**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**----//----**

****

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**MÔN: TRUY VẤN THÔNG TIN ĐA PHƯƠNG TIỆN CS336.J21**

**ĐỀ TÀI: TRUY VẤN CƠ SỞ DỮ LIỆU VỚI ẢNH MẪU**

Giảng viên hướng dẫn:  **TS. NGUYỄN VINH TIỆP**

Sinh viên thực hiện: **Nguyễn Xuân Duy Hiển – 16521670**

**Trần Phước Lợi – 16521723**

**Nguyễn Nhật Trương – 16521336**

*TP.Hồ Chí Minh, tháng 6 năm 2019*

**MỤC LỤC**

[1.Tổng quan đề tài 4](#_Toc12046036)

[1.1. Đặt vấn đề 4](#_Toc12046037)

[1.2. Mục tiêu nghiên cứu 4](#_Toc12046038)

[1.3. Nội dung thực hiện 4](#_Toc12046039)

[1.4 Đề tài kế thừa 4](#_Toc12046040)

[2. Cơ sở lý thuyết 4](#_Toc12046041)

[2.1. Rút trích đặc trưng SIFT (Scale-Invariant Feature Transform) 4](#_Toc12046042)

[a. Scale-space extrema detection 4](#_Toc12046043)

[b. Keypoint localization 4](#_Toc12046044)

[c. Orientation Assignment 4](#_Toc12046045)

[d. Keypoint descriptor 4](#_Toc12046046)

[2.2. Cải tiến rút trích đặc trưng ROOTSIFT 4](#_Toc12046047)

[2.3. Thuật toán K-Means 4](#_Toc12046048)

[2.4. Mô hình Bag of Visual Word 4](#_Toc12046049)

[2.5 Các bước thực hiện 4](#_Toc12046050)

[3.1. Phân tích bài toán. 5](#_Toc12046051)

[3.2. Phân tích cấu trúc dữ liệu. 5](#_Toc12046052)

[3.3. Tìm hiểu giải thuật. 5](#_Toc12046053)

[3.3.1. Các bước giải thuật. 5](#_Toc12046054)

[3.3.2. Vấn đế 1: Rút trích đặc trưng SIFT của từng ảnh. 6](#_Toc12046055)

[3.3.3. Vấn đế 2: Rút trích đặc trưng rootSIFT của từng ảnh. 6](#_Toc12046056)

[3.3.5. Vấn đề 3: Đưa ảnh về dạng vector (sparse vector). 6](#_Toc12046057)

[3.3.6. Vấn đề 4: Đánh chỉ mục. 6](#_Toc12046058)

[3.3.7. Vấn đề 5: Truy vấn ảnh. 6](#_Toc12046059)

[4. Chạy chương trình 7](#_Toc12046060)

[5. Cải tiến truy vấn. 10](#_Toc12046061)

[5.1. Mở rộng câu truy vấn riêng biệt – Discriminative Query Expansion (DQE) 10](#_Toc12046062)

[5.2. Database-side feature augmentation 12](#_Toc12046063)

[6. Phân công 13](#_Toc12046064)

***Lời mở đầu***

*Môn học Truy vấn đa phương tiện cung cấp cho chúng em những kiến thức cơ bản về ngôn ngữ lập trình python, mathlap…, những nguyên lý, phương pháp và kỹ thuật xử lý ảnh cũng như văn bản đa phương tiện. Đồng thời cung cấp kiến thức giúp chúng em hiểu rõ các cơ chế xử lý của các thành phần về lĩnh vực đa phương tiện*

*Đề tài Truy vấn cơ sở dữ liệu với ảnh mẫu là một chủ để quen thuộc với nhiều sinh viên IT, để hiểu rõ hơn cơ chế của phân tích ảnh chúng em quyết định tìm hiểu sâu về truy vấn ảnh cũng như cơ chế hoạt trong bên trong đó cùng với các vấn để liên quan. Đây cũng là một chủ để chúng em rất thích, chính vì vậy chúng em đã chọn đề tài này cho đồ án cuối kì này.*

*Trong quá trình hoàn thành đề tài, chúng em đã tìm hiểu được các khái niệm về biến nói chung và một số khái niệm, đặc điểm về biến trong các ngôn ngữ lập trình phổ biến như: python, mathlap…vv cũng như kỹ năng tìm hiểu tham khảo tài liệu và các làm việc nhóm thuyết trình.*

*Chúng em xin trân trọng cám ơn sự hướng dẫn tận tình của thầy Nguyễn Vinh Tiệp về mặc chuyên môn cũng như định hướng. Vì kiến thức còn hạn hẹp nên trong quá trình thực hiện đề tài không thể tránh khỏi sai sót. Vì vậy rất mong nhận được sự góp ý của thầy để đề tài có thể hoàn thiện hơn nữa.*

# **1.Tổng quan đề tài**

## **1.1**. **Đặt vấn đề**

Trong thời đại công nghệ số những chiếc camera càng ngày càng phổ biến, kéo theo đó số lượng ảnh được sinh ra mỗi ngày một nhiều. Với số lượng ảnh lớn như thế vấn đề tìm kiếm những bức ảnh có nội dung liên quan nhau cũng trở nên khó khăn. Để giải quyết vấn đề đó và sử dụng những kiến thức về truy vấn thông tin đa phương tiện đã học được, chúng em quyết định chọn đề tài “Truy vấn cơ sở dữ liệu với ảnh mẫu” để thực hiện.

## **1.2.** **Mục tiêu nghiên cứu**

Mục tiêu của đề tài này là xây dựng một hệ thống truy vấn ảnh đơn giản có sử dụng mô hình bag of visaul word để giải quyết vấn đề tìm kiếm các bức ảnh có nội dung liên quan trong một kho ảnh lớn (vài nghìn ảnh).

## **1.3**. **Nội dung thực hiện**

Tìm hiểu về phương pháp bag of word và các phương pháp liên quan từ đó xây dựng module để nhận ảnh input và trả về danh sách kết quả.

Xây dựng giao diện cho chương trình giúp việc sử dụng dể dàng hơn.

