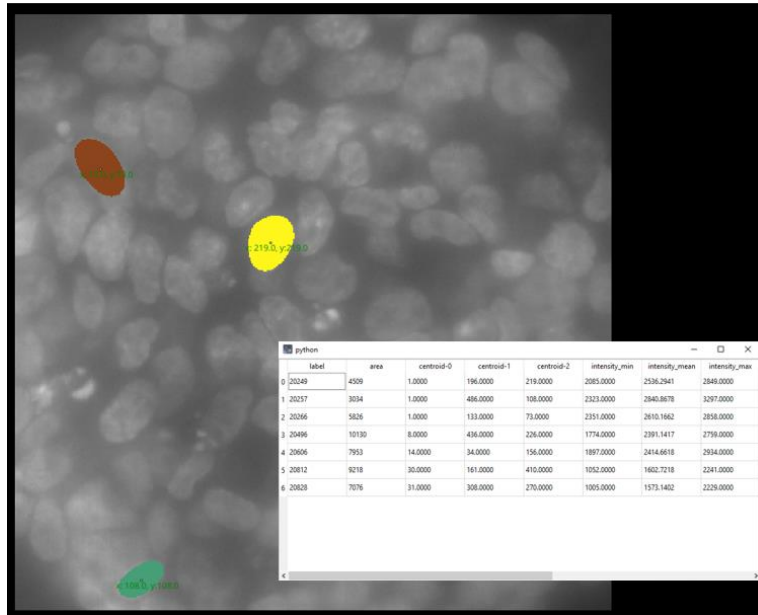
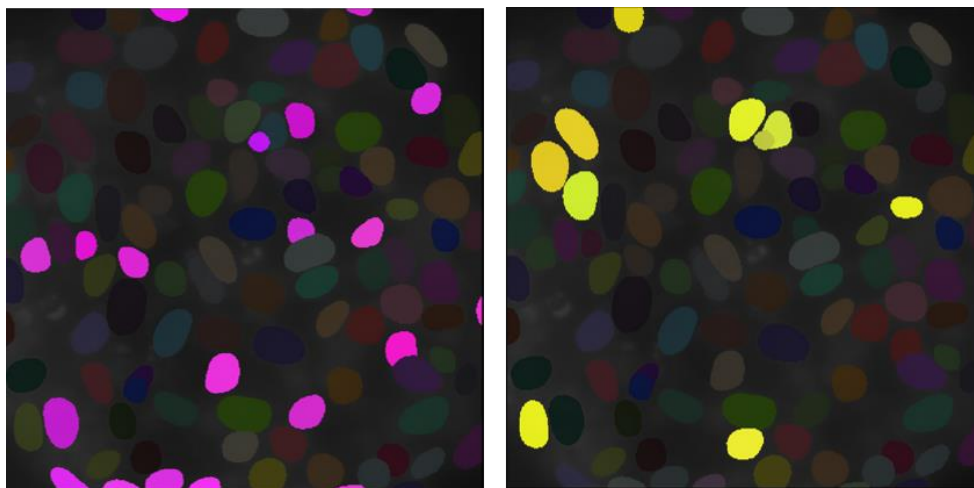


Figure 47: Nuclei localization image example with centroids localization and properties table displayed. These functions allow us to have statistical information about the nuclei contained in the image. Table properties function allows the user to export the table into a csv file.

Area and Intensity thresholding sliders. I implemented these 2 sliders to filter the 3D labels with respect to their area and intensity, allowing the user to generate new label layers (Figure 48).

The implementation of all the functions mentioned above can be seen in Figure 49. This is the view of NAHP Napari plugin.



a) Annotation generated using the Area threshold slider

b) Annotation generated using the Intensity threshold slider

Figure 48: Examples of new annotations generated using the area (magenta) and intensity (yellow) sliders vs. the original annotations done. These sliders will allow user to generate extra annotation what can be really helpful to increase our GT dataset.

3.4 Discussion

During the past years, there has been a great effort in benchmarking 2D and 3D nuclei segmentation methods using different datasets covering samples from various species. However, most of them are of limited size, and only a few include 3D labels (Lin et al., 2021). During the development of the ground truth dataset presented in this thesis, I annotated 4657 3D nuclei, making it one of the biggest 3D nuclei annotated datasets. This dataset can be used for training, testing, and validating nuclei segmentation algorithms, and it aims to be a milestone in the field of 3D nuclei segmentation. More specifically, I hope it will be widely adopted as a benchmarking dataset for new 3D segmentation algorithms.

In parallel, I developed Napari Annotation Helper (NAHP) to facilitate the annotation process within the team. While developed specifically for the 3D annotation of cell nuclei, I hope NAHP can also apply to a variety of other images that present 3D structures. The main aim of NAHP was to drastically reduce the annotation time while limiting the number of errors.

I achieved this goal by implementing a set of functions addressing some of the most concerning problems that arise when annotating, such as the confusion caused by constant changes in label value and transition between frames. All the widgets I added to NAHP help users by indicating critical information, such as the last 3D label added, the remaining 2D labels to handle or the coordinates of a specific label. Combined with the functions for detecting and localizing labelling errors, NAHP allows a swift reconnection of 2D labels into 3D objects. Moreover, I added several quantitative measurements of the 3D labels as additional features, to help users understand their data via filtering and visualization.

Ultimately, I believe that the work done in this part of my thesis will have a meaningful impact on the community. Indeed, I tackled the two main directions in relation with ground truth generation: I both developed a tool to facilitate it and provided a corpus of 3D labels of substantial size. In the future, these two directions will be more and more needed by the community.

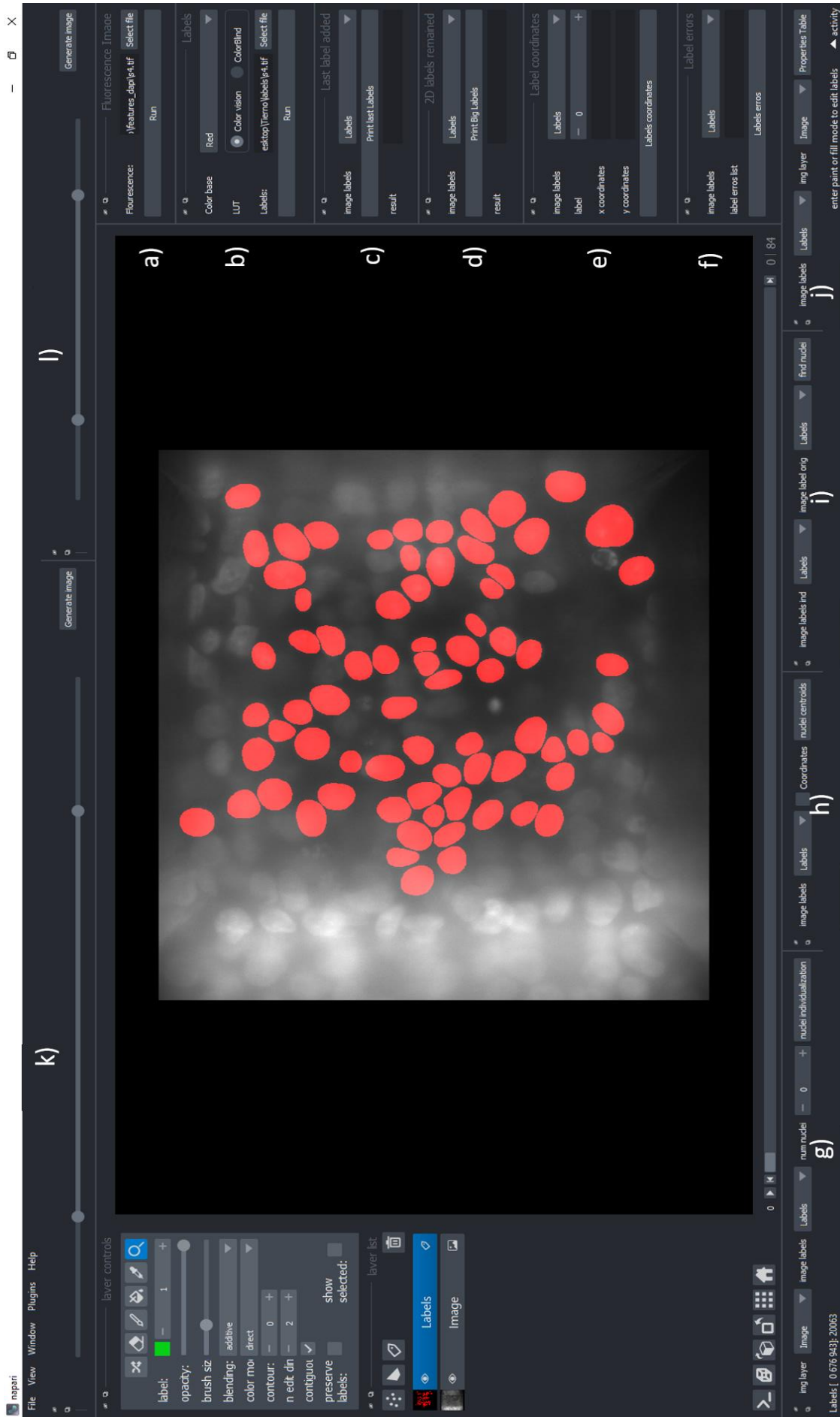


Figure 49: Napari Annotation Helper plugin full view. Functions to facilitate the annotation process: a) Fluorescence file upload, b) Labels file upload, c) Last label added, d) Original labels remained, e) Label coordinates, and f) Label error localization. Annotation helper extra functions: g) Nuclei individualization, h) Nuclei localization, i) Centroid localization, j) Properties table, k) Area threshold slider, and l) Intensity threshold slider.

4 | *Quantifying the nuclei organization of 3D cell cultures with StarDist*

Main contributions presented in this Section:

- A **complete pipeline** to train StarDist in 2D and 3D using the ground truth dataset and the plugin presented in Section 3.
- An **application** of this pipeline on a biological model (spheroids and neuroectoderm organoids) to **quantify their nuclei organization** in 3D.

In recent years, deep-learning based methods have taken by storm the biomedical field with dozens of techniques being developed each year. For segmenting nuclei, StarDist and Cellpose have been released and have easily surpassed traditional image processing methods. As I had to segment close to one hundred stacks of 3D cultures as part of a collaboration with team of V. Viasnoff, StarDist appeared to be the perfect solution. Unfortunately, no accurate pre-trained model for StarDist in 3D existed because of the lack of ground truth in 3D. Therefore, I had to create this dataset, train StarDist in 3D and quantify the nuclei organization of these samples. Along the way, I realized a few interesting observations on the functioning of StarDist.

4.1 Image preparation

After installing my Python environment to run StarDist, it became rapidly obvious that I would not be able to directly use the images acquired with the soSPIM system. Indeed, the size of those 3D images would not fit into my graphics card memory, and as I had no access to a GPU server, I realized I would need to resize them. As StarDist was provided with Jupyter notebooks, I had access to example of images on which StarDist was trained. I was therefore able to determine in a clever way what resizing was required to suit the networks' hyperparameters requirements and field of view.

I defined the resized ratio as the ratio between the average nuclei size of our original labeled images (Table 1, dataset) and the StarDist 2D and StarDist 3D sample images' nuclei (Schmidt, 2021a, 2021b), respectively. The average length of the nuclei in our dataset was 127 pixels, while the average nuclei size in StarDist 2D and StarDist 3D sample datasets was 49 and 28 pixels, respectively. Doing a simple computation, I determined that the resized ratio was 1:2.6 for StarDist 2D and 1:4.5 for StarDist 3D. After determining the resizing ratios, I used Fiji's resized function with no interpolation to avoid change in the label's values. It is important to mention that resizing not only helps to adapt the images to the neural network's hyperparameter but also to cope with memory allocation problems. Figure 50 shows a diagram of the resizing process and Figure 51 show some examples of the images after the preparation process.

I also had to convert the images from 32 bits to 16 bits.

Table 1: Image sizes in pixels before and after preprocessing.

Dataset		Original image size	Image size after cropping and slicing	Image size for StarDist 2D training	Image size for StarDist 3D training	
Original	sample 1	2048x2048x100	1532x1398x85	589x537	340 x310x85	
	sample 2					
	sample 3		886x1110x80	341x427	197x247x80	
Complementary	sample 1	1024x1024x71	1024x1024x71	394x394	228x22x71	
	sample 2					
	sample 3	1024x1024x50	1024x1024x50			228x22x50
	sample 4					

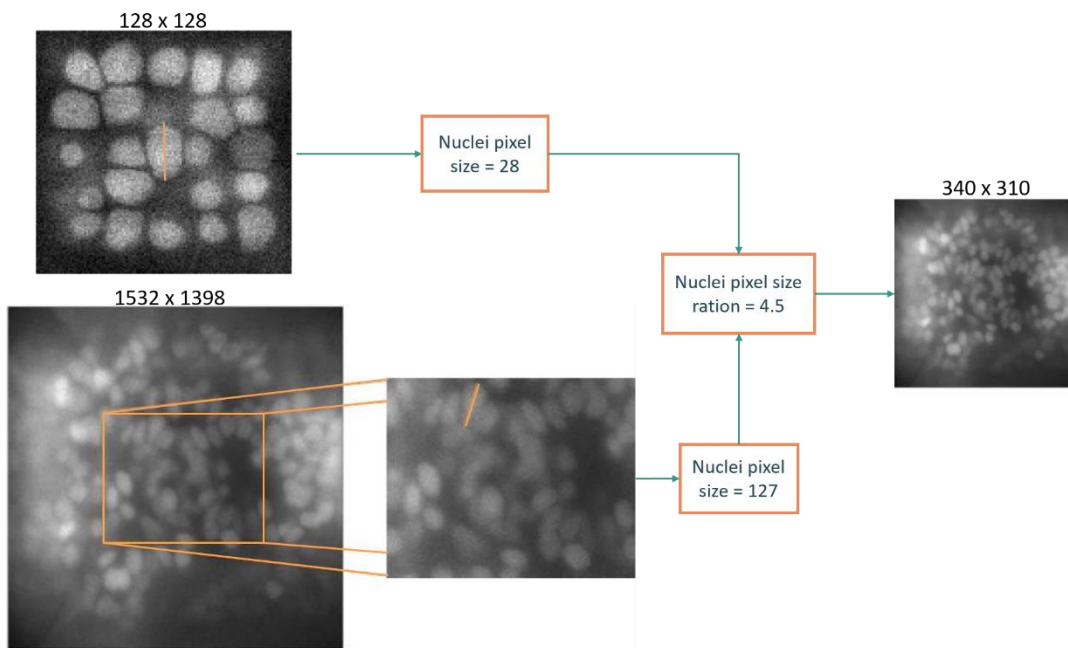


Figure 50: Re-sizing process diagram. The resizing process allow us to avoid memory allocation problems and have image patches inside the neural network field of view.

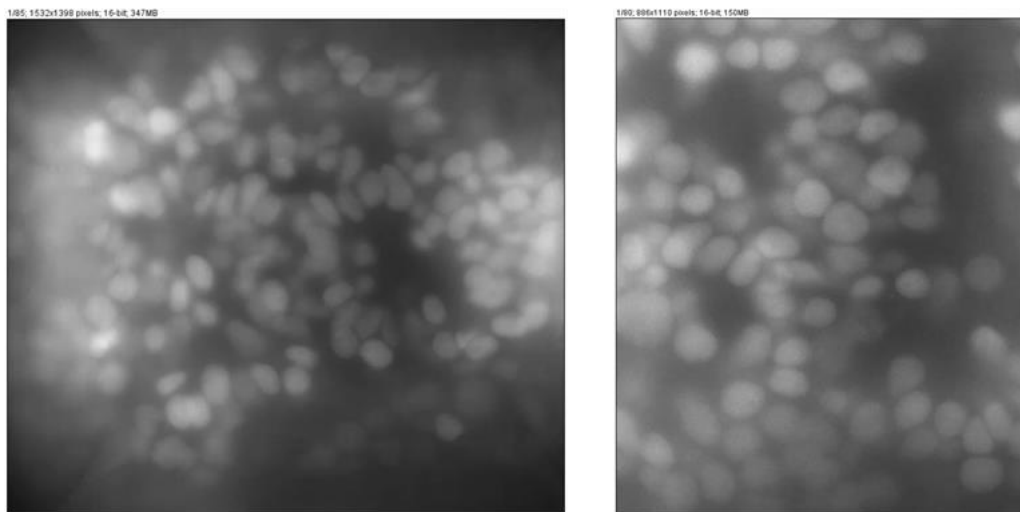


Figure 51: Examples of images samples after cropping and preprocessing. Original data set sample 1 and 3.

4.2 Nuclei segmentation results

During my PhD, I realized around 70 different trainings using different types of image combinations and preprocessing methods. Although many experiments were performed, not all yielded relevant information. In the following sections, I will only discuss the most relevant experiments for this project's development and their respective results.

As convention, the first three image stacks that I fully manually annotated will be called "original-DAPI dataset" when using the DAPI-stained images and "original-SOX dataset" when using the Sox2-stained images. Of the four stacks that were subsequently annotated in 3D from 2D segmented labels, three were used for training and will be referred to as "complementary dataset", while the last one was used for evaluating the models' accuracy and will be named the "prediction dataset."

To implement StarDist 2D and StarDist 3D, I used:

- A set of Jupyter notebooks containing the training and prediction implementation. These notebooks can be found in StarDist GitHub:
 - StarDist 2D: <https://github.com/stardist/stardist/tree/master/examples/2D>
 - StarDist 3D: <https://github.com/stardist/stardist/tree/master/examples/3D>
- Anaconda Navigator to launch the Jupyter notebooks used.

4.2.1 StarDist 2D implementation

Although our focus was to perform 3D nuclei segmentation, it rapidly became obvious that I could take benefit of StarDist 2D in order to speed up the annotation process. But, as I said before, several ground truth datasets exist for nuclei segmentation in 2D. I was therefore able to train different StarDist 2D models, whether it be with these existing datasets, our data or a combination of both.

To implement StarDist 2D, each frame of the fluorescence image stack is saved as an individual file and paired with its corresponding mask. StarDist 2D trainings were performed using the same configuration, with the corresponding details being shown in Table 2. In all cases, the training lasted around 15 min, and the prediction took around 40 seconds.

Table 2: StarDist 2D segmentation model details.

StarDist 2D model training configuration	
No. of epochs	200
Patch size	256, 256
Train steps per epoch	100
Train batch size	1

4.2.1.1 StarDist 2D segmentation results using Data Science Bowl 2018.

The first attempt to train StarDist 2D was made using the same dataset used in the Jupyter notebook example provided on StarDist's GitHub website, which happens to be the DSB2018 (Booz Allen Hamilton, Inc., 2018). Using this dataset, I developed a segmentation model using the same configuration mentioned in Table 2. To test the accuracy of the corresponding segmentation model, the prediction dataset was separated in individual frames to create a 2D test dataset.

The DSB2018 dataset comprises images acquired with two major types of light microscopy (brightfield and fluorescence) under different experimental conditions and nuclei densities. The DSB2018 contains 497 images with an average of 43 nuclei per image. Even so, no sample acquired with light-sheet

microscopy was present. This unfortunately means that this set does not encapsulate the type of data we wanted to segment, as both the illumination scheme and the nuclei density and organization or our 3D cell cultures was absent. This discrepancy between the training dataset and the prediction dataset can be observed in the segmentation results, as this model presents certain problems when segmenting overexposed areas, very crowded regions, cell divisions, and dim objects as shown in Figure 52.

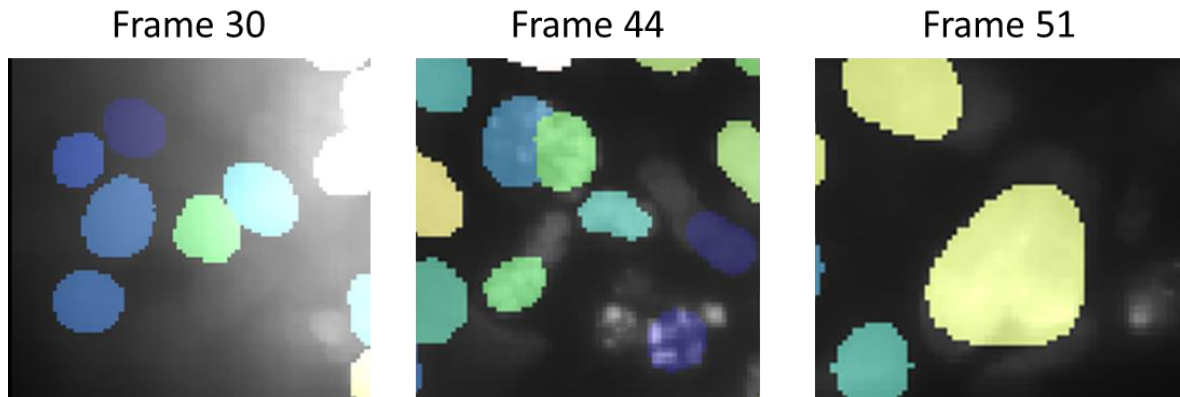


Figure 52: Prediction using the StarDist 2D model developed using DSB2018. In this zoom made in the different frames we can observe the over- and under-segmentation performed by this segmentation model.

Due to the results obtained with the segmentation model trained using the DSB2018 dataset, I realized it was necessary to train a model with images representing our data.

4.2.1.2 StarDist 2D segmentation results using the original -DAPI dataset

I trained StarDist 2D using the original-DAPI dataset, which images were preprocessed and resized according to the methods mentioned in the Image section. To develop this model, I used Original-sample 1 and sample 3 for training, Original-sample 2 for validating and Complementary-sample 2 for testing. Figure 53 shows some examples of the segmentation obtained using this trained StarDist 2D model. Unfortunately, even visually it was obvious that this model was performing very poorly, with a majority of the nuclei being missed.

Figure 54 shows the different metrics computed during the testing phase of the training, as well as the computation of the true positives (TP), false positives (FP), and false negatives (FN) according to the different intersection over union (IoU) thresholds. As usual, the values of the metrics decrease with the incrementation of the IoU threshold values, highlighting the importance of correctly defining the IoU threshold. In the literature, segmentations algorithm performance is evaluated using single values of IoU, with the most common ones being between 50% and 75% (Padilla et al., 2020). Because of this, I decided to use an IoU threshold of 0.6 to define my model's accuracy. With this value, I got an accuracy of 0.8 and an F1-score of 0.9.

Although the accuracy values are high, the model performed very poorly. This may be related to two causes: first, the training set only comprises of a couple of hundreds of images, which is certainly insufficient to achieve a good model generalization. Second, the image from the prediction set was acquired in a different condition than the one from the original-DAPI set. This also could explain why the model performed poorly. This is a clear example that to determine if segmentation is accurate or not, just looking at the number is not enough; inspecting the images is always necessary.

Consequently, I decided to develop a new StarDist 2D model using the original-DAPI and the DSB2018 dataset. By merging those, I expected to obtain better results thanks to combining images taken under different imaging conditions, with different nuclei densities and using different cell culture methods.

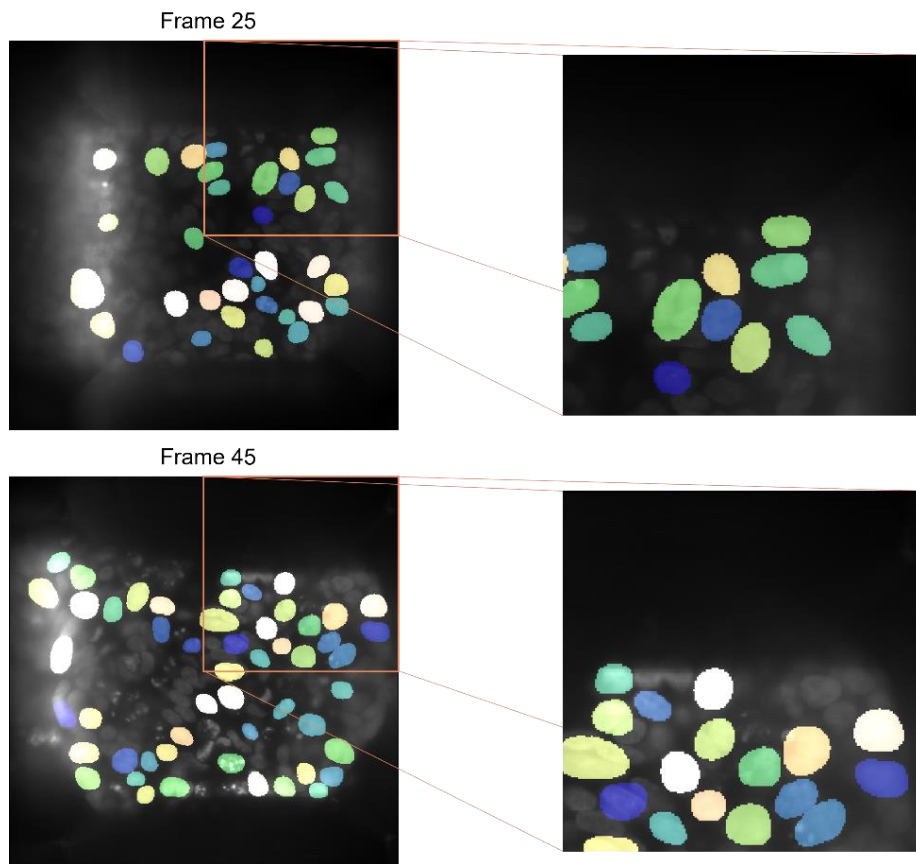


Figure 53: Fluorescence image vs. StarDist 2D segmentation comparison using original-DAPI training dataset. In these images we can observe the poor performance achieved by this model.

