TRƯỜNG ĐH SPKT TP. HỒ CHÍ MINH
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ CÔNG NGHIỆP – Y SINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
ĐỘC LẬP - TỰ DO - HẠNH PHÚC
----o0o----

Tp. HCM, ngày 5 tháng 7 năm 2018

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên:	Nguyễn Thị Đài Trang	MSSV: 13141378
	Hà Tiến Dương	MSSV: 13141047
Chuyên ngành:	Kỹ thuật Điện tử - Truyền thông	Mã ngành:
Hệ đào tạo:	Đại học chính quy	Mã hệ:
Khóa:	2013	Lớp: 13141

I. TÊN ĐỀ TÀI: **NHẬN DẠNG CẢM XÚC KHUÔN MẶT NGƯỜI**

II. NHIỆM VỤ

1. Các số liệu ban đầu:
(*ghi những thông số, tập tài liệu tín hiệu, hình ảnh,…*)
.....
.....
.....
.....
.....

2. Nội dung thực hiện:
(*ghi những nội dung chính cần thực hiện như trong phần tổng quan*)
.....
.....
.....
.....
.....
.....

III. NGÀY GIAO NHIỆM VỤ:
IV. NGÀY HOÀN THÀNH NHIỆM VỤ:
V. HỌ VÀ TÊN CÁN BỘ HƯỚNG DẪN: Ts. Nguyễn Thanh Hải

CÁN BỘ HƯỚNG DẪN BM. ĐIỆN TỬ CÔNG NGHIỆP – Y SINH

TRƯỜNG ĐH SPKT TP. HỒ CHÍ MINH
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ CÔNG NGHIỆP – Y SINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
ĐỘC LẬP - TỰ DO - HẠNH PHÚC
----o0o----

Tp. HCM, ngày 5 tháng 7 năm 2018

LỊCH TRÌNH THỰC HIỆN ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên 1: Hà Tiến Dương
Lớp: 13141DT MSSV: 13141047
Họ tên sinh viên 2:
Lớp: 13141DT MSSV: 13141378
Tên đề tài: NHẬN DẠNG CẢM XÚC KHUÔN MẶT NGƯỜI

Tuần/ngày	Nội dung	Xác nhận GVHD
Tuần 1	Tìm đề tài	
Tuần 2,3	Nghiên cứu đề tài cũ. Tìm hiểu hoạt động Arduino và Matlab.	
Tuần 4	Cài đặt Matlab, cài Arduino, cài webcam trên Matlab.	
Tuần 5	Lập trình Arduino với các chân I/O để nhúng dữ liệu.	
Tuần 6	Lập trình và xây dựng bộ ảnh huấn luyện để nhận dạng.	
Tuần 7	Lập trình nhận dạng cảm xúc từ bộ ảnh huấn luyện.	
Tuần 8,9	Lập trình nhận dạng cảm xúc qua Webcam.	
Tuần 10	Hiệu chỉnh toàn bộ chương trình.	
Tuần 11,12	Viết luận văn.	
Tuần 13,14,15	Chỉnh sửa, in đồ án.	

GV HƯỚNG DẪN
(Ký và ghi rõ họ và tên)

LỜI CAM ĐOAN

Đề tài này là do Nguyễn Thị Đài Trang và Hà Tiến Dương tự thực hiện dựa vào một số tài liệu trước đó và không sao chép từ tài liệu hay công trình đã có trước đó.

Người thực hiện đề tài

Nguyễn Thị Đài Trang
Hà Tiến Dương

LỜI CẢM ƠN

Trong thời gian thực hiện đề tài, nhóm thực hiện được sự giúp đỡ của gia đình, quý thầy cô và bạn bè nên đề tài đã được hoàn thành. Nhóm thực hiện xin chân thành gửi lời cảm ơn đến:

Thầy Nguyễn Thanh Hải, giảng viên trường Đại Học Sư Phạm Kỹ Thuật Tp.HCM đã trực tiếp hướng dẫn và tận tình giúp đỡ tạo điều kiện để nhóm có thể hoàn thành tốt đề tài.

Nhóm thực hiện cũng xin chân thành cám ơn đến các thầy cô trong khoa Điện - Điện tử của trường Đại Học Sư Phạm Kỹ Thuật Tp.HCM đã tận tình dạy dỗ, chỉ bảo, cung cấp cho những người thực hiện những kiến thức nền, chuyên môn làm cơ sở để hoàn thành đề tài này.

Cảm ơn gia đình đã động viên và luôn luôn bên cạnh trong những lúc khó khăn nhất.

Xin gửi lời cảm ơn đến những người bạn sinh viên khoa Điện-Điện tử đã giúp đỡ những người thực hiện đề tài để có thể hoàn thành tốt đề tài này.

Xin chân thành cảm ơn!

Người thực hiện đề tài:

Nguyễn Thị Đài Trang
Hà Tiến Dương

MỤC LỤC

Contents

LIỆT KÊ HÌNH VẼ

Hình **Trang**

LIỆT KÊ BẢNG

Bảng **Trang**

Chương 1. TỔNG QUAN

ĐẶT VẤN ĐỀ

Trong những năm gần đây, xử lý ảnh đang được nghiên cứu và phát triển với tốc độ nhanh chóng bởi các trung tâm nghiên cứu, trường đại học và học viện. Trong đó, nhận dạng và phân loại hình ảnh là một trong những lĩnh vực được theo đuổi một cách tích cực. Ý tưởng cốt lõi từ việc nhận dạng và phân loại hình ảnh là phân tích ảnh từ dữ liệu thu được bởi các cảm biến hình ảnh như camera, webcam. Nhờ hệ thống xử lý hình ảnh mà con người đã giảm bớt khối công việc cũng như tăng sự chính xác trong việc đưa ra các quyết định liên quan đến xử lý ảnh trên nhiều lĩnh vực: quân sự và quốc phòng, các hệ thống kỹ nghệ hoá sinh, giải phẫu, các hệ thống thông minh, robotics, các hệ thống an ninh [1].

Cùng với các hình thức nhận dạng khác như: nhận dạng giọng nói, chữ viết, dấu vân tay, võng mạc [2], thì bài toán nhận dạng cảm xúc trên khuôn mặt người đang được nhóm quan tâm chú ý. Trên cơ thể con người có rất nhiều đặc điểm để nhận dạng, nhưng khuôn mặt là nơi thể hiện nhiều trạng thái cảm xúc: vui, buồn, giận dữ, phẫn nộ.

Điều quan trọng nhất trong xã hội hiện nay là các thiết bị điện tử hầu như tự động hoá, thông minh, hiểu ý con người mà không cần phải thiết bị điều khiển trung gian nào. Các thuật toán nhận dạng và xử lý ngày càng được sử dụng rộng rãi. Vì vậy từ những vấn đề trên, ĐATN kiến nghị thực hiện đề tài "***Nhận diện cảm xúc khuôn mặt người***" bằng xử lý ảnh.

1.1. MỤC TIÊU

Xây dựng hệ thống nhận diện cảm xúc khuôn mặt bằng cách nối board mạch Arduino với phần mềm Matlab. Mô hình sẽ nhận diện cảm xúc trên khuôn mặt người, đồng thời ứng với mỗi cảm xúc Arduino sẽ điều khiển trạng thái của cửa và đèn led. Khi nhận được cảm xúc vui (cười) sẽ điều khiển servo mở cửa. Khi buồn (không cười) sẽ điều khiển mở đèn. Khi nhận được cảm xúc ngạc nhiên sẽ điều khiển servo đóng cửa và tắt đèn.

1.2. NỘi DUNG NGHIÊN CỨU

Mục tiêu xây dựng hệ thống: "**Nhận dạng cảm xúc trên khuôn mặt**" như trên thì nhóm sẽ thực hiện những nội dung như sau :

- Tìm hiểu các tài liệu, đồ án trước đó.
- Xử lý ảnh đầu vào từ camera hoặc từ thư mục có sẵn.
- Kết nối và cài đặt nguồn thư viện cho Arduino trên phần mềm Matlab.
- Xây dựng hệ thống nhận dạng cảm xúc trên khuôn mặt.
- Lắp ráp các khối điều khiển vào mô hình.
- Lập trình và viết code cho các ứng dụng trên kit Arduino.
- Chạy thử và điều chỉnh mô hình.
- Đánh giá kết quả thực hiện.
- Viết báo cáo.

1.3. GIỚI HẠN

Mô hình chỉ nhận dạng cảm xúc trên khuôn mặt với ba trạng thái: vui, buồn và ngạc nhiên.

Sử dụng webcam từ máy tính .

1.4. BỐ CỤC

Đồ án tốt nghiệp: "**Nhận dạng cảm xúc trên khuôn mặt**" trình bày trong 6 chương với bố cục như sau:

- **Chương 1:** Tổng quan

Chương này trình bày đặt vấn đề dẫn nhập lý do chọn đề tài, mục tiêu, nội dung nghiên cứu, các giới hạn thông số và bố cục đồ án.

- **Chương 2:** Cơ sở lý thuyết.

Giới thiệu về kit Arduino, giới thiệu về cách cài đặt thư viện cho Arduino và camera webcam trên phần mềm Matlab.

Giới thiệu về phép phân tích thành phần chính PCA và áp dụng vào bài toán nhận dạng cảm xúc trên khuôn mặt người.

- **Chương 3:** Thiết kế và Tính toán

Phân tích, xây dựng sơ đồ khối, sơ đồ phần cứng, thiết kế về chương trình cho hệ thống nhận dạng cảm xúc trên khuôn mặt người được viết trên phần mềm Matlab, sử dụng thư viện của Matlab cho kit Arduino.

- **Chương 4:** Thi công hệ thống

Xây dựng chương trình hoàn chỉnh cho toàn hệ thống, các hàm, các lưu đồ, các chương trình được sử dụng. Tạo tập tin huấn luyện, lắp ráp và test cho chương trình. Viết tài liệu hướng dẫn sử dụng, quy trình thao tác.

- **Chương 5:** Kết quả, Nhận xét và đánh giá

Nêu các kết quả đạt được khi thực hiện chương trình, phân tích, nhận xét, đánh giá kết quả thực thi được.

- **Chương 6:** Kết luận và Hướng phát triển

Tóm tắt những kết quả đạt được, những hạn chế và nêu lên các hướng phát triển trong tương lai.

Chương 2. CƠ SỞ LÝ THUYẾT

2.1 PHẦN CỨNG

2.1.1 Kit Arduino

Arduino ra đời tại thị trấn Ivrea – Italy, được giới thiệu vào năm 2005 như một công cụ cho sinh viên tìm tòi, học hỏi, phát triển, nhưng đến nay Arduino đã được sử dụng rộng rãi trên thế giới và chứng tỏ được hiệu quả thông qua vô số ứng dụng từ người dùng.

Arduino là một board mạch vi xử lý được dùng để tương tác với các thiết bị phần cứng như cảm biến, động cơ, đèn hay các thiết bị khác. Một board Arduino bao gồm một vi điều khiển AVR với nhiều linh kiện bổ sung đã làm nên board mạch với nhiều thế mạnh hơn so với các vi điều khiển khác như: chạy được trên nhiều hệ điều hành khác nhau (Windows, Linux..), dễ dàng lắp ráp và mở rộng phần cứng, phát triển dựa trên nguồn mở, dễ dàng chia sẻ mã nguồn với nhau mà không phải lo lắng ngôn ngữ hay hệ điều hành đang sử dụng…

Arduino thường sử dụng các dòng chip megaAVR, đặc biệt là ATmega8, ATmega168, ATmega328, ATmega1280 và ATmega2560. Hầu hết các mạch gồm một bộ điều chỉnh tuyến tính 5V và một thạch anh dao động 16 MHz, các board Arduino hiện tại được lập trình thông qua cổng USB.

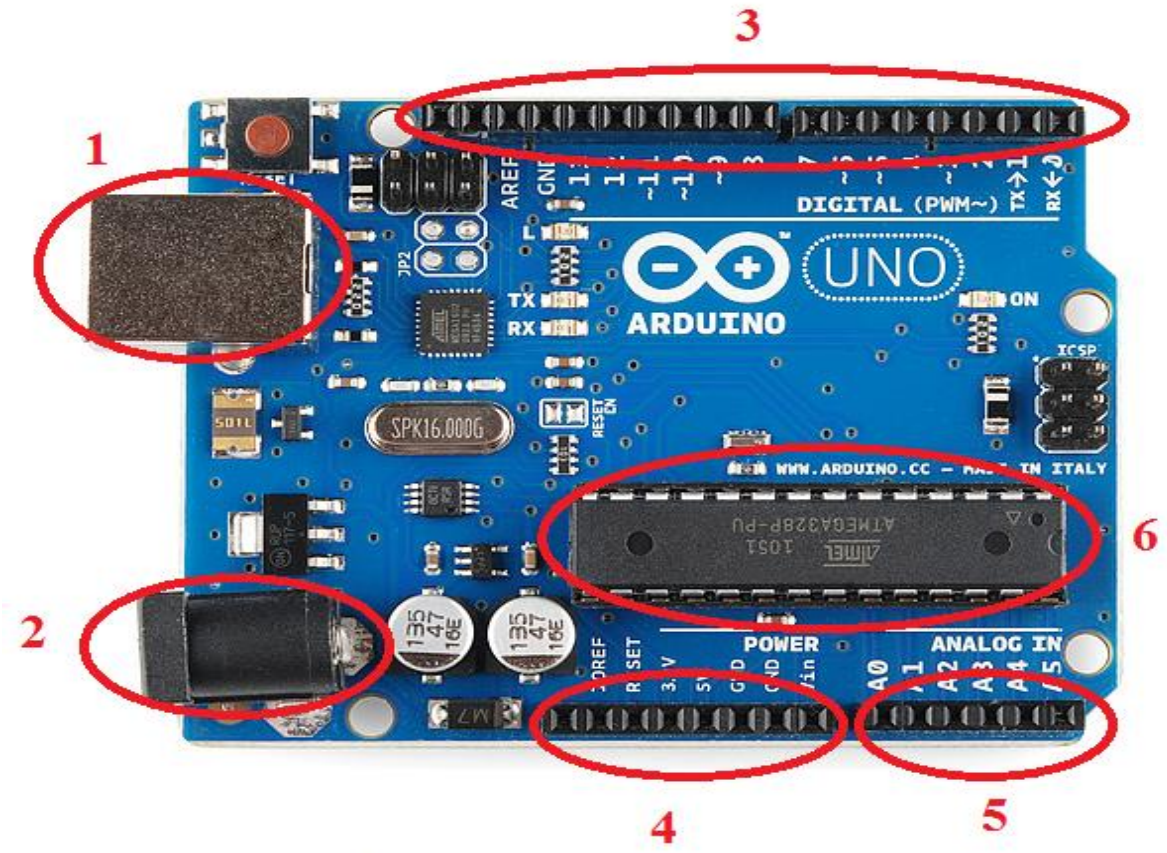
2.1.2. Kit Arduino Uno

Kit Arduino có nhiều phiên bản với tính năng và mục đích sử dụng khác nhau. Board Arduino Uno là một trong những phiên bản được sử dụng rộng rãi nhất bởi chi phí và tính linh động của nó.

Arduino Uno [3] là một board mạch vi điều khiển dựa trên chip Atmega328 với 14 chân vào/ra bằng tín hiệu số được đánh số từ 0 đến 13, trong đó 6 chân có thể tạo xung PWM được đánh dấu "~" trước mã số của chân, 6 chân nhận tín hiệu analog được đánh dấu từ A0 đến A5, có thể sử dụng như là 6 chân I/O số. Có 2 mức điện áp là 0V và 5V với dòng vào/ra tối đa trên mỗi chân là 30 mA.

Bảng 2.1. Thông số kỹ thuật Arduino Uno.

Vi điều khiển	Atmega328P họ 8 bit
Điện áp hoạt động	5V
Điện áp đầu vào	7-12V
Điện áp đầu vào giới hạn	6-20V
Số chân Digital I/O	14 (có 6 chân PWM)
Số chân Analog	6 (độ phân giải 10bit)
DC current per I/O pin	20mA
DC current for 3.3V pin	50mA
Flash Memory	32KB (ATmega328P)
SRAM	2KB (Atmega328P)
EEPROM	1KB (Atmega328P)
Clock Speed	16MHZ
Length	68.6 mm
Width	53.4 mm
Weight	25 g



Hình 2.1 Cấu trúc phần cứng của Arduino Uno.

- Cổng USB(1): Cổng USB dùng để kết nối với máy tính và thông qua đó để upload chương trình cho Arduino từ máy tính, đồng thời cung cấp nguồn cho Arduino.
- Nguồn (2 và 4): Sử dụng jack cấm nguồn 2.1mm (cực dương ở giữa) hoặc có thể dùng chân Vin và GND để cấp nguồn cho Arduino. Board mạch hoạt động ở điện áp ngoài khoảng 5-20V, nhưng nếu cấp nguồn lớn hơn 5V thì ngõ ra chân 5V sẽ lớn hơn, không nên cấp nguồn lớn hơn 12V vì board sẽ nóng và dễ bị hỏng. Chân 5V và chân 3.3v là 2 chân lấy nguồn từ Arduino ra ngoài, không được cấp nguồn vào nó, sẽ làm hỏng.
- Chip Atmega328P(6): Có 32KB bộ nhớ flash trong đó có 0.5KB dùng cho bootloader, 2KB SRAM, 1KB EEPROM.
- Output và input(3 và 5): Arduino có 14 chân Digital với các chức năng Input và Output.

2.1.3. PHẦN MỀM MATLAB.

Matlab (Matrix Laboratory) là một môi trường tính toán số và lập trình, được thiết kế bởi công ty MathWorks.

Matlab [4] là ngôn ngữ lập trình thực hành bậc cao được sử dụng nhiều để giải các bài toán kỹ thuật. Matlab tích hợp việc tính toán thể hiện kết quả cho phép lập trình, giao diện làm việc rất dễ dàng cho người sử dụng. Dữ liệu cũng với thư viện được lập trình sẵn cho phép người dùng có được những ứng dụng như:

- Tính toán các phép toán học thông thường, tính toán ma trận,
- Lập trình tạo ra những ứng dụng mới.
- Cho phép mô phỏng các mô hình thực tế.
- Phân tích, khảo sát, hiển thị dữ liệu.
- Matlab được sử dụng trong nhiều lĩnh vực, bao gồm xử lý tín hiệu và ảnh, truyền thông, thiết kế điều khiển tự động, đo lường kiểm tra, phân tích mô hình tài chính, hay tính toán sinh học. Matlab cung cấp giải pháp chuyên dụng gọi là Toolbox. Toolbox là một tập hợp toàn diện các hàm của Matlab (M-file).