## **1.4 Đề tài kế thừa**

Nhóm chúng em kế đồ án Nguyễn Ngọc Thừa – 15520859 về đề tài *Truy vấn trên ảnh sử dụng mô hình*

Quá trình kế thừa nhóm chúng em có thực hiện cải tiến tối ưu giao diện và cải tiến sử dụng rootSIFT và minibatchKmeans. Ngoài ra tìm hiểu thêm được hướng tối ưu về sau cho đồ án.

1. Cơ sở lý thuyết
   1. **Rút trích đặc trưng SIFT (Scale-Invariant Feature Transform)**

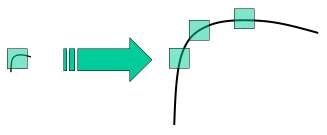
Để phát hiện 1 điểm gọi là đặc trưng của ảnh thì ta sử dụng phát hiện corner (corner detection) như: harris corner sẽ phát hiện được sự thay đổi về cường độ của bức ảnh theo nhiều chiều khác nhau, từ đó sẽ tìm ra được điểm đặc trưng (local feature). Nhưng điểm yếu của thuật toán này nếu ta phóng to bức ảnh lên, điểm corner đó sẽ không còn phải là góc nữa. Khắc phục điểm yếu này David Lowe đã phát minh và đăng ký bản quyền vào năm 1999 thuật toán SIFT (scale-invariant feature transform), thuật toán cho phép nhận dạng được những điểm gọi là đặc trưng của bức ảnh (local feature) mang nhiều thông tin và sẽ biến những thành những vector nhiều chiều thường là 128 chiều và những thông tin về feature từ đó có thể dễ dàng tìm kiếm, so sánh và đối chiếu. SIFT có tính rotation-invariant (bất biến với mọi phép tỷ lệ, xoay,…)

Scale-Invarient Feature Transform (SIFT) được trải qua bốn giai đoạn chính:

* Scale-space extrema detection.
* Keypoint localization.
* Orientation assignment
* Keypoint descriptor.

Các giai đoạn cụ thể:

1. Scale-space extrema detection



*Hình 1. Sự khác nhau về tỉ lệ dẫn đến sự bất hợp lý về corner trong cửa sổ*

Đa phần các bài toán của computer vision chúng ta sẽ định nghĩa các cửa sổ sau đó quét theo các hướng khác nhau để tìm ra những điểm đặc trưng (feature)

Trong hình ta thấy bên trái là 1 feature cụ thể là corner, corner này phóng to lên và ta vẫn sử dụng cửa sổ như bên trái thì ta sẽ thu được 2 cửa sổ không phải là feature, chỉ có cửa sổ ở giữa là corner. Vậy nếu tất cả đều là corner thì là bất hợp lý. Do đó, scale-space filtering được sử dụng để giải quyết vấn đề này. SIFT sẽ tính DoG(Difference of Gaussians) trên mọi pixel của bức ảnh bằng cách lấy diff của Guassian Blur với 2 sigma khác nhau. Sau đó so sánh pixel đang xét với 8 pixels (neighbors pixel) lân cận và 9 pixels tương của scale ảnh ngay trên và9 pixels tương ứng ở scale dưới, nếu pixel đó là local extrema thì đó được gọi là keypoint ở scale đó. Điểm này chỉ được gọi ban đầu là keypoint tiềm năng

1. Keypoint localization

Sau khi lấy được các keypoint tiềm năng, SIFT sử dụng chuỗi Taylor mở rộng để lấy vị trí extrema chính xác hơn. Sau đó xét intensity nhỏ hơn ngưỡng 0.03 thì sẽ loại keypoint tiềm năng đi

DoG rất nhạy cảm với đường (edge), cần phải loại bỏ edge keypoint thì SIFT dùng ý tưởng giống Harris Corner Detector, dùng ma trận Hessian 2x2 tính ra đường cong chính. Với giá trị eigenvalue mà lớn hơn thresholder nào đó thì keypoint ở cạnh sẽ bị loại bỏ

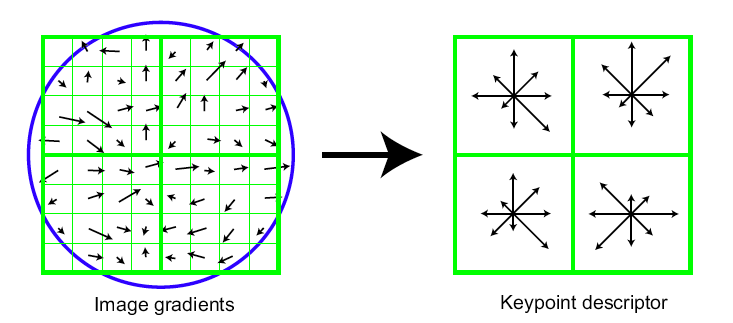
Từ đó SIFT giữ được những keypoint hợp lý

1. Orientation Assignment

Với từng vị trí keypoint tìm được sẽ gán cho một hoặc nhiều orientation (hướng) các phép tính sau đó trên ảnh sẽ tương ứng với orientation, scale và vị trí

1. Keypoint descriptor

Keypoint descriptor được tính bằng cách lấy 16x16 điểm liền kề (neighborhood) của keypoint đó, chia thành 16 sub-blocks với kích thước 4x4. 1 sub-block tạo ra 8 bin orientation vậy nên ta có 128 bin giá trị tương ứng là 1 vector biểu hiện cho keypoint descriptor



*Hình 2. Gradient (bên trái) và vector mô tả (bên phải)*

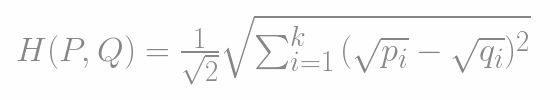
* 1. **Cải tiến rút trích đặc trưng ROOTSIFT**

RootSIFT là một phiên bản mở rộng của SIFT giúp tăng độ chính xác khi giải quyết các vấn đề cần áp dụng SIFT descriptor (chẳng hạn như Object Recognition) vì RootSIFT cho phép các Feature Vector được so sánh bằng Hellinger distance. Chính vì hiệu quả hơn nên RootSIFT được ưu tiên hơn SIFT.