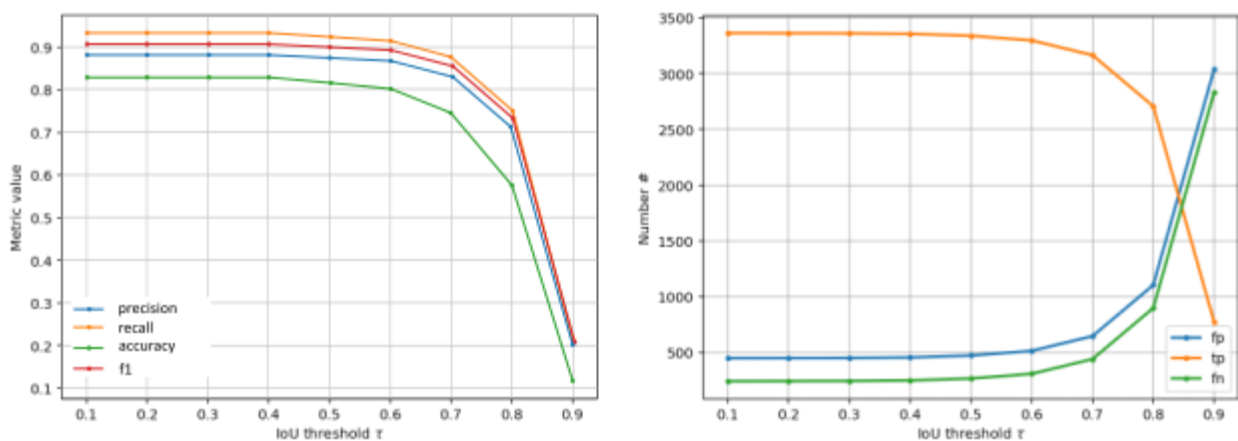


Figure 54: StarDist 2D training performance using DAPI stained cell images. Although these graphs give the impression that the model works accurately due to the high values shown by the different metrics, a quick inspection of the segmentation obtained contradicts these conclusions.

4.2.1.3 2D StarDist segmentation results using original-DAPI and DSB2018 datasets

To develop this model, the original-DAPI images were preprocessed accordingly to the steps mentioned in the Image Preparation section while the images from the DSB2018 dataset remained unmodified. I will refer to this training as original+DSB2108 in the text. Figure 55 shows examples of the segmentation obtained using this model. To develop this model, I used Original-sample 1 and sample 3 for training Original-sample 2 for validating plus randomly assign images of the DSB for training and validating phases, and Complementary-sample 2 for testing.

The improvement in the segmentation when applying this model to the prediction set can be seen when comparing Figure 53 and Figure 55. Figure 56 shows that adding the DSB218 dataset to the training dataset highly improved the segmentation model accuracy. This improvement can be due to the employment of images with different nuclei density, illumination, and experimental conditions together with images taken using the soSPIM, which represent the illumination condition we need our model to learn. Thanks to this, our models not only learn to segment kernels in different distributions but also with different types of illumination, which is one of the characteristics of our images. All this is further evidence that training a neural network using images that reflect diverse imaging and experimental conditions is always a wise decision.

From the results obtained from this experiment, we can see that the diversity of images benefited the accuracy of the segmentation results. These results demonstrated that I managed to train a very accurate StarDist 2D model thanks to a diverse training set. This model can therefore be used to segment all the frames of 3D cultures image stacks, a critical step for performing semi-automatic 3D labeling.

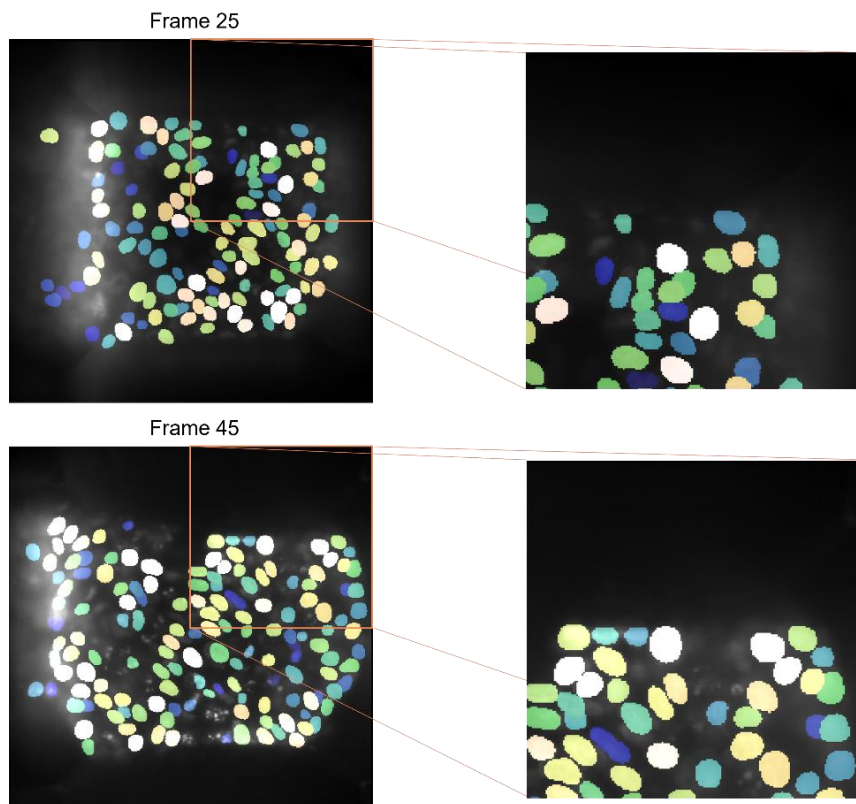


Figure 55: Fluorescence image vs. StarDist 2D segmentation comparison using original-DAPI and the DSB2018 datasets.

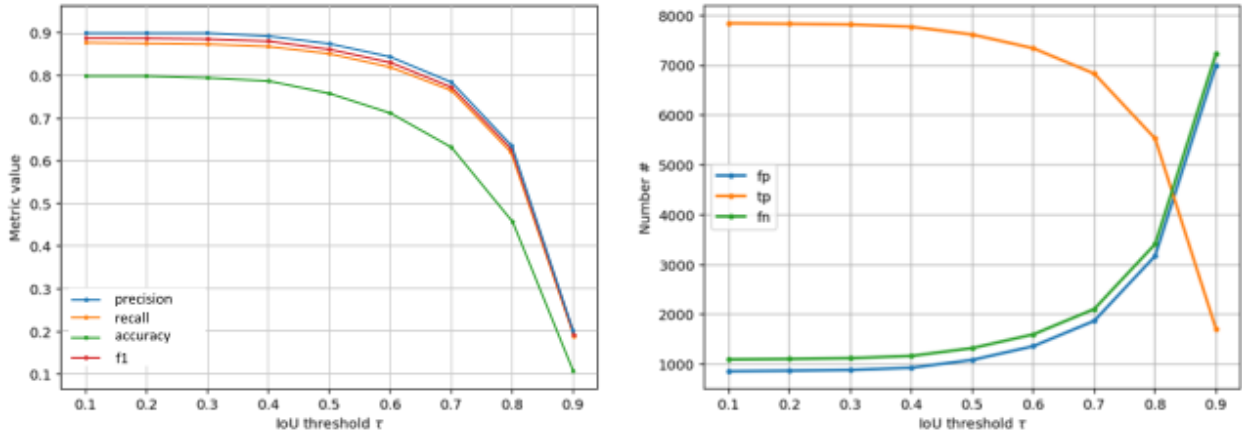


Figure 56: StarDist 2D training performance using original-DAPI and DSB2018 datasets.

4.2.2 StarDist 3D implementation

I then used the StarDist 2D original-DAPI+DSB2018 model to segment the complementary dataset in 2D and perform their 3D annotation, as described in the previous chapter. Using original-DAPI, original-SOX, and complementary datasets, I trained several sets of StarDist 3D models. Of all the trainings performed, three gave relevant results, one using the original-SOX dataset, another one using original-DAPI and complementary dataset, and the last one using original-SOX and complementary dataset. I will refer to these training as Sox2, DAPI, and SOX-DAPI. All carried out trainings had the same configuration, as shown in Table 4.

4.2.2.1 3D StarDist segmentation results using original-SOX dataset

As mentioned before, just the original dataset composed of three stacks was acquired using both the SOX2 and DAPI labeling. In this case, I developed a segmentation model using two of these images as training dataset and the last one as prediction dataset using the configuration shown in Table 3.

Table 3: StarDist 3D segmentation model details.

StarDist 3D model training configuration	
Time	2.5 hours
No. of epochs	200
Patch size	48, 128, 128
Train steps per epoch	100
Train batch size	2
Total number of stacks	2
Number of stacks for training	1
Number of stacks for evaluation	1
Number of stacks for validation	1

Some examples of the results obtained can be seen in Figure 57 and its 3D view can be seen in Figure 58. In Figure 59, the reader can observe the performance of the corresponding training. This image shows that the training presents a poor performance for IoU values above 0.6, where all the metrics descend rapidly to very low values. By computing the corresponding IoU values between GT and predicted objects using the method presented in Caicedo et al., 2019, I obtained the amount of TP, FP, and FN for an IoU = 0.5 along its F1-score, as can be seen in Table 4. This result are not a surprise since all images were taken under the same experimental conditions and preprocessed using the same method. Although the results observed for this first implementation of StarDist 3D seemed promising,

I was aware that more data were needed to develop a better segmentation model applicable to a wide range of organoids.

Table 4: True positives, false positives, and false negatives of the predictions done using Sox2 labeled images image

Model	Prediction	Total nuclei	TP	FP	FN	F1-score
StarDist Sample1&3	Sample 2	705	322	111	383	0.566

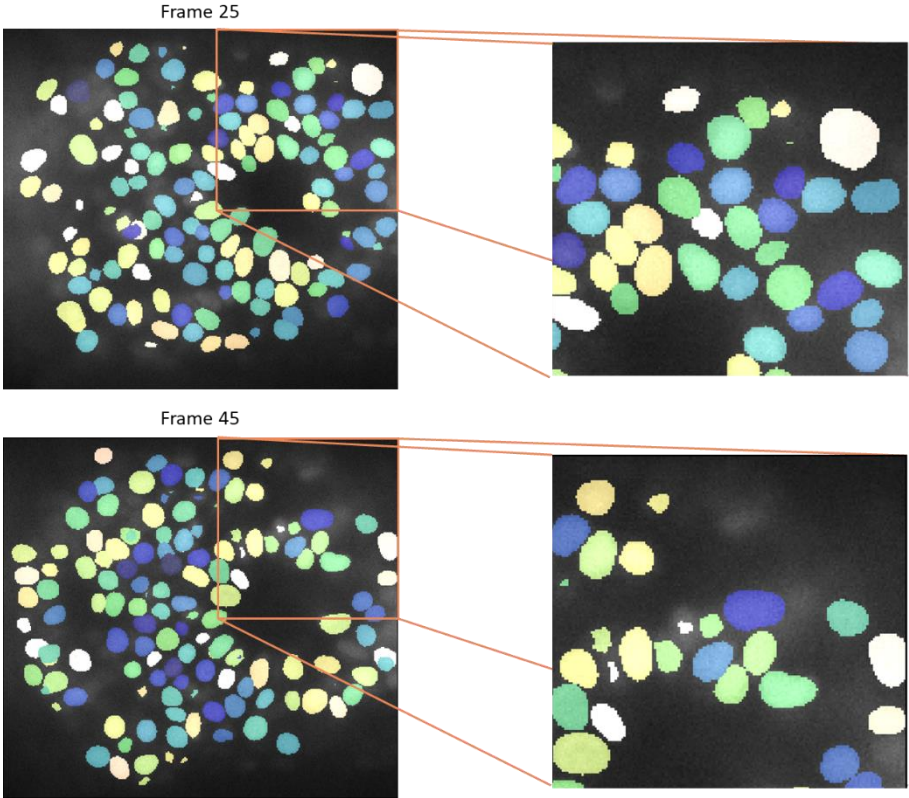


Figure 57: Fluorescence image stack vs. original-SOX training prediction.

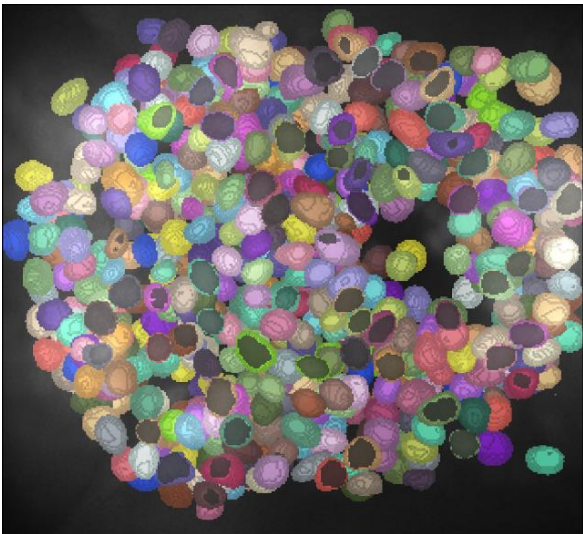


Figure 58: 3D view of the original-SOX training prediction.

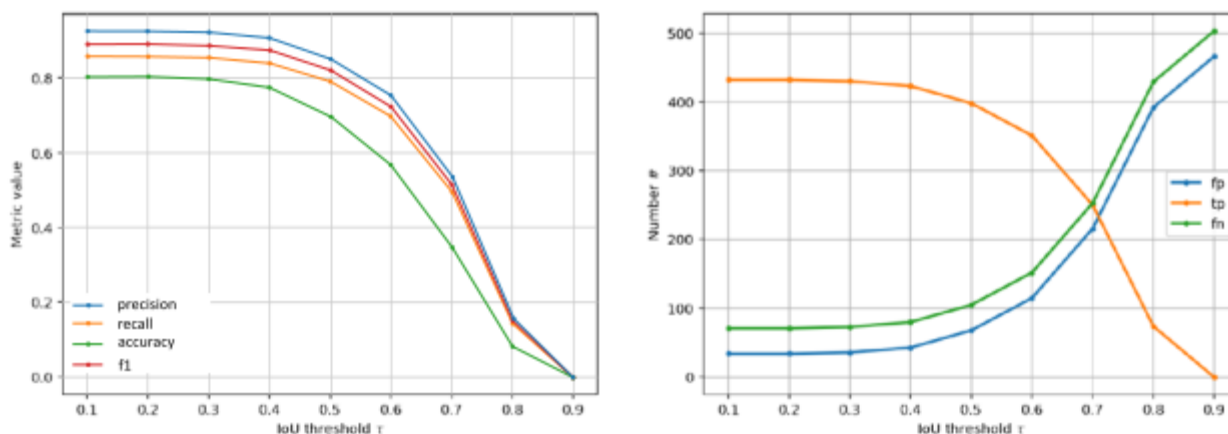


Figure 59: StarDist 3D training performance comparison between trainings performed with Sox2 stained cells.

4.2.2.2 3D StarDist segmentation results using original-DAPI, original-SOX and complementary datasets.

As the original training dataset was only composed of 3 images, and it was clear that it was not sufficient to train an accurate StarDist 3D model, I annotated the complementary dataset. After completion, I performed a set of trainings using original-SOX, original-DAPI, and the complementary dataset combined. The first training was done using original-DAPI and the complementary dataset, and the second was performed using original-SOX and the complementary dataset. To develop these models, I used the Original dataset (SOX and DAPI respectively) plus complementary-sample 1 and 3 for training, complementary- sample 2 for validating, and Complementary-sample 4 for testing.

I decided to implement these two training because I wanted to compare the results obtained when using a single and a mix of nuclei labels. Because the complementary images were only labeled with DAPI, I will refer to these two trainings as DAPI and SOX-DAPI.

Both trainings were performed using the same configuration to assure the comparison is possible. Figure 60 and Figure 61 show two frames from the prediction dataset corresponding to the segmentation done using each model. These frames were chosen because they best represent the results obtained. These frames are located at the center of the stack, where all the nuclei are on focus and does not exhibit any blur, which is not the case for the nuclei located in the first or last frames of the stack. In addition, Figure 62 shows the performance plots for the corresponding validation process of these trainings.

Overexposed regions present some segmentation problems in both cases, which is not surprising because this is the more difficult area to segment due to challenging illumination. Figure 61 shows that the SOX-DAPI model performs better than the DAPI model in the areas where the light reflection is higher, as seen in Figure 60. SOX-DAPI model seems to under-segment objects in frames with low illumination, and the DAPI model seems to over-segment them, giving segmentations that do not correspond to the size or position of the nuclei present in the frames. However, the segmentation of the nuclei located in the middle frames of the stack seems to be very similar in size and location in both cases.

The behavior of both segmentation nuclei seems to be very similar with 1041 and 1037 objects segmented for SOX-DAPI and DAPI respectively. However, it is important to notice that although small, the performance of the SOX-DAPI model appears to be better than the DAPI model. This can be due to using different immunostaining labels, creating a more generalized model. Although the TP, FP, and FN were not computed due to a lack ground truth, as the prediction dataset was not labeled, using

only two immunostaining labels and six images to train StarDist 3D produced very good results, even though this is still considered a very limited dataset. Finally, the SOX-DAPI model was the one I used to segment almost one hundred of 3D cultures of our collaborators, and to quantify the nuclei organization as can be seen in the next Section.

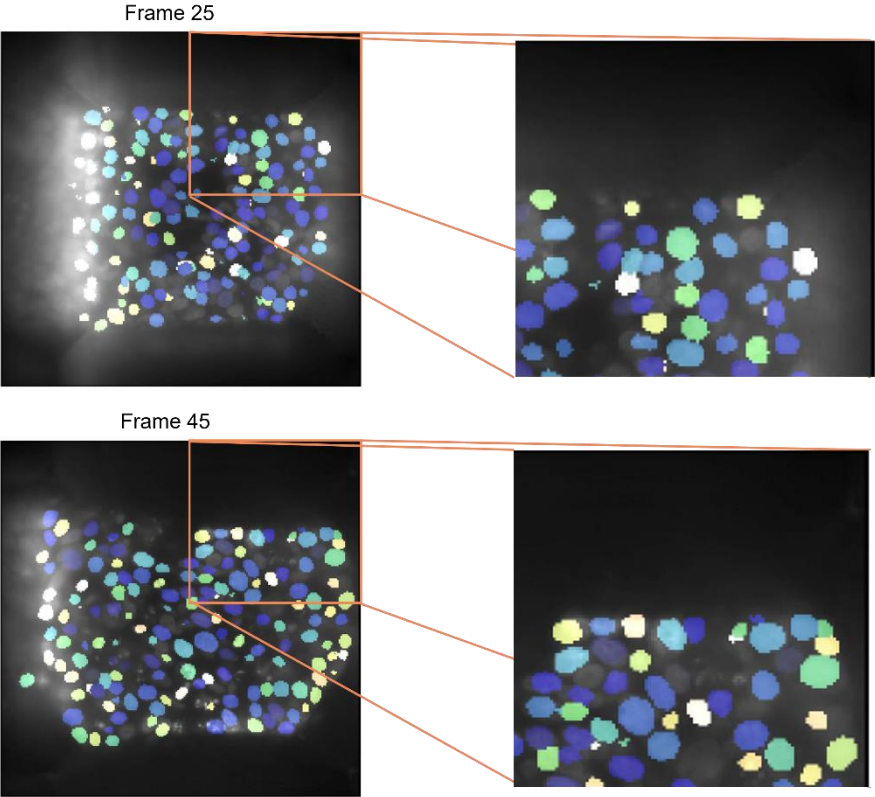


Figure 60: Fluorescence image stack vs. original-DAPI training prediction.

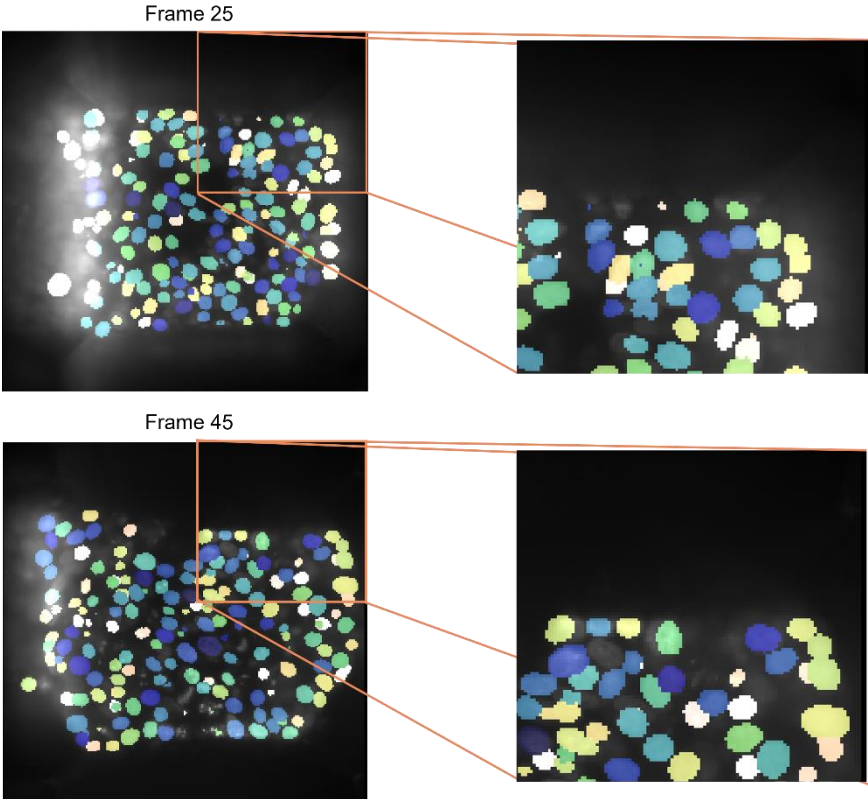
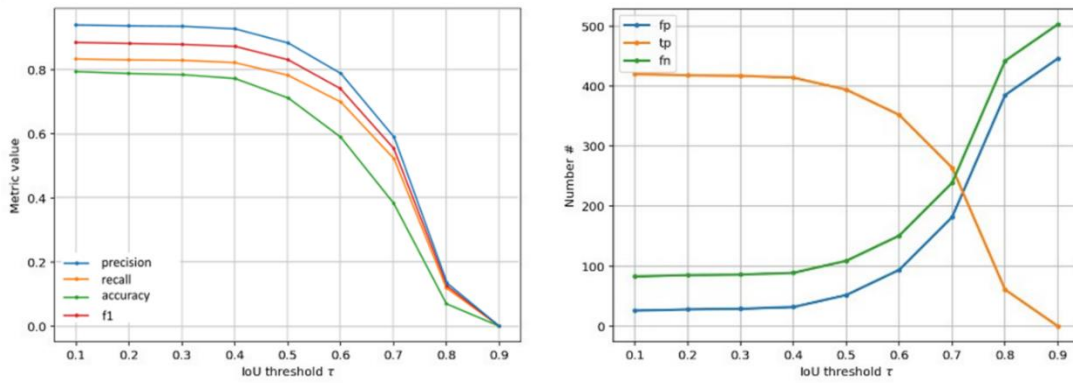
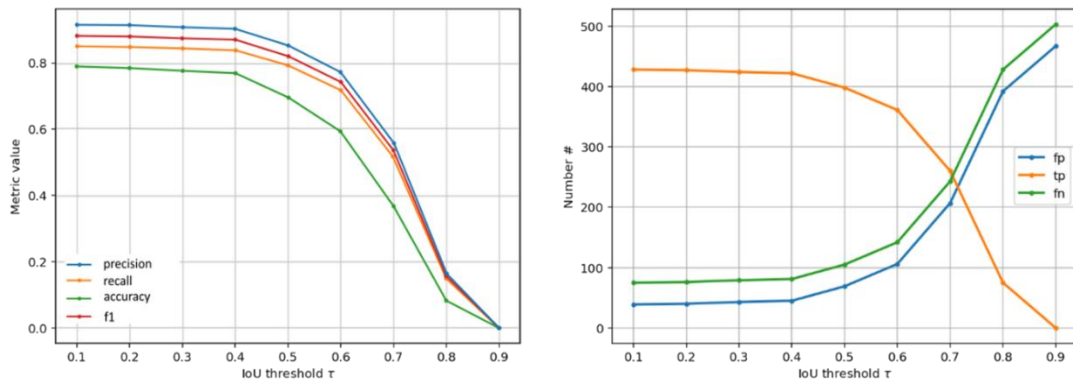


Figure 61: Fluorescence image vs. SOX-DAPI training prediction.



a) StarDist 3D DAPI labeled images training performance.



b) StarDist 3D SOX-DAPI labeled images training performance.

