

PHẦN A: GIỚI THIỆU

LỜI CẢM ƠN

Trước hết em xin gửi lời cảm ơn sâu sắc đến thầy Lê Minh Thành, người đã giúp đỡ em rất nhiều về định hướng nghiên cứu, hướng dẫn cho em trong suốt thời gian thực hiện đề tài này.

Cuốn đồ án này được hoàn thành theo đúng thời gian quy định của nhà trường cũng như của khoa không chỉ là sự nỗ lực của em mà còn sự giúp đỡ, chỉ bảo của thầy hướng dẫn, của quý thầy cô và các bạn sinh viên.

Chúng em xin chân thành cảm ơn thầy cô đã giảng dạy chúng em, đặc biệt là các thầy cô giáo trong khoa Điện-Điện tử.

Xin cảm ơn các bạn sinh viên trong khoa đã giúp đỡ tôi rất nhiều mặt: như phương tiện, sách vở, ý kiến ...

Mặc dù đã rất cố gắng hoàn thành đồ án này song cung không tránh khỏi những sai sót, mong thầy cô và các bạn đóng góp những ý kiến quý báu để đồ án được thành công hơn.

Sinh viên thực hiện

Võ Hồng Hoan

LỜI MỞ ĐẦU

Hơn một thập kỷ qua có rất nhiều công trình nghiên cứu về bài toán nhận dạng khuôn mặt người từ ảnh đen trắng, xám đến ảnh màu như ngày hôm nay. Các nghiên cứu đi từ bài toán đơn giản, mỗi ảnh chỉ có một khuôn mặt người nhìn thẳng vào thiết bị thu hình và đầu ở tư thế thẳng đứng trong ảnh đen trắng. Cho đến ngày hôm nay bài toán mở rộng cho ảnh màu, có nhiều khuôn mặt trong cùng một ảnh, có nhiều tư thế thay đổi trong ảnh. Không những vậy mà còn mở rộng cả phạm vi từ môi trường xung

quanh khá đơn giản cho đến môi trường xung quanh rất phức tạp nhằm đáp ứng nhu cầu của con người.

Mục tiêu của đề tài “ Nhận dạng mặt người trên matlab” là thực hiện chương trình tìm kiếm một bức ảnh có khuôn mặt một người trong tập ảnh cơ sở giống với khuôn mặt của người trong bức ảnh cần kiểm tra bằng ngôn ngữ matlab.

Để tiện theo dõi tôi xin trình bày đề tài theo ba phần như sau:

- Phần đầu là giới thiệu về thuật toán PCA là ứng toán được sử dụng rất nhiều trong viễn thông. Và đề tài này sử dụng thuật toán PCA.
- Phần tiếp theo là giới thiệu các lệnh được sử dụng trong chương trình.
- Phần cuối cùng là giới thiệu giao diện chương trình và code nguồn.

Do tài liệu tham khảo hạn chế, trình độ có hạn và kinh nghiệm trong thực tiễn còn non kém, nên đề tài không tránh khỏi những thiếu sót. Rất mong được nhận những ý kiến đóng góp, giúp đỡ chân tình, quý báu của quý thầy cô cùng các bạn sinh viên.

Tp. Hồ Chí Minh, tháng 06 năm 2010

Người thực hiện đề tài

MỤC LỤC

PHẦN A: GIỚI THIỆU

<i>LỜI CẢM ƠN</i>	ii
<i>LỜI MỞ ĐẦU</i>	iii
MỤC LỤC	iv
LIỆT KÊ HÌNH.....	vi
LIỆT KÊ BÀNG	vii

PHẦN B: NỘI DUNG**CHƯƠNG 1: DẪN NHẬP**

1.1 Đặt vấn đề.....	3
1.2 Lý do chọn đề tài.....	3
1.3 Mục đích nghiên cứu.....	3
1.4 Giới hạn nghiên cứu của đề tài	3

CHƯƠNG 2: CÁC THUẬT TOÁN NHẬN DẠNG KHÔN MẶT

2.1 Định nghĩa bài toán xác định khuôn mặt người	6
2.2 Ứng dụng của phương pháp xác định khuôn mặt người.....	6
2.3 Phương pháp xác định khuôn mặt người	7
2.4 Nhận dạng khuôn mặt dùng thuật toán PCA	8
2.5 Nhận dạng ảnh dựa trên PCA	8

CHƯƠNG 3: ẢNH MÀU TRÊN MATLAB VÀ CÁC LỆNH XỬ LÝ ẢNH

3.1 Giới thiệu ảnh số	14
3.1.1 Biểu diễn ảnh số.....	14
3.1.2 Ảnh màu	14
3.1.3 Các định dạng ảnh cơ bản trong xử lý ảnh	16
3.2 Các kiểu hình ảnh trong Matlab.....	18
3.3 Chuyển đổi giữa các kiểu dữ liệu.....	19
3.4 Các phép toán số học cơ bản đối với dữ liệu ảnh.....	20
3.5 Các hàm hiển thị ảnh trong Matlab	20
3.6 Các hàm khác được sử dụng trong đề tài.....	22

CHƯƠNG 4: GIỚI THIỆU CHƯƠNG TRÌNH

4.1 Giới thiệu chương trình	26
-----------------------------------	----

CHƯƠNG 5: SƠ ĐỒ KHỐI VÀ CODE CHƯƠNG TRÌNH

5.1 Sơ đồ khối.....	32
5.2 Code chương trình.....	32

CHƯƠNG 6: PHẠM VI GIỚI HẠN VÀ HƯỚNG MỞ RỘNG ĐỀ TÀI

6.1 Phạm vi giới hạn của đề tài.....	42
6.2 Hướng mở rộng của đề tài	42

LIỆT KÊ HÌNH

Hình 3.1: Ảnh màu.....	14
Hình 3.2: Các màu cơ sở.....	15
Hình 3.3: Mô hình màu RGB.....	16
Hình 3.4: Ảnh GIF.....	18
Hình 3.5: Ảnh dạng JPEG.....	20
Hình 4.1: Mở chương trình trên Matlab.....	26
Hình 4.2: Giao diện chương trình.....	27
Hình 4.3: Giao diện chương trình chính.....	27
Hình 4.4: Chọn ảnh cần kiểm tra.....	28
Hình 4.5: Ảnh cần kiểm tra.....	28
Hình 4.6: Ảnh trung bình.....	29
Hình 4.7: Hình chiếu ảnh lên không gian ảnh.....	29
Hình 4.8: Ảnh cần tìm.....	30
Hình 5.1: Sơ đồ khối tổng quát của chương trình.....	32

LIỆT KÊ BÀNG

Bảng 3.1: Các thông tin khi gọi hàm imfinfo.....	21
Bảng 3.2 Các phép toán số học trên ảnh.....	22
Bảng 3.3 Các hàm xử lý hình ảnh khác trong Matlab.....	23

PHẦN B: NỘI DUNG

CHƯƠNG 1 DẪN NHẬP

1.1 Đặt vấn đề

Chúng ta đã biết, ngày nay phần lớn các thiết bị điện tử đều dần phát triển theo xu hướng tự động hóa, thông minh, càng hiểu ý con người, chúng giao tiếp với con người mà không cần một thiết bị trung gian nào, để làm được điều đó các thiết bị cảm biến, thuật toán nhận dạng ra đòi ngày càng hiện đại hơn, chính xác hơn, an toàn và rất bảo mật, chúng có thể chúng nhận biết các hoạt động của con người, hình gián của con người và hoạt động theo ý muốn con người. Thì bài toán “Nhận dạng mặt người” là một trong số đó.

1.2 Lý do chọn đề tài

Ngày nay các thiết bị sử dụng thuật toán xử lý ảnh được sử dụng ngày càng rộng rãi, với nhiều mục đích khác nhau. Dùng cho các hệ thống bảo mật như khóa bằng vân tay, giọng nói, giác mạc mắt đến các thiết bị an ninh, truy tìm tội phạm..

Xuất phát từ những yêu cầu thực tế trên người thực hiện tiến hành tìm hiểu và nghiên cứu đề tài : “NHẬN DẠNG MẶT NGƯỜI TRÊN MATLAB”.

1.3 Mục đích nghiên cứu

Người thực hiện đề tài này nhằm mục đích:

- Tìm hiểu các thuật toán nhận dạng và xử lý ảnh màu, cấu trúc ảnh màu.
- Nâng cao kỹ năng thiết kế và lập trình bằng ngôn ngữ matlab.
- Rèn luyện kỹ năng nghiên cứu, tìm hiểu tài liệu.

1.4 Giới hạn nghiên cứu của đề tài

Với thời gian có hạn nên người nghiên cứu chỉ thực hiện nghiên cứu những vấn đề cơ bản sau:

- Nghiên cứu và tiềm hiểu các thuật toán nhận dạng, mà cụ thể là thuật toán PCA.
- Nghiên cứu cấu trúc ảnh màu, các lệnh xử lý ảnh màu trên matlab 7.0.
- Nghiên cứu giải thuật và thực hiện phần mềm nhận dạng trên matlab 7.0.

CHƯƠNG 2
CÁC THUẬT TOÁN
NHẬN DẠNG KHUÔN MẶT

2.1 Định nghĩa bài toán xác định khuôn mặt người

Xác định khuôn mặt người (Face Detection) là một kỹ thuật máy tính để xác định các vị trí và các kích thước của các khuôn mặt người trong các ảnh bất kỳ (ảnh kỹ thuật số). Kỹ thuật này nhận biết các đặc trưng của khuôn mặt và bỏ qua những thứ khác, như: tòa nhà, cây cối, cơ thể..