Trong xác xuất thống kê, khoảng cách Hellinger được dùng để biểu diễn sự khác biệt giữa hai phân bố xác suất.

Giả sử cho hai tập hợp P = {p_1,p_2,...,p_k}; Q = {q_1,q_2,...,q_k}.

Khoảng cách Hellinger giữa hai tập hợp P và Q là:



d(P,Q) = \sqrt{\sum_{i=1}^{k}{(p_i-q_i)^2}} Vì khoảng cách Hellinger khá tương đồng với khoảng cách

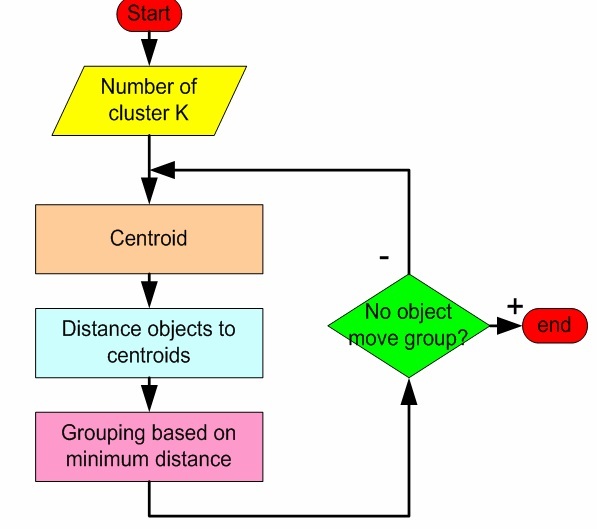
Euclidean nên chúng ta chỉ cần tinh chỉnh một chút ở các Feature Vector của các keypoint và vẫn có thể sử dụng SIFT descriptor như bình thường.

* 1. **Thuật toán K-Means**

K-Means là thuật toán rất quan trọng và được sử dụng phổ biến trong kỹ thuật phân cụm (clustering) . Ý tưởng chính của thuật toán K-Means là tìm cách phân nhóm các đối tượng (objects) đã cho vào K cụm (K là số các cụm được xác định trước, K nguyên dương) sao cho tổng bình phương khoảng cách giữa các đối tượng đến tâm nhóm (centroid) là nhỏ nhất.

Cụ thể bao gồm các bước:

* Bước 1: Chọn ngẫu nhiên K tâm (centroid) tương ứng với K cluster. Các cluster được đại diện bằng tâm (centroid)
* Bước 2: Tính toàn bộ khoảng cách từ đối tượng đến tâm (sử dụng độ đo Euclidean)
* Bước 3: Với các giá trị khoảng cách nhóm các đối tượng vào nhóm gần nhất
* Bước 4: Xác định lại tâm mới cho các nhóm
* Bước 5: Lặp lại bước 2 cho đến khi không có sự thay đổi nhóm giữa các đối tượng hoặc tâm của các Cluster sẽ không thay đổi



*Hình 3. Các bước thuật toán K-Means*

* 1. **Mô hình Bag of Visual Word**

Biểu diễn ảnh là một bước quan trọng trong phân loại ảnh trong đó sử dụng phương pháp để biểu diễn ảnh trong nhiều điều kiện khác nhau đó là sử dụng nét đặc trưng cục bộ (local feature). Mô hình được dựa trên một mô hình trong phân tích văn bản đó là (bag of word). Từ đó muốn áp dụng lên ảnh, ta cần phải định nghĩa được các “word” cho ảnh (được gọi là visual words của ảnh). Biểu diễn ảnh theo mô hình này ta cần thực hiện 3 bước chính:

* Bước 1: phát hiện và biểu diễn các nét đặc trưng cục bộ (local feature), bước này là toàn bộ quá trình của mục 3.1 Rút trích đặc trưng SIFT(scale-invariant feature transform)
* Bước 2: Xây dựng dictionary (từ điển) của các visual words
* Bước 3: Biểu diễn ảnh dưới dạng vector

Cụ thể

* Bước 1: Chúng ta đã tìm được các keypoint mang các thông tin về tọa độ, scale, orientation và descriptor
* Bước 2: Ta thiết lập các visual words từ các mô tả cục bộ (local feature) ở đây là các keypoint đã được tính. Sau đó ta sử dụng 1 thuật giải K-means được thực hiện trên các để thực hiện trên các vector mô tả để phân các vector SIFT vào các nhóm (cluster) và mỗi cluster này được gọi là 1 visual word một dictionary (từ điển)
* Bước 3: Mỗi vector mô tả hay là mỗi visual word sẽ được gán vào cluster gần nhất (khoảng cách mỗi vector đến các tâm của các cluster thì được định nghĩa trước đó – Kmeans). Tiếp theo, ảnh sẽ được biểu diễn bằng tần số của visual words.

