

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



Computer Vision  
Lớp L01

Panoramic photos (Image Stitching)

Giảng viên hướng dẫn: Võ Thanh Hùng

Sinh viên: Đinh Vũ Hà - 2113269

Thành phố Hồ Chí Minh, Tháng 05/2024



## Mục lục

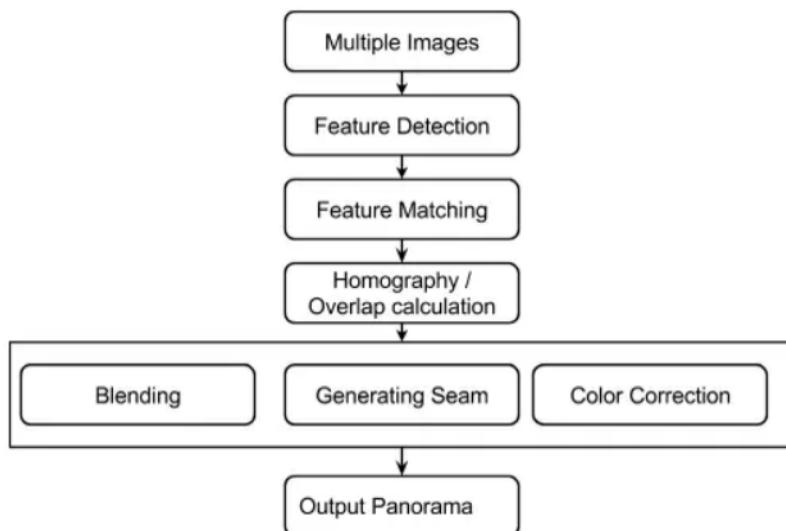
1 Yêu cầu	2
2 Lý thuyết	2
3 Code	3
4 Thực hiện	3

## 1 Yêu cầu

- \_ Cho 3-4 hình chụp một cảnh từ cùng một vị trí nhưng với các góc xoay khác nhau.
- \_ Hãy hiện thực các giải thuật để ghép các hình rời lại để tạo thành một bức hình góc rộng.
- \_ Các hình ví dụ được đính kèm, sinh viên được khuyến khích tìm/tự chụp thêm các hình khác để thử nghiệm thêm.
- \_ Hãy sử dụng 2 phương pháp trích feature khác nhau để so sánh và đánh giá.

## 2 Lý thuyết

Image Stitching là kỹ thuật kết hợp và ghép 1 tập các ảnh con để tạo nên 1 ảnh lớn, các ảnh con này có các vùng đè (overlap) lên nhau. Để giải quyết bài toán này, người ta sử dụng phương án giải quyết feature-based cho bài toán Image Stitching. Dưới đây là flow xử lý của nó:



Flow xử lý của feature-based

- Phát hiện điểm chung (Feature Detection): Các điểm chung trên các hình ảnh được tìm ra để xác định các điểm có thể ghép lại.
- Khớp điểm chung (Feature Matching): Các điểm chung được khớp giữa các hình ảnh để xác định cách chúng phải được sắp xếp lại khi ghép ảnh.



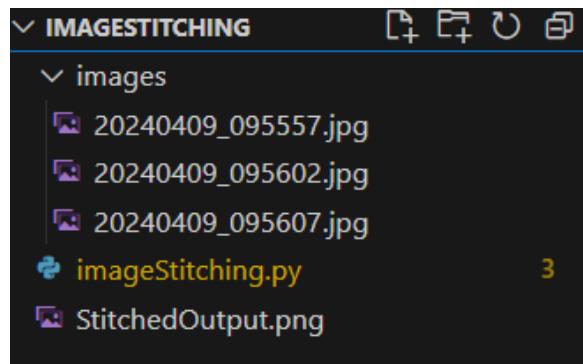
- Ước lượng ma trận chuyển đổi (Homography Estimation): Một ma trận chuyển đổi (homography) được tính toán để biến đổi các hình ảnh sao cho các điểm chung trên chúng trùng khớp.
- Tạo hình ảnh ghép (Image Warping): Các hình ảnh gốc được biến đổi bằng cách sử dụng ma trận chuyển đổi, làm cho các điểm chung giữa chúng trùng khớp.
- Ghép ảnh (Blending): Các hình ảnh được ghép lại với nhau để tạo ra hình ảnh cuối cùng. Quá trình này bao gồm việc chọn lựa và kết hợp các phần tử từ các hình ảnh gốc để tạo ra hình ảnh hoàn chỉnh mà không có hiện tượng rạn nứt hoặc biên giới rõ ràng.

