Documentation

Segmentation de la dentine

by Synchrotron SOLEIL - HELIOBIO - Sandra SARA

Description de la segmentation en utilisant les différents méthodes et logiciels.

Contents

1. Installation de l'environnement Python 2
2. Scripts de segmentation 2
2.1. Segmentation watershed 2D 2
2.1.1. Mise à jour des chemins pour les fichiers et les images 4
2.1.2. Vérification du script watershed_2D 6
2.1.3. Lancement du script watershed_2D avec python 6
2.2. Segmentation watershed 3D 7
2.3. Segmentation par K- means 7
3. Segmentation sur le logiciel Dragonfly 8
3.1. Chargment des données 8
3.1.1. Paramètres d'image 9
3.2. Chargement des masques de segmentation 9
3.3. Segmentation de la lumière tubulaire 10
3.3.1. Extraction des ROIs à partir du masque lumiere-3d 10
3.3.2. Suppresion du fond (ou background) 12
3.3.3. Nettoyage de la ROI lumière 14
3.3.3.1. Extraction des composants connectés 14
3.3.3.2. Suppression des pixels de bruits (semi_automatique) 15
3.3.3.3. Vérification manuelle des ROIs 17
3.3.3.4. Correction manuelle de la segmentation 18
3.4. Segmentation de l'extérieur 20
3.4.1. Segmentation de l'extérieur 21
3.5. Segmentation de la matrice 23
3.5.1. Importation des masques Collagène et HAP 23
3.5.2. Extraction des ROIs du masque HAP 24
3.5.3. Extraction des ROIs du masque Collagène 26

§1. Installation de l'environnement Python

Pour exécuter les différents scripts Python de segmentation de la dentine, il est nécessaire de créer un environnement Python, puis d'y installer les bibliothèques requises. Voici les étapes à suivre :

- Ouvrir un terminal et suivez les lignes de codes suivantes:

```
# Vérifier la version de python
python --version
# La version de python utilisé est Python 3.12.9

# Créer un environnement virtuel appelé seg_dentine avec venv
python -m venv pyenv_seg_dentine

# Activer l'environnement seg_dentine
source pyenv_seg_dentine/bin/activate

# Installation des bibliothèques
pip install -r /path/to/requirements.txt
```

§2. Scripts de segmentation

La segmentation des jeux de données s'effectue à l'aide de scripts Python. Toutefois, il est recommandé de tester la segmentation individuellement pour chaque jeu de données à l'aide des notebooks Jupyter.

Le dossier **02_pipeline** est organisé en deux sous-dossiers principaux :

Notebooks

Ce dossier contient les notebooks Jupyter (.ipynb), qui permettent une exploration interactive du code et des résultats. Ils sont utiles pour tester et comprendre les étapes de traitement sur des cas individuels. On y trouve :

- **2D_watershed.ipynb** : réalise un traitement 2D sur une seule image.
- **3D_watershed.ipynb** : applique un traitement 3D sur un volume.

Scripts

Ce dossier contient les scripts Python (.py). Ils sont conçus pour être exécutés de manière autonome et permettent de traiter l'ensemble du jeu de données. Les résultats, notamment les masques de segmentation, sont automatiquement sauvegardés. On y trouve :

- **2D_watershed.py** : traite les images en 2D (une image à la fois).
- **3D_watershed.py** : traite les volumes en 3D.

§2.1. Segmentation watershed 2D

Pour tester la segmentation watershed 2D sur votre jeu de données, lancez Jupyter Lab depuis un terminal.

```
# Activer l'environnement Python nommé seg_dentine
source seg_dentine/bin/activate

# Se déplacer dans le répertoire ou vous avez sauvegardé les fichiers et les
scripts avec la commande cd

# Lancer Jupyter Lab
jupyter lab
```

Une nouvelle fenêtre Jupyter Lab s'ouvrira automatiquement dans votre navigateur.

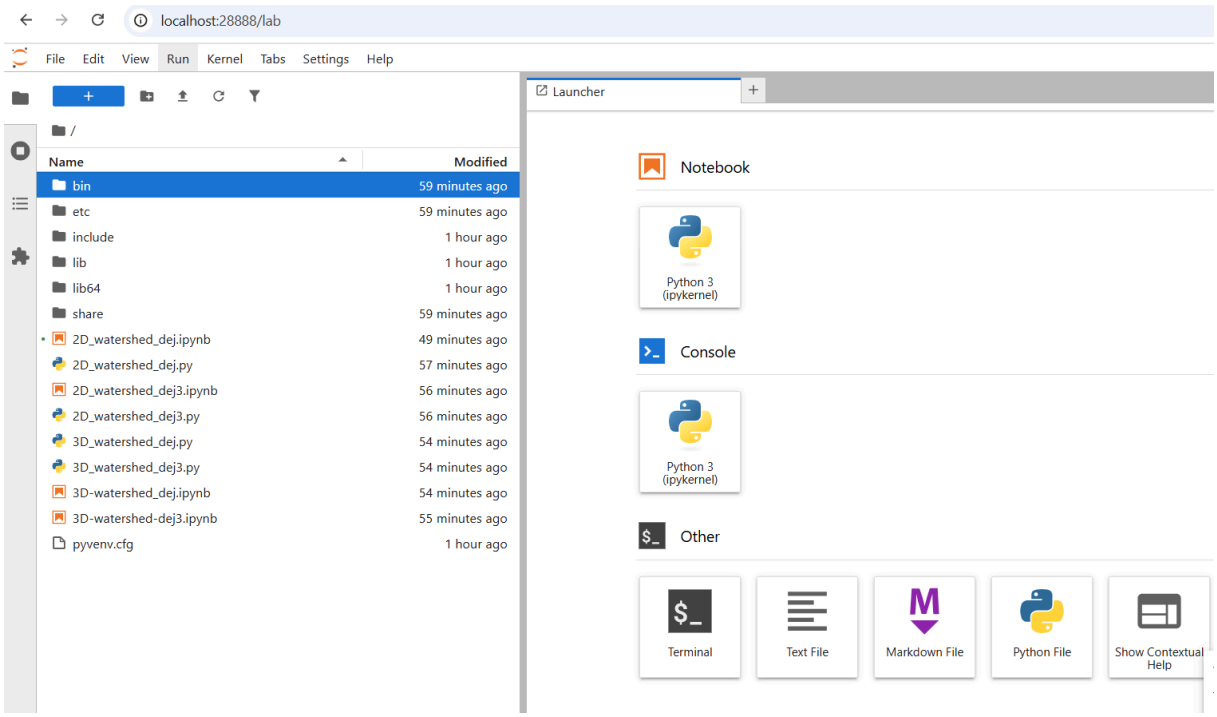

Figure 1: Fenêtre: Jupyter lab

Dans Jupyter Lab, double-cliquez sur le fichier suivant pour lancer le notebook et exécuter les cellules de segmentation.

`2D_watershed_dej.ipynb`

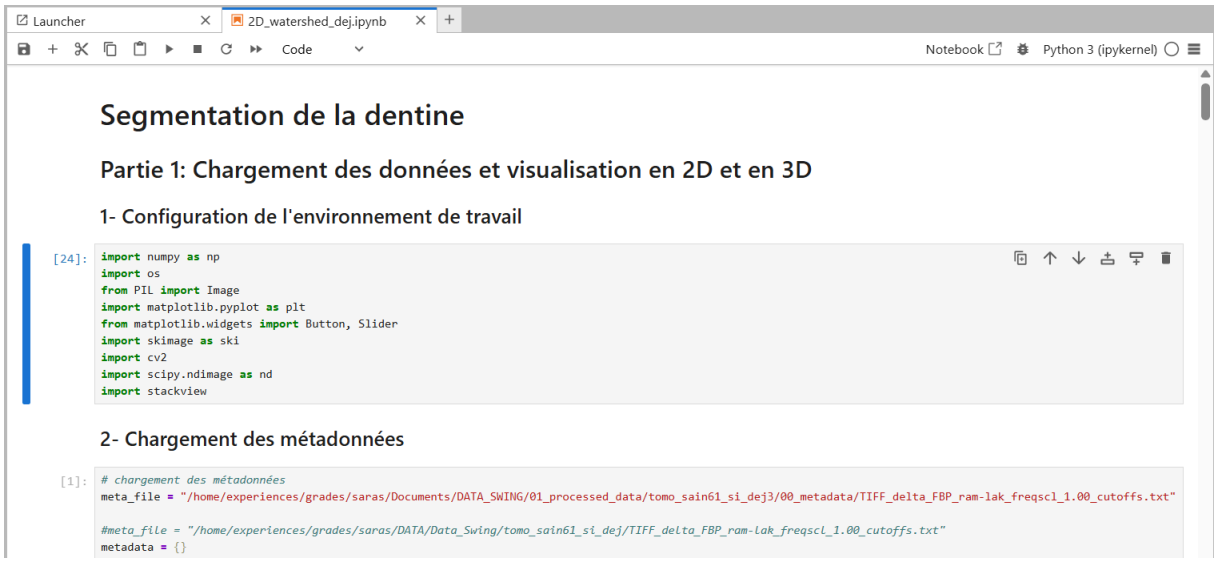

Figure 2: Jupyter lab: 2D_watershed_dej.ipynb

§2.1.1. Mise à jour des chemins pour les fichiers et les images

Dans la Partie 1, sections 2 et 3, il est nécessaire de mettre à jour les chemins d'accès aux images ainsi qu'au fichier cutoffs.txt, afin qu'ils correspondent à l'emplacement réel de vos données.