2.2 Ứng dụng của phương pháp xác định khuôn mặt người

Có nhiều ứng dụng đã được và đang thiết kế, tôi chỉ xin đưa ra một số loại ứng dụng sau:

- Hệ thống tương tác giữa người và máy: giúp những người bị tật hoặc khiếm khuyết có thể trao đổi. Những người dùng ngôn ngữ tay có thể giao tiếp với những người bình thường. Những người bị bại liệt thông qua một số ký hiệu nháy mắt có thể biểu lộ những gì họ muốn, .. Đó là các bài toán điều bộ của bàn tay (hand gesture), điều bộ khuôn mặt.
- Nhận dạng người A có phải là tội phạm truy nã hay không? Giúp cơ quan an ninh quản lý tốt con người. Công việc nhận dạng có thể ở trong môi trường bình thường cũng như trong bóng tối (sử dụng camera hồng ngoại).
- Hệ thống quan sát, theo dõi và bảo vệ. Các hệ thống camera sẽ xác định đâu là con người và theo dõi con người đó xem họ có vi phạm gì không, ví dụ xâm phạm khu vực không được vào, ..
- Lưu trữ (rút tiền ATM, để biết ai rút tiền vào thời điểm đó), hiện nay có tình trạng những người bị người khác lấy mất thẻ ATM hay mất mã số PIN và những người ăn cắp này đi rút tiền, hoặc những người chủ thẻ đi rút tiền nhưng lại báo cho ngân hàng là mất thẻ và mất tiền. Các ngân hàng có nhu cầu khi có giao dịch tiền sẽ kiểm tra hay lưu trữ khuôn mặt người rút tiền để sau đó đối chứng và xử lý.
 - Thẻ căn cước, chứng minh nhân dân (Face Identification)
 - Điều khiển vào ra: văn phòng, công ty, trụ sở, máy tính, Palm, .. Kết hợp thêm vân tay và mống mắt. Cho phép nhân viên được ra vào nơi cần thiết, hay mỗi người sẽ đăng nhập máy tính cá nhân của mình mà không cần nhớ tên đăng nhập cũng như mật khẩu mà chỉ cần xác định thông qua khuôn mặt.
 - An ninh sân bay, xuất nhập cảnh (hiện nay cơ quan xuất nhập cảnh Mỹ đã áp dụng). Dùng để xác thực người xuất nhập cảnh và kiểm tra có phải là nhân vật khủng bố không.
 - Tương lai sẽ phát triển các loại thẻ thông minh có tích hợp sẵn đặc trưng của người dùng trên đó, khi bắt cứ người dùng khác dùng để truy cập hay xử lý tại các hệ thống sẽ được yêu cầu kiểm tra các đặc trưng khuôn mặt so với thẻ để biết nay có phải là chủ thẻ hay không.
 - Tìm kiếm và tổ chức dữ liệu liên quan đến con người thông qua khuôn mặt người trên nhiều hệ cơ sở dữ liệu lưu trữ thật lớn, như internet, các hãng truyền hình, Ví dụ: tìm các đoạn video có tổng thống Bush phát biểu, tìm các phim có diễn viên Lý Liên Kiệt đóng, tìm các trận đá banh có Ronaldo đá, ..

- Hiện nay có nhiều hướng tiếp cận để xác định một ảnh có phải là ảnh khuôn mặt hay không? Khuôn mặt người được xem như một yếu tố để xác định cho một hướng tiếp cận mà được dùng gần đây.
 - Ứng dụng trong video phone.
 - Phân loại trong lưu trữ hình ảnh trong điện thoại di động. Thông qua bài toán xác định khuôn mặt người và trích đặc trưng, rồi dựa vào đặc trưng này để sắp xếp lưu trữ, giúp người sử dụng dễ dàng truy tìm khi cần thiết.
 - Kiểm tra trạng thái người lái xe có ngủ gật, mất tập trung hay không, và hỗ trợ thông báo khi cần thiết.
 - Phân tích cảm xúc trên khuôn mặt.
 - Trong lãnh vực thiết kế điều khiển robot.
 - Hàng máy chụp hình Canon đã ứng dụng bài toán xác định khuôn mặt người vào máy chụp hình thế hệ mới để cho kết quả hình ảnh đẹp hơn, nhất là khuôn mặt người.

2.3 Phương pháp xác định khuôn mặt người

Có nhiều nghiên cứu tìm phương pháp xác định khuôn mặt người, từ ảnh xám đến ngày nay là ảnh màu. Tôi sẽ trình bày một cách tổng quát nhất những hướng giải quyết chính cho bài toán, từ những hướng chính này nhiều tác giả thay đổi một số ý nhỏ bên trong để có kết quả mới.

Dựa vào tính chất của các phương pháp xác định khuôn mặt người trên ảnh. Các phương pháp này được chia làm bốn hướng tiếp cận chính. Ngoài bốn hướng này, nhiều nghiên cứu có khi liên quan đến không những một hướng tiếp cận mà có liên quan nhiều hơn một hướng chính:

- **Hướng tiếp cận dựa trên tri thức:** Mã hóa các hiểu biết của con người về các loại khuôn mặt người thành các luật. Thông thường các luật mô tả quan hệ của các đặc trưng.
- **Hướng tiếp cận dựa trên đặc trưng không thay đổi:** Mục tiêu các thuật toán đi tìm các đặc trưng mô tả cấu trúc khuôn mặt người mà các đặc trưng này sẽ không thay đổi khi tư thế khuôn mặt, vị trí đặt thiết bị thu hình hoặc điều kiện ánh sáng thay đổi.
- **Hướng tiếp cận dựa trên so khớp mẫu:** Dùng các mẫu chuẩn của khuôn mặt người (các mẫu này được chọn lựa và lưu trữ) để mô tả cho khuôn mặt người hay các đặc trưng khuôn mặt (các mẫu này phải chọn làm sao cho tách biệt nhau theo tiêu chuẩn mà các tác giả định ra để so sánh). Các mối tương quan giữa dữ liệu ảnh đưa vào và các mẫu dùng để xác định khuôn mặt người.
- **Hướng tiếp cận dựa trên diện mạo:** Trái ngược hẳn với so khớp mẫu, các mô hình (hay các mẫu) được học từ một tập ảnh huấn luyện trước đó. Sau đó hệ thống (mô hình) sẽ xác định khuôn mặt người. Hay một số tác giả còn gọi hướng tiếp cận này là hướng tiếp cận theo phương pháp học.

2.4 Nhận dạng khuôn mặt dùng thuật toán PCA

Kohonen đã đưa ra phương pháp dùng vector riêng để nhận dạng khuôn mặt, ông dùng một mạng neural đơn giản để chứng tỏ khả năng của phương pháp này trên các ảnh đã được chuẩn hóa. Mạng neural tính một mô tả của khuôn mặt bằng cách xấp xỉ các vector riêng của ma trận tương quan của ảnh. Các vector riêng sau này được biết đến với cái tên Eigenface. Kirby và Sirovich chứng tỏ các ảnh có các khuôn mặt có thể được mã hóa tuyến tính bằng một số lượng vừa phải các ảnh cơ sở. Tính chất

này dựa trên biến đổi Karhunen-Loeve, mà còn được gọi dưới một cái tên khác là PCA và biến đổi Hotelling. Ý tưởng này được xem là của Pearson trình bày đầu tiên vào năm 1901 và sau đó là Hotelling vào năm 1933. Cho một tập các ảnh huấn luyện có kích thước $n \times m$ được mô tả bởi các vector có kích thước $m \times m$, các vector cơ sở cho một không gian con tối ưu được xác định thông qua lỗi bình phương trung bình khi chiếu các ảnh huấn luyện vào không gian con này. Các tác giả gọi tập các vector cơ sở tối ưu này là ảnh riêng sau đó gọi cho đơn giản là vector riêng của ma trận hiệp phương sai được tính từ các ảnh khuôn mặt đã vector hóa trong tập huấn luyện. Nếu cho 100 ảnh, mà mỗi khuôn mặt có kích thước 91×50 thì có thể chỉ dùng 50 ảnh riêng, trong khi vẫn duy trì được một khả năng giống nhau hợp lý (giữ được 95% tính chất).

Turk và Pentland áp dụng PCA để xác định và nhận dạng khuôn mặt. Tương tự, dùng PCA trên tập huấn luyện ảnh các khuôn mặt để sinh các ảnh riêng (còn gọi là eigenface) để tìm một không gian con (không gian khuôn mặt) trong không gian ảnh. Các ảnh khuôn mặt được chiếu vào không gian con này và được gom nhóm lại. Tương tự các ảnh không có khuôn mặt dùng để huấn luyện cũng được chiếu vào cùng không gian con và gom nhóm lại. Các ảnh khi chiếu vào không gian khuôn mặt thì không bị thay đổi tính chất cơ bản, trong khi chiếu các ảnh không có khuôn mặt thì xuất hiện sự khác nhau cũng không ít. Xác định sự có mặt của một khuôn mặt trong ảnh thông qua tất cả khoảng cách giữa các vị trí trong ảnh và không gian ảnh. Khoảng cách này dùng để xem xét có hay không có khuôn mặt người, kết quả khi tính toán các khoảng cách sẽ cho ta một bản đồ về khuôn mặt. Có thể xác định được từ cực tiểu địa phương của bản đồ này. Có nhiều nghiên cứu về xác định khuôn mặt, nhận dạng, và trích đặc trưng từ ý tưởng vector riêng, phân rã, và gom nhóm. Sau đó Kim phát triển cho ảnh màu, bằng cách phân đoạn ảnh để tìm ứng để không gian tìm kiếm bớt đi.

2.5 Nhận dạng ảnh dựa trên PCA

Khuôn mặt con người có rất nhiều nét để nhận biết, nếu như ta gặp lại một người bạn sau một thời gian dài, ta có thể nhận ra ngay người đó dù những chi tiết cụ thể trên mặt có thể thay đổi như da, mái tóc. Ta nhận ra không phải vì nhớ đôi mắt, hay mũi hay môi hay tóc, lông mày người đó mà ta nhận ra vì nhớ diện mạo của người đó. Tức là trên khuôn mặt tồn tại một nét tổng thể nào đó để có thể nhận diện, thuật toán của ta bắt đầu từ ý tưởng này.

Chương 2: Thuật toán nhận dạng khuôn mặt

Phân tích thành phần chính (Principal Component Analysis) gọi tắt là PCA là thuật toán nhận dạng ảnh dựa trên những nét tổng thể của khuôn mặt, ta sẽ áp dụng thuật toán này để thực hiện hai công việc sau :

- Thứ nhất là tìm một khuôn mặt giống với khuôn mặt cho trước
- Thứ hai là xác định vị trí những khuôn mặt người trong một bức ảnh.