Hệ thống Matlab gồm 5 phần chính:

Môi trường làm việc: bao gồm các phương tiện cho việc quản lý các biến trong không gian làm việc Workspace cũng như xuất nhập dữ liệu. Nó cũng bao gồm các công cụ phát triển, quản lý, gỡ rối và định hình M-file.

Xử lý đồ họa: bao gồm các lệnh cao cấp cho trực quan hóa dữ liệu hai chiều và ba chiều, xử lý ảnh, ảnh động. Cung cấp các giao diện tương tác giữa người sử dụng và máy tính.

Thư viện toán học: các hàm cơ bản như cộng, trừ, nhân, chia, sin, cos..các hàm phức tạp như tính ma trận nghịch đảo, trị riêng, chuyển đổi fourier, laplace, symbolic library.

Giao diện người dùng (Application Program Interface): cho phép viết chương trình tương tác với các ngôn ngữ khác C, C++.

Simulink là một chương trình đi kèm với Matlab, là một hệ thống tương tác với việc mô phỏng các hệ thống động phi tuyết, mô phỏng mạch.

2.2.CÁC CẢM XÚC TRÊN KHUÔN MẶT

Cảm xúc trên khuôn mặt là một quy luật rất quan trọng trong giao tiếp giữa người với người. Những công nghệ giao tiếp tiến bộ nhanh chóng gần đây cùng với sựu phát triển của khoa học máy tính đã cho chúng ta những hi vọng rằng cảm xúc trên khuôn mặt sẽ trở thành một giải pháp then chốt trong lĩnh vực giao diện người máy và các hướng phát triển giao tiếp khác trong tương lai.

Trong các cách biểu đạt ngôn ngữ cơ thể, cảm xúc là nơi thể hiện rõ nhất những gì mà người khác cảm nhận được. Qua nghiên cứu của nhà tâm lý học Mehrabian năm 1968 đã chỉ ra rằng khi một thông điệp từ một người chuyển tới người khác, phần từ ngữ chỉ chiếm 7% ảnh hưởng, 38% âm lượng của giọng nói, trong khi ngôn ngữ cơ thể của người nói chiếm tới 55% ảnh hưởng của thông điệp đó [5]. Nhận dạng cảm xúc được ứng dụng trong rất nhiều lĩnh vực khác nhau như y học [6], tương tác giao diện giữa người và máy [7], chuyển động thân người [8]…

Trong đề tài đồ án tốt nghiệp này, chúng em xây dựng một hệ thống nhận dạng cảm xúc dựa trên 3 cảm xúc cơ bản trên đó là: vui, buồn, ngạc nhiên.

Các dấu hiệu nhận biết cảm xúc:

- Vui: Cảm xúc này thể hiện qua nét mặt rạng rỡ trên khuôn mặt. Nụ cười chính là dấu hiệu đơn giản nhất để nhận biết cảm xúc này.

- Buồn: Biểu hiện của cảm xúc này thông qua khuôn mặt là vùng trán nhăn lại, đôi mắt trùng xuống, không có thần thái, vẻ mặt biểu lộ rõ sự buồn khổ, sầu não kèm theo những tiếng thở dài. Nhìn vào dễ dàng tạo cảm giác lặng lẽ, trầm buồn.
- Ngạc nhiên: Đây là trạng thái dễ gây cảm giác thú vị khi nhìn vào. Sự ngạc nhiên thể hiện qua lông mày nhô cao, mắt mở to, phần hàm dưới trề xuống, miệng mở tròn ra.

2.3. PHƯƠNG PHÁP NHẬN DẠNG PCA - EIGENFACES.

2.3.1. Phương pháp nhận dạng PCA

Phân tích thành phần chính PCA (Principal Component Analysis) là một thuật toán sử dụng phép biến đổi trực giao để biến đổi một tập hợp dữ liệu từ một không gian nhiều chiều sang một không gian mới ít chiều hơn (2 hoặc 3 chiều) nhằm tối ưu hóa việc thể hiện sự biến thiên của dữ liệu. PCA một trong những ứng dụng hữu ích trong việc nhận dạng mặt và nén ảnh, là phương pháp phân tích dữ liệu nhiều biến đơn giản nhất.

Phép biến đổi tạo ra những ưu điểm như:

- Giảm số chiều của không gian chứa dữ liệu: tạo ra một ảnh mới từ ảnh ban đầu, ảnh này có kích thước nhỏ hơn nhiều so với ảnh ban đầu nhưng vẫn giữ lại những nét đặc trưng nhất từ ảnh ban đầu.
- Thay vì giữ lại các trục tọa độ của không gian cũ. PCA xây dựng những trục tọa độ mới nhưng có khả năng biểu diễn dữ liệu tương đương, và đảm bảo độ biến thiên của dữ liệu trên mỗi chiều mới.
- Trong không gian mới, các liên kết tiềm ẩn của dữ liệu có thể được khám phá, mà nếu đặt trong không gian cũ thì khó phát hiện hơn vì những liên kết này không thể hiện rõ.
- Các trục tọa độ trong không gian mới là tổ hợp tuyến tính của không gian cũ. Các trục này luôn được trực giao đôi một với nhau mặc dù trong không gian ban đầu các trục có thể không trực giao.

⇨ Nói một cách ngắn gọn, mục tiêu của PCA là tìm một không gian mới với số chiều nhỏ hơn không gian cũ. Các trục tọa độ không gian mới được xây dựng sao cho trên mỗi trục, độ biến thiên của dữ liệu trên đó là lớn nhất.

Các khái niệm toán học được sử dụng trong PCA bao gồm: Độ lệch chuẩn, phương sai, hiệp phương sai, véctơ riêng, giá trị riêng.

a. Độ lệch chuẩn.

Độ lệch chuẩn được sử dụng để đo lường mức độ phân tán của dữ liệu, kí hiệu là S. Cho một tập dữ liệu : X = [1 3 6 12 25 68 67 65 98] .

Chúng ta dùng kí hiệu X để chỉ tập dữ liệu. Để truy xuất đến từng phần tử trong tập dữ liệu, ta sử dụng chỉ số dưới, ví dụ X_1 sẽ là định danh của phân tử thứ 1, X_2 sẽ là định danh cho phần tử thứ 2.

Giá trị trung bình cho chúng ta biết giá trị trung bình của tập dữ liệu, được tính theo công thức:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n} \tag{2.1}$$

- Trong đó: $\bar{X}$ là giá trị trung bình.
  X_i là giá trị thứ i.
  i là số thứ tự phần tử trong mảng X
  n là tổng số phần tử có trong mảng

Để tính độ lệch chuẩn chúng ta sử dụng công thức sau:

$$S = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}} \tag{2.2}$$

- Trong đó: $\bar{X}$ là giá trị trung bình.
  X_i là giá trị thứ i.
  i là số thứ tự phần tử trong mảng X
  n là tổng số phần tử có trong mảng

Ví dụ: Cho tập dữ liệu X = [2 10 14 22] và Y= [5 12 13 18]

Muốn tính độ lệch chuẩn cho tập dữ liệu X:

Bước 1: Ta dùng công thức 2.1 để tính giá trị trung bình của dữ liệu X:

$$\bar{X} = \frac{2+10+14+22}{4} = 12$$

Bước 2: Ta sử dụng công thức 2.2 để tính độ lệch chuẩn:

$$S_X = \sqrt{\frac{(2-12)^2+(10-12)^2+(14-12)^2+(22-12)^2}{4-1}} = 8.3$$

Tương tự ta tính cho tập dữ liệu Y:

Bước 1: Tính giá trị trung bình cho tập dữ liệu Y:

$$\bar{Y} = \frac{5+12+13+18}{4} = 12$$

Tính độ lệch chuẩn:

$$S_X = \sqrt{\frac{(5-12)^2+(12-12)^2+(13-12)^2+(18-12)^2}{4-1}} = 5.4$$

⇨ Ta thấy $S_X > S_Y$, cho thấy tập X có độ phân tán dữ liệu lớn hơn tập Y mặc dù cả hai đều có chung giá trị trung bình là 12.

b. Phương sai.

Phương sai dùng để đo lường độ phân tán dữ liệu của một tập dữ liệu.

Công thức của phương sai là :

$$S^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1} \quad (2.3)$$

- Trong đó: $\bar{X}$ là giá trị trung bình.

  X_i là giá trị thứ i.

  i là số thứ tự phần tử trong mảng X

  n là tổng số phần tử có trong mảng

c. Hiệp phương sai.

Hiệp phương sai: là một giá trị đo để xem xét mối liên hệ giữa 2 biến. Kí hiệu là Cov(X,Y) với X,Y là 2 biến.

Công thức tính hiệp phương sai:

$$\text{Cov(X,Y)} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2(Y_i - \bar{Y})^2}{n-1} \quad (2.4)$$

Giả sử chúng ta nghiên cứu về mối liên hệ giữa số giờ của học sinh ôn tập và điểm của họ. Vậy là dữ liệu ta cần thu thập có 2 chiều, chiều X là số giờ ôn tập và chiều Y là điểm. Nếu hiệp phương sai là một giá trị dương cho thấy số giờ ôn thi tăng thì điểm cũng tăng ngược lại thì số giờ học tăng thì điểm số giảm, cuối cùng hiệp phương sai bằng không thì cho thấy chúng không liên quan gì đến nhau.

Đặc tính của hiệp phương sai là chỉ tính toán với dữ liệu có hai chiều và Cov(X,Y) và Cov(Y,X) bằng nhau.

d. Ma trận hiệp phương sai

Hiệp phương sai chỉ dùng để tính toán cho dữ liệu có 2 chiều. Vậy đối với dữ liệu có nhiều chiều (n>2), chúng ta có rất nhiều hiệp phương sai. Ma trận hiệp phương sai là ma trận chứa tất cả các hiệp phương sai có thể được tạo ra. Dữ liệu 3 chiều x,y,z thì ma trận hiệp phương sai sẽ là:

$$C = \begin{pmatrix} \mathrm{Cov}(X,X) & \mathrm{Cov}(X,Y) & \mathrm{Cov}(X,Z) \\ \mathrm{Cov}(Y,X) & \mathrm{Cov}(Y,Y) & \mathrm{Cov}(Y,Z) \\ \mathrm{Cov}(Z,X) & \mathrm{Cov}(Z,Y) & \mathrm{Cov}(Z,Z) \end{pmatrix} \quad (2.5)$$

e. Vector riêng.

Vector riêng của một ma trận vuông (n x n) là vector mà khi nhân ma trận đó với vector riêng thì sẽ thu được 1 vector mà giá trị của từng phần tử tương ứng sẽ bằng k lần so với giá trị của phần tử của eigenvector.

Ví dụ: ta có một phép tính như sau:

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

Trong đó: Vector $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ gọi là vector riêng

Các tính chất của vectơ riêng:

- Chỉ các ma trận vuông (n x n) mới có vectơ riêng.
- Không phải mọi ma trận vuông đều có vectơ riêng.
- Nếu 1 ma trận vuông (n x n) có vectơ riêng thì sẽ có n vectơ riêng.
- Nếu nhân vectơ riêng với 1 số thì kết quả sau khi nhân với ma trận chuyển đổi, vectơ kết quả vẫn là vectơ ban đầu.
- Tất cả các vector riêng của 1 ma trận đều trực giao với nhau

f. Giá trị riêng.

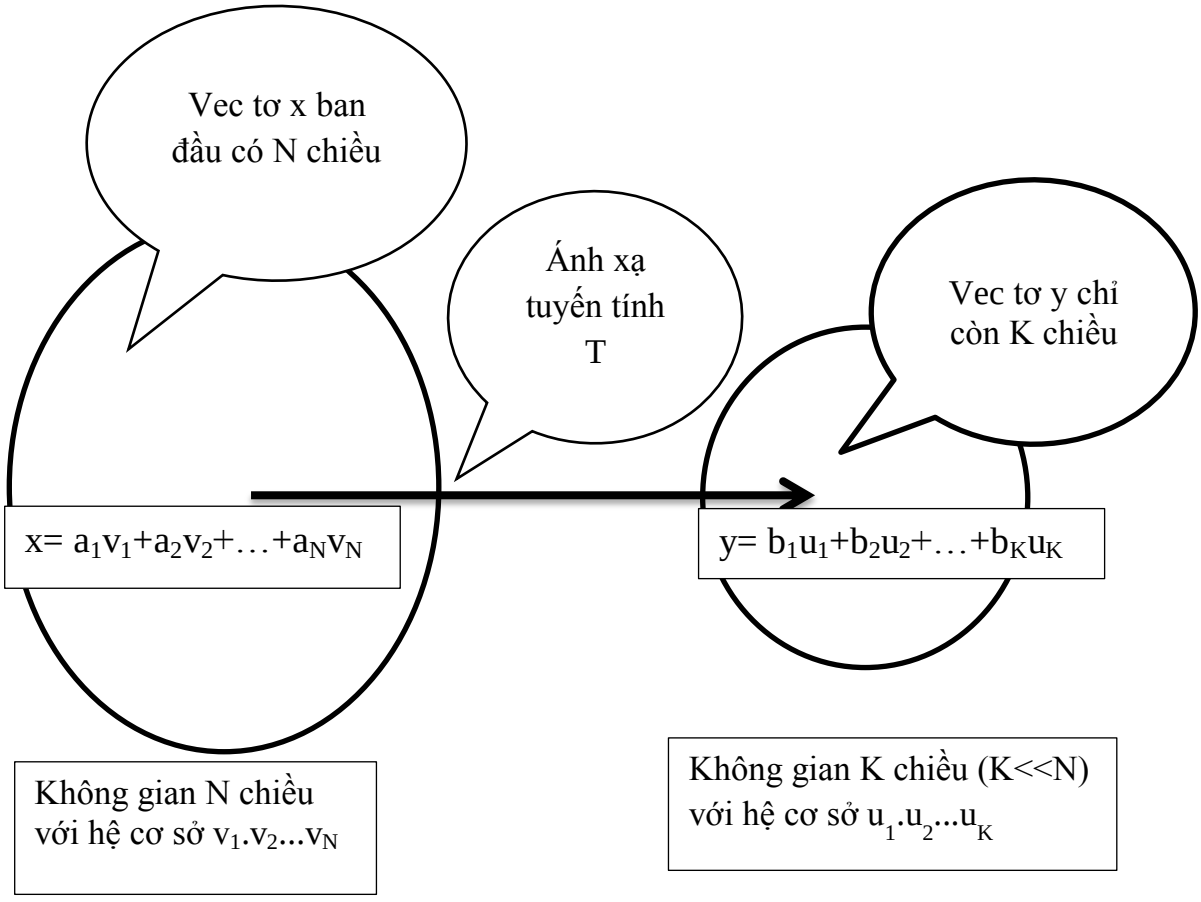
Giá trị riêng và vector riêng là hai khái niệm liên hệ mật thiết với nhau và chúng luôn đi thành cặp. Ở ví dụ trên ta được giá trị riêng là 4 .

2.3.2 Eigenfaces trong nhận dạng cảm xúc trên khuôn mặt.

Trong bài toán nhận dạng, thông thường cần phải "nghiên cứu" dữ liệu trước

khi xây dựng các mô hình dựa trên dữ liệu đó. Tuy nhiên đôi khi dữ liệu có số chiều lớn, trong không gian 2 hay 3 chiều, do đó cần phải tìm cách đưa dữ liệu về không gian có số chiều nhỏ hơn. PCA đã giải quyết bài toán trên.

Eigenfaces là một phương pháp đã được nghiên cứu về nhận dạng trên khuôn mặt dựa trên phân tích các thành phần chính (PCA). Eigenface là phương pháp áp dụng trực tiếp phép phân tích các thành phần PCA, nó đã được áp dụng rất nhiều vào biểu diễn, phát hiện và nhận dạng mặt. Ưu điểm của phương pháp này là biểu diễn được toàn bộ ảnh và có độ nén rất tốt (loại bỏ nhiễu và dư thừa). Eigenfaces được phát triển từ các chuyên đề nghiên cứu của Turk & Pentland, mặc dù phương pháp này hiện đã được thay thế bằng những phương pháp chính xác hơn nhưng vẫn được dùng để so sánh hiệu suất với các phương pháp khác. Eigenfaces là một cách để làm quen với nhận dạng khuôn mặt dựa trên không gian con.



Hình 2.2 Không gian dữ liệu mới với các Eigen vector.

2.3.3. Các bước cơ bản trong Eigenfaces

a. Xử lý ảnh đầu vào.

Bước đầu tiên cần làm là phải có một cơ sở dữ liệu cho riêng mình. Các yêu cầu chính của cơ sở dữ liệu:

- Chuyển ảnh màu sang ảnh xám. Thống nhất về độ phân giải (600x800), ảnh phải được cắt, chỉnh sửa sao cho có cùng kích thước nhất định.
- Tạo cơ sở dữ liệu dựa trên các đặc trưng trên khuôn mặt như mắt, mũi và miệng.
- Ở đồ án này, tác giả xây dựng cơ sở dữ liệu: 390 hình ảnh với 3 trạng thái (vui, ngạc nhiên,buồn).

Các hình ảnh I_1, I_2…I_n được chuyển thành một vectơ cột Γ_i và được nạp vào một ma trân kích thước M x N với N là số lượng điểm ảnh và M là tổng số hình ảnh.

Để đơn giản dùng 5 ảnh minh họa với kích thước 3x3:

$$\Gamma_1 = \begin{bmatrix} 89 \\ 91 \\ 23 \\ 122 \\ 133 \\ 122 \\ 86 \\ 67 \\ 37 \end{bmatrix} \quad \Gamma_2 = \begin{bmatrix} 91 \\ 92 \\ 20 \\ 126 \\ 123 \\ 122 \\ 87 \\ 67 \\ 36 \end{bmatrix} \quad \Gamma_3 = \begin{bmatrix} 83 \\ 78 \\ 19 \\ 95 \\ 118 \\ 117 \\ 78 \\ 54 \\ 28 \end{bmatrix} \quad \Gamma_4 = \begin{bmatrix} 83 \\ 90 \\ 22 \\ 112 \\ 128 \\ 114 \\ 78 \\ 66 \\ 20 \end{bmatrix} \quad \Gamma_5 = \begin{bmatrix} 94 \\ 92 \\ 18 \\ 130 \\ 136 \\ 128 \\ 84 \\ 71 \\ 41 \end{bmatrix}$$

b. Huấn luyện ảnh.

