Rapport traitement d'image biomédicale
Imagerie rétinienne

Réalisé par : ICHOU Maryem
Encadré Par : Pr. SERRAR Ouafae

Année universitaire : 2020/2021

Rapport traitement d'image II
Imagerie rétinienne

•••

PLAN

•••

I. INTRODUCTION • • •

L'imagerie médicale évolue sans cesse grâce aux contributions de domaines scientifiques comme la physique, la chimie, les mathématiques, les sciences de l'ingénieur et la médecine.

La compréhension du traitement d'images commence par la compréhension de ce qu'est une image. Le mode et les conditions d'acquisition et de numérisation des images traitées conditionnent largement les opérations qu'il faudra réaliser pour extraire de l'information.

Le but de l'imagerie médicale est de créer une représentation visuelle intelligible d'une information à caractère médical. Cette problématique s'inscrit plus globalement dans le cadre de l'image scientifique et technique : l'objectif est en effet de pouvoir représenter sous un format relativement simple une grande quantité d'informations issues d'une multitude de mesures acquises selon un mode bien défini.

L'image obtenue peut être traitée informatiquement pour obtenir par exemple : la reconstruction tridimensionnelle d'un organe ou d'un tissu, un film ou une animation montrant l'évolution ou les mouvements d'un organe au cours du temps, une imagerie quantitative qui représente les valeurs mesurées pour certains paramètres biologiques dans un volume donné.

Dans un sens plus large, le domaine de l'imagerie médicale englobe toutes les techniques permettant de stocker et de manipuler ces informations. Ainsi, il existe une norme pour la gestion informatique des données issues de l'imagerie médicale : la norme ***DICOM***.

•••

II. Standart DICOM• • •

Digital imaging and communications in medicine, couramment abrégée DICOM, est un standard pour la gestion informatique des données issues de l'imagerie médicale.

Historique du standard dicom

Il a été créé en 1985 par l'ACR (American College of Radiology) et la NEMA [1] (National Electric Manufacturers Association) dans le but de standardiser les données transmises entre les différents appareils de radiologie.

L'ACR-NEMA standards Publication No. 300-1985, publiée en 1985, a été appelée version 1.0. Deux révisions de la norme ont suivi : No. 1, datée d'octobre 1986 et No. 2, datée de janvier 1988. Ces publications de norme spécifiaient une interface matérielle, un ensemble minimal de commandes logicielles et un jeu cohérent de formats de données.

L'ACR-NEMA standards Publication No. 300-1988, publiée en 1988, a été appelée version 2.0. Elle comprenait la version 1.0, les révisions publiées ainsi que d'autres révisions. Elle incluait également de nouveaux documents permettant la prise en charge de commandes pour les dispositifs d'affichage, l'introduction d'un nouveau schéma hiérarchique pour identifier une image et l'ajout d'éléments de données pour une description plus spécifique de l'image.

En 1993, l'ACR-NEMA/Standard 300 a été révisée en profondeur et remplacée par le présent document appelé Imagerie numérique et communication en médecine (DICOM). Il contient un certain nombre d'améliorations majeures par rapport aux versions précédentes de la norme ACR-NEMA, dont voici la liste.

- Il s'applique à un environnement en réseau. La norme ACR-NEMA n'était auparavant applicable que dans un environnement point à point. Pour un environnement en réseau, une unité d'interface réseau était requise. La norme DICOM est compatible avec un fonctionnement dans un environnement en réseau avec le protocole réseau TCP/IP, conforme aux normes du secteur.

- Il s'applique à l'échange de supports hors ligne. La norme ACR-NEMA ne spécifiait pas de format de fichier ni ne proposait de choix de supports physiques ou un système de fichiers logique. La norme DICOM prend en charge le fonctionnement dans un environnement de supports hors ligne à l'aide de supports tels que CD-R, DVD-R et USB et de systèmes de fichiers courants conformes aux normes du secteur.

- Il s'agit d'un protocole orienté service, qui spécifie la sémantique des commandes et des données associées et qui indique comment les dispositifs censés être conformes à la norme DICOM réagissent aux commandes et aux données échangées. Les services spécifiés comprennent la gestion du déroulement des opérations d'un service d'imagerie. La norme ACR-NEMA se limitait au transfert des données avec des exigences de service implicites uniquement.

- Il précise des niveaux de conformité. La norme ACR-NEMA spécifiait un niveau de conformité minimal. DICOM décrit explicitement comment un exécuteur doit structurer une déclaration de conformité pour sélectionner des options spécifiques.

En 1995, avec l'ajout des fonctionnalités DICOM pour l'imagerie en cardiologie prises en charge par l'ACR, le comité mixte ACR-NEMA a été réorganisé en tant que comité des normes DICOM, qui représente une large collaboration de parties prenantes intervenant dans toutes les spécialités de l'imagerie médicale.

Principes

- ***Applicabilité globale et localisation***

DICOM est une norme de portée mondiale pouvant être utilisée dans toutes les régions. Elle propose des mécanismes de traitement des données compatibles avec les exigences culturelles : différents systèmes d'écriture, jeux de caractères, langues et structures pour les adresses et noms de personne. Elle prend en charge la diversité des flux de travail, processus et politiques utilisés en imagerie médicale dans différentes régions du monde, spécialités médicales et pratiques locales.

La localisation permettant de respecter les exigences des politiques nationales ou locales en matière de santé et de flux de travail est possible sans écart par rapport à la norme DICOM. Elle peut inclure la spécification de jeux de code (par ex., codes de procédure) ou le profilage d'utilisation des éléments de données (les deux s'attachant à préciser les valeurs autorisées localement et à rendre obligatoires des éléments facultatifs de la norme DICOM pour un usage local).

- **Mise à jour continue**

La norme DICOM est une norme qui évolue et qui est mise à jour conformément aux modes opératoires du comité des normes DICOM. Les propositions d'amélioration de la part de tous les utilisateurs de la norme DICOM sont les bienvenues et peuvent être soumises au Secrétariat. Les ajouts et corrections de la norme DICOM sont soumis à un vote et à approbation plusieurs fois par an. Une fois le texte final approuvé, chaque modification devient officielle, fait l'objet d'une publication séparée et entre en vigueur sur-le-champ. Périodiquement, toutes les modifications approuvées du texte final sont consolidées et publiées dans une édition mise à jour de la norme DICOM. Une fois les modifications consolidées dans une édition mise à jour de la norme DICOM, les documents relatifs à chaque modification ne sont pas actualisés. Les lecteurs sont invités à utiliser l'édition consolidée de la norme DICOM.

La mise à jour de la norme DICOM exige de préserver la compatibilité avec les éditions précédentes. Le processus de mise à jour peut impliquer le retrait de certaines sections de la norme DICOM.

Le retrait ne signifie pas que ces caractéristiques ne peuvent pas être utilisées. Cependant, le comité des normes DICOM ne tiendra pas à jour la documentation relative aux fonctions retirées. Le lecteur est renvoyé aux précédentes éditions de la norme DICOM.

•••

- **Objets d'information et identification d'objet unique**

De nombreux services DICOM impliquent l'échange d'objets d'information persistants, tels que les images. Toute instance d'un tel objet d'information peut être échangée entre de nombreux systèmes et dans de nombreux contextes organisationnels au fil du temps. Tandis que des modifications mineures peuvent être apportées aux attributs d'une instance afin de faciliter son traitement au sein d'une organisation particulière (par exemple, en forçant un ID patient sur la valeur utilisée dans un contexte local), le contenu sémantique d'une instance ne change pas.

Chaque instance est identifiée par un identifiant d'objet unique à l'échelle mondiale que l'instance conserve dans tous les échanges. Les modifications apportées au contenu sémantique d'une instance sont définies pour créer une nouvelle instance, qui se voit attribuer un nouvel identifiant d'objet unique à l'échelle globale.

- **Conformité**

La conformité à la norme DICOM s'énonce en termes de classes de parité service/objet (SOP), qui représentent les services (tels que le stockage en réseau, sur support ou Web) opérant sur les types d'objets d'information (tels que la tomographie assistée par ordinateur ou les IRM).

Les spécifications des classes SOP de la norme DICOM ne sont modifiées que dans le but d'être compatibles avec toutes les éditions existantes et à venir de la norme DICOM. Les exigences de conformité et les déclarations de conformité font par conséquent référence à l'identifiant de la classe SOP et jamais à une édition de la norme DICOM.

Chaque implémentation doit fournir une déclaration de conformité, respectant une structure pro forma cohérente, facilitant la comparaison de produits à des fins d'interopérabilité.

- **Cohérence du modèle d'information**

Un grand nombre d'objets d'information définis dans la norme DICOM suivent un modèle commun d'informations composites avec des entités d'information représentant le patient, l'étude, la série, l'équipement, le cadre de référence et le type de données d'instance spécifique. Ce modèle d'information schématise les concepts et activités de l'imagerie médicale du monde réel; pour les modalités d'acquisition, une étude est quasiment équivalente à une procédure ordonnée, et une série est quasiment équivalente à un élément de protocole d'acquisition de données effectué. Dans d'autres domaines, tels que la radiothérapie, l'étude et la série sont moins clairement liées aux entités ou aux activités du monde réel, mais elles sont nécessaires à des fins de cohérence. Ce modèle simplifié suffit pour les besoins pragmatiques de la gestion des données des images et des données associées collectées dans le cadre de la pratique courante.

Les nouveaux objets d'information définis dans la norme DICOM seront généralement conformes à ce modèle d'information courant existant. Ils permettront la réutilisation d'implémentations avec un minimum de changement pour la prise en charge de nouveaux objets.

•••

Objectifs et avantages

Un Service de Radiologie produit plusieurs milliers d'images chaque jour, ainsi, un scanner, travaillant au rythme de 3 patients par heure produit environ 150 images par heures. Il n'est pas possible de classer ces images dans un format courant de type JPEG ou GIF car il aurait un risque de pertes des données démographiques de l'image, (nom du patient, type d'examen, hôpital, date d'examen, type d'acquisition etc...). Le format DICOM permet de rendre unique chaque image produite et de leur associer des informations spécifiques. Ainsi chaque image est autonome, si elle est perdue, reproduite ou renommée, il est toujours possible d'identifier formellement son origine, le patient, la date, la série d'où elle provient, les paramètres d'acquisition etc...