Ban đầu ta có một tập ảnh khuôn mặt gọi là tập ảnh huấn luyện (training set). Giả sử mỗi ảnh có kích thước $M \times N$, ta coi mỗi bức ảnh này là một vector trong không gian $M \times N$ chiều. Bây giờ mỗi khuôn mặt là một vector, ta thấy những vector này không phân bố ngẫu nhiên trong không gian ảnh mà phân bố theo một quy luật tương đối nào đó, ta có thể nói những vector này nằm trong một không gian con gọi là không gian khuôn mặt. Từ những vector trong tập huấn luyện, ta sẽ tìm một cơ sở trực chuẩn cho không gian khuôn mặt. Những vector thuộc cơ sở này có thể coi là những vector mang những nét tổng thể đặc trưng về khuôn mặt.

Giả sử tập huấn luyện có P ảnh, khi đó ta sẽ có P vector : $\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_P$.

$$\mathbf{m} = \frac{1}{P} \sum_{i=1}^P \mathbf{T}_i$$

Tính vector ảnh trung bình : $m = \frac{1}{P} \sum_{i=1}^P \mathbf{T}_i$.

Sự khác biệt giữa những khuôn mặt với ảnh trung bình là những vector :

$$\mathbf{A}_i = \mathbf{T}_i - \mathbf{m}, i=1 \dots P$$

Ý tưởng của việc phân tích thành phần chính là tìm một tập những vector trực chuẩn \mathbf{u}_k sao cho những vector này mô tả tốt nhất sự phân bố những vector khuôn mặt trong không gian. Những vector \mathbf{u}_k được chọn sao cho :

$$\begin{aligned} - \langle \mathbf{u}_i | \mathbf{u}_j \rangle &= \begin{cases} 1 & \text{nếu } i = j \\ 0 & \text{nếu } i \neq j \end{cases} \\ - \lambda_k &= \sum_{i=1}^P \langle \mathbf{u}_k | \mathbf{A}_i \rangle^2 \text{ lớn nhất.} \end{aligned}$$

Những vector \mathbf{u}_k và giá trị vô hướng λ_k chính là những vector riêng và trị riêng tương ứng của ma trận \mathbf{AA}^T .

$\langle \mathbf{u} | \mathbf{v} \rangle$ là tích vô hướng giữa hai vector \mathbf{u}, \mathbf{v} . $A = [\mathbf{A}_1 \ \mathbf{A}_2 \ \dots \ \mathbf{A}_P]$

Ta thấy ma trận A có kích thước $M \times N \times P$, còn ma trận \mathbf{AA}^T có kích thước $M \times N \times M \times N$, do kích thước ma trận này quá lớn nên ta không thể tìm được những vector riêng và những trị riêng trực tiếp được, thay vào đó ta sẽ tìm những vector riêng của ma trận $\mathbf{A}^T \mathbf{A}$ có kích thước $P \times P$.

Nếu v là một vector riêng của $\mathbf{A}^T \mathbf{A}$ và λ là trị riêng tương ứng, khi đó ta có :

Chương 2: Thuật toán nhận dạng khuôn mặt

$A^T A v = \lambda v \Leftrightarrow AA^T A v = \lambda Av$, tức là Av là một trị riêng của ma trận AA^T .

Thông thường ta chỉ lấy một số Q vector riêng ứng với Q trị riêng có giá trị lớn nhất.

Sau khi có các vector riêng của ma trận AA^T , ta sẽ chuẩn hóa chúng để thu được một cơ sở trực chuẩn của không gian khuôn mặt.

Đặt $L=AA^T$, tìm V là tập hợp các vector riêng của L , D là tập hợp các trị riêng tương ứng.

V bao gồm Q vector riêng ứng với những trị riêng lớn hơn một giá trị nào đó hoặc ứng với Q trị riêng lớn nhất trong D .

$E = AV$ là tập các vector riêng của AA^T . Do đây là những vector riêng, mà nó lại có dạng khuôn mặt nên còn được gọi là Eigenfaces. E là ma trận $M*N \times Q$, mỗi cột là một vector riêng.

Chuẩn hóa các vector cột trong E (chia mỗi vector cho độ dài của vector đó).

Bây giờ, ta có thể coi E là một cơ sở trực chuẩn của không gian khuôn mặt.

Với H là bức ảnh có cùng kích thước với những bức ảnh trong tập huấn luyện. Ta sẽ xét nó có phải là bức ảnh khuôn mặt hay không, cũng như tìm bức ảnh giống với nó nhất trong tập huấn luyện.

H được xem là một vector trong không gian $M*N$ chiều.

Đặt $K=H-m$ với m là vector ảnh trung bình.

Cho V là một không gian có tích vô hướng hữu hạn chiều và W là một không gian con của V .

Giả sử W có một cơ sở trực chuẩn là $\{u_1, \dots, u_Q\}$. Khi đó hình chiếu trực giao của vector u bất kỳ lên W được xác định như sau :

$$pr_W u = u_0 + \sum_{i=1}^Q (u|u_i) u_i$$

Độ dài $\|u - u_0\|$ được gọi là khoảng cách từ u đến W .

Tập hợp $c_i = \|u|u_i\|$, $i=1, \dots, Q$ được gọi là tọa độ của u_0 trong không gian W .

Tìm $C=E^T K$ là tọa độ của hình chiếu K_f của K lên không gian khuôn mặt. C là vector cột $Q \times 1$

$$K_f = \sum_{i=1}^Q c_i \cdot e_i \quad \text{với } c_i = C(i, 1); e_i = E(:, i).$$

Với A_i là một cột trong ma trận A (tương ứng với bức ảnh T_i trong tập huấn luyện). Ta tính

$C_i = E^T A_i$ là tọa độ của hình chiếu $A_i f$ của A_i lên không gian khuôn mặt.

Ta tính hai đại lượng sau :

- $s = \|K - K_f\|$ xem như khoảng cách từ bức ảnh H đến không gian mặt

- $s_i = \|C - C_i\|$ xem như khoảng cách từ H đến bức ảnh T_i trong tập huấn luyện

Xét α và β là hai ngưỡng nào đó.

Chương 2: Thuật toán nhận dạng khuôn mặt

- $s < \alpha$ thì H là bức ảnh khuôn mặt (do H đủ gần với không gian mặt)
- $s_i < \beta$ thì T_i là bức ảnh của cùng một người với H . (H đủ gần với T_i)

Vậy là ta đã có thể tìm bức ảnh trong tập huấn luyện giống với bức ảnh H hay xác định đó có phải là bức ảnh khuôn mặt hay không . Tuy nhiên ảnh H phải có cùng kích thước với những bức ảnh trong tập huấn luyện . Bây giờ trong một bức ảnh lớn H có nhiều khuôn mặt , ta sẽ xác định vị trí những khuôn mặt trong bức ảnh .

Tại mỗi vị trí (x,y) trong H , đặt $H(x,y)$ là một vùng trong ảnh H có kích thước $M \times N$ tại (x,y) , ta xem ảnh con $H(x,y)$ là một vector $M \times N$ chiều .

$$K(x,y) = H(x,y) - m ;$$

Tìm $K^f(x,y)$ là hình chiếu của $K(x,y)$ lên không gian khuôn mặt .

$$\text{Tính } s(x,y) = \|K(x,y) - K^f(x,y)\| .$$

Tập hợp các giá trị $s(x,y)$ tạo thành một bản đồ khuôn mặt (face map) của H , từ đó ta có thể xác định vị trí những khuôn mặt trong ảnh .

CHƯƠNG 3
ẢNH MÀU TRÊN MATLAB VÀ
CÁC LỆNH XỬ LÝ ẢNH MÀU TRONG MATLAB

3.1 Giới thiệu ảnh số

Ảnh số là tập hợp các điểm ảnh với mức xám phù hợp dùng để mô tả ảnh gần với ảnh thật. Ảnh là một sự vật đại diện cho con người, sinh vật hay sự vật nào đó .v.v... ảnh động như ta thấy trên truyền hình thực chất là tập hợp của rất nhiều ảnh tĩnh liên tiếp. Khi một ảnh được số hóa thì nó trở thành ảnh số và ảnh số này lại là một tập hợp của rất nhiều phần tử ảnh được gọi là điểm ảnh hay là "pixel". Mỗi điểm ảnh lại được biểu diễn dưới dạng một số hữu hạn các bit.

Chúng ta có thể chia ảnh ra làm ba loại khác nhau :

- Ảnh đen trắng : Mỗi điểm ảnh được biểu diễn bởi một bit
- Ảnh Gray – scale : Mỗi điểm ảnh được biểu diễn bằng các mức chói khác nhau, thường thì ảnh này được biểu diễn bằng 256 mức chói hay là 8 bit cho mỗi điểm ảnh.
- Ảnh màu : Mỗi điểm ảnh chia ra thành tín hiệu chói và tín hiệu màu



泡泡网 PCPOP.COM

Hình 3.1: Ảnh màu

3.1.1 Biểu diễn ảnh số

Trong biểu diễn ảnh, người ta thường dùng các phần tử đặc trưng của ảnh là Pixel. Nhìn chung có thể xem một hàm 2 biến chứa các thông tin biểu diễn của một ảnh. Các mô hình biểu diễn ảnh cho ta một mô tả logic hay định lượng các tính chất của hàm này.

Việc xử lý ảnh số phải được lấy mẫu và lượng tử hóa. Việc lượng tử hóa là chuyển đổi tín hiệu tương tự sang tín hiệu số của một ảnh đã lấy mẫu sang một số hữu hạn mức xám.

Một số mô hình thường dùng biểu diễn ảnh: mô hình toán, mô hình thống kê.