Để tạo một cơ sở dữ liệu huấn luyện cần các bước sau:

Bước 1: Tính vector khuôn mặt trung bình Ψ theo công thức:

$$\Psi = \frac{1}{M}\sum_{i=1}^{M}\Gamma_i \tag{2.6}$$

Ta tính được Ψ cho 5 ảnh minh họa trên.

$$\Psi = \begin{bmatrix} 88 \\ 88.6 \\ 20.4 \\ 117 \\ 127.6 \\ 120.6 \\ 82.6 \\ 65 \\ 32.4 \end{bmatrix}$$

Bước 2: Trừ vector ảnh đầu vào cho vector ảnh khuôn mặt trung bình được Φ_i :

$$\Phi_i = \Gamma_i - \Psi \tag{2.7}$$

Ta tính được:

$$\Phi_1 = \begin{bmatrix} 1 \\ 2.4 \\ 2.6 \\ 5 \\ 5.4 \\ 1.4 \\ 3.4 \\ 2 \\ 4.6 \end{bmatrix} \quad \Phi_2 = \begin{bmatrix} 3 \\ 3.4 \\ -0.4 \\ 9 \\ -4.6 \\ 1.4 \\ 4.4 \\ 2 \\ 3.6 \end{bmatrix} \quad \Phi_3 = \begin{bmatrix} -5 \\ -10.6 \\ -1.4 \\ -22 \\ -9.6 \\ 3.6 \\ -4.6 \\ -11 \\ -4.4 \end{bmatrix} \quad \Phi_4 = \begin{bmatrix} -5 \\ 1.4 \\ 1.6 \\ -5 \\ 0.4 \\ 6.6 \\ -4.6 \\ 1 \\ -12.4 \end{bmatrix} \quad \Phi_5 = \begin{bmatrix} 6 \\ 3.4 \\ -2.4 \\ 13 \\ 8.4 \\ 7.4 \\ 1.4 \\ 6 \\ 8.6 \end{bmatrix}$$

Bước 3: Tính ma trận hiệp phương sai (Covariance) C:

C sẽ có kích thước N^2 x N^2.

$$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T = A.A^T \tag{2.8}$$

$$A = [\ \Phi_1\ \Phi_2 \dots \Phi_M] \tag{2.9}$$

A sẽ có kích thước N^2 x M.

$$A = \begin{bmatrix} 1 & 3 & -5 & -5 & 6 \\ 2.4 & 3.4 & -10.6 & 1.4 & 3.4 \\ 2.6 & -0.4 & -1.4 & 1.6 & -2.4 \\ 5 & 9 & -22 & -5 & 13 \\ 5.4 & -4.6 & -9.6 & 0.4 & 8.4 \\ 1.4 & 1.4 & 3.6 & 6.6 & 7.4 \\ 3.4 & 4.4 & -4.6 & -4.6 & 1.4 \\ 2 & 2 & -11 & 1 & 6 \\ 4.6 & 3.6 & -4.4 & -12.4 & 8.6 \end{bmatrix}$$

Bước 4: Tính các vector riêng u_i tương ứng với các giá trị riêng của hiệp phương sai $A.A^T$(C có kích thước N^2xN^2):

Xét ma trận $A^T.A$ có kích thước MxM.

Tìm vector riêng v_i của ma trận $A^T.A$ có kích thước Mx1.

$$v_1 = \begin{bmatrix} 9.06 \\ 9.55 \\ -28.84 \\ -10.35 \\ 20.58 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0.26 \\ 2.25 \\ 7.51 \\ -12.89 \\ 2.87 \end{bmatrix} \quad v_3 = \begin{bmatrix} 2.18 \\ -8.04 \\ 1.11 \\ 0.28 \\ 4.47 \end{bmatrix} \quad v_4 = \begin{bmatrix} 4.41 \\ -0.41 \\ -0.41 \\ -0.84 \\ -2.74 \end{bmatrix}$$

Các giá trị riêng ứng với các vector riêng của ma trận $A^T.A$.

$\lambda_1 = 384.15$

$\lambda_2 = 58.98$

$\lambda_3 = 22.67$

$\lambda_4 = 7$

Tìm vector riêng u_i có kích thước N^2x1:

$$u_i = Av_i \tag{2.10}$$

Chú ý nên chuẩn hóa các vector u_i ($\|u_i\| = 1$), nghĩa là: $u_i = \frac{u_i}{\|u_i\|}$.

Vector u_i đã chuẩn hóa:

$$u_1 = \begin{bmatrix} 0.23 \\ 0.27 \\ -0.01 \\ 0.71 \\ 0.3 \\ 0.23 \\ 0.18 \\ 0.3 \\ 0.33 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.22 \\ -0.34 \\ -0.16 \\ -0.18 \\ -0.26 \\ 0.35 \\ 0.17 \\ -0.31 \\ 0.68 \end{bmatrix} \quad u_3 = \begin{bmatrix} -0.02 \\ -0.2 \\ -0.03 \\ -0.32 \\ 0.84 \\ 0.21 \\ -0.31 \\ 0.04 \\ 0.12 \end{bmatrix} \quad u_4 = \begin{bmatrix} -0.25 \\ 0.11 \\ 0.62 \\ -0.15 \\ 0.22 \\ -0.28 \\ 0.54 \\ -0.17 \\ -0.26 \end{bmatrix}$$

Bước 5: Chỉ giữ lại K vector riêng trong số M vector nói trên với K được xác định như sau:

- Sắp xếp theo thứ tự dãy giảm dần các eigenvalues tìm được.
- Theo dõi sự biến thiên của dãy trên, khi không còn biến thiên (hoặc xấp xỉ bằng không) thì lúc đó ta đã chọn đủ K.

Ở ví dụ trên có 4 giá trị riêng sắp xếp theo thứ tự giảm dần chọn K = 4.

Bước 6: Biểu diễn các khuôn mặt có sẵn (tập huấn luyện) vào trong không gian vector mới. Mỗi khuôn mặt Φ_i trong tập huấn luyện có thể được biểu diễn lại là 1 tổ hợp tuyến tính của K vector riêng giới hạn:

$$\Omega_i = u_j^T \Phi_i \qquad \text{i=1,2,…,M} \qquad \Omega_i = \begin{bmatrix} u_1^T.\Phi_i \\ u_2^T.\Phi_i \\ u_3^T.\Phi_i \\ \cdots \cdots \\ u_K^T.\Phi_i \end{bmatrix}$$

$$\text{J= 1,2,…,K} \qquad (2.11)$$

c. Nhận dạng cảm xúc.

Bước 1: Chuyển ảnh cần nhận dạng sang chiều không gian mới như đã làm với ảnh huấn luyện.

Bước 2: Tìm khuôn mặt thứ *l* trong tập mẫu có khoảng cách gần nhất với khuôn mặt có cảm xúc cần nhận dạng.

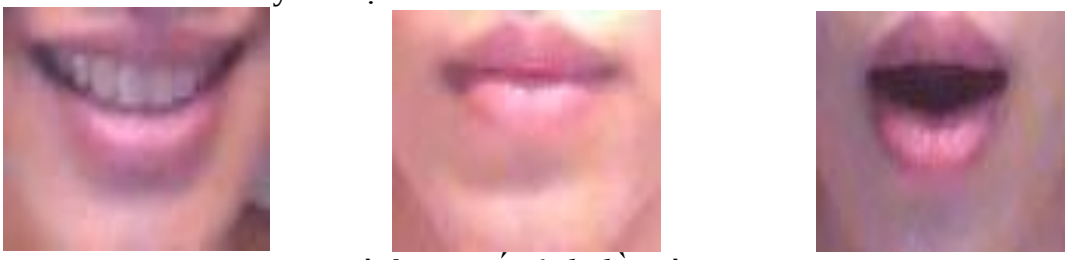
$$e_r = \min_l \|\Omega - \Omega^l\| \qquad (2.12)$$

Ta có thể tính e_r bằng khoảng cách Euclid.

2.3.4. Phân tích thành phần chính (PCA)

Các bước cơ bản trong PCA:

- Bước 1 Lấy dữ liệu làm mẫu.



Hình 2.3 Lấy ảnh đầu vào

- Bước 2 : Trừ trung bình mẫu

Hình 2.4 Ảnh trong tập mẫu

Với mỗi chiều dữ liệu giả sử ở chiều x, ta đều có 1 trung bình mẫu

Công việc ở bước này chỉ là trừ tất cả giá trị trong mẫu cho trung bình mẫu

- Bước 3: Tính ma trận hiệp phương sai.
- Bước 4:Tính các vector riêng,giá trị riêng của ma trận hiệp phương sai.
- Bước 5: Chọn các thành phần chính

=> Trong bước này tùy thuộc vào số lượng thành phần chính cần lấy, ta lấy lần lượt các thành phần vector riêng tương ứng với các giá trị cao nhất.

2.3.5 Nhận dạng cảm xúc dựa trên PCA:

2.3.5.1 Trích chọn đặc trưng:

Xây dựng một tập các vector đặc trưng (S1,S2,…Sk) cho mỗi hình huấn luyện.

Mỗi cảm xúc gồm một tập huấn luyện

- Cảm xúc vui:

  $I_{(vui\ 1)} = (b_{vui\ 1\ 1}, b_{vui\ 1\ 2}, b_{vui\ 1\ 3} \ldots . b_{vui\ 1\ n})$

  $I_{(vui\ 2)} = (b_{vui\ 2\ 1}, b_{vui\ 2\ 2}, b_{vui\ 2\ 3} \ldots . b_{vui\ 2\ n})$

  $I_{(vui\ 3)} = (b_{vui\ 3\ 1}, b_{vui\ 3\ 2}, b_{vui3\ 3} \ldots . b_{vui\ 3\ n})$

  .

  .

  $I_{(vui\ m)} = (b_{vui\ m\ 1}, b_{vui\ m\ 2}, b_{vui\ m\ 3} \ldots . b_{vui\ m\ n})$
- Cảm xúc buồn:

  $I_{(buồn\ 1)} = (b_{buồn\ 1\ 1}, b_{buồn\ 1\ 2}, b_{buồn\ 1\ 3} \ldots . b_{buồn\ 1\ n})$

  $I_{(buồn\ 2)} = (b_{buồn\ 2\ 1}, b_{buồn\ 2\ 2}, b_{buồn\ 2\ 3} \ldots . b_{buồn\ 2\ n})$

$I_{(\text{buồn } 3)} = (b_{\text{buồn } 3\ 1}, b_{\text{buồn } 3\ 2}, b_{\text{buồn3 } 3} \ldots . b_{\text{buồn } 3\ n})$

.

.

$I_{(\text{buồn } m)} = (b_{\text{buồn } m\ 1}, b_{\text{buồn } m\ 2}, b_{\text{buồn } m\ 3} \ldots . b_{\text{buồn } m\ n})$

Cảm xúc ngạc nhiên:

$I_{(\text{ngạc nhiên } 1)} = (b_{\text{ngạc nhiên } 1\ 1}, b_{\text{ngạc nhiên } 1\ 2}, b_{\text{ngạc nhiên } 1\ 3} \ldots . b_{\text{ngạc nhiên } 1\ n})$

$I_{(\text{ngạc nhiên } 2)} = (b_{\text{ngạc nhiên } 2\ 1}, b_{\text{ngạc nhiên } 2\ 2}, b_{\text{ngạc nhiên } 2\ 3} \ldots . b_{\text{ngạc nhiên } 2\ n})$

$I_{(\text{ngạc nhiên } 3)} = (b_{\text{ngạc nhiên } 3\ 1}, b_{\text{ngạc nhiên } 3\ 2}, b_{\text{ngạc nhiên3 } 3} \ldots . b_{\text{ngạc nhiên } 3\ n})$

.

.

$I_{(\text{ngạc nhiên } m)} = (b_{\text{ngạc nhiên } m\ 1}, b_{\text{ngạc nhiên } m\ 2}, b_{\text{ngạc nhiên } m\ 3} \ldots . b_{\text{ngạc nhiên } m\ n})$

Với một ảnh đầu vào bất kí muốn nhận dạng cảm xúc, đầu tiên phải qua bước tiền xử lý là nhận diện được vùng chứa khuôn mặt và xác định được miệng trong khuôn mặt đó. Từ đó ta sử dụng PCA cho ra tập các giá trị riêng như sau:

Cảm xúc cần nhận dạng

$I_{(\text{nhan dang})} = (b_{\text{nhan dang } 1}, b_{\text{nhan dang } 2}, b_{\text{nhan dang } 3} \ldots . b_{\text{nhan dang } n})$

2.3.5.2 Quá trinh nhận dạng

Ta lần lượt tính các khoảng cách Euclid từ ảnh cần nhận dạng đến mỗi ảnh trong tập huấn luyện

$S_{(\text{vui } 1)} = (b_{\text{vui } 1\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{vui } 1\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{vui } 1\ n} - b_{\text{nhan dang } n})^2$

$S_{(\text{vui } 2)} = (b_{\text{vui } 2\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{vui } 2\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{vui } 2\ n} - b_{\text{nhan dang } n})^2$

….

$S_{(\text{vui } m)} = (b_{\text{vui } m\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{vui } m\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{vui } m\ n} - b_{\text{nhan dang } n})^2$

$S_{(\text{buồn } 1)} = (b_{\text{buồn } 1\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{buồn } 1\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{buồn } 1\ n} - b_{\text{nhan dang } n})^2$

$S_{(\text{buồn } 2)} = (b_{\text{buồn } 2\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{buồn } 2\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{buồn } 2\ n} - b_{\text{nhan dang } n})^2$

….

$S_{(\text{buồn } m)} = (b_{\text{buồn } m\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{buồn } m\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{buồn } m\ n} - b_{\text{nhan dang } n})^2$

$S_{(\text{ngạc nhiên } 1)} = (b_{\text{ngạc nhiên } 1\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{ngạc nhiên } 1\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{ngạc nhiên } 1\ n} - b_{\text{nhan dang } n})^2$

$S_{(\text{ngạc nhiên } 2)} = (b_{\text{ngạc nhiên } 2\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{ngạc nhiên } 2\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{ngạc nhiên } 2\ n} - b_{\text{nhan dang } n})^2$

….

$S_{(\text{ngạc nhiên } m)} = (b_{\text{ngạc nhiên } m\ 1} - b_{\text{nhan dang } 1})^2 + (b_{\text{ngạc nhiên } m\ 2} - b_{\text{nhan dang2}})^2 + \ldots . + (b_{\text{ngạc nhiên } m\ n} - b_{\text{nhan dang } n})^2$

Sau đó cảm xúc sẽ của ảnh cần nhận dạng sẽ được xác định bằng cảm xúc ảnh trong tập huấn luyện có khoảng cách mà khoảng cách Euclid từ ảnh đó đến ảnh cần nhận dạng là bé nhất

2.4. Các hàm xử lý trong matlab

Các hàm chính hiển thị ảnh trong matlab:

- Hàm **image(x,y,c)** hiển thị hình ảnh biểu diễn bởi ma trận c kích thước m x n lên hệ trục tọa độ, x,y là các véctơ xác định vị trí của các pixel c(m,n).

- Hàm **imagesc** tương tự hàm image, dữ liệu ảnh sẽ được co giãn để sử dụng toàn bộ bản đồ màu hiện hành.
- 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 khác được sử dụng trong đề tài

- 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…
- **textscan()** đọc dữ liệu từ file .txt.
- **length()** lấy chiều dài.
- **zeros(m,n)** tạo ma trận m hàng n cột.
- **[...] = princomp(X,'econ')** hàm thực hiện phân tích thành phần chính PCA kết quả trả về vector riêng, giá trị riêng và không gian cơ sở dữ liệu mới chứa các vector riêng.
- **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
- **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 .
- **Min(X):** Trả về vị trí của phần tử nhỏ nhất của ma trận X.

Bảng 2.2 Các hàm xử lý hình ảnh khác trong Matlab.

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
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
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	
Imcrop	Trích xuất một phần ảnh
Imresize	Thay đổi kích thước ảnh
Imrotate	Thực hiện phép quay ảnh
Imtranform	Thực hiện phép biến đổi hình học tổng quát

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

3.1 GIỚI THIỆU

Nhận dạng cảm xúc trên khuôn mặt được hy vọng sẽ được áp dụng cho nhiều ứng dụng trong cuộc sống. Trong khóa luận này nhóm nghiên cứu chỉ xây dựng một chương trình mô hình nhỏ để minh họa cho các lý thuyết ở trên. Cụ thể, đây là chương trình huấn luyện và nhận dạng cảm xúc trên khuôn mặt mặt được viết trên phần mềm Matlab, sử dụng các gói hỗ trợ để điều khiển minh họa các ứng dụng trong thực thế trên kit Arduino.

Nhiệm vụ chính của chương trình là nhận dạng cảm xúc trên khuôn mặt từ ảnh đầu vào (ảnh chụp sẵn có trong file hoặc lấy trực tiếp từ webcam). Nhận dạng bằng cách so sánh với một cơ cở dữ liệu huấn luyện có sẵn đưa ra kết quả. Cuối cùng nhúng dữ liệu cảm xúc nhận dạng được xuống Kit Arduino điều khiển thiết bị ứng với mỗi cảm xúc đã nhận dạng được.

Như vậy, ở chương này nhóm sẽ thực hiện các nội dung:

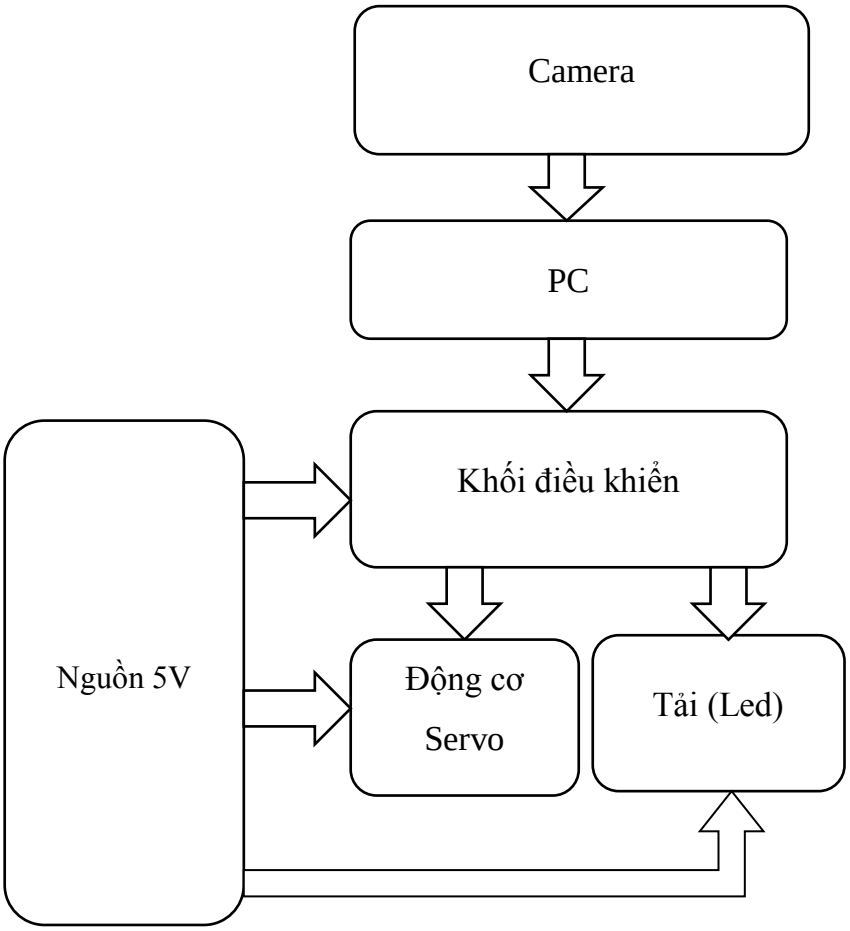
- Cài đặt các gói hỗ trợ cho phần mềm Matlab
- Giải thích sơ đồ nguyên lý của kit Arduino.
- Tính toán và thiết kế sơ đồ khối cho toàn hệ thống

3.2 TÍNH TOÁN VÀ THIẾT KẾ HỆ THỐNG

3.2.1 Thiết kế sơ đồ khối hệ thống

Với mục tiêu xây dựng cơ sở dữ liệu huấn luyện cho hệ thống và chương trình nhận dạng ảnh đầu vào từ cơ sở dữ liệu huấn luyện.

