THỰC HÀNH 4: PHÂN VÙNG ẢNH

Sau bài thực hành này, sinh viên có thể

- Viết được chương trình phân vùng ảnh theo histogram
- Viết được chương trình phân vùng ảnh theo Region
- Viết được chương trình thay đổi ảnh

Phân vùng ảnh là quá trình chia ảnh thành nhiều vùng có cùng chung đặc tính.

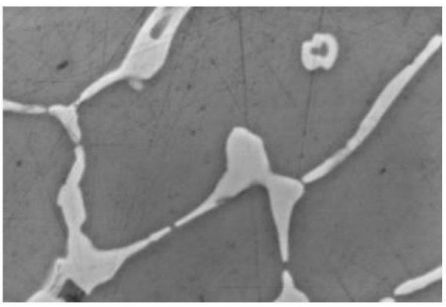
1. CÀI ĐẶT THƯ VIỆN

```
pip install opencv-python
```

2. VIẾT CHƯƠNG TRÌNH PHÂN VÙNG ẢNH

2.1. Phân vùng theo histogram

Một ngưỡng được xác định dựa theo histogram của ảnh. Mỗi pixel trong ảnh được so sánh với ngưỡng, nếu giá trị pixel nhỏ hơn ngưỡng, thì pixel trong phân vùng được gán giá trị 0. Ngược lại, gán giá trị 1.



(a) Input image.

(b) Output image.

2.1.1. Phương pháp Otsu

```
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
import matplotlib.pylab as plt
from skimage.filters.thresholding import threshold_otsu

data = Image.open('fruit.jpg').convert('L')

a = np.asarray(data)

# performing Otsu's thresholding
thres = threshold_otsu(a)

# pixels with intensity greater than theshold are kept
b = a > thres

b = Image.fromarray(b)

plt.imshow(b)
plt.show()
```

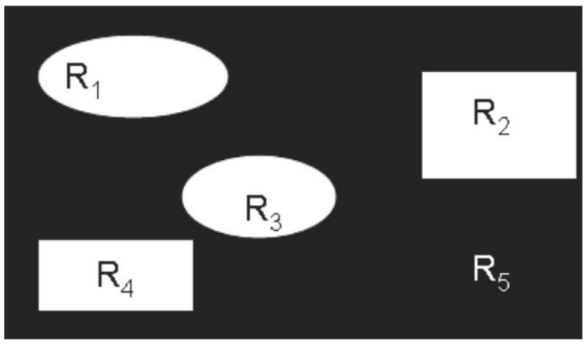
2.1.2. Phương pháp Adaptive Thresholding

Cải tiến phân vùng chính xác hơn Otsu. Chia ảnh thành nhiều ảnh nhỏ và tính threshold cho từng ảnh nhỏ

```
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
import matplotlib.pylab as plt
from skimage.filters import threshold_local

data = Image.open('fruit.jpg').convert('L')

a = np.asarray(data)

# performing local thresholding
b = threshold_local(a, 39, offset=10)

b = Image.fromarray(b)

plt.imshow(b)
plt.show()
```

2.2. Phân vùng theo region

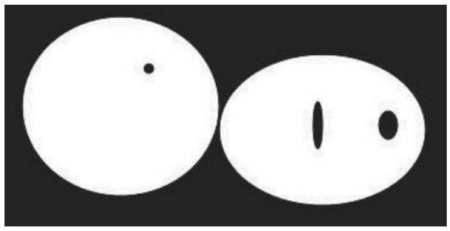
Một region là một nhóm các pixel có cùng thuộc tính.




```
import cv2
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
from scipy.ndimage import label
import matplotlib.pylab as plt
from skimage.filters import threshold_local

# opening the image and converting it to grayscale
data = cv2.imread('fruit.jpg')
# covnerting image from color to grayscale
a = cv2.cvtColor(data, cv2.COLOR_BGR2GRAY)
# thresholding the image to obtain cell pixels
thresh, b1 = cv2.threshold(a, 0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)
# since Otsu's method has over segmented the image
# erosion operation is performed
b2 = cv2.erode(b1, None, iterations = 2)
# distance transform is performed
dist_trans = cv2.distanceTransform(b2, 2, 3)
# thresholding the distance transform image to obtain
# pixels that are foreground
thresh, dt = cv2.threshold(dist_trans, 1, 255, cv2.THRESH_BINARY)
# performing labeling
labelled, ncc = label(dt)
# labelled is converted to 32-bit integer
labelled = labelled.astype(np.int32)
# performing watershed
cv2.watershed(data, labelled)
b = Image.fromarray(labelled)

plt.imshow(b)
plt.show()
```

2.3. Biến đổi đối tượng trong ảnh

Dilation cho phép các pixel ở foreground của 1 ảnh có thể co giãn.