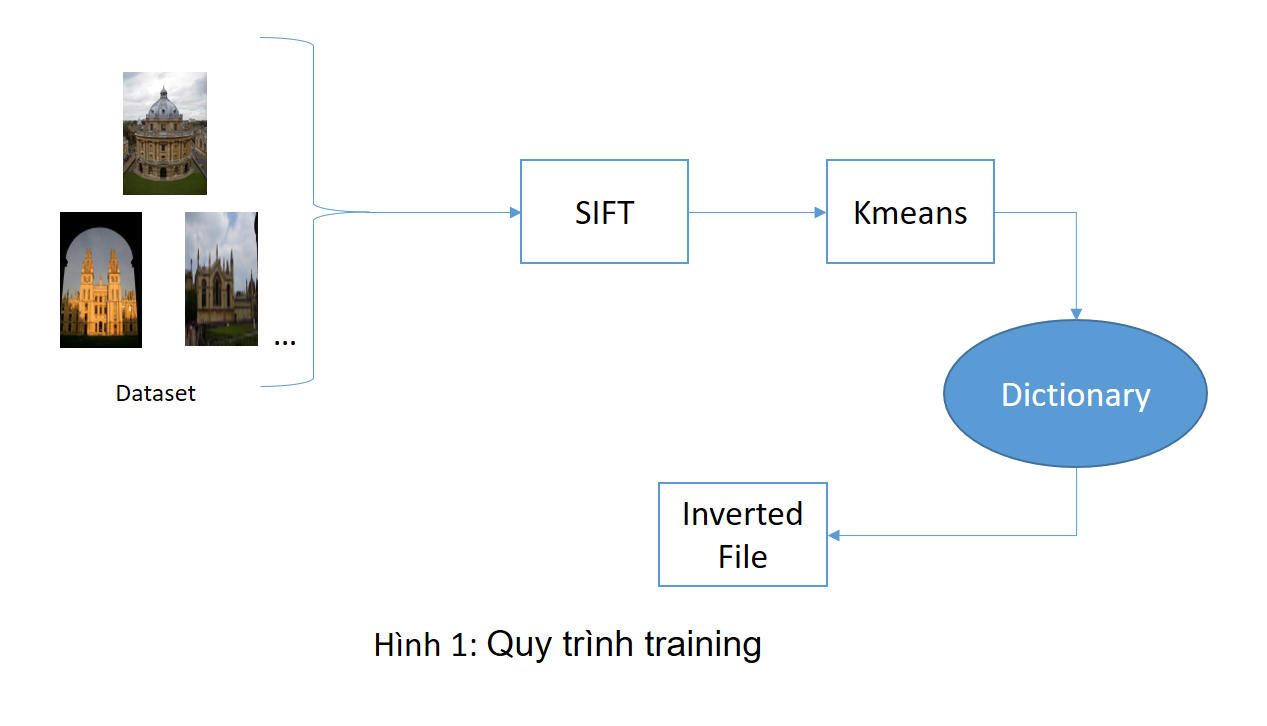
A close up of a device

Description generated with high confidence

*Hình 4. Tóm tắt mô hình Bag of Visual word*

* Phương pháp BOVW đã giải quyết tốt các nhược điểm về không gian lưu trữ các vectơ đặc trưng. Thay vì biểu diển ảnh bằng một tập các vector đặc trưng thì nay biểu diễn ảnh bằng một vector duy nhất, từ đó giảm không gian lưu trữ, tăng tốc độ truy vấn. Tạo tiền đề ứng dụng phương pháp này cho nhiều hệ thống xác thực người dùng dựa vào đặc trưng sinh trắc.

**2.5 Các bước thực hiện**

**\* Quy trình training.**

**Bước 1: Rút trích đặc trưng.**

* Tính đặc trưng SIFT trên các keypoint.
* Tính đặc trưng rootSIFT, rootSIFT = sqrt(sift / sum(sift) )
* Số chiều: 128.

**Bước 2: Xây dựng Dictionary.**

* Sử dụng thuật toán gôm cụm MiniBacth Kmeans Clustering.
* Số cụm: 1000.
* Số lần lặp: 10.

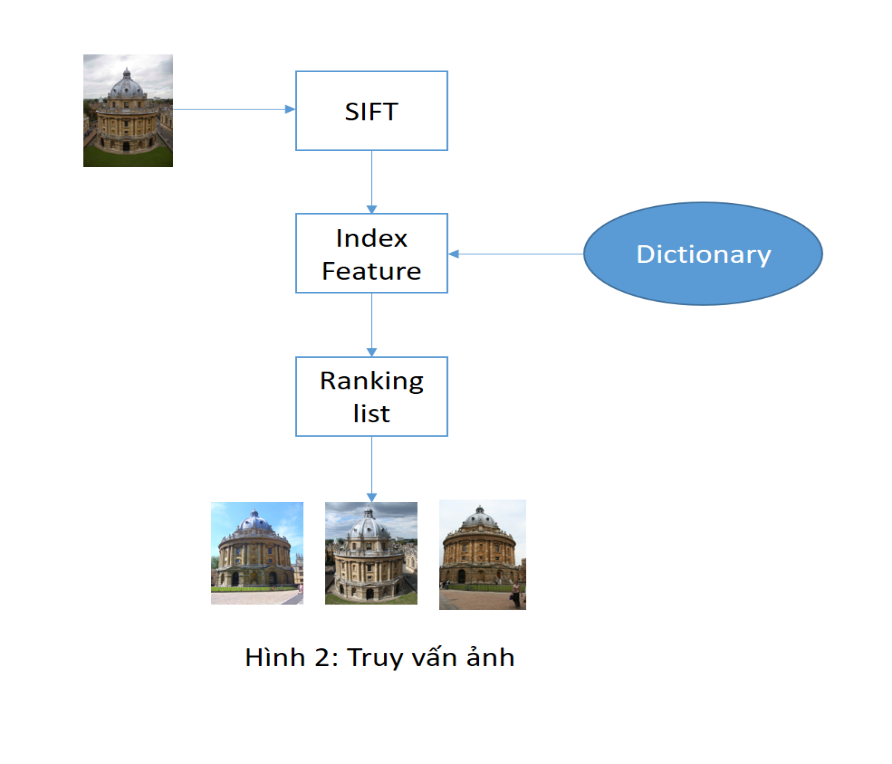
**Bước 3: Tính Index-feature cho từng ảnh.**

* Với mổi rootSIFT feature trong từng ảnh, ta tìm Index-feature dựa trên Dictionary

**Bước 4: Xây dựng Inverted File.**

* Theo thứ tự bình thường, với mỗi image, ta sẽ biết được trong image này có các feature nào.
* Inverted file: với mỗi feature, ta sẽ lưu danh sách những ảnh có chứa nó.
* ‘tf-idf’ weighting: các feature xuất hiện ở nhiều class of image thì càng ít có vai trò để phân loại một image nên được đánh trọng số thấp hơn. Các feature xuất hiện ở càng ít các class of image thì có trọng số cao hơn.

**\* Truy vấn ảnh.**



**Bước 5: Rút trích đặc trưng rootSIFT của bức ảnh ( giống bước 1)**

**Bước 6: Tính Index Feature cho từng feature trong ảnh ( giống bước 2).**

**Bước 7: Ranking list:**

* + Xây dựng bảng Feature Count để thống kê các feature và tần số xuất hiện tương ứng của nó.
  + Sử dụng Inverted File để so sánh Query Image với tất cả các image trong trong Inverted File -> List Score Distance.
  + Sắp xếp List Score giảm dần.

