

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG.**

**Khoa Công Nghệ Thông Tin**

**BÁO CÁO**

**Môn: Hệ cơ sở dữ liệu đa phương tiện**

**Đề** **tài: Nhận diện ảnh hoa**

**GVHD: TS. Nguyễn Đình Hóa**

**Nhóm môn học: 1**

**Nhóm BTL: 23**

**Thành viên:**

|  |  |  |
| --- | --- | --- |
| 1. | Đỗ Bùi Phương Anh | B15DCCN044 |
| 2. | Trần Thị Sen | B15DCCN457 |
| 3. | Hà Thu Thủy | B15DCCN549 |
|  |  |  |
|  |  |  |

***LỜI NÓI ĐẦU***

Trong sự phát triển của khoa học công nghệ, thiết bị số đang dần chiếm ưu thế giúp ta lưu trữ dữ liệu một cách phong phú và đa dạng. Ngoài dữ liệu văn bản đã có từ lâu, chúng ta có thêm các kiểu dữ liệu mới như: hình ảnh, âm thanh, video. Vì vậy việc tìm hiểu về các loại dữ liệu mới này là rất có ý nghĩa.

Trên cơ sở những kiến thức đã học trong môn Cơ sở dữ liệu đa phương tiện, chúng em tìm hiểu về các kỹ thuật lấy ra đặc trưng hình dạng của đối tượng trong ảnh dự theo HOG( Histogram of oriented gradients) để nhận diện hình hoa. Các nội dung chính bao gồm:

1. Tổng quan về xử lý ảnh.

2. Ký thuật HOG

3. Demo chương trình

Do kiến thức còn hạn hẹp nên bài tập lớn của chúng em còn nhiều sai sót, hạn chế. Chúng em mong thầy góp ý và sửa chữa để có thể hoàn thiện hơn. Chân thành cảm ơn thầy!

# 1. TỔNG QUAN VỀ XỬ LÝ ẢNH

1.1 Xử lý ảnh là gì?

Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ hoạ đó phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Xử lý ảnh và đồ hoạ đóng một vai trò quan trọng trong tương tác người máy.

Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh “tốt hơn” hoặc một kết luận.

Ảnh

Ảnh “Tốt hơn”

Kết luận

XỬ LÝ ẢNH

**Hình 1.1. Quá trình xử lý ảnh**

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến P(c1, c2,..., cn). Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều.

Sơ đồ tổng quát của một hệ thống xử lý ảnh:

Lưu trữ

Hậu xử lý

Hệ quyết định

Thu nhận ảnh (Scanner, Camera, Sensor)

Tiền xử lý

Trích chọn

đặc trưng

Đối sánh rút ra kết luận

**Hình 1.2. Các bước cơ bản trong một hệ thống xử lý ảnh**

##### Các vấn đề cơ bản trong xử lý ảnh

* + 1. Một số khái niệm cơ bản

*\* Ảnh và điểm ảnh:*

Điểm ảnh được xem như là dấu hiệu hay cường độ sáng tại 1 toạ độ trong không gian của đối tượng và ảnh được xem như là 1 tập hợp các điểm ảnh.

*\* Mức xám, màu*

Là số các giá trị có thể có của các điểm ảnh của ảnh

* + 1. Nắn chỉnh biến dạng

Ảnh thu nhận thường bị biến dạng do các thiết bị quang học và điện

tử.

Pi

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | P’i |  |  |  |  |  |  |  |  |  |
|  |  | f(P | i) |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

**Ảnh thu nhận Ảnh mong muốn**

**Hình 1.3. Ảnh thu nhận và ảnh mong muốn**

Giả sử (Pi, Pi’) i = 1, *n* có n các tập điều khiển Tìm hàm f: Pi a f (Pi) sao cho



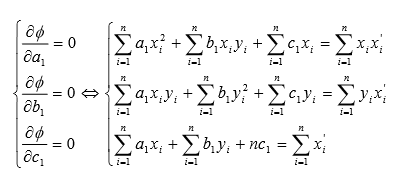
Giả sử ảnh bị biến đổi chỉ bao gồm: Tịnh tiến, quay, tỷ lệ, biến dạng bậc nhất tuyến tính. Khi đó hàm f có dạng:

f (x, y) = (a1x + b1y + c1, a2x + b2y + c2)

Ta có:



Để cho   min



Giải hệ phương trình tuyến tính tìm được a1, b1, c1

Tương tự tìm được a2, b2, c2

 Xác định được hàm f

* + 1. Khử nhiễu

Có 2 loại nhiễu cơ bản trong quá trình thu nhận ảnh

* + - * Nhiều hệ thống: là nhiễu có quy luật có thể khử bằng các phép biến đổi
* Nhiễu ngẫu nhiên: vết bẩn không rõ nguyên nhân khắc phục bằng các phép lọ
  + 1. Chỉnh mức xám:

Nhằm khắc phục tính không đồng đều của hệ thống gây ra. Thông thường có 2 hướng tiếp cận:

* + - * + Giảm số mức xám: Thực hiện bằng cách nhóm các mức xám gần nhau thành một bó. Trường hợp chỉ có 2 mức xám thì chính là chuyển về ảnh đen trắng. Ứng dụng: In ảnh màu ra máy in đen trắng.
        + Tăng số mức xám: Thực hiện nội suy ra các mức xám trung gian bằng kỹ thuật nội suy. Kỹ thuật này nhằm tăng cường độ mịn cho ảnh
    1. Trích chọn đặc điểm

Các đặc điểm của đối tượng được trích chọn tuỳ theo mục đích nhận dạng trong quá trình xử lý ảnh. Có thể nêu ra một số đặc điểm của ảnh sau đây:

**Đặc điểm không gian:** Phân bố mức xám, phân bố xác suất, biên độ, điểm uốn v.v..

**Đặc điểm biến đổi:** Các đặc điểm loại này được trích chọn bằng việc thực hiện lọc vùng (zonal filtering). Các bộ vùng được gọi là “mặt nạ đặc

điểm” (feature mask) thường là các khe hẹp với hình dạng khác nhau (chữ nhật, tam giác, cung tròn v.v..)

**Đặc điểm biên và đường biên:** Đặc trưng cho đường biên của đối tượng và do vậy rất hữu ích trong việc trích trọn các thuộc tính bất biến được dùng khi nhận dạng đối tượng. Các đặc điểm này có thể được trích chọn nhờ toán tử gradient, toán tử la bàn, toán tử Laplace, toán tử “chéo không” (zero crossing) v.v..

Việc trích chọn hiệu quả các đặc điểm giúp cho việc nhận dạng các đối tượng ảnh chính xác, với tốc độ tính toán cao và dung lượng nhớ lưu trữ giảm xuống.

* + 1. Nhận dạng

Nhận dạng tự động (automatic recognition), mô tả đối tượng, phân loại và phân nhóm các mẫu là những vấn đề quan trọng trong thị giác máy, được ứng dụng trong nhiều ngành khoa học khác nhau. Tuy nhiên, một câu hỏi đặt ra là: mẫu (pattern) là gì? Watanabe, một trong những người đi đầu trong lĩnh vực này đã định nghĩa: “Ngược lại với hỗn loạn (chaos), mẫu là một thực thể (entity), được xác định một cách ang áng (vaguely defined) và có thể gán cho nó một tên gọi nào đó”. Ví dụ mẫu có thể là ảnh của vân tay, ảnh của một vật nào đó được chụp, một chữ viết, khuôn mặt người hoặc một ký đồ tín hiệu tiếng nói. Khi biết một mẫu nào đó, để nhận dạng hoặc phân loại mẫu đó có thể:

Hoặc **phân loại có mẫu** (supervised classification), chẳng hạn phân tích phân biệt (discriminant analyis), trong đó mẫu đầu vào được định danh như một thành phần của một lớp đã xác định.

Hoặc **phân loại không có mẫu** (unsupervised classification hay clustering) trong đó các mẫu được gán vào các lớp khác nhau dựa trên một tiêu chuẩn đồng dạng nào đó. Các lớp này cho đến thời điểm phân loại vẫn chưa biết hay chưa được định danh.

Hệ thống nhận dạng tự động bao gồm ba khâu tương ứng với ba giai đoạn chủ yếu sau đây:

1o. Thu nhận dữ liệu và tiền xử lý. 2o. Biểu diễn dữ liệu.

3o. Nhận dạng, ra quyết định.

Bốn cách tiếp cận khác nhau trong lý thuyết nhận dạng là:

1o. Đối sánh mẫu dựa trên các đặc trưng được trích chọn.

2o. Phân loại thống kê.

3o. Đối sánh cấu trúc.