2.3.1. Sử dụng binary_dilation

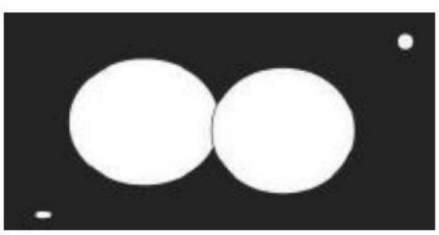
```
import cv2
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
from scipy.ndimage import label
import matplotlib.pylab as plt
from skimage.filters import threshold_local

# opening the image and converting it to grayscale
data = Image.open('dil_img.gif').convert('L')
b = nd.binary_dilation(data, iterations=50)


c = Image.fromarray(b)
c.show()
plt.imshow(c)
plt.show()
```

2.3.2. Sử dụng binary_opening

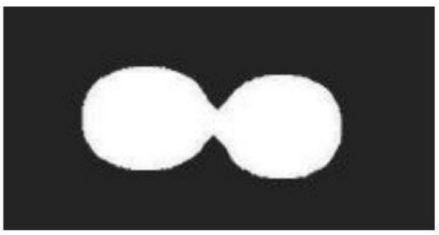
```
import cv2
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
from scipy.ndimage import label
import matplotlib.pylab as plt
from skimage.filters import threshold_local

# opening the image and converting it to grayscale
data = Image.open('dil_img.gif').convert('L')
# defining the structuring element
s = [[0, 1, 0], [1, 1, 1], [0, 1, 0]]
b = nd.binary_opening(data, structure=s, iterations=25)

c = Image.fromarray(b)
c.show()
plt.imshow(c)
plt.show()
```

2.3.3. Sử dụng binary_erosion

Dùng để co đối tượng bằng cách loại bỏ pixels ở biên của đối tượng



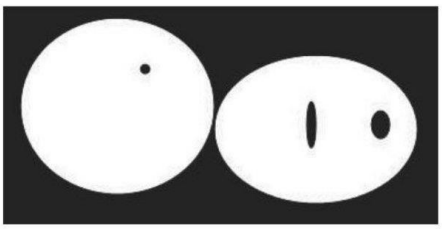
(a) Input image for erosion.



(b) Output image after 10 iterations.

```
import cv2
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
from scipy.ndimage import label
import matplotlib.pylab as plt
from skimage.filters import threshold_local

# opening the image and converting it to grayscale
data = Image.open('dil_img.gif').convert('L')
# defining the structuring element
s = [[0, 1, 0], [1, 1, 1], [0, 1, 0]]
b = nd.binary_erosion(data, structure=s, iterations=50)

c = Image.fromarray(b)
c.show()
plt.imshow(c)
plt.show()
```

2.3.4. Sử dụng binary_closing



(a) Input image for closing.

(b) Output image after closing.

```
import cv2
from PIL import Image
import numpy as np
import imageio.v2 as iio
import scipy.ndimage as nd
from scipy.ndimage import label
import matplotlib.pylab as plt
from skimage.filters import threshold_local

# opening the image and converting it to grayscale
data = Image.open('dil_img.gif').convert('L')
# defining the structuring element
s = [[0, 1, 0], [1, 1, 1], [0, 1, 0]]
b = nd.binary_closing(data, structure=s, iterations=50)

c = Image.fromarray(b)
c.show()
plt.imshow(c)
plt.show()
```

3. BÀI TẬP

1. Viết chương trình chọn LangBiang trong ảnh Đà Lạt từ thư mục exercise. Tịnh tiến vùng chọn sang phải 100px. Sử dụng phương pháp Otsu để phân vùng LangBiang theo ngưỡng 0.3. Lưu vào máy với tên lang_biang.jpgvà hiển thị trên màn hình.

2. Viết chương trình chọn Hồ Xuân Hương trong ảnh Đà Lạt từ thư mục exercise. Xoay đối tượng vừa chọn 1 góc 45^o và dùng phương pháp Adaptive Thresholding với ngưỡng 60 và lưu vào máy với tên là ho_xuan_huong.jpg.

3. Viết chương trình chọn Quản trường Lâm Viên trong ảnh Đà Lạt từ thư mục exercise. Dùng phương pháp Coordinate Mapping và Binary Closing cho vùng vừa chọn. Lưu vào máy với tên là quan_truong_lam_vien.jpg.

4. Tạo menu như hình sau:

```
├──geometric_transformation
│   ├──coordinate_mapping
│   ├──Rotate
│   ├──Scale
│   └──Shift
└──segment
    ├──Adaptive_thresholding
    ├──Binary_dilation
    ├──Binary_erosion
    └──Otsu
```

Viết chương trình cho phép người dùng nhập chức năng muốn xử lý. (Có thể chọn 1 chức năng duy nhất hoặc kết hợp 2 chức năng của geometric_transformation và segment)