### 3 Code

Link tới thư mục drive của Assignment: [https://drive.google.com/drive/folders/1nPL-t1QplLVR1Fb0hCCvusp=drive\\_link](https://drive.google.com/drive/folders/1nPL-t1QplLVR1Fb0hCCvusp=drive_link)

### 4 Thực hiện

Dưới đây là cấu trúc file của source code:



Cấu trúc file

Trong đó: thư mục images chứa các ảnh cần được ghép lại với nhau. File imageStitching.py để xử lý bài toán iamge stitching và StitchedOutput.png chính là ảnh sau khi đã được ghép.

Trong bài này, ta sẽ sử dụng các ảnh sau làm input:



Chúng ta sẽ đến với source code chính của bài tập lớn này. Đầu tiên, ta cần import các thư viện cần thiết. Ta sử dụng hàm glob.glob để có thể lấy để lấy tất cả các đường dẫn ảnh có phần mở rộng là .jpg



(ở đây chính là các ảnh mà ta cần ghép lại với nhau). Ta thực hiện resize lại cho bức ảnh, với chiều dài là 1920px và chiều rộng là 1440px và thực hiện hiển thị ra các ảnh

```
1 import numpy as np
2 import cv2
3 import glob
4 import imutils
5
6 image_paths = glob.glob('images/*.jpg')
7 images = []
8
9
10 for image in image_paths:
11     img = cv2.imread(image)
12     w , h = 1920 , 1440
13     img = cv2.resize(img,(w,h))
14     images.append(img)
15
16     cv2.imshow("Images", img)
17     cv2.waitKey(0)
18
```

Xử lý ban đầu

Tiếp đến, ta thực hiện stitch ảnh bằng hàm cv2.Stitcher\_create():

```
19
20 imageStitcher = cv2.Stitcher_create()
21
22 error, stitched_img = imageStitcher.stitch(images)
23
```

Thực hiện stitch ảnh

Tiếp theo, ta sẽ xử lý ảnh đã được stitch. Ta có code phần này như sau:



```
24 if not error:
25
26     stitched_img = cv2.copyMakeBorder(stitched_img, 10, 10, 10, 10, cv2.BORDER_CONSTANT, (0,0,0))
27
28     gray = cv2.cvtColor(stitched_img, cv2.COLOR_BGR2GRAY)
29     thresh_img = cv2.threshold(gray, 0, 255 , cv2.THRESH_BINARY)[1]
30
31
32
33     contours = cv2.findContours(thresh_img.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
34
35     contours = imutils.grab_contours(contours)
36     areaOI = max(contours, key=cv2.contourArea)
37
38
39     mask = np.zeros(thresh_img.shape, dtype="uint8")
40     x, y, w, h = cv2.boundingRect(areaOI)
41     cv2.rectangle(mask, (x,y), (x + w, y + h), 255, -1)
42
43     minRectangle = mask.copy()
44     sub = mask.copy()
45
46     while cv2.countNonZero(sub) > 0:
47         minRectangle = cv2.erode(minRectangle, None)
48         sub = cv2.subtract(minRectangle, thresh_img)
49
50
51     contours = cv2.findContours(minRectangle.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
52
53     contours = imutils.grab_contours(contours)
54     areaOI = max(contours, key=cv2.contourArea)
55
56
57     x, y, w, h = cv2.boundingRect(areaOI)
58
59     stitched_img = stitched_img[y:y + h, x:x + w]
60
61     cv2.imwrite("StitchedOutput.png", stitched_img)
62
63     cv2.imshow("Stitched Image Processed", stitched_img)
64
65     cv2.waitKey(0)
```

Thực hiện hậu stitch

Giải thích code:

- Ta thêm viền bằng cv2.copyMakeBorder với màu đen.

```
26
27     stitched_img = cv2.copyMakeBorder(stitched_img, 10, 10, 10, 10, cv2.BORDER_CONSTANT, (0,0,0))
28
```

