Sup-Galilée

Projet de traitement des images

Trajectoires de particules

Réalisé par : Omar BOUZEKRI
Encadrant : John CHAUSSARD

Table des matières

Introduction

Une chambre a bulles est un moyen de détecter des particules lors de certaines expériences. Les particules traversent un liquide qu'elles chauffent, ce qui produit de microscopiques bulles que l'on peut photographier. En étudiant ces bulles, les scientifiques en déduisent la trajectoire, et les propriétés des particules émises.

L'objectif de ce projet consiste en l'établissement et l'implantation d'un algorithme permettant de trouver l'angles des trajectoires dans une image. Le projet est réalisé en python et utilise les librairies opencv,numpy,sys et les fonctions du cours.

Ce projet est composé de deux parties, nous traiterons dans la première une image contenant une trajectoire de particule dont il faut calculer l'angle grâce aux caclul du nombre de composantes 8 connexes. Nous commencerons par étudier une trajectoire d'angle connu dont nous allons compter le nombre de composantes 8 connexes, effectuez une fermeture par une ligne de 20 pixels de long et d'angle variant de 0 à 90 degres, puis en déduire l'angle. Nous trouverons à la fin un programme qui permet d'identifier l'angle d'une trajectoire. Nous allons traiter dans la deuxième partie une image contenant plusieurs trajectoires à identifier. Pour cela, nous allons commencer par tracer une fermeture par une ligne de 20 pixels de long et d'angle variant de 0 à 90 degres, par pas de 10 degrés, calculer la taille de la plus grande composantes 8 connexes. Nous allons en déduire un programme qui permet de calculer l'angle de plusieurs trajectoires dans une image.

le dernier chapitre est consacrée au code python avec des images des scripts et résultats obtenus.

Chapitre 1

Une seule trajectoire

1.1 Question 1

Nous allons compter le nombre de composantes 8-connexes de l'image : une trajectoire angle_50.png Pour cela, on va utiliser une fonction remove_object qui permet de trouver les composantes 8 connexes et une fonction count object qui permet de calculer le nombre de composantes 8 connexes. Voici le pseudo code de la fonction remove_object :

Algorithm remove_object: Pseudo code

Paramètres : objects_members : liste contenant les coordonnées des différents objets '

start_point : coordonnées de la première valeur de la liste

(x,y) ← start_point

connex = [(x - 1, y), (x, y - 1),
(x + 1, y), (x, y + 1),
(x - 1, y+1), (x-1, y-1),
(x + 1, y+1), (x+1, y-1)]

POUR point dans connex :

ON_ESSAYE :

On supprime le point de object_members

remove_object (object_members,point)

SI erreur Alors

On passe

Fin_SI

Fin_POUR

Voici le pseudo code de la fonction count_object :

```
Algorithm count_object: Pseudo code
Paramètre : Image
    width, height ← Taille de l'image
    fonction objects_members
        Paramètre : width, height ← Taille de l'image
        A ← liste vide
        pour x de 0 à largeur(width) :
            POUR y de 0 à longueur(height) :
                SI image[y][x] = 1
                    On rajoute (x,y) à A
                FIN_SI
            FIN_POUR
        FIN_POUR
        Retourne : A : liste de (x,y)
    objects_count←0
    TANT_QUE objects_members différent du vide
        remove_object(objects_members, objects_members.pop(0))
        objects_count← objects_count+1
    FIN_TANT_QUE
Résultat : nombres de composante 8 connexes
```

Le nombre de composante 8 connexes de l'image d'angle 50 est : 325.

1.2 Question 2

Voici le pseudo code de l'algorithme qui permet d'afficher l'angle et le nombre de composantes 8 connexes :

```
Algorithm Question 2: Pseudo code
    image ←  une trajectoire angle 50.png
    POUR angle ALLANT_DE 0 A 90 PAR_PAS_DE 10
        el = construction d'une ligne de 20 pixels de rotation angle
        fe = fermeture de image et el
        con = fonction count object pour l'image fe
        ECRIRE('l'angle est :', angle)
        ECRIRE('le nombre de composantes 8 connexes :', con)
    FIN_POUR
,
```

E est une ligne de 20 pixels de long et d'angle α	α=0	α=10	α=20	α=30	α=40	α=50	α=60	α=70	α=80	α=90
Nb de composantes 8-connexes de I•E	185	283	250	234	97	1	72	223	267	181

Nous remarquons que quand le nombre de composantes 8 connexes est égale à 1 on retrouve l'angle de la trajectoire.