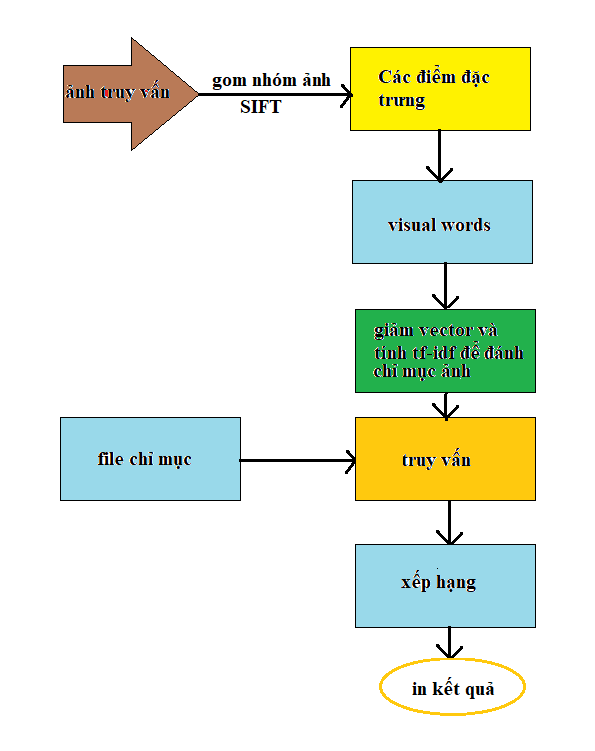
**Bước 8: Evaluation**

* Tập ảnh truy vấn gồm 55 ảnh khoanh vùng các đối tượng truy vấn chính
* Các ảnh thuộc tập ground truth đứng ở những vị trí đầu tiên của rank\_list thì độ chính xác càng cao và ngược lại.

**Bước 9: Geometric verification:**

* Xác thực hình dạng của vật thể trong các ảnh trả về so với ảnh truy vấn
* Tính số lượng inlier trong các ảnh trả về đó. Re-ranking list kết quả ảnh trả về Phân tích hệ thống truy vấn ảnh mẫu

# **3. Phân tích hệ thống truy vấn ảnh mẫu**

**3.1. Phân tích bài toán.**

Mỗi khi phát sinh một yêu cầu truy vấn ảnh thì hệ thống chỉ rút trích các đặc trưng của ảnh truy vấn và bắt đầu so sánh tỉ lệ với các vector được rút trích từ các ảnh mẫu. Hệ thống truy vấn này để xây dựng ứng dụng truy vấn ảnh bằng sử dụng thuật toán Bag of visual words để sử lí đưa hình ảnh về dạng vector so sánh sử dụng if-idf để xác định mức độ tương đồng của các vector khi so sánh.

**3.2. Phân tích cấu trúc dữ liệu.**

* Hệ thống truy vấn sử dụng rút trích đăc trưng SIFT để rút trính các đặt trưng ảnh rồi lưu vào CSDL để làm tiền đề cho việc truy vấn đặc trưng ảnh.
* Xây dựng CSDL các đặc trưng ảnh mẫu.
* Kiểm tra và khởi tạo lấy các giá trị để lưu vào bằng cách gọi đến các file images rồi xử lí các file ảnh để lưu các giá trị.
* Hàm im2single(rgb2gray(imread(imgPath)) lấy ảnh ra và chuyển ảnh thành ảnh xám sau đó chuyển sang ma trận các số.
* Hàm imresize chuyển các thành các ảnh nhỏ hơn
* Trích xuất các tính năng cục bộ của tất cả các tập dữ liệu hình ảnh.
* vl\_covdet (I, 'method', 'Hessian', 'estimateAffineShape', true) hàm này để rút trích đặc trưng SIFT.
* Lấy dữ liệu từ file AKM.
* Xây dựng hệ thống từ điểm truy vấn.
* Kiểm tra và khởi tạo các giá trị cho từ điển truy vấn.
* ccvBowGetDict hàm dùng để khởi tạo và lưu các words
* ccvBowGetWordsInit hàm dùng khởi tạo ban đầu cho hàm ccvBowGetWords hoạt động.
* ccvBowGetWords hàm cắt bớt số lượng words trong mảng bằng cách so sánh chúng với nhau.
* Đánh chỉ mục file.
* Sử dụng TF-IDF để xác định vector so sánh.
* Hình ảnh dùng truy vấn.
* Tải các hình ảnh truy vấn lên từ một file.
* Trích xuất các đặt trưng ảnh.
* So sánh ảnh với các vector.
* So sánh các ảnh có độ tương đồng giống nhau .
* In ra những ảnh có độ tương đồng nhiều nhất.
* Tính phần trăm độ tương đồng và kết thúc

**3.3. Tìm hiểu giải thuật.**

**3.3.1. Các bước giải thuật.**

Mở các ảnh mẫu trong tập dữ liệu ảnh

Rút trích đặc trưng SIFT của từng ảnh

Xây dựng từ điển

Đưa ảnh về dạng vector

Đánh chỉ mục cho từng ảnh

Mở ảnh cần truy vấn

Rút trích đặc trưng SIFT của ảnh truy vấn

Rút trích đặc trưng rootSIFT của ảnh truy vấn

Đưa ảnh truy vấn về dạng vector

Truy vấn ảnh bằng cách sử dụng độ đo Euclidean

**3.3.2. Vấn đế 1: Rút trích đặc trưng SIFT của từng ảnh**.

Chạy vòng lặp: i = 1 đến n (bức ảnh):

* Đọc ảnh dưới dạng ảnh grayscale.
* Rút trích đặc trưng SIFT sử dụng hàm vl\_covdet.
* Lưu thông tin đặc trưng SIFT vừa rút trích được vào mảng “features”
* Lưu thông tin số đặc trưng rút trích được trong bức ảnh i vào mảng “features\_per\_images “.