Nhóm xây dựng sơ đồ khối toàn bộ hệ thống như sau:



Hình 3.1 Sơ đồ khối của hệ thống

Khối Camera: là khối có chức năng lấy ảnh từ webcam máy tính.

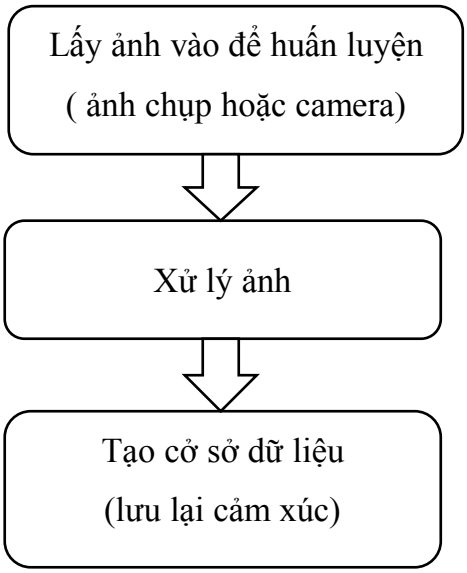
Khối PC: là khối có chức năng nhận ảnh từ khối ảnh đầu vào, tiền xử lý ảnh trước khi xử lý, so sánh ảnh cần nhận dạng với ảnh dữ liệu đã được tập huấn thông qua phần mềm Matlab. Sau khi xử lí xong sẽ trả về các giá trị tương ứng với cái trạng thái cảm xúc khuôn mặt, đồng thời ứng với mỗi giá trị cảm xúc đó khối xử lý trung tâm sẽ đưa dữ liệu xuống khối điều khiển để điều khiển thiết bị ra.

Khối điều khiển: có chức năng nhận tín hiệu từ máy tính và điều khiển Servo, Led ứng với mỗi tín hiệu được gửi đến từ máy tính.

Khối Nguồn: có chức năng nhận cung cấp nguồn cho khối điều khiển, động cơ servo và tải (Led).

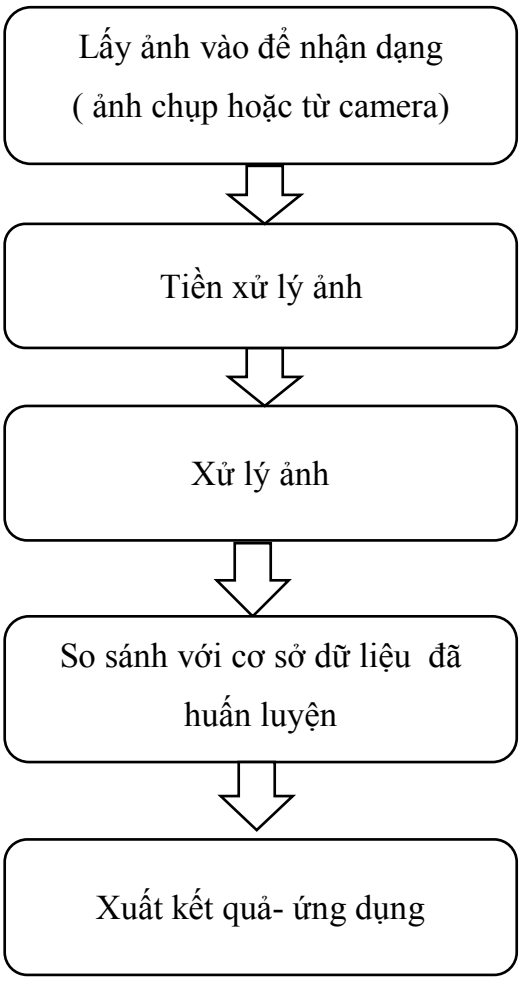
Hệ thống được chia thành thành 2 phần chính: Huấn luyện và nhận dạng.

- Huấn luyện gồm các quá trình: Lấy ảnh đầu vào, xử lý ảnh đầu vào, tạo cơ sở dữ liệu.



Hình 3.2. Sơ đồ khối quá trình tạo cơ sở dữ liệu huấn luyện.

- Nhận dạng gồm các quá trình: Lấy ảnh đầu vào, xử lý đầu vào, so sánh với cơ sở dữ liệu huấn luyện, đưa ra kết quả



Hình 3.3. Sơ đồ khối quá trình nhận dạng.

Khối lấy ảnh đầu vào (bao gồm khối lấy ảnh đầu vào để huấn luyện và khối lấy ảnh đầu vào để nhận dạng). Chương trình nhận đầu vào từ ảnh chụp hoặc từ camera, tuy nhiên việc nhận dạng cảm xúc khuôn mặt được thực hiện trên các bức ảnh, do đó việc lấy ảnh từ camera ta phải chuyển thành các ảnh tĩnh và xử lý trên từng ảnh tĩnh. Khi đã có ảnh đầu vào, tiếp tục chuyển ảnh cho quá trình xử lý tiếp theo.

Khối xử lý ảnh đầu vào: Hệ thống đòi hỏi ảnh đầu vào, ảnh huấn luyện và ảnh nhận dạng phải có cùng kích thước với nhau. Do đó, ảnh sau khi lấy ảnh, ảnh phải được cắt, chỉnh sửa sao cho có cùng kích thước nhất định.

Khối tạo cơ sở dữ liệu: Các ảnh sau khi được xử lý trong quá trình huấn luyện sẽ được lưu cảm xúc tương ứng vào cơ sở dữ liệu để phục vụ nhận dạng.

Khối so sánh với cơ sở dữ liệu đã huấn luyện: Các ảnh sau khi được xử lý cảm xúc ở quá trình nhận dạng sẽ được đem đi so sánh với ảnh được huấn luyện trong cơ sở dữ liệu để đưa ra kết luận.

Khối xuất kết quả- ứng dụng: Hệ thống sẽ đưa ra thông tin ảnh cần nhận dạng có cảm xúc như thế nào, sau đó chạy ứng dụng trên kit Arduino.

3.2.2.Thiết kế các khối hệ thống

Hệ thống được xây dựng toàn bộ trên nền Matlab nhúng dữ liệu xuống kit Arduino nên quá trình tính toán và thiết kế được thực hiện chủ yếu trên phần mềm.

a. Thiết kế khối đầu lấy ảnh đầu vào

Ở khối đầu vào nhóm sử dụng 2 bộ ảnh để luyện:

- Một lấy ảnh từ ảnh chụp trước đó, có sẵn,
- Một chụp từ camera webcam của máy tính hoặc camera.

Mỗi bộ ảnh có tất cả 390 ảnh bao gồm: 130 ảnh cho cảm xúc vui, 130 ảnh cho cảm xúc buồn, 130 ảnh cho cảm xúc ngạc nhiên và. Các bức ảnh này phải có cùng kích thước, được chụp ở cùng một khoảng cách để hổ trợ cho việc xử lý vào cở sở dữ liệu phục vụ cho nhận dạng.

Khi lấy ảnh chụp từ camera ta dùng các hàm:

- *camlist= webcamlist*: hiển thông tin camera.
- *cam= webcam()* : kết nối với webcam.
- *preview(cam)* : mở cửa sổ webcam để ảnh

- *snapshot(cam):* chụp ảnh

b. Thiết kế khối xử lý đầu vào

Ở khối xử lý đầu vào: xây dựng một chương trình để xử lý bằng các lệnh, hàm được thư viện Matlab hỗ trợ. Sau khi có ảnh đầu vào chuyển đổi chúng sang cùng kích thước, chuyển đổi hình ảnh màu RGB sang xám bằng cách sử dụng chức năng *imresize(), rgb2gray().* Tiếp theo tách ảnh khuôn mặt bằng công cụ Image Processing toolbox bằng hàm *regionprops(L,properties).*

c. Thiết kế khối tạo cơ sở dữ liệu

Ở khối này hệ thống sẽ lưu trữ các các ảnh đầu vào đã được xử lý trước đó để phục vụ quá trình nhận dạng bằng cách tạo một file.txt lưu trữ tên cảm xúc từng ảnh đã được xử lý.

d. Thiết kế so sánh với cơ sở dữ liệu huấn luyện.

Ở khối này hệ thống sử dụng phương pháp Eigenfaces bằng cách chuyển các ảnh trong cơ sở dữ liệu thành các véctơ riêng, biểu diễn chúng trong một không gian mới, sau đó làm tương tự cho ảnh cần nhận dạng. Tìm khoảng cách gần nhất giữa ảnh cần nhận dạng với ảnh trong cơ sở dữ liệu bằng phương pháp Euclid và ảnh đó chính là cảm xúc cần nhận dạng..

e. Thiết kế khối xuất kết quả- ứng dụng.

Khối này sẽ đọc kết quả cảm xúc nhận dạng được từ khối so sánh với cơ sở dữ liệu huấn luyện, tương ứng với mỗi cảm xúc điều khiển ứng dụng trên kit Arduino:

Cảm xúc vui điều khiển mở cửa.

Cảm xúc buồn điều khiển mở đèn

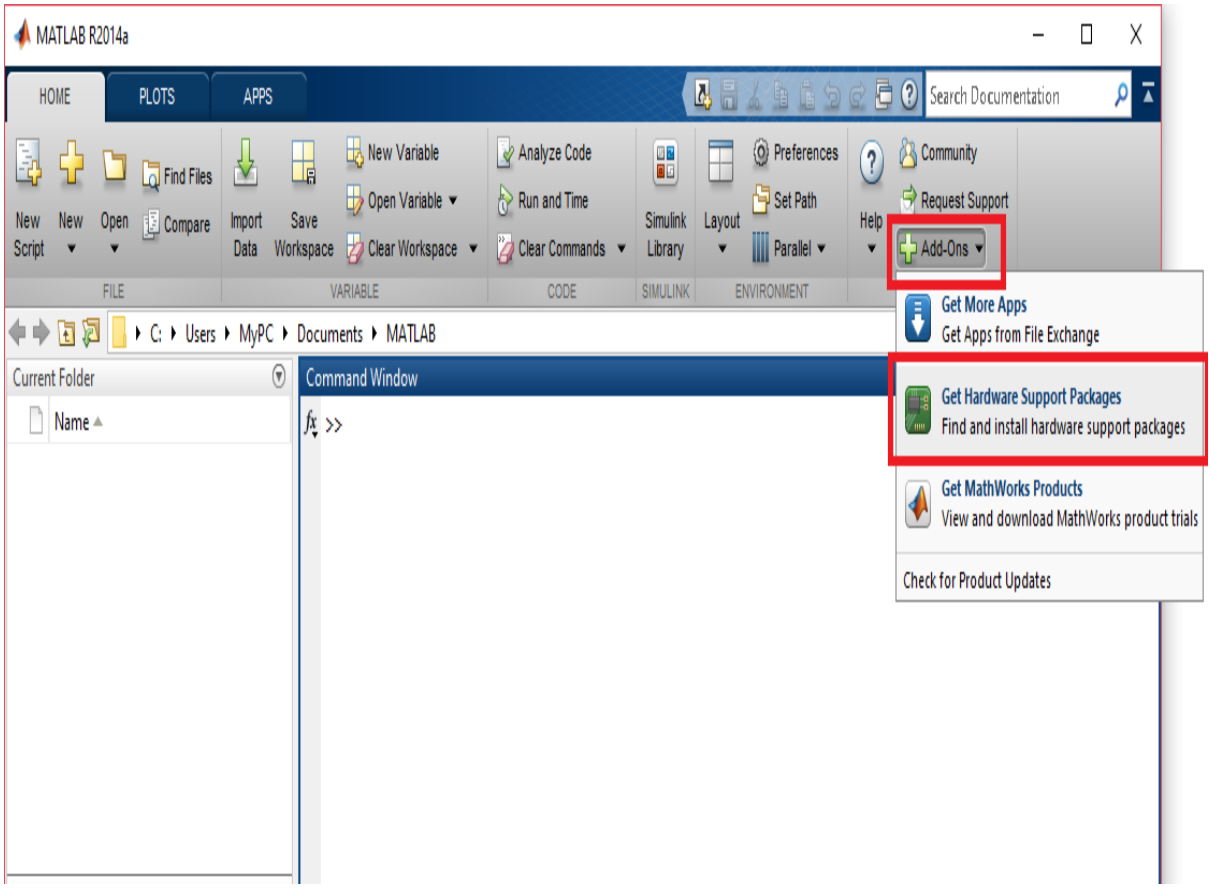
Cảm xúc ngạc nhiên điều khiển đóng cửa tắt đèn

3.3. CÀI ĐẶT CÁC GÓI HỖ TRỢ PHẦN CỨNG CHO MATLAB

3.3.1. Kết nối Arduino với Matlab

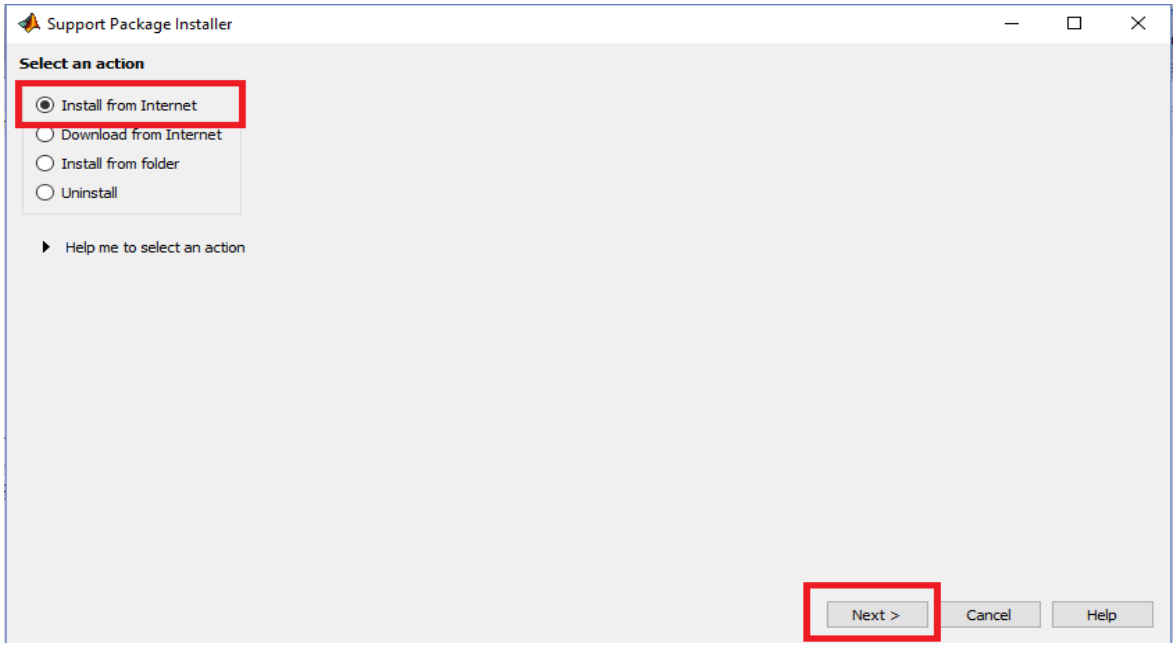
Đầu tiên tiến hành cài đặt gói Arduino trên nền Matlab. Matlab có hỗ trợ các Package từ phần cứng Ardunio và giao tiếp với nhau thông qua cổng USB. Các gói Package được tạo ra dựa trên hoạt động của các chương trình trên board, nó có hỗ trợ trên môi trường như Windows, Mac OS, Linux. Trong phần này nhóm chỉ tập trung vào giao tiếp giữa Ardunio và Matlab.Để cài đặt gói Package làm như sau:

- Bước 1: Từ giao diện Matlab, click vào **Add-Ons** chọn **Get Hardware Support Package**



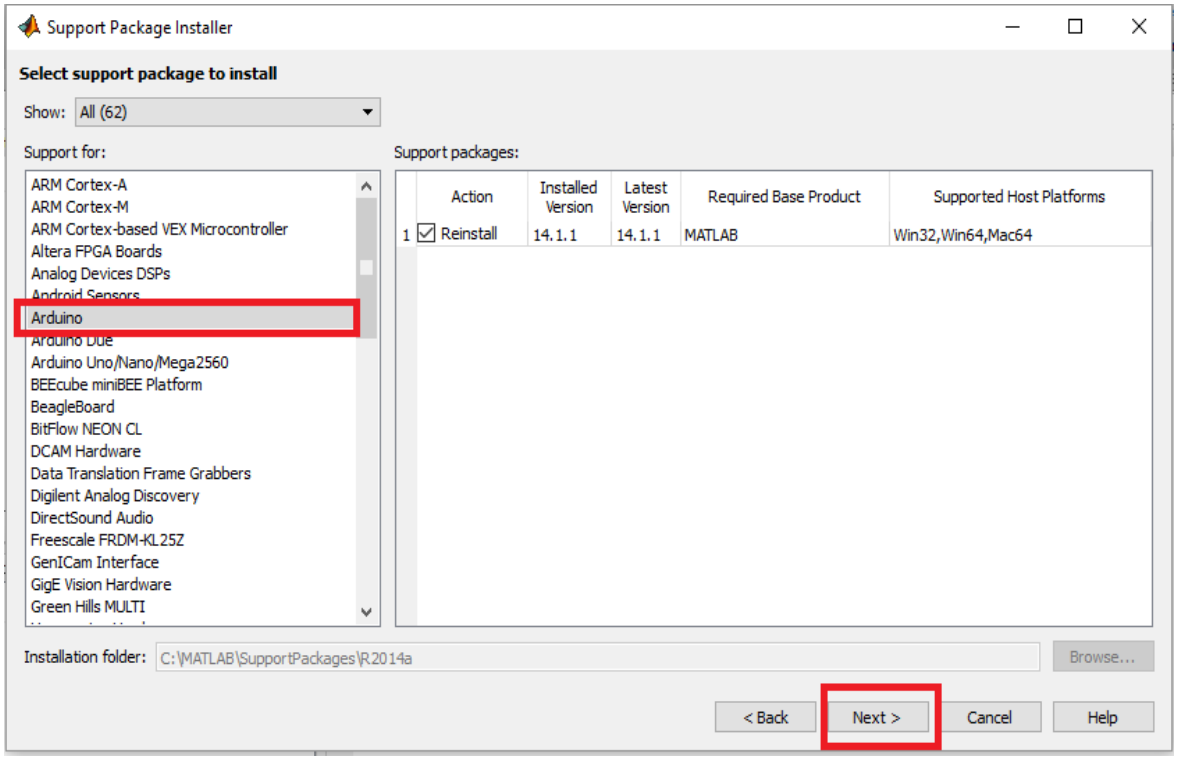
Hình 3.4. Get Hardware Support Package.