3.1.2 Ảnh màu

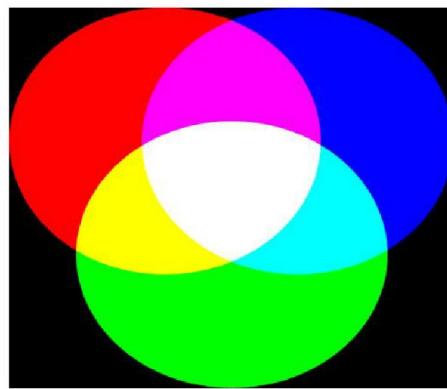
*cơ sở về màu :

Như ta đã biết thì khi cho ánh sáng trắng đi qua lăng kính ta sẽ thu được một dãy phô màu bao gồm 6 màu rộng : tím, lam, lục, vàng, cam, đỏ. Nếu nhìn kỹ thì sẽ không có ranh giới rõ ràng giữa các màu mà màu này sẽ từ từ chuyển sang màu kia. Mắt chúng ta nhìn thấy được là do ánh sáng phản xạ từ vật thể.

Tất cả các màu được tạo ra từ 3 màu cơ bản (màu sơ cấp) là : đỏ (R), lam (B) và lục (G). Các màu cơ bản trộn lại với nhau theo một tỉ lệ nhất định để tạo ra các màu thứ cấp

Phương trình màu :

$$Y = 0.2989 * R + 0.58662 * G + 0.11448 * B$$



Hình 3.2: Các màu cơ sở

Vd : đỏ + lục = vàng

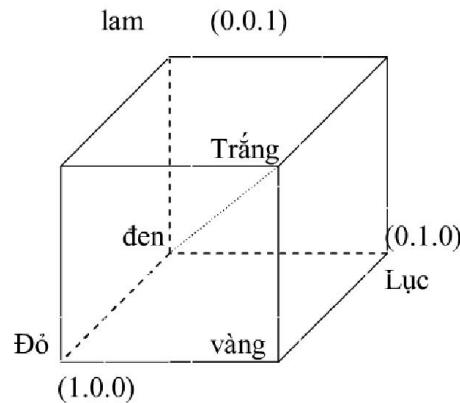
Lục + lam = xanh

Trộn ba màu sơ cấp hoặc trộn một màu thứ cấp với màu sơ cấp ngược với nó sẽ tạo ra được ánh sáng trắng

Các màu gốc có liên quan đến các khái niệm sinh học hơn là vật lý, nó dựa trên cơ sở phản ứng sinh lý học của mắt người đối với ánh sáng. Mắt người có các tế bào cảm quang có hình nón nên còn được gọi là tế bào hình nón, các tế bào này thông thường có phản ứng cực đại với ánh sáng vàng - xanh lá cây (tế bào hình nón L), xanh lá cây (tế bào hình nón M) và xanh lam (tế bào hình nón S) tương ứng với các bước sóng khoảng 564 nm, 534 nm và 420 nm. Ví dụ, màu vàng thấy được khi các tế bào cảm nhận màu xanh ánh sáng được kích thích nhiều hơn một chút so với tế bào cảm nhận màu xanh lá cây và màu đỏ cảm nhận được khi các tế bào cảm nhận màu vàng - xanh lá cây được kích thích nhiều hơn so với tế bào cảm nhận màu xanh lá cây.

Các đặc trưng dùng để phân biệt một màu với màu khác là : độ sáng (brightness) , sắc màu (hue) và độ bảo hòa màu (Saturation)

- Màu sắc có liên quan đến bước sóng ánh sáng . Thông thường, sắc màu chính là tên của màu. Ví dụ: đỏ, cam, lục...
 - Độ sáng thể hiện về cường độ ánh sáng : mô tả nó sáng hay tối như thế nào
 - Độ bảo hòa màu : thể hiện độ thuần khiết của màu. Khi độ bảo hòa cao, màu sẽ sạch và rực rỡ.
- Có nhiều mô hình màu như RGB,CYM,YIQ,CIE...Ở đây chỉ trình bày về mô hình màu RGB



Hình 3.3: Mô hình màu RGB

Các màu R,G,B nằm ở các đỉnh trên trục tọa độ của khối vuông. Màu đen nằm ở gốc tọa độ, màu trắng nằm ở góc xa nhất so với điểm gốc. Thang màu xám kéo dài từ đen đến trắng (đường chấm).

Hình ảnh trong mô hình màu RGB bao gồm 3 mặt phẳng ảnh độc lập (dùng cho các màu sơ cấp).

Thường thì ta giả thiết là tất cả các giá trị màu được chuẩn hóa (tức là khôi vuông là khôi đơn vị), tất cả các giá trị màu nằm trong khoảng [0,1]

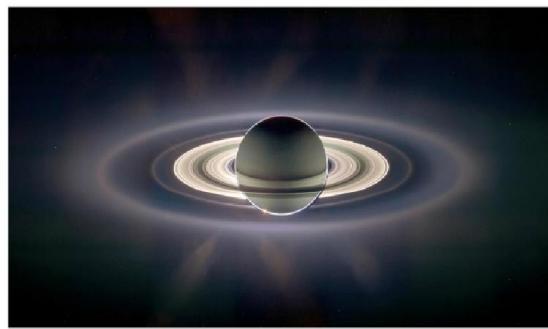
Vì vậy trong hệ màu RGB các màu có thể mô tả như là những điểm bên trong hình lập phương. Ở gốc tọa độ (0;0;0) là màu đen. Trên các trục tọa độ dương là các màu đỏ lục, lam. Khi đó ánh sáng từ các điểm riêng biệt sẽ được cộng với nhau để tạo ra các màu khác nhau.

- (0, 0, 0) là màu đen
- (255, 255, 255) là màu trắng
- (255, 0, 0) là màu đỏ
- (0, 255, 0) là màu xanh lá cây
- (0, 0, 255) là màu xanh lam
- (255, 255, 0) là màu vàng
- (0, 255, 255) là màu xanh ngọc
- (255, 0, 255) là màu hồng sẫm

3.1.3 Các định dạng ảnh cơ bản trong xử lý ảnh

Ảnh thu được sau quá trình số hóa thường được lưu lại cho các quá trình xử lý tiếp theo hay truyền đi. Trong quá trình phát triển của kỹ thuật xử lý ảnh, tồn tại nhiều định dạng ảnh khác nhau từ ảnh đen trắng (với định dạng IMG), ảnh đa cấp xám cho đến ảnh màu: (BMP, GIF, JPEG...).

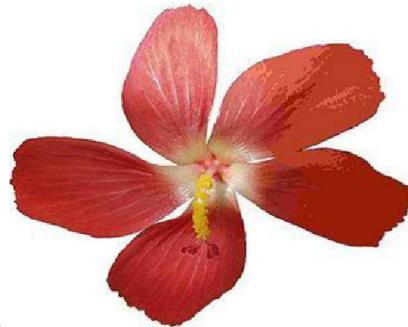
- Định dạng ảnh IMG là ảnh đen trắng, phần đầu của IMG có 16byte chứa thông tin.
- Định dạng ảnh GIF:GIF (viết tắt của *Graphics Interchange Format*; trong tiếng Anh nghĩa là "Định dạng Trao đổi Hình ảnh") là một định dạng tập tin hình ảnh bitmap cho các hình ảnh dùng ít hơn 256 màu sắc khác nhau và các hoạt hình dùng ít hơn 256 màu cho mỗi khung hình. GIF là định dạng nén dữ liệu đặc biệt hữu ích cho việc truyền hình ảnh qua đường truyền lưu lượng nhỏ. Định dạng này được CompuServe cho ra đời vào năm 1987 và nhanh chóng được dùng rộng rãi trên World Wide Web cho đến nay. Tập tin GIF dùng nén dữ liệu bảo toàn trong đó kích thước tập tin có thể được giảm mà không làm giảm chất lượng hình ảnh, cho những hình ảnh có ít hơn 256 màu. Số lượng tối đa 256 màu làm cho định dạng này không phù hợp cho các hình chụp (thường có nhiều màu sắc), tuy nhiên các kiểu nén dữ liệu bảo toàn cho hình chụp nhiều màu cũng có kích thước quá lớn đối với truyền dữ liệu trên amngj hiện nay. Định dạng JPEG là nén dữ liệu thất thoát có thể được dùng cho các ảnh chụp, nhưng lại làm giảm chất lượng cho các bức vẽ ít màu, tạo nên những chỗ nhòe thay cho các đường sắc nét, đồng thời độ nén cũng thấp cho các hình vẽ ít màu. Như vậy, GIF thường được dùng cho sơ đồ, hình vẽ nút bấm và các hình ít màu, còn JPEG được dùng cho ảnh chụp. Định dạng GIF dựa vào các bảng màu: một bảng chứa tối đa 256 màu khác nhau cho biết các màu được dùng trong hình.



Hình 3.4: Ảnh GIF

- Định dạng JPEG: Phương pháp nén ảnh **JPEG** (tiếng Anh, viết tắt cho *Joint Photo-graphic Experts Group*) là một trong những phương pháp nén ảnh hiệu quả, có tỷ lệ nén ảnh tới vài chục lần. Tuy nhiên ảnh sau khi giải nén sẽ khác với ảnh ban đầu. Chất lượng ảnh bị suy giảm sau khi giải nén. Sự suy giảm này tăng dần theo hệ số nén. Tuy nhiên sự mất mát thông tin này là có thể chấp nhận được và việc loại bỏ những thông tin không cần thiết được dựa trên những nghiên cứu về hệ nhãn thị của mắt người. Phần mở rộng của các file JPEG thường có dạng .jpeg, .jfif, .jpg, .JPQ, hay .JPE; dạng .jpg là dạng được dùng phổ biến nhất. Hiện nay dạng nén ảnh JPEG rất được phổ biến trong ĐTDD cũng như những trang thiết bị lưu giữ có dung lượng nhỏ. Công đoạn chính là chia nhỏ bức ảnh thành nhiều vùng nhỏ (thông thường là những vùng 8x8 pixel) rồi sử dụng biến đổi cosin rời rạc để biến đổi những vùng thể hiện này thành dạng ma trận có 64 hệ số thể hiện "thực trạng" các pixel. Điều quan trọng là ở