L'objectif du standard DICOM est donc de faciliter les transferts d'images entre les machines de différents constructeurs. En effet, avant la généralisation de ce format, chaque constructeur de matériel d'imagerie utilisait un format de données propriétaire, entrainant d'importants problèmes de gestion et de maintenance (incompatibilités, coût, perte d'information) dans les établissements de santé.

Les images au format DICOM accompagnant les dossiers médicaux sont lisibles sur tout matériel informatique compatible, et rendent obsolète le transport des clichés par les moyens de communication traditionnels, principalement les envois par courrier.

Le format DICOM est indépendant des machines et des protocoles de communication : La norme DICOM est standardisée au niveau "applicatif" c'est à dire elle permet la communication d'un programme à un autre, ceci sous-entends que les connections de bas niveau, câblages et protocoles réseaux soient établies.

Étant donné que les images basées sur DICOM sont de haute qualité et que plusieurs images d'un seul patient nécessitent beaucoup d'espace de stockage, des dispositions spéciales doivent être prises pour stocker et récupérer les images au format DICOM. La base de données et le système serveur qui stocke les images DICOM est appelé PACS (Picture Archiving and communication System). En général, chaque hôpital dispose de son propre serveur PACS interne, et les images acquises auprès des patients de cet hôpital seul y sont stockées. L'inconvénient de cela est que les patients qui changent d'hôpital pour diverses raisons peuvent ne pas être en mesure d'accéder aux images passées.

L'introduction du PACS basé sur le cloud a facilité l'affichage et l'accès aux fichiers DICOM. La technologie Cloud permet de stocker et de traiter les fichiers DICOM via Internet. Ces fichiers sont accessibles depuis n'importe où, à l'aide de n'importe quel périphérique disposant des autorisations et des logiciels requis. Il simplifie l'accès aux dossiers médicaux d'un patient à partir de différents endroits géographiques.

Traitement d'image sous Python

But : extraire les différentes données, visualiser les images en 2D, et effectuer les différentes opérations possibles sur les images.

III. Extraction des différentes données

Ce travail a été réalisé avec python.

Pourquoi Python :

Python est un langage de programmation très puissant en traitement d'image.
Les tâches courantes du traitement d'images incluent l'affichage des images, les manipulations basiques comme le recadrage, le retournement, la rotation, ou encore la segmentation, la classification et les extractions de caractéristiques, la restauration et la reconnaissance d'images, Python devient un choix judicieux pour de telles tâches de traitement d'images. Cela est dû à sa popularité croissante en tant que langage de programmation scientifique et à la disponibilité gratuite de nombreux outils de traitement d'images de pointe dans son écosystème.

Les principales boites à outils pour faire du traitement d'images en python sont OpenCV, scikit-image et Pillow. Les librairies scientifiques Python plus générales que sont Numpy et Scipy fournissent aussi certains outils de traitement des images. Toutes ces librairies peuvent dialoguer facilement grâce à l'utilisation commune de tableaux Numpy pour stocker les images. Une image en Python en niveau de gris est généralement stockée dans une tableau Numpy bidimensionnel à valeur entières ou réelles avec H lignes et W colonnes (W=width,H=height).

Pour le jeu de données :

Un échantillon humain a été obtenu à partir d'un patient qui a subi une énucléation de l'œil en raison d'un mélanome uvéal. La moitié de l'échantillon a subi un examen histopathologique conventionnel comprenant la fixation dans du formol tamponné au phosphate à 4%, l'inclusion de paraffine, la coloration à l'hématoxyline-éosine. Des tranches minces obtenues à partir du bloc de paraffine ont ensuite été imagées par microscopie à lumière transmise et scannées en utilisant un scanner de lames histologiques Aperio2 (Leica, Allemagne). La moitié restante du segment antérieur de l'œil, deux échantillons de la rétine et du cristallin ont été sélectionnés lors de l'examen macroscopique dans le service de pathologie et ensuite placés dans du formol tamponné au phosphate à 4% pendant 48 heures.

Le dossier « ***retina*** » comporte 950 images ‘’.dcm’’.

•••

Les packages utilisées

```
# Packages
import os
import scipy
import matplotlib as mpl
import skimage
fromskimageimport exposure
import cv2
import numpy as np
import pydicom as dicom
fromPIL import Image
import matplotlib.pyplot as plt
import pathlib
from pathlib import Path
import shapely
```

Lire le fichier dicom

```
## Lire l'image I0012.dcm
dataset=dicom.read_file("D:\\1.Master BIO-MSCS\\Traitement d'image
II\\rapport\\retina\\I0012.dcm")
```

Impression de toutes les métadonnées :

Après avoir lire le fichier DICOM, On a affiché toutes les informations (les métadonnées) en utilisant la commande suivante :

```
print(dataset)
```

Résultat :

```
C:\Users\PC\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/PC/PycharmProjects/pythonProject1/dicom.py
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 216
(0002, 0001) File Meta Information Version       OB: b'\x00\x01'
(0002, 0002) Media Storage SOP Class UID         UI: CT Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 1.2.826.0.1.3680043.8.435.517520846.60951.0012
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
(0002, 0012) Implementation Class UID            UI: 1.2.826.0.1.3680043.2.1143.107.104.103.115.2.4.3
(0002, 0013) Implementation Version Name         SH: 'GDCM 2.4.3'
(0002, 0016) Source Application Entity Title     AE: 'GDCM'
-------------------------------------------------
(0008, 0005) Specific Character Set              CS: 'ISO_IR 100'
(0008, 0008) Image Type                          CS: ['ORIGINAL', 'PRIMARY ', 'AXIAL']
(0008, 0016) SOP Class UID                       UI: CT Image Storage
(0008, 0018) SOP Instance UID                    UI: 1.2.826.0.1.3680043.8.435.517520846.60951.0012
(0008, 0020) Study Date                          DA: '20190823'
(0008, 0021) Series Date                         DA: '20190823'
(0008, 0022) Acquisition Date                    DA: '20190823'
(0008, 0030) Study Time                          TM: '091956.076000'
(0008, 0031) Series Time                         TM: '091956.000076'
(0008, 0032) Acquisition Time                    TM: '091956.000076'
(0008, 0033) Content Time                        TM: '091956.000076'
```

•••

```
(0008, 0033) Content Time                       TM: '091956.000076'
(0008, 0050) Accession Number                   SH: ''
(0008, 0060) Modality                           CS: 'CT'
(0008, 0070) Manufacturer                       LO: 'Carl Zeiss X-ray Microscopy, Inc.'
(0008, 0080) Institution Name                   LO: ''
(0008, 0090) Referring Physician's Name         PN: ''
(0008, 1030) Study Description                  LO: 'Xradia'
(0008, 103e) Series Description                 LO: 'Xradia'
(0010, 0010) Patient's Name                     PN: 'D:.Experiments.eye_human_retina.eye_human_retina_2019-08-22_192101.4X_60kV_5W_LE1_0.5s_2.5um_
(0010, 0020) Patient ID                         LO: 'Synthetic Dataset'
(0010, 0030) Patient's Birth Date               DA: ''
(0010, 0040) Patient's Sex                      CS: 'O'
(0010, 1010) Patient's Age                      AS: ''
(0018, 0050) Slice Thickness                    DS: "0.00250128602981567"
(0018, 0060) KVP                                DS: "60.0"
(0018, 5100) Patient Position                   CS: 'HFS'
(0020, 000d) Study Instance UID                 UI: 1.2.826.0.1.3680043.8.435.517520846.60951
(0020, 000e) Series Instance UID                UI: 1.2.826.0.1.3680043.8.435.517520846.60951.0
(0020, 0010) Study ID                           SH: ''
(0020, 0011) Series Number                      IS: "1"
(0020, 0012) Acquisition Number                 IS: "0"
(0020, 0013) Instance Number                    IS: "12"
```

```
(0020, 0013) Instance Number                    IS: "12"
(0020, 0032) Image Position (Patient)           DS: [0, 0, 0.0300154323577881]
(0020, 0037) Image Orientation (Patient)        DS: [1, 0, 0, 0, 1, 0]
(0020, 0052) Frame of Reference UID             UI: 6.6.6.6.6.6.6.6.1
(0020, 1040) Position Reference Indicator       LO: ''
(0020, 1041) Slice Location                     DS: "0.0300154332071543"
(0020, 4000) Image Comments                     LT: ''
(0028, 0002) Samples per Pixel                  US: 1
(0028, 0004) Photometric Interpretation         CS: 'MONOCHROME2'
(0028, 0010) Rows                               US: 1015
(0028, 0011) Columns                            US: 980
(0028, 0030) Pixel Spacing                      DS: [0.00250128602981567, 0.00250128602981567]
(0028, 0100) Bits Allocated                     US: 16
(0028, 0101) Bits Stored                        US: 16
(0028, 0102) High Bit                           US: 15
(0028, 0103) Pixel Representation               US: 0
(0028, 1050) Window Center                      DS: "7000.0"
(0028, 1051) Window Width                       DS: "60000.0"
(0028, 1052) Rescale Intercept                  DS: "0.0"
(0028, 1053) Rescale Slope                      DS: "1.0"
(0028, 1054) Rescale Type                       LO: 'US'
(0028, 1055) Window Center & Width Explanation  LO: 'WINDOW1'
(7fe0, 0010) Pixel Data                         OW: Array of 1989400 elements
```

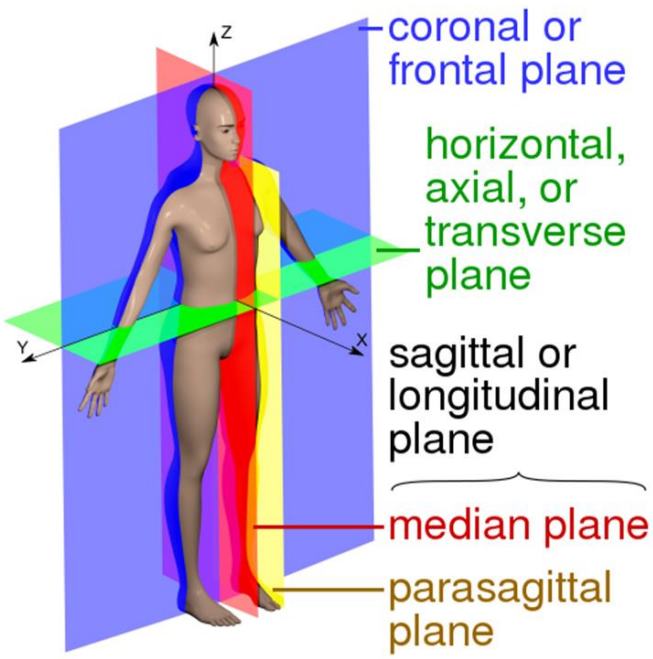
On a pu noter que chaque champ a un code individuel (0000, 0000), appelé étiquette. Les quatre premiers chiffres désignent le groupe du champ (par exemple 0010 = Patient) et les deuxièmes désignent «l'élément». Les numéros individuels des 4 balises numérotées sont hexadécimaux donc au lieu de prendre des valeurs de 0 à 9 pour signifier 0 à 9, ils peuvent s'étendre au-delà pour prendre des valeurs jusqu'à 15 en incluant les 6 premières lettres de l'alphabet.