**3.3.3. Vấn đế 2: Rút trích đặc trưng rootSIFT của từng ảnh.**

Giải thuật của RootSIFT như sau:

- Trích xuất SIFT Feature Vectors cho tất cả keypoints, chẳng hạn như sử dụng function SIFT\_create() của OpenCV;

- Thực hiện L1-normalizing cho từng SIFT Feature Vector;

- Tính căn bậc hai cho từng phần tử trong SIFT Feature Vector đó.

**3.3.4. Vấn đề 2: Xây dựng từ điển.**

Sử dụng hàm ccvBowGetDict để lấy thông tin đặc trưng trong mảng “features”, sử dụng phương pháp gom nhóm k-mean và dựa vào đó, trả về words (1 từ là 1 cluster), danh sách các word này chính là từ điển, từ điển được lưu vào biến “dict\_words”

**3.3.5. Vấn đề 3: Đưa ảnh về dạng vector (sparse vector).**

Sử dụng hàm ccvBowGetWords để đưa thông tin các đặc trưng có trong mảng features về vector, mỗi vector tương ứng với 1 phần tử trong mảng “words”.

**3.3.6. Vấn đề 4: Đánh chỉ mục.**

Sử dụng hàm ccvInvFileInsert để thực hiện thao tác đánh chỉ mục.

Sử dụng hàm ccvInvFileCompStats để tính toán những chỉ số cần thiết cho việc tìm kiếm. Với một số điều kiện: sử dụng tf-idf weighting, chuẩn hóa các giá trị trong vector để tổng histogram = 1.

**3.3.7. Vấn đề 5: Truy vấn ảnh.**

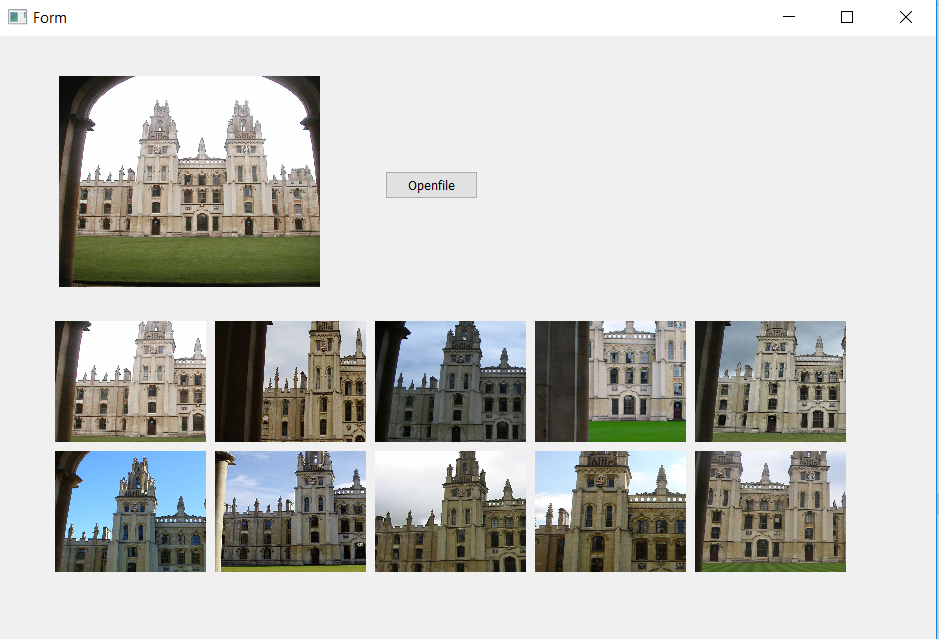
Sau khi đã rút trích đặc trưng của ảnh truy vấn và đưa ảnh truy vấn về vector. Sử dụng hàm ccvInvFileSearch để tiến hành truy vấn, với độ đo được sử dụng là Euclidean l1, chuẩn hóa các giá trị trong vector để tổng histogram = 1. Kết quả trả về sẽ là những ảnh gần giống với ảnh truy vấn nhất cùng với độ đo khoảng cách tương ứng.

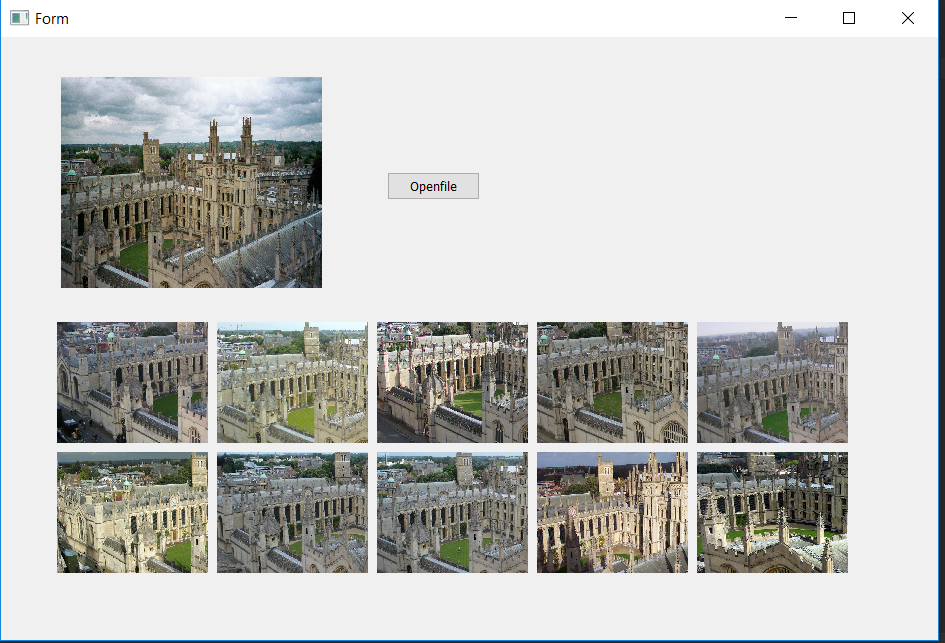
Danh sách ảnh kết quả từ liên quan nhất đến ít liên quan nhất được lưu vào biến “ids”, cùng với khoảng cách tương ứng giữa ảnh kết quả và ảnh truy vấn được lưu vào biến “dists”.