4o. Phân loại dựa trên mạng nơ-ron nhân tạo.

Trong các ứng dụng rõ ràng là không thể chỉ dùng có một cách tiếp cận đơn lẻ để phân loại “tối ưu” do vậy cần sử dụng cùng một lúc nhiều phương pháp và cách tiếp cận khác nhau. Do vậy, các phương thức phân loại tổ hợp hay được sử dụng khi nhận dạng và nay đã có những kết quả có triển vọng dựa trên thiết kế các hệ thống lai (hybrid system) bao gồm nhiều mô hình kết hợp.

Việc giải quyết bài toán nhận dạng trong những ứng dụng mới, nảy sinh trong cuộc sống không chỉ tạo ra những thách thức về thuật giải, mà còn đặt ra những yêu cầu về tốc độ tính toán. Đặc điểm chung của tất cả những ứng dụng đó là những đặc điểm đặc trưng cần thiết thường là nhiều, không thể do chuyên gia đề xuất, mà phải được trích chọn dựa trên các thủ tục phân tích dữ liệu.

* + 1. Nén ảnh

Nhằm giảm thiểu không gian lưu trữ. Thường được tiến hành theo cả hai cách khuynh hướng là nén có bảo toàn và không bảo toàn thông tin. Nén không bảo toàn thì thường có khả năng nén cao hơn nhưng khả năng phục hồi thì kém hơn. Trên cơ sở hai khuynh hướng, có 4 cách tiếp cận cơ bản trong nén ảnh:

* Nén ảnh thống kê: Kỹ thuật nén này dựa vào việc thống kê tần xuất xuất hiện của giá trị các điểm ảnh, trên cơ sở đó mà có chiến lược mã hóa thích hợp. Một ví dụ điển hình cho kỹ thuật mã hóa này là \*.TIF
* Nén ảnh không gian: Kỹ thuật này dựa vào vị trí không gian của các điểm ảnh để tiến hành mã hóa. Kỹ thuật lợi dụng sự giống nhau của các điểm ảnh trong các vùng gần nhau. Ví dụ cho kỹ thuật này là mã nén \*.PCX
* Nén ảnh sử dụng phép biến đổi: Đây là kỹ thuật tiếp cận theo hướng nén không bảo toàn và do vậy, kỹ thuật thướng nến hiệu quả hơn. \*.JPG chính là tiếp cận theo kỹ thuật nén này.
* Nén ảnh Fractal: Sử dụng tính chất Fractal của các đối tượng ảnh, thể hiện sự lặp lại của các chi tiết. Kỹ thuật nén sẽ tính toán để chỉ cần lưu trữ phần gốc ảnh và quy luật sinh ra ảnh theo nguyên lý Fractal
* Nén ảnh sử dụng phép biến đổi: Đây là kỹ thuật tiếp cận theo hướng nén không bảo toàn và do vậy, kỹ thuật thướng nến hiệu quả hơn. \*.JPG chính là tiếp cận theo kỹ thuật nén này.
* Nén ảnh Fractal: Sử dụng tính chất Fractal của các đối tượng ảnh, thể hiện sự lặp lại của các chi tiết. Kỹ thuật nén sẽ tính toán để chỉ cần lưu trữ phần gốc ảnh và quy luật sinh ra ảnh theo nguyên lý Fractal

## **2. KỸ THUẬT HOG**

## HOG(histogram of oriented gradients) là một feature descriptor được sử dụng trong computer vision và xử lý hình ảnh, dùng để detec một đối tượng. Các khái niệm về HOG được nêu ra từ năm 1986 tuy nhiên cho đến năm 2005 HOG mới được sử dụng rộng rãi sau khi Navneet Dalal và Bill Triggs công bố những bổ sung về HOG. Hog tương tự như các biểu đồ edge orientation, scale-invariant feature transform descriptors(như sift, surf,..), shape contexts nhưng hog được tính toán trên một lưới dày đặc các cell và chuẩn hóa sự tương phản giữa các block để nâng cao độ chính xác. Hog được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một object trong ảnh Bài toán tính toán Hog thường gồm 5 bước:

1. Chuẩn hóa hình ảnh trước khi xử lý
2. Tính toán gradient theo cả hướng x và y .
3. Lấy phiếu bầu cùng trọng số trong các cell
4. Chuẩn hóa các block
5. Thu thập tất cả các biểu đồ cường độ gradient định hướng để tạo ra feature vector cuối cùng.