đây hệ số đầu tiên có khả năng thể hiện "thực trạng" cao nhất, khả năng đó giảm rất nhanh với các hệ số khác. Nói cách khác thì lượng thông tin của 64 pixels tập trung chủ yếu ở một số hệ số ma trận theo biến đổi trên. Trong giai đoạn này có sự mất mát thông tin, bởi không có biến đổi ngược chính xác. Nhưng lượng thông tin bị mất này chưa đáng kể so với giai đoạn tiếp theo. Ma trận nhận được sau biến đổi cosin rời rạc được lược bỏ sự khác nhau giữa các hệ số. Đây chính là lúc mất nhiều thông tin vì người ta sẽ vứt bỏ những thay đổi nhỏ của các hệ số. Như thế khi bung ảnh đã nén ta sẽ có được những tham số khác của các pixel. Các biến đổi trên áp dụng cho thành phần U và V của ảnh với mức độ cao hơn so với Y (mất nhiều thông tin của U và V hơn). Sau đó thì áp dụng phương pháp mã hóa của Gernot Hoffman: phân tích dãy số, các phần tử lặp lại nhiều được mã hóa bằng ký hiệu ngắn (*marker*). Khi bung ảnh người ta chỉ việc làm lại các bước trên theo quá trình ngược lại cùng với các biến đổi ngược



Hình 3.5: Ảnh dạng JPEG

3.2 Các kiểu hình ảnh trong Matlab

Image Processing Toolbox của Matlab hỗ trợ bốn kiểu biểu diễn hình ảnh cơ bản gồm: Ảnh chỉ số(indexed images), ảnh độ sáng(intensity images), ảnh nhị phân (binary images), ảnh RGB(RGB images).

Ảnh chỉ số

Với cách biểu diễn ảnh này mỗi ảnh sẽ được biểu diễn bởi hai ma trận, một ma trận dữ liệu ảnh X và một ma trận màu (còn gọi là bản đồ màu). Ma trận dữ liệu có thể thuộc kiểu uint8, uint16, hoặc double. Ma trận màu là ma trận kích thước $m \times 3$ gồm các phần tử kiểu double có giá trị nằm trong khoảng [0,1]. Mỗi hàng của ma trận xác định các thành phần red, green, blue của một màu trong tổng số m màu được sử dụng trong ảnh, giá trị của mỗi phần tử trong ma trận dữ liệu cho biết màu của điểm ảnh đó nằm ở hàng nào trong ma trận màu. Nếu ma trận dữ liệu thuộc kiểu double, giá trị 1 sẽ tương ứng với hàng thứ 1 trong bảng màu, giá trị thứ 2 sẽ tương ứng với hàng thứ hai trong bảng màu.. Nếu ma trận dữ liệu thuộc kiểu uint8 hoặc uint16, giá trị 0 tương ứng với hàng 1, giá trị 1 tương ứng

với hàng 2,.. Riêng với kiểu uint6, Matlab không hỗ trợ đủ các phép toán so với kiểu uint8 nên khi cần sử lý ta chuyển sang kiểu dữ liệu uint8 hoặc double bằng các hàm **imapprox** hoặc **im2double**.

Ảnh biểu diễn theo độ sáng

Mỗi ảnh được biểu diễn bởi một ma trận hai chiều, trong đó giá trị của mỗi phần tử cho biết độ sáng (hay mức xám) của điểm ảnh đó. Ma trận này có thể thuộc một trong các kiểu uint8, uint16 hoặc double. Trong đó giá trị nhỏ nhất 0 tương ứng với màu đen còn giá trị lớn nhất(255 hoặc 65535 tùy kiểu dữ liệu nào) ứng với màu trắng. Như vậy, ảnh biểu diễn theo kiểu này gọi là ảnh “trắng đen” hoặc ảnh gray scale.

Ảnh nhị phân

Ảnh nhị phân cũng được biểu diễn bằng ma trận hai chiều nhưng thuộc kiểu logical, có nghĩa là mỗi điểm ảnh chỉ có thể nhận một trong hai giá trị 0 (đen) hoặc 1 (trắng).

Ảnh RGB

Ảnh RGB còn gọi là ảnh “truecolor” do tính trung thực của nó. Ảnh này được biểu diễn bởi một ma trận 3 chiều có kích thước $m \times n \times 3$, với $m \times n$ là kích thước ảnh theo pixels. Ma trận này định nghĩa các thành phần màu red, green, blue cho mỗi điểm ảnh, các phần tử của nó có thể thuộc kiểu uint8, uint16, hoặc double. Ví dụ, điểm ảnh ở vị trí (10,5) sẽ có ba thành phần màu được xác định bởi các giá trị (10,5,1), (10,5,2) và (10,5,3). Các file ảnh hiện nay thường sử dụng 8 bit cho thành phần màu, nghĩa là mất 24bit cho mỗi điểm ảnh (khoảng 16 triệu màu).

3.3 Chuyển đổi giữa các kiểu dữ liệu

Chúng ta có thể chuyển đổi giữa các kiểu dữ liệu uint8, uint16 và double nhờ sử dụng các hàm chuyển đổi của Matlab như **im2double**, **im2uint8**, **im2uint16**. Cú pháp của các hàm này rất đơn giản, chỉ cần nhập vào ma trận cần chuyển kiểu, riêng với ảnh indexed cần thêm vào chuỗi “**indexed**”.

Tuy nhiên cần lưu ý các vấn đề sau khi chuyển đổi ảnh:

- Khi chuyển đổi từ ảnh nhiều bit sang ảnh ít bit hơn, như chuyển từ uint16 sang uint8 thì sẽ làm mất đi một số thông tin của ảnh ban đầu, chất lượng ảnh sẽ giảm.

- Khi chuyển đổi dữ liệu với kiểu indexed, thì lưu ý các thông tin ma trận là địa chỉ trong bảng đồ màu chứ không phải giá trị màu nên không phải lúc nào cũng chuyển đổi được. Muốn chuyển được đầu tiên ta phải dùng hàm **imapprox** để giảm số màu cần biểu diễn ảnh xuống(bằng cách cho các màu gần giống nhau thành một) rồi mới chuyển.

Tên thuộc tính	Mô tả
Filename	Chuỗi chứa tên file
FileModDate	Ngày chỉnh file gần nhất

FileSize	Số nguyên chỉ kích thước file(byte)
Format	Chuỗi cho biết định dạng ảnh
FormatVersion	Tên phiên bản định dạng ảnh
Width	Chiều rộng ảnh(pixel)
Height	Chiều cao ảnh(pixel)
BitDepth	Số bit trên một pixel
ColorType	Cho biết kiểu ảnh(truecolor, indexed..)

Bảng 3.1: Các thông tin khi gọi hàm **imfinfo**

3.4 Các phép toán số học cơ bản đối với dữ liệu ảnh

Các phép toán bao gồm các phép cộng, trừ, nhân và chia. Đây là các thao tác xử lý ảnh cơ bản trước khi thực hiện các phép biến đổi phức tạp khác. Người sử dụng có thể sử dụng các hàm số học mà Matlab cung cấp để tác động lên dữ liệu ảnh. Tuy nhiên Matlab chỉ hỗ trợ các phép toán này trên kiểu dữ liệu double nên cần phải chuyển đổi kiểu trước khi thực hiện. Để đơn giản hơn, Matlab cung cấp các hàm thực hiện các phép toán số học có thể chấp nhận bất kỳ bất kỳ kiểu dữ liệu ảnh nào và trả về kết quả giá thuộc cùng kiểu với các toán hạng.

Cú pháp	Mô tả
<code>z=imabsdiff(x,y)</code>	Trừ tương ứng mỗi phần tử y cho mỗi phần tử của x, trả về trị tuyệt đối hiệu
<code>z=imadd(x,y,out_class)</code>	Cộng hai ảnh, cộng ảnh với hằng số, out_class kiểu dữ liệu tổng
<code>im2= imcomplement(im)</code>	Lấy bù của ảnh <i>im</i>
<code>z=imdivide(x,y)</code>	Chia các phần tử x cho các phần tử y, kết quả làm tròn
<code>z=imlincomb(k1,a1,k2,a2...,out_class)</code>	Lấy tổ hợp tuyến tính $z=k1*a1+k2*a2+...$
<code>z=immultiply(x,y)</code>	Nhân hai ảnh, ảnh với hằng số
<code>z=imsubtract(x,y)</code>	Trừ hai ảnh, ảnh với hằng số

Bảng 3.2 Các phép toán số học trên ảnh

3.5 Các hàm hiển thị ảnh trong Matlab

Để hiển thị ảnh, Matlab cung cấp 2 hàm cơ bản là **image** và **imagesc**. Ngoài ra, trong Image Processing Toolbox cũng có hai hàm hiển thị khác là **imview** và **imshow**

- Hàm **image(x,y,c)** hiển thị hình ảnh biểu diễn bởi ma trận c kích thước mxn lên hệ trục tọa độ. x,y là các véctơ xác định vị trí của các điểm c(1,1) và c(m,n).
- Hàm **imagesc** có chức năng tương tự hàm image, ngoại trừ việc dữ liệu ảnh sẽ được co giãn để sử dụng toàn bộ bản đồ màu hiện hành.
- Hàm **imview** cho phép hiển thị ảnh trên của sô riêng nền Java, gọi là Image Viewer.
- Hàm **imshow** cho phép hiển thị ảnh trên một Figure và tự động thiết lập giá trị các đối tượng image, axes, figure để hiển thị hình ảnh.

