

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP THỰC PHẨM TP. HCM
KHOA CÔNG NGHỆ ĐIỆN – ĐIỆN TỬ



ĐỒ ÁN TỐT NGHIỆP

**KHÓA ĐIỆN BẢO MẬT 2 LỚP BẰNG VÂN TAY VÀ
NHẬN ĐIỆN KHUÔN MẶT**

GVHD: Th.S Trần Hoàn

SVTH: Hà Duy Thiện

Lớp: 06DHDT2

MSSV: 2002150158

TP.HỒ CHÍ MINH, THÁNG 6, NĂM 2019

(Phiếu này phải đóng vào trang đầu tiên của báo cáo)

5. Ngày hoàn thành và nộp về khoa:

Giảng viên hướng dẫn

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

[illegible]

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

[illegible]

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi đến Thầy Trần Hoàn lời cảm ơn chân thành và sâu sắc nhất. Nhờ có sự hướng dẫn và giúp đỡ tận tình của Thầy trong suốt thời gian qua, em đã có thể thực hiện và hoàn thành Đồ Án Tốt Nghiệp. Những lời nhận xét, góp ý và hướng dẫn tận tình của Thầy đã giúp em có một định hướng đúng đắn trong suốt quá trình thực hiện Đề tài, giúp em nhìn ra được những ưu khuyết điểm của Đề tài và từng bước hoàn thiện hơn.

Đồng thời, em xin trân trọng cảm ơn các Thầy Cô của Trường đại học Công Nghiệp Thực Phẩm nói chung và của khoa Điện - Điện Tử nói riêng đã dạy dỗ chúng em suốt quãng thời gian ngồi trên ghế giảng đường Đại học. Những lời giảng của Thầy Cô trên bục giảng đã trang bị cho chúng em những kiến thức và giúp chúng em tích lũy thêm những kinh nghiệm.

Bên cạnh đó, tôi xin cảm ơn sự hỗ trợ và giúp đỡ của bạn bè trong thời gian học tập tại Trường Đại Học đại học Công Nghiệp Thực Phẩm và trong quá trình hoàn thành Luận Văn Tốt Nghiệp này.

Cuối cùng, con cũng chân thành cảm ơn sự động viên và sự hỗ trợ của gia đình và cha mẹ trong suốt thời gian học tập. Con xin gửi lời cảm ơn trân trọng nhất đến cha mẹ, người đã sinh ra và nuôi dưỡng con nên người. Sự quan tâm, lo lắng và hy sinh lớn lao của cha mẹ luôn là động lực cho con cố gắng phấn đấu trên con đường học tập của mình. Một lần nữa, con xin gửi đến cha mẹ sự biết ơn sâu sắc nhất.

TÓM TẮT ĐỒ ÁN

Thị giác máy tính và sinh trắc vân tay là những lĩnh vực khá mới mẻ và hứa hẹn nhiều bước phát triển nhảy vọt trong tương lai. Ở nước ta các đề tài nhận diện khuôn mặt vẫn chưa được phát triển đầy đủ, việc xây dựng hệ thống bảo mật ứng dụng đề tài này đã và đang được các nước trên thế giới hướng đến, ứng dụng nhiều trong thực tế như xác minh tội phạm, camera chống trộm, hệ thống chấm công, lưu trữ thông tin ở các máy ATM, các bãi giữ xe siêu thị, v.v nếu nhận diện khuôn mặt kết hợp với nhận diện dấu vân tay, thì sẽ tạo ra được một hệ thống bảo mật cực kỳ an toàn với người dùng. Đặc biệt là những nơi có yêu cầu bảo mật cao như các cơ quan pháp luật, nơi lưu trữ thông tin,...

Đề tài: **“Khóa điện bảo mật 2 lớp bằng vân tay và nhận diện gương mặt ”** được thực hiện dựa trên ý tưởng đó. Đây là một đề tài rất có ý nghĩa trong việc giúp bảo vệ tài sản của cá nhân hay một doanh nghiệp hay nhà nước, giúp chúng ta có cuộc sống an nhàn và an toàn hơn.

Trong đề tài này, ngoài những kiến thức về xử lý ảnh số, OpenCV, nhận dạng vân tay từ arduino, giao tiếp giữa raspberry và arduino, đề tài còn tập trung đi sâu nghiên cứu về mô hình Cascade of Boosted Classifier dùng đặc trưng Haar-like Feature, thực hiện huấn luyện các bộ nhận dạng dùng ứng dụng HaarTraining.

Kết hợp các kiến thức trên lại để xây dựng, thực hiện đề tài này. Sử dụng raspicam thu nhận hình ảnh chuyển về cho raspberry xử lý nhận dạng khuôn mặt và sử dụng arduino thực thi chức năng quét nhận dạng vân tay và điều khiển khóa điện.

Tp. Hồ Chí Minh, ngày 12 tháng 6 năm 2019

Hà Duy Thiện

MỤC LỤC

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN	2
NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN	3
LỜI CẢM ƠN	4
TÓM TẮT ĐỒ ÁN.....	5
DANH MỤC HÌNH ẢNH.....	8
DANH MỤC CHỮ VIẾT TẮT	10
CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI	11
1.1. Đặt vấn đề	11
1.2. Mục tiêu đề tài.....	11
1.3. Đối tượng và phạm vi nghiên cứu	12
1.3.1. Đối tượng nghiên cứu	12
1.3.2. Phạm vi nghiên cứu	12
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	13
2.1. Công nghệ sinh trắc	13
2.2. Nhận dạng vân tay.....	14
2.3. Nhận dạng khuôn mặt.....	14
2.4. Hệ thống xử lý ảnh	15
2.5. Kỹ thuật Adaboost.....	20
2.6. Đặc trưng Haar-like.....	22
2.7. Xử lý vân tay.....	24
2.8. Giới thiệu linh kiện.....	26
2.8.1. Giới thiệu Raspberry pi 3 model B+	26
2.8.2. Giới thiệu Arduino uno	27
2.8.3. Cảm biến nhận dạng vân tay R305	30
2.8.4. Module 4 Relay với opto cách ly (5VDC)	31

2.8.5.	Camera Raspberry pi V1 5MP	32
3.1.	Các phần mềm hỗ trợ	33
3.1.1.	Advanced IP Scanner	33
3.1.2.	MobaXterm	33
3.1.3.	QT Creator.....	35
3.1.4.	Arduino IDE.....	37
3.2.	Thư viện.....	38
3.2.1.	Thư viện OpenCV.....	38
3.2.2.	Thư viện Adafruit_Fingerprint	42
3.3.	Sơ đồ khối và lưu đồ	44
3.3.1.	sơ đồ khối.....	44
3.3.2.	lưu đồ	45
CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆP		47
4.1.	Sơ đồ nối dây	47
4.2.	Mô hình thực tế.....	47
4.3.	Giao diện hiển thị.....	49
CHƯƠNG 5. KẾT LUẬN VÀ ĐỊNH HƯỚNG ĐỀ TÀI		50
5.1.	Kết quả đạt được	50
5.2.	Hạn chế	50
