Computer Assignment 3

CPE 261453 (Digital Image Processing)

โดย

นายเธียร สุวรรณกุล

รหัสนักศึกษา 620610176

เสนอ

ผศ.ดร.ศันสนีย์ เอื้อพันธ์วิริยะกุล

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่

Introduction

This report has been created for 261453 Digital Image Processing at Chiang Mai University. The purpose of this report is to demonstrate my understanding of image processing methods. The task assigned in this report is to find the number and positions of wormholes in given eggplant image, without using any toolboxes or external programs.

Thian Suwannakul

620610176

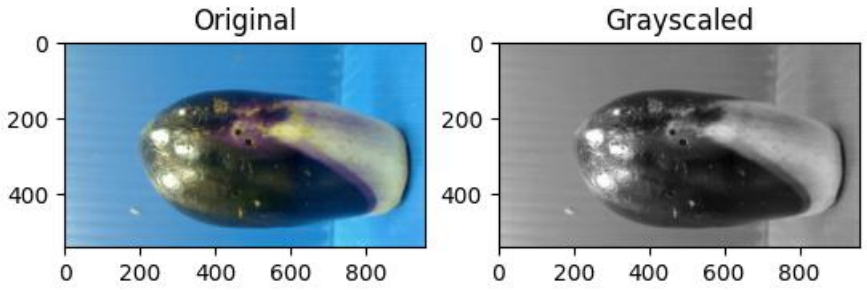
Methods

1. Define a function ToGrayscale() that takes an image and converts it to grayscale by taking a dot product of the image array with a weight vector [0.25, 0.25, 0.25].
2. Load an image and convert it to grayscale using the ToGrayscale() function.
3. Define a binary threshold value of 35 and use it to create a binary image where values below the threshold are set to 1 and the rest to 0.
4. Define a 5x5 kernel and use it to erode and then dilate the binary image to remove small objects and fill gaps in the wormholes.
5. Find the contours of the wormholes in the dilated binary image using the cv2.findContours() function. This returns a list of contours and a hierarchy of their relationships.
6. Make a copy of the original image and convert it to a numpy array.
7. Define a minimum circularity threshold of 0.82.
8. Iterate over the contours found in step 5 and calculate their circularity using the formula 4 * pi * area / perimeter^2. If the circularity is above the threshold, append the contour to a list of circular contours and draw it on the copy of the original image using the cv2.drawContours() function.
9. Count the number of circular contours and print their positions by iterating over them and using the cv2.moments() function to calculate the centroid of each contour.
10. Display the copy of the original image with the circular contours and their positions using the matplotlib.pyplot.imshow() and .show() functions.

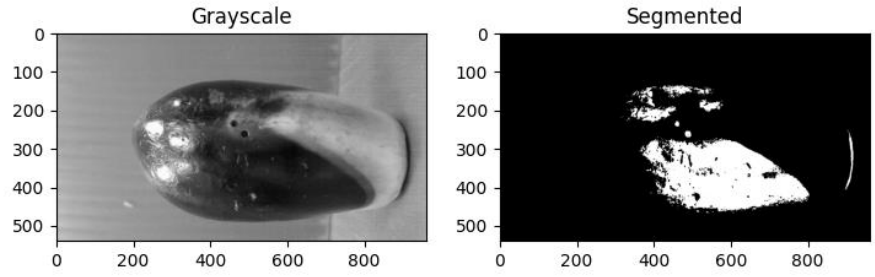
Result

After I've implemented the program following my methods

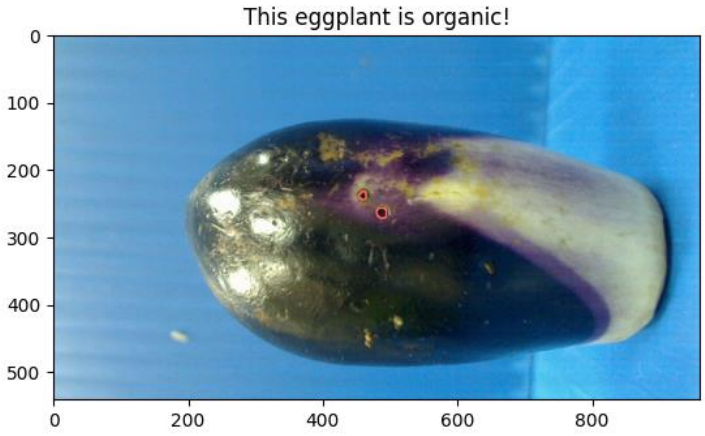
1. Convert the image to grayscale



2. Apply a binary threshold to segment the wormholes from the background. (using threshold < 35)

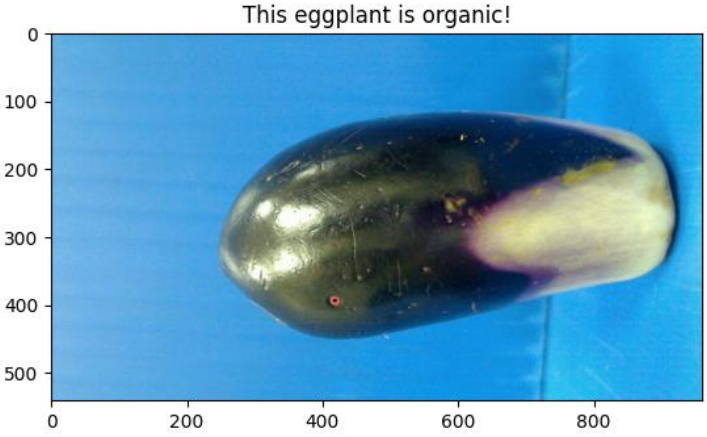


3. Process through 5x5 Kernel and dilated (Final output)



“WormHole_2H.tif”

```
Number of wormholes: 2
Wormhole #1 at position (487, 263)
Wormhole #2 at position (459, 237)
```