•••

Les différentes données

- ✓ **Voici quelques points clés sur les informations de balise ci-dessus :**
  - ***Pixel Data (7fe0,0010)***(dernière entrée) - C'est là que les données de pixel brutes sont stockées. L'ordre des pixels codés pour chaque plan d'image est de gauche à droite, de haut en bas, c'est-à-dire que le pixel supérieur gauche (étiqueté 1,1) est codé en premier.
  - ***PhotometricInterpretation*** (0028, 0004) : il s'agit de MONOCHROME2 où les données de pixels sont représentées comme un seul plan d'image monochrome où la valeur d'échantillonnage minimale est destinée à être affichée sous forme d'informations noires.
  - ***Samples per Pixel***(0028, 0002) - Cela devrait être 1 car cette image est monochrome. Cette valeur serait de 3 si l'espace colorimétrique était RGB par exemple.
  - ***Bits Stored*** (0028 0101) - Nombre de bits stockés pour chaque échantillon de pixel.
  - ***Pixel Represenation***(0028 0103) - peut être non signée (0) ou signée (1).
  - ***Rows***Représente le nombre des lignes = 1015.
  - ***Columns*** – Représente le nombre de colonnes = 980.
  - ***Patient ID*** : SyntheticDataset
  - ***PatientName***: D:.Experiments.eye_human_retina.eye_human_retina_2019-08-22_192101.4X_60kV_5W_LE1_0.5s_2.5um_2401p.eye_human_retina_4X_60k…
  - ***PatientSex***: '0'
  - ***PatientBirthDath***: ' '
  - ***StudyID***: ' '
    Certaines informations sont anonymisées (comme le nom et l'identifiant), ce qui est normal pour les données médicales publiques.
  - ***StudyDate***: '20190823'
  - ***StudyTime***: '091956.076000'
  - ***Manufacturer*** : 'Carl Zeiss X-ray Microscopy, Inc.'

- ***Image type*** : ['ORIGINAL', 'PRIMARY ', **'AXIAL'**] ➔ coupe Axial



Source: David Richfield and Mikael Häggström, M.D. and cmgleeCC BY-SA 4.0

- ***Modality*** de l'image : 'CT' : (**scanner**)
  Imagerie à rayons X
  • Taille de l'image512 x 512.
  • Obtention de sériesd'images.
  • Reconstructionvolumique (3D) àpartir des coupes.

  Le scanner est une technique d'imagerie à visée diagnostique qui consiste à « balayer » une région du corps donnée afin de créer des images en coupe, grâce à l'utilisation d'un faisceau de rayons X. Le terme « scanner » est en réalité le nom de l'appareil médical, mais il est communément utilisé pour nommer l'examen. On parle aussi de tomodensitométrie ou encore de scanographie.

- ***…***

La plupart du temps, nous ne nous intéressons qu'à quelques-uns de ces domaines. Plutôt que de jeter toute la liste, nous pouvons choisir ceux que nous voulons.
Comme on ne se souvient pas toujours du nom exact d'un élément de données, la bibliothèque pydicom fournit une méthode pratique dir (), utile pendant les sessions interactives à l'invite python. En utilisant cette commande, nous pouvons trouver rapidement les noms de tous les éléments de données contenant le mot «patient» ainsi :

```
Print(dataset . dir ( "patient" ))
```

•••

Résultat :

['ImageOrientationPatient', 'ImagePositionPatient', 'PatientAge', 'PatientBirthDate', 'PatientID', 'PatientName', 'PatientPosition', 'PatientSex']

Nous pouvons maintenant examiner n'importe lequel de ces éléments un par un, comme indiqué ci-dessous :

```
print(dataset.PatientName) # show patient name
print(dataset.PatientSex) # show patient gender
# ect…
```

IV. Caractéristiques des images

Parmi les caractéristiques de l'image on a : sa taille en points ou pixels, ses dimensions réelles et sa résolution …

✓ **Pour l'affichage de la dimension : (Rows, columns)**

```
#  Dimension : (Rows,columns)
import shapely
print(dataset.pixel_array.shape)
```

Résultat :

Dimension de l'image =

```
(1015, 980)
```

✓ **Taille de l'image**

Pour calculer la taille d'une image numérique, il suffit de multiplier le nombre de pixels sur la hauteur par le nombre de pixels sur la largeur de l'image.

En utilisant python :

La méthode lit un fichier dicom et retourne numpy.array() des valeurs de pixels .Elle sert a transformer l'image en matrice de pixel. L'entrée de cette méthode est un chemin vers un fichier appelé "filename".

```
defdicom_to_array(filename):
    d = dicom.read_file(filename)
    a = d.pixel_array
return np.array(a)

a1 = dicom_to_array("D:\\1.Master BIO-MSCS\\Traitement d'image
II\\rapport\\retina\\I0012.dcm")
print(a1.size)
>>> 994700
```

Résultat

Size =

```
994700
```

✓ **Min et max des pixels**

```
## Use numpy.ndarray.min() and
#numpy.ndarray.max() to find the smallest and largest values in the array a1
print( np.ndarray.min(a1) )
print( np.ndarray.max(a1) )
```

Résultat : **Min = 0**

Max = 17779

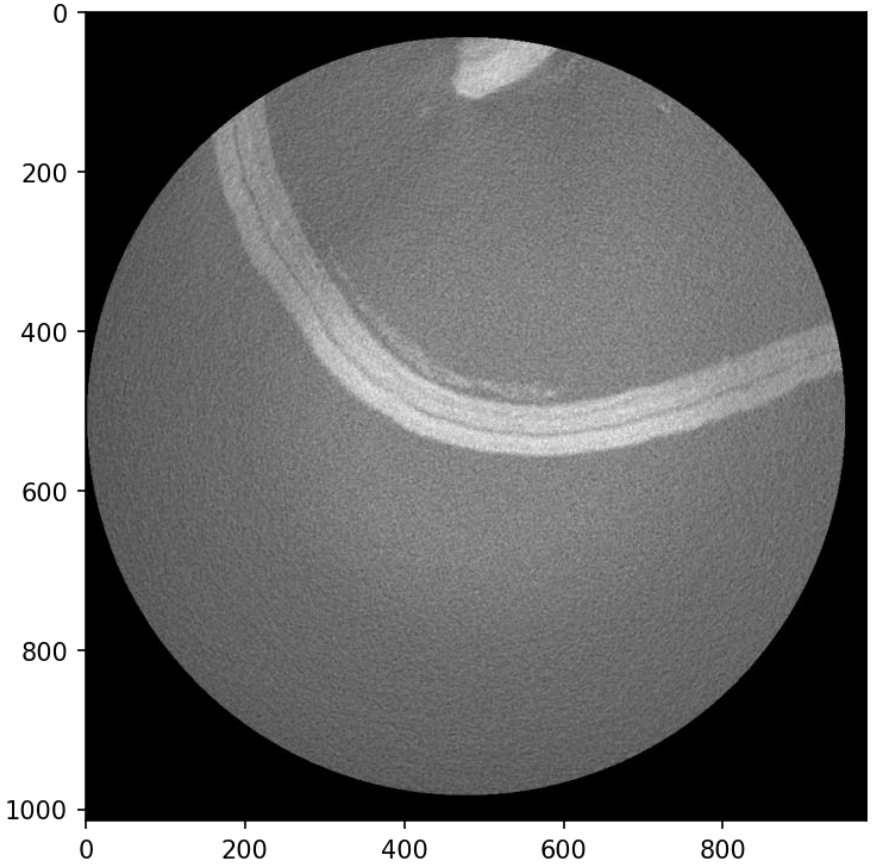
•••

✓ Résolution

La résolution spatiale est une mesure de la finesse des détails d'une image pour une dimension donnée. Cette résolution spatiale, qui indique la « densité de pixels », est couramment simplement nommée « résolution ». Le ***DPI*** est une unité de mesure de la résolution, utilisée principalement pour les écrans et les imprimantes.

- *Dpi = 150*

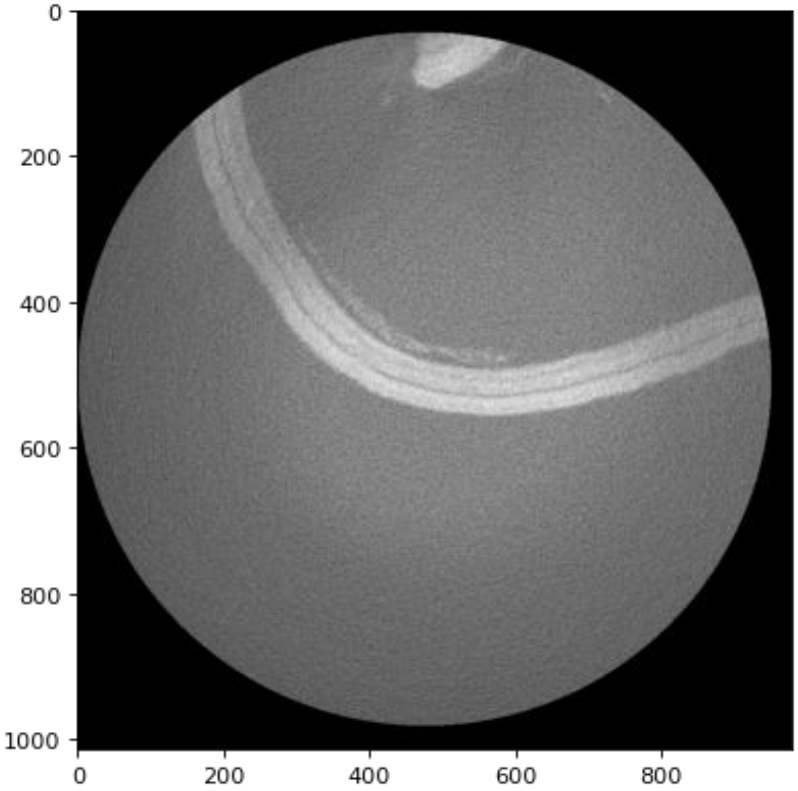
```
# tell matplotlib to display our images as a 6 x 6 inch image, with resolution of
150 dpi
plt.figure(figsize= (6,6), dpi=150)

# tell matplotlib to display our image, using a gray-scalelookup table.
plt.imshow(a1, 'gray')
```