Bước 2: Cửa sổ **Support Package Installer** hiện lên, chọn **Install from Internet**



Hình 3.5.Cửa sổ Support Package Installer.

Bước 3: Sau đó chọn vào **Arduino** và tick vào cài đặt các Package



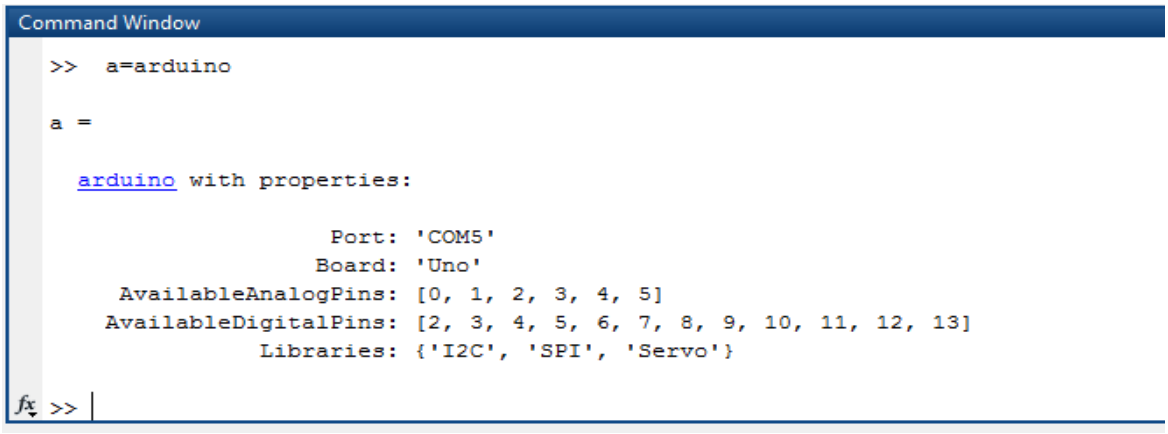
Hình 3.6. Giao diện cài Package cho Arduino.

Bước 4: Đăng nhập vào tài khoản **Matlab**. Sau khi đăng nhập xong, nhấn Next để cài đặt đến hết. Nhấn Finish để kết thúc phần cài đặt.

Muốn kiểm tra kết nối Arduino với Matlab thì ta có thể làm như sau:

- Bước 1: Trên Command Window gõ lệnh: a= arduino

Đây là câu lệnh để kiểm tra kết nối giữa Arduino với Matlab và các Packages đã được cài đặt để hỗ trợ cho Arduino hay chưa.

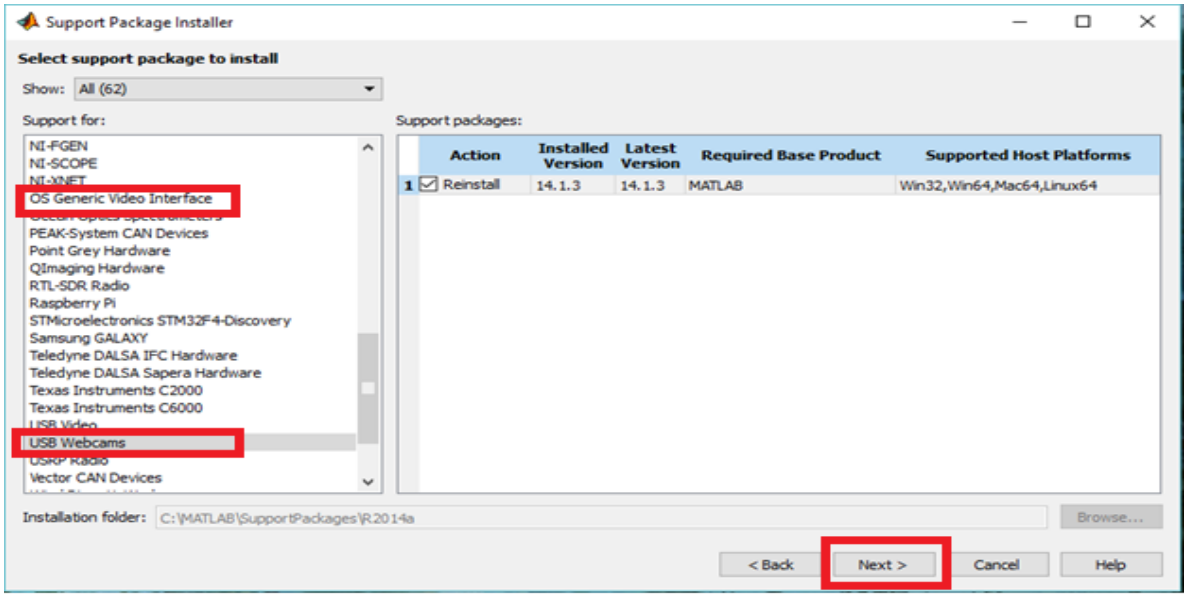


Hình 3.7.Kết nối Arduino và Matlab thành công.

3.3.2. Cài đặt Camera cho Matlab

Để kết nối Camera , chúng ta cài đặt các gói hỗ trợ phần cứng bổ sung vào Matlab. Từ cửa sổ **Support Package Installer** ở **Hình 2.4**, chúng ta cài đặt tương tự giống như cho Arduino.

Bước 1: Chọn vào **USB webcam** và **OS Generic video Interface** để cài đặt **.**



Hình 3.8.Cài đặt Camera cho Matlab.

Bước 2: Đăng nhập vào tài khoản Matlab và nhấn Next cho đến khi cài đặt xong.

Chương 4. THI CÔNG HỆ THỐNG

4.1 GIỚI THIỆU

Chương này nhóm giới thiệu về hệ thống nhóm đã thực hiện, lưu đồ cho chương trình nhận dạng cảm xúc trên khuôn mặt, chương trình chính và các bước thao tác để sử dụng mô hình.

4.2 THI CÔNG HỆ THỐNG

4.2.1 Thi công bo mạch

Mô hình của nhóm dùng hoàn toàn các module thiết kế sẵn được bán trên thị trường.

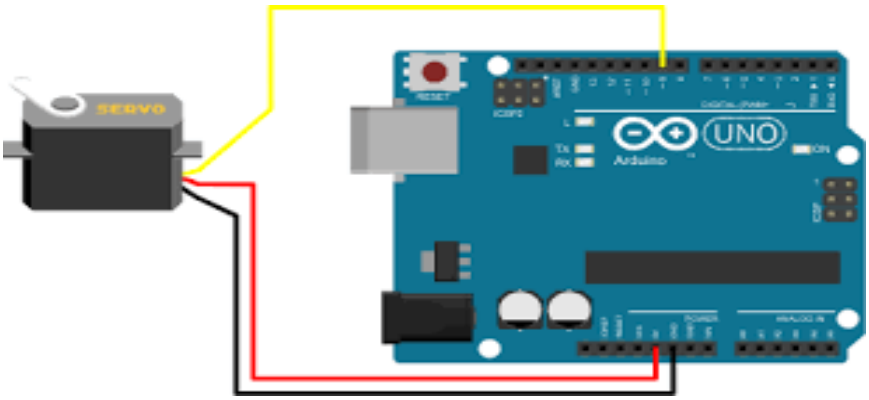
Bảng 4.1. Danh sách các linh kiện, module.

STT	Tên linh kiện	Số lượng	Loại	Chú thích
1	Arduino	1	Arduino Uno	
2	Động cơ servo	1	SG90	
3	Led	4	Led trắng, đỏ	
4	Camera	1	Webcam	

4.2.2 Lắp ráp và kiểm tra

a. Kết nối Arduino với động cơ servo SG90

Bước 1: chuẩn bị một board Arduino và một động cơ servo



Hình 4.1. Sơ đồ đấu nối dây của Arduino với Servo.

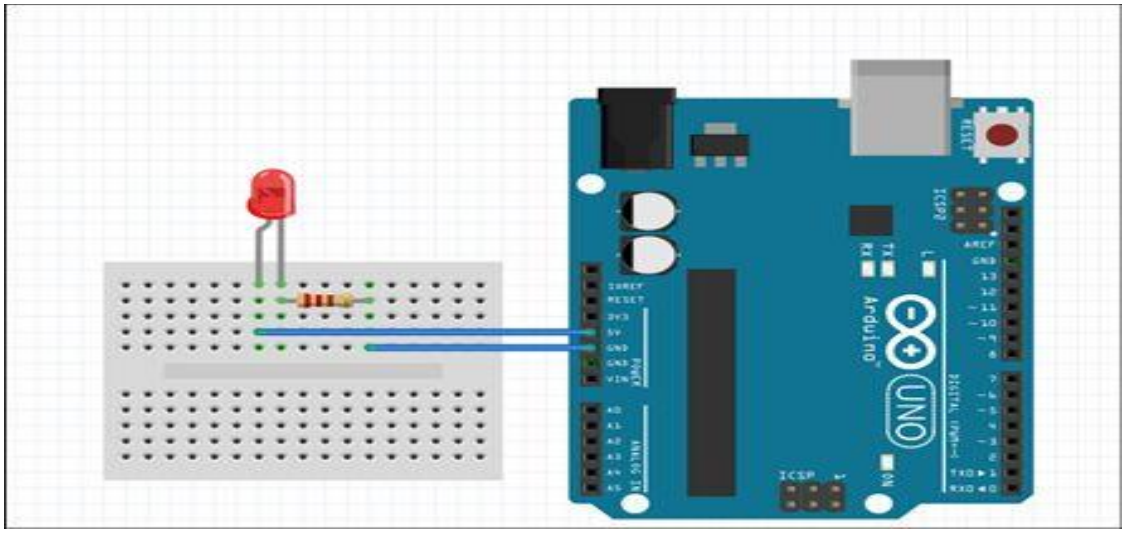
Bước 2: Vì động cơ servo có ba dây: đỏ, đen, vàng

- Dây đỏ: dây cấp nguồn 5V cho động cơ servo, ta kết nối với ngõ ra 5V của Arduino.
- Dây đen: dây cấp nguồn 0V cho động cơ servo, ta kết nối với ngõ ra 0V của Arduino.
- Dây vàng: dây tín hiệu điều khiển của động cơ servo, ta kết nối với ngõ ra có kí hiệu "~" trên các chân ngõ ra của Arduino.

Bước 3. Kiểm tra: Ta có thể dùng chương trình mẫu của phần mềm Arduino cung cấp sẵn để kiểm tra động cơ servo hoạt động có ổn định hay không?

b. Kết nối Arduino với Led.

Bước 1: Chuẩn bị đèn Led và Arduino và điện trở để hạn dòng cho led



Hình 4.2 Sơ đồ đấu nối dây của Arduino với led.

Bước 2. Ta kết nối chân dương của đèn led với môt ngõ ra tín hiệu của arduino, chân còn lại của đèn led ta kết nối với một đầu của điện trở, đầu kia của điện trở kết nối với GND của board arduino.

Ta có công thức tính điện trở hạn dòng cho led: $R = \frac{U}{I} (\Omega)$

Trong đó: R là điện trở.(Ω)

U: là điện áp.(V)

I: là dòng điện.(A)

Ta có: dòng điện của led là 15mA, điện áp cấp là 5V, điện áp của led là 0,7V

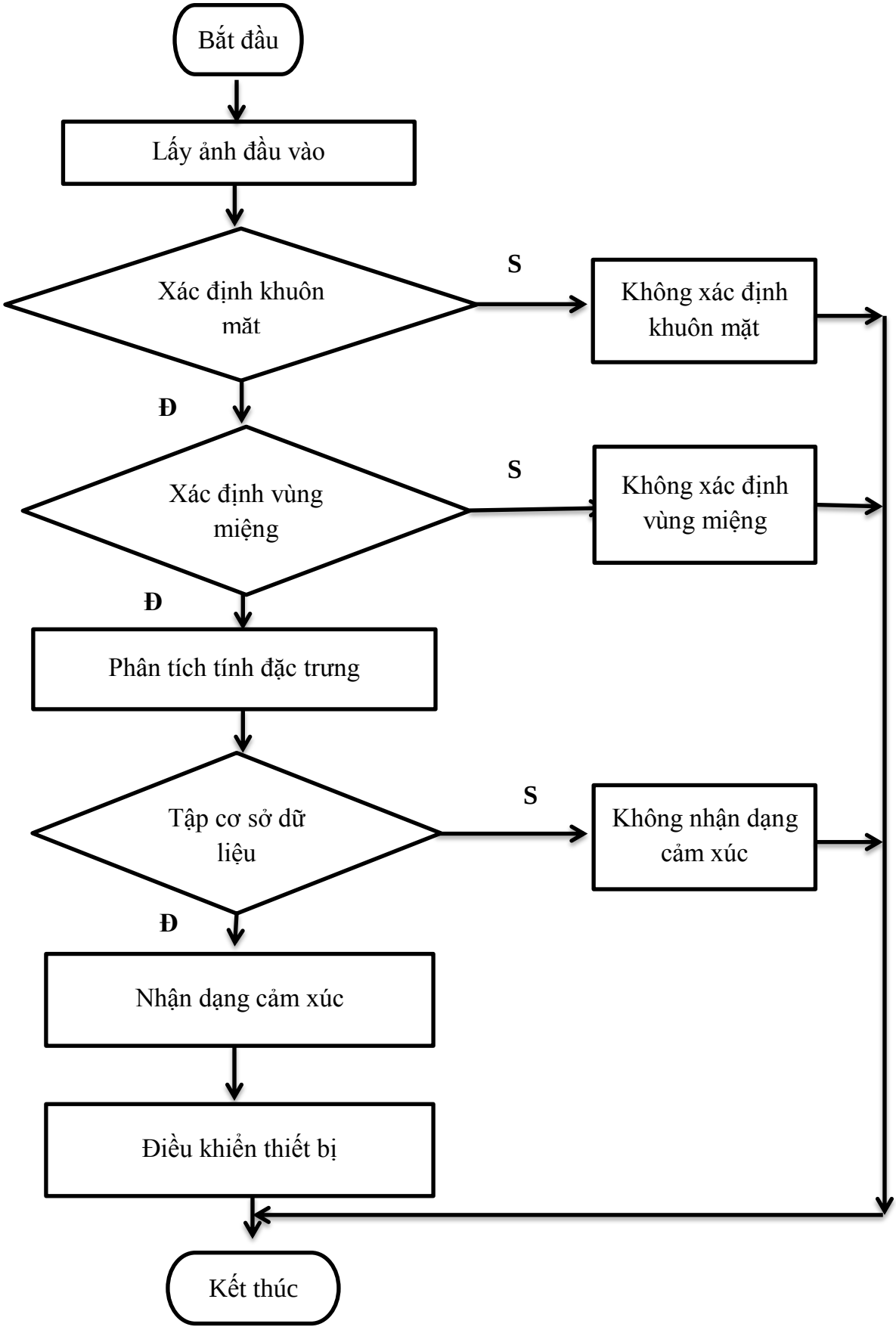
⇨ $R = \frac{5-0.7}{15} = 330\Omega$

Bước 3. Kiểm tra: Ta có thể dùng chương trình mẫu của phần mềm Arduino cung cấp sẵn để kiểm tra đèn led hoạt động có ổn định hay không?

4.3 LẬP TRÌNH HỆ THỐNG

4.3.1 Lưu đồ giải thuật

Lưu đồ hệ thống:



Hình 4.3. Lưu đồ chương trình chính.

Để đi đến mục tiêu là nhận dạng được cảm xúc trên khuôn mặt, ta phải xây dựng thành công chương trình huấn luyện ảnh và chương trình nhận dạng ảnh đầu vào.

Đầu tiên khởi tạo cơ sở dữ liệu hay nói cách khác là tạo bộ ảnh huấn luyện bằng cách tạo chương trình con huấn luyện ảnh.

Sau đó tiến hành nhận dạng ảnh và cho ra kết quả nhận dạng để điều khiển ứng dụng. Hệ thống nhận dạng trên hai loại ảnh:

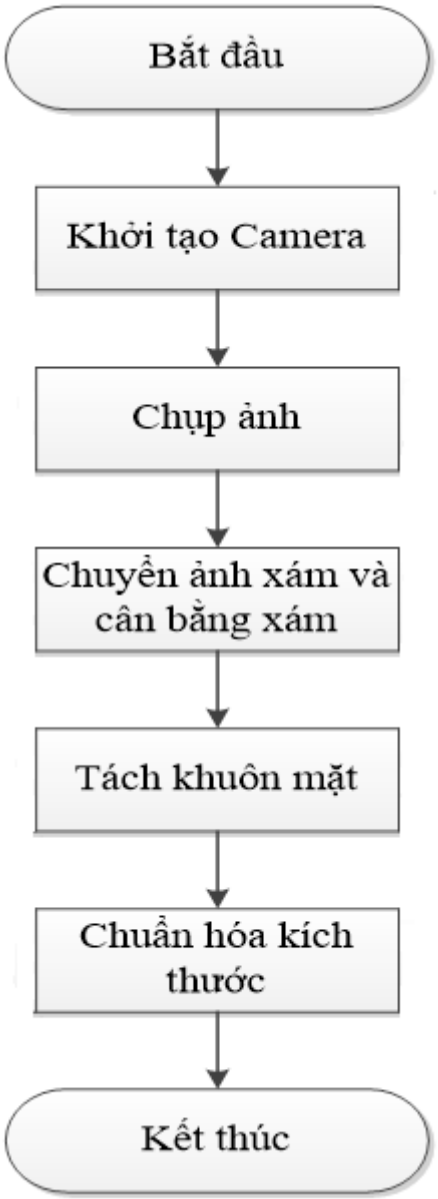
- Ảnh có chụp sẵn lưu trong máy.
- Ảnh trực tiếp chụp bằng webcam máy tính.