Pour charger le fichier cutoffs.txt correspondant à votre jeu de données, indiquez son chemin dans la variable meta_file.

```
# Chargement des métadonnées
meta_file = "/home/experiences/grades/saras/Documents/DATA_SWING/01_processed_data/
tomo_sain61_si_dej3/00_metadata/TIFF_delta_FBP_ram-lak_freqscl_1.00_cutoffs.txt"
metadata = {}
with open(meta_file, "r") as file_in:
    for line in file_in:
        lines = line.split()
        if lines[1]== "Conversion":
            break
        metadata[lines[1]]= float(lines[-1])

print(metadata)
type(metadata)
```

Cette commande doit afficher les informations contenues dans le fichier cutoffs.txt, comme illustré dans la figure ci-dessous :

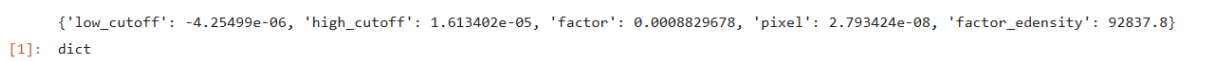

Figure 3: Chargement des métadonnées

Pour charger vos images en 2D, indiquez son chemin dans la variable raw_dir.

```
# Charger le chemin des images
raw_dir = "/home/experiences/grades/saras/Documents/DATA_SWING/01_processed_data/
tomo_sain61_si_dej_S00017_to_S00504_480x480_gpu_1modes_200DM500ML_recons_S/00_crop_data/"
images = sorted([img for img in os.listdir(raw_dir) if img.endswith(".tif")])

# Vérifier le trie des images en affichant les 5 premières images
print("premières images triées", images[:5])

# Charger les images dans un stack en 2D
stack_2d = np.array([ski.io.imread(os.path.join(raw_dir, img)) for img in images])

# Visualisation interactive avec stackview
stackview.slice(stack_2d)
```

Cette commande permet d'afficher votre jeu de données avec un slicer interactif pour explorer l'ensemble du volume, comme montré dans la figure ci-dessous :

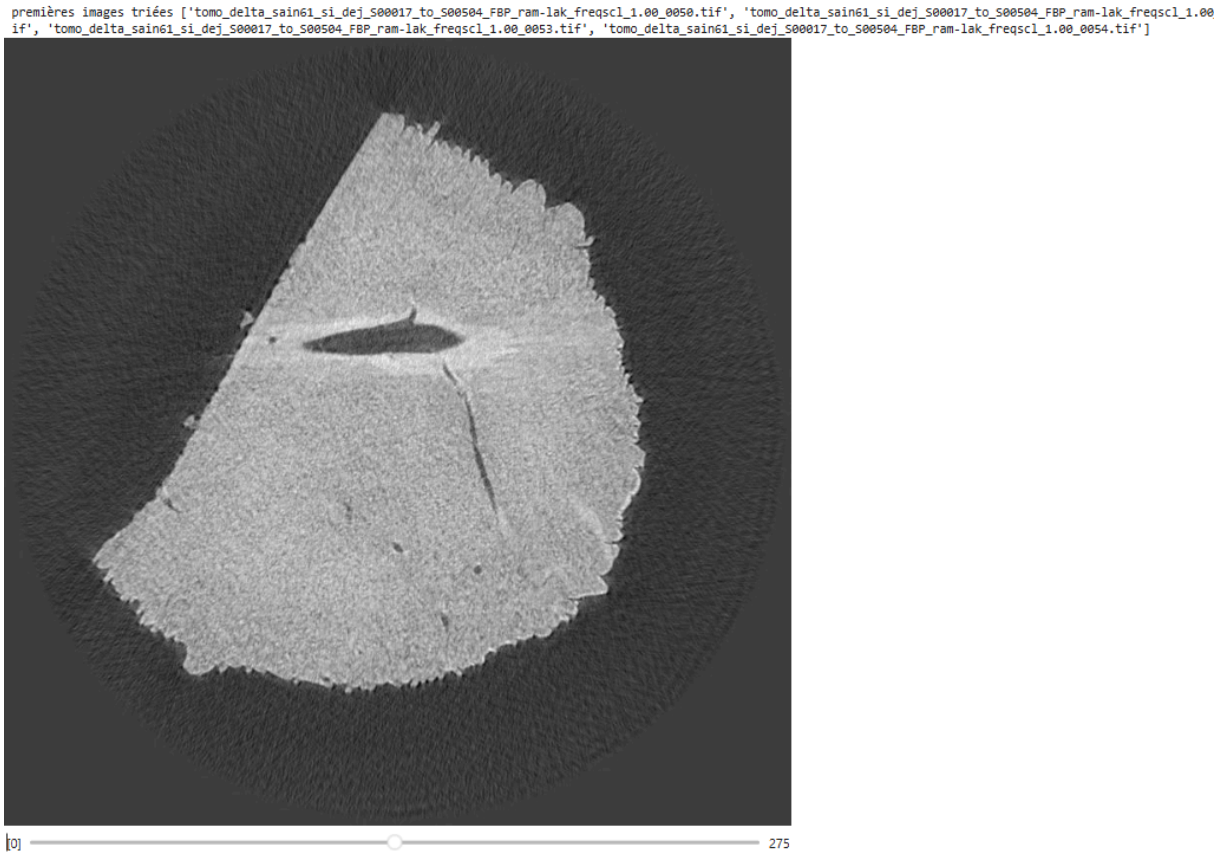

Figure 4: Chargement des images en 2D

La suite du notebook Jupyter vous permet d'appliquer les différentes opérations sur une image sélectionnée, afin de vérifier la qualité de la segmentation avant d'exécuter le script sur l'ensemble des images.

Une fois la vérification terminée, ouvrez un terminal dans Jupyter Lab pour lancer le script sur toutes les images.

§2.1.2. Vérification du script watershed_2D

Ouvrez le fichier 2D_watershed_dej.py dans Jupyter Lab.

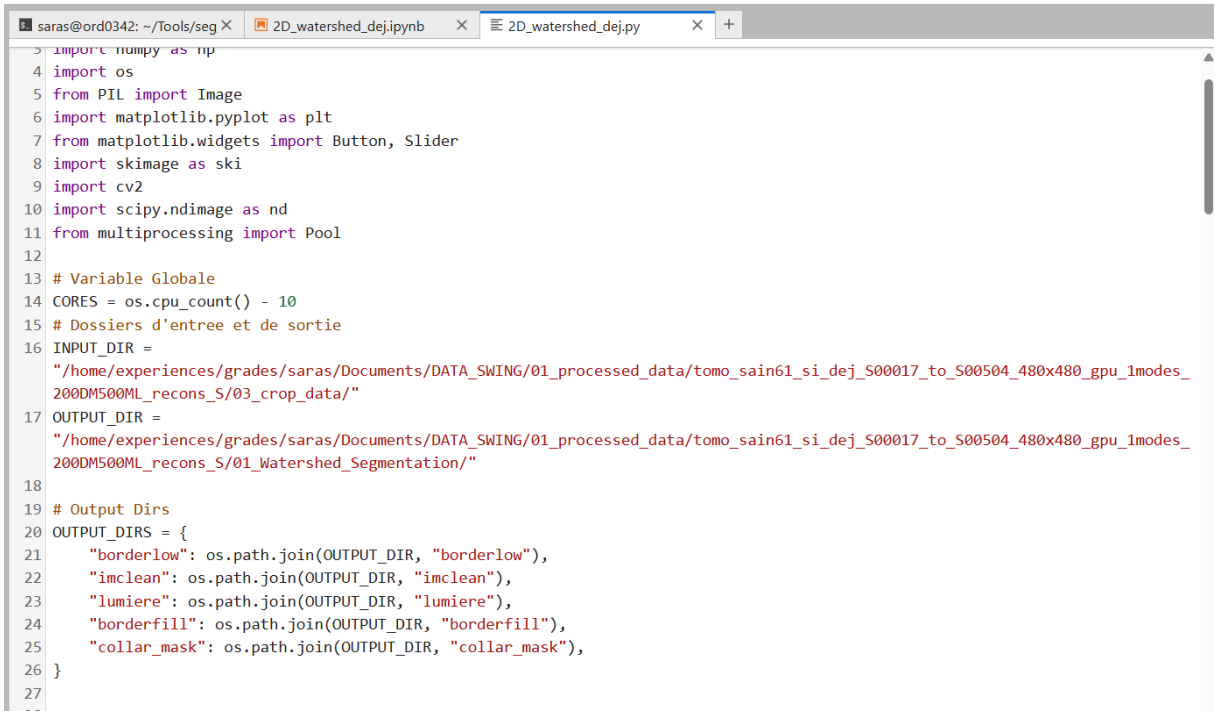

Figure 5: Script Python Watershed 2D

Avant d'exécuter ce script, modifiez les deux lignes suivantes :

- Ligne 16 : INPUT_DIR — Spécifiez le chemin d'accès aux images 2D de votre jeu de données.
- Ligne 17 : OUTPUT_DIR — Indiquez le chemin vers un dossier vide, qui servira à sauvegarder les résultats et les masques de sortie.

Sauvegarder les changements et ouvrir un nouveau terminal

§2.1.3. Lancement du script watershed_2D avec python

- Vérifier si l'environnement venv est activé

Pour savoir si un environnement virtuel Python (venv) est activé, vous pouvez vérifier dans votre terminal si le nom de l'environnement apparaît dans le prompt. Lorsque l'environnement est activé, son nom s'affiche généralement au début de la ligne de commande entre parenthèses. Exemple :

(seg_dentine) usermachine: /project

Ensuite, lancer la commande suivante:

```
# Se déplacer dans le dossier seg_dentine
cd seg_dentine

# Lancer le scripts
python 2D_watershed_dej.py
```

Ce script génère trois masques distincts, nommés comme suit :

- borderlow
- lumiere
- borderfill

§2.2. Segmentation watershed 3D

Comme pour la segmentation 2D, plusieurs étapes sont nécessaires :

- Vérifier la segmentation en exécutant le notebook Jupyter correspondant (3D_watershed.ipynb
- Adapter les chemins d'accès aux données :

  Attention, pour la 3D, il faut spécifier un seul fichier TIFF contenant lintégralité du volume de données.
- Lancer le traitement complet en exécutant le script Python 3D_watershed.py.

Ce script génère automatiquement trois masques de segmentation au format TIFF:

- Lumiere_3d.tiff : correspond à la lumière ou région principale segmentée,
- seeds_3d.tiff : contient les marqueurs utilisés pour l'algorithme watershed,
- borderlow_3d.tiff : représente les frontières ou zones de transition détectées.