Figure 62: StarDist 3D training performance comparison between DAPI and SOX-DAPI trainings. These figure shows clearly the difference between TP, FP and FN objects for each of the models for particular IoU values.

4.2.3 Quantification of the nuclei organization in 3D cultures using StarDist 3D

As part of a collaboration with the team of V. Viasnoff located in Singapore, our team wanted to upgrade the soSPIM system to allow acquiring automatically hundreds of 3D cultures. It's in this context that I trained the SOX-DAPI StarDist 3D model, to segment the organoids they acquired. In total, I segmented 96 organoids, resulting in 55,976 identified nuclei. The next step was to assess the results' accuracy, as I used all the labeled datasets to train the model. For this, I performed a careful visual assessment to count the amount of FP and FN and computed the F1-score on four of the 96 organoids segmented. The four organoids chosen can be seen in Figure 63, and the corresponding results obtained for each of the images selected are shown in Table 5: Example of the segmentation results presented in (Beghin et al., 2022).

Table 5: Example of the segmentation results presented in (Beghin et al., 2022)

Image	TP	FP	FN	F1-score
Sample 1	694	3	25	0.98
Sample 2	301	4	6	0.98
Sample 3	705	9	27	0.97
Sample 4	475	2	14	0.98

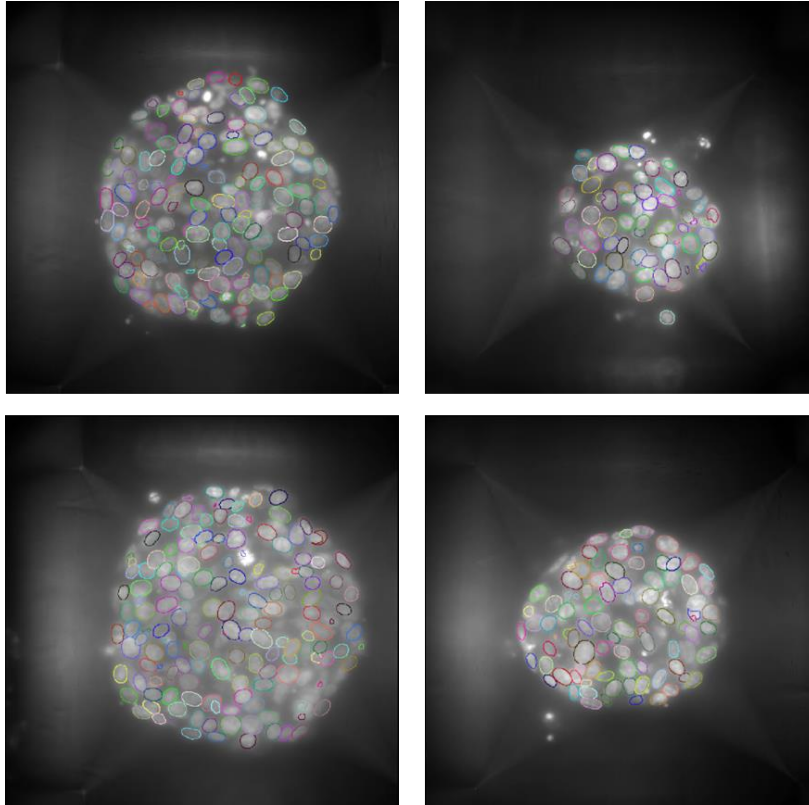
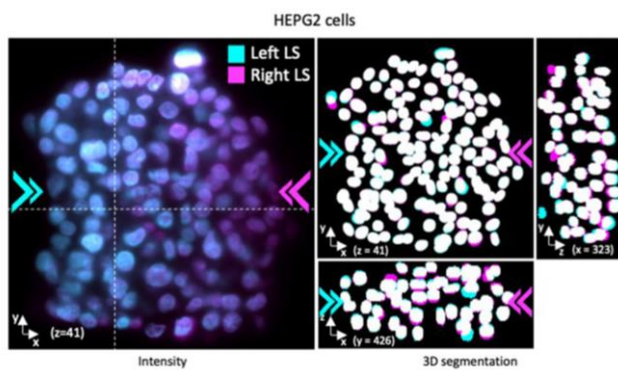


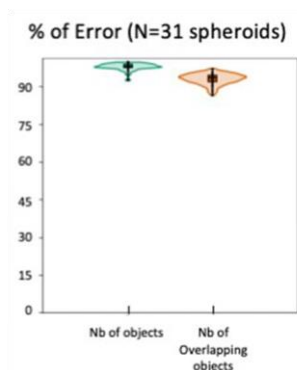
Figure 63: Examples of some of the 96 organoids acquired and segmented presented in (Beghin et al., 2022).

Besides analyzing the quality of the performed segmentation, we were also interested in knowing if the illumination side of the soSPIM system could induce a spatial bias in the segmentation results. For this, we decided to perform a segmentation of images taken with an illumination coming from the left side and from the right side of the same organoids, and compared its segmentation as shown in Figure 64. The cyan segmentation was done using the image taken from the left side, and the magenta segmentation was obtained from the image taken from the right side; in white, we can see the results superposition. From the results shown in Figure 64b, it is observable that there is no spatial bias in the segmentation performed, i.e., the segmentation remains constant with the change in illumination configuration.

One interest in being able to segment and quantify the nuclei organization is that we are then able to pinpoint the location of specific cellular events in this distribution. In Singapore, the team of V. Viasnoff trained a Yolo network (Redmon et al., 2016), a deep-learning network designed to identify the location of specific cellular events such as mitosis. First, I used the segmentations obtained previously with my SOX-DAPI model to perform a 3D reconstruction of the segmented nuclei by surface rendering with the PoCA software (Levet, 2021) developed in the team. I then precisely located the position of the different cell cycle events inside the segmented nuclei, making it possible to obtain extra information that could allow a deeper analysis of cell nuclei development, like measuring the time for a cell to go from G0 to mitosis or using the mitosis position as the starting point for nuclei tracking inside the sphere. Figure 65a shows the 3d rendering reconstruction of the cell nuclei, and Figure 65b and 64c show the 3D localization of each cell cycle phase represented in 3D by a sphere. The color code for the proliferation stages is blue for early G1 (G1-phase, growing cell phase), yellow for G1/S/G2 (S-phase, DNA replication phase and G2-phase, Organelles and proteins develop in preparation for cell division), and red for mitosis, middle and right panels.



a) Comparison of the 3D segmentation obtained between left-side and right-side illumination images



b) Quantification of the difference between the absolute number of detected nuclei in the two illumination directions and nuclei distribution.

Figure 64: Nuclei segmentation spatial bias analysis. a) 3D segmentation comparison between left-side and right-side illumination images. b) Quantification of the difference between the absolute number of nuclei detected in the two illumination directions and nuclei distribution. Edited from (Beghin et al., 2022).

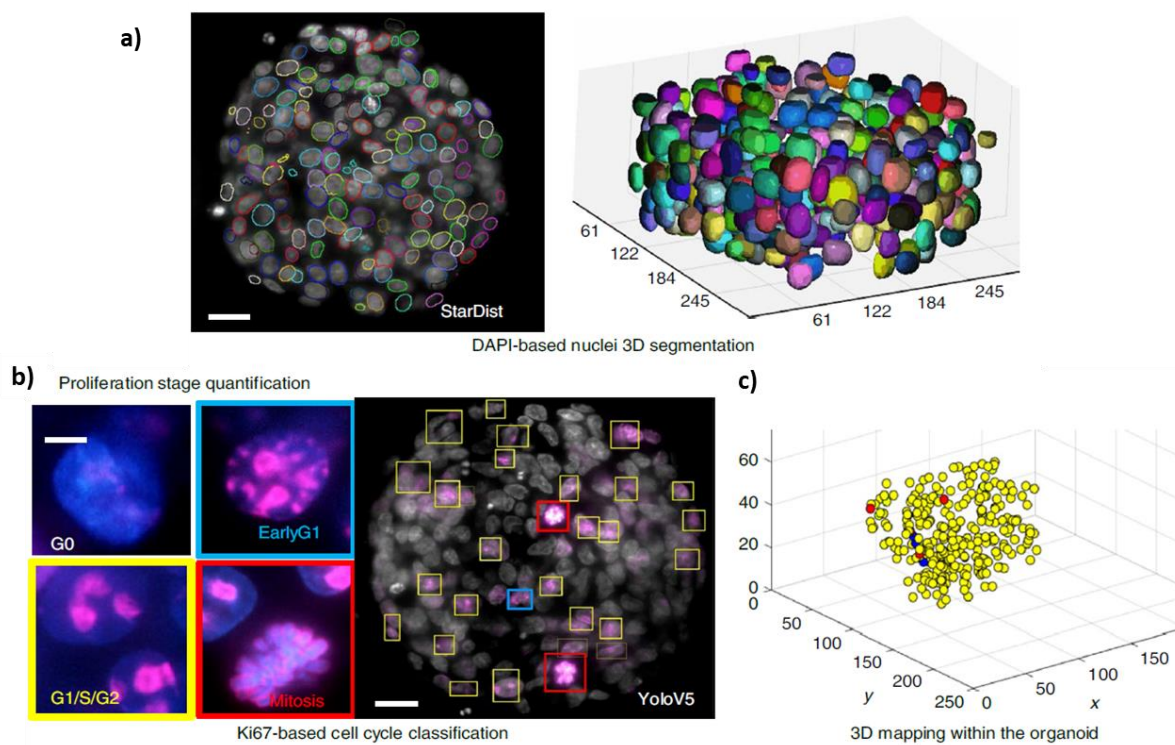


Figure 65: Subcellular quantification of cell proliferation in oncospheres. a) StarDist segmentation contours and 3D reconstruction of segmented nuclei by surface rendering. b) detection of three different stages of cell division. c) 3D localization of individual proliferating nucleus represented in 3D. The colors code for the proliferation stages (blue for early G1, yellow for G1/S/G2, red for mitosis, middle and right panels). Edited from (Beghin et al., 2022).

4.3 The Wavelet transform (WT) as a preprocessing step to improve StarDist segmentation

As in light-sheet microscopy the illumination comes from a side, shadowing is to be expected depending on the size of the sample being imaged. In our case, this results in uneven illumination with part of the sample exhibiting blurred nuclei. Another issue specific to the soSPIM system is that,

sometimes, reflection of the light on the Jewell mirrors will be present in the image and saturates it. Because of this, I decided to use the À-Trous Wavelet transformation as a preprocessing step, as these wavelets have been traditionally used to denoise and filter images in image processing, such as detecting edges and overcoming random noise while preserving sharp details (Dammertz et al., n.d.; Zhang & Li, 2001).

4.3.1 The Stationary Wavelet transform theory

A wavelet (Debnath & Shah, 2015; POLIKAR, 2016) is a mathematical function that divides a given function or continuous-time signal into different scale components providing a time-frequency representation of the signal analyzed. The wavelet transformations (WT) are classified into discrete (DWTs) and continuous wavelet transforms (CWTs) (POLIKAR, 2016). The continuous wavelet transformation was developed as an alternative approach to the Short Time Fourier transform (STFT) (Debnath & Shah, 2015; POLIKAR, 2016). The wavelet analysis is done in a similar way to the STFT analysis, in the sense that the signal is multiplied with a function, the wavelet, and the transform are computed separately for different segments of the time-domain signal (POLIKAR, 2016)

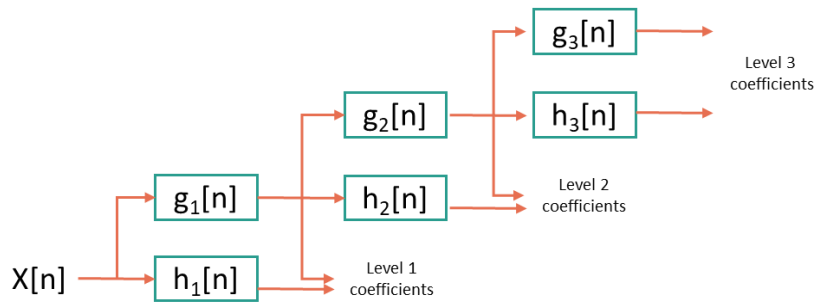
The CWT is a similarity measure between the base functions (wavelets) and the analyzed signal. CWT coefficients refer to the closeness of the signal to the wavelet at the current scale. For example, If the signal has a major component of the frequency corresponding to the current scale, then the wavelet at the current scale will be similar or close to the signal at the particular location where this frequency component occurs (POLIKAR, 2016).

On the other hand, DWT provides sufficient information for analysis and synthesis of the original signal, with a significant reduction in the computation time. DWT is a time-scale representation of a digital signal using digital filtering techniques. When applying this transformation, the signal is passed through a series of high (resp. low) pass filters to analyze its high (resp. low) frequencies (POLIKAR, 2016).

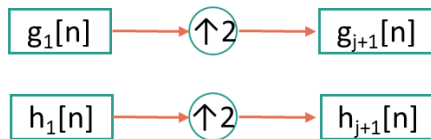
The Stationary wavelet transform (SWT) or À-Trous (with holes) Filter is similar to the DWT. (Al Jumah, 2013; Debnath & Shah, 2015). In conventional DWT, at each level, the input signal is firstly convolved with low $g[m]$ and high $h[m]$ pass filter and then decimated by a factor of two to obtain wavelet transform coefficients. In SWT, the input signals are convolved with low ($g[m]$) and high ($h[m]$) pass filters as in DWT, but no decimation is performed, and the two new sequences have the same length as the original sequence. (Goebel et al., 2008; Qayyum et al., 2017; Somayeh, 2021). The SWT decomposition diagram can be seen in Figure 66.

When applied to an image, SWT can be seen as a convolution between an image and a 3x3 filter, the no-decimation of the image is achieved by padding out the filter with zeroes at each level, as can be seen in Figure 67, resulting in the non-zero entries remaining constant (Al Jumah, 2013; Qayyum et al., 2017).The different coefficients obtained from when applying the SWT to an image can be seen in Figure 68.The wavelet filter used as part of our preprocessing pipeline is the B3 spline (Dammertz et al., 2010; Zhang & Li, 2001), which matrix expression is:

$$\frac{1}{256} \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$



a) Wavelet decomposition tree.



b) SWT filters.

Figure 66: a) Stationary wavelet transformation (SWT) decomposition diagram. Filters in each sample are the up-sampled versions of the previous b) Stationary wavelet filters. Edited from (Hurley, 2018). SWT is commonly used in signal processing and pattern recognition.

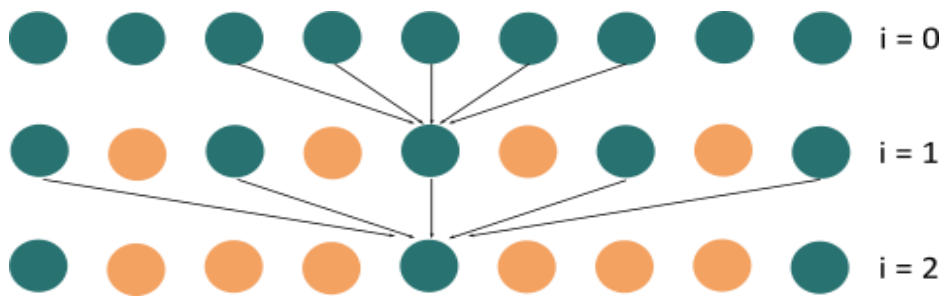


Figure 67: Three levels of the À-Trous wavelet transform. Arrows indicate pixels corresponding to non-zero entries in the filter h_i and are used to compute the center pixel at the next level. Orange dots are positions that full the undecimated wavelet transform would consider but that are skipped by the À-Trous algorithm. Edited from (Dammertz et al., n.d.).

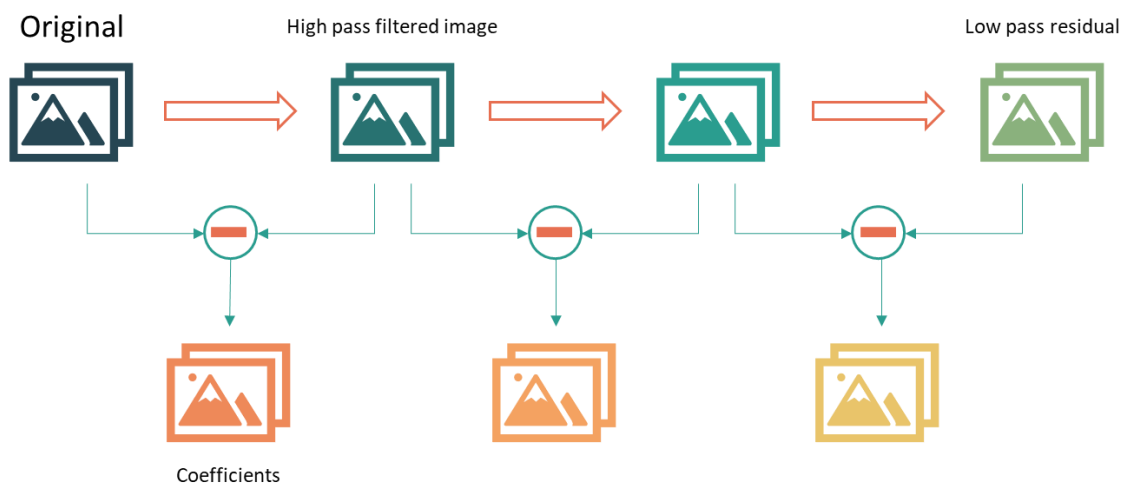


Figure 68: Image scales obtained using À-Trous Wavelet; adding the coefficient to the next image generated will give us the reconstructed version of the original image.

4.3.2 Thresholding of the wavelet coefficients

As an image can be deconstructed using a WT, it can also be reconstructed using the wavelet coefficients. Interestingly, those coefficients can be thresholded before the reconstruction to denoise the original image (Figure 69), as it was demonstrated that Gaussian noise tends to be represented by small values in the wavelet domain (Chang et al., 2000). There exist two main ways to perform thresholding of the wavelet coefficients:

- **Visu-Shrink:** this approach employs a single, universal threshold to all wavelet coefficients. This threshold is designed to remove additive Gaussian noise with high probability, resulting in an overly smooth image appearance. A more visually agreeable result can be obtained by specifying a sigma smaller than the true noise standard deviation (Chang et al., 2000; scikit-image, 2014; van der Walt et al., 2014)
- **Bayes-Shrink:** is an adaptive approach to wavelet soft thresholding where a unique threshold is estimated for each wavelet sub-band. This generally improves what can be obtained with a single threshold (Chang et al., 2000; Don & Johnstone, 1994; scikit-image, 2014; van der Walt et al., 2014).

Importantly, these thresholds also depend on the mode argument, that can be either hard or soft thresholding:

- **hard:** data values for which their absolute value is less than a certain parameter are replaced with a substitute value. Data values with absolute value greater or equal to the thresholding value stay untouched (PyWavelets, 2022).
- **soft:** data values with absolute value less than a certain value are replaced with a substitute value. Data values with absolute value greater or equal to the thresholding value are shrunk toward zero by value (PyWavelets, 2022).

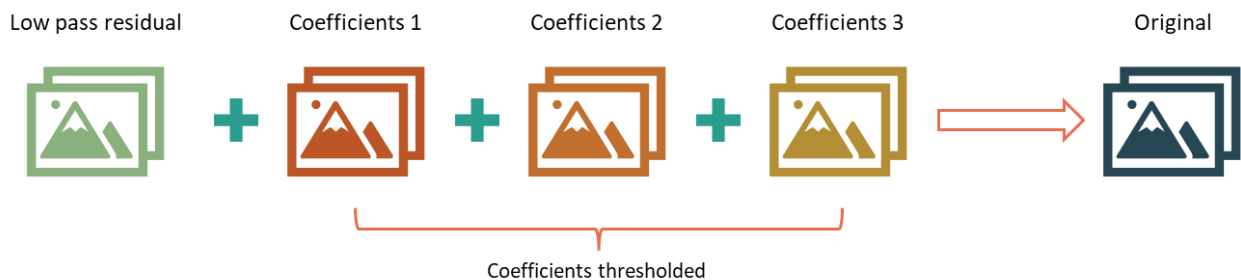


Figure 69: Image reconstruction using thresholded coefficients. During this project we perform the reconstruction of the original image using coefficients 2, 3 and 4.

4.3.3 StarDist 3D implementation using Stationary Wavelet Transform preprocessed images

To perform computation of the SWT, I used a python package specialized in À-Trous transformation, the libatrous python package (Zindy, 2019). In addition to providing different kernels or filters for the wavelets, the libatrous library gives also access to 2D and 3D À-Trous Wavelet transformations by controlling on which dimensions the separable filter kernel is applied.

I decided to apply the À-Trous Wavelet using a B3 Spline 5x5 filter, which enhances the edges of the objects, to our image stacks in the x and y directions to generate what I called the 3D-XY wavelet

dataset. Then, I applied the filter to our images in the x, y, and z directions; I called this set of images the 3D-XYZ wavelet dataset. In both cases, I computed the first five scales of the wavelet transformation as can be seen in Figure 70.

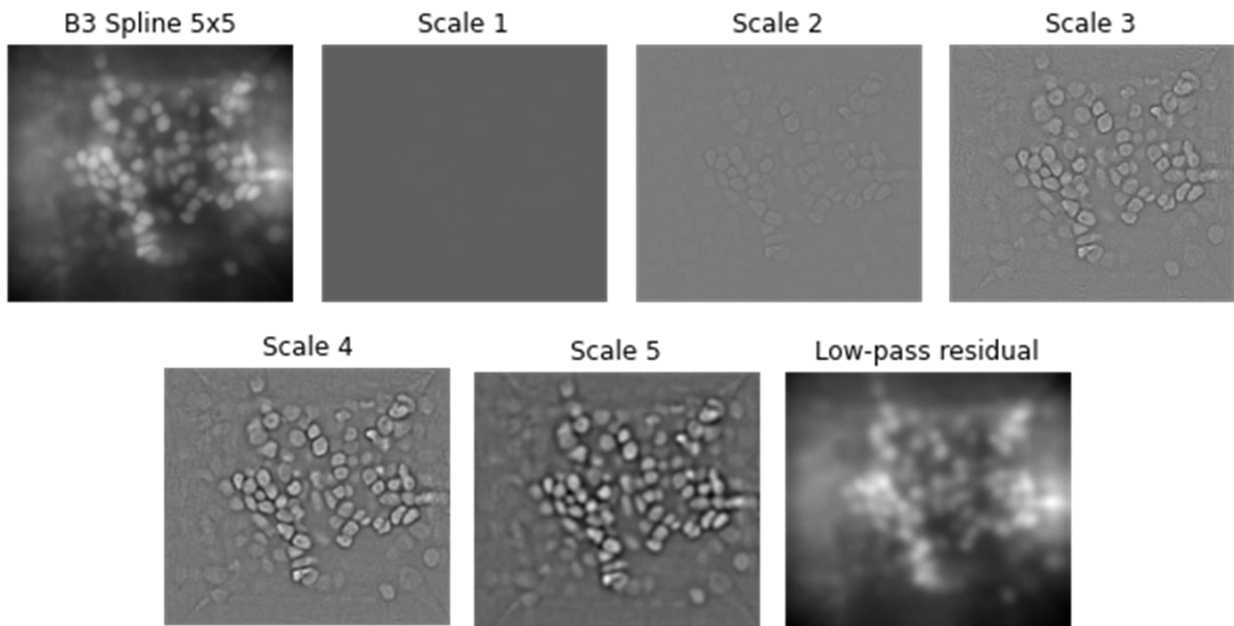


Figure 70: Example of wavelet scales computed using À-Trous algorithm. These images were obtained using the Jupyter Notebook available at (Zindy, 2019).