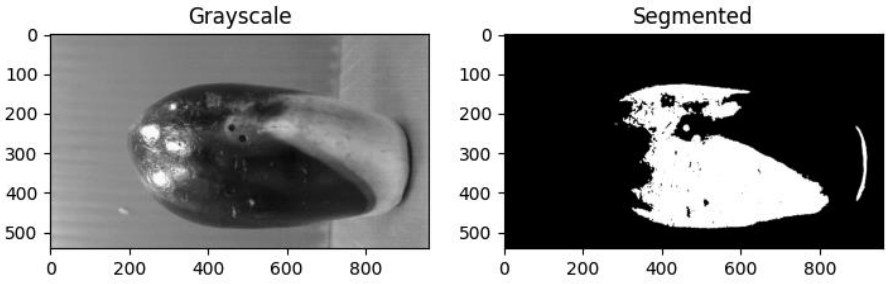
“WormHole_1H.tif”

```
Number of wormholes: 1
Wormhole #1 at position (418, 392)
```

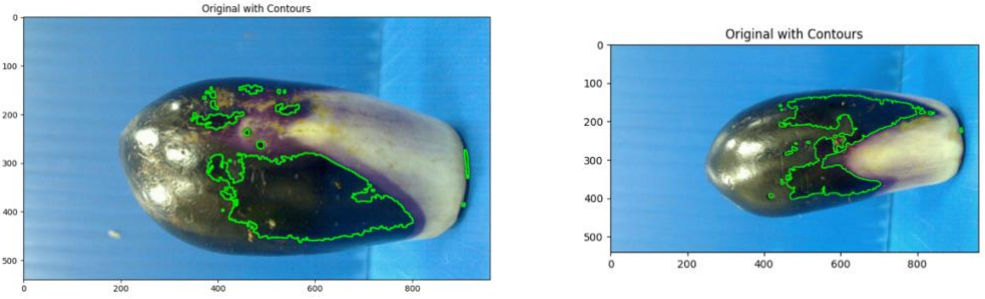
Discussion

Through the process of implementing this program, I've tried many methods, for example directly erosion and dilation without Kernel but it's wasn't work. Here're some of my problem between the implement.

Too much threshold



Not filtering out others than wormholes



And many more obstacles!

Conclusion

After many hours in this program, I find out that there’re many ways to detected objects in the picture. It’s very useful in many industries such as medical engineering and many more. In the process sometimes you have to use trials and errors method to adjust your argument which lead to the best result, for example threshold value and minimum circularity value.

Code (All implemented in Python)

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import cv2

# Define function to convert image to grayscale
def ToGrayscale(img):
    return np.dot(np.array(img, dtype='float32'), [0.25, 0.25, 0.25])

# Load image and convert to grayscale
img = Image.open('WormHole_1H.tif')
gray = ToGrayscale(img)

# Define binary threshold to segment wormholes
threshold = 35
binary = np.zeros_like(gray)
binary[gray < threshold] = 1

# Apply morphological operations to remove small objects and fill gaps in the wormholes
kernel = np.ones((5, 5), np.uint8).astype(np.uint8)
eroded = cv2.erode(binary, kernel, iterations=1)
dilated = cv2.dilate(eroded, kernel, iterations=1)

# Find contours of wormholes and draw them on the original image
dilated = dilated.astype(np.uint8)
contours, hierarchy = cv2.findContours(dilated, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
img_with_contours = img.copy()
img_with_contours = np.array(img_with_contours)


# Define minimum circularity threshold
min_circularity = 0.82

# Filter out non-circular contours and highlight circular ones
circular_contours = []
```

```
for contour in contours:
    area = cv2.contourArea(contour)
    perimeter = cv2.arcLength(contour, True)
    circularity = 4 * np.pi * area / (perimeter ** 2)
    if circularity > min_circularity:
        circular_contours.append(contour)
        cv2.drawContours(img_with_contours, [contour], -1, (255, 100, 100), 2)

# Count the number of wormholes and print their positions
print("Number of wormholes:", len(circular_contours))
for i, contour in enumerate(circular_contours):
    M = cv2.moments(contour)
    x = int(M['m10'] / M['m00'])
    y = int(M['m01'] / M['m00'])
    print("Wormhole #{} at position ({}, {})".format(i+1, x, y))

# Display original image with circular contours and wormhole positions
plt.imshow(img_with_contours)
plt.title('This eggplant is organic!')
plt.show()
```

This is my GitHub : https://github.com/Naihtton/DIP65-2