- *Dpi = 80*

```
# tell matplotlib to display our images as a 6 x 6 inch image, withresolution of
80 dpi
plt.figure(figsize= (6,6), dpi=80)
plt.imshow(a1, 'gray')
```

•••



- ***On remarque bien qu'en changeons la résolution, la qualité de l'image change.***

✓ **La matrice des pixels**

Une image numérique en niveaux de gris est un tableau de valeurs. Chaque case de ce tableau, qui stocke une valeur, se nomme un pixel. En notant n le nombre de lignes et p le nombre de colonnes de l'image, on manipule ainsi un tableau de n×p pixels.

```
print(dataset.PixelData [: 200])
print(dataset.pixel_array, dataset.pixel_array.shape)
```

Résultat :

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]] (1015, 980)
```

On obtient que des zéros. Il est possible que le début et la fin du tableau soient vraiment des zéros. Il est généralement plus significatif de regarder quelque chose au milieu de l'image. Par exemple :

```
#Il est possible que le début et la fin du tableau soient vraiment des zéros.
#Il est généralement plus significatif de regarder quelque chose au milieu de
l'image.
midrow = dataset.Rows // 2
```

•••

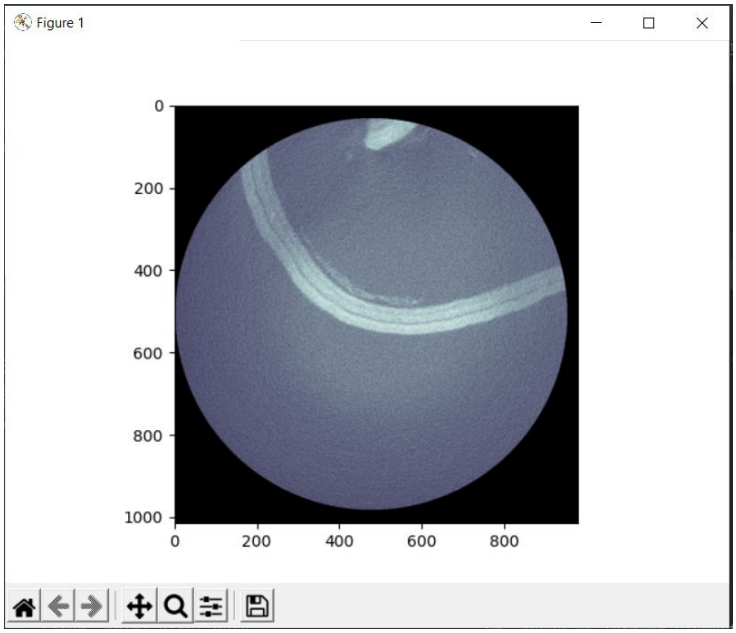
```
midcol = dataset.Columns // 2
print(dataset.pixel_array[midrow-20:midrow+20, midcol-20:midcol+20])
```

```
[[13737 12908 13609 ... 10870 11992 11732]
 [14072 13463 13896 ... 11708 12389 12533]
 [15176 15651 15332 ... 12405 12802 12527]
 ...
 [15166 15220 15542 ... 11610 11725 12703]
 [15620 14880 14603 ... 10634 10245 11394]
 [15293 14458 14374 ... 11021 11185 11502]]
```

V. visualisation de l'image en 2D

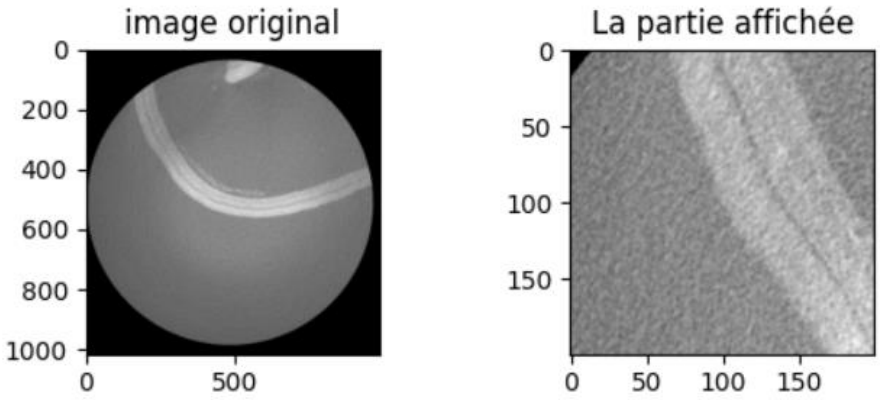
Affichage d'une l'image DICOM

```
plt.imshow(dataset.pixel_array,plt.cm.bone)
plt.show()
```



Affichage d'une partie de l'image

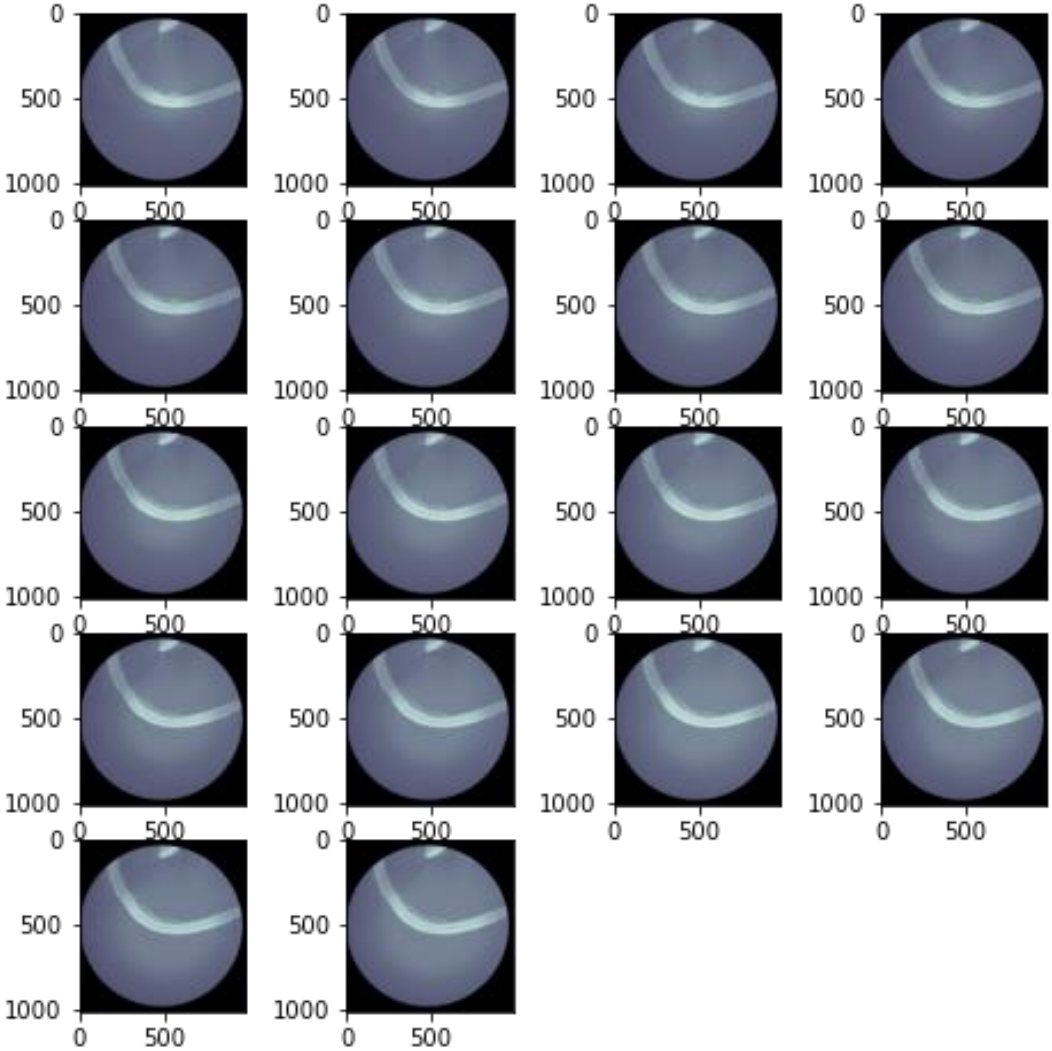
```
# Affichage d'une partie de l'image :
part=a1[200:400,100:300]
plt.subplot(2,2,1),plt.imshow(a1,cmap = 'gray')
plt.title('image')
plt.subplot(2,2,2),plt.imshow(part,cmap= 'gray')
plt.title('la partie affiché ')

plt.show()
```



•••

Affichage d'une collection d'image DICOM

```
### Affichage de 18 images
import matplotlib.pyplotas plt
import pydicom
folder = 'D:\\1.Master BIO-MSCS\\Traitement d\'image II\\rapport\\retiina\\'
fig=plt.figure(figsize=(8, 8))
columns=4
rows=5

# plot 18 images
for i in range(1,19):
    x= folder + str(i) + '.dcm'
ds = pydicom.filereader.dcmread(x)
fig.add_subplot(rows,columns,i)
# load image pixels
plt.imshow(ds.pixel_array, cmap=plt.cm.bone)
plt.show()
```

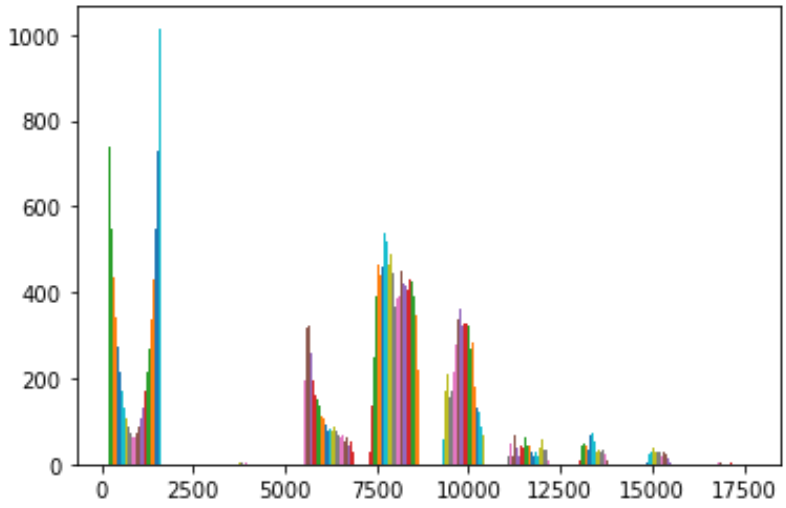


VI. Histogramme

En imagerie numérique, l'histogramme représente la distribution des intensités (ou descouleurs) de l'image. C'est un outil fondamental du traitement d'images, avec de trèsnombreuses applications. Les histogrammes sont aussi très utilisés en photographie et pour la retouche d'images.

Histogramme d'une image