After computing the WT, I trained four different segmentation models. Because of the practical implementation of StarDist, the two first ones were done using reconstructed RGB images for each frame of the image stack, with channel 1 (resp. 2 and 3) being the SWT scale 2 (resp. 3 and 4), and the two models depending on the wavelet being used (3D-XY and 3D-XYZ). For the third and fourth segmentation models, I reconstructed the 3D image stacks after applying the Baye-Shrink and Visu-Shrink methods to the SWT coefficients. In all cases, I used those reconstructed images combined with the labeled stacks to perform the training.

4.3.3.1 RGB wavelet reconstruction image segmentation

I trained two StarDist 3D segmentation models using the RGB reconstructed images, RGB 3D-XY, and RGB 3D-XYZ as training datasets. After completing the corresponding trainings, lasting 3.5 hours each approximately, I performed the corresponding prediction, lasting around 40 seconds each. To develop these models, I used Original-DAPI plus complementary-sample 1 and 3 for training, complementary-sample 2 for validating, and Complementary-sample 4 for testing. As in 3D StarDist segmentation results using original-DAPI, original-SOX and complementary datasets.

The goal of these trainings was to determine if adding these preprocessing steps would help overcome the segmentation problems present in the areas with high illumination, i.e., the refraction area. Through a visual comparison of the predictions obtained and the fluorescence images, I observed that the prediction that gave a better nuclei segmentation of the ones located in overexposed areas was achieved using the model trained with the RGB 3D-XY dataset. However, the rest of the prediction did not seem to be affected by the wavelet dimensionality used to reconstruct the images.

Figure 71 shows a comparison between a fluorescence image and its RGB 3D-XY and RGB 3D-XYZ versions, while the Figure 72 and Figure 73 show the results after applying the corresponding trained StarDist 3D models, with 1156 and 874 objects segmented respectively. We can observe that RGB 3D-XY performs better than RGB 3D-XYZ since the latter tends to over-segment the nuclei in both the

overexposed areas and the dim ones, which are the regions of extreme importance during our evaluation. The validation plots corresponding to these trainings are shown in Figure 74. The performance of the models trained using RGB reconstructed images gives satisfactory results, with a TP close to 400 in both cases. Its accuracy can also be observed in the metric vs. IoU plot, where the F1, recall, precision, and accuracy metrics are above 0.8 for the first three metrics and above 0.5 for the accuracy.

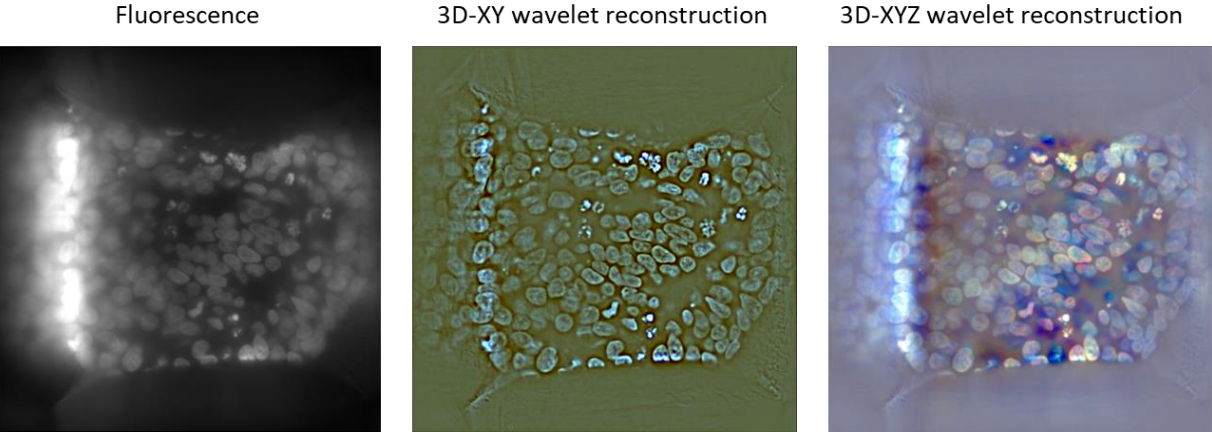


Figure 71: Fluorescence image stack (frame 35) and RGB image reconstructions using 3D-XY wavelet and 3D-XYZ wavelet scale 2, scale 3, and scale 4 as RGB color channels respectively.

Although these results seem promising, the similarity between these and the results obtained with the SOX-DAPI segmentation model is very high, as can be seen in Figure 75 and 76. Nevertheless, I think further experiments must be done using these models to decide if this preprocessing step should be or not added to the segmentation pipeline.

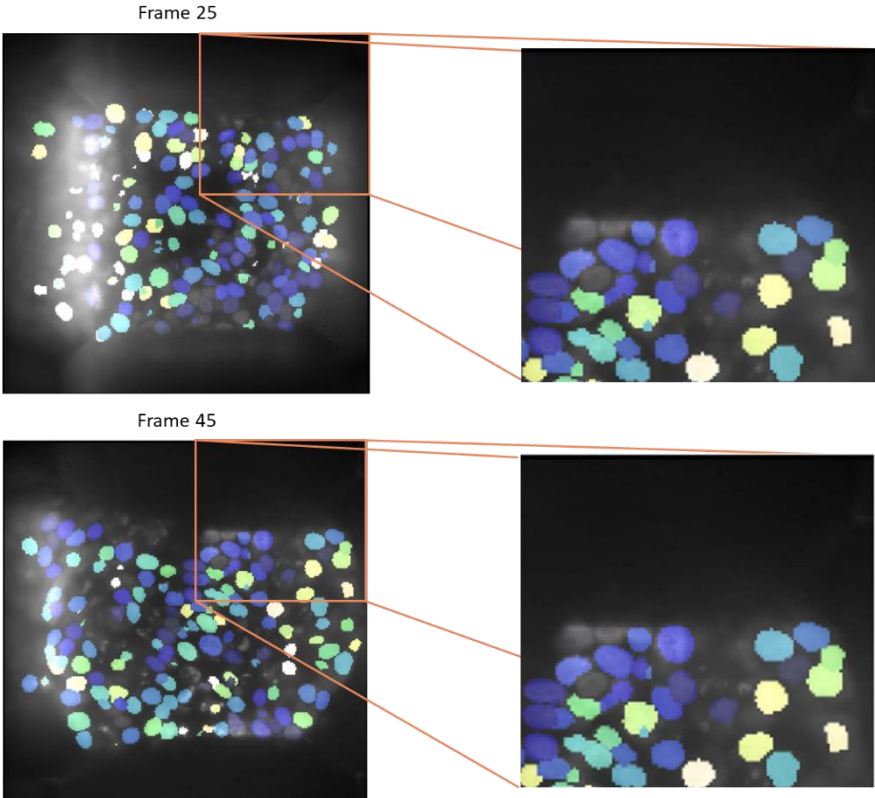


Figure 72: Fluorescence image stack vs. 3D-XY wavelet image training prediction.

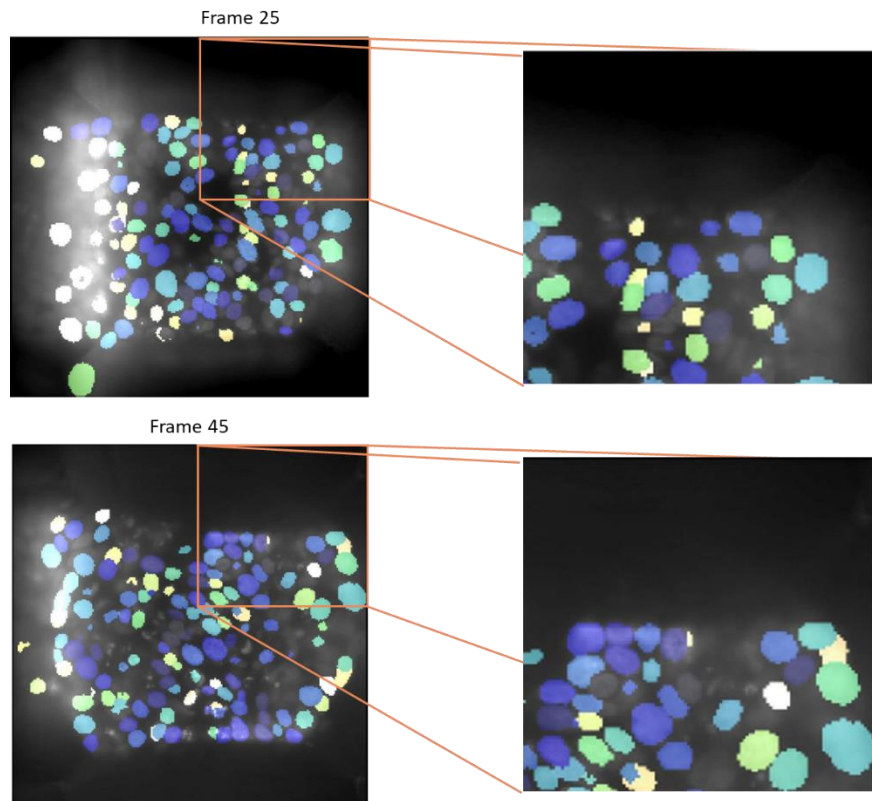
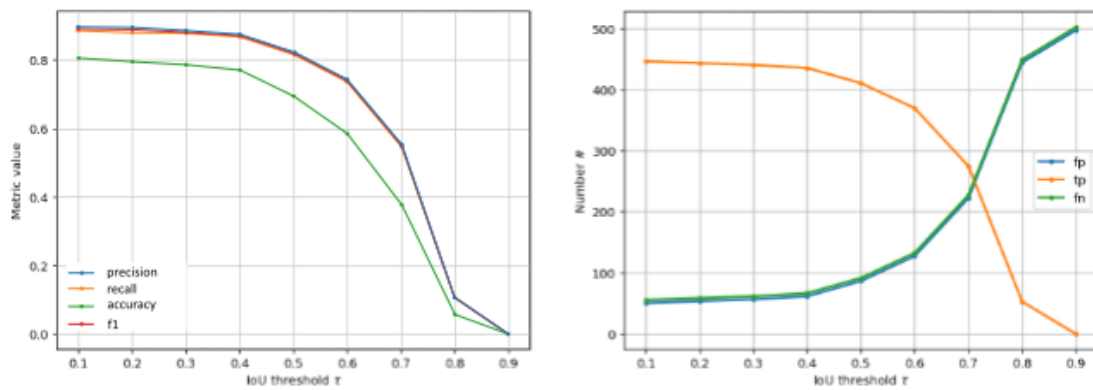
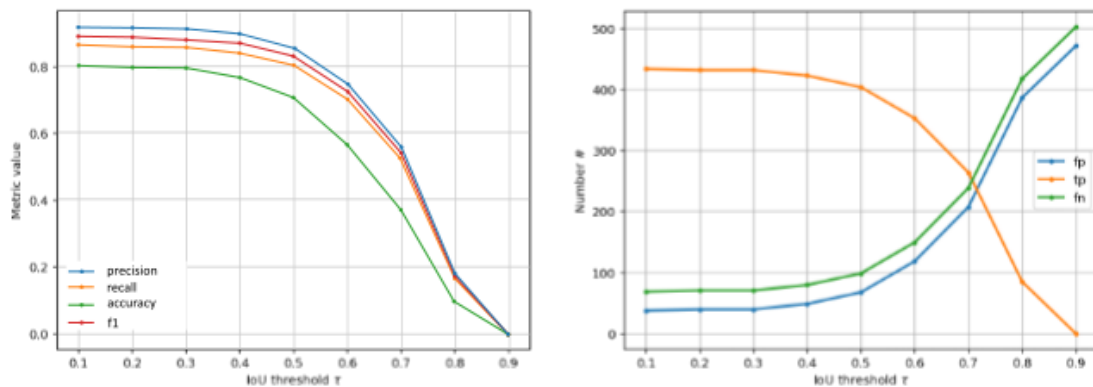


Figure 73: Fluorescence image stack vs. 3D-XYZ wavelet image training prediction.



a) 3D-XY RGB image stacks StarDist 3D training performance



b) 3D-XYZ RGB image stacks StarDist 3D training performance

Figure 74: StarDist 3D training performance comparison between trainings performed with RGB 3D-XY wavelet images (a), and RGB 3D-XYZ wavelet images(b).

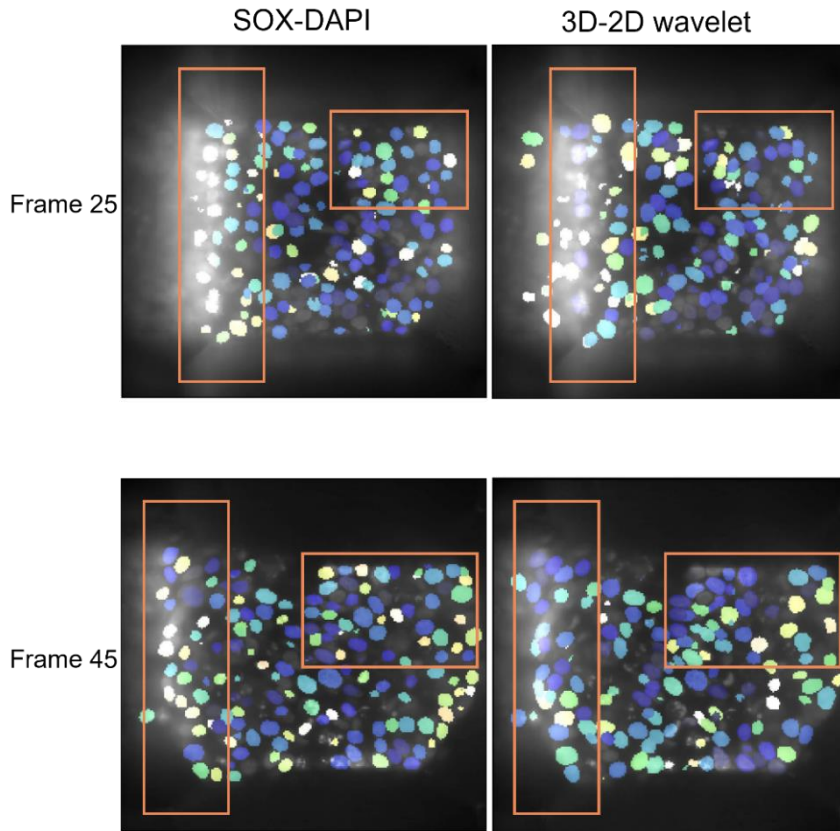


Figure 75: Comparison between SOX-DAPI and RGB3D-XY segmentation results.

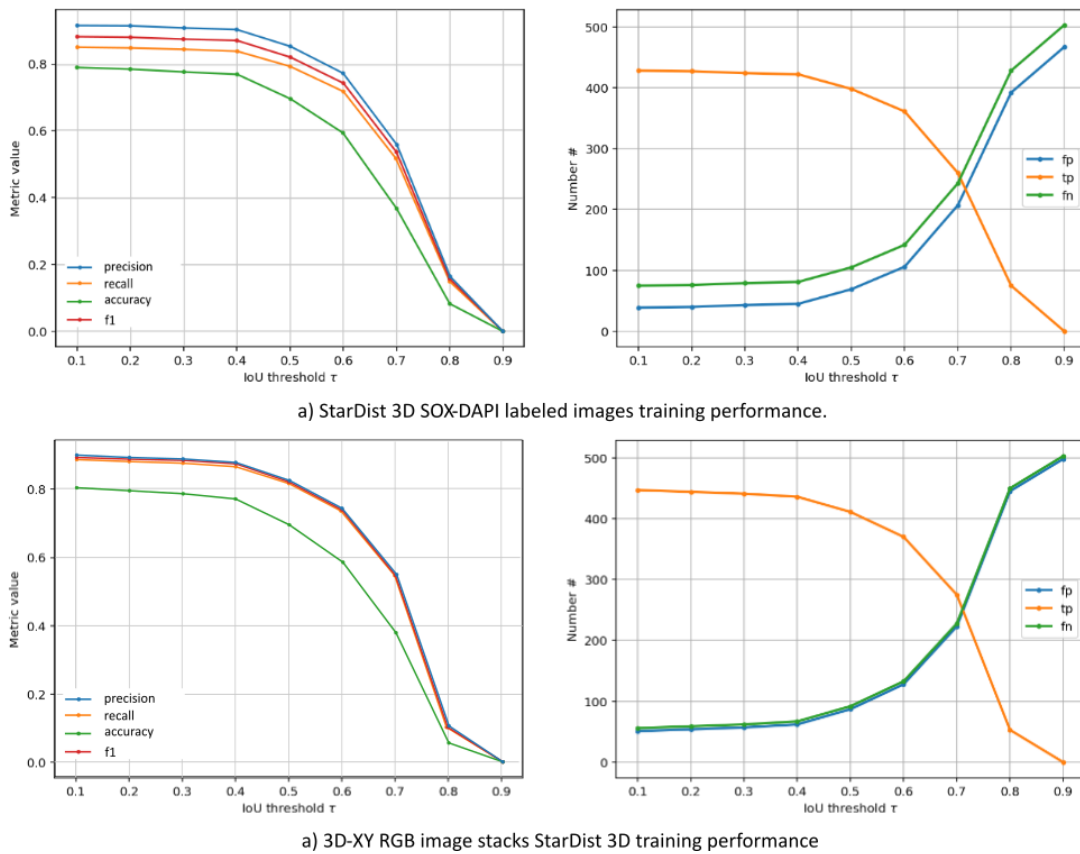


Figure 76: Comparison between SOX-DAPI (a) and RGB 3D-XY (b) performance plots.

4.3.3.2 StarDist 2D results using thresholded wavelet image reconstruction

In parallel to using the WT coefficients to train StarDist, I also investigated the benefit that thresholding those coefficients could bring. I therefore tested different combinations of the thresholding methods and modes and finally decided to use the hard method to perform the respective coefficient thresholding, since it presented the best contrast between object and background. In the case of Visu-Shrink threshold, the thresholding of the images was done using the universal threshold defined as $\lambda_u = \sigma \sqrt{2 \cdot \log(M)}$, where M is the number of pixels on the image and σ the noise variance of such image. The noise variance is computed by $\sigma = \frac{\text{Median}|y_{i,j}|}{0.6745}$. To compute Bayes-Shrink threshold the threshold value for each image is computed using $\text{Threshold} = \sigma^2 / \sigma_x$ (Chang et al., 2000; scikit-image, 2014).

To reconstruct the images, I exclusively used and thresholded the WT coefficient scales 2, 3 and 4. Figure 77 shows examples of the reconstructions performed using Baye Shrink and Visu Shrink with mode hard thresholding methods and different wavelet scales.

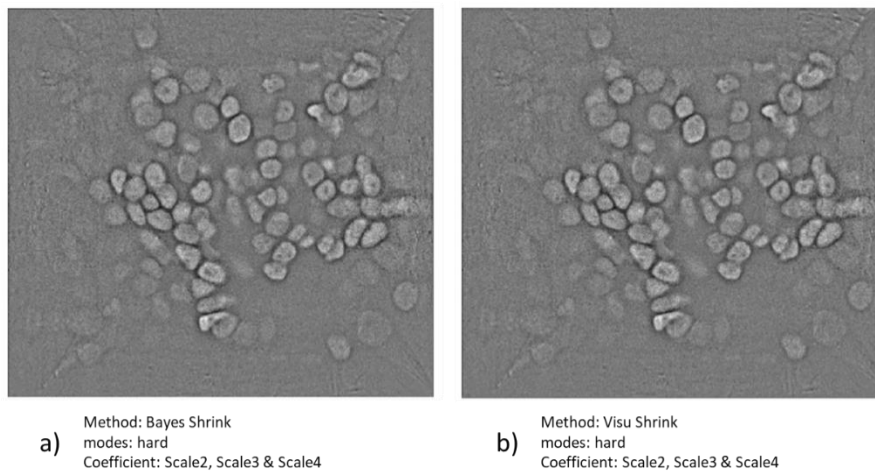


Figure 77: Example of image reconstruction using Bayes-Shrink and Visu-Shrink and threshold with mode hard. a) image reconstruction performed using Bayes-Shrink method with model hard using scale 2, 3 and 4. b) image reconstruction performed using Visu-Shrink method with model hard using scale 2, 3 and 4.

Using the reconstructed images as a training dataset, I trained StarDist 2D using 200 epochs. To develop this model, I used Original-DAPI plus complementary- datasets for training, and validating, and Complementary-sample 4 for testing. In this case, the training and validation samples were chosen randomly by StarDist 2D Jupyter notebook as in 2D StarDist segmentation results using original-DAPI and DSB2018 datasets.

The prediction obtained and its comparison with the 2D ground truth can be observed in Figure 78. It is noticeable that both models performed well in the areas that present over exposed sections. In Figure 78, the white nuclei show the intersection between the 2D GT, and the segmentation performed using the different models. These images demonstrate that the Visu-Shrink-hard model performs slightly better because it contains close to 12% less false negatives than Bayes-Shrink.

Figure 79 shows the training performance for the two models. The behavior of the metrics indicates very good values even above the IoU = 0.8, a fact also reflected in the high number of true positives nuclei computed. In both cases, the value of the TP does not decrease drastically until reaching an IoU = 0.8, which is considered a very high value of IoU.

4.4 Discussion

As the use of 3D cell cultures becomes more and more widespread in the scientific community, the limitations of analyzing them with traditional methods puts an obstacle in the research pipeline. On the other hand, deep-learning methods, such as StarDist or Cellpose, have demonstrated their ability to achieve accurate results in segmenting nuclei. However, the segmentation quality of these models strongly depends on the size, accuracy and diversity of the training dataset (Stringer et al., 2020). Therefore, it is necessary to analyze the variables that affect the algorithm's performance. The results presented in this thesis allow us to see how a dataset's diversity can highly influence a segmentation algorithm's behavior.

As seen in the StarDist 2D implementation section, I demonstrated that the diversification in experimental conditions, nuclei density, and nuclei size is critical to improve the results of StarDist in 2D. I thus illustrated that the best combination for a training dataset is a dataset exhibiting a wide variety of sizes and conditions, while still ensuring that some of your own data are present.

I then demonstrated a similar pattern for StarDist in 3D. In this case I did not have access to any free ground truth datasets. Nevertheless, I was able to bring some diversity to the model by using both the SOX and DAPI immunostaining labeling in the training set. The main objective to train these models was to be able to automatically segment 3D cell cultures labeled with DAPI. Therefore, training a model with only the SOX labeled image stacks gave poor results. It was to be expected for two reasons: first, the images acquired for the DAPI labeling were too different from the SOX ones, and the network was not acquainted with them. Second, the training set was too small, as only three stacks were labeled with SOX. However, training StarDist 3D with all my labeled images, combining both SOX and DAPI labeling, resulted in good enough segmentations to be used to automatically segment close to one hundred 3D cell cultures (Beghin et al., 2022). I then demonstrated that one could use this accurate nuclear segmentation models to bring interested biological insights, for instance determining where in the 3D cell culture are located nuclei at different development stage.

Finally, I wanted to see if using wavelet as preprocessing step in the segmentation pipeline could improve the segmentation accuracy by overcoming some issues inherent to the soSPIM system such as shadowing or uneven illumination. To enhance essential features in our images, I decided to use the stationary wavelet transform as a preprocessing step, in an attempt to overcome the uneven illumination and enhance nuclei edges. Thanks to the wavelet transformation applied to the images, its illumination became more homogeneous, and nuclei edges sharper, thus overcoming one of the major problems presented in our images. Furthermore, looking at the results obtained from the wavelet segmentation models, we can observe that objects were better segmented in the overexposed regions compared to segmentations done using the original images, tackling one of the main problems when segmenting soSPIM images. Although the segmentation results obtained from wavelet models in other areas of the images did not show substantial improvement; however, those are not the areas we are mainly concerned about. This gives us the idea that a good perspective for the future of these projects could be to develop a StarDist segmentation model using the original images and the wavelet images as part of the training dataset to see if this model can segment fluorescence images without the need to preprocess them. Combining wavelet and original images could be a way to generalize our training dataset even more and obtain a better segmentation of the overexposed areas of the fluorescence images.