Chế độ huấn luyện ảnh sẽ tiến hành chụp 390 ảnh có chứa các cảm xúc trên khuôn mặt gồm: cảm xúc vui, cảm xúc buồn, và cảm xúc ngạc nhiên. Đầu tiên đặt một biến đếm bằng 0, sau đó tiến hành khởi động camera. Camera sẽ tiến hành chụp ảnh các cảm xúc trên khuôn mặt sau đó hệ thống sẽ cân bằng xám và hiệu chỉnh kích thước ảnh theo kích thước đã quy định trước sau đó lưu thông tin ảnh. Quá trình được thực hiện đến khi hoàn tất với một thông báo.

Chương trình tiến hành nhận dạng ảnh theo sơ đồ:

Hình 4.4. Sơ đồ chương trình nhận dạng.

- Xử lý cơ sở dữ liệu: quá trình bắt đầu với việc tạo file .mat trong matlab sau đó tiến hành xử lý tập ảnh huấn luyện và lưu lại dữ liệu. Khi cần so sánh ảnh chụp với cơ sở dữ liệu phải tiến hành nạp lại file .mat.
- Xử lý ảnh chụp: xử lý ảnh đầu vào chụp từ camera theo sơ đồ



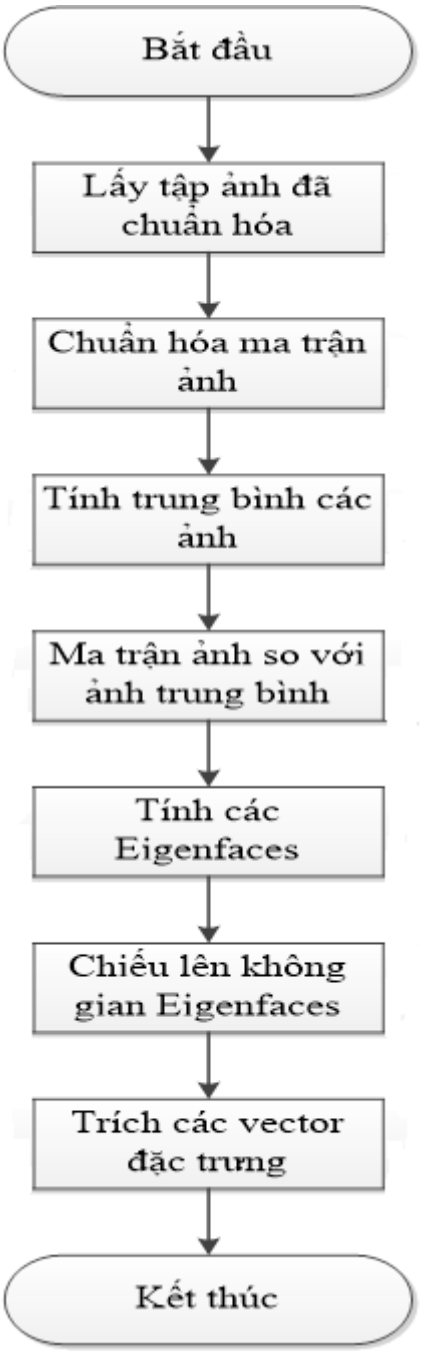
Hình 4.5. Sơ đồ chương trình xử lý ảnh chụp.

- Trích đặc trưng sử dụng thuật toán để lấy thông tin mang tính chất riệng biệt của một ảnh.

Các khâu trong quá trình trích đặc trưng:

- Đầu vào sử dụng ảnh đã được chuẩn hóa.
- Đầu ra cho ra vector đặc trưng của ảnh đầu vào.

Quá trình trích đặc trưng thể hiện theo lưu đồ:



Hình 4.6. Sơ đồ chương trình trích đặc trưng.

- So sánh ảnh chụp với cơ sở dữ liệu : Tiến hành so sánh ảnh chụp với cơ sở dữ liệu cụ thể là dùng phương pháp Euclid để tìm ảnh thuộc cơ sở dữ liệu trong không gian Eigenfaces có khoảng cách gần nhất với ảnh cần nhận dạng. Từ đó, xuất ra trạng thái cảm xúc tương ứng.

4.4 VIẾT TÀI LIỆU HƯỚNG DẪN SỬ DỤNG, THAO TÁC.

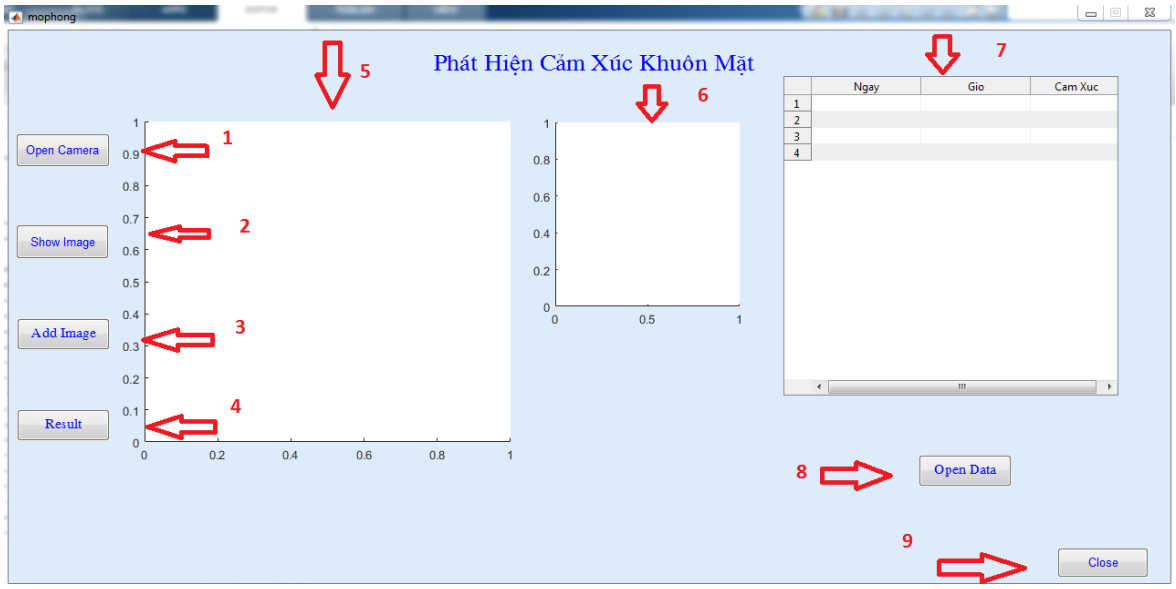
4.4.1 Tài liệu hướng dẫn sử dụng.

Bước 1: Khởi động phần mềm và kit arduino.

Hình 4.7 Khởi động phần mềm Matlab

Hình 4.8 Giao diện chương trình mô phỏng

Bước 2: Tiến hành chạy chương trình: Đầu tiên bấm chọn biểu Mô Phỏng trên màn hình để tiến hành chạy chương trình. Sau đó ta sẽ nhìn thấy giao diện như sau:



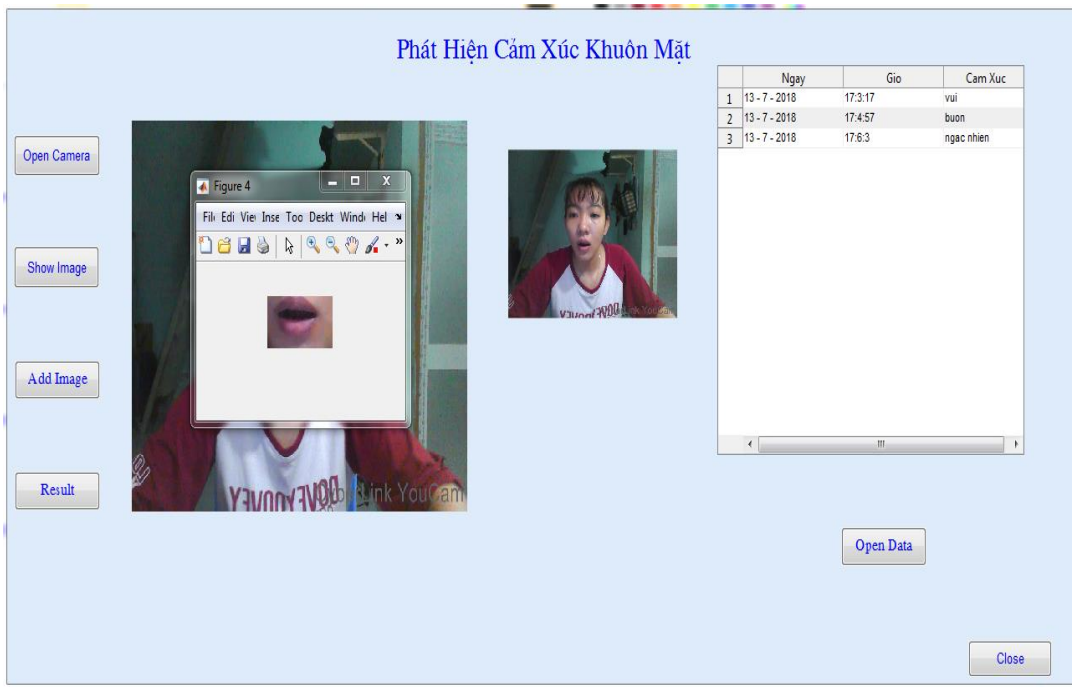
Hình 4.9 Giao diện bên trong chương trình mô phỏng

Trong đó:
1. Open Camera là biểu tượng mở camera.
2. Show Image là biểu tượng chụp lấy hình ảnh.
3. Add Image là biểu tượng lấy ảnh có sẵn trong máy.
4. Result là biểu tượng kết quả nhận dạng được
5. Axes1 khung hiển thị ảnh từ webcam máy tính.
6. Axes2 khung hiển thị ảnh chụp hoặc ảnh lưu sẵn từ webcam máy tính.
8. Open Data là biểu tượng mở dữ liệu nhận dạng trước đó.
9. Close là biểu tượng đóng chương trình.

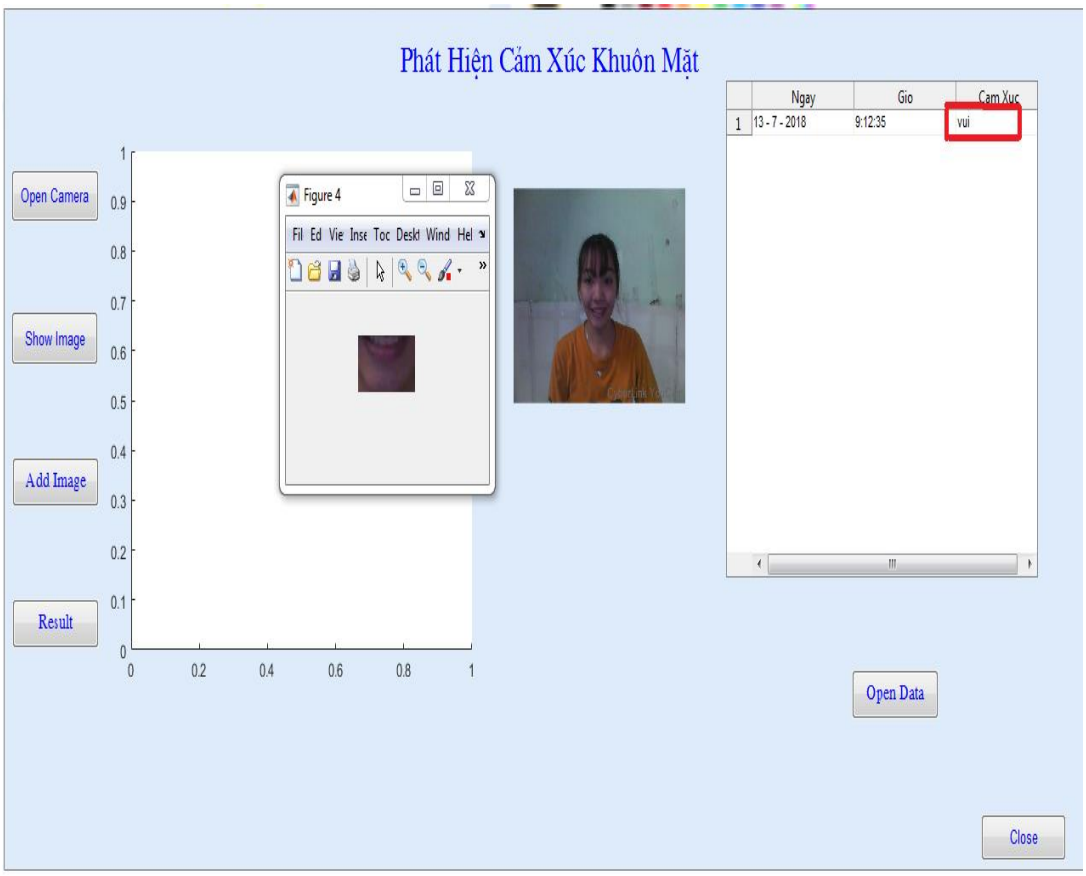
Bước 3: Ta chọn biểu tưởng Open Camera để mở camera, điều kiện là đưa khuôn mặt trực diện với khoảng cách nhận diện tốt nhất là 50cm so với webcam, ánh sáng đủ sáng, không quá tối hoặc quá chói.

Bước 4: Chọn biểu tượng Result để hiện thị kết quả nhận dạng được lên bảng hiện thị.

Bước 5: Quan sát kết quả.



Hình 4.10 Kết quả nhận được từ ảnh chụp trực tiếp



Hình 4.11 Kết quả nhận được từ ảnh lưu sẵn

Chương 5. KẾT QUẢ_NHẬN XÉT_ĐÁNH GIÁ

5.1 KẾT QUẢ

5.1.1. Tổng quan kết quả đạt được

Sau hơn 3 tháng tìm hiểu tài liệu chuyên môn, tài liệu trên Internet cùng với sự giúp đỡ tận tình của giáo viên hướng dẫn, nhóm thực hiện đề tại : "Nhận dạng cảm xúc trên khuôn mặt" đã hoàn thành xong theo yêu cầu và đúng thời gian quy định với những nội dung sau:

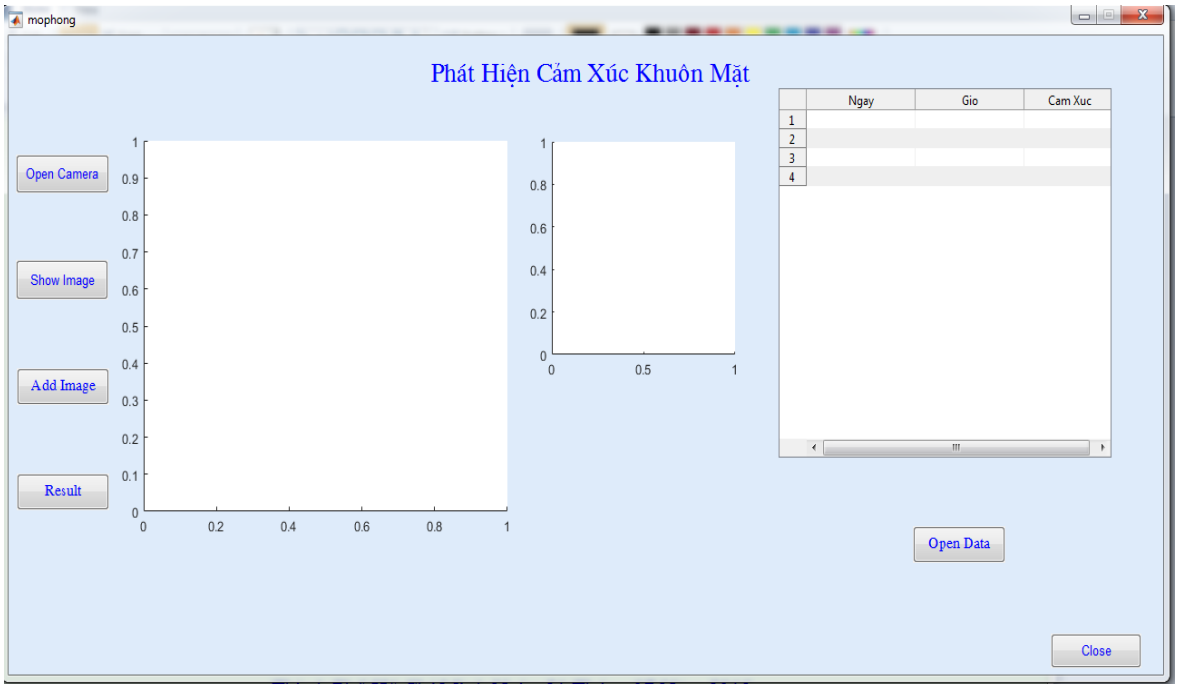
- Nắm được kiến thức cơ bản về kit Arduino, cài đặt được gói hỗ trợ Arduino, webcam trên phần mềm Matlab.
- Nắm được phương pháp trích đặc trưng PCA, đồng thời áp dụng được phương pháp này vào bài toán nhận dạng cảm xúc trên khuôn mặt.
- Xây dựng tốt chương trình huấn luyện ảnh (từ ảnh chụp và webcam) để tạo cơ sở dữ liệu nhận dạng .
- Xây dựng được chương trình nhận dạng cảm xúc từ ảnh huấn luyện.
- Nhận dạng thành công trên 80% cho từng loại cảm xúc.
- Tương ứng với từng loại cảm xúc, điều khiển thành công ứng dụng cơ bản (đóng mở cửa) cho kit Arduino từ phần mềm Matlab.