- Chuyển đổi sang ảnh xám (gray) và áp dụng phương pháp ngưỡng hóa để tạo ra ảnh nhị phân (thresh\_img).



```
29     gray = cv2.cvtColor(stitched_img, cv2.COLOR_BGR2GRAY)
30     thresh_img = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY)[1]
31
```

- Tìm các đường viền (contours) trong ảnh nhị phân và xác định vùng quan tâm (areaOI) bằng cách tìm đường viền lớn nhất.

```
34     contours = cv2.findContours(thresh_img.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
35
36     contours = imutils.grab_contours(contours)
37     areaOI = max(contours, key=cv2.contourArea)
38
```

- Tạo mặt nạ (mask) cho vùng quan tâm và tìm hình chữ nhật bao quanh nó, điều chỉnh hình chữ nhật bằng cách loại bỏ các pixel trắng từ ảnh nhị phân.

```
39     mask = np.zeros(thresh_img.shape, dtype="uint8")
40     x, y, w, h = cv2.boundingRect(areaOI)
41     cv2.rectangle(mask, (x,y), (x + w, y + h), 255, -1)
42
43     minRectangle = mask.copy()
44     sub = mask.copy()
45
```

- Tìm lại các đường viền cho hình chữ nhật đã được điều chỉnh.

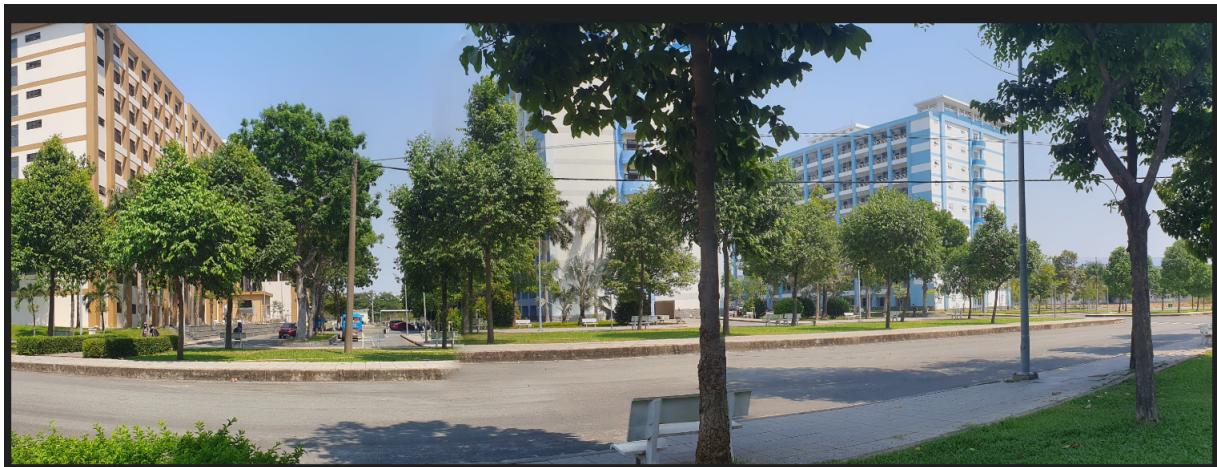
```
46     while cv2.countNonZero(sub) > 0:
47         minRectangle = cv2.erode(minRectangle, None)
48         sub = cv2.subtract(minRectangle, thresh_img)
49
50
51     contours = cv2.findContours(minRectangle.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
52
53     contours = imutils.grab_contours(contours)
54     areaOI = max(contours, key=cv2.contourArea)
55
```

- Cắt ảnh stitched dựa trên hình chữ nhật đã được điều chỉnh và lưu ảnh kết quả vào file Stitched-Output.png và hiển thị lên màn hình.



```
57     x, y, w, h = cv2.boundingRect(areaOI)
58
59     stitched_img = stitched_img[y:y + h, x:x + w]
60
61     cv2.imwrite("StitchedOutput.png", stitched_img)
62
63     cv2.imshow("Stitched Image Processed", stitched_img)
64
65     cv2.waitKey(0)
```

Sau khi thực hiện các bước trên, chạy code bằng python imageStitching.py, ta có ảnh kết quả:



Kết quả