1.3 Question 3

Voici le pseudo code de l'algorithme permettant de trouver et d'afficher l'angle de la trajectoire de la particule sur une image avec une seule trajectoire :

Algorithm Angle de la trajectoire : Pseudo code

```
Paramètres : Une image

POUR angle ALLANT_DE 0 A 100 PAR_PAS_DE 10
        el = construction d'une ligne de 20 pixels de rotation angle
        fe = fermeture de image et el
        con = fonction count object pour l'image fe
        SI con == 1 ALORS
                Ecrire('l'angle est :',angle)
        FIN_SI
FIN_POUR
```

,

Chapitre 2

Plusieurs trajectoires

2.1 Question 1

Nous allons compter la taille en pixel de la plus grande composante 8-connexes de l'image : quatres_trajectoires_angle_30_50_130_170.png
Pour cela, nous allons utiliser la fonction max-comp-objet et taille-objet

Voici le pseudo code de la fonction max-comp-objet

```
Algorithm max_comp_objet: Pseudo code

Paramètres ; objects_members : liste contenant les coordonnées des différents objets

Paramètres :  start_point : coordonnées de la première valeur de la liste

    (x,y) ← coordonnées de la composante de valeur 255

     connex = [(x - 1, y), (x, y - 1),
               (x + 1, y), (x, y + 1),
                (x - 1, y+1), (x-1, y-1),
                 (x + 1, y+1), (x+1, y-1)]

    A ← liste vide

    POUR point dans connex :

        A = On rajoute le point à la liste

        On essaye :

            On supprime le point de object_members

            remove_object (object_members,point)

        SI Erreur Alors

             On passe

        Fin_SI

    Fin_POUR

    Max_element ← maximum de A

Résultat : liste Max_element
```

Voici le pseudo code de la fonction taille-objet :

```
Algorithm taille_ objet : Pseudo code
Paramètres : Une image
    (x,y) ← taille de l'image
    fonction objects_members
        Paramètre : width,height ← Taille de l'image
            POUR x de 0 à largeur(width) :
                POUR y de 0 à longueur(height) :
                    si image[y][x] = 1
                FIN_POUR
            FIN_POUR
        Résultat: liste de (x,y)
    taille_image←liste vide
    TANT_QUE objects_members different du vide
        Max_element ←max_comp_objet(object_memebers,object_members.pop(0))
        On rajoute Max_element à la liste taille _image
    FIN TANT_QUE
    taille← maximum de taille_image
    taille_finale ← taille[0]*taille[1]
Resultat : taille_finale
```

Voici le pseudo code de l'algorithme permettant de calculer et afficher la taille de la plus grande composante 8-connexe de l'image.

```
Algorithme final
    Image ← image quatre_ trajectoires_angle_30_50_130_160.png.
    POUR angle ALLANT_DE 0 A 100 PAR_PAS_DE 10
        el = construction d'une ligne de 20 pixels de rotation angle
        fe = fermeture de l'image et el
        taille = fonction taille_ objet pour l'image fe
        Ecrire ('la taille de la plus grande composante est :', taille)
    FIN_POUR
```

On retrouve les valeures suivantes :

E est une ligne de 20 pixels de long et d'angle α	α=0	α=10	α=20	α=30	α=40	α=50	α=60	α=70	α=80	α=90
Taille (en pixels) de la plus grande composantes 8-connexes de I•E	13600	113600	431200	452234	452234	116800	113600	443200	377600	113600

On remarque que l'angle de la plus grande valeur est : 30 et 40 (130=40+90) qui sont deux angles recherchés.