5.1.2. Kết quả thực tế

Giao diện chính:

Hình 5.1. Giao diện chính

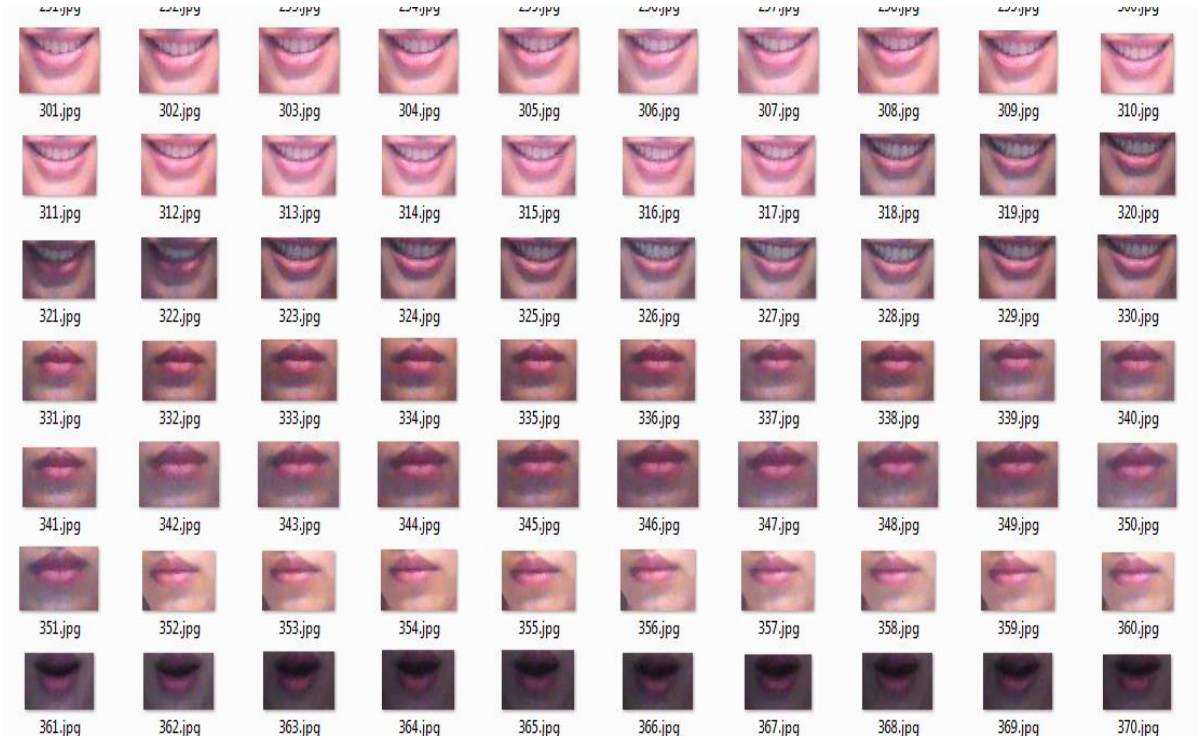
Giao diện chương trình mô phỏng



Hình 5.2. Giao diện chương trình mô phỏng

Nhóm xây dựng chương trình huấn luyện gồm 390 ảnh gồm :130ảnh cảm xúc vui, 130 ảnh cảm xúc buồn, 130 ảnh cảm ngạc nhiên.

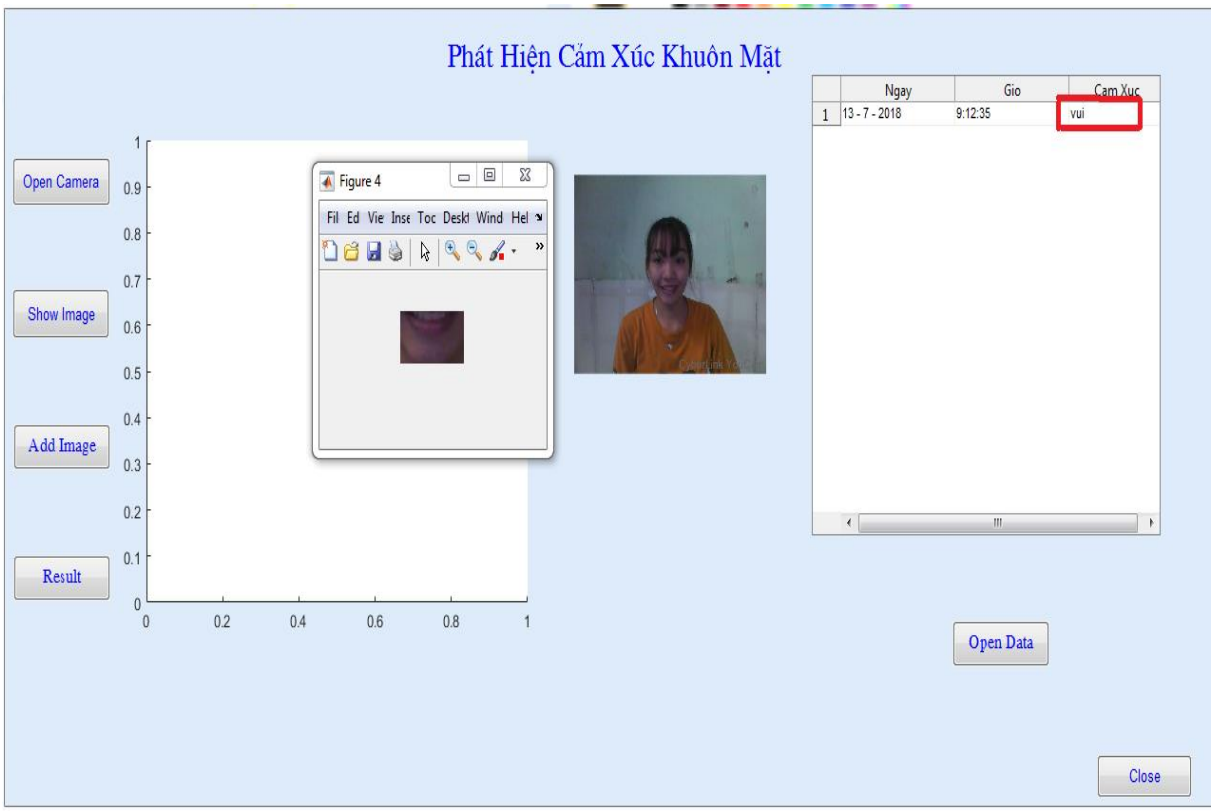
Hình 5.3. Ảnh chụp để kiểm tra



Hình 5.4 Data ảnh huấn luyện

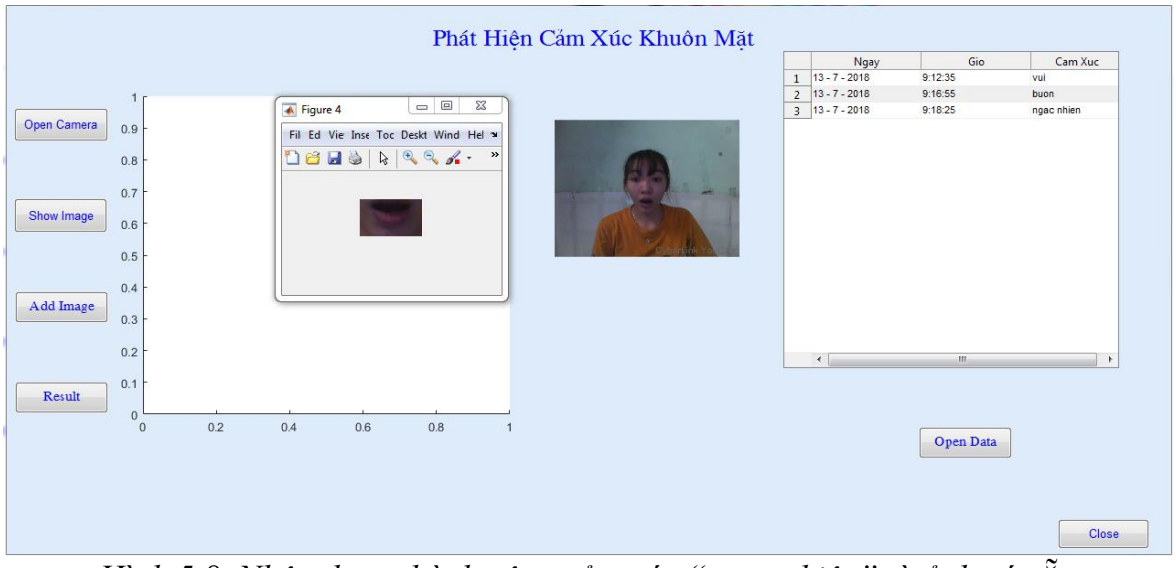
a. Kết quả nhận dạng từ ảnh chụp và chụp sẵn

- Nhận dạng từ ảnh chụp sẵn



Hình 5.5 Nhận dạng thành công cảm xúc "vui" từ ảnh có sẵn.

Hình 5.7 Nhận dạng thành công cảm xúc "buồn" từ ảnh có sẵn.



Hình 5.8 Nhận dạng thành công cảm xúc "ngạc nhiên" từ ảnh có sẵn.

- Nhận dạng ảnh chụp trực tiếp từ webcam

Hình 5.8 Nhận dạng thành công cảm xúc "vui" từ ảnh webcam.

Hình 5.8 Nhận dạng thành công cảm xúc "buồn" từ ảnh chụp

Hình 5.8 Nhận dạng thành công cảm xúc "ngạc nhiên" từ ảnh chụp.

- Kết quả nhận được.

Hình 5.6 Nhận dạng thành công cảm xúc "vui" mở cửa.

Hình 5.8. Nhận dạng thành công cảm xúc "buồn" mở đèn

Hình 5.9 Nhận dạng thành công cảm xúc "ngạc nhiên" đóng cửa, tắt đèn

b. Kết quả điều khiển ứng dụng trên Arduino

Sau quá trình nhận dạng cảm xúc từ chương trình chính là quá trình nhúng dữ liệu từ Matlab xuống kit Arduino:

- Cảm xúc vui điều khiển cửa mở
- Cảm xúc buồn điều khiển mở đèn.
- Cảm xúc ngạc nhiên điều khiển đóng cửa tắt đèn.

Ứng dụng hoạt động chưa ổn định lắm, thiếu chính xác chỉ đáp ứng một phần yêu cầu nhóm.

c. Kết quả thống kê

- Thống kê kết quả nhận dạng 100 ảnh từ ảnh chụp cho mỗi trạng cảm xúc (ảnh huấn luyện và nhận dạng đều là ảnh chụp sẵn, đồng thời ảnh nhận dạng được chụp trong cùng điều kiện ảnh huấn luyện) .

Bảng 5.1. Thống kê kết quả nhận dạng từ ảnh chụp.

Cảm xúc	Vui	Buồn	Ngạc nhiên
Số lần thử	100	100	100
Số lần đúng	95	82	89
Số lần sai	5	18	11
Độ chính xác	95%	82%	89%

Thống kê kết quả nhận dạng 100 ảnh lấy trực tiếp từ camera cho mỗi trạng thái cảm xúc, đồng thời các ảnh lấy trực tiếp từ camera trong cùng điều kiện môi trường huấn luyện.

Bảng 5.2. Thống kê kết quả nhận dạng từ camera.

Cảm xúc	Vui	Buồn	Ngạc nhiên
Số lần thử	100	100	100
Số lần đúng	90	83	86
Số lần sai	10	17	14
Độ chính xác	90%	83%	86%

5.2 NHẬN XÉT VÀ ĐÁNH GIÁ

Đồ án do nhóm thực hiện đã đạt được khoảng 90% mục tiêu do nhóm đặt ra.

- Điều khiển các ứng dụng trên Arduino tương đối tốt, đúng yêu cầu đặt ra.Tuy nhiên, do chưa có nhiều kinh nghiệm nên việc cài đặt Arduino cho Matlab lúc ban đầu gặp nhiều khó khăn làm tốn nhiều thời gian.
- Tỉ lệ nhận dạng đúng cao nhất cho cả hai trường hợp (nhận dạng từ ảnh chụp và webcam) là khoảng 95% ở cảm xúc "vui" và thấp nhấp nhất là 82% ở cảm xúc "buồn". Việc nhận dạng cảm xúc còn phụ thuộc rất nhiều vào môi trường huấn luyện, nếu khác môi trường huấn luyện độ chính xác cho từng loại cảm xúc sẽ giảm.

Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 KẾT LUẬN

Sau khi tìm hiểu và thực hiện đề tài: "Nhận dạng cảm xúc trên khuôn mặt", nhóm đã đạt được những kết quả như sau:

- Mô hình "Nhận dạng cảm xúc trên khuôn mặt" hoạt động ổn định, đáp ứng được 90% yêu cầu đặt ra ban đầu.
- Mô hình đã nhận dạng được 3 cảm xúc vui, buồn, ngạc nhiên trên khuôn mặt với ảnh đầu đầu vào là ảnh được chụp từ webcam trên máy tính và ảnh đã được chụp sẵn.
- Ứng với trạng thái cảm xúc vui thì mô hình đã điều khiển được động cơ mở cửa. Cảm xúc buồn mô hình điều khiển mở đèn. Khi cảm xúc ngạc nhiên sẽ điều khiển đóng cửa và tắt đèn.

6.2 Ưu và nhược điểm

6.2.1 Ưu điểm

- Mô hình đơn giản, nhỏ gọn, dễ vận hành hệ thống.
- Dễ dàng thay đổi tăng số lượng nhận dạng sao cho phù hợp khi có nhu cầu.
- Các linh kiện trên mạch dễ dàng thay thế nếu gặp sự cố.

6.2.2 Nhược điểm

- Hướng mặt phải tương đối trực diện với webcam. Nghiêng, xoay trái, xoay phải, ngẩng đầu và cúi xuống góc lớn hơn 30 thì giải thuật có thể không phát hiện được cảm xúc hoặc phát hiện được nhưng tỉ lệ nhận dạng đúng rất thấp.
- Khi khoảng cách càng xa (lớn hơn 50cm) thì tỉ lệ nhận dạng đúng càng giảm xuống.
- .Mô hình vẫn còn xảy ra tình trạng nhận dạng không chính xác cảm xúc buồn và ngạc nhiên hoặc không nhận dạng. Dẫn đến hệ thống điều khiển hoạt động với độ chính xác không cao.

6.3 HƯỚNG PHÁT TRIỂN

Trong phạm vi đồ án nhóm chỉ trình bày những phần cơ bản được lập trình trên Matlab. Tuy nhiên việc mở rộng mô hình nhận dạng cảm xúc để điều khiển các ứng dụng thực tế hơn thì cần có thời gian nghiên cứu và cải tiến. Nhóm nghiên cứu nhận thấy có thể phát triển thêm các tính năng sau:

- Tăng thêm số lượng nhận dạng cảm xúc khác trên khuôn mặt người như nhận dạng cảm xúc giận dữ, hốt hoảng, sợ hãi. Để có thể ứng dụng vào thực tiễn như điều khiển các thiết bị trong gia đình, robot, các cảnh báo .
- Xây dựng một chương trình hoàn chỉnh để chương trình nhận dạng không phụ thuộc vào khoảng cách và điều kiện môi trường nhận dạng, có thể nhận dạng ở mọi khoảng cách, điều kiện môi trường khác nhau. Đồng thời có giao diện để tương tác trực tiếp với người sử dụng thông qua web hoặc phần mền trên điện thoại.

Sách tham khảo

[1] Đinh Xuân Nhất, *Nghiên cứu các thuật toán nhận dạng cảm xúc khuôn mặt trên ảnh 2D*, Khóa luận tốt nghiệp, Trường Đại học Công nghệ-ĐHQGHN, 2010.

[2] Lê Mạnh Tuấn, *Phát hiện mặt người trong ảnh và ứng dụng*, Khóa luận tốt nghiệp, Trường Đại học Công nghệ-ĐHQGHN, 2009.

[3] Võ Hồng Hoan, *Nhận dạng mặt người trên Matlab*, Khóa luận tốt nghiệp, Trường Đại học sư pham kỹ thuật TP.HCM, 2010.

[4] Nguyễn Thanh Hải, *Giáo trình: Xử lý ảnh*, Nhà xuất bản ĐH Quốc Gia TP. Hồ Chí Minh, 2014.

[5] Lê Hồng Anh – Nguyễn Thái Bình, *Ứng dụng ngôn ngữ hình thể trên Matlab*, Khóa luận tốt nghiệp, Trường Đại học Công Nghiệp HN, 2011

[6] Nguyễn Trung Hiếu- Bùi Ngọc Liêm, *Dò Tìm và Cắt Ảnh Mặt Người Dùng PCA*, Khóa luận tốt nghiệp, Trường Đại học Công Nghiệp TP.HCM, 2010

[7]Ngô Quốc Tạo, Ngô Phương Đông, Nguyễn Thanh Hòa, Phạm Việt Bình (2003), *Báo cáo "Nhận dạng mặt người trong môi trường độ sáng không đồng nhất"*, Hội thảo Công nghệ thông tin quốc gia lần thứ VIII, Thái Nguyên, 29-31/8/2003

[8]. Các Wedsite:

http://www.mathworks.com/matlabcentral/fileexchange/?term=Variable

http://www.picvietnam.com/forum//showthread.php?t=387

www. diendandientu. com

www. dientuvienthong. net

PHỤ LỤC

1. Hàm train

```
function [isSucceed] = EigenFace(strTrainPath, strLabelFile, strTestPath)

isSucceed = 0;
if (exist('strTrainPath')==0)
   strTrainPath = input('Enter Train Folder Name:','s');
end
if (exist('strLabelFile')==0)
   strLabelFile = input('Enter Label File Name:','s');
end

somau = input('Enter So Mau:','s');

fid=fopen(strLabelFile);
imageLabel=textscan(fid,'%s %s','whitespace',',');
fclose(fid);

numImage = length(imageLabel{1,1});  % Total Observations: Number of Images
in training set

numImage = length(imageLabel{1,1});
TrainImages='';
for i = 1:numImage
   TrainImages{i,1} = strcat(strTrainPath,'\',imageLabel{1,1}(i));
end
imageSize = [60,40];
% load train data
img = zeros(imageSize(1)*imageSize(2),numImage);
for i = 1:numImage
   ab =
imresize(imresize(imread(cell2mat(TrainImages{i,1})),[375,300]),imageSize);
   a1= rgb2gray(ab);
   aa=histeq(uint8(a1));
   img(:,i) = aa(:);
   disp(sprintf('Loading Train Image # %d',i));
end
meanImage = mean(img,2);     % trung binh mau

img = (img - meanImage*ones(1,numImage))';   % dau vao PCA

[C,S,L]=princomp(img,'econ');% Performing PCA Here

EigenRange = [1:(somau-1)]; % doi tuy thuoc vao so anh dau vao
C = C(:,EigenRange);
```

```
save('somau.mat','somau');
save('C.mat','C');
save('S.mat','S');
save('L.mat','L');
save('imageLabel.mat','imageLabel');
save('meanImage.mat','meanImage');
```