§2.3. Segmentation par K- means

La segmentation de l'hydroxyapatite, du collagène et de l'extérieur de la structure est réalisée à l'aide d'une méthode de classification non supervisée : K-means.

Le script correspondant se trouve dans le notebook Jupyter K_means_segmentation.ipynb. Ce notebook permet de charger un volume de données 3D, d'appliquer la segmentation par K-means, puis d'enregistrer les masques segmentés pour l'air, l'HAP et le collagène.

§3. Segmentation sur le logiciel Dragonfly

Une fois que les deux scripts ont étés lancés sous python, ouvrir le logiciel Dragonfly ORS.

§3.1. Chargment des données

Pour charger la séquence d'image dans Dragonfly (Version 2024.1):

- Ouvrir une nouvelle session Dragonfly
  - Dans le **menu** déroulant, sélectionnez :
    - **File**
      - **Import Image Files...**
        - Une fenêtre intitulée **"Import image"** s'affiche à l'écran.

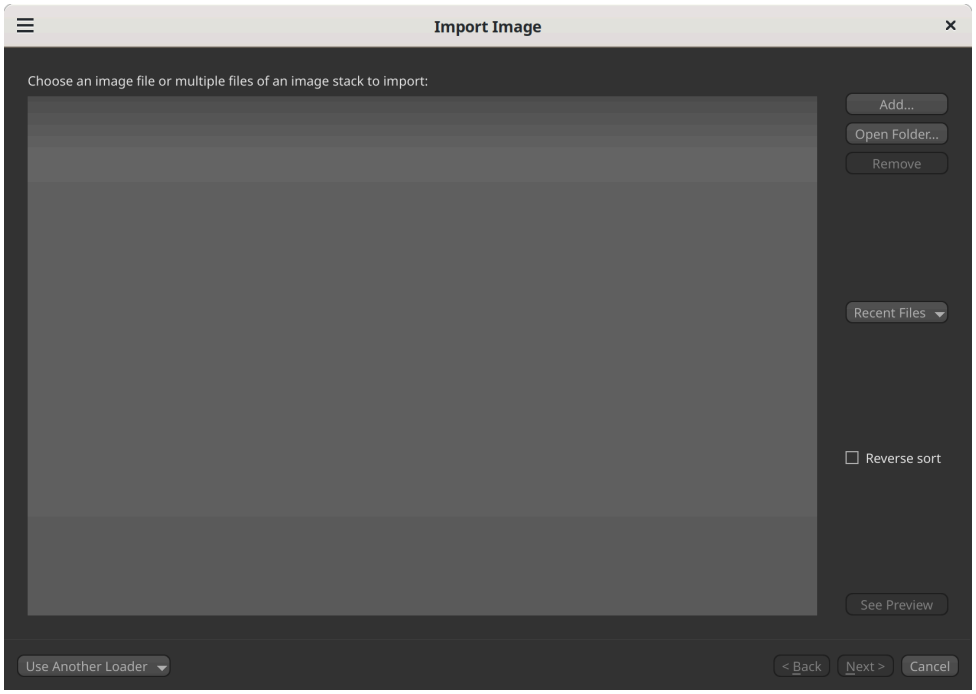

Figure 6: Interface: Import Image

Pour charger vos images:

- Cliquer sur: **Add**
  - Sélectionner les fichiers à charger
    - Cliquer sur : **Next**
      - La fenêtre des paramètres d'images s'affiche à l'écran
        - Choisir les paramètres de vos images (cf Paramètres d'image)
          - Cliquer sur : **Finish**

§3.1.1. Paramètres d'image

Lors de l'importation d'une séquence d'images, ou de données brutes, il est essentiel de spécifier les deux paramètres suivants:

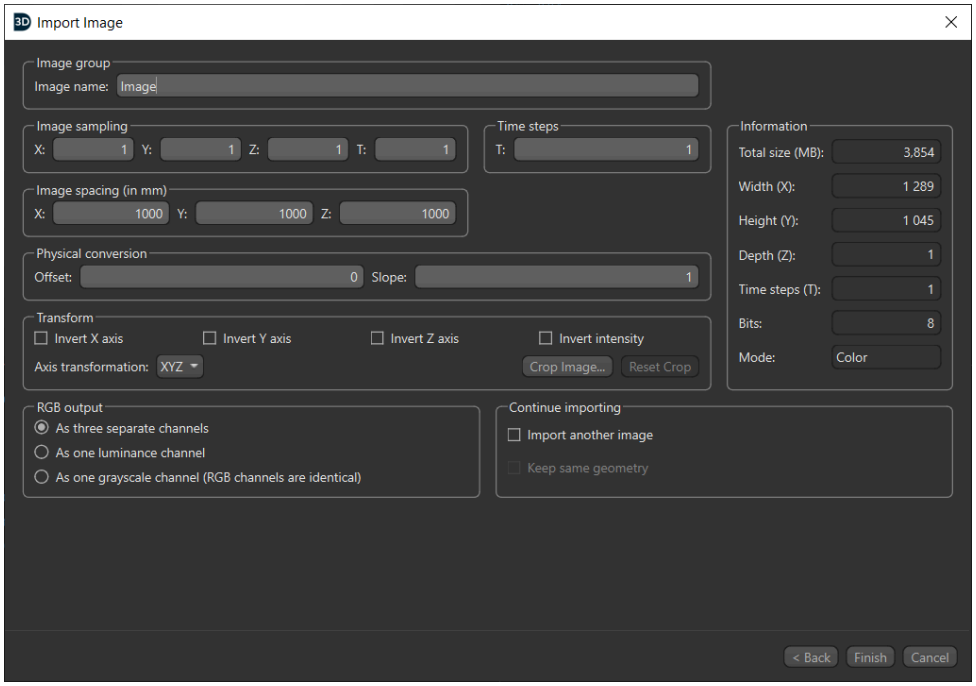

Figure 7: Les paramètres d'image

1. **Image Group :** Donner un nom à vos données. Exemple: "Dataset-Dejé
2. **Image spacing (in nm) :** C'est la taille de pixel. Vous pouvez trouver la taille de pixel dans le fichier "cutoffs.txt" fournie avec vos données.

Plus d'informations sont présentes via le lien suivant: Paramètres d'image

§3.2. Chargement des masques de segmentation

Une fois le jeu de données chargé, nous pouvons désormais importer les masques générés par le script Python.

Le processus d'importation des masques dans Dragonfly est identique à celui utilisé pour les images.

Il est essentiel de s'assurer que la taille de pixel spécifiée correspond à celle utilisée lors du chargement des images.

Les masques qu'ils faut charger sont:

- lumiere_3d.tiff : dans le dossier 003_watershed_3d
- borderlow
- borderfill

§3.3. Segmentation de la lumière tubulaire

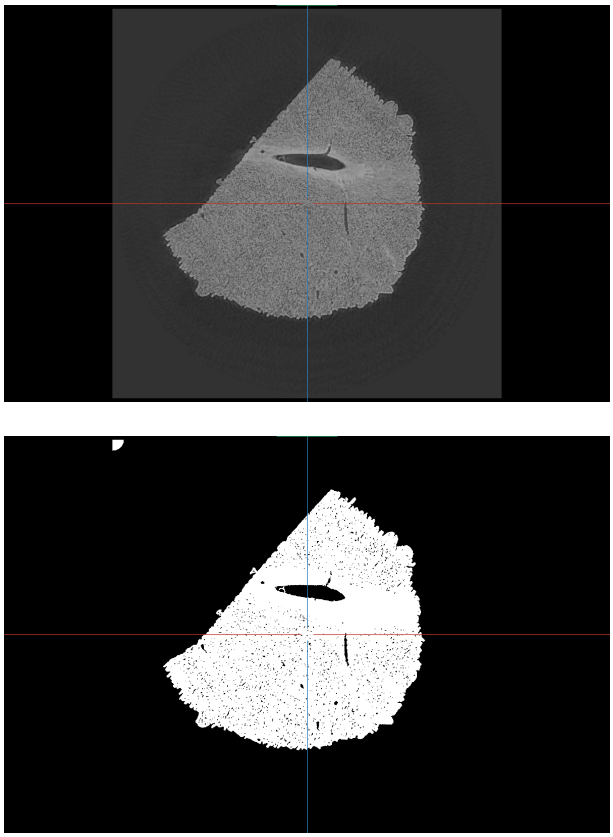

Figure 8: En haut, l'image brut et en bas le masque lumiere_3d importés.

Les masques importés sont des fichiers binaires, et non des régions d'intérêt (ROIs). Pour pouvoir les analyser, il est nécessaire d'extraire les ROIs à partir de ces masques.

§3.3.1. Extraction des ROIs à partir du masque lumiere-3d

Pour extraire les ROIs:

└── Sélectionner le masque **lumiere-3d** dans **Data Properties and Settings**
  └── Faire un clic droit sur le masque
    └── Cliquer sur : **Extract ROIs**

Cette opération génère deux ROIs distinctes :

- La première ROI (en bleu) correspond à l'extérieur de la structure, incluant la lumière tubulaire.
- La deuxième ROI (en orange) correspond à l'intérieur de la structure, excluant la lumière tubulaire.