2.2 Question 2

Nous allons effacer les points correspondant à la trajectoire trouvée précédemment, et recommencez le même calcul. On trouve :

E est une ligne de 20 pixels de long et d'angle α	α=0	α=10	α=20	α=30	α=40	α=50	α=60	α=70	α=80	α=90
Taille (en pixels) de la plus grande composantes 8-connexes de I•E	444000	443200	439200	446400	443200	440800	436800	451200	450400	116800

Nous remarquons que la taille maximal correspond à l'angle 70 qui est une des deux trajectoire restant à identifier (160=70+90).

Chapitre 3

Codes Python

3.1 Partie 1

3.1.1 Question 1

```python
import cv2
import numpy as np

def remove_object(objects_members, start_point):
# la première coordonnées de la composante dont la valeur est 255
    x, y = start_point
# On regarde si la composante est 8 connexes :
    connex = [(x - 1, y), (x, y - 1),
              (x + 1, y), (x, y + 1),
              (x - 1, y+1), (x-1, y-1),
              (x + 1, y+1), (x+1, y-1)]
    for point in connex:
        try:
# On enlève le point  de l'objet
            objects_members.remove(point)
# On exerce la fonction avec les nouveaux paramètres
            remove_object(objects_members, point)
        except ValueError:
            pass

def count_objects(image):
# taille de l'image
    width, height = len(image[0]), len(image)
# On parcoure l'image et on retourne les coordonnées des pixels blanc
    objects_members = [(x, y) for x in range(width) for y in range(height) if image[y][x] == 255]
    objects_count = 0
    while objects_members != []:
        remove_object(objects_members, objects_members.pop(0))
        objects_count += 1
    return objects_count
```

La fonction count-objects retourne pour l'image d'angle 50 :

```python
imag = cv2.imread('une_trajectoire_angle_50.png', cv2.IMREAD_GRAYSCALE)
print(count_objects(imag))
```

```
trait  OO
/Users/omarbouzekri/opt/anaconda3/envs/facereco/bin/python /Users/omarbouzekri/opt/anaconda3/bin/facereco/trait.py
325

Process finished with exit code 0
```

3.1.2 Question 2

```python
import numpy as np
import cv2
from Commun import morpho, strel
from trait import count_objects
import sys
sys.setrecursionlimit(10 ** 6)


imag = cv2.imread('une_trajectoire_angle_50.png', cv2.IMREAD_GRAYSCALE)

for angle in range(0, 100,10):
# On construit la ligne
    el = strel.build("ligne", 20, angle)
# On effectue la fermeture
    fe = morpho.myclose(imag, el)
# On compte le nombre de composantes 8 connexes
    con = count_objects(fe)
    print(angle,con)

```

```
first × OO ×
0 185
10 283
20 250
30 234
40 97
50 1
60 72
70 223
80 267
90 181
```

3.1.3 Question 3

La fonction qui permet de trouver l'angle d'une trajectoire dans une image. Dans notre cas, j'ai pris l'image d'angle 30 :

```python
import numpy as np
import cv2
from Commun import morpho, strel

from trait import count_objects
import sys

sys.setrecursionlimit(10 ** 6)

def angle_traj(image):
    for angle in range(0, 100, 10):
        el = strel.build("ligne", 20, angle)
        fe = morpho.myclose(image, el)
        con = count_objects(fe)
        if con == 1:
            print("l'angle est:", angle)

imag = cv2.imread('une_trajectoire_angle_30.png', cv2.IMREAD_GRAYSCALE)

print(angle_traj(imag))
```

```
angle_traj() › for angle in range(0, 100, 10) › if con == 1
second × angle_trajectoire ×
/Users/omarbouzekri/opt/anaconda3/envs/facereco/bin/python /Users/omarbouzekri/opt/anaconda3/bin/facerec
l'angle est: 30
```