4. Chạy chương trình

Sau khi chạy chương trình, thực hiện trên ngôn ngữ lập trình python

Kết quả của chương trình đa phần cho ra kết quả ảnh mẫu tương tự với ảnh truy vấn ban đầu tuy nhiên vẫn còn nhiều bức hình vẫn chưa ra kết quả đúng chẳng hạn như bức hình dưới:





Hình truy vấn: ashmolean\_00058, ảnh mẫu bị sai: oxford\_01751.jpg, balliol\_00090.jpg,…

Ưu và nhược điểm

* Ưu điểm: Các kết quả đúng với đa phần ảnh, ít tốn không gian lưu trữ cũng như đánh chỉ mục, tốc độ là nhanh hơn so với các phương pháp lưu trữ vector thông thường
* Nhược điểm: Chưa thực sự chạy nhanh với tập dữ liệu lớn cũng như một số kết quả còn sai, độ chính xác còn thấp (pricision)

|  |  |  |
| --- | --- | --- |
| **STT** | **Query image** | **Accuracy** |
| 1 | all\_souls\_000013 | 1.0 |
| 2 | all\_souls\_000026 | 0,8946 |
| 3 | all\_souls\_000051 | 0.7536 |
| 4 | ashmolean\_000000 | 0.1643 |
| 5 | ashmolean\_000007 | 0.2972 |
| 6 | ashmolean\_000058 | 0.7477 |
| 7 | ashmolean\_000269 | 0.9666 |
| 8 | ashmolean\_000305 | 0.125 |
| 9 | balliol\_000051 | 0.8333 |
| 10 | balliol\_000167 | 0.6791 |
| 11 | balliol\_000187 | 1.0 |
| 12 | balliol\_000194 | 0.3333 |
| 13 | bodleian\_000107 | 0 |
| 14 | bodleian\_000108 | 0.8333 |
| 15 | bodleian\_000132 | 0.3666 |
| 16 | bodleian\_000163 | 0.8055 |
| 17 | bodleian\_000407 | 0.1111 |
| 18 | christ\_church\_000179 | 0.875 |
| 19 | christ\_church\_000999 | 0.7876 |
| 20 | christ\_church\_001020 | 0.6166 |
| 21 | cornmarket\_000019 | 0.8515 |
| 22 | cornmarket\_000047 | 0.7809 |
| 23 | cornmarket\_000105 | 0.4655 |
| 24 | cornmarket\_000131 | 0.5879 |
| 25 | hertford\_000015 | 0 |
| 26 | hertford\_000027 | 1.0 |
| 27 | hertford\_000063 | 0 |
| 28 | keble\_000028 | 1.0 |
| 29 | keble\_000055 | 1.0 |
| 30 | keble\_000214 | 0.95 |
| 31 | keble\_000227 | 0.8556 |
| 32 | keble\_000245 | 0.8894 |
| 33 | magdalen\_000058 | 0.8133 |
| 34 | magdalen\_000078 | 0.9022 |
| 35 | magdalen\_000560 | 0.6485 |
| 36 | oxford\_000317 | 1.0 |
| 37 | oxford\_000545 | 1.0 |
| 38 | oxford\_001115 | 0.9166 |
| 39 | oxford\_001752 | 0.7 |
| 40 | oxford\_001753 | 1.0 |
| 41 | oxford\_002416 | 1.0 |
| 42 | oxford\_002562 | 0 |
| 43 | oxford\_002734 | 0.9166 |
| 44 | oxford\_002904 | 0.125 |
| 45 | oxford\_002985 | 1.0 |
| 46 | oxford\_003335 | 0.8333 |
| 47 | oxford\_003410 | 1.0 |
| 48 | pitt\_rivers\_000033 | 1.0 |
| 49 | pitt\_rivers\_000058 | 0.5 |
| 50 | pitt\_rivers\_000087 | 0.5833 |
| 51 | pitt\_rivers\_000119 | 0.8512 |
| 52 | pitt\_rivers\_000153 | 0.5420 |
| 53 | radcliffe\_camera\_000095 | 0.8469 |
| 54 | radcliffe\_camera\_000519 | 0.3840 |
| 55 | radcliffe\_camera\_000523 | 0.7807 |

5. Cải tiến truy vấn.

Mục tiêu của cải tiến truy vấn là để tăng tốc độ truy vấn bằng hình ảnh của mô hình này với tập dữ liệu với kích thước lớn hoặc rất lớn. Có 2 phương pháp cải tiến truy vấn (improve retrieval) đó là:

* Phương pháp mở rộng câu truy vấn riêng biệt

- Database-side feature augmentation (sử dụng đồ thị)

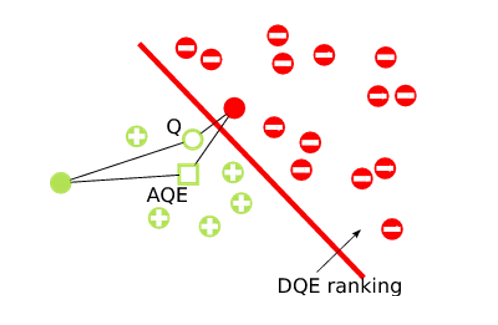
**5.1. Mở rộng câu truy vấn riêng biệt – Discriminative Query Expansion (DQE)**

* Các vector của Bag of Visual Words trong không gian được xây dựng nên mô hình truy vấn đa dạng hơn (riêng biệt)
* Average Query Expansion (AQE)

+ Được tính bằng cách lấy trung bình của các vector Bag of Visual Word và truy vấn lại