StarDist 2D segmentation models developed using wavelet reconstructed images, RGB 3D-XY and RGB 3D-XYZ, seem to perform the same as the models developed using the original + DSB2018 images, even though the reconstructed images appeared to have better illumination distribution and sharper nuclei edges. This could be because the wavelet dataset is not general enough or because the algorithm's

limit has been reached. Additionally, the segmentations performed using Bayes-Shrink, and Visu-shrink models present good segmentation in the overexposed areas; however, due to lack of time, I was not able to explore the possible result of 3D segmentation, although I suspect that if StarDist 2D gave very good results, using this preprocessing in 3D segmentation will also give promising results

5 | *Conclusions and perspectives*

As 3D cultures are increasingly adopted by the scientific community, the segmentation and quantification of its constituents in 3D has become critical. In parallel, deep learning-based methods have taken by storm the biomedical field with reasons as they allow, under certain conditions, a fast and automatic segmentation of structures exhibiting complex shapes. Naturally, deep-learning methods such as CellPose (Stringer et al., 2020) and StarDist (Weigert et al., 2020) have been developed to segment nuclei in 3D with great successes. But while successful some limitations still exist, the main one being linked to the availability of labeled ground truth in 3D. In the course of my PhD, I developed a 3D nuclei segmentation pipeline that resulted in three main contributions, along with some interesting insights and results.

My first main contribution is related to the availability of ground truth datasets for nuclei segmentation in 3D. At the time I started it was basically non-existent. In the context of a collaboration with the team of V. Viasnoff located in Singapore, I labeled several 3D cultures cells to generate a ground truth dataset of sufficient size for training an accurate StarDist model in 3D. This corpus contains 4 neuroectoderm organoids image stacks, 1 neuroectoderm organoid, and 3 cancer spheroids, and represents more than 4,500 individual nuclei in 3D. I have specifically chosen these samples to ensure that they were from different experimental conditions and exhibited varying labeling, nuclei density, and cell type, bringing diversity to the training set. Because of its variety and size, this ground truth dataset stands as the only one of its kind and will benefit to the whole community. Hopefully, it will become a reference for developing new segmentation algorithms and improving existing ones.

Although annotating is very time-consuming, developing the right tools to make this task easier, faster, and error-proof has not caught enough attention from the scientific community. My second main contribution have been the development of a Napari plugin called “Napari Annotation Helper” (NAHP). Starting from an image stack already labeled in 2D (by using StarDist in 2D for instance), this plugin allows users to immediately know what was the last finished 3D label, to identify the remaining 2D labels to be treated and to easily identify annotation errors. It also helps users to obtain significant statistical information from the annotated images. Ultimately, this tool facilitates the 3D annotation process and is therefore beneficial to our team and can motivate other research teams to carry out new labeling.

Finally, my last main contribution has been to use a trained StarDist 3D model in production and demonstrate that it can be used to quantify interesting biological phenomena. This model was trained with the ground truth dataset I labeled and was used to segment almost 100 organoids and spheroids acquired by the team in Singapore. These segmentations were later used to quantify several features of both the soSPIM system and the biological samples. For instance, I was able to pinpoint different cell cycle events inside the segmented nuclei distribution and obtain extra information allowing deeper analysis of the cell nuclei development. In another example, I segmented one organoid acquired twice with a light sheet coming from a different direction. Comparison of the two segmentations shown that we had similar results, demonstrating that using multi-view reconstruction was unnecessary.

Nevertheless, I have found several more interesting insights and results by training a multitude of StarDist models, both in 2D and 3D. I first verified a well-known fact in the deep-learning field: you should always train a network with some of your data. When I tried the pre-trained StarDist model in

2D, that was trained with the Data Science Bowl 2018 dataset, I obtained poorly segmented nuclei. On the contrary, combining the DSB2018 dataset with some of the images belonging to our ground truth dataset gave really accurate results. This was due to increased diversity, demonstrating that having a dataset with images taken under different experimental conditions and with different nuclei density was able to achieve very good 2D segmentations. Similarly, training StarDist in 3D with our ground truth dataset brought better results when I used two different labels.

Another interesting result was achieved when I decided to preprocess the data before training StarDist. Because of its advantages, I used a stationary wavelet transform since it tends to homogenize intensities in the image, resulting in enhancing the nuclei's edges. It therefore allows to overcome uneven illumination, a common artifact in light sheet microscopy. By having more nuclei segmented in the saturated region of the images, I managed to slightly increment the segmentation accuracy.

I then decided to threshold the wavelet coefficients to improve the model accuracy even more, as this technique is widely used in image processing for denoising and filtering. While I was unfortunately only able to test it in 2D because of a lack of time, the results seemed very promising with a trained model exhibiting the best metrics of all my trainings. This is again certainly related to a better segmentation in overexposed areas. These results lay the groundwork for promising future research, and I hope that in the future this work will be extended to 3D.

Another promising future direction of our research is the extension of the NAHP plugin by implementing new functions to improve the user experience and convert it into an essential tool for annotators. For example, functions such as visualizing the final annotations, incorporating wavelet transformation functions to visualize better images with challenging illumination, and coloring nuclei according to certain user-selected properties, among others, can be added to NAPH to increment the user's capability to obtain extra annotations from the ones performed and to be able to obtain extract statistical information from the images to annotate.

Bibliography

- [1] *3D Bin Packing*. (2020, April 6). <https://github.com/enzoruz/3dbinpacking>
- [2] Abien Fred Agarap. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. <https://doi.org/10.48550/arXiv.1803.08375>
- [3] Aggarwal, C. C. (2018). *Neural Networks and Deep Learning: A Textbook*. Springer International Publishing. <https://doi.org/10.1007/978-3-319-94463-0>
- [4] Al Jumah, A. (2013). Denoising of an Image Using Discrete Stationary Wavelet Transform and Various Thresholding Techniques. *Journal of Signal and Information Processing*, 04(01), 33–41. <https://doi.org/10.4236/jsip.2013.41004>
- [5] Alom, M. Z., Yakopcic, C., Taha, T. M., & Asari, V. K. (2018). Nuclei Segmentation with Recurrent Residual Convolutional Neural Networks based U-Net (R2U-Net). *NAECON 2018 - IEEE National Aerospace and Electronics Conference*, 228–233. <https://doi.org/10.1109/NAECON.2018.8556686>
- [6] Arzt, M., Deschamps, J., Schmied, C., Pietzsch, T., Schmidt, D., Tomancak, P., Haase, R., & Jug, F. (2022). LABKIT: Labeling and Segmentation Toolkit for Big Image Data. *Frontiers in Computer Science*, 4, 777728. <https://doi.org/10.3389/fcomp.2022.777728>
- [7] Avinash Sharma V. (2017, March 30). Understanding Activation Functions in Neural Networks. *The Theory Of Everything*. <https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0#:~:text=So%20what%20does%20an%20artificial,the%20flow%20for%20a%20moment%20>.
- [8] Aytac, S. (2018). Using Color Blindness Simulator During User Interface Development for Accelerator Control Room Applications [PDF]. *Proceedings of the 16th Int. Conf. on Accelerator and Large Experimental Control Systems, ICALEPCS2017*, 6 pages, 1.960 MB. <https://doi.org/10.18429/JACOW-ICALEPCS2017-THSH103>
- [9] Bankhead, P., Loughrey, M. B., Fernández, J. A., Dombrowski, Y., McArt, D. G., Dunne, P. D., McQuaid, S., Gray, R. T., Murray, L. J., Coleman, H. G., James, J. A., Salto-Tellez, M., & Hamilton, P. W. (2017). QuPath: Open source software for digital pathology image analysis. *Scientific Reports*, 7(1), 16878. <https://doi.org/10.1038/s41598-017-17204-5>
- [10] Beghin, A., Greci, G., Sahni, G., Guo, S., Rajendiran, H., Delaire, T., Mohamad Raffi, S. B., Blanc, D., de Mets, R., Ong, H. T., Galindo, X., Monet, A., Acharya, V., Racine, V., Levet, F., Galland, R., Sibarita, J.-B., & Viasnoff, V. (2022). Automated high-speed 3D imaging of organoid cultures with multi-scale phenotypic quantification. *Nature Methods*. <https://doi.org/10.1038/s41592-022-01508-0>

- [11] Behl, H. S., Baydin, A. G., Gal, R., Torr, P. H. S., & Vineet, V. (2020). *AutoSimulate: (Quickly) Learning Synthetic Data Generation* (arXiv:2008.08424). arXiv. <http://arxiv.org/abs/2008.08424>
- [12] Berg, S., Kutra, D., Kroeger, T., Straehle, C. N., Kausler, B. X., Haubold, C., Schiegg, M., Ales, J., Beier, T., Rudy, M., Eren, K., Cervantes, J. I., Xu, B., Beuttenmueller, F., Wolny, A., Zhang, C., Koethe, U., Hamprecht, F. A., & Kreshuk, A. (2019). ilastik: Interactive machine learning for (bio)image analysis. *Nature Methods*, 16(12), 1226–1232. <https://doi.org/10.1038/s41592-019-0582-9>
- [13] Booz Allen Hamilton, Inc. (2018). *2018 Data Science Bowl*. <https://www.kaggle.com/competitions/data-science-bowl-2018/data>
- [14] Broad Institute. (2021, June 29). *Introducing Piximi Annotator, an easily accessible, user-friendly annotator tool*. <https://carpenter-singh-lab.broadinstitute.org/blog/introducing-piximi-annotator>
- [15] Caicedo, J. C., Roth, J., Goodman, A., Becker, T., Karhohs, K. W., Broisin, M., Molnar, C., McQuin, C., Singh, S., Theis, F. J., & Carpenter, A. E. (2019). Evaluation of Deep Learning Strategies for Nucleus Segmentation in Fluorescence Images. *Cytometry Part A*, 95(9), 952–965. <https://doi.org/10.1002/cyto.a.23863>
- [16] Cella Zanacchi, F., Lavagnino, Z., Perrone Donnorso, M., Del Bue, A., Furia, L., Faretta, M., & Diaspro, A. (2011). Live-cell 3D super-resolution imaging in thick biological samples. *Nature Methods*, 8(12), 1047–1049. <https://doi.org/10.1038/nmeth.1744>
- [17] Chang, S. G., Bin Yu, & Vetterli, M. (2000). Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9), 1532–1546. <https://doi.org/10.1109/83.862633>
- [18] Chen, J., Ding, L., Viana, M. P., Lee, H., Sluezwski, M. F., Morris, B., Hendershott, M. C., Yang, R., Mueller, I. A., & Rafelski, S. M. (2018). *The Allen Cell and Structure Segmenter: A new open source toolkit for segmenting 3D intracellular structures in fluorescence microscopy images* [Preprint]. Cell Biology. <https://doi.org/10.1101/491035>
- [19] Chollet, F. (2018). *Deep learning with Python*. Manning Publications Co.
- [20] Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016). 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, & W. Wells (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016* (Vol. 9901, pp. 424–432). Springer International Publishing. https://doi.org/10.1007/978-3-319-46723-8_49
- [21] Cole Matt R. (2019). *Deep Learning with C#, .Net and Kelp.Net*.
- [22] Dammertz, H., Sewtz, D., Hanika, J., & Lensch, H. P. A. (2010). *Edge-Avoiding À-Trous Wavelet Transform for fast Global Illumination Filtering*. 9.

- [23] Debnath, L., & Shah, F. A. (2015). *Wavelet Transforms and Their Applications*. Birkhäuser Boston. <https://doi.org/10.1007/978-0-8176-8418-1>
- [24] *DeepCell Label*. (2016). <https://label.deepcell.org/>
- [25] Don, D. L., & Johnstone, I. M. (1994). *Ideal spatial adaptation by wavelet shrinkage*. 31.
- [26] Dube, E., Kanavathy, L. R., & Avenue, C. (n.d.). *OPTIMIZING THREE-DIMENSIONAL BIN PACKING THROUGH SIMULATION*. 7.
- [27] Dunn, K. W., & Young, P. A. (2006). Principles of Multiphoton Microscopy. *Nephron Experimental Nephrology*, 103(2), e33–e40. <https://doi.org/10.1159/000090614>
- [28] Early Stopping. (n.d.). *Papers With Code*. Retrieved April 23, 2022, from <https://paperswithcode.com/method/early-stopping>
- [29] Engelbrecht, C. J., & Stelzer, E. H. (2006). Resolution enhancement in a light-sheet-based microscope (SPIM). *Optics Letters*, 31(10), 1477. <https://doi.org/10.1364/OL.31.001477>
- [30] Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., Deubner, J., Jäckel, Z., Seiwald, K., Dovzhenko, A., Tietz, O., Dal Bosco, C., Walsh, S., Saltukoglu, D., Tay, T. L., Prinz, M., Palme, K., Simons, M., ... Ronneberger, O. (2019). U-Net: Deep learning for cell counting, detection, and morphometry. *Nature Methods*, 16(1), 67–70. <https://doi.org/10.1038/s41592-018-0261-2>
- [31] Galland, R., Greci, G., Aravind, A., Viasnoff, V., Studer, V., & Sibarita, J.-B. (2015). 3D high- and super-resolution imaging using single-objective SPIM. *Nature Methods*, 12(7), 641–644. <https://doi.org/10.1038/nmeth.3402>
- [32] Ghosh, S. K. (2007). *Visibility Algorithms in the Plane*. Cambridge.
- [33] Goebel, E. R., Siekmann, J., & Wahlster, W. (2008). *Lecture Notes in Artificial Intelligence*. 427.
- [34] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. The MIT Press.
- [35] Guerrero-Pena, F. A., Fernandez, P. D. M., Ren, T. I., Yui, M., Rothenberg, E., & Cunha, A. (2018). Multiclass Weighted Loss for Instance Segmentation of Cluttered Cells. *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2451–2455. <https://doi.org/10.1109/ICIP.2018.8451187>
- [36] Hasan, S. M. K., & Linte, C. A. (2019). U-NetPlus: A Modified Encoder-Decoder U-Net Architecture for Semantic and Instance Segmentation of Surgical Instrument. *ArXiv:1902.08994 [Cs]*. <http://arxiv.org/abs/1902.08994>
- [37] Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., Pal, C., Jodoin, P.-M., & Larochelle, H. (2017). Brain tumor segmentation with Deep Neural Networks. *Medical Image Analysis*, 35, 18–31. <https://doi.org/10.1016/j.media.2016.05.004>

- [38] He, F., Liu, T., & Tao, D. (2019). *Why ResNet Works? Residuals Generalize* (arXiv:1904.01367). arXiv. <http://arxiv.org/abs/1904.01367>
- [39] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- [40] Hollandi, R., Moshkov, N., Paavolainen, L., Tasnadi, E., Piccinini, F., & Horvath, P. (2022). Nucleus segmentation: Towards automated solutions. *Trends in Cell Biology*, 32(4), 295–310. <https://doi.org/10.1016/j.tcb.2021.12.004>
- [41] Hosang, J., Benenson, R., & Schiele, B. (2017). Learning Non-maximum Suppression. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6469–6477. <https://doi.org/10.1109/CVPR.2017.685>
- [42] Huisken, J., Swoger, J., Bene, F. D., Wittbrodt, J., & Stelzer, E. H. K. (2004). *Optical Sectioning Deep Inside Live Embryos by Selective Plane Illumination Microscopy*. 305, 18.
- [43] Jensen, C., & Teng, Y. (2020). Is It Time to Start Transitioning From 2D to 3D Cell Culture? *Frontiers in Molecular Biosciences*, 7, 33. <https://doi.org/10.3389/fmolb.2020.00033>
- [44] Jerome, W. G., & Price, R. L. (Eds.). (2018). *Basic Confocal Microscopy*. Springer International Publishing. <https://doi.org/10.1007/978-3-319-97454-5>
- [45] Jo, T. (2021). *Machine Learning Foundations: Supervised, Unsupervised, and Advanced Learning*.
- [46] Kai, Y., & Kaizhu, H. (2021). Scaffold-A549: A Benchmark 3D Fluorescence Image Dataset for Unsupervised Nuclei Segmentation. *Cognitive Computation*, 13, 1603–1608. <https://doi.org/10.1007/s12559-021-09944-4>
- [47] Kaiming He, Georgia Gkioxari, Piotr Dollar, & Ross Girshick. (2017). Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*.
- [48] Kapałczyńska, M., Kolenda, T., Przybyła, W., Zajączkowska, M., Teresiak, A., Filas, V., Ibbs, M., Bliźniak, R., Łuczewski, Ł., & Lamperska, K. (2016). 2D and 3D cell cultures – a comparison of different types of cancer cell cultures. *Archives of Medical Science*. <https://doi.org/10.5114/aoms.2016.63743>
- [49] Kapuscinski, J. (1995). DAPI: A DNA-Specific Fluorescent Probe. *Biotechnic & Histochemistry*, 70(5), 220–233. <https://doi.org/10.3109/10520299509108199>
- [50] Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2017). *On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima* (arXiv:1609.04836). arXiv. <http://arxiv.org/abs/1609.04836>
- [51] Kotsiantis, S. B. (2017). *Supervised Machine Learning: A Review of Classification Techniques*. 20.

- [52] Ku Wee Kiat. (2018, December 6). *DEEP LEARNING BEST PRACTICES: CHECKPOINTING YOUR DEEP LEARNING MODEL TRAINING*. <https://nusit.nus.edu.sg/services/hpc-newsletter/deep-learning-best-practices-checkpointing-deep-learning-model-training/>
- [53] Levet, F. (2021). *PoCA: Point Cloud Analyst*.
- [54] Lin, Z., Wei, D., Petkova, M. D., Wu, Y., Ahmed, Z., K, K. S., Zou, S., Wendt, N., Boulanger-Weill, J., Wang, X., Dhanyasi, N., Arganda-Carreras, I., Engert, F., Lichtman, J., & Pfister, H. (2021). *NucMM Dataset: 3D Neuronal Nuclei Instance Segmentation at Sub-Cubic Millimeter Scale* (Vol. 12901, pp. 164–174). https://doi.org/10.1007/978-3-030-87193-2_16
- [55] Lindbloom, B. J. (2017, April 8). Delta E (CIE 2000). *Bruce Lindbloom*. http://www.brucelindbloom.com/index.html?Eqn_DeltaE_CIE2000.html
- [56] Maarouf, W. F., Barbar, A. M., & Owayjan, M. J. (2008). A New Heuristic Algorithm for the 3D Bin Packing Problem. In K. Elleithy (Ed.), *Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering* (pp. 342–345). Springer Netherlands. https://doi.org/10.1007/978-1-4020-8735-6_64
- [57] Mahesh, B. (2018). *Machine Learning Algorithms—A Review*. 9(1), 7.
- [58] Mela, C. A., & Liu, Y. (2021). Application of convolutional neural networks towards nuclei segmentation in localization-based super-resolution fluorescence microscopy images. *BMC Bioinformatics*, 22(1), 325. <https://doi.org/10.1186/s12859-021-04245-x>
- [59] Müller, D., Rey, I. S., & Kramer, F. (2022). *Towards a Guideline for Evaluation Metrics in Medical Image Segmentation*. <https://doi.org/10.5281/ZENODO.5877797>
- [60] Narotamo, H. (2019). Segmentation of Cell Nuclei in Fluorescence Microscopy Images Using Deep Learning. In *Pattern Recognition and Image Analysis* (pp. 53–64).
- [61] Naylor, P., Lae, M., Reyal, F., & Walter, T. (2019). Segmentation of Nuclei in Histopathology Images by Deep Regression of the Distance Map. *IEEE Transactions on Medical Imaging*, 38(2), 448–459. <https://doi.org/10.1109/TMI.2018.2865709>
- [62] Ong, K., Haw, S.-C., & Ng, K.-W. (2019). Deep Learning Based-Recommendation System: An Overview on Models, Datasets, Evaluation Metrics, and Future Trends. *Proceedings of the 2019 2nd International Conference on Computational Intelligence and Intelligent Systems*, 6–11. <https://doi.org/10.1145/3372422.3372444>
- [63] O’Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv:1511.08458 [Cs]*. <http://arxiv.org/abs/1511.08458>
- [64] Padilla, R., Netto, S. L., & da Silva, E. A. B. (2020). A Survey on Performance Metrics for Object-Detection Algorithms. *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 237–242. <https://doi.org/10.1109/IWSSIP48289.2020.9145130>

- [65] POLIKAR, R. (2016). *THE WAVELET TUTORIAL*. 67.
- [66] PyWavelets. (2022, March 11). *Thresholding functions*. <https://pywavelets.readthedocs.io/en/latest/ref/thresholding-functions.html>
- [67] Qayyum, H., Majid, M., Anwar, S. M., & Khan, B. (2017). Facial Expression Recognition Using Stationary Wavelet Transform Features. *Mathematical Problems in Engineering*, 2017, 1–9. <https://doi.org/10.1155/2017/9854050>
- [68] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection* (arXiv:1506.02640). arXiv. <http://arxiv.org/abs/1506.02640>
- [69] Riverbank Computing. (n.d.). *What is PyQt?* <https://www.riverbankcomputing.com/software/pyqt/>
- [70] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *ArXiv:1505.04597 [Cs]*. <http://arxiv.org/abs/1505.04597>
- [71] Ryobi Systems Co. (2020). *Visolve: The assistive software for people with color blindness* Company's logo. <https://www.ryobi.co.jp/products/visolve/en/>
- [72] Sakalem, M. E., De Sibio, M. T., da Costa, F. A. da S., & de Oliveira, M. (2021). Historical evolution of spheroids and organoids, and possibilities of use in life sciences and medicine. *Biotechnology Journal*, 16(5), 2000463. <https://doi.org/10.1002/biot.202000463>
- [73] Sanderson, J. B. (2018). *Understanding light microscopy* (First edition). John Wiley & Sons.
- [74] Sarkar, A., & Hochedlinger, K. (2013). The Sox Family of Transcription Factors: Versatile Regulators of Stem and Progenitor Cell Fate. *Cell Stem Cell*, 12(1), 15–30. <https://doi.org/10.1016/j.stem.2012.12.007>
- [75] Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.-Y., White, D. J., Hartenstein, V., Eliceiri, K., Tomancak, P., & Cardona, A. (2012). Fiji: An open-source platform for biological-image analysis. *Nature Methods*, 9(7), 676–682. <https://doi.org/10.1038/nmeth.2019>
- [76] Schmidt, U. (2021a, March 15). *Stardist 2D examples*. https://github.com/stardist/stardist/blob/master/examples/2D/1_data.ipynb
- [77] Schmidt, U. (2021b, March 15). *StarDist 3D examples*. https://github.com/stardist/stardist/blob/master/examples/3D/1_data.ipynb
- [78] Schmidt, U., Weigert, M., Broaddus, C., & Myers, G. (2018). *Cell Detection with Star-convex Polygons*. https://doi.org/10.1007/978-3-030-00934-2_30

- [79] scikit-image. (2014). *Wavelet denoising*. https://scikit-image.org/docs/stable/auto_examples/filters/plot_denoise_wavelet.html
- [80] SHAFFER, J. (2016, April 20). 5 tips on designing colorblind-friendly visualizations. *Tableau*. <https://www.tableau.com/about/blog/examining-data-viz-rules-dont-use-red-green-together#:~:text=Use%20a%20colorblind%2Dfriendly%20palette%20when%20appropriate&text=For%20example%2C%20blue%2Forange%20is,blue%20to%20someone%20with%20CVD>
- [81] Shao, C., Chi, J., Zhang, H., Fan, Q., Zhao, Y., & Ye, F. (2020). Development of Cell Spheroids by Advanced Technologies. *Advanced Materials Technologies*, 2000183. <https://doi.org/10.1002/admt.202000183>
- [82] Shariff, A., Kangas, J., Coelho, L. P., Quinn, S., & Murphy, R. F. (2010). Automated Image Analysis for High-Content Screening and Analysis. *Journal of Biomolecular Screening*, 15(7), 726–734. <https://doi.org/10.1177/1087057110370894>
- [83] Sharma, G., Wu, W., & Dalal, E. N. (2005). The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1), 21–30. <https://doi.org/10.1002/col.20070>
- [84] Siddique, N., Paheding, S., Elkin, C. P., & Devabhaktuni, V. (2021). U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications. *IEEE Access*, 9, 82031–82057. <https://doi.org/10.1109/ACCESS.2021.3086020>
- [85] Sofroniew, N., & Lambert, T. (2019, November 28). *napari: A multi-dimensional image viewer for python*. napari.org
- [86] Somayeh, M. (2021). *Wavelet Theory*.
- [87] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. 30.
- [88] Stringer, C., Wang, T., Michaelos, M., & Pachitariu, M. (2020). *Cellpose: A generalist algorithm for cellular segmentation* [Preprint]. *Bioinformatics*. <https://doi.org/10.1101/2020.02.02.931238>
- [89] Targ, S., Almeida, D., & Lyman, K. (2016). *Resnet in Resnet: Generalizing Residual Architectures* (arXiv:1603.08029). *arXiv*. <http://arxiv.org/abs/1603.08029>
- [90] Thakur, A. (2021). Fundamentals of Neural Networks. *International Journal for Research in Applied Science and Engineering Technology*, 9(VIII), 407–426. <https://doi.org/10.22214/ijraset.2021.37362>
- [91] van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., & Yu, T. (2014). scikit-image: Image processing in Python. *PeerJ*, 2, e453. <https://doi.org/10.7717/peerj.453>

- [92] Van Valen, D. A., Kudo, T., Lane, K. M., Macklin, D. N., Quach, N. T., DeFelice, M. M., Maayan, I., Tanouchi, Y., Ashley, E. A., & Covert, M. W. (2016). Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. *PLOS Computational Biology*, 12(11), e1005177. <https://doi.org/10.1371/journal.pcbi.1005177>
- [93] Vuola, A. O., Akram, S. U., & Kannala, J. (2019). Mask-RCNN and U-net Ensembled for Nuclei Segmentation. *ArXiv:1901.10170 [Cs]*. <http://arxiv.org/abs/1901.10170>
- [94] Wang, H., Shang, S., Long, L., Hu, R., Wu, Y., Chen, N., Zhang, S., Cong, F., & Lin, S. (2018). Biological image analysis using deep learning-based methods: Literature review. *Digital Medicine*, 4(4), 157. https://doi.org/10.4103/digm.digm_16_18
- [95] Wang, X., & Lai, Y. (2021). Three basic types of fluorescence microscopy and recent improvement. *E3S Web of Conferences*, 290, 01031. <https://doi.org/10.1051/e3sconf/202129001031>
- [96] Weigert, M., Schmidt, U., Haase, R., Sugawara, K., & Myers, G. (2020). Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 3655–3662. <https://doi.org/10.1109/WACV45572.2020.9093435>
- [97] Wolfram Research, Inc. (2022). *Convex Polygon*. <https://mathworld.wolfram.com/ConvexPolygon.html>
- [98] Wolfram Research, Inc. (2022). *Star Polygon*. <https://mathworld.wolfram.com/StarPolygon.html>
- [99] Xia, X., & Kulis, B. (2017). *W-Net: A Deep Model for Fully Unsupervised Image Segmentation* (arXiv:1711.08506). arXiv. <http://arxiv.org/abs/1711.08506>
- [100] Yuste, R. (2005). Fluorescence microscopy today. *Nature Methods*, 2(12), 902–904. <https://doi.org/10.1038/nmeth1205-902>
- [101] Zhang, X., & Li, D. (2001). [Agrave] Trous Wavelet Decomposition Applied to Image Edge Detection. *Annals of GIS*, 7(2), 119–123. <https://doi.org/10.1080/10824000109480563>
- [102] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., & Liang, J. (2018). *UNet++: A Nested U-Net Architecture for Medical Image Segmentation* (arXiv:1807.10165). arXiv. <http://arxiv.org/abs/1807.10165>
- [103] Zindy, E. (2019, June 18). *(Edge aware) libatrous*. <https://github.com/zindy/libatrous>
- [104] Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B., & Chen, Y. (2015). Convolutional recurrent neural networks: Learning spatial dependencies for image representation. *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 18–26. <https://doi.org/10.1109/CVPRW.2015.7301268>

Appendix 1 : Software Used



Fiji: Fiji is a distribution of the open-source software ImageJ that focuses on biological-image analysis. Fiji enables rapid prototyping of image-processing algorithms. Fiji facilitates the transformation of new algorithms into ImageJ plugins that can be shared with end users through an integrated update system.