Các hàm chuyển đổi loại ảnh và kiểu dữ liệu ảnh	
dither	Tạo ảnh nhị phân hay ảnh RGB
gray2ind	Chuyển ảnh trắng đen thành ảnh indexed
grayslice	Chuyển ảnh trắng đen thành ảnh indexed bằng lây ngưỡng
im2bw	Chuyển ảnh thành ảnh kiểu dữ liệu nhị phân
im2double	Chuyển ảnh thành ảnh kiểu dữ liệu double
im2uint16	Chuyển ảnh thành ảnh kiểu dữ liệu uint16
im2uint8	Chuyển ảnh thành ảnh kiểu dữ liệu uint8
imapprox	Xấp xỉ ảnh indexed bằng cách giảm số màu
ind2gray	Chuyển ảnh indexed thành ảnh gray scale
ind2rgb	Chuyển ảnh indexed thành ảnh RBG
mat2gray	Tạo ảnh gray scale từ ma trận
rgb2ind	Chuyển ảnh RBG thành ảnh indexed
rgb2gray	Chuyển ảnh RBG thành ảnh gray scale
Các hàm truy xuất dữ liệu ảnh	
imfinfo	Truy xuất thông tin ảnh
imread	Đọc ảnh từ file và xuất ra ma trận ảnh
imwrite	Lưu ma trận ảnh thành file ảnh
Các hàm biến đổi hình học	
cp2tform	Định nghĩa phép biến đổi hình học từng cặp tương ứng
imcrop	Trích xuất một phần ảnh
imresize	Thay đổi kích thước ảnh
imrotate	Thực hiện phép quay ảnh
imtransform	Thực hiện phép biến đổi hình học tổng quát
maketform	Định nghĩa phép biến đổi hình học tổng quát

Bảng 3.3 Các hàm xử lý hình ảnh khác trong Matlab

3.6 Các hàm khác được sử dụng trong đề tài

- **[filename, pathname]=uigetfile(filterspec, title):** hiển thị hộp thoại chọn đường dẫn file. Giá trị trả về tên file, và đường dẫn.
- **T=strcat(s1,s2,s3...):** ghép các chuỗi lại với nhau, trả về chuỗi nối tiếp s1s2s3...
- **strcmp(s1,s2):** hàm so sánh, trả về 1 nếu s1 giống s2, ngược lại trả về 0
- **T=dir(pathname):** Lấy thông tin của một Folder bao gồm: số file chứa trong folder, tên file, ngày tạo, kích thước file...
- **S=int2str(x):** Chuyển đổi số kiểu integer thành chuỗi ký tự
- **N=num2str(x):** Chuyển đổi các số(bất kỳ có thể số nguyên hoặc thực) thành chuỗi ký tự.
- **D=size(a):** Trả về giá trị là ma trận có dạng [x,y] là kích thước của ma trận a
- **T=reshape(X,M,N):** Trả về ma trận có kích thước MxN với các phần tử là các phần tử nằm trong ma trận X.
- **mean(X):** Ma trận X có kích thước MxN, hàm trả về ma trận có kích thước 1xN mỗi phần tử là trung bình từng cột trong ma trận X
- **mean(X,dim):** với dim là chiều lấy trung bình, nếu dim bằng 1 lấy trung bình theo cột, nếu dim bằng 2 lấy trung bình theo hàng. Không có tham số dim thì mặc định dim bằng 1.
- **double(X):** Chuyển đổi gấp đôi chính xác giá trị ma trận X .
- **E=eig(X):** Trả về một vector chứa các giá trị riêng của ma trận vuông X.
[V, D] = eig(X): tạo ra một ma trận đường chéo D của các giá trị riêng và một ma trận V có các cột tương ứng là các vector riêng, do đó: $X * V = V * D$
- **diag(V,K):** Trong đó V là một vector với các thành phần N là một ma trận vuông kiểu N+ABS(K) với các phần tử của V trên đường chéo thứ K. K = 0 là đường chéo chính, K > 0 là ở phía trên đường chéo chính và K < 0 là ở phía dưới đường chéo chính.
Diag(V): Giống như DIAG (V, 0) và đặt vector V trên đường chéo chính.
- **Sort(X):** Phân loại tăng dần hay giảm.
Đối với các vector, **Sort(X)** sắp xếp các phần tử của X thứ tự tăng dần.
Đối với ma trận, **Sort(X)** các loại mỗi cột của X thứ tự tăng dần.
Khi X là một mảng di động của chuỗi, **Sort(X)** sắp xếp các ký tự theo thứ tự bảng mã ASCII.
- **Norm(X):** Chuẩn hóa ma trận và vector X.
- **Min(X):** Trả về vị trí của phần tử nhỏ nhất của ma trận X.

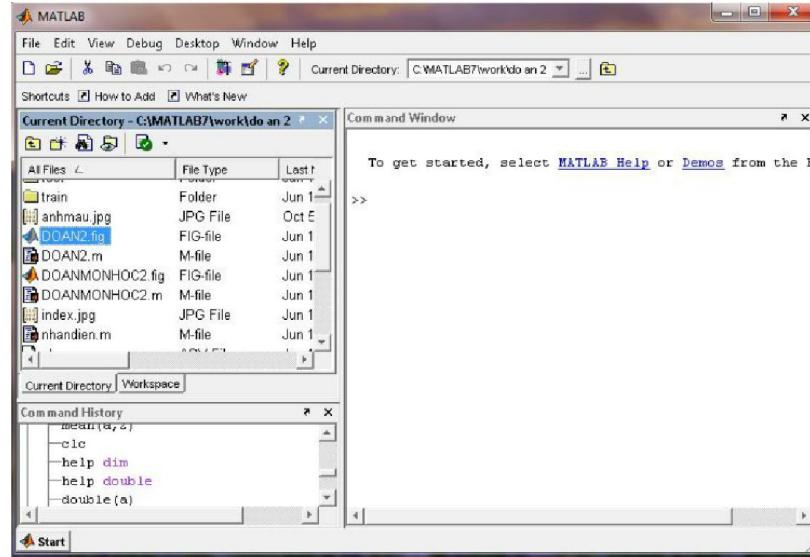
**CHƯƠNG 4
GIỚI THIỆU
CHƯƠNG TRÌNH**

4.1 Giới thiệu chương trình

Chương trình “Nhận dạng mặt người trên Matlab” là chương trình được thiết kế trên giao diện người dùng GUI của phần mềm Matlab 7.0. Rất đơn giản và dễ dàng sử dụng.

Để mở chương trình bạn có thể làm theo 2 cách

Mở trực tiếp trên chương trình Matlab 7.0. Ta chỉ đường dẫn đến thư mục **DOAN2** sau đó Run file **DOAN2.fig**. Như hình dưới đây:



Hình 4.1: Mở chương trình trên Matlab

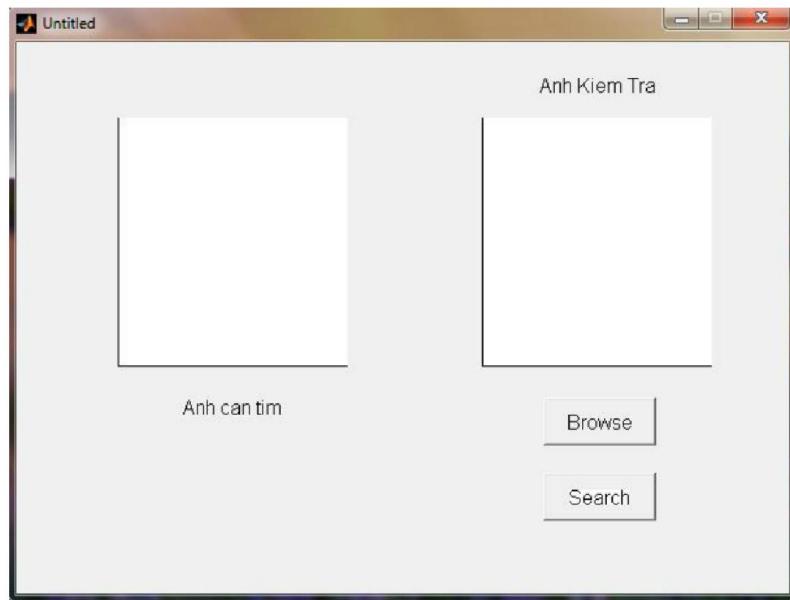
Hoặc các bạn cũng có thể vào trực tiếp thư mục **DOAN2** để mở file **DOAN2.fig**. Sau khi chạy chương trình, sẽ xuất hiện hộp thoại như hình:



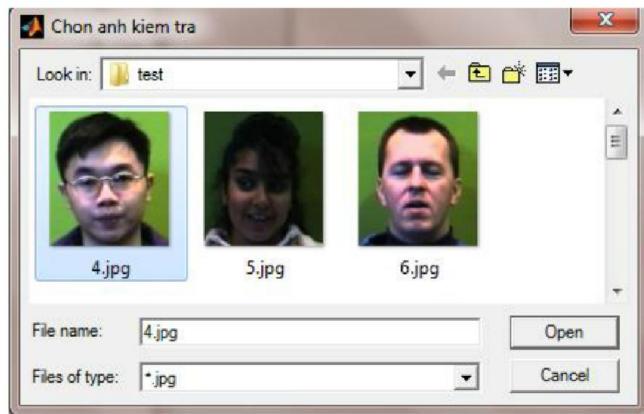
Chương 4: Giới thiệu chương trình

Hình 4.2: Giao diện chương trình

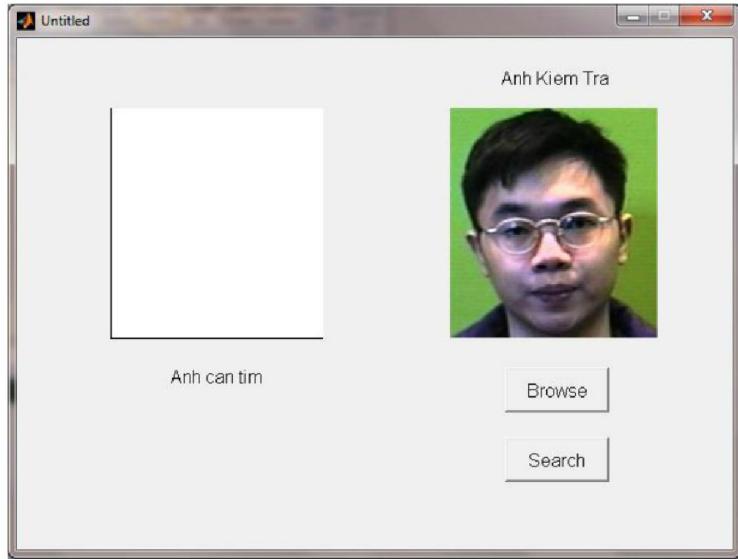
Đây chính là giao diện giới thiệu của chương trình. Trên giao diện có 2 nút nhấn **Next** và **Close**. Nếu chọn **Close**, sẽ thoát khỏi chương trình. Nếu chọn **Next**, ta sẽ đến giao diện tiếp theo của chương trình chính. Như hình dưới đây:

*Hình 4.3: Giao diện chương trình chính*

Đây chính là giao diện chính của chương trình gồm hai khung hiển thị hình ảnh và hai nút nhấn **Browse** và **Search**. Để bắt đầu tìm kiếm ta cần load ảnh khuôn mặt người cần kiểm tra bằng cách nhấn vào phím **Browse**. Hộp thoại mở file xuất hiện, bạn chọn file ảnh bạn muốn kiểm tra. Ở đây ta lấy ví dụ ảnh kiểm tra là ảnh “4.jpg” như hình vẽ:

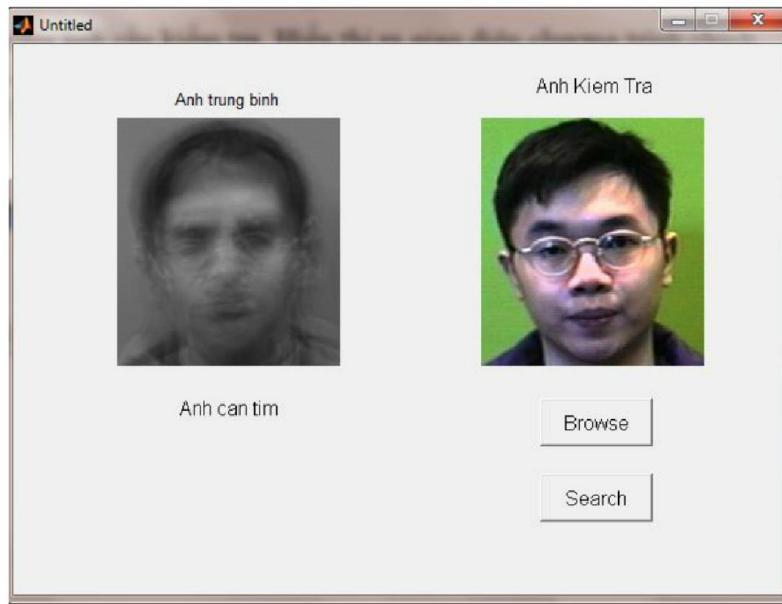
*Hình 4.4: Chọn ảnh cần kiểm tra*

Ảnh cần kiểm tra sẽ được đưa ra giao diện chương trình chính như hình:

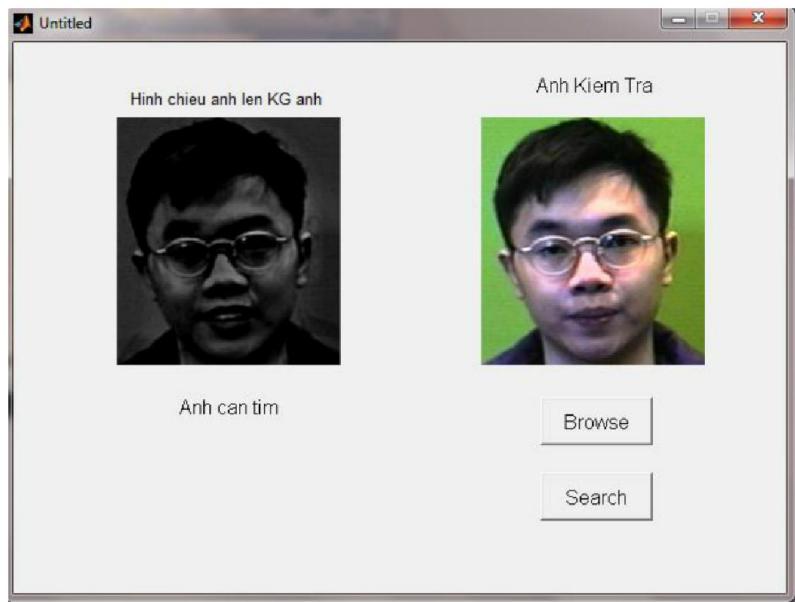


Hình 4.5: *Ảnh cần kiểm tra*

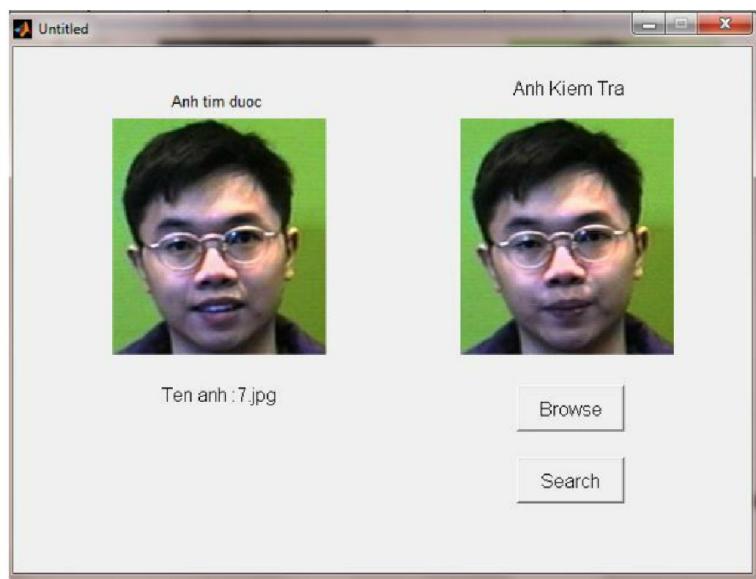
Để kiểm tra xem khuôn mặt người trong ảnh cần kiểm tra chúng ta nhấn nút **Search** chương trình sẽ chạy và tìm trong CSDL bức ảnh có khuôn mặt giống với khuôn mặt người trong ảnh cần kiểm tra. Hiển thị ra giao diện chương trình chính.



Hình 4.6: *Ảnh trung bình*



Hình 4.7: Hình chiếu ảnh lên không gian ảnh

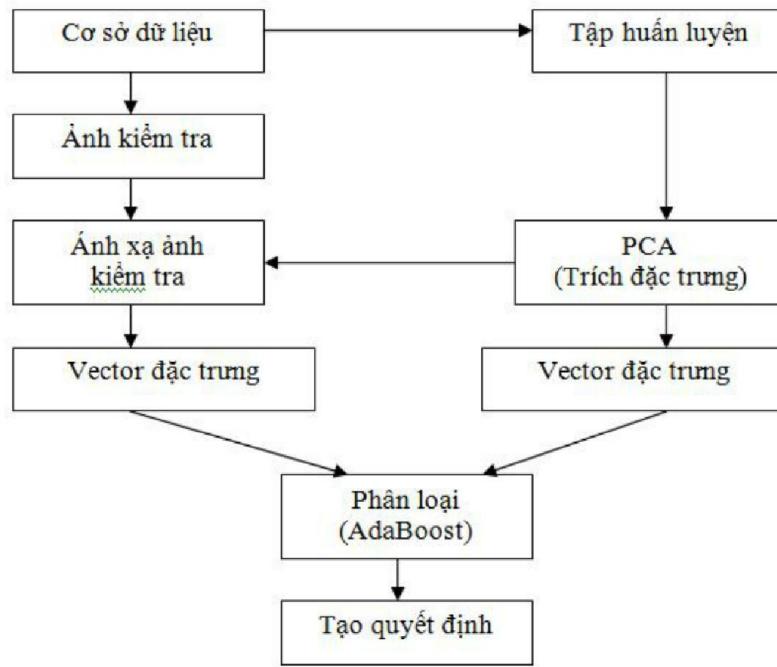


Hình 4.8: Ảnh cần tìm

Chương trình sẽ tìm ra khuôn mặt gần giống nhất với khuôn mặt cần kiểm tra. Và ở đây kết quả tìm được là hình có tên “7.jpg”.

CHƯƠNG 5
SƠ ĐỒ KHÓI VÀ
CODE CHƯƠNG TRÌNH

5.1 Sơ đồ khái



Hình 5.1: Sơ đồ khái quát của chương trình

5.2 Code chương trình

```

function varargout = DOANMONHOC2(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',     'mfilename', ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @DOANMONHOC2_OpeningFcn, ...
    'gui_OutputFcn', @DOANMONHOC2_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else

```

```
gui_mainfcn(gui_State, varargin{:});  
end  
  
function DOANMONHOC2_OpeningFcn(hObject, eventdata, handles, varargin)  
handles.output = hObject;  
% Update handles structure  
guidata(hObject, handles);  
clc;  
% --- Outputs from this function are returned to the command line.  
function varargout = DOANMONHOC2_OutputFcn(hObject, eventdata, handles)  
varargout{1} = handles.output;  
  
function Search_Callback(hObject, eventdata, handles)  
load TestImage;  
axes(handles.anhtimduoc);  
TrainPath='train';  
T = taoCSDL(TrainPath);  
[m, A, Eigenfaces] = taoEF(T);  
OutputName = nhandien(TestImage, m, A, Eigenfaces);  
  
anhtim = strcat(TrainPath,'\',OutputName);  
anhtim = imread(anhtim);  
  
imshow(anhtim);  
title('Anh tim duoc');  
str = strcat('Ten anh :',OutputName);  
set(handles.tenanh,'String',str);  
function Browse_Callback(hObject, eventdata, handles)  
[file_name file_path] = uigetfile ('*.jpg','Chon anh kiem tra ','test\2.jpg');  
if file_path ~= 0  
    TestImage = imread ([file_path,file_name]);  
    end  
axes(handles.anhkiemtra);  
if file_path ~= 0  
    imshow(TestImage);
```