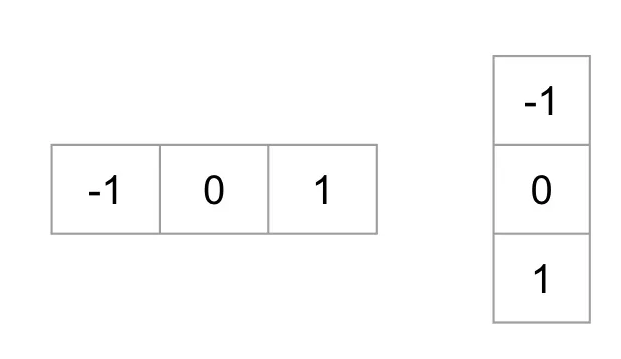
*2.1 Chuẩn hóa hình ảnh trước khi xử lý*

Bước chuẩn hóa này hoàn toàn không bắt buộc, nhưng trong một số trường hợp, bước này có thể cải thiện hiệu suất của bộ mô tả HOG. Có ba phương pháp chuẩn hóa chính mà chúng ta có thể xem xét:

* Quy định về chuẩn Gamma /power
* Chuẩn hoá gốc-vuông
* Variance normalization

*2.2 Tính toán gradient*

Để lấy được hình ảnh gradient, trước tiên chúng ta cần tính toán độ dốc ngang và dọc. Điều này có thể dễ dàng đạt được bằng cách nhân tích chập mã trận ảnh gốc với ma trận ngang dọc Dx và Dy:

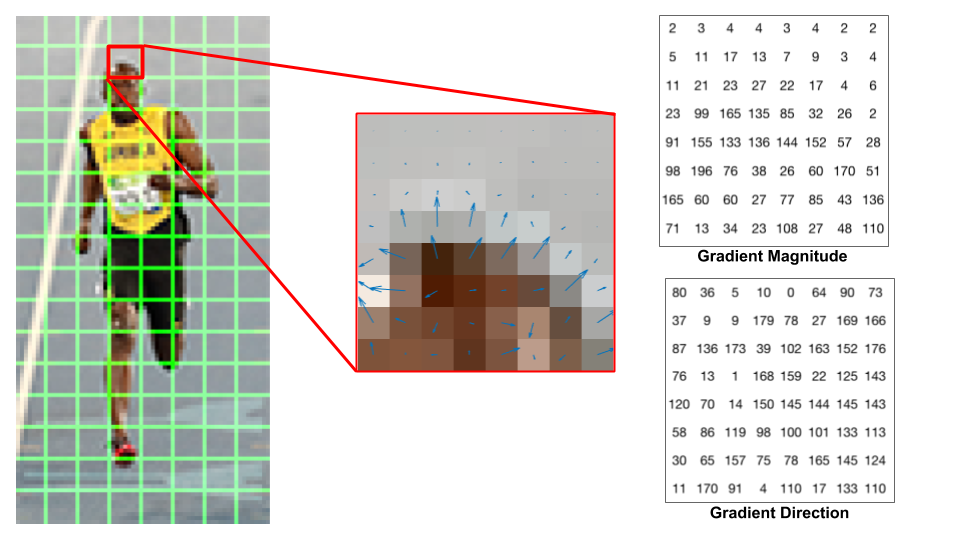


Sau đó có thể tính được các cường độ gradient của hình ảnh dựa theo công thức:



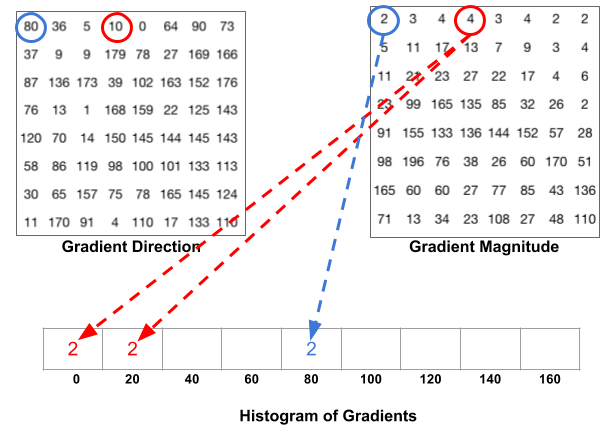
Cuối cùng, định hướng của gradient cho mỗi pixel trong hình ảnh ban đầu được tính bằng cách:

*θ*  =*arctan*2(*Gy*​,*Gx*​)

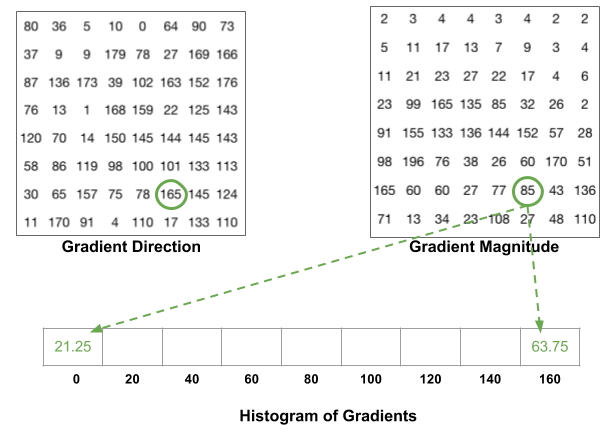


*2.3 Lấy votes trong mỗi cell*

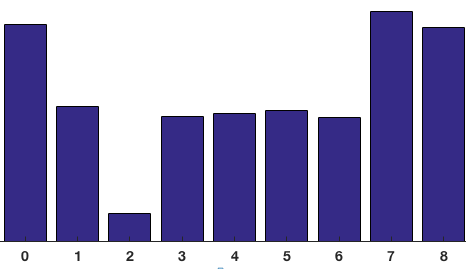
Bây giờ chúng ta cần chia hình ảnh của chúng ta thành các cell và block Một cell là một vùng hình chữ nhật được xác định bởi số điểm ảnh thuộc mỗi cell. Ví dụ: nếu ta có một hình ảnh 128 x 128 với pixel\_per\_cell = 8 x 8 thì sẽ có 16 x 16 = 256 cell. Với mỗi cell trong bức ảnh, ta cần xây dựng 1 biểu đồ cường độ gradient, biểu đồ này chứa 9 cột tương ứng vớ 9 góc từ 0 đến 160. Như trong ví dụ dưới, đầu tiên là pixel có bao quanh màu xanh lam. Nó có hướng 80 độ và cường độ là 2, vì vậy ta thêm 2 vào cột thứ 5 (hướng 80 độ). Tiếp theo là pixel có bao quanh màu đỏ. Nó có hướng 10 độ và cường độ 4. Vì không có cột 10 độ, nên ta vote cho cột 0 độ và 20 độ, mỗi cột thêm 2 đơn vị



Mỗi pixcel sẽ được vote vào vào biểu đồ, trọng số của mỗi vote chính là cường độ gradient tại pixel đó. Trong trường hợp góc lớn hơn 160 độ, nó nằm giữa 160 độ và 180 độ ta biết góc bao quanh tạo thành 0 độ và 180 độ là tương đương. Như trong ví dụ dưới đây, xét pixel bao quanh màu xanh lá, ta thấy 165 đóng góp vào cột 0 độ và 160 độ. Vì tỷ lệ giữa 165 với 0 và 160 là khác nhau nên ta vote vào 2 cột là khác nhau.



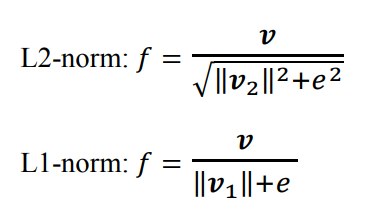
Cuối cùng, mỗi pixel đóng góp một phiếu bầu có trọng số vào biểu đồ - trọng lượng của phiếu chỉ đơn giản là cường độ gradient |G| tại pixel đó.Lúc này, chúng ta có thể thu thập và ghép các biểu đồ này để tạo ra feature vector cuối cùng:



Chuẩn hóa các block để có kết quả tốt hơn

*2.4 Chuẩn hóa các block*

Một lần nữa, ta cần chia các block giống như chia cell ở phía trên. Đơn vị của ta không còn là các điểm ảnh nữa mà là các cell. Người ta thường sử dụng hoặc 2 x 2 hoặc 3 x 3 cell\_per\_block có được độ chính xác hợp lý trong hầu hết các trường hợp.Các block này sẽ chồng lên nhau. Ví du: ta có 3x3 cells và cell\_per\_block= 2x2 thì ta sẽ có 4 block. Để cuẩn hóa các block t có thể sử dụng các công thức chuẩn hóa L1-norm hoặc L2-norm