```
hist,bins = np.histogram(a1, bins=256)
plt.figure()
plt.hist(a1)
```

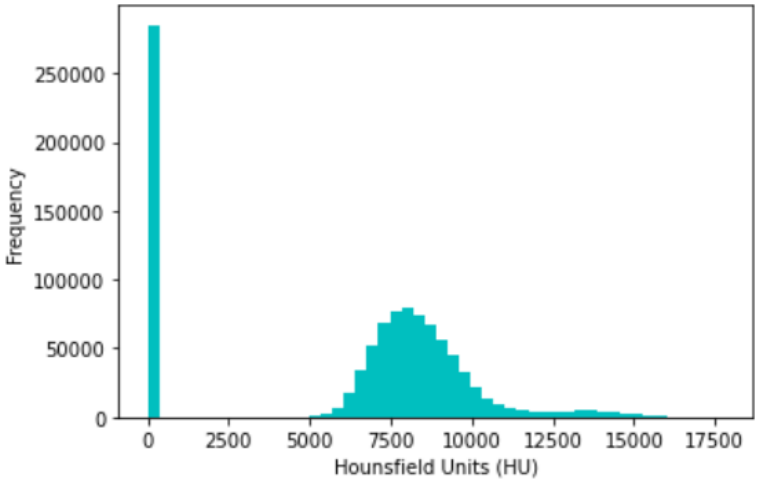


Histogram of Hounsfield Units of CT mouse image

Les HU sont utiles car ils sont normalisés pour tous les scans CT quel que soit le nombre absolu de photons capturés par le détecteur du scanner.

```
## Histogram of Hounsfield Units of CT mouse image
plt.hist(a1.flatten(), bins=50, color='c')
plt.xlabel("Hounsfield Units (HU)")
plt.ylabel("Frequency")
plt.show()
```

•••



Critiquer l'histogramme

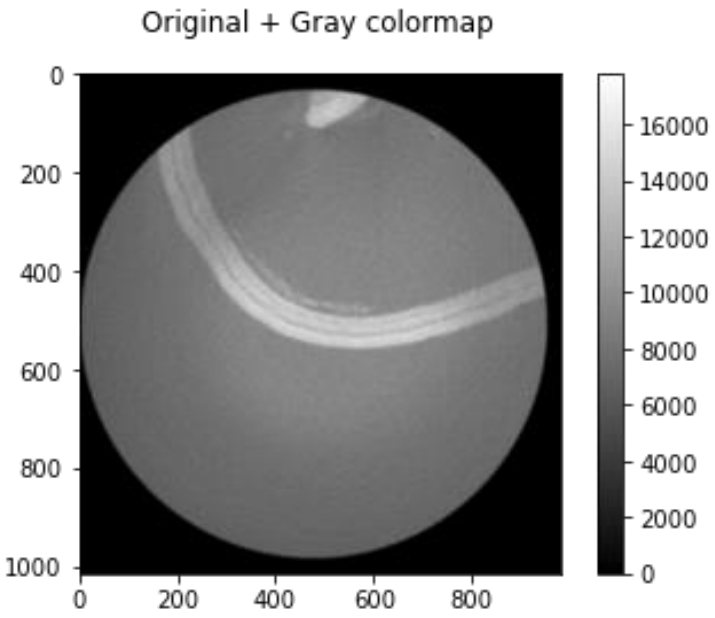
L'histogramme suggère ce qui suit :
Il y a beaucoup d'eau (0).

Nous allons tracer l'image originale et son égalisation d'histogramme avec la palette de couleurs grise.

Image aux niveaux de gris

Niveaux de gris est une gamme de monochromatique nuances du noir au blanc. Par conséquent, une image en niveaux de gris ne contient que des nuances de gris et aucune couleur.

```
# grayscale
fig1  =plt.figure()
plt.imshow(a1, cmap="gray", interpolation="bicubic")
plt.colorbar()
fig1.suptitle("Original + Gray colormap", fontsize=12)
```

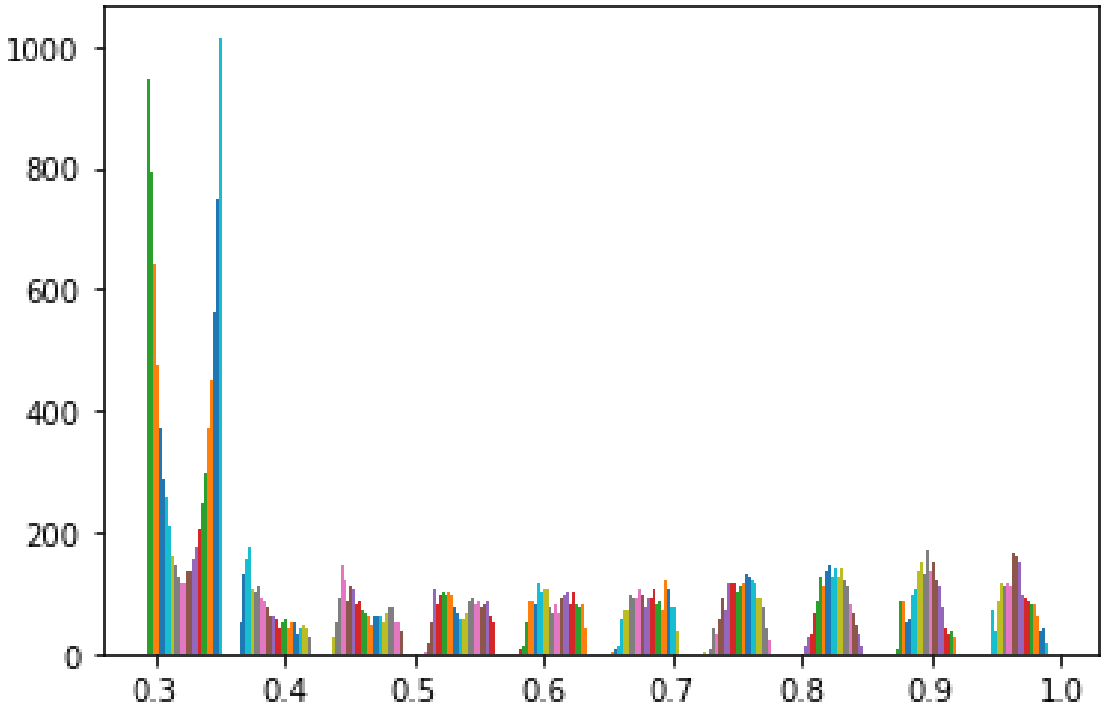


•••

Améliorer le contrast :

Le contraste est une propriété intrinsèque d'une image qui quantifie la différence de luminosité entre les parties claires et sombres d'une image.

```
#Améliorer contrast
a1_eq = exposure.equalize_hist(a1)
hist_eq,bins_eq = np.histogram(a1_eq, bins=256)
plt.figure()
plt.hist(a1_eq)
```

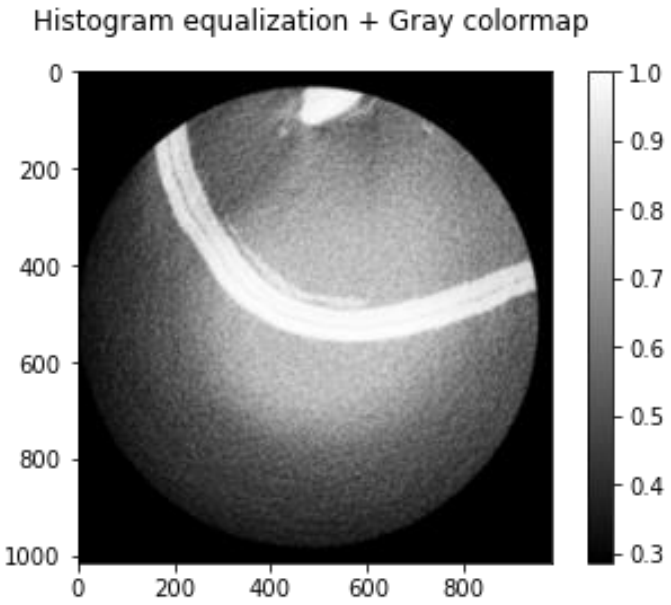


•••

Histogram equalization + Gray colormap

En traitement d'images, l'égalisation d'histogramme est une méthode d'ajustement du contraste d'une image numérique qui utilise l'histogramme. Elle consiste à appliquer une transformation sur chaque pixel de l'image, et donc d'obtenir une nouvelle image à partir d'une opération indépendante sur chacun des pixels. Cette transformation est construite à partir de l'histogramme cumulé de l'image de départ.