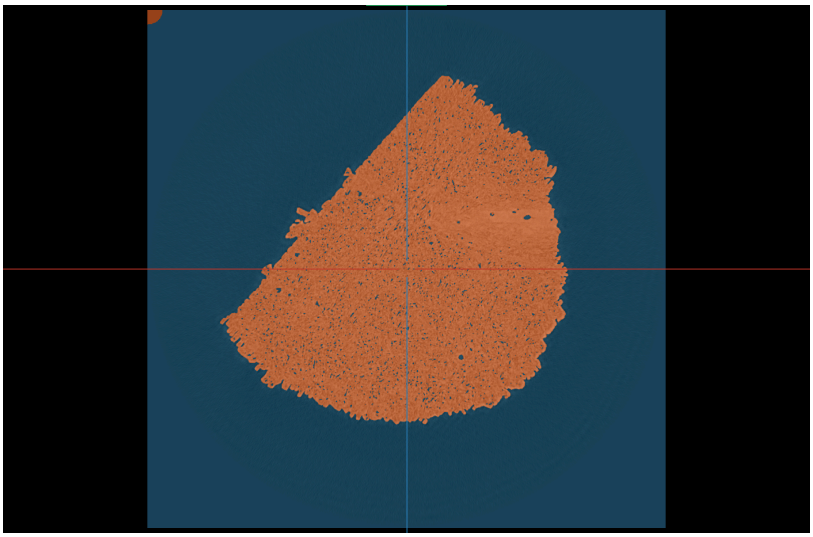

Figure 9: Les paramètres d'image

💡 **Conseil :** Pensez à renommer les ROIs générées à chaque étape afin d'éviter toute confusion lors des manipulations et analyses ultérieures.

Dans ce cas, nous vous recommandons de renommer les ROIs comme suit:

- ROI bleue (extérieur) → **Background + lumière**
- ROI orange (intérieur) → **Matrice**

Notre objectif est d'obtenir une ROI représentant uniquement la lumière tubulaire, sans inclure l'extérieur de la structure. Pour cela, nous allons utiliser le masque **borderlow_stack**.

Avant de poursuivre la segmentation de la lumière tubulaire, il est nécessaire d'extraire les ROIs à partir du masque borderlow_stack, en suivant la même procédure que celle utilisée pour le masque lumiere_3d.

Le masque borderlow_stack permet de générer les deux ROIs suivantes:

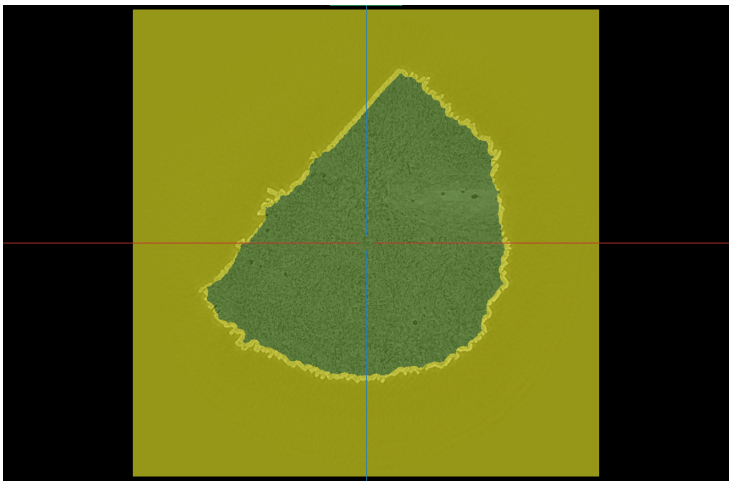

Figure 10:

- ROI extérieur (en jaune)
- ROI intérieur (en vert)

Nous allons renommer les ROIs:

- ROI jaune (extérieur) → **Background**
- ROI verte (intérieur) → **Structure**

§3.3.2. Suppresion du fond (ou background)

Pour isoler uniquement la lumière tubulaire, nous allons effectuer une opération booléenne de soustraction.

Étapes pour extraire la lumière tubulaire :

└── Sélectionner la ROI: **Background + lumiere**
  ├── Maintenir la touche CTRL enfoncée pour ajouter une sélection
  └── Sélectionner ensuite la ROI : **Background**
    └── Une fenêtre Boolean Operation s'ouvre automatiquement

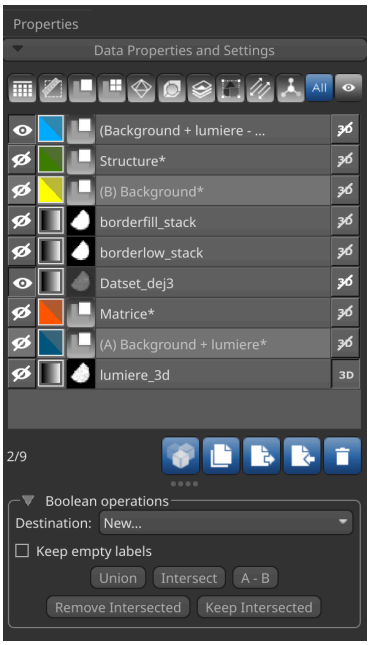

Figure 11: Boolean operations

└— Choisir l'option : **A - B**

└— Une nouvelle fenêtre intitulée New Region of Interest apparaît

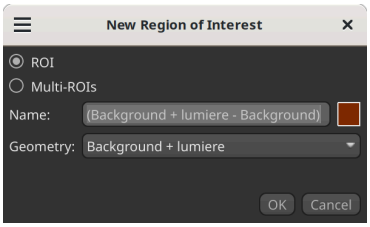

Figure 12: Boolean operations

└— Sélectionner le type ROI et définir le nom : lumière

└— Cliquer sur : **OK**

Nous obtenons ainsi une ROI correspondant à la lumière tubulaire.

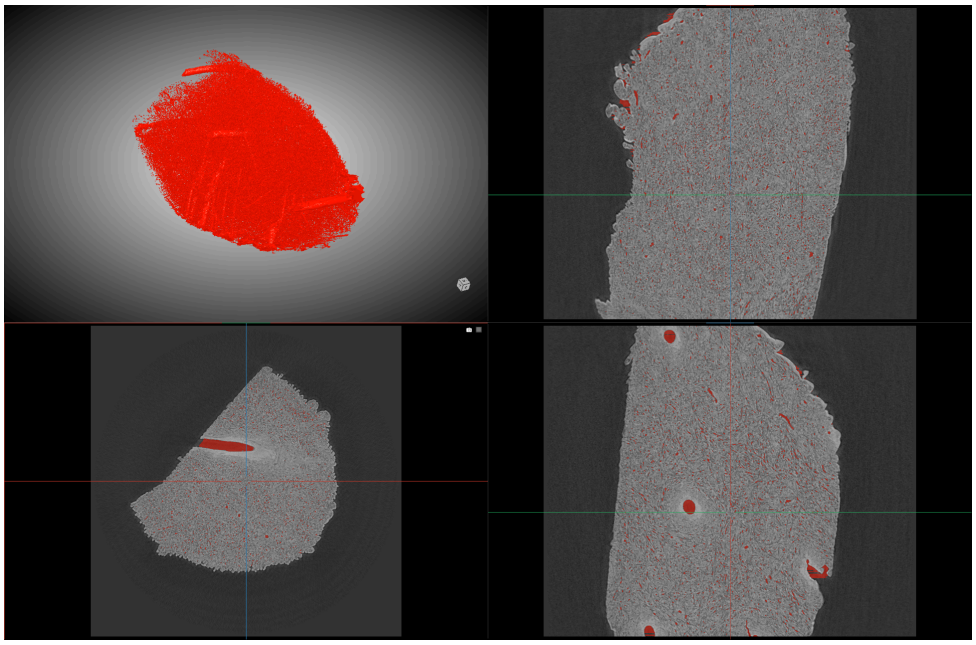

Figure 13: ROI Lumiere

La ROI lumière contient bien les zones associées à la lumière tubulaire, mais inclut également d'autres régions non désirées, telles que le collagène. Il est donc nécessaire de nettoyer cette ROI à l'aide d'un processus combinant des étapes semi-automatiques et manuelles.

§3.3.3. Nettoyage de la ROI lumière

§3.3.3.1. Extraction des composants connectés

La première étape du nettoyage consiste à distinguer les pixels appartenant à des structures tubulaires (branches tubulaires et intertubulaires) des pixels isolés ou du bruit.

Étapes à suivre :

└— Sélectionner la ROI: **lumiere**
  └— Faire un clic droit sur la ROI
    └— Choisir l'option : **Connected Components**
      └— Sélectionner : **New Multi-ROI (26 connected)**

Une nouvelle Multi-ROI est alors générée, contenant les différents composants connectés détectés dans la ROI lumière.

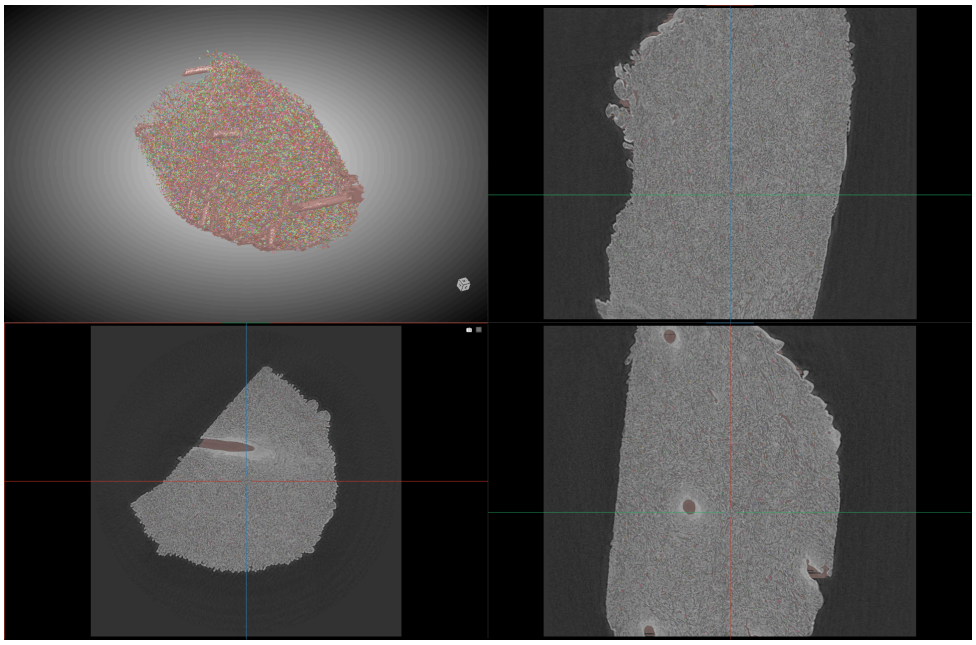

Figure 14: Multi-ROI lumiere

§3.3.3.2. Suppression des pixels de bruits (semi_automatique)

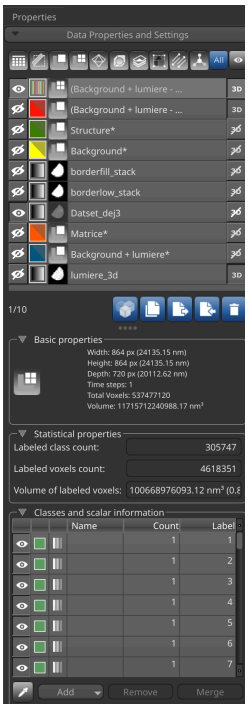

Figure 15: Multi-ROI lumiere

La Multi-ROI générée contient 305 747 ROIs.