3.2 Partie 2

3.2.1 Question 1

```python
import cv2
import numpy as np
from Commun import morpho, strel
import sys
sys.setrecursionlimit(10 ** 6)


def max_comp_objet(objects_members, start_point):
    # la première coordonnées de la composante dont la valeur est 255
    x, y = start_point
    # Composante 8 connexes :
    connex = [(x - 1, y), (x, y - 1),
              (x + 1, y), (x, y + 1),
              (x - 1, y + 1), (x - 1, y - 1),
              (x + 1, y + 1), (x + 1, y - 1)]
    A = []
    for point in connex:
        # on rajoute le point dans la liste A
        A.append(point)
        try:
            # On enlève le point  de l'objet
            objects_members.remove(point)
            # On exerce la fonction avec les nouveaux paramètres
            max_comp_objet(objects_members, point)
        except ValueError:
            pass
        # le maximum des composantes des différentes listes
        B = max(A)
    return B
```

```python
def taille_objet(image):
    # la taille de l'image
    width, height = len(image[0]), len(image)
    # On parcoure l'image et on retourne les coordonnées des pixels blanc
    objects_members = [(x, y) for x in range(width) for y in range(height) if image[y][x] == 255]
    n = []
    while objects_members != []:
        o = max_comp_objet(objects_members, objects_members.pop(0))
        # On rajoute ce maximum dans une liste
        n.append(o)
    # On calcule le maximum des différentes listes qu'on a rajouté dans n
    m = max(n)
    # On retourne la taille
    return m[0] * m[1]
```

On trouve les valeures données dans le tableau :

```
imag = cv2.imread('quatre_trajectoires_angle_30_50_130_160.png', cv2.IMREAD_GRAYSCALE)


for angle in range(0, 100, 10):
    # On construit la ligne
    el = strel.build("ligne", 20, angle)
    # On effectue la fermeture
    fe = morpho.myclose(imag, el)
    # On calcule la taille
    e = taille_objet(fe)
    print(angle,e)
```

```
max_comp_objet() › for point in connex › try
second × part2 ×
/Users/omarbouzekri/opt/anaconda3/envs/facereco/bin/python /Users/omarbouzekri/opt/anaconda3/bin/facerec
0 113600
10 113600
20 431200
30 452234
40 452234
50 116800
60 113600
70 443200
80 377600
90 113600

Process finished with exit code 0
```

Pour trouver l'angle directement, nous utilisons la fonction max_comp_objet et :

```
def taille_objet(image):
    # la taille de l'image
    width, height = len(image[0]), len(image)
    # On parcoure l'image et on retourne les coordonnées des pixels blanc
    objects_members = [(x, y) for x in range(width) for y in range(height) if image[y][x] == 255]
    n = []
    while objects_members != []:
        o = max_comp_objet(objects_members, objects_members.pop(0))
        # On rajoute ce maximum dans une liste
        n.append(o)
    # On calcule le maximum des différentes listes qu'on a rajouté dans n
    m = max(n)
    # On retourne la taille
    return m,m[0] * m[1]
```

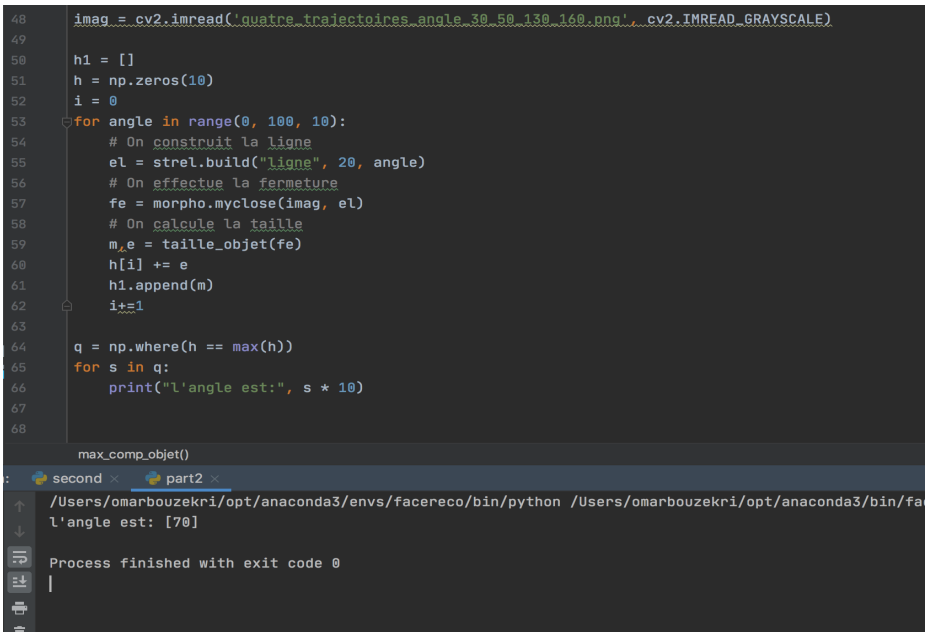
```
imag = cv2.imread('quatre_trajectoires_angle_30_50_130_160.png', cv2.IMREAD_GRAYSCALE)

h1 = []
h = np.zeros(10)
i = 0
for angle in range(0, 100, 10):
    # On construit la ligne
    el = strel.build("ligne", 20, angle)
    # On effectue la fermeture
    fe = morpho.myclose(imag, el)
    # On calcule la taille
    m,e = taille_objet(fe)
    h[i] += e
    h1.append(m)
    i+=1

q = np.where(h == max(h))
for s in q:
    print("l'angle est:", s * 10)
```

```
taille_objet()
second × part2 ×
/Users/omarbouzekri/opt/anaconda3/envs/facereco/bin/python /Users/omarbouzekri/opt/anaconda3/bin/facereco/part2
l'angle est: [30 40]

Process finished with exit code 0
```