```
## IMAGE APRES EGALISATION DE L'HISTOGRAMME
fig2 = plt.figure()
## a1-eq=a1_eq = exposure.equalize_hist(a1)
plt.imshow(a1_eq, cmap="gray", interpolation="bicubic")
plt.colorbar()
fig2.suptitle("Histogramequalization + Gray colormap", fontsize=12)
```

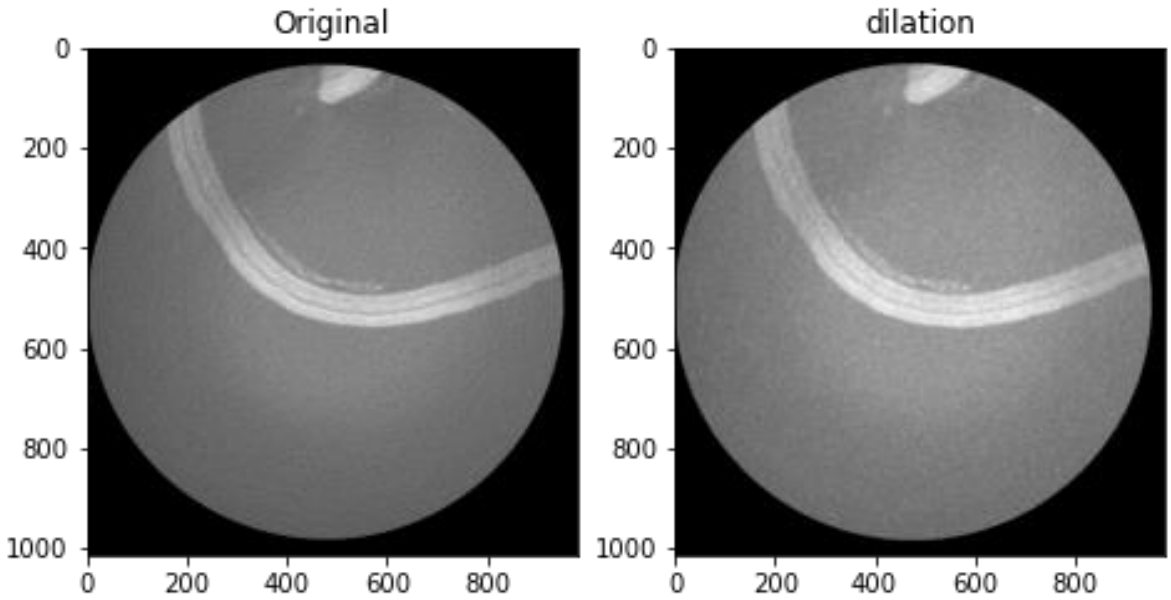


VII. Transformations morphologiques et filtres

Dilatation

Une dilatation morphologique consiste à déplacer l'élément structurant sur chaque pixel de l'image, et à regarder si l'élément structurant « touche » (ou plus formellement intersecte) la structure d'intérêt. Le résultat est une structure qui plus grosse que la structure d'origine (Figure ci-dessous). En fonction de la taille de l'élément structurant, certaines particules peuvent se trouver connectées, et certains trous disparaître. La dilatation agrandit la taille des objets dans l'image.

```
## Dilatation : C'est juste l'opposé de l'érosion.
dilation = cv2.dilate(a1,kernel,iterations = 1)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(dilation, 'gray')
plt.title('dilation')
plt.show()
```

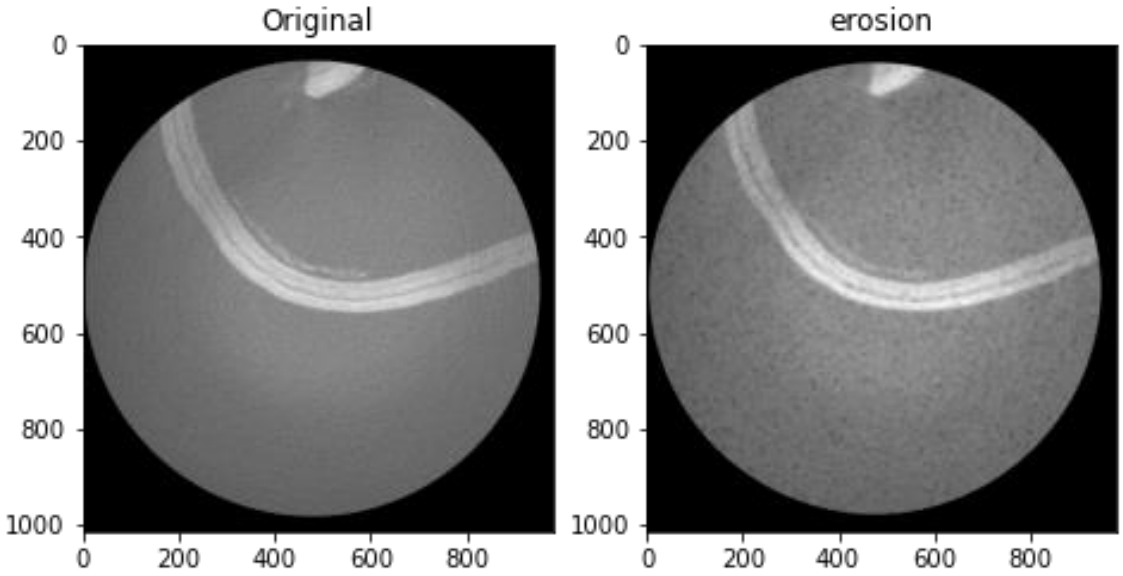


•••

Erosion

L'érosion est l'opération inverse, qui est définie comme une dilatation du complémentaire de la structure. Elle consiste à chercher tous les pixels pour lesquels l'élément structurant centré sur ce pixel touche l'extérieur de la structure. Le résultat est une structure rognée. On observe la disparition des particules plus petites que l'élément structurant utilisé, et la séparation éventuelle des grosses particules.

```
## Package opencv
import cv2
import numpyas np
a1 = dicom_to_array("D:\\1.Master BIO-MSCS\\Traitement d'image
II\\rapport\\retina\\I0012.dcm")

kernel = np.ones((5,5),np.uint8)
erosion = cv2.erode(a1,kernel,iterations = 2)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(erosion, 'gray')
plt.title('Result')
plt.show()
```



On peut remarquer qu'en niveaux de gris, l'érosion est équivalente à l'application d'un filtre minimum, tandis que la dilatation est équivalente à l'application d'un filtre maximum.

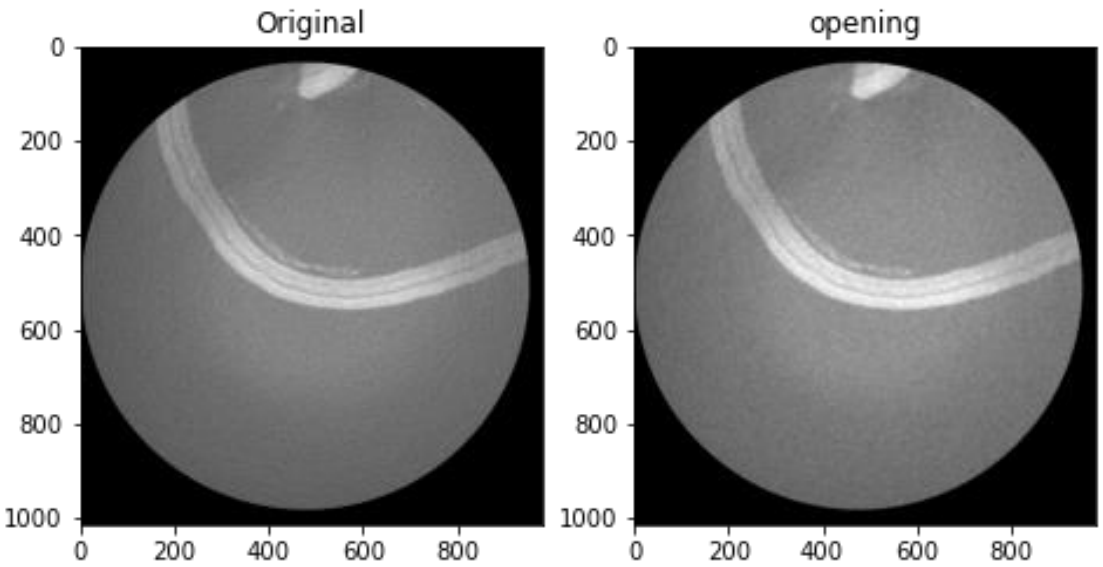
Ouverture morphologique

L'érosion et la dilatation ont l'inconvénient de modifier fortement la taille des structures dans l'image. Pour réduire cet effet, on les utilise souvent en combinaison.

On définit l'ouverture morphologique comme une érosion suivie d'une dilatation. L'ouverture a pour effets :

- Faire disparaître les petites particules (dont la taille est inférieure à celle de l'élément structurant)
- Séparer les grosses particules aux endroits où elles sont plus fines
- Il est utile pour supprimer le bruit.

```
## Ouverture :L'ouverture n'est qu'un autre nom d' érosion suivi de dilatation .
Il est utile pour supprimer le bruit
opening = cv2.morphologyEx(a1, cv2.MORPH_OPEN, kernel)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(opening, 'gray')
plt.title('opening')
plt.show()
```



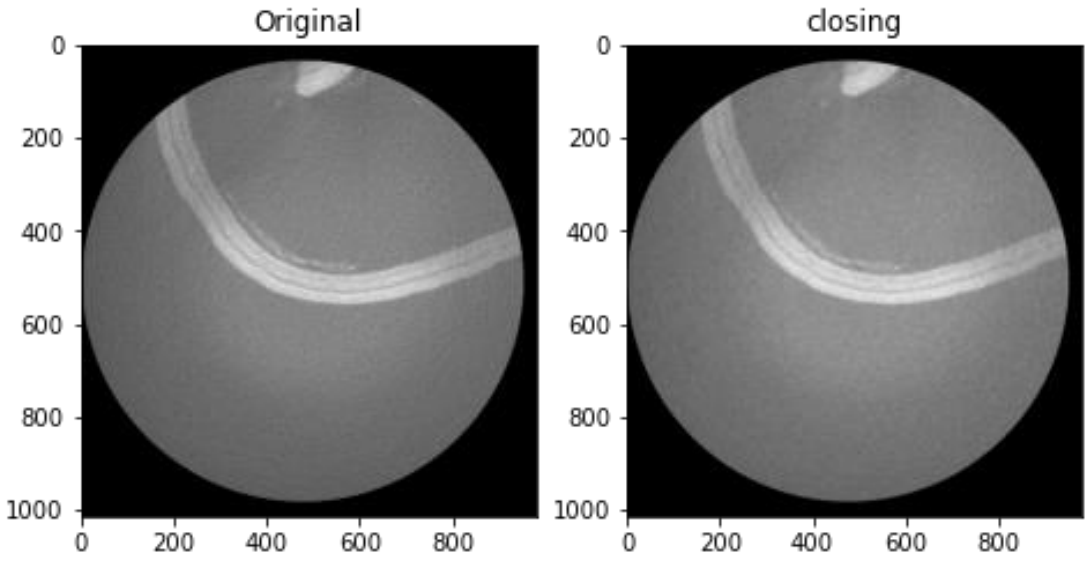
•••

Fermeture morphologique

On définit la fermeture morphologique comme une dilatation suivie d'une érosion. La fermeture a pour effets :

- Faire disparaître les trous de petite taille dans les structures
- Connecter les structures proches