La majorité de ces régions correspondent à du bruit, introduit notamment par la segmentation via la méthode watershed. Afin de ne conserver que les régions pertinentes, nous allons supprimer les petits objets ne représentant pas la lumière tubulaire.

En pratique, les ROIs de taille inférieure à 200 voxels sont généralement considérées comme du bruit. Ce seuil a été établi à la suite d'une analyse statistique portant sur plus de 7 000 ROIs issues du jeu de données Dej.

Pour supprimer les petites ROIs considérées comme du bruit :

└— Sélectionner la Multi-ROI: **lumiere**

└— Dans la section **Classes and Scalar Information**, identifier et sélectionner les ROIs à supprimer

└— Cliquer sur **Remove** pour supprimer la sélection

⚠️ Attention : Ne sélectionnez pas tous les pixels en une seule fois, cela risque de faire planter le logiciel. Procédez par lots ou par sélection progressive pour éviter les surcharges.

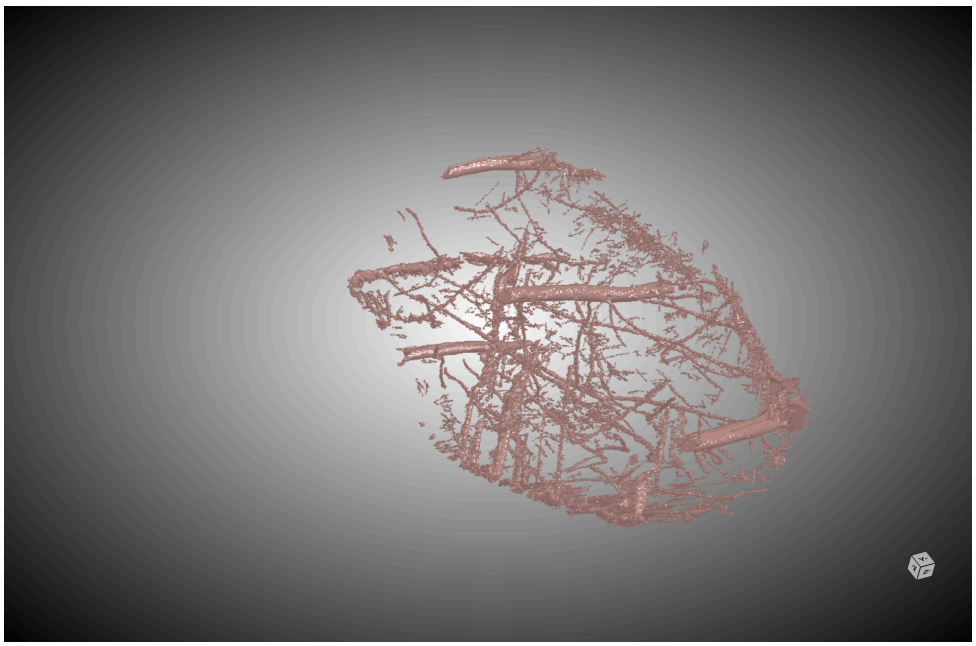

Figure 16: Multi-ROI lumiere après suppression des ROIs < 200 voxels

§3.3.3.3. Vérification manuelle des ROIs

Sur ce jeu de données, 305 086 ROIs ont été supprimées sur un total initial de 305 747, laissant 661 ROIs à analyser manuellement.

L'objectif est de déterminer, pour chacune de ces ROIs restantes, si elle appartient à la classe lumière tubulaire ou à la classe bruit. Cette vérification se fait ROI par ROI.

Étapes pour examiner chaque ROI :

└— Sélectionner la Multi-ROI: **lumiere**
└— Dans la section **Classes and Scalar Information**, sélectionner une ROI
└— Faire un clic droit sur la ROI sélectionnée
└— Cliquer sur : **"Look at Center of Mass"**

Cette action permet de centrer automatiquement l'affichage sur la ROI sélectionnée, facilitant ainsi son évaluation visuelle.

Exemple de figure:

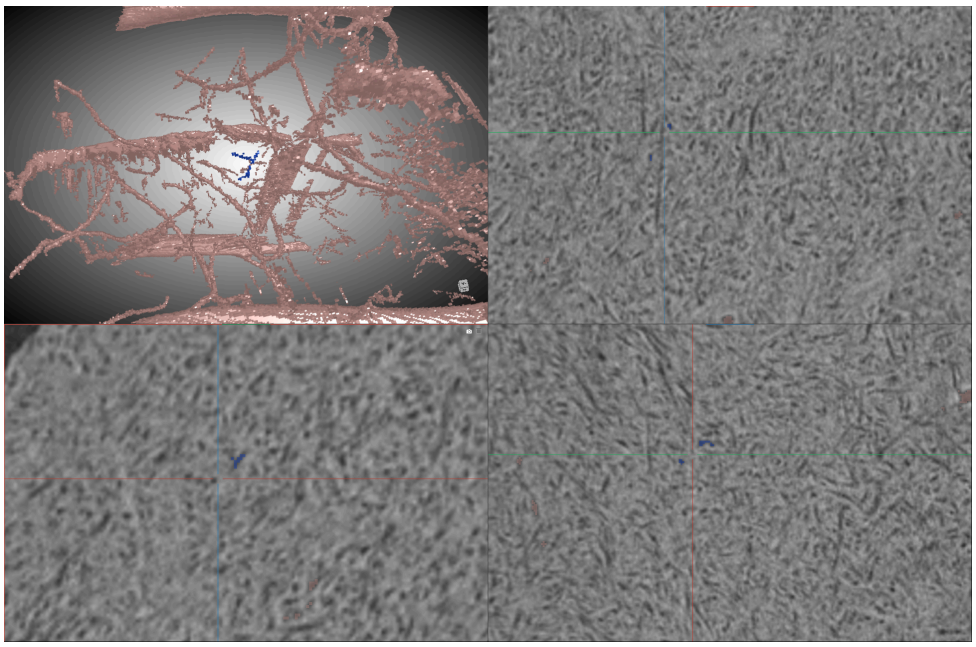

Figure 17: ROI en bleus d'une taille de 223 Voxels qui ne correspond pas à de la lumière tubulaire

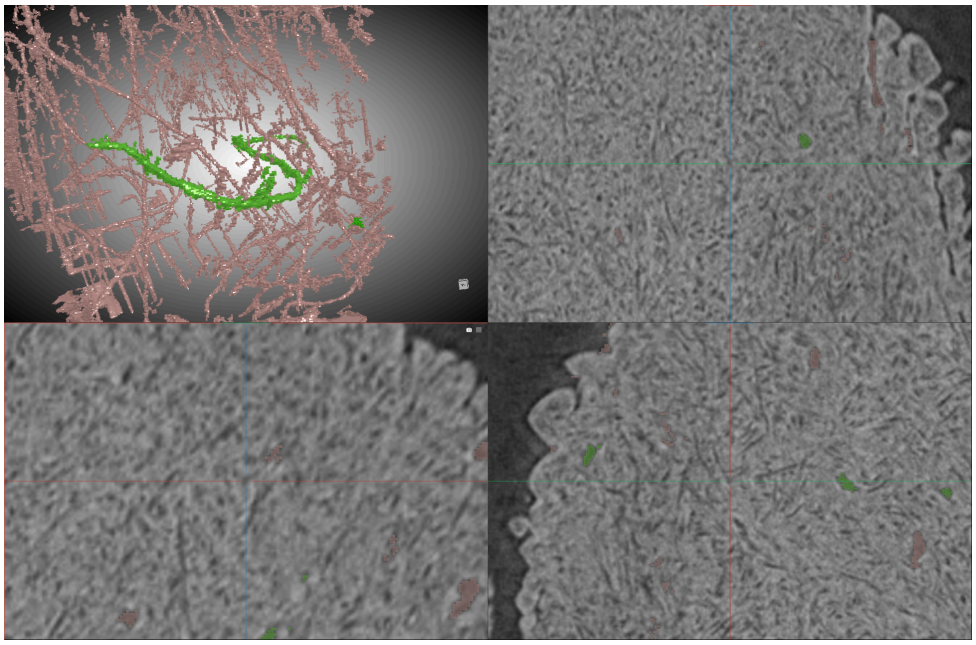

Figure 18: ROI en Vert d'une taille de 12405 Voxels qui correspond à de la lumière tubulaire

§3.3.3.4. Correction manuelle de la segmentation

Après l'analyse complète de l'ensemble des ROIs, certaines régions de la lumière tubulaire peuvent demeurer partiellement non détectées par l'algorithme Watershed 3D. Une intervention manuelle est alors requise pour corriger ces omissions, comme le montre l'exemple ci-dessous :