2. Chương trình lấy mẫu đầu vào

```
 vid=videoinput('winvideo', 1,'YUY2_640x480');
preview(vid);
for k=0:10%
   capcha = getsnapshot(vid);
    videoFrame=ycbcr2rgb(capcha);
    %figure(1), imshow(videoFrame);
     FaceDetect = vision.CascadeObjectDetector;
      BB = step(FaceDetect,videoFrame);
      figure(2),imshow(videoFrame);
       hold on
     %BB(:,3)
     %BB(:,4)
     %if c > 6400
      for i = 1:size(BB,1)
         c=BB(i,3)*BB(i,4);
         % vung chua khuon mat
         if c > 6400
          a=rectangle('Position',BB(i,:),'LineWidth',3,'LineStyle','-','EdgeColor','r');
          J= imcrop(videoFrame,[BB(i,1),BB(i,2),BB(i,3),BB(i,4)+15]);
          [x y]=size(J(:,:,1));
          MouthDetect                                                                    =
vision.CascadeObjectDetector('Mouth','MergeThreshold',16);
          B=step(MouthDetect,J);
         % tim vung chua mouth
           j=size(B,1);
           if j>1
              a=max(B(:,2));
              [ik jk]=find(B(:,2)==a);
              if B(ik,2)> x/2
                  if B(ik,2)< 0.65*x

rectangle('Position',[BB(i,1)+B(ik,1),BB(i,2)+B(ik,2)+x/15,B(ik,3),B(ik,4)],'LineWi
dth',4,'LineStyle','-','EdgeColor','b');
              I= imcrop(J,[B(ik,1),B(ik,2)+x/15,B(ik,3),B(ik,4)]);
               pathname='D:\DAITRANG\DOANTOT  NGHIEP\DATN_TRANG\cam
xuc khuon mat\cam xuc khuon mat\data';
              filename = [pathname num2str(j+k*(size(B,1))) '.jpg'];
    imwrite(I,filename);
```

```
                else

rectangle('Position',[BB(i,1)+B(ik,1),BB(i,2)+B(ik,2),B(ik,3),B(ik,4)],'LineWidth',4
,'LineStyle','-','EdgeColor','b');
            I= imcrop(J,[B(ik,1),B(ik,2),B(ik,3),B(ik,4)]);
            pathname='D:\DAITRANG\DOAN TOT NGHIEP\DATN_TRANG\cam
xuc khuon mat\cam xuc khuon mat\data';
            filename = [pathname num2str(j+k*(size(B,1))) '.jpg'];
    imwrite(I,filename);
                end

            end
          elseif j==1
            if B(1,2)>x/2
                if B(1,2)< 0.65*x

rectangle('Position',[BB(i,1)+B(1,1),BB(i,2)+B(1,2)+x/15,B(1,3),B(1,4)],'LineWidt
h',4,'LineStyle','-','EdgeColor','b');
            I= imcrop(J,[B(1,1),B(1,2)+x/15,B(1,3),B(1,4)]);
             pathname='D:\DAI TRANG\DO AN TOT
NGHIEP\DATN_TRANG\cam xuc khuon mat\cam xuc khuon mat\data';
            filename = [pathname num2str(j+k*(size(B,1))) '.jpg'];
    imwrite(I,filename);
                else

rectangle('Position',[BB(i,1)+B(1,1),BB(i,2)+B(1,2),B(1,3),B(1,4)],'LineWidth',4,'L
ineStyle','-','EdgeColor','b');
            I= imcrop(J,[B(1,1),B(1,2),B(1,3),B(1,4)]);
            pathname='D:\DAI TRANG\DO AN TOT
NGHIEP\DATN_TRANG\cam xuc khuon mat\cam xuc khuon mat\data';
            filename = [pathname num2str(j+k*(size(B,1))) '.jpg'];
    imwrite(I,filename);
                end
            end
          end


        end
      end


end

%figure(10), imshow(J);
```

3. Giao diện chính chương trình

```
function varargout = giaodienchuongtrinh(varargin)
% GIAODIENCHUONGTRINH MATLAB code for giaodienchuongtrinh.fig
```

```
%      GIAODIENCHUONGTRINH, by itself, creates a new
GIAODIENCHUONGTRINH or raises the existing
%      singleton*.
%
%      H = GIAODIENCHUONGTRINH returns the handle to a new
GIAODIENCHUONGTRINH or the handle to
%      the existing singleton*.
%
%      GIAODIENCHUONGTRINH('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in GIAODIENCHUONGTRINH.M with the
given input arguments.
%
%      GIAODIENCHUONGTRINH('Property','Value',...) creates a new
GIAODIENCHUONGTRINH or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before giaodienchuongtrinh_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to giaodienchuongtrinh_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help giaodienchuongtrinh

% Last Modified by GUIDE v2.5 04-May-2018 22:48:07

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @giaodienchuongtrinh_OpeningFcn, ...
                   'gui_OutputFcn',  @giaodienchuongtrinh_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
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before giaodienchuongtrinh is made visible.
function giaodienchuongtrinh_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to giaodienchuongtrinh (see VARARGIN)
axes(handles.axes1);
A=imread('logo.jpg');
imshow(A);

% Choose default command line output for giaodienchuongtrinh
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes giaodienchuongtrinh wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = giaodienchuongtrinh_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close
mophong

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
close


% --- Executes during object creation, after setting all properties.
function text2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

4. Chương trình mô phỏng

```
function varargout = mophong(varargin)
% MOPHONG MATLAB code for mophong.fig
%      MOPHONG, by itself, creates a new MOPHONG or raises the existing
%      singleton*.
%
%      H = MOPHONG returns the handle to a new MOPHONG or the handle to
%      the existing singleton*.
%
%      MOPHONG('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in MOPHONG.M with the given input
arguments.
%
%      MOPHONG('Property','Value',...) creates a new MOPHONG or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before mophong_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to mophong_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help mophong

% Last Modified by GUIDE v2.5 17-Jul-2018 12:13:54

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mophong_OpeningFcn, ...
                   'gui_OutputFcn',  @mophong_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
% End initialization code - DO NOT EDIT


% --- Executes just before mophong is made visible.
function mophong_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to mophong (see VARARGIN)
biendem=1;
biendem1=1;
save('dulieu.mat')
save('bien.mat','biendem');
save('bien1.mat','biendem1');


% Choose default command line output for mophong
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes mophong wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = mophong_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% show webcam

vid = videoinput('winvideo', 1,'YUY2_640x480');
% only capture one frame per trigger, we are not recording a video

vid.FramesPerTrigger = 1;
% output would image in RGB color space

vid.ReturnedColorspace = 'rgb';
% tell matlab to start the webcam on user request, not automatically
triggerconfig(vid, 'manual');
% we need this to know the image height and width
vidRes = get(vid, 'VideoResolution');
% image width
imWidth = vidRes(1);
% image height

imHeight = vidRes(2);
% number of bands of our image (should be 3 because it's RGB)

nBands = get(vid, 'NumberOfBands');
% create an empty image container and show it on axPreview

hImage = image(zeros(imHeight, imWidth, nBands), 'parent', handles.axes1);
% begin the webcam preview

preview(vid, hImage);
handles.vid=vid;
guidata(hObject, handles);
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
vid= handles.vid;
axes(handles.axes2);
capcha = getsnapshot(vid);
%videoFrame=ycbcr2rgb(capcha);
%FaceDetect = vision.CascadeObjectDetector;
%       BB = step(FaceDetect,capcha);
     %figure(2),imshow(videoFrame);
```

```
      % hold on
         % vung chua khuon mat
         %if c > 6400
          %   i=size(BB,1)
%
a=rectangle('Position',[BB(i,1),BB(i,2),BB(i,3),BB(i,4)+15],'LineWidth',3,'LineStyl
e','-','EdgeColor','r');
%           J= imcrop(BB,[BB(i,1),BB(i,2),BB(i,3),BB(i,4)+15]);
          %[x y]=size(J(:,:,1));
%           MouthDetect =
vision.CascadeObjectDetector('Mouth','MergeThreshold',16);
%           B=step(MouthDetect,J);
imshow(capcha);
%imshow(BB)

%videoFrame=ycbcr2rgb(capcha);
    %figure(1), imshow(videoFrame);


handles.capcha=capcha;
guidata(hObject, handles);


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
fclose(instrfind);
close


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%%%%%
%%%%%

load('bien.mat')
load('bien1.mat')
a=load('C.mat');
b=load('S.mat');
c=load('L.mat');
d=load('imageLabel.mat');
e=load('meanImage.mat');
```

```
f=load('somau.mat');
C=a.C;
S=b.S;
L=c.L;
somau=f.somau;
meanImage=e.meanImage;
imageLabel=d.imageLabel;
imageSize = [60,40];
numImage=size(S,1);
global w;
w = serial('COM5'); %% khai báo chân k?t n?i v?i arduino

videoFrame=handles.capcha;
FaceDetect = vision.CascadeObjectDetector;
      BB = step(FaceDetect,videoFrame);
     %figure(2),imshow(videoFrame);
      % hold on

        for i = 1:size(BB,1)
         c=BB(i,3)*BB(i,4);

         % vung chua khuon mat
         if c > 6400
            i=size(BB,1)
      %
a=rectangle('Position',[BB(i,1),BB(i,2),BB(i,3),BB(i,4)+15],'LineWidth',3,'LineStyl
e','-','EdgeColor','r');
          J= imcrop(videoFrame,[BB(i,1),BB(i,2),BB(i,3),BB(i,4)+15]);
          [x y]=size(J(:,:,1));
          MouthDetect =
vision.CascadeObjectDetector('Mouth','MergeThreshold',16);
          B=step(MouthDetect,J);
          %figure(3), imshow(B);     da sua
         % tim vung chua mouth
           j=size(B,1);
           if j>1
              a=max(B(:,2));
              [ik jk]=find(B(:,2)==a);
              if B(ik,2)> x/2
                 if B(ik,2)< 0.65*x
      %
rectangle('Position',[BB(i,1)+B(ik,1),BB(i,2)+B(ik,2)+x/15,B(ik,3),B(ik,4)],'LineWi
dth',4,'LineStyle','-','EdgeColor','b');
               I= imcrop(J,[B(ik,1),B(ik,2)+x/15,B(ik,3),B(ik,4)]);
               %I= imcrop(J,[B(ik,1),B(ik,2),B(ik,3),B(ik,4)]);
              figure(4), imshow(I);
```

```
                else
        %
rectangle('Position',[BB(i,1)+B(ik,1),BB(i,2)+B(ik,2),B(ik,3),B(ik,4)],'LineWidth',4
,'LineStyle','-','EdgeColor','b');
              I= imcrop(J,[B(ik,1),B(ik,2),B(ik,3),B(ik,4)]);
              figure(4), imshow(I);

                 end
                 % chac chan co cam xuc
                              % nhan dang cam xuc
img = zeros(imageSize(1)*imageSize(2),1);
ab = imresize(imresize(I,[375,300]),imageSize); %% cat anh ra lay khuon mieng
doianh=size(ab,3);
if doianh==1
      a1=ab;

else
     a1=rgb2gray(ab);

end
aa=histeq(uint8(a1));
img(:,1) = aa(:);% luu hinh test dang cot%
img = (img - meanImage*ones(1,1))';
Projected_Test = img*C;
EigenRange = [1:(somau-1)];
TestImage = Projected_Test(1,:);
Other_Dist = zeros(1,numImage);
for i = 1:numImage
      Other_Dist(1,i) = sqrt((TestImage'-S(i,EigenRange)')' ...
         *(TestImage'-S(i,EigenRange)'));

   end

[Min_Dist,Min_Dist_pos] = min(Other_Dist,[],2);
Expr = cell2mat(imageLabel{1,2}(Min_Dist_pos));


fopen(w);
pause(2); % thoi gian delay
fwrite(w,1);
fclose(instrfind); %buon


  %%%%%%%%%%%%%%%%%%%%%%
%data{k}=Expr;
%so = cell2mat(imageLabel{1,1}(Min_Dist_pos));
%set(handles.edit1,'string',so);
```

```
a=clock;
s1=sprintf('%d - %d - %d',a(3),a(2),a(1));
s2=sprintf('%d:%d:%d',a(4),a(5),round(a(6)));

%data = {s1,s2,Expr};

load('dulieu.mat');
data{biendem,1}=s1;
data{biendem,2}=s2;
data{biendem,3}=Expr;
save('dulieu.mat','data');
load('dulieu.mat')
biendem=biendem+1;
save('bien.mat','biendem')
set(handles.uitable1,'Data',data);
data
% nhan dang xong cam xuc
            end

          elseif j==1
             if B(1,2)>x/2
                if B(1,2)<0.65*x
                 %
rectangle('Position',[BB(i,1)+B(1,1),BB(i,2)+B(1,2)+x/8,B(1,3),B(1,4)],'LineWidth'
,4,'LineStyle','-','EdgeColor','b');
             I= imcrop(J,[B(1,1),B(1,2)+x/15,B(1,3),B(1,4)]);
            %I= imcrop(J,[B(1,1),B(1,2),B(1,3),B(1,4)]);
             figure(4), imshow(I);


                else

%rectangle('Position',[BB(i,1)+B(1,1),BB(i,2)+B(1,2),B(1,3),B(1,4)],'LineWidth',4,'
LineStyle','-','EdgeColor','b');
             I= imcrop(J,[B(1,1),B(1,2),B(1,3),B(1,4)]);
             figure(4), imshow(I);


                end
                        % chac chan co cam xuc
                      % nhan dang cam xuc
 img = zeros(imageSize(1)*imageSize(2),1);
ab = imresize(imresize(I,[375,300]),imageSize);
doianh=size(ab,3);
if doianh==1
      a1=ab;
```

```
else
      a1=rgb2gray(ab);

end
aa=histeq(uint8(a1));
   img(:,1) = aa(:);% luu hinh test dang cot%
img = (img - meanImage*ones(1,1))';
Projected_Test = img*C;
EigenRange = [1:(somau-1)];
TestImage = Projected_Test(1,:);
Other_Dist = zeros(1,numImage);
for i = 1:numImage
       Other_Dist(1,i) = sqrt((TestImage'-S(i,EigenRange)')' ...
          *(TestImage'-S(i,EigenRange)'));
   end

[Min_Dist,Min_Dist_pos] = min(Other_Dist,[],2);
Expr = cell2mat(imageLabel{1,2}(Min_Dist_pos));




%%global w;
%%w = serial('COM5');

fopen(w);
pause(2);
fwrite(w,2); % vui
fclose(instrfind);

%%%%%%%%%%%%%%%%%%%%

  %%%%%%%%%%%%%%%%%%%%%%
%so = cell2mat(imageLabel{1,1}(Min_Dist_pos));
%set(handles.edit1,'string',so);
%data{k}=Expr;
% nhan dang xong cam xuc
a=clock;
s1=sprintf('%d - %d - %d',a(3),a(2),a(1));
s2=sprintf('%d:%d:%d',a(4),a(5),round(a(6)));


%d = {s1,s2,Expr};

load('dulieu.mat');
data{biendem,1}=s1;
data{biendem,2}=s2;
```

```
data{biendem,3}=Expr;
save('dulieu.mat','data');
load('dulieu.mat')
biendem=biendem+1;
save('bien.mat','biendem')
set(handles.uitable1,'Data',data);

%save('luu.mat','data');
data
if (strcmp(Expr,'ngac nhien'))
   tam = 3;
   fopen(w);
   pause(2);
   fwrite(w,3); %ngac nhien
   fclose(instrfind);
   %tam
end

end

          end


         end

      end
biendem1=biendem1+1;
save('bien1.mat','biendem1')

if biendem<biendem1

                        a=clock;
s1=sprintf('%d - %d - %d',a(3),a(2),a(1));
s2=sprintf('%d:%d:%d',a(4),a(5),round(a(6)));

%d = {s1,s2,Expr};
load('dulieu.mat');
data{biendem,1}=s1;
data{biendem,2}=s2;
data{biendem,3}='Khong Chua Khuon Mat';
save('dulieu.mat','data');
load('dulieu.mat')
biendem=biendem+1;
save('bien.mat','biendem')
set(handles.uitable1,'Data',data);

end
```

```
% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load('dulieu.mat')
xlswrite('data.xls',data);
winopen('data.xls');



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile('*','LOAD AN IMAGE');
A=imread(fullfile(pathname, filename));
axes(handles.axes2);
imshow(A);
handles.capcha= A;
guidata(hObject, handles);
```

```
% --- Executes when entered data in editable cell(s) in uitable1.
function uitable1_CellEditCallback(hObject, eventdata, handles)
% hObject    handle to uitable1 (see GCBO)
% eventdata  structure with the following fields (see
MATLAB.UI.CONTROL.TABLE)
%   Indices: row and column indices of the cell(s) edited
%   PreviousData: previous data for the cell(s) edited
%   EditData: string(s) entered by the user
%   NewData: EditData or its converted form set on the Data property. Empty if
Data was not changed
%   Error: error string when failed to convert EditData to appropriate value for Data
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate axes1


% --- Executes during object creation, after setting all properties.
function axes2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate axes2


% --- Executes during object creation, after setting all properties.
function figure1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

5.Chương trình code Arduino.

```
#include <Servo.h>
#include <EEPROM.h>
Servo myservo;
int servopin = 10;  // analog pin used to connect the potentiometer
int led = 8;
```

```
int tam = 0;
int tam2 = 0;
int data = 0;
int pos = 0;

void setup()
{
  myservo.attach(servopin);
  Serial.begin(9600);
  pinMode(led,OUTPUT); // Cài d?t chân D2 du?i d?ng OUTPUT
  //pinMode(13,OUTPUT); // Cài d?t chân D2 du?i d?ng OUTPUT
  pos = EEPROM.read(2);
  myservo.write(pos);              // sets the servo position according to the scaled
value
  delay(1000);
  if (EEPROM.read(1)== 2)
  {
    //digitalWrite(led,LOW);
    myservo.write(0);              // sets the servo position according to the scaled
value
     delay(15);
  }
  else if (EEPROM.read(1)== 3)
  {
    //digitalWrite(led,LOW);
    myservo.write(180);              // sets the servo position according to the scaled
value
     delay(15);
  }
  else if (EEPROM.read(1)== 1)
  {
    digitalWrite(led,HIGH);
    pos = EEPROM.read(2);
    myservo.write(pos);              // sets the servo position according to the scaled
value
    delay(500);
  }


}

void loop()
{

  if (Serial.available()>0)
  {
    data = Serial.read();
```

```
  }

 if (data==2) // vui
  {

     myservo.write(0);                // sets the servo position according to the scaled
value
      delay(15);
     tam = 2;
     EEPROM.write(1, tam);
     delay(100);
     EEPROM.write(2, 0);
     delay(100);
   }

  else if(data==1)
  {
     digitalWrite(led,HIGH); // buon - Đèn led sáng
     tam = 1;
     EEPROM.write(1, tam);
     delay(100);
  }
 else if (data==3)//ngac nhien - dong cua, tat den
 {
    digitalWrite(led,LOW); //
    tam = 3;
    myservo.write(180);               // sets the servo position according to the scaled
value
    delay(15);
    EEPROM.write(1, tam);
    delay(100);
    EEPROM.write(2, 180);
    delay(100);

 }
}
```