3.2.2 Question 2

```python
import cv2
import numpy as np
from Commun import morpho, strel
import sys
sys.setrecursionlimit(10 ** 6)


def max_comp_objet(objects_members, start_point):
    # la première coordonnées de la composante dont la valeur est 255
    x, y = start_point
    # Composante 8 connexes :
    connex = [(x - 1, y), (x, y - 1),
              (x + 1, y), (x, y + 1),
              (x - 1, y + 1), (x - 1, y - 1),
              (x + 1, y + 1), (x + 1, y - 1)]
    A = []
    for point in connex:
        # on rajoute le point dans la liste A
        A.append(point)
        try:
            # On enlève le point  de l'objet
            objects_members.remove(point)
            objects_members.remove((800,554))
            # On exerce la fonction avec les nouveaux paramètres
            max_comp_objet(objects_members, point)
        except ValueError:
            pass
        # le maximum des composantes des différentes listes
        B = max(A)
    return B
```

```python
def taille_objet(image):
    # la taille de l'image
    width, height = len(image[0]), len(image)
    # On parcoure l'image et on retourne les coordonnées des pixels blanc
    objects_members = [(x, y) for x in range(width) for y in range(height) if image[y][x] == 255]
    n = []
    while objects_members != []:
        o = max_comp_objet(objects_members, objects_members.pop(0))
        # On rajoute ce maximum dans une liste
        n.append(o)
    # On calcule le maximum des différentes listes qu'on a rajouté dans n
    m = max(n)
    # On retourne la taille
    return m,m[0] * m[1]
```



```
imag = cv2.imread('quatre_trajectoires_angle_30_50_130_160.png', cv2.IMREAD_GRAYSCALE)

h1 = []
h = np.zeros(10)
i = 0
for angle in range(0, 100, 10):
    # On construit la ligne
    el = strel.build("ligne", 20, angle)
    # On effectue la fermeture
    fe = morpho.myclose(imag, el)
    # On calcule la taille
    m,e = taille_objet(fe)
    h[i] += e
    h1.append(m)
    i+=1

q = np.where(h == max(h))
for s in q:
    print("l'angle est:", s * 10)
```

max_comp_objet()

second | part2

```
/Users/omarbouzekri/opt/anaconda3/envs/facereco/bin/python /Users/omarbouzekri/opt/anaconda3/bin/fa
l'angle est: [70]

Process finished with exit code 0
```

Conclusion

Ce projet nous a permis de détecter l'angle des particules dans une image. Nous avons remarquer que pour trouver l'angle d'une seule trajectoire dans une image, il faut compter le nombres de composantes 8 connexes alors que dans une image avec plusieurs trajectoires il faut calculer la taille de ces composantes 8 connexes. Ce projet m'a permis de découvrir l'univers très vaste du traitement d'images et nous avons ainsi pu entrevoir une partie de la diversité des choses qu'il permet de faire. Nous avons pu toucher du doigt quelque chose de très utilisé aujourd'hui.