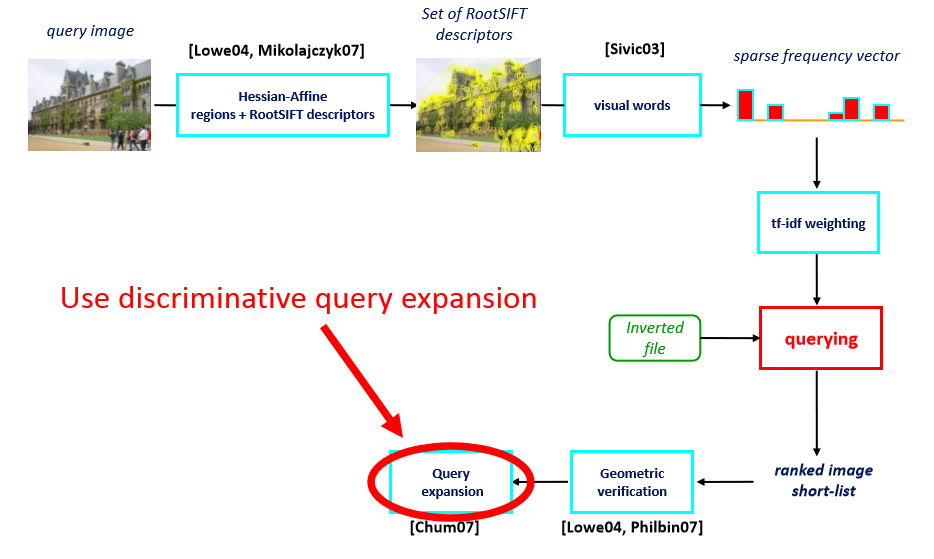
+ Đặc điểm là tốc độ cũng tương tự nhưng 1 số trường hợp còn chạy nhanh hơn

* Để mở rộng câu truy vấn riêng biệt cần thực hiện như sau:
* Sử dụng câu truy vấn mở rộng của vector Bag of Visual Words vừa tính ở AQE gán nhãn là positive
* Đối với những bức ảnh có rank thấp thì sẽ được gán là negative
* Tính khoảng từ vector đến các ảnh, sau đó chia ra 2 phần được ngăn cách bởi đường dicision boundary



*Quá trình gán nhãn cho các ảnh training*

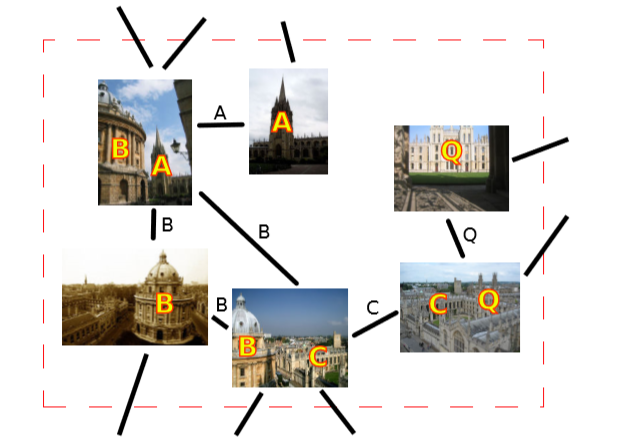
* Sắp hạng lại các ảnh theo cấu trúc chỉ mục ngược
* Đặc điểm của mô hình này là Avarage Query Expansion thì vector BoW có trọng số trung bình, còn đối với Discriminative Query Expansion thì các vector đã được training dự liệu và khoảng thời gian tính toán là không đáng kể
* Mô hình này gia tăng đáng kể về hiệu năng cũng như không tốn thêm chi phí



*Sử dụng truy vấn mở rộng riêng biệt cho mô hình BoW*

**5.2. Database-side feature augmentation**

* Mô hình này được biểu diễn dưới dạng đồ thị
* Với Đỉnh: hình ảnh
* Cạnh sẽ nối các hình ảnh với các Object giống nhau lại
* Tính toán trên đồ thị thông qua hệ thống truy vấn trên từng ảnh cơ sở và lần lượt lưu lại các hình ảnh trong không gian
* Ở đây mỗi hình sẽ được tăng cường(Augment) bởi các ảnh lân cận
* Sắp xếp lại các hình theo thứ tự
* Độ hiệu quả là 28% so với phương pháp gốc, cải thiện được chỉ số Recall tuy nhiên Pricision bị giảm đi



*Đồ thị ảnh*

6. Phân công

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên** | **Công việc được giao** | **Mức độ hoàn thành (%)** |
| 1 | Hiển Trương | Báo cáo phần 4.3, 4.4 | 100% |
| 2 | Lợi Trương | Báo cáo phần 3, 5, 6 | 100% |
| 3 | Trương | Báo cáo phần 4.1 , 4.2, 4.4 | 100% |

7. Tài liệu tham khảo

* Bài báo “Three things everyone should know to improve object retrieval” R.Arandjelovic:

<https://www.robots.ox.ac.uk/~vgg/publications/2012/Arandjelovic12/arandjelovic12.pdf>

* Đặc trưng SIFT:

<https://viblo.asia/p/gioi-thieu-ve-scale-invariant-feature-transform-z3NVRkoLR9xn>

* K-means clustering:

<https://en.wikipedia.org/wiki/K-means_clustering>

* Bag of Visual Words:

<http://www.robots.ox.ac.uk/~az/icvss08_az_bow.pdf>

* <https://github.com/mohamedadaly/caltech-image-search>
* [*https://quyv.wordpress.com/2016/07/24/bag-of-word-model/*](https://quyv.wordpress.com/2016/07/24/bag-of-word-model/)
* [*https://machinelearningcoban.com/2017/01/01/kmeans/*](https://machinelearningcoban.com/2017/01/01/kmeans/)
* [*https://kipalog.com/posts/Gioi-thieu-giai-thuat-SIFT-de-nhan-dang-anh*](https://kipalog.com/posts/Gioi-thieu-giai-thuat-SIFT-de-nhan-dang-anh)
* [*https://viblo.asia/p/gioi-thieu-ve-scale-invariant-feature-transform-z3NVRkoLR9xn*](https://viblo.asia/p/gioi-thieu-ve-scale-invariant-feature-transform-z3NVRkoLR9xn)
* [*https://machinelearningcoban.com/2017/01/01/kmeans/#tom-tat-thuat-toan*](https://machinelearningcoban.com/2017/01/01/kmeans/)

**Nhận xét của giảng viên hướng dẫn:**

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................