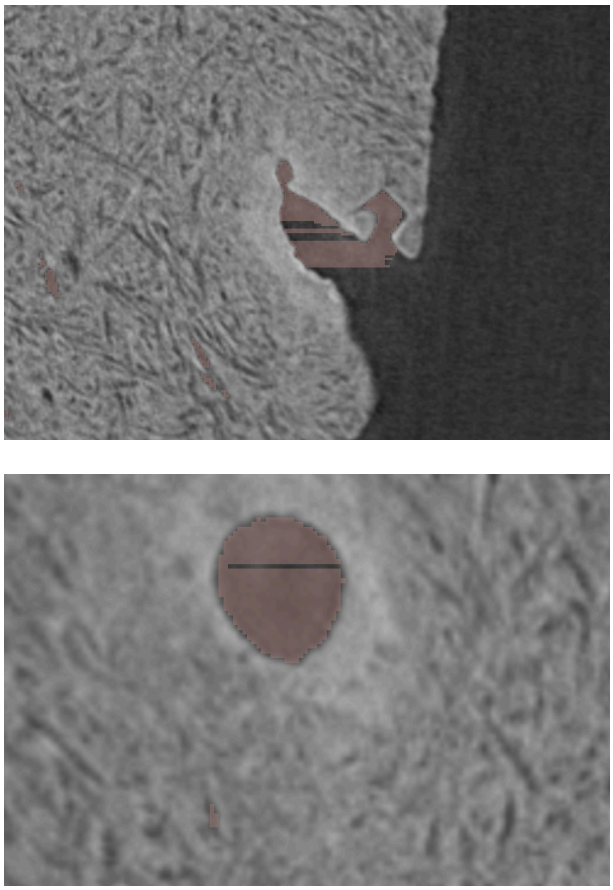

Figure 19: Exemple de correction manuelle nécessaire

Une fois la correction manuelle terminée, la segmentation 3D de la lumière tubulaire se présente comme suit pour le jeu de données tomo_sain61_si_dej3

Figure 20: Segmentation de la lumière tubulaire en 3D

§3.4. Segmentation de l'extérieur

Nous avons besoin de séparer l'extérieur de la zone intérieur ou structure pour les analyses quantitatives et statistiques

Pour cela, nous importerons dans Dragonfly le masque nommé mask_air.tif, généré par le script segmentation_kmeans.py. L'importation du masque suit la même procédure que celle décrite dans la section 3.1 – Chargement des données.

Dans le cas du jeu de données Dej, le masque de l'air est illustré ci-dessous :

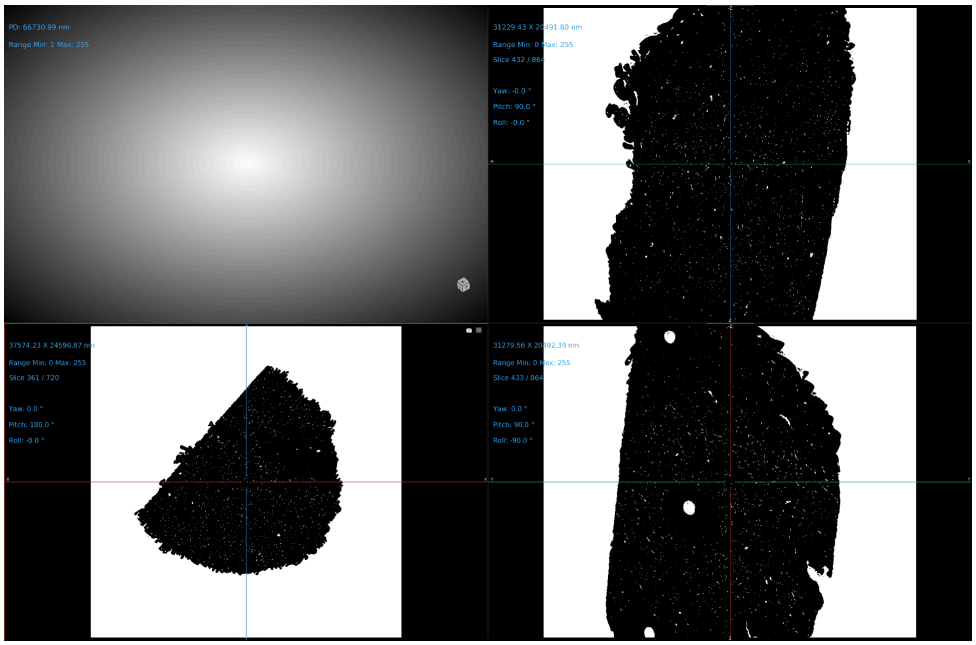

Figure 21: Masque de l'air

Une fois le masque importé, nous procédons à l'extraction des ROIs.

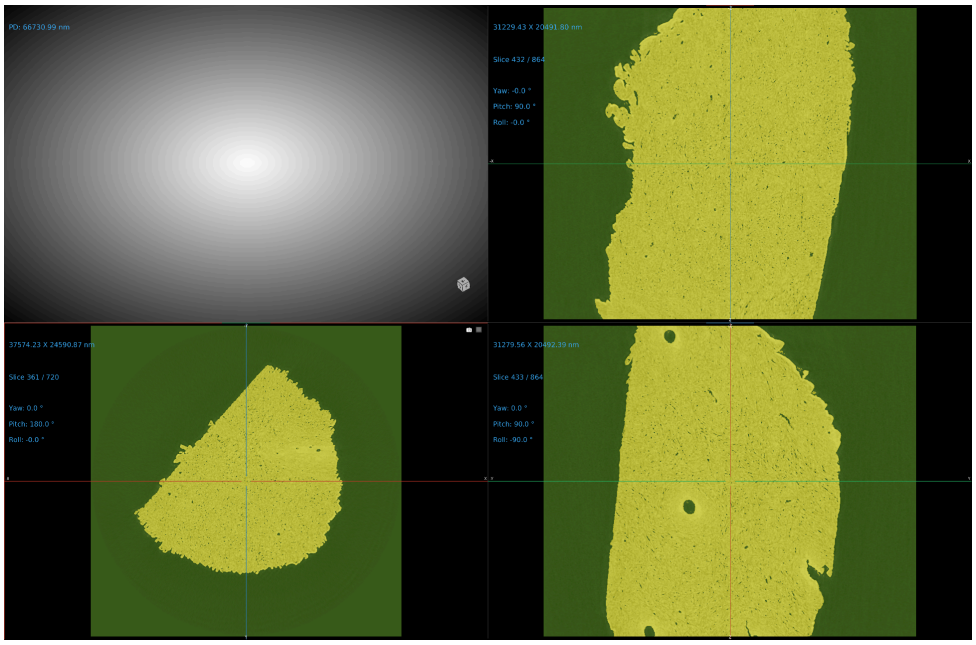

Figure 22: ROI extraite à partir du masque de l'air

Ce masque permet d'obtenir deux ROIs distinctes : l'une représentant l'ensemble de la structure, et l'autre correspondant à son environnement extérieur.

§3.4.1. Segmentation de l'extérieur

La ROI affichée en vert représente à la fois l'extérieur de l'échantillon, la lumière tubulaire, ainsi que certaines zones de collagène.

Etape 1: Suppression de l'air

Nous allons soustraire la ROI verte, nommée OUTSIDE, de la ROI correspondant à la lumière tubulaire précédemment segmentée.

Cette opération permet d'isoler une ROI représentant uniquement l'extérieur de la structure, ainsi que quelques petits éléments résiduels de collagène.

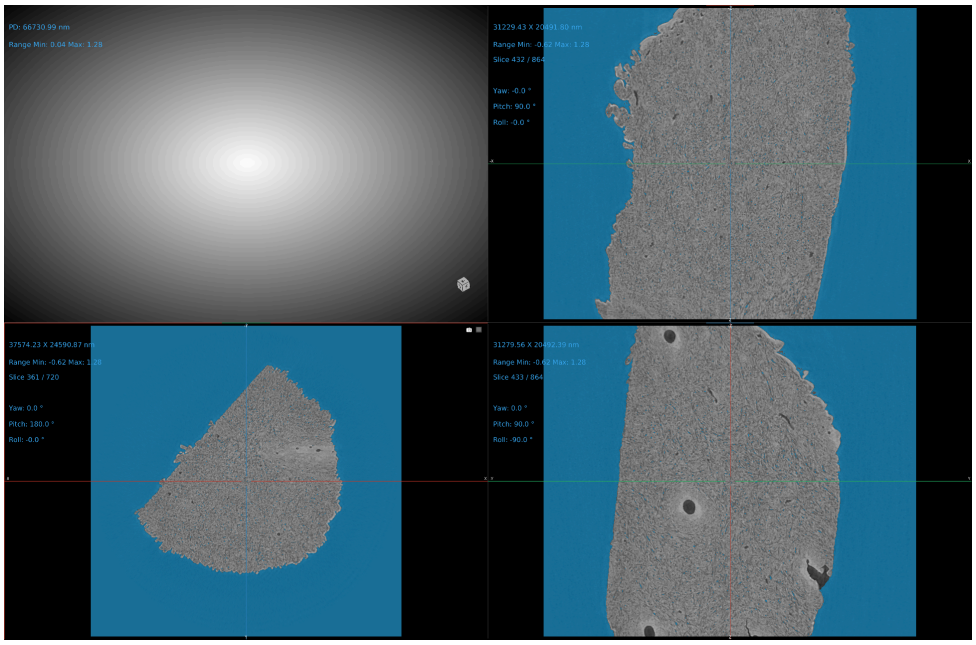

Figure 23: ROI OUTSIDE + pixels collagène

Etape 2: Extraction des zones collagènes

Les zones de collagène segmentées ne doivent pas être supprimées. Elles devront être intégrées à la ROI de collagène, que nous définirons dans la section suivante.

Pour extraire les zones de collagène, suivre les étapes ci-dessous :

└— Sélectionner la ROI **OUTSIDE (as Multi-ROI)**
  └— Faire un clic droit et choisir **Connected Components**
    └— Sélectionner l'option **New Multi-ROI (26 connected)**

Cette opération génère une nouvelle Multi-ROI comprenant :

- une ROI principale, contenant la majorité des voxels, qui représente l'extérieur de la structure,
- plusieurs petites ROIs correspondant aux zones de collagène à conserver.