Napari: Napari is designed for browsing, annotating, and analyzing large multi-dimensional images. It is built on top of Qt (for the GUI), vispy (for performant GPU-based rendering), and the scientific Python stack (NumPy, scipy). It includes critical viewer features out-of-the-box, such as support for large multi-dimensional data and layering and annotation. Napari can be coupled to leading machine learning and image analysis tools (e.g. scikit-image, scikit-learn, TensorFlow, PyTorch), enabling more user-friendly automated analysis.



Jupyter notebook: Jupyter Notebook is an open-source web-based interactive application that use to create and contain live code, equations, visualizations, and text. Jupyter Notebook allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning.



Anaconda Navigator: Anaconda Navigator is a desktop graphical user interface (GUI) that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository.

Appendix 2 : Color difference formula

The color difference is the distance between two colors in a color-independent color device. The computation of the color difference allows the quantification and examination of the adjectives that are usually used to describe colors: hue, chroma, lightness, and brightness. Color coordinates, like RGB coordinates, can be located in a color-geometrical space like the Euclidian geometrical space. Because of this, it is understandable that if we have two color points, $c_1 = (R_1, G_1, B_1)$ and $c_2 = (R_2, G_2, B_2)$, and wish to find the distance between them, we would like to use the standard Euclidean distance formula:

$$\text{Euclidean distance} = \sqrt{(R_2 - R_1)^2 + (G_2 - G_1)^2 + (B_2 - B_1)^2}$$

The Euclidean distance computes the distance between two points, assuming the space containing them is linear, which is not the case for most color spaces. Using the standard Euclidean distance formula to compute the color difference between two colors in any of these spaces gives inaccurate results. To measure the color difference between two colors more accurately, scientists have worked on developing perceptually uniform color spaces like CIELAB and CIELUV and their corresponding color difference formulas. The CIELAB color space, also referred to as $L^*a^*b^*$, was defined in 1976 by the International Commission on Illumination (CIE). CIELAB defines colors using three color coordinates: L^* for perceptual lightness, and $\pm a^*$ and $\pm b^*$ for the four unique colors of human vision: red, green, and blue. The CIELAB color difference formula, CIEDE2000, is accepted as the more accurate color difference formula (Lindbloom, 2017; Sharma et al., 2005). The CIEDE2000 formula is given by:

$$\Delta E_{00} = \sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + RT \left(\frac{\Delta C'}{k_C S_C}\right) \left(\frac{\Delta H'}{k_H S_H}\right)}$$

Where:

$$\Delta L' = L_2^* - L_1^*$$

$$\Delta C' = C_2' - C_1'$$

$$\Delta H = 2\sqrt{C_1' C_2'} \sin\left(\frac{\Delta h'}{2}\right)$$

$$\bar{C}_i' = \sqrt{(a_i')^2 + b_i'^2}$$

$$h_i' = \begin{cases} 0, & b_i^* = a_i' = 0 \\ \tan^{-1}(b_i^*, a_i'), & \text{otherwise} \end{cases}$$

$$\Delta h' = \begin{cases} 0, & C_1' C_2' = 0 \\ h_1' - h_2', & |h_1' - h_2'| \leq 180^\circ; C_1' C_2' \neq 0 \\ (h_1' - h_2') - 360^\circ, & (h_1' - h_2') > 180^\circ; C_1' C_2' \neq 0 \\ (h_1' - h_2') + 360^\circ, & (h_1' - h_2') < -180^\circ; C_1' C_2' \neq 0 \end{cases}$$

The CIELAB color space, L^*_1, a^*_1, b^*_1 and L^*_2, a^*_2, b^*_2 , uses:

- A hue rotation term (RT)

$$R_t = -\sin(2\Delta\theta)R_c$$

Where:

$$R_c = 2 \sqrt{\frac{\bar{C}'^7}{\sqrt{\bar{C}'^7 + 25^7}}}$$

$$\bar{C}' = \frac{(C'_1 - C'_2)}{2}$$

- Compensation for lightness (SL)

$$S_L = 1 + \frac{0.015(\bar{L} - 50)^2}{\sqrt{20 + (\bar{L} - 50)^2}}$$

Where:

$$\bar{L}' = \frac{(L_1^* - L_2^*)}{2}$$

- Compensation for chroma (SC)

$$S_C = 1 + 0.045 \left(\frac{C'_1 + C'_2}{2} \right)$$

Where:

$$\bar{C}'_i = \sqrt{(a'_i)^2 + b'^*_i}$$

$$a'_i = (1 + G)a_i^*$$

$$G = 0.5 \left(1 - \sqrt{\frac{\bar{C}_{ab}^{*7}}{\bar{C}_{ab}^{*7} + 25^7}} \right)$$

- Compensation for hue (SH)

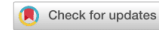
$$S_H = 1 + 0.015\bar{C}'^1 T$$

Where:

$$T = 1 - 0.17 \cos(\bar{h}' - 30^\circ) + 0.24 \cos(2\bar{h}') + 0.32 \cos(3\bar{h}' - 6^\circ) - 0.20 \cos(4\bar{h}' - 63^\circ)$$

$$\bar{h}' = \begin{cases} \frac{h'_1 + h'_2}{2}, & |h'_1 - h'_2| \leq 180^\circ; C'_1 C'_2 \neq 0 \\ \frac{h'_1 + h'_2 + 360^\circ}{2}, & |h'_1 - h'_2| > 180^\circ; (h'_1 + h'_2) < 360^\circ; C'_1 C'_2 \neq 0 \\ \frac{h'_1 + h'_2 - 360^\circ}{2}, & |h'_1 - h'_2| > 180^\circ; (h'_1 + h'_2) \geq 360^\circ; C'_1 C'_2 \neq 0 \\ h'_1 + h'_2, & C'_1 C'_2 = 0 \end{cases}$$

Appendix 3 : Publication



Automated high-speed 3D imaging of organoid cultures with multi-scale phenotypic quantification

Anne Beghin^{1,2,3}, Gianluca Greci^{1,4}, Geetika Sahni¹, Su Guo¹, Harini Rajendiran¹, Tom Delaire⁵, Saburnisha Binte Mohamad Raffi¹, Damien Blanc⁶, Richard de Mets¹, Hui Ting Ong¹, Xareni Galindo⁵, Anais Monet¹, Vidhyalakshmi Acharya¹, Victor Racine⁶, Florian Levet^{5,7}, Remi Galland⁵, Jean-Baptiste Sibarita⁵ and Virgile Viasnoff^{1,8,9}

Current imaging approaches limit the ability to perform multi-scale characterization of three-dimensional (3D) organotypic cultures (organoids) in large numbers. Here, we present an automated multi-scale 3D imaging platform synergizing high-density organoid cultures with rapid and live 3D single-objective light-sheet imaging. It is composed of disposable microfabricated organoid culture chips, termed JeWells, with embedded optical components and a laser beam-steering unit coupled to a commercial inverted microscope. It permits streamlining organoid culture and high-content 3D imaging on a single user-friendly instrument with minimal manipulations and a throughput of 300 organoids per hour. We demonstrate that the large number of 3D stacks that can be collected via our platform allows training deep learning-based algorithms to quantify morphogenetic organizations of organoids at multi-scales, ranging from the subcellular scale to the whole organoid level. We validated the versatility and robustness of our approach on intestine, hepatic, neuroectoderm organoids and oncospheres.

In organotypic three-dimensional (3D) cell cultures, referred to hereafter as organoids, stem cells differentiate and self-organize into spatial structures that share strong morphological and functional similarities with real organs. Organoids offer valuable models to study human biology and development outside the body^{1–3}. An increasing number of organoids are being developed to mimic intestines, liver, brain, kidney, lung and many other organs^{4,5}. Organoid differentiation is directed by exposure to a precise sequence of soluble growth factors and extracellular matrix during their development. However, in marked contrast to organs, the morphologies of organoids could be heterogeneous. In organs, phenotypic alterations can be used as a proxy to diagnose a diseased state while organoids differentiate in minimally reproducible micro-environments resulting in a large variability of their development path and morphology at the individual organoid level.

This has resulted in several technological roadblocks^{6,7} that demand new approaches such as fast, automated multi-scale imaging to quantify and control organoid diversity. In a recent study⁸, quantitative imaging of intestine organoid shape coupled to genetic marker assessment helped unravel the phenotypic development landscapes of intestinal organoids. The different organization paths that the organoids spontaneously adopted were quantified and causally linked to the biological programs orchestrating the spatial induction of different cell fates. However, such studies^{8,9} required innovative methodological and technological approaches (<https://bioengineeringcommunity.nature.com/posts/intestinal-regeneration-lessons-from-organoids>) that involved the use a multi-well plate for organoid growth, and a commercially available high-content imaging techniques performed on fixed

organoids at specific time point during their growth. Arguably, the ability to relate the diversity of genomic expression in organoids to their phenotypic behavior is a major step to guide the establishment of optimal culture conditions¹⁰ as well as to provide a system-biology view of developmental pathways. Therefore, such efforts demand the development of dedicated high-content 3D imaging methods to characterize organoids on subcellular, multicellular and whole organoid scales, both in 3D (ref. ¹⁰) and during their growth^{11,12}.

The next generation of high-content screening platforms for organoids would certainly benefit from the unsurpassed performance of light-sheet microscopy¹³ compared to confocal or multiphoton laser scanning, as implemented in the current commercially available instruments (for example, PerkinElmer's Opera or Molecular Devices' ImageXpress). Indeed, light-sheet fluorescence microscopy (LSFM), also called selective plane illumination microscopy (SPIM), has proved to be the most appropriate imaging technique to visualize fine cellular details at the whole tissue level, with maximal speed and minimal phototoxicity and photobleaching^{14,15}. LSFM has thus been favored for the study of live organoids, as it allows long-term monitoring in physiological conditions¹⁶.

However, since LSFM traditionally involves several objectives and has cumbersome sample mounting requirements, it results in low imaging throughput despite its superior imaging performances. Recent studies using dual objective SPIM have tried to engineer solutions to improve the throughput¹⁷ using a liquid handling robot and complex dedicated imaging hardware. However, these solutions do not permit long-term sterile culture, as a periodic medium or extracellular matrix exchange is mandatory for organoid development and differentiation. To overcome these important limits,

¹Mechanobiology Institute, National University of Singapore, Singapore, Singapore. ²Immunology Translational Research Programme, Yong Loo Lin School of Medicine, National University of Singapore, Singapore, Singapore. ³Department of Microbiology and Immunology, National University of Singapore, Singapore, Singapore. ⁴Biomedical Engineering Department, National University of Singapore, Singapore, Singapore. ⁵University of Bordeaux, CNRS, Interdisciplinary Institute for Neuroscience, IINS, UMR 5297, Bordeaux, France. ⁶QuantaCell, Pessac, France. ⁷University Bordeaux, CNRS, INSERM, Bordeaux Imaging Center, BIC, UAR 3420, US 4, Bordeaux, France. ⁸Department of Biological Sciences, National University of Singapore, Singapore, Singapore. ⁹IRL 3639 CNRS, Singapore, Singapore. [✉]e-mail: mbianne@nus.edu.sg; jean-baptiste.sibarita@u-bordeaux.fr; dbsvnr@nus.edu.sg

several approaches have been developed to perform LSM with a single objective. High speed, high-content SCAPE¹⁸ was used to obtain ultra-fast images of *in vivo* processes in individual *Caenorhabditis elegans* and *Danio rerio*. Alleviating the multi-objective constraint, other approaches have enabled imaging multiple samples. Epi-illumination SPIM (ssOPM) enabled parallel 3D volume acquisition of a single cell at the single molecule level in 96-well plates¹⁹. It also allowed the imaging of living oncospheres in 3D seeded in agarose to detect glucose uptake²⁰, but only during a short period of time (3 hours). Moreover, the random positioning of the spheroids in the agarose limited the study to 42 oncospheres.

It follows that existing technological solutions still have room for improvement to allow 3D high-content imaging of live and fixed organoids.

In this context, we developed an imaging platform capable of streamlining organoid culture with fast light-sheet 3D imaging. It integrates new miniaturized 3D cell culture devices, called JeWell chips, which are compatible with organoids up to 300 μm in diameter. They comprise thousands of well-arrayed microwells with pyramidal shapes flanked with 45° mirrors (JeWells) to enable single-objective SPIM (soSPIM)²¹. Compatible with standard commercial inverted microscopes after simple addition of a dedicated beam-steering unit and control software, our system can perform 3D imaging of up to 300 organoids in 1 hour with subcellular resolution. We tested our platform on various organoids and tumoroids (hepatocytes, neuroectoderm, intestinal, HCT116), and performed high-content quantitative analysis by training public-domain convolutional neural networks (CNN) to extract and correlate relevant biological features at different scales (subcellular, cellular, multicellular and whole organoid). We demonstrate how our technique enabled correlative imaging between the live development and the fixed immunolabeled endpoint imaging of organoids.

Results

Working principles. JeWells are disposable microfabricated chips used to create high-density arrays of organotypic cell aggregates for automated high-content 3D soSPIM imaging (Fig. 1a, Extended Data Fig. 1a and Methods). They are truncated pyramidal microcavities molded in a photocurable resin (NOA73) with 45° gold-coated faces. The JeWell microcavities are arrayed at a typical surface density of 10 JeWells per mm^2 , depending on individual JeWell dimensions. It corresponds to the equivalent of 96 organoids in a single well of a standard 384-well plate. This ensures an optimal scanning distance between JeWells for high speed 3D soSPIM imaging of multiple organoids.

We devised JeWell chips in format of single Petri dishes, six-well format or 96-well plates (Extended Data Fig. 1b) with JeWells of various sizes and shapes (Extended Data Fig. 2a–c). Coating with Lipidure provided an enduring antifouling treatment to prevent any cell adhesion over months allowing long-term cultures (detailed protocol in Methods). Single cells or small-cell aggregates seeded homogeneously over the chips sedimented to the bottom of the JeWells (Fig. 1b). The partial reflectivity (roughly 70%) of the JeWell mirrors allows phase-contrast imaging of the cell aggregates. The initial cell concentration and the seeding time allow the control of the number of cells per JeWell (for example, 0.5×10^6 cells per ml leads to 221 ± 67 s.d. cells per JeWell ($n=775$ JeWells)).

We then devised an automated cell counting scheme (Extended Data Fig. 3a,b and Methods) to create a heat map of the initial number of individual stem cells in each JeWell over the entire chip ($n=775$ JeWells) (Extended Data Fig. 3c). The deliberate inhomogeneous seeding of cells allowed to simultaneously test several initial cell densities. The histogram of initial number of cells per well illustrates that, for any average cell number between 150 and 300 cells per JeWell, the chip contains at least 96 JeWells with a maximum spread of ± 15 cells. If needed, a more homogeneous

distribution per JeWell could be achieved with stirring during the seeding phase.

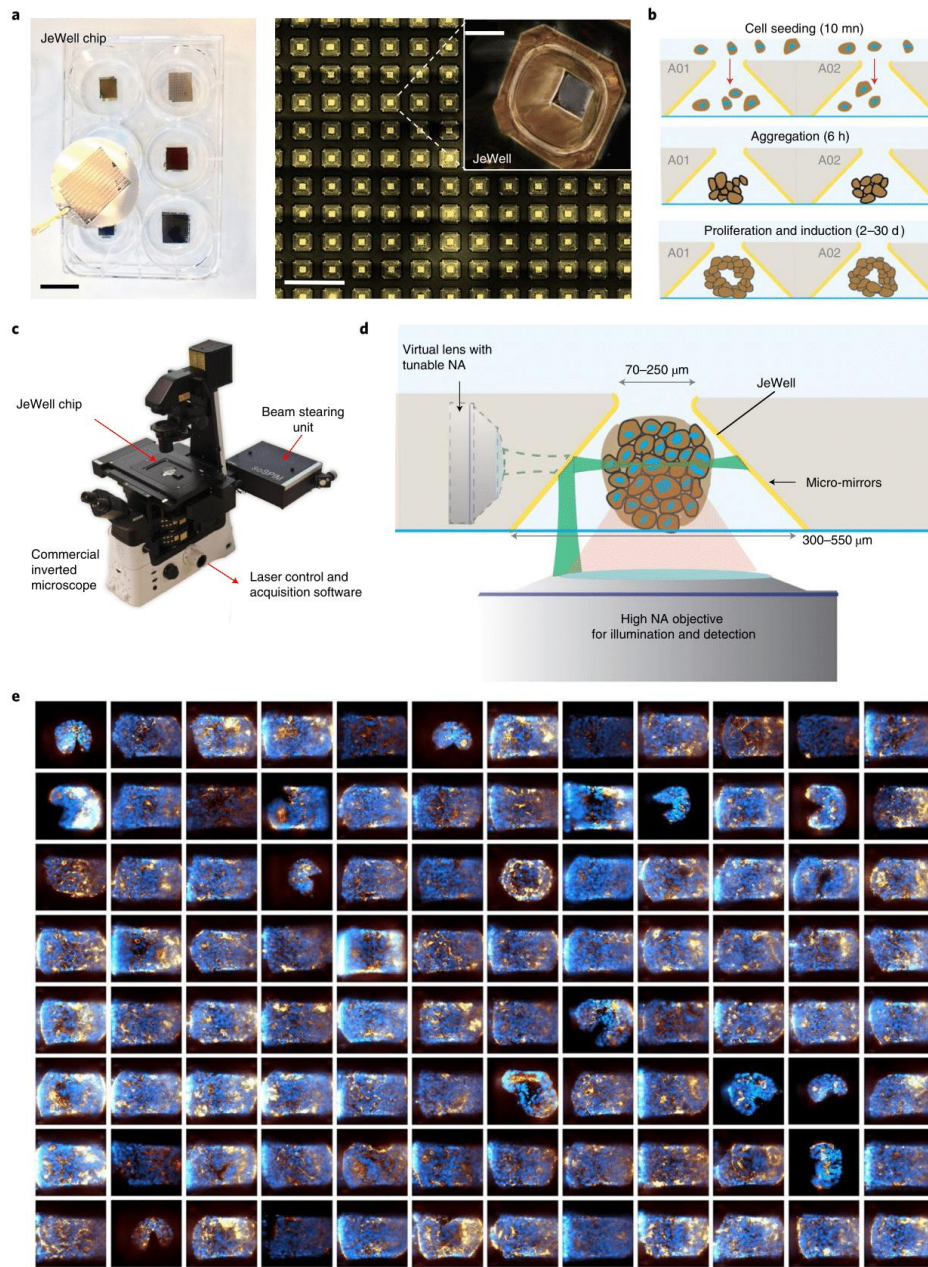
We then monitored the projected area of the cell aggregates formed during the first few hours postseeding (Extended Data Fig. 3d and Methods). The variability of the size of the aggregates after 6 hours was less than 10% between our three tested batches (Supplementary Fig. 3d). After 10 days, we obtained a 25% dispersion of aggregate sizes within each batch ($n=100$ organoids) (Extended Data Fig. 3e). This procedure proved to be robust for all cell types we tested. In the rest of the paper, cells are subsequently differentiated and grown in the JeWell cavities, except when specified otherwise.

Light-sheet imaging was performed on a commercial inverted microscope (Nikon Eclipse) equipped for soSPIM as described in Galland et al.²¹ (Methods and Extended Data Fig. 4a). It is composed of: (1) interchangeable lenses ($\times 20$ air (numerical aperture (NA)=0.75), $\times 40$ WI (NA=1.15) and $\times 60$ WI (NA=1.27) used here but not limited to), (2) a custom laser scanning unit and (3) an acquisition module operating inside MetaMorph commercial software (Fig. 1c). In brief, the 45° gold-coated micromirrors of the JeWells reflect a laser beam (Fig. 1d and Extended Data Fig. 4b) focused on the sample by the same objective used for imaging. Scanning the laser beam along the mirror axis (y direction) creates a light-sheet perpendicular to the optical axis of the objective. Synchronizing the light-sheet position with the focal plane of the objective ensures optimal 3D optical sectioning. The integration of the optical components inside the chips eliminates the steric and alignment constraints of traditional multi-objective light-sheet microscopes. A tunable diaphragm allows adjustment of the laser beam diameter of the light-sheet thickness between 2 and 3.7 μm , and its length between 12 and 60 μm , respectively, using a $\times 60$ magnification. A light sheet 5.7 μm thick and 135 μm long was obtained with a $\times 20$ magnification objective (Extended Data Fig. 4c).

For high-content imaging, we devised an automatic acquisition pipeline that integrates (1) the automated identification and user-assisted selection of the JeWells containing organoids to image, and (2) the precise automatic positioning of the JeWell mirrors for reproducible light-sheet positioning (Methods and Extended Data Fig. 5a,b). It allows rapid automatic setup of the sequential 3D multi-color imaging of hundreds of organoids in a single acquisition workflow, with minimal user interaction (Extended Data Fig. 5c). We used the JeWells as proxies to automatically align and calibrate the light sheet and to correct for drifts. Performing the calibration steps and setting the acquisition parameters (including the localization of 96 organoids) took typically 10 min. Using a $\times 60$ 1.27 NA objective, we could acquire images with a lateral resolution of around 300 nm (Supplementary Fig. 5d).

This new integrated high-content imaging platform provides an efficient way to perform long-term automatic 3D imaging of various organoids with high throughput and limited manipulation. Figure 1e illustrates a collection of 96 neuroectoderm organoids differentiated from hESCs, fixed at day 8 and immunostained for nuclei (405 nm), Sox2 (488 nm) and N-cadherin (647 nm) inside a JeWell chip (Methods). Each 3D stack acquisition took about 7 s per wavelength for 70 optical sections using one side light-sheet illumination. The time between two adjacent JeWell acquisitions was 1.5 s, and the light-sheet repositioning accuracy below 50 nm. With these settings, we could acquire 96 organoids with three wavelengths in 1 h. It represented 173 GB of raw data using a 2,048 \times 2,048 sCMOS camera.