```
## La fermeture est l'inverse de l'ouverture, la dilatation suivie de l'érosion
closing = cv2.morphologyEx(a1, cv2.MORPH_CLOSE, kernel)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(closing, 'gray')
plt.title('closing')
plt.show()
```



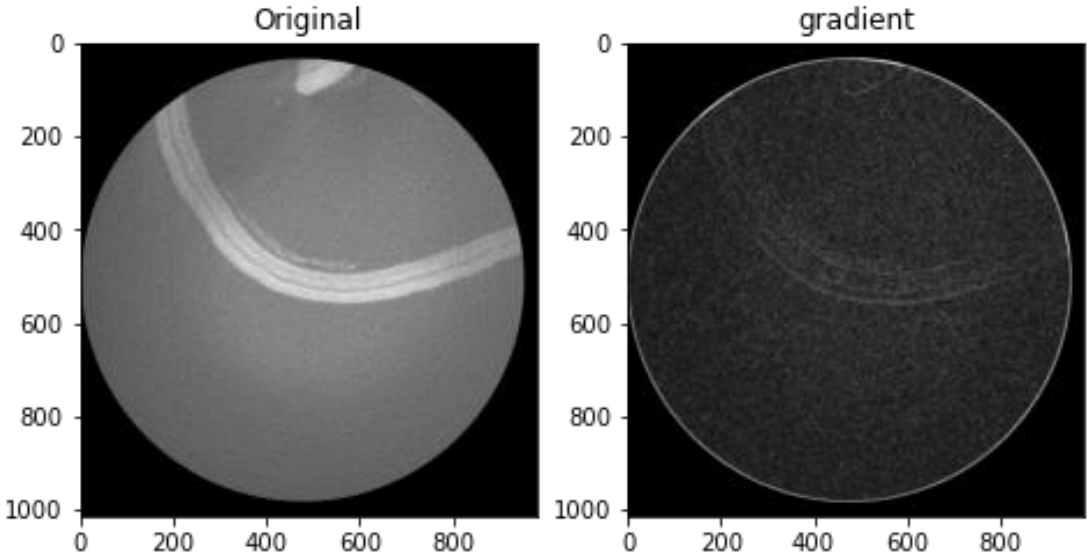
*L'ouverture et la fermeture morphologique ont une **propriété d'idempotence** : le résultat ne change pas si on applique plusieurs fois l'opérateur, il suffit de l'appliquer une seule fois.*
L'ouverture et la fermeture morphologique changent relativement peu la forme des grosses particules. Par contre, elles permettent de faire disparaitre facilement les petites particules isolées, ou les petits trous à l'intérieur des structures. On les utilise donc souvent pour nettoyer le résultat d'une binarisation.

Gradient morphologique

Une hypothèse courante dans l'analyse d'images consiste à considérer les objets de l'image comme des régions homogènes en niveaux de gris. Cela induit que sur la frontière entre deux objets, il existe une variation du niveau de gris. Les opérateurs du gradient visent à mettre en évidence ces variations. C'est la différence entre la dilatation et l'érosion d'une image. Le résultat ressemblera au contour de l'objet.

- La technique du gradient est moins sensible au bruit mais de complexité plus importante

```
## Gradient morphologique : C'est la différence entre la dilatation et l'érosion
d'une image.
gradient = cv2.morphologyEx(a1, cv2.MORPH_GRADIENT, kernel)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(gradient, 'gray')
plt.title('gradient')
plt.show()## Le résultat ressemblera au contour de l'objet.
```

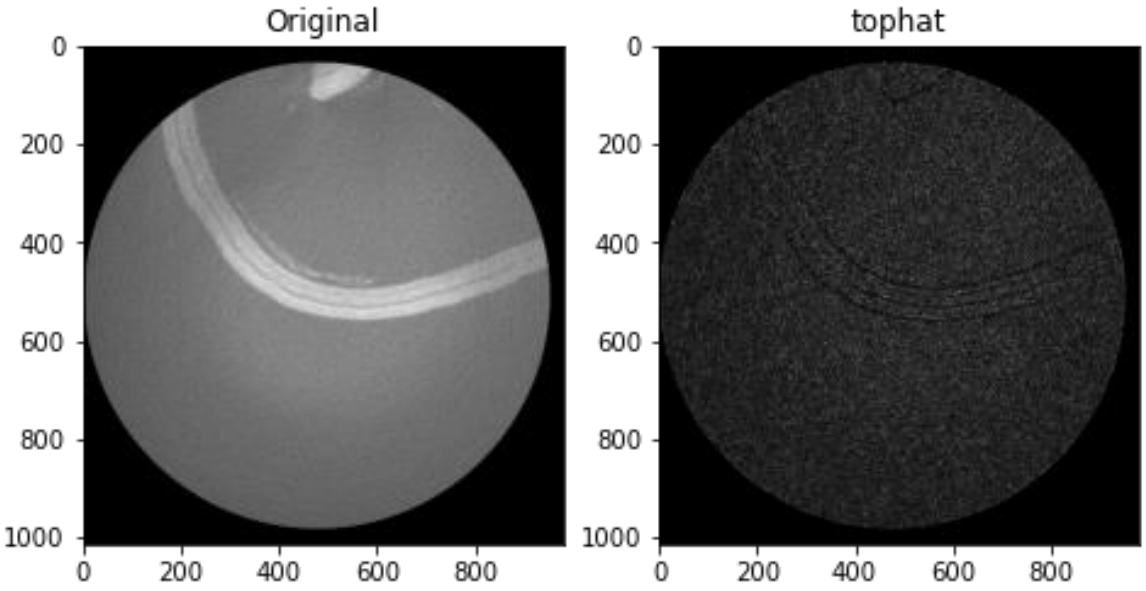


•••

Tophat

Tophat est une opération qui extrait de petits éléments et des détails d'images données. C'est la différence entre l'image d'entrée et l'ouverture de l'image.

```
## tophat : C'est la différence entre l'image d'entrée et l'ouverture de l'image.
tophat = cv2.morphologyEx(a1, cv2.MORPH_TOPHAT, kernel)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(tophat, 'gray')
plt.title('tophat')
plt.show()
```

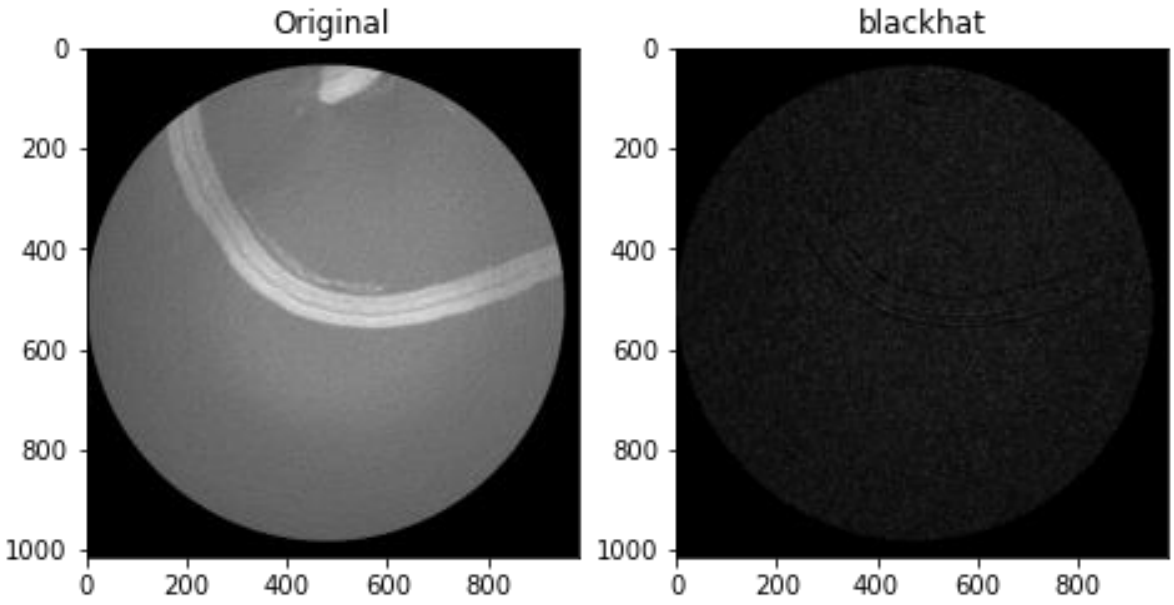


•••

Blackhat

C'est la différence entre la fermeture de l'image d'entrée et de l'image d'entrée.

```
##la différence entre la fermeture de l'image d'entrée et de l'image d'entrée.
blackhat = cv2.morphologyEx(a1, cv2.MORPH_BLACKHAT, kernel)
plt.figure(figsize=(12, 12))
plt.subplot(131)
plt.imshow(a1, 'gray')
plt.title('Original')
plt.subplot(132)
plt.imshow(blackhat, 'gray')
plt.title('blackhat')
plt.show()
```

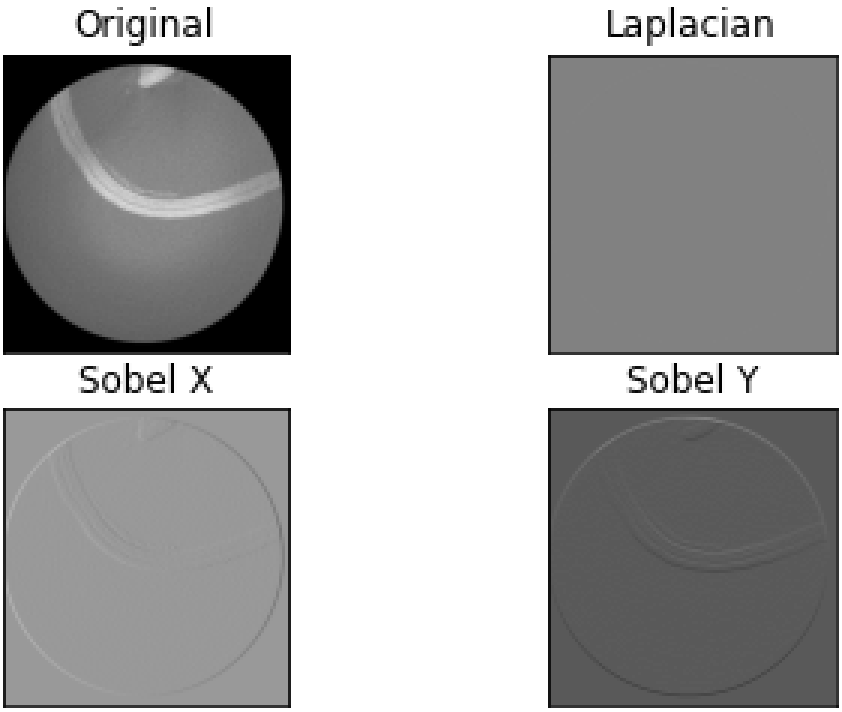


•••

Dérivés laplaciens

Le laplacien est un opérateur différentiel égal à la somme de toutes les deuxièmes dérivées partielles non mixtes d'une variable dépendante.