Pour extraire spécifiquement la partie correspondant à l'extérieur :

└— Sélectionner la Multi-ROI **OUTSIDE**
  └— Identifier la ROI contenant le plus grand nombre de voxels,
    └— Faire un clic droit sur cette ROI,
      └— Choisir l'option **Extract class as a ROI**

La nouvelle ROI ainsi créée représente uniquement l'extérieur de la structure. NOus allons la renommer: OUTSIDE pour la suite

Cependant, la Multi-ROI OUTSIDE contient toujours les petites ROIs correspondant aux zones de collagène. Afin de les intégrer ultérieurement à la ROI de collagène, nous allons les regrouper au sein d'une seule ROI.

Pour cela:

└── Sélectionner la Multi-ROI **OUTSIDE**
  └── Supprimer la classe contenant le plus grand nombre de voxels
    ├── Faire un clic droit sur la Multi-ROI,
    └── CHoisir l'option **New ROI**

Cela nous permet d'obtenir une nouvelle ROI que l'on va appelée: zone-collagène.

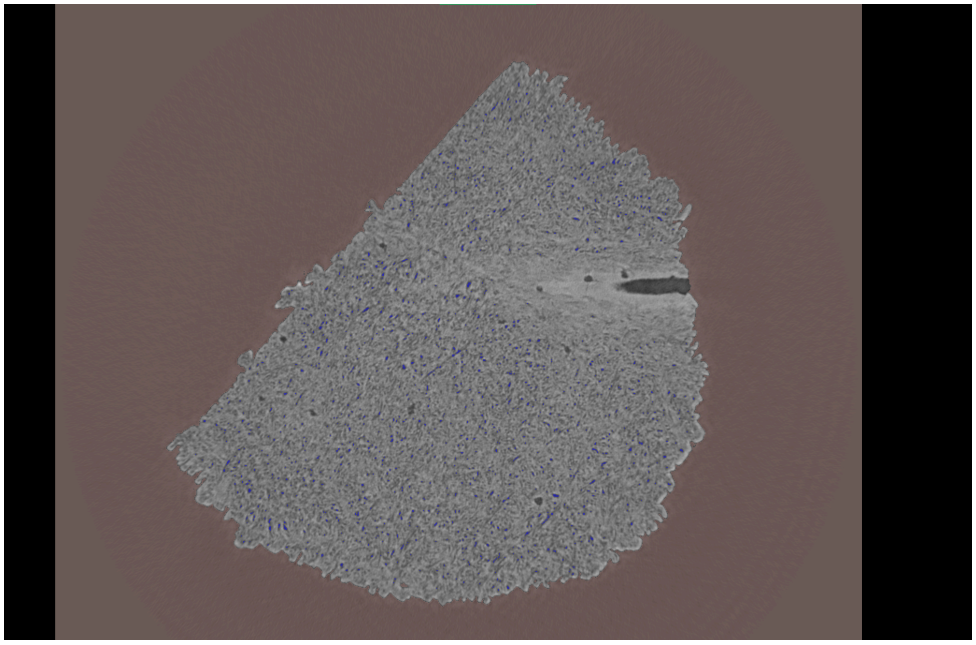

Figure 24: ROI OUTSIDE + ROI zone_collagene

§3.5. Segmentation de la matrice

La matrice extracellulaire est constituée principalement de fibres de collagène et de cristaux d'hydroxyapatite (HAP), qui jouent un rôle essentiel dans la structure et la minéralisation des tissus.

Dans cette section, nous allons segmenter les éléments suivants :

- la matrice (constituée du collagène et du HAP),
- le HAP (hydroxyapatite),
- le collagène.

§3.5.1. Importation des masques Collagène et HAP

L'importation des masques se fait en suivant la procédure décrite dans la section 3.1 – Chargement des données. Les masques à importer sont :

- masque_hap.tif
- masque_collagen.tif

Ces deux masques ont été générés à l'aide du script segmentation_kmeans.py.

Dans le cas du jeu de données Dej3, le masque de l'air est illustré ci-dessous :

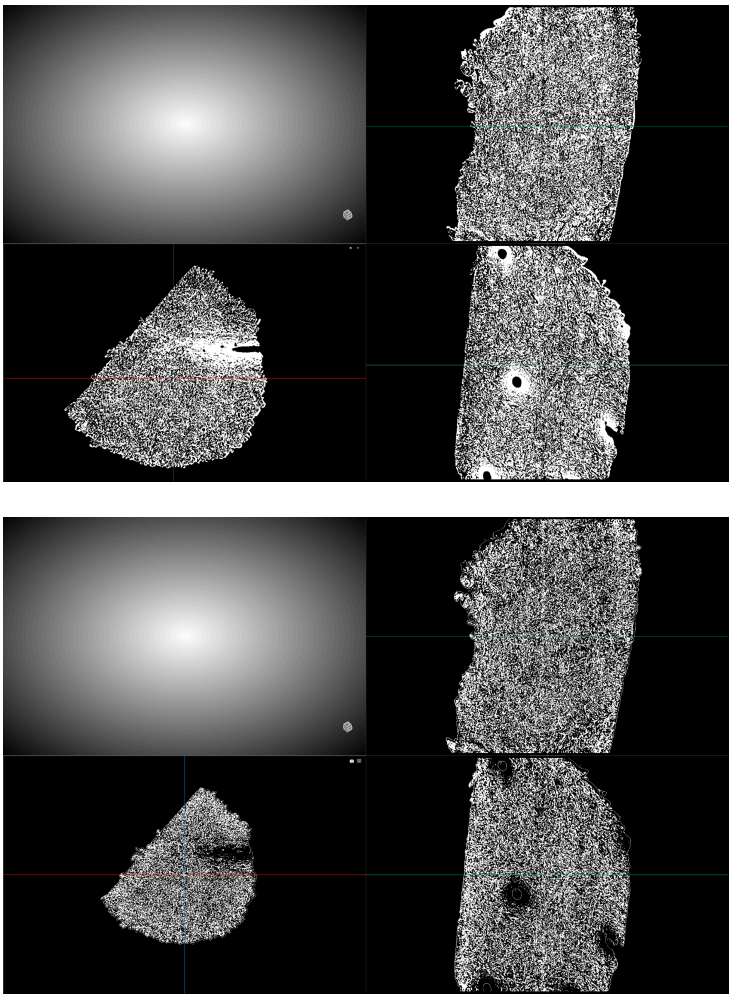

Figure 25: En haut, le masque de l'HAP et en bas le masque de collagène.

§3.5.2. Extraction des ROIs du masque HAP

Une fois le masque importé, nous procédons à l'extraction des ROIs.

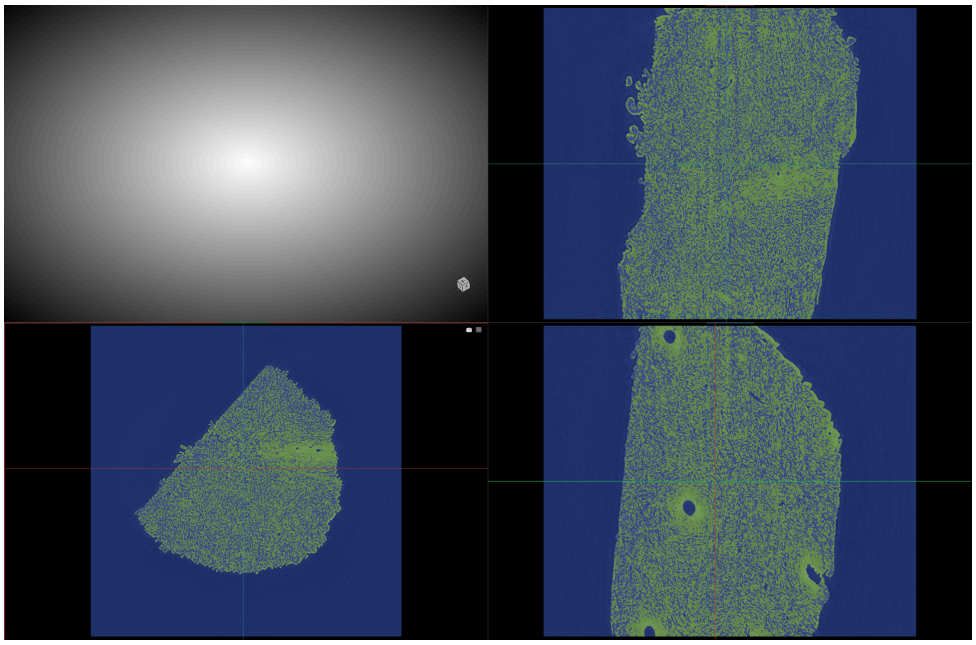

Figure 26: ROI extraite à partir du masque HAP

Ce masque permet d'extraire deux ROIs distinctes : l'une correspondant à l'hydroxyapatite (HAP), et l'autre englobant le reste du volume de données. L'intérêt se porte ici sur la ROI affichée en vert, représentant le HAP, que nous nommerons HAP.

Il est important de s'assurer que ROI-Lumière et ROI-HAP ne se chevauchent pas. Pour cela, nous allons vérifier l'existence de voxels communs entre les deux ROIs.

Procédure :

└— Sélectionner les deux ROIs : Lumière et ROI-HAP

└— Cliquer sur **Intersect**

└— Une nouvelle ROI est alors générée, représentant les voxels partagés entre les deux régions.