Trong đó: v là vector cần chuẩn hóa chứa tất cả các histogram của một khối

||v(k)|| là giá trị chuẩn hóa của v theo các chuẩn k = 1,3

e là một hằng số rất nhỏ

*2.5 Tính toán vector đặc trưng của HOG*

* Với mỗi hình ảnh kích thước 64x128, chia thành các block 16x16 chồng nhau, sẽ có 7 block ngang và 15 block dọc, nên sẽ có 7x15=105 blocks.
* Mỗi block gồm 4 cell. Khi áp dụng biểu đồ 9-bin cho mỗi cell, mỗi block sẽ được đại diện bởi một vector có kích thước 36x1.
* Vì vậy, khi nối tất cả các vector trong một block lại với nhau, ta sẽ thu được vector đặc trưng HOG của ảnh có kích thước 36x1x105 = 3780x1.

3. **DEMO CHƯƠNG TRÌNH**

3.**1** Sử dụng HOG để lấy các feature

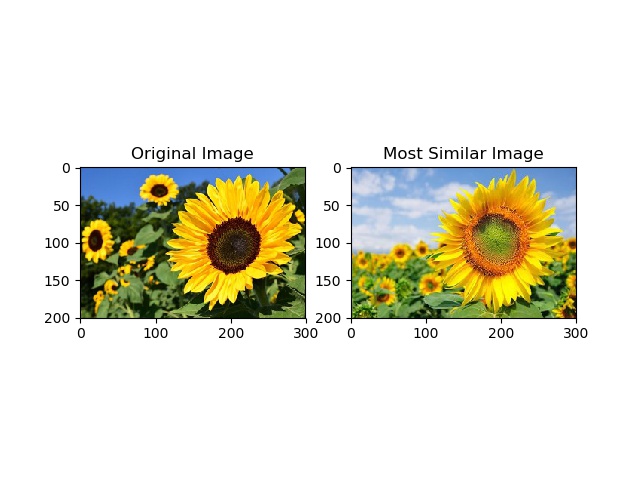
path = "./feature"  
for filename in os.listdir(path):  
 if filename[-3:] == "npy":  
 continue  
 print(filename)  
 img = imread(os.path.join(path, filename))  
 # imshow(img)  
 # plt.show()  
 features, hog\_img = hog(img, orientations=12,  
 pixels\_per\_cell=(8, 8),  
 cells\_per\_block=(2, 2),  
 transform\_sqrt=True,  
 visualise=True,  
 feature\_vector=True)  
 np.save(os.path.join(path, filename[:-4]), features)

3.2 So sánh các feature

for file\_img in os.listdir(path\_check):  
 src = imread(path\_check + file\_img)  
 dst = src  
 mindist = 10000000000000000  
 featsrc, hog\_img = hog(src, orientations=12,  
 pixels\_per\_cell=(8, 8),  
 cells\_per\_block=(2, 2),  
 transform\_sqrt=True,  
 visualise=True,  
 feature\_vector=True)  
  
 path = "./feature"  
 for filename in os.listdir(path):  
 if filename[-3:] == "npy":  
 featdst = np.load(os.path.join(path, filename))  
 if mindist > distance.euclidean(featsrc, featdst):  
 mindist = distance.euclidean(featsrc, featdst)  
 dst = imread(os.path.join(path, filename.replace(filename[-3:], "jpg")))  
  
 fig, ax = plt.subplots(1, 2)  
  
 for i, axi in enumerate(ax.flat):  
 axi.imshow(src)  
 axi.set\_title("Original Image")  
 break  
  
 for i, axi in enumerate(ax.flat):  
 if i == 0:  
 continue  
 axi.imshow(dst)  
 axi.set\_title("Most Similar Image")  
 break  
  
 # plt.show()  
 plt.savefig(path\_out+file\_img)

4. KẾT QUẢ THỰC NGHIỆM

4.1 Kết quả



4.2 Hạn chế

Vì chỉ tính toán dựa trên giá trị điểm ảnh mà không dựa trên màu sắc nên kết quả cho ra chưa đúng ở nhiều trường hợp.{\ displaystyle k = 1 + \ log \_ {2} (n) + \ log \_ {2} \ left (1 + {\ frac {| g\_ {1} |} {\ sigma \_ {g\_ {1}}} }\đúng)}

.