Testing different cell types. We then tested the versatility of the approach by culturing and imaging a variety of organotypic systems in JeWells. We adjusted the dimensions of the JeWells to the final size and shape of the targeted organoids (Fig. 2a).



96 organoids, 70 z planes each, 3 wavelengths, 173 GB of (2,048 × 2,048 × 70) stacks, 1 h acquisition time

Fig. 1 | Working principles. **a**, JeWell chips in a six-well dish (left) and close-up image of the JeWells array with a density of 16 JeWells per mm^2 (right). Inset shows a zoom on a JeWell inverted pyramidal microcavities flanked with four 45° mirroring surfaces. Scale bars, 3 cm (left), $500\ \mu\text{m}$ (right) and $70\ \mu\text{m}$ (inset). **b**, Schematic of the seeding procedure. **c**, Photograph of the imaging setup comprising a commercial inverted microscope, combined with JeWell chips, a laser scanning unit and its custom-made control software. **d**, Principles of the soSPIM. **e**, Representative gallery of 96 neuroectoderm organoids from a library of >400 organoids (median plane of the 3D stacks) labeled with actin (gold), Sox2 (magenta) and DAPI (blue) acquired in an automated workflow in less than 1 hour.

NATURE METHODS | www.nature.com/naturemethods

Figure 2 shows the compatibility of the platform for culturing primary rat hepatocytes, human embryonic stem cells (hESCs) and cancer cell lines. The protocol for each cell type involved different culture times from 2 to 20 days, different extracellular matrix concentrations (Matrigel) and culture medium with specified mix of soluble growth factors to induce differentiation or to maintain cell cultures (Methods). For example, the seeding protocol involved single cells dissociation for primary cells, hESCs and cell lines (Fig. 2b–d) but for intestine organoids (Fig. 2f), the hESCs were first differentiated to mid/hindgut and further dissociated into larger cell clumps (50–75 μm in diameter) before seeding.

This resulted in development of expected morphological traits for each organoid type: bile canaliculi formation for hepatocytes revealed by MRP2 (ref. 22) (bile transporter) staining (Fig. 2b and Supplementary Video 1), rosettes for neuroectoderm differentiation revealed by actin and N-cadherin staining²³ (Fig. 2c), hypoxic cores for HCT116 colon tumorspheres (Fig. 2d), cysts for hESCs (Fig. 2e) and polarized elongated lumens in intestinal organoids (Fig. 2f) revealed by the gradual localization of vili²⁴ on the membrane of cells facing the lumen. In addition, the intestinal organoid culture shows that the elongated JeWells can provide a scaffold to grow long nonspherical organoids and image them at different magnifications ($\times 60$, $\times 20$).

In all cases, the soSPIM image quality met the expectations of state-of-the-art single side Gaussian light-sheet imaging. We observed an enhance penetration of the imaging depth compared to a spinning disc (Extended Data Fig. 6a), a higher photon capture efficiency (Extended Data Fig. 6b) and low photobleaching (Extended Data Fig. 6c). A gradient of optical sectioning was also observed along the direction of propagation of the light sheet as expected for single side illumination (Extended Data Fig. 6a). Taken together, these data demonstrate the adaptability of the JeWells approach to a variety of standardized protocols and its suitability for an extended panel of cell types known for their strong sensitivity to the culture environment.

3D multi-scale analysis of organoid morphologies. We then tested whether the single side illumination we implemented could prove a good compromise between acquisition speed and phototoxicity. It showed that it provided sufficient image quality to implement automated multi-scale quantitative analyses using the various CNNs summarized in Extended Data Fig. 7. To demonstrate the feasibility of a multi-scale analysis using our platform, we performed several quantifications from different organoid models.

We first acquired and analyzed HCT116 aggregates (human colon cells) grown in the JeWell chips, which we fixed and stained for nuclei (4,6-diamidino-2-phenylindole, DAPI) and proliferation status (Ki67, Alexa647). We trained the StarDist CNN²⁵ using 471 DAPI images containing 5,274 nuclei manually annotated in 3D. We compared the predictions with the ground truths in 3D on ten organoids, assessing a detection accuracy superior to 85% (Supplementary Video 2). Figure 3a,b illustrates the quality of the

3D segmentation where the nuclei contours are properly detected. We then trained the YOLOv5 CNN on the Ki67 channel to detect and classify dividing cells²⁶ into three proliferation phases: G1/S/G2 (large dense clusters of Ki67), mitosis (condensed chromatid marked by Ki67) and early G1 (nuclei with scattered Ki67 aggregates), with an accuracy of 90% (Fig. 3c,d and Supplementary Video 3). G0 cells were Ki67 negative. Combining both analysis schemes, we could assign a given proliferation state to each detected nucleus (Fig. 3e). We checked that the number of nuclei scaled linearly with the volume of the organoids (Fig. 3f). However, the fraction of Ki67 positive nuclei decreased with the total number of nuclei per organoids (Fig. 3g), indicative of a reduced proliferation rate with an increase in organoid size. Further analysis showed that cells were arrested more specifically at the G0 phase as the organoids grew. Indeed, among the proliferative cells, the proportion of cells in the early G1 and mitosis phases was insensitive to organoid volume (Fig. 3h), indicating that the division cycle time was independent of the organoid size with no arrest at any proliferation check point.

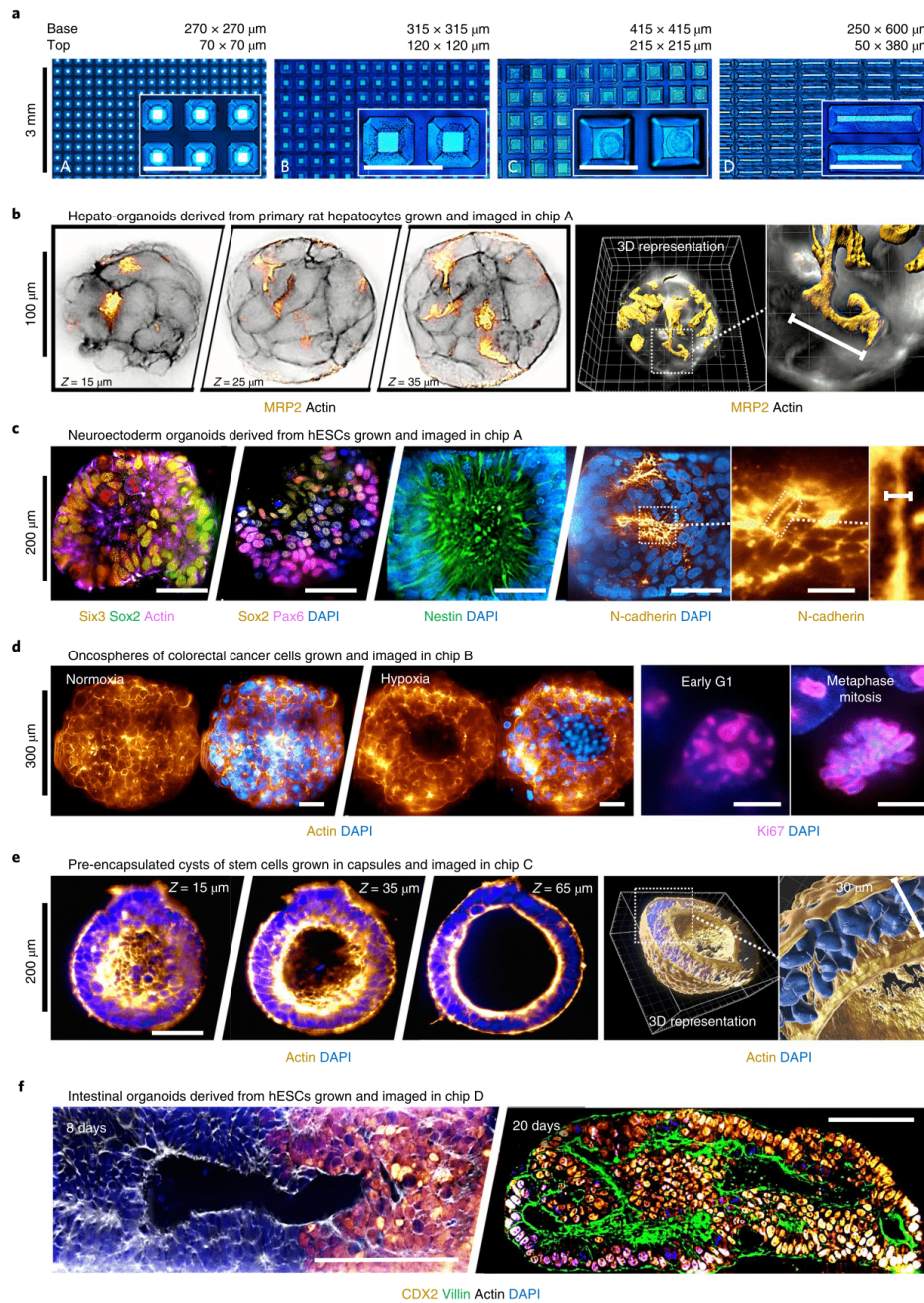
We then applied the automated classification of the proliferation events to another type of organoid (neuroectoderm organoids at day 8). We reached a classification accuracy of 89% (4% false positive, 7% false negative, benchmarked on 20 test organoids by a trained human) with YOLOv5 trained on 28 annotated organoids using the DAPI channel only. Figure 4a shows the 3D organization of the mitotic and cell death events. They appeared homogeneously distributed within the organoid volume, as expected for healthy proliferating organoids. This spatial homogeneity was confirmed by the equal average density of mitotic and cell death computed in 3D regions of interest located in the central versus peripheral regions of the organoids (Fig. 4b and Methods). The batch-to-batch variability of the cell number, mitotic index and cell death is detailed in Extended Data Fig. 8a.

Beyond nuclei segmentation and proliferation analysis, we characterized the spatial distribution of cells expressing differentiation markers. Neuroectoderm organoids organized locally in multicellular rosettes (Fig. 4c) characterized by layers of aligned cells surrounding a central streak that develops into a lumen^{25,27–30}. We used the 3D StarDist CNN to segment individual nuclei from the DAPI channel and computed the average expression of Sox2 in each segmented nucleus. The distribution of expression levels proved to be bimodal (Fig. 4d). We thus binarized the distribution between Sox2 positive and Sox2 negative cells by setting a threshold value at the bimodal intersection point (680 a.u.). We hence built a 3D map of the Sox2 expressing cells in each organoid (Fig. 4e), clearly revealing the rosette organization with Sox2 positive cells surrounding the central streak. We checked the homogeneity of detection of the nuclei across the organoid volume (Extended Data Fig. 8b). We also compared the quality nuclei segmentation for the single-sided versus two-sided illumination by sequentially illuminating the opposing mirrors on 30 organoids. Extended Data Fig. 8c,d shows the consistency of the nuclei detection: we observed 98% similarity

Fig. 2 | Matching JeWell chips to different types of organoid. **a**, Examples of JeWell cavities dimensioned to fit the size of the organoids (scale bars, 500 μm). **b**, Representative examples of 3D stacks ($\times 60$, WI) and 3D reconstruction of primary rat hepatocyte aggregates stained for actin (gray) and MRP2 (canaliculi marker) (gold) after 4 days of culture ($n=96$). Scale bar, 20 μm . **c**, Representative median plane images ($\times 60$, WI) of hESCs derived neuroectoderm organoids (8 days old) immunolabeled for various transcription factors (as indicated) and for N-cadherin enriched in the rosette core ($n=400$). Scale bar 100 μm (left) and 20 μm and 500 μm (right). **d**, Typical images ($\times 20$) of oncospheres (HCT116) (roughly 200 μm in diameter) grown in normoxic and hypoxic conditions. Images of nuclei in various proliferation stages (G0, G1, S, mitosis) labeled with Ki67 staining ($n=400$). Scale bars are 100 μm unless otherwise noted. **e**, Representative 3D stack ($\times 20$) and 3D reconstructions of hESCs cells pre-encapsulated in alginate capsules. The encapsulated cells were grown outside the JeWells and the capsules were seeded in pyramids with large top openings (215 μm). Scale bars, 100 μm (left) and 30 μm (right). **f**, Representative median plane ($\times 20$) of intestinal organoids derived from hESCs at day 8 (left) and day 20 (right) grown in rectangle grooves and stained (CDX2, vili, actin and nuclei) at an early stage (D08) and a later stage (D20) ($n=96$). All images in the figure are raw images from soSPIM setup without any postprocessing. Scale bars, 100 μm .

between both illumination schemes. Additionally, the overlap of the segmented volumes of the nuclei reached 93% overlap, demonstrating the accuracy of StarDist properly trained on images acquired with a single side illumination.

We then endeavored to segment the rosettes based on actin organization: actin is a commonly used marker in live cell imaging. To this end we trained a 3D U-Net CNN³¹ (Methods) to segment the rosettes in 3D (Fig. 4f and Supplementary Videos 4 and 5), from



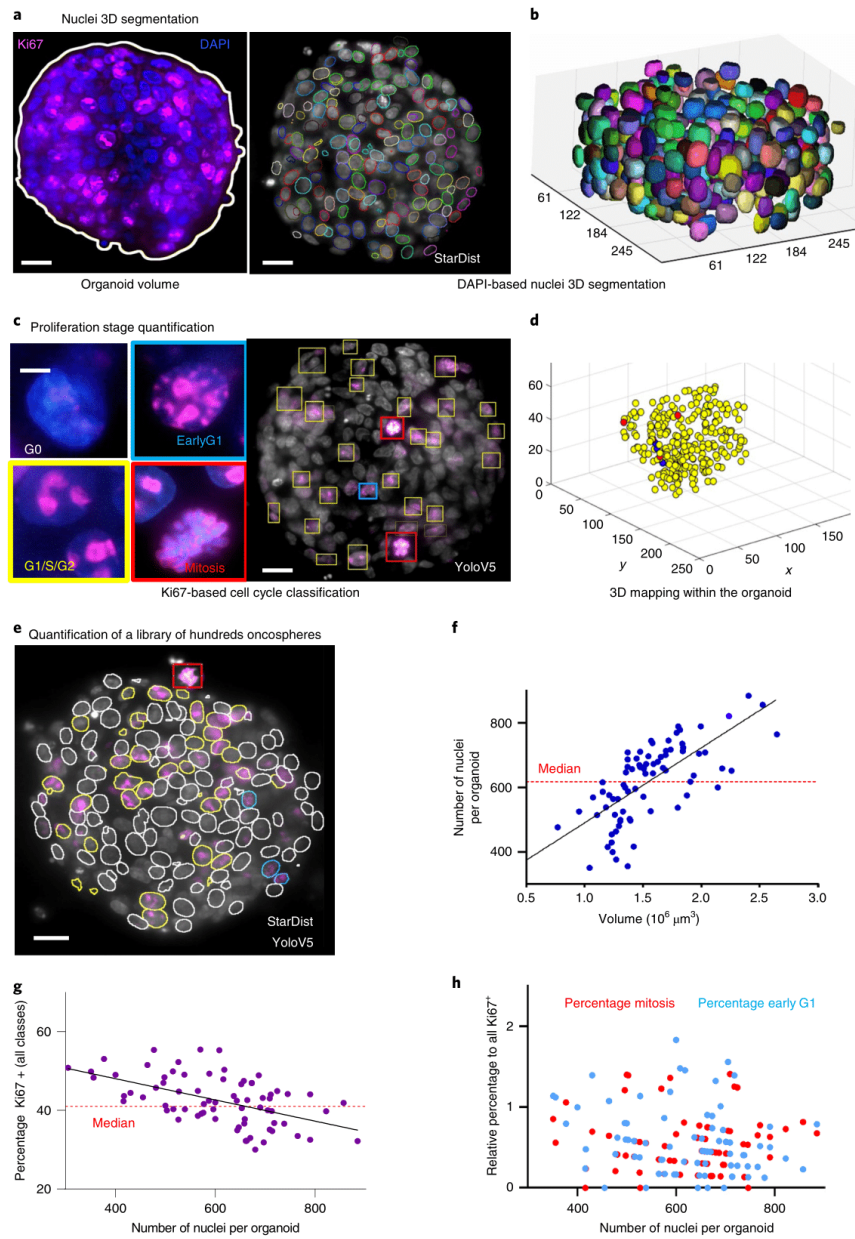


Fig. 3 | Subcellular quantification of cell proliferation in oncospheres. **a**, Image of the median plane of an oncosphere (HCT116) stained for nuclei (DAPI) and ki67 in normoxia conditions. The volume of the oncosphere is measured by thresholding the aggregate contour in 3D. Scale bars, 20 μm . **b**, Individual nucleus positions are obtained by segmenting the DAPI channel using the 3D StarDist neural network (middle panel). 3D reconstruction of segmented nuclei by surface rendering. **c**, Artificial intelligence detection of three different stages of cell division based on the Ki67 signal using YoloV5 (left panels). Scale bars 5 μm (left) and 20 μm (right). **d**, The 3D localization of individual proliferating nucleus is represented in 3D by a sphere. The colors code for the proliferation stages (blue for early G1, yellow for G1/S/G2, red for mitosis, middle and right panels). **e**, Correlation between the predictions of both artificial intelligence networks. The color code of the nuclei contours corresponds to the class of proliferation stages. Scale bar, 20 μm . **f**, The total number of nuclei linearly scaled with the approximated volume of the oncosphere. **g**, The fraction of cells in each proliferative state decreased proportionally to the total number of cells. **h**, Proportion of each division phase relative to the total number of proliferative cells. Each point represents one organoid.

which we could quantify the number of rosettes per organoids as well as their morphological features for further classification (Supplementary Table 1 and Extended Data Fig. 9) ($n = 415$ streaks longer than two cells ($20\ \mu\text{m}$)).

Finally, we assessed the neuroectoderm variability of the organoid whole shapes using the DenseNet121 (ref. ³²) shape recognition CNN (Methods) to classify automatically the organoids in two morpho-classes: bean shaped (B-shaped) or spherical (O-shaped) (Fig. 4g) with 99% accuracy. We observed that B-shaped organoids systematically displayed fewer nuclei and internal rosettes than O-shaped ones.

Altogether, we demonstrated the capability of our platform to perform multi-scale analysis on the same organoid, from the single cell level, up to the whole organoid level.

Rare events detection by multi-magnification imaging. We leveraged the architecture of the platform to sequentially image the organoids at different magnifications. This facilitates the localization of rare cellular clusters composed of a few cells expressing specific markers. We used early-stage intestinal organoids that were grown and differentiated for 8 days (Fig. 5). We fixed and stained them for nuclei (DAPI), actin, CDX2 (a major regulator of intestine-specific genes) and E-cadherin (E-cad). As expected²⁴, at these early stages of differentiation, only a fraction of the cells were CDX2 positive. We used JeWells array to perform correlative multi-magnification 3D imaging.

First, we performed organoid localization in brightfield at $\times 4$ magnification (Fig. 5a), and then rapidly scanned 100 organoids in 3D (10 min for 100 organoids) at $\times 20$ magnification (air objective, ten Z planes, $7\ \mu\text{m}$ Z steps) for two channels (CDX2 and actin) (Fig. 5b). The CDX2 channel was then analyzed in ImageJ using the simple thresholding scheme described in Methods. We identified the CDX2 positive organoids (CDX2⁺) using a basic segmentation routine detailed in the Methods. CDX2⁺ organoids represented 20% of the whole library. We next performed a second scanning only of the CDX2⁺ organoids for the two other channels (E-cad and DAPI) to screen the E-cad expression profile at this early stage, still at low magnification ($\times 20$) but with higher axial sampling (Fig. 5c, 50 planes, $1\text{-}\mu\text{m}$ Z steps, 10 min for 20 organoids). An automated segmentation (Methods) of the E-cad channel (Fig. 5c) allowed identification of E-cad expressing cell clusters (E-cad⁺) in 25% of the CDX2⁺ organoids (5% of the whole library of 100 organoids). Finally, a third round of 3D acquisitions at high resolution using a $\times 60$ WI 1.27 NA magnification objective was performed to exclusively image the E-cad⁺ regions (four channels, 50 Z planes, $1\ \mu\text{m}$ Z steps, Fig. 5d, upper panel). The high-resolution 3D images enabled us to reconstruct the E-cad⁺ junctions of CDX2 positive cells in 3D, which was not possible at $\times 20$ magnification (Fig. 5d, lower panel).

Figure 5e (lower panel) displays one representative example of 3D reconstruction of E-cad⁺ cellular cluster, where one central cell was surrounded and linked to four cells (only five E-cad⁺ cells among thousands constituting one single organoid). These five cells, localized at the border of the lumen, also showed a very high

level of CDX2 compared to the adjacent cells, confirming the link between CDX2 and E-cad expression profiles (Fig. 5e, upper panel). Altogether, this acquisition and the analysis took less than 1 h. In conclusion, finding rare cellular clusters within the organoids population was greatly facilitated by the capability of multi-magnification imaging of the soSPIM imaging system.

Live imaging modalities. We finally tested how our approach can perform correlative imaging between the morphology of live developing organoids and the endpoint analysis of their transcription profiles. We followed the live 3D development of neuroectoderm organoids. We first stained the intercellular clefts in the organoids (Fig. 6a) with extracellular dyes (Calcein, Alexa Fluor 647). Extended Data Fig. 10a demonstrates that the dyes infiltrated the entire organoid intercellular volume within 15 minutes. It demonstrated that small soluble compound or electrolytes can swiftly diffuse in the organoid. It also allowed us to monitor the development of rosettes and streaks in 3D (Supplementary Video 6). To establish the maturation stage of the cells around the rosettes, we fixed the organoids at day 7 and stained them for Sox2 (early development marker) and Pax6 (later development marker)²⁷. The high-quality automatic repositioning of individual JeWells enabled us to overlap the last images of the live sequence with the immunostained images of the same organoid after fixation (Fig. 6b) by a simple registration in Z. It revealed the presence of Pax6 in some rosettes and Sox2 in others. We then used the 3D time lapse to back track the rosette formation process (Fig. 6a), which revealed that the Sox2⁺/Pax6⁻ rosettes (white arrow) appeared 3 days before the Sox2⁺/Pax6⁺ rosettes. However, immunostaining only provided access to the spatial distribution of a limited number of transcription factors or proteins of interest.