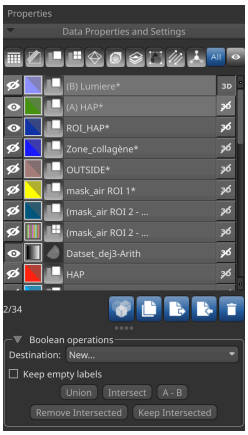

Figure 27: Procédure Instersection

Voici une visualisation de cette intersection :

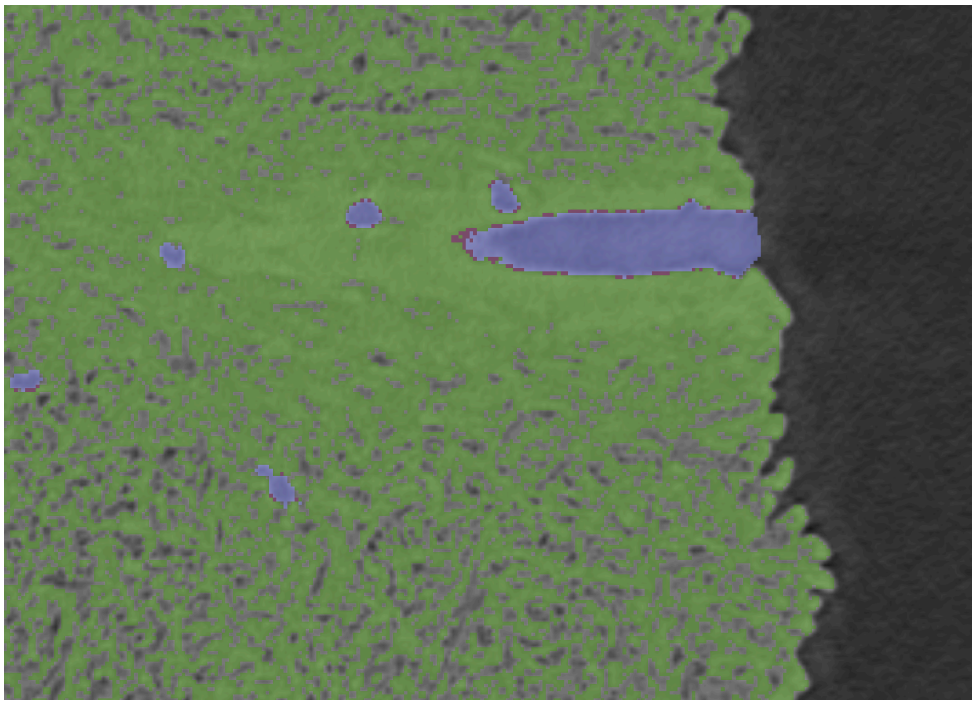

Figure 28: Voxels d'intersection visualisés en rouge

Ces voxels seront attribués à ROI-HAP. Pour cela, deux étapes sont nécessaires :

- Supprimer les voxels d'intersection de ROI-Lumière,
- Ajouter ces voxels à ROI-HAP.

§3.5.3. Extraction des ROIs du masque Collagène

Nous procédons à l'extraction des ROIs du masque collagène

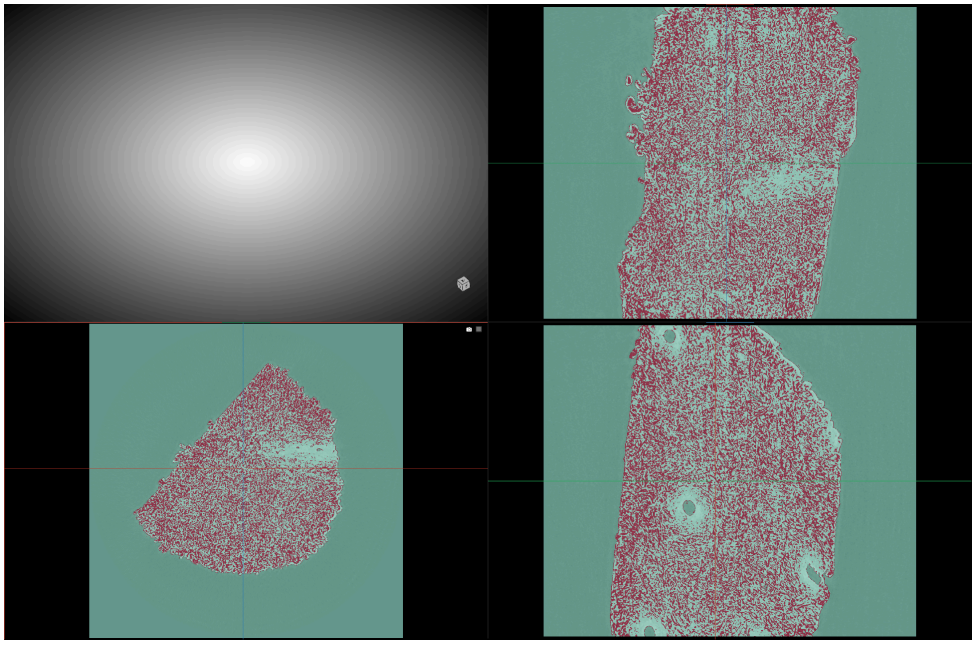

Figure 29: ROI extraite à partir du masque collagène

Ce masque permet d'extraire deux ROIs distinctes : l'une correspondant aux collagènes, et l'autre englobant le reste du volume de données. L'intérêt se porte ici sur la ROI affichée en rouge, représentant le collagène, que nous nommerons Collagene.

Il est important de s'assurer que ROI-Lumière et ROI-Collagène ne se chevauchent pas. Pour cela, nous allons vérifier l'existence de voxels communs entre les deux ROIs.

Voici une visualisation de cette intersection :

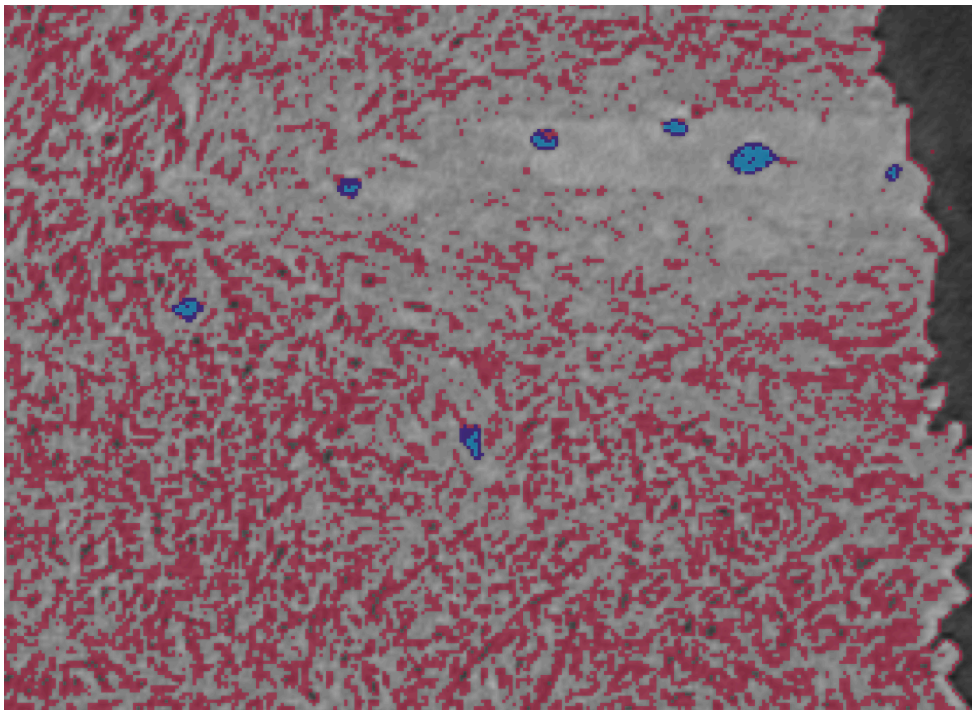

Figure 30: Voxels d'intersection visualisés en Bleu foncés

Dans ce cas, les voxels issus de l'intersection seront attribués à ROI-Lumière. Pour cela, deux étapes sont nécessaires :

- Supprimer ces voxels de ROI-Collagène,
- Les ajouter à ROI-Lumière.

Ensuite, nous fusionnerons la ROI Zone_Collagène avec la ROI-Collagène afin d'unifier l'ensemble des régions contenant du collagène.

À ce stade, la segmentation est structurée autour des quatre ROIs suivantes :

- La lumière tubulaire,
- Le collagène,
- L'hydroxyapatite (HAP),
- L'extérieur de la structure.

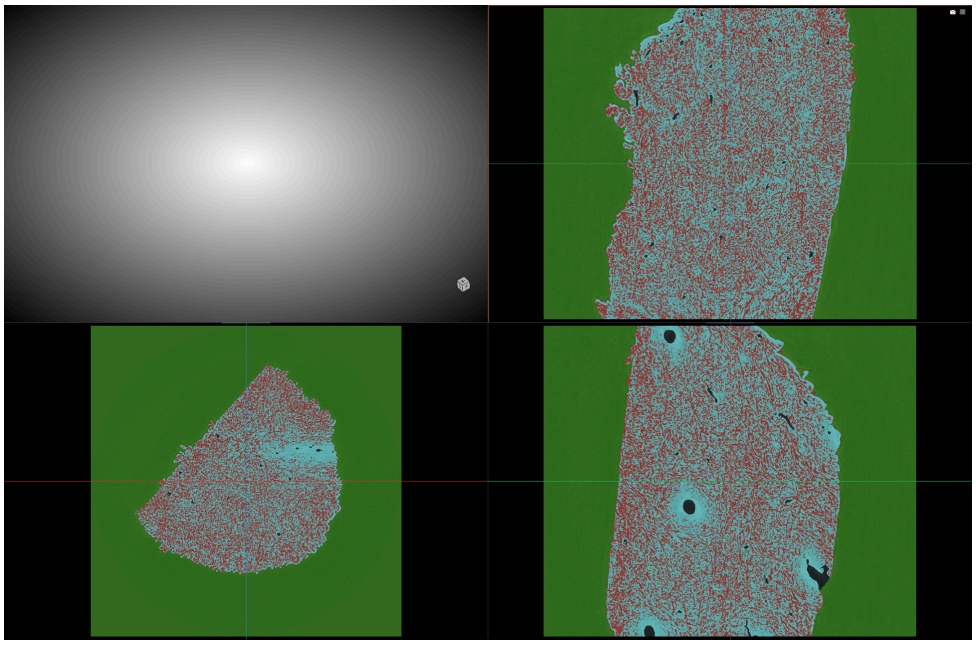

Figure 31: Extérieur (en vert), Collagène (en rouge), HAP (en bleu), lumiere (en noir)

Ces données sont sauvegardées dans une session nommée: 20250604_SEGMENTATION_4CLASSES.ORSSession