Chương 5: Sơ đồ khái và code chương trình

```
end
```

```
save TestImage;
```

function T = taoCSDL(trainPath)

```
% trainPath là đường dẫn tới thư mục csdl ảnh . Thư mục này ngoài những
```

```
% file ảnh con chưa nhung file khác có tên : . , .. , Thumbs.db
```

```
% Biến tât ca ảnh kích thuoc MxN thanh vector cot M*Nx1 , sử dụng
```

```
% ham reshape của Matlab , sau đó đặt vào ma tran T , cuối cùng ma tran T
```

```
% sẽ có kích thước M*NxP
```

```
% return T
```

```
csdl = dir(trainPath);
```

```
soanh = 0;
```

```
for i = 1:size(csdl,1) % đếm những file là ảnh trong csdl
```

```
if not(strcmp(csdl(i).name,'.')|strcmp(csdl(i).name,'..')|strcmp(csdl(i).name,'Thumbs.db'))
```

```
    soanh = soanh + 1; % số ảnh chưa trong tap csdl
```

```
end
```

```
end
```

```
% Tao ma tran tu nhung tam anh
```

```
T = [];
```

```
for i = 1 : soanh
```

```
% Trong csdl của vi du nay thi cac file ảnh có tên : 1.jpg , 2.jpg ...
```

```
str = int2str(i);
```

```
str = strcat('\\',str,'.jpg');
```

```
str = strcat(trainPath,str); % lấy tên đầy đủ của file ảnh
```

```
img = imread(str);
```

```
img = rgb2gray(img);
```

```
[dong cot] = size(img);
```

```
tam = reshape(img',dong*cot,1); % biến ảnh thành vector
```

```
T = [T tam]; % tăng dimension kích thước ma tran T
```

```
end
```

function [m, A, E] = taoEF(T)

```
% T là một ma tran kích thước M*NxP chứa tất cả ảnh trong csdl , mỗi ảnh là
```

```
% một vector cột trong ma tran T .
```

Chương 5: Sơ đồ khái và code chương trình

```

% Theo thuat toan PCA , dau tien ta se tinh ra m la trung binh cua tat ca cac anh
% trong ma tran T.

% Sau do ta se lay tung buc anh trong T tru cho anh trung binh , ta se duoc mot ma
% tran A kich thuoc M*NxP

% Ta can tim Eigenface la nhung vector rieng cua ma tran A'*A' , nhung ma
% tran A'*A' kich thuoc la M*NxM*N qua lon , ta se tim nhung vector rieng
% cua ma tran A'*A co kich thuoc PxP . Ta se tim nhung vector rieng
% bang ham eig trong MatLab

% Gia su v la mot vector rieng cua ma tran A'*A' , khi do A*v la vector
% rieng cua ma tran A'*A' .

% tap hop nhung vector rieng cua ma tran A'*A' goi la Eigenfaces

% tra ve 3 gia tri :

%      m anh trung binh
%      A tap hop nhung (anh-anh trung binh )
%      E nhung vector rieng cua ma tran A'*A'

%tinh toan anh trung binh
m = mean(T,2);
soanh = size(T,2);
%xuat ra hinh anh cua m
show(m,'Anh trung binh');pause;
% tinh do lech giua anh moi buc anh voi anh trung binh
A = [];
for i = 1 : soanh
    temp = double(T(:,i)) - m; % T(:,i): vector cot thu i chinh la mot buc anh
    if i<6
        show(temp,'Anh - Anh TB');pause;
    end
    A = [A temp];
end
% tim nhung tri rieng va nhung vector rieng cua ma tran A'*A' , tu do tim
% nhung vector rieng cua ma tran A'*A' la nhung Eigenface
L = A'*A;
[V D] = eig(L);
%V chua nhung vector rieng , con D chua nhung tri rieng trong do vector
%rieng V(:,i) ung voi tri rieng D(i,i)

```

Chương 5: Sơ đồ khái và code chương trình

```

D1=diag(D);
D1=sort(D1);
s=size(D1);s=s(1);
D1=D1(s-18);
LeigV = [];%tap hop vector rieng cua ma tran L=A'*A
for i = 1 : size(V,2)
    if( D(i,i)>D1 )
        LeigV = [LeigV V(:,i)];
    end
end
%ta chi lay 18 vector rieng ung voi 10 tri rieng lon nhat
%onhu da noi o tren , sau khi co cac vector rieng cua ma tran A'*A , ta tim
%cac vector rieng cua ma tran A'*A' bang cach lay ma tran A nhan voi cac
%vector rieng nay , tap hop cac vector rieng cua ma tran A'*A' con duoc goi
%la Eigenface do day la nhung vector rieng va no giong hinh khuon mat .
E = A * LeigV;
%xuat ra mot so hinh anh cua cac Eigenface
for i=1:5
    anh=E(:,i);
    show(anh,'Eigenface');pause;
end
%E la mot co so gom nhung vector truc giao , ta se chuan hoa no de E bien
%thanh mot co so truc chuan
sovector=size(E,2);
for i=1:sovector
    dodai=norm(E(:,i));
    E(:,i)=E(:,i)/dodai;
end
end

function show(m,t)
%ham show duoc su dung de hien thi hinh anh voi kich thuoc mau
%m la matran anh can hien thi
%t la chuoi the hien tieu de cua anh

```

```

im=imread('anhmau.jpg');
try
    im=rgb2gray(im);
catch
end
[dong,cot]=size(im);% lay kich thuoc (anhmau.jpg coi nhu bien tam luu kich thuoc)
tam=reshape(m,cot,dong);
tam=tam';
imshow(tam);
dem=1;
for i=1:dong
    for j=1:cot
        im(i,j)=tam(i,j);
    end
end
imshow(im);title(t)
end

```

```

function anhtim = nhandien(InputImage, m, A, E)
%Ham nay se so sanh buc anh kiem tra voi tung buc anh trong CSDL .
%Dau tien tinh toa do hinh chieu cua buc anh kiem tra , sau do tinh toa do
%hinh chieu cua tat ca buc anh trong csdl . Cuoi cung do khoang cach giua
%toa do hinh chieu cua nhung buc anh trong csdl voi toa do hinh chieu cua
%anh kiem tra . Buc anh trong csdl co khoang cach ngan nhat voi anh kiem
%tra chinh la buc anh tuong ung voi buc anh kiem tra .
%Inputimage la duong dan toi buc anh can kiem tra
%m la anh trung binh cua cac anh trong csdl
%A la ma tran , moi cot la do lech giua mot anh trong csdl so voi anh trung
%binh , con goi la vector anh trung tam
%E la tap hop nhung vector rieng cua ma tran A*A'
%m,A va E duoc lay tu ham 'taoEF'
%anhtim la ten cua buc anh tim duoc trong csld

```

```

toado = [];%tap toa do hinh chieu cua moi buc anh trong csdl
sovector = size(E,2);%so vector rieng trong E ( la so cot )

```

Chương 5: Sơ đồ khái và code chương trình

```

for i = 1 : sovector
    tam = E'*A(:,i); %toa do hinh chieu cua buc anh Ai
    toado = [toado tam];
end
tam = rgb2gray(InputImage);
[dong cot] = size(tam);
InImage = reshape(tam',dong*cot,1);
%tim do lech giua anh kiem tra va anh trung binh trong csdl , do lech la
%mot vector cot giuong nhu cac vector cot trong ma tran A
dolech = double(InImage)-m;
%tuong tu nhu khi tim toa do hinh chieu cua cac vector cot cua A , bay gio ta
%tim toa do hinh chieu cua buc anh kiem tra
toadoKT = E'*dolech;
%
hinhhieuKT=double(InImage)*0;
for i=1:sovector
    hinhhieuKT=hinhhieuKT + toadoKT(i,1)*E(:,i);
end
show(hinhhieuKT,'Hinh chieu anh len KG anh');pause;%figure;
kc=norm(double(InImage)-hinhhieuKT);
str=num2str(kc);
str=strcat('Khoang cach tu anh kiem tra toi khong gian khuon mat : ',str);
disp(str);
%Bay gio ta se tinh khoang cach giua toa do hinh chieu cua buc anh kiem tra voi
%tat ca toa do hinh chieu cua cac buc anh trong csdl . Toa do hinh chieu cua
%anh kiem tra se co khoang cach ngan nhat voi hinh chieu cua buc anh tuong
%ung trong csdl (Hai buc anh cung la khuon mat cua mot nguoi )
%
%ta luu y la moi toa do hinh chieu la mot vector , ta se dung chuan Euclid de tinh
%khoang cach giua 2 vector (2 toa do hinh chieu)
khoangcach = [];
for i = 1 : sovector
    q = toado(:,i);
    tam = ( norm( toadoKT - q ) )^2;
    khoangcach = [khoangcach tam];

```

Chương 5: Sơ đồ khái và code chương trình

```
endA  
%lay ra khoang cach ngan nhat va vi tri cua buc anh tim duoc trong csdl ,  
%ta luu y la nhung file anh trong csdl co ten la : 1.jpg , 2.jpg ....  
[minKC , vitri] = min(khoangcach);  
str=num2str(minKC);  
str=strcat('Min khoang cach hai toa do hinh chieu : ',str);  
disp(str);  
anhTim = strcat(int2str(vitri),'.jpg');
```


CHƯƠNG 6
PHẠM VI GIỚI HẠN VÀ
HƯỚNG MỞ RỘNG CỦA ĐỀ TÀI

6.1 Phạm vi giới hạn của đề tài

Do thời gian tiến hành nghiên cứu tài liệu tham khảo có hạn. Đề tài “Nhận dạng mặt người trên Matlab” chỉ sử dụng một thuật toán duy nhất là thuật toán PCA. Nên làm cho chương trình nhận dạng phụ thuộc rất nhiều vào tập huấn luyện, vị trí các khuôn mặt trong hình. Trong quá trình chạy chương trình các ảnh được xử lý tạo ra như: ảnh trung bình, ảnh-ảnh trung bình, EigFace không được lưu lại. Nên khi chạy lại chương trình các ảnh trên sẽ được khởi tạo lại từ đầu.

6.2 Hướng mở rộng của đề tài

Đề tài có thể được phát triển thành một phần mềm nhận dạng mặt người tốt hơn, bằng cách kết hợp với một số thuật toán nhận dạng và xử lý ảnh hiện đại hơn. Cho ra kết quả chính xác hơn. Có thể phát triển thành đề tài nhận dạng qua webcam...