```
#LaplacianDerivatives : dérivés laplaciens
import cv2
import numpyas np
frommatplotlibimport pyplotas plt


laplacian = cv2.Laplacian(a1,cv2.CV_64F)
sobelx = cv2.Sobel(a1,cv2.CV_64F,1,0,ksize=5)
sobely = cv2.Sobel(a1,cv2.CV_64F,0,1,ksize=5)

plt.subplot(2,2,1),plt.imshow(a1,cmap = 'gray')
plt.title('Original'), plt.xticks([]), plt.yticks([])
plt.subplot(2,2,2),plt.imshow(laplacian,cmap= 'gray')
plt.title('Laplacian'), plt.xticks([]), plt.yticks([])
plt.subplot(2,2,3),plt.imshow(sobelx,cmap= 'gray')
plt.title('Sobel X'), plt.xticks([]), plt.yticks([])
plt.subplot(2,2,4),plt.imshow(sobely,cmap= 'gray')
plt.title('Sobel Y'), plt.xticks([]), plt.yticks([])

plt.show()
```



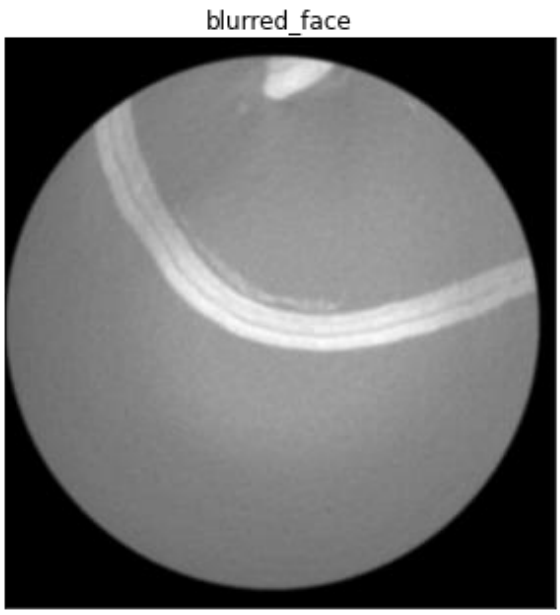
•••

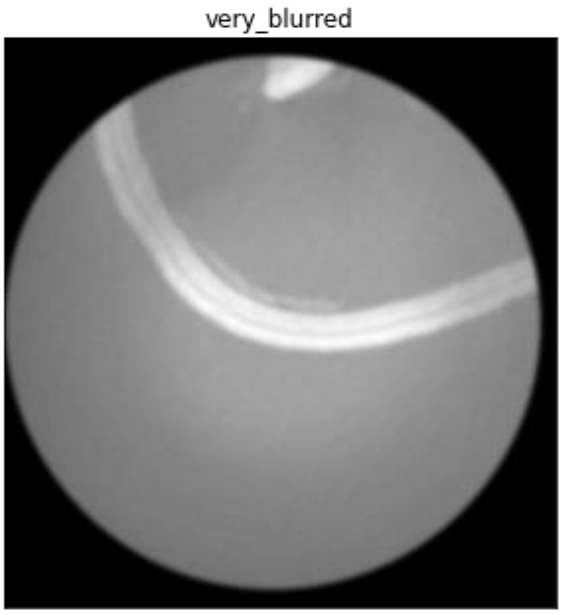
Filtre gaussien

Ce type de filtre est couramment utilisé dans les logiciels internes des appareils photo numériques, ou dans les logiciels de traitement d'image.

Les applications abordées sont le filtre de lissage pour atténuer le bruit, le filtre dérivateur pour détecter les bords, et la technique d'accentuation de netteté par masque flou.

```
blurred_face = ndimage.gaussian_filter(a1, sigma=3)
very_blurred = ndimage.gaussian_filter(a1, sigma=5)
plt.figure(figsize=(11, 6))


plt.subplot(121), plt.imshow(blurred_face, cmap='gray')
plt.title('blurred_face'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(very_blurred, cmap='gray')
plt.title('very_blurred'), plt.xticks([]), plt.yticks([])

plt.show()
```



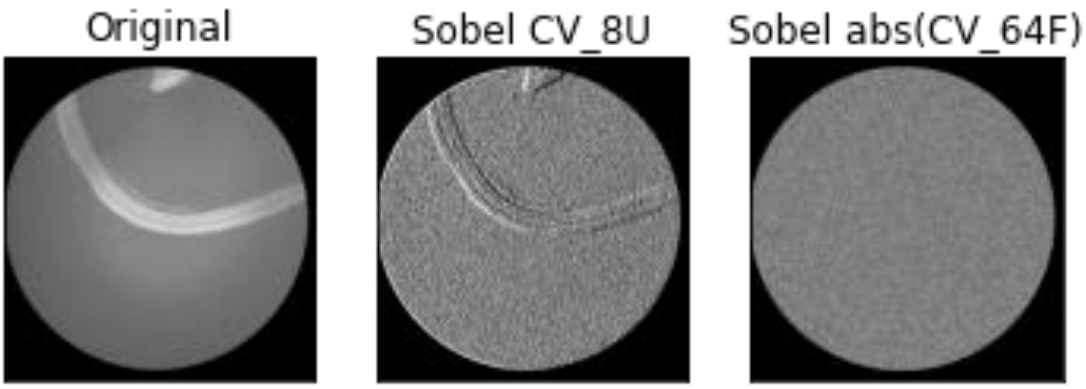


•••

Filtre Sobel

Le filtre de Sobel est un opérateur utilisé en traitement d'image pour la détection de contours. Il s'agit d'un des opérateurs les plus simples qui donne toutefois des résultats corrects.

```
## filtre Sobel horizontal et la différence des résultats
import cv2
import numpyas np
frommatplotlibimport pyplotas plt


# Output dtype = cv2.CV_8U
sobelx8u = cv2.Sobel(a1,cv2.CV_8U,1,0,ksize=5)

# Output dtype = cv2.CV_64F. Thentakeitsabsolute and convert to cv2.CV_8U
sobelx64f = cv2.Sobel(a1,cv2.CV_64F,1,0,ksize=5)
abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)

plt.subplot(1,3,1),plt.imshow(a1,cmap = 'gray')
plt.title('Original'), plt.xticks([]), plt.yticks([])
plt.subplot(1,3,2),plt.imshow(sobelx8u,cmap = 'gray')
plt.title('Sobel CV_8U'), plt.xticks([]), plt.yticks([])
plt.subplot(1,3,3),plt.imshow(sobel_8u,cmap = 'gray')
plt.title('Sobel abs(CV_64F)'), plt.xticks([]), plt.yticks([])

plt.show()
```

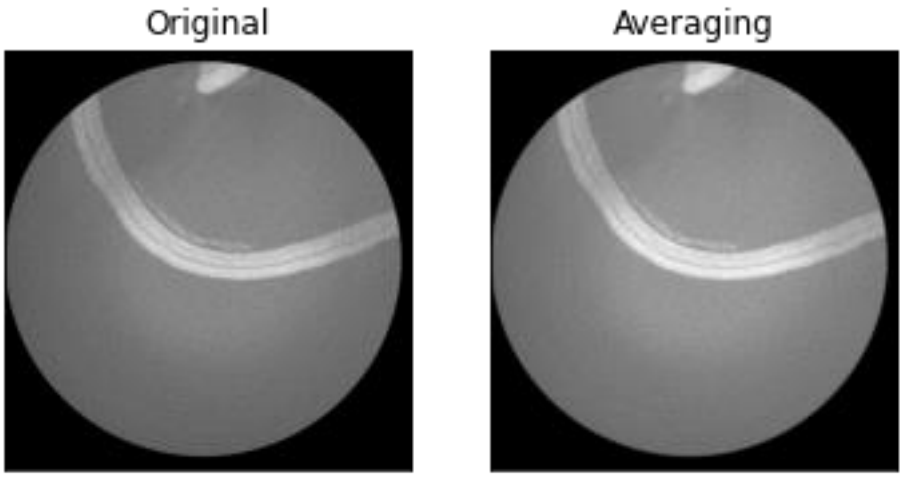


•••

Convolution 2D (filtrage d'image)

La convolution est souvent utilisée pour le lissage, l'accentuation et la détection des contours. OpenCV fournit une fonction, cv2.filter2D( ), pour convoluer un noyau avec une image.

```
## convolution
kernel = np.ones((5,5),np.float32)/25
dst = cv2.filter2D(a1,-1,kernel)
plt.subplot(121),plt.imshow(a1,'gray'),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(dst,'gray'),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
plt.show()
```



VIII. Conclusion

L'imagerie médicale est un élément essentiel à la recherche clinique, l'étude des maladies et la mise au point de nouveaux traitements. Il existe de nombreuses techniques d'imagerie complémentaires. L'imagerie recouvre à une grande variété de technologies développées grâce à l'exploitation des grandes découvertes de la physique du 20e siècle : les ondes radio et rayons X, la radioactivité de certains éléments…

La production quotidienne massive d'images médicales ne peut être archivée dans un format commun de type JPEG au risque de perdre des données associées à l'image tels que nom du patient, type d'examen, hôpital, etc ... Le format *DICOM* permet de rendre unique chaque image produite et de lui associer des informations spécifiques. Cela a pour conséquence de produire des images autonomes dans la mesure où il est toujours possible d'identifier formellement leurs origines en cas de perte, de renommage ou de reproduction.

IX. Références bibliographiques

[1] W D Bidgood Jr 1, S C Horii, F W Prior, D E Van Syckle. *Understanding and using DICOM, the data interchange standard for biomedicalimaging : Journal of the American MedicalInformatics Association (1997)*

[2]Bairagi, VinayakMemorial, ShivajiView, HusainDwt, DimensionalBairagi, Vinayak :*The Role of DICOM Technology in Telemedicine The Role of DICOM Technology in Telemedicine (2014)*

[3] *https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html*

[4] https://www.postdicom.com/fr/blog/medical-imaging-science-and-applications