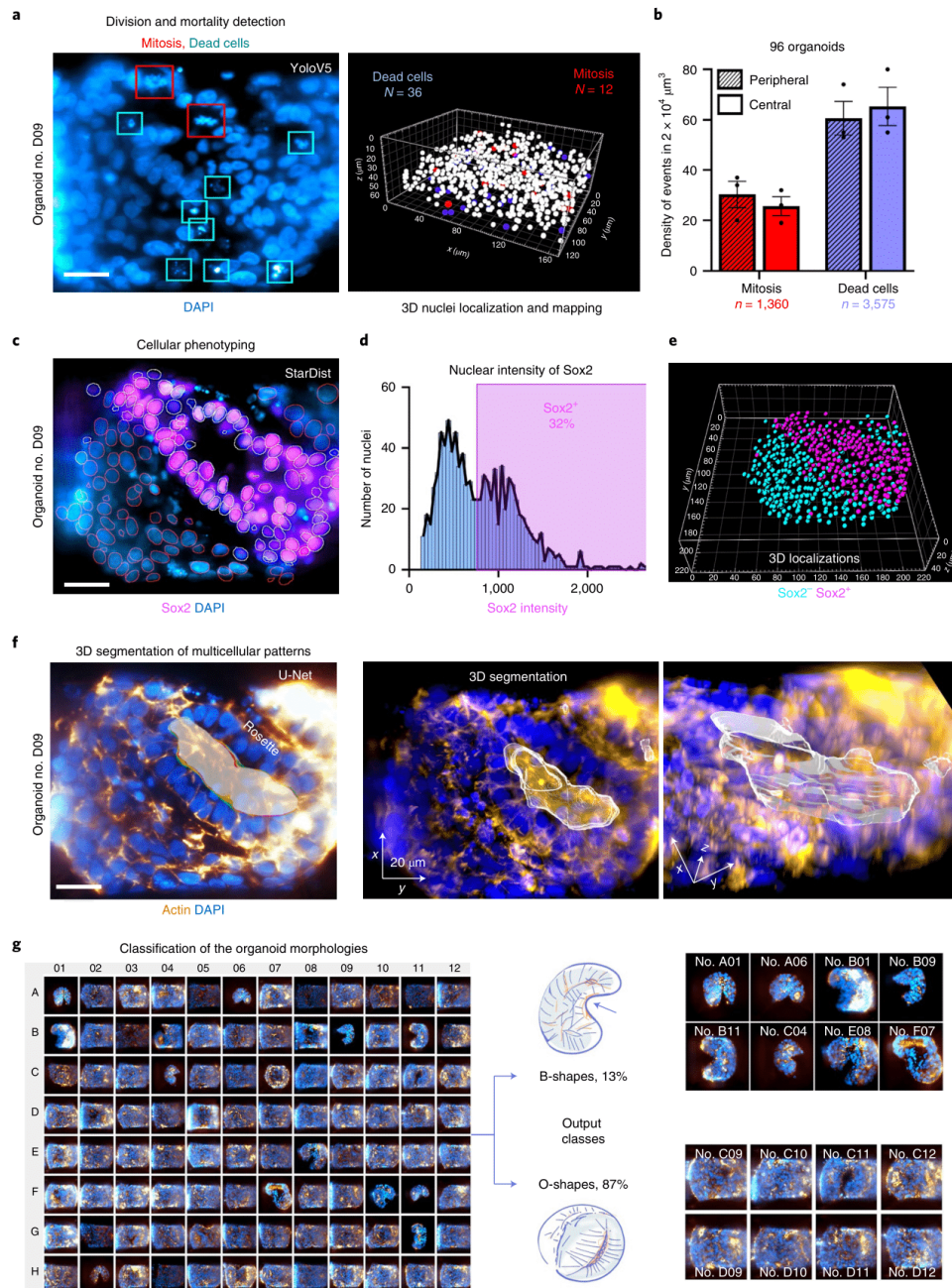
To identify the expression level of a higher number of expression factors and proteins of interest, we performed quantitative reverse transcription PCR (RT-qPCR) on libraries of organoids after imaging. We first tracked the development of neuroectoderm organoids using stem cells expressing lifeact green fluorescent protein (GFP) and Histone 2B-mCherry³³ (Supplementary Video 7). We monitored 25 organoids, capturing one z stack of 70 optical planes every 15 minutes for 8 days, representing a total of 500 image stacks per organoid acquired in 125 h (Methods). Backtracking the rosettes from day 8, we could follow the time course of the first development steps based on local cell arrangements (Fig. 6c and Supplementary Video 8). It revealed that rosette-like structures started to appear in the organoids after 4–5 days, consistent with the data from Fig. 6a. During these 8 days, we did not notice any effect of photodamage on cell mitosis and proliferation.

We then collected the live organoids at day 8 by peeling off the JeWell membrane from the glass slide with a sterilized tweezer (Fig. 6d), and subsequently performed RT-qPCR on 11 specific markers (Fig. 6e). Results unambiguously demonstrated the differentiation into neuroectoderm lineage after 8 days of culture in the JeWells relative to day 2 (undifferentiated stem cell aggregates), characterized by a tenfold reduction in E-cad and NANOG and 500-fold increase in Pax6, for example. This demonstrates that

Fig. 4 | Multi-scale analysis of neuroectoderm organoids. **a**, Detection of mitosis and cell death based on DAPI signal using YoloV5 CNN (left panel). 3D localization of mitosis and cell death events within one full organoid. White, all nuclei ($n = 680$) (detected by StarDist); blue, cell death ($n = 36$) and red, mitosis ($n = 12$). Scale bar, $20\ \mu\text{m}$. **b**, Averaged density of mitosis (red) and dead cells (blue) in subvolumes (dots) ($2 \times 10^4\ \mu\text{m}^3$) sampled in periphery (hatched) or central (plain) regions ($n = 96$ organoids, 1,360 mitosis, 3,575 dead cells). The chart bars the average over the three zones \pm s.d. **c**, 3D segmentation of the nuclei in neuroectoderm organoid using 3D StarDist CNN on DAPI signal, left panel. Sox2⁻ cells (red contours), Sox2⁺ cells (white contours). Scale bar, $20\ \mu\text{m}$. **d**, Bimodal distribution of the average intensity levels of Sox2 signals within each nucleus. **e**, 3D representation of the Sox2⁻ (cyan) and Sox2⁺ (purple) nuclei based on the cut-off derived from the histogram. **f**, Different views of the 3D U-Net CNN based segmentation in 3D of a streak regions for the rosettes (actin staining). Scale bar, $20\ \mu\text{m}$. **g**, DenseNet121 CNN classification of the neuroectoderm organoid morphologies in two classes: B-shape or O-shape ($n = 96$).

organoids cultured in JeWell chips may be collected for further omics analysis or also be reseeded in other devices or even injected in vivo as the sterility and viability has been conserved.

Our comparative test for neuroectoderm cultured in JeWells or in U-Bottom dishes show only a difference on neuroectoderm markers for a few genes by RT-qPCR (Extended Data Fig. 10b).



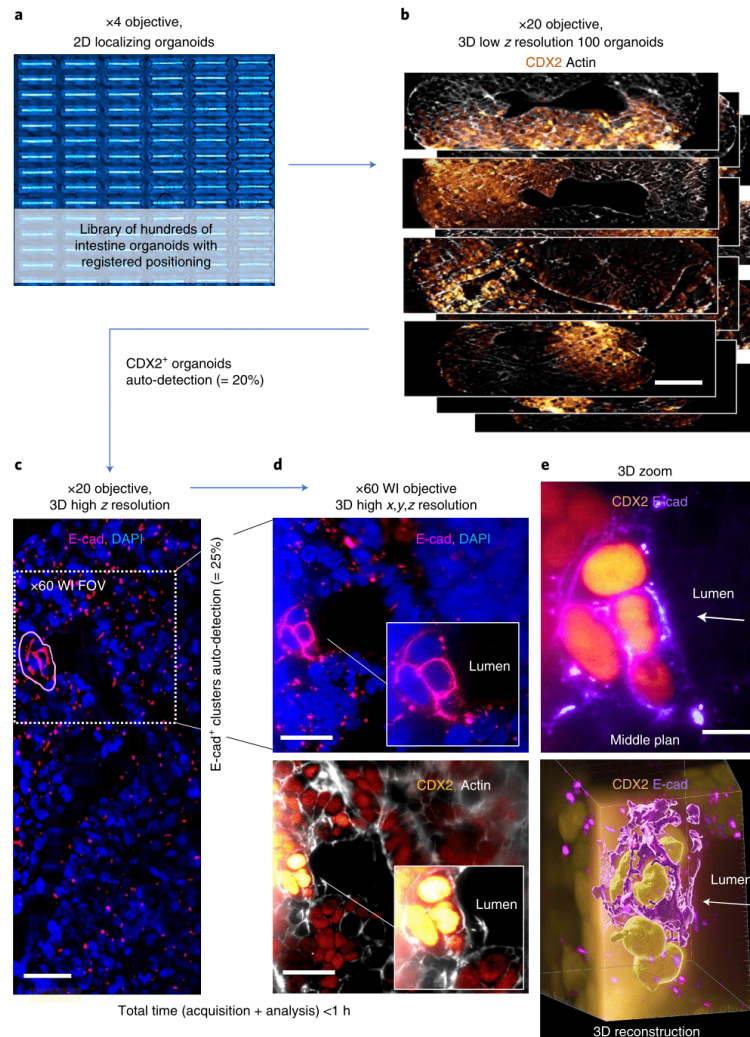


Fig. 5 | Efficient rare events detection by multi-magnification imaging pipeline. **a**, A representative brightfield image using a $\times 4$ objective of a rectangular JeWell chip with intestinal organoids (day 8). **b**, A collection of 100 intestine organoids with registered positions were imaged in less than 10 min using a $\times 20$ objective (air) in soSPIM (seven planes per organoid, step size of $10\ \mu\text{m}$, two channels: CDX2 and actin). We automatically selected the CDX2 positive organoids (roughly 20% of the whole batch, $n=100$). Scale bar, $70\ \mu\text{m}$. **c**, Rare clusters of E-cad⁺ cells (pink contour) were automatically detected by a second round of imaging of the selected CDX2⁺ organoids ($\times 20$ with a higher Z resolution of $1\ \mu\text{m}$, 50 planes, 10 min) (25% of the CDX2⁺ organoids and 5% of the whole batch). FOV, field of view. Scale bar, $30\ \mu\text{m}$. **d**, Optical zoom using a $\times 60$ WI, 1.27 NA objective on the selected fields of view containing E-cad⁺/CDX2⁺ positive clusters. Scale bar, $30\ \mu\text{m}$. **e**, 2D and 3D reconstruction of the cropped E-cad⁺/CDX2⁺ region. Scale bar, $10\ \mu\text{m}$.

This is likely due to the difference in observed size and growth rate. Nonetheless, organoids grown in JeWells displayed the expected gene expression profile.

Finally, we confirmed the similarity of rosette development comparing their morphologies in organoids that were live imaged then fixed in JeWells versus fixed organoids grown in U-bottom dishes (Extended Data Fig. 10c).

Altogether, these data establish the suitability of the approach for live imaging of organoid development.

Discussion

The performance of our imaging platform is dependent on the capability to grow organoids in the vicinity of micromirrors that are used to perform single-objective light-sheet illumination. The JeWell geometry offers the advantage that only one single organoid grows in each well (no organoid fusions). The spatial registrations of the images during live or correlative imaging are also greatly eased by the geometry. However, the JeWell architecture imposes some constraints on the organoid. From a biological standpoint, the physical

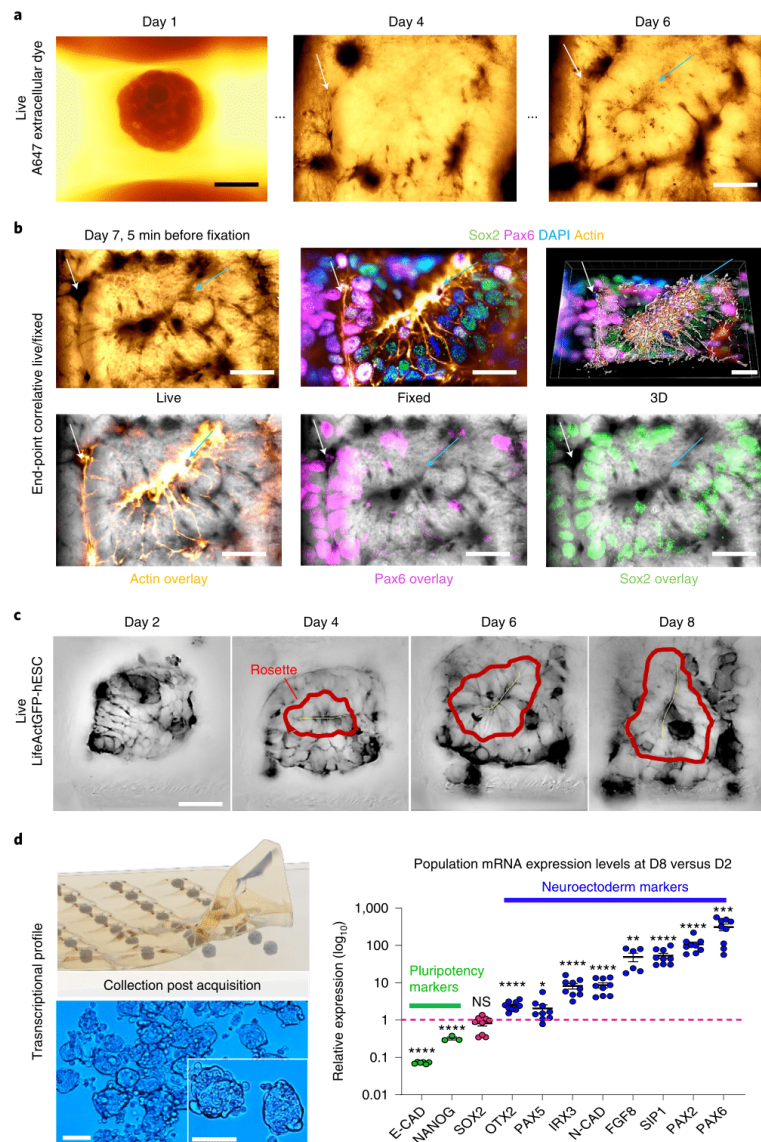


Fig. 6 | Correlative imaging of live versus transcription factors expression. **a**, Live imaging performed on neuroectoderm organoid during 7 days of development. Alexa647 staining of the intercellular clefts. 3D time-lapse imaging (one frame every 15 min for 8 days) revealed the sequential formation of a first (white arrow) and second (blue arrow) rosette (in the organoids). Scale bars 50 μm (left) and 30 μm (right). **b**, Immunostaining images of the organoids at day 7 (actin (gold), DAPI (blue), Pax6 (magenta) and Sox 2 (green)) could be overlaid with the last image of the live imaging (5 min before fixation) by manual registration of the focus offset ($n=6$). It revealed the presence of Pax6⁺ cells exclusively around the first older rosette. Scale bars, 30 μm . **c**, Representative time-lapse imaging of live neuroectoderm (lifeact GFP) ($n=100$) imaged sequentially in 3D (70 optical sections separated by 1 μm), every 15 min during 125 h (spread over eight consecutive days). The rosettes were manually tracked (red contour) from the day of induction (day 2). Scale bar, 50 μm . **d**, Schematic of the peeling of the JeWell chip and collection of the organoids after 3D live monitoring to enable messenger RNA extraction. RT-qPCR performed on 11 genes pluripotent reporters (green) and neuroectoderm markers (blue) ($n=3-6$ replicates, roughly 100-200 organoids per replicate). Scale bar, 150 μm . Statistical tests: P values were obtained from several unpaired t -tests to compare the relevance of the fold change of gene expression to the profile of stem cells at day = 0 on the same batch. P values ****<0.0001, ***<0.001, **<0.01, *<0.05, NS, nonsignificant. Exact P values can be found in Supplementary Table 2. Mean, s.d. and whole distributions are shown for each replicate of each batch.

confinement of the organoid can appear as an impediment to its development. When we pooled 100 of them for the RT-qPCR, the gene expression profile of the neuroectoderm was correct, although it differed for certain genes from the batches grown in U-bottom dishes. At present, we cannot retrieve single organoids and we needed to pool them. On one hand, we lost the singular expression levels for each organoid; on the other, had we still had some measured batch-to-batch variability since 100 organoids do not cover the total range of development diversity within the batch. Hence direct comparison of RT-qPCR from batch to batch or with culture in U-bottom dishes is not straightforward. Our data prove, however, that within a large limit, the inverted pyramid geometry offers a proper access to the culture medium above. The development of the organoids we tested appeared normal in terms of morphology and of the localization of the markers we used.

From an imaging standpoint, the dimension of the JeWell is a key, tunable parameter that has to be carefully optimized. The depth of the Jewell we present here is limited to 200 μm . Using another fabrication technique, we could fabricate deeper prototypic wells (500 μm) the production of which will soon be scaled up, alleviating the height constraints of the technique. The lateral dimensions of JeWells can vary easily from 100 μm to close to a millimeter. They can be tuned to the desired size of the organoids. However, the imaging protocol requires the sample and the mirror to fit together in the field of view of the objective. This restriction is mitigated by the possibility to (1) stitch the images along the direction of the mirror and (2) change the objective magnification. A $\times 60$ magnification limits the imaging region to 200 μm , whereas a $\times 20$ magnification increases it to 600 μm . The constraints in JeWell size as for presented here therefore restrict our technique to 'small' organoids or cell aggregates, and so it is not suited for organoids with a mm³ volume range. However, it still covers a very large range of reported organoid types such as cancer pancreatic³⁴, 100 μm ; intestinal²⁴, 50–200 μm ; gastruloids³⁵, 100 μm in diameter and 400 μm long; liver²³, 300 μm ; micro-kidney³⁶, 150–200 μm and cancer spheroids³⁷, 250–750 μm . Note that for each new system a systematic check of the proper organoid development will still be necessary.

Another salient feature of the technique is the single side illumination that proved a good compromise between speed and quantitative analysis. The CNNs we used to detect, classify and segment the 3D images were relatively insensitive to the gradient of optical sectioning along the direction of the light sheet. To achieve this result, we took great care of training the different network with ground truth images of nuclei, rosettes and mitosis evenly spread over the light-sheet sectioning gradient. The lack of clear bias in the final analysis allows us to use single-sided illumination for fast, although quantitative, illumination. However, there are several ways to improve the quality of the images, such as deconvolution and multi-side illumination¹⁴. Multi-side illumination is straightforward using the four different mirrors composing the JeWell cavities, but it comes with a cost of instrumentation complexity, acquisition time, photobleaching and processing time

Finally, a natural extension of our method is the development of a high-content screening platform allowing multi-condition assays. With the current acquisition speed, the 3D imaging of 9,216 organoids grown in the 96-well plates described in the first result section would take 1.3 days (respectively 4 days) to be imaged in one (respectively, three) color(s). It would result in 0.6 TB (respectively, 1.7 TB) of data. The typical analysis time per organoid for the longest CNN we used was 40 s per organoid on a standard workstation equipped with graphical processing units (GPUs). This suggests that the analysis of the full 96-well plate could be done in about 4 days, and could hence be performed almost in parallel with the acquisition, using a proper workflow. Integrating all the process in a single workflow requires ongoing developments in computer science, bioinformatics, microfabrication and robotization.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at <https://doi.org/10.1038/s41592-022-01508-0>.

Received: 15 December 2020; Accepted: 2 May 2022;

Published online: 13 June 2022

References

- Kim, J., Koo, B. K. & Knoblich, J. A. Human organoids: model systems for human biology and medicine. *Nat. Rev. Mol. Cell Biol.* <https://doi.org/10.1038/s41580-020-0259-3> (2020).
- Takebe, T. & Wells, J. M. Organoids by design. *Science* **364**, 956–959 (2019).
- Kratochvil, M. J. et al. Engineered materials for organoid systems. *Nat. Rev. Mater.* **4**, 606–622 (2019).
- Rossi, G., Manfrin, A. & Lutolf, M. P. Progress and potential in organoid research. *Nat. Rev. Genet.* **19**, 671–687 (2018).
- O'Connell, L. & Winter, D. C. Organoids: past learning and future directions. *Stem. Cells Dev.* **29**, 281–289 (2020).
- Vives, J. & Batlle-Morera, L. The challenge of developing human 3D organoids into medicines. *Stem Cell Res. Ther.* <https://doi.org/10.1186/s13287-020-1586-1> (2020).
- Busslinger, G. A. et al. The potential and challenges of patient-derived organoids in guiding the multimodality treatment of upper gastrointestinal malignancies. *Open Biol.* <https://doi.org/10.1098/rsob.190274> (2020).
- Lukonin, I. et al. Phenotypic landscape of intestinal organoid regeneration. *Nature* <https://doi.org/10.1038/s41586-020-2776-9> (2020).
- Renner, H. et al. A fully automated high-throughput workflow for 3D-based chemical screening in human midbrain organoids. *eLife* <https://doi.org/10.7554/eLife.52904> (2020).
- Bock, C. et al. The Organoid Cell Atlas. *Nat. Biotechnol.* <https://doi.org/10.1038/s41587-020-00762-x> (2021).
- Rios, A. C. & Clevers, H. Imaging organoids: a bright future ahead. *Nat. Methods* <https://doi.org/10.1038/nmeth.4537> (2018).
- Dekkers, J. F. et al. High-resolution 3D imaging of fixed and cleared organoids. *Nat. Protoc.* **14**, 1756–1771 (2019).
- Lukonin, I., Zinner, M. & Liberali, P. Organoids in image-based phenotypic chemical screens. *Exp. Mol. Med.* <https://doi.org/10.1038/s12276-021-00641-8> (2021).
- Wan, Y., McDole, K. & Keller, P. J. Light-sheet microscopy and its potential for understanding developmental processes. *Annu. Rev. Cell Dev. Biol.* <https://doi.org/10.1146/annurev-cellbio-100818-125311> (2019).
- Yang, Q. et al. Cell fate coordinates mechano-osmotic forces in intestinal crypt formation. *Nat. Cell Biol.* <https://doi.org/10.1038/s41556-021-00700-2> (2021).
- Serra, D. et al. Self-organization and symmetry breaking in intestinal organoid development. *Nature* <https://doi.org/10.1038/s41586-019-1146-y> (2019).
- Eismann, B. et al. Automated 3D light-sheet screening with high spatiotemporal resolution reveals mitotic phenotypes. *J. Cell Sci.* <https://doi.org/10.1242/jcs.245043> (2020).
- Voleti, V. et al. Real-time volumetric microscopy of in vivo dynamics and large-scale samples with SCAPE 2.0. *Nat. Methods* <https://doi.org/10.1038/s41592-019-0579-4> (2019).
- Yang, B. et al. Epi-illumination SPIM for volumetric imaging with high spatial-temporal resolution. *Nat. Methods* <https://doi.org/10.1038/s41592-019-0401-3> (2019).
- Maioli, V. et al. Time-lapse 3-D measurements of a glucose biosensor in multicellular spheroids by light sheet fluorescence microscopy in commercial 96-well plates. *Sci. Rep.* <https://doi.org/10.1038/srep37777> (2016).
- Galland, R. et al. 3D high- and super-resolution imaging using single-objective SPIM. *Nat. Methods* <https://doi.org/10.1038/nmeth.3402> (2015).
- Sorrentino, G. et al. Mechano-modulatory synthetic niches for liver organoid derivation. *Nat. Commun.* <https://doi.org/10.1038/s41467-020-17161-0> (2020).
- Fedorova, V. et al. Differentiation of neural rosettes from human pluripotent stem cells in vitro is sequentially regulated on a molecular level and accomplished by the mechanism reminiscent of secondary neurulation. *Stem Cell Res.* <https://doi.org/10.1016/j.scr.2019.101563> (2019).
- Yoshida, S., Miwa, H., Kawachi, T., Kume, S. & Takahashi, K. Generation of intestinal organoids derived from human pluripotent stem cells for drug testing. *Sci. Rep.* <https://doi.org/10.1038/s41598-020-63151-z> (2020).

25. Weigert, M., Schmidt, U., Haase, R., Sugawara, K. & Myers, G. Star-convex polyhedra for 3D object detection and segmentation in microscopy. In *Proc. 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020* 3655–3662 (IEEE, 2019); <https://doi.org/10.1109/WACV45572.2020.9093435>
26. Sun, X. & Kaufman, P. D. Ki-67: more than a proliferation marker. *Chromosoma* <https://doi.org/10.1007/s00412-018-0659-8> (2018).
27. Lancaster, M. A. et al. Cerebral organoids model human brain development and microcephaly. *Nature* <https://doi.org/10.1038/nature12517> (2013).
28. Hříbková, H., Grabiec, M., Klemová, D., Slaninová, I. & Sun, Y. M. Calcium signaling mediates five types of cell morphological changes to form neural rosettes. *J. Cell Sci.* **131**, jcs206896 (2018).
29. Meinhardt, A. et al. 3D reconstitution of the patterned neural tube from embryonic stem cells. *Stem Cell Rep.* <https://doi.org/10.1016/j.stemcr.2014.09.020> (2014).
30. Chandrasekaran, A. et al. Comparison of 2D and 3D neural induction methods for the generation of neural progenitor cells from human induced pluripotent stem cells. *Stem Cell Res.* <https://doi.org/10.1016/j.scr.2017.10.010> (2017).
31. Ronneberger, O., Fischer, P., Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: *Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science()*, vol 9351. Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28
32. Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. Densely connected convolutional networks. In *Proc. 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 21–26 July, Honolulu, HI, USA* (IEEE, 2017); <https://doi.org/10.1109/CVPR.2017.243>
33. Karzbrun, E., Kshirsagar, A., Cohen, S. R., Hanna, J. H. & Reiner, O. Human brain organoids on a chip reveal the physics of folding. *Nat. Phys.* <https://doi.org/10.1038/s41567-018-0046-7> (2018).
34. Driehuis, E. et al. Pancreatic cancer organoids recapitulate disease and allow personalized drug screening. *Proc. Natl Acad. Sci. USA* <https://doi.org/10.1073/pnas.1911273116> (2019).
35. Beccari, L. et al. Multi-axial self-organization properties of mouse embryonic stem cells into gastruloids. *Nature* <https://doi.org/10.1038/s41586-018-0578-0> (2018).
36. Kumar, S. V. et al. Kidney micro-organoids in suspension culture as a scalable source of human pluripotent stem cell-derived kidney cells. *Dev.* <https://doi.org/10.1242/dev.172361> (2019).
37. Perche, F. & Torchilin, V. P. Cancer cell spheroids as a model to evaluate chemotherapy protocols. *Cancer Biol. Ther.* <https://doi.org/10.4161/cbt.21353> (2012).

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© The Author(s), under exclusive licence to Springer Nature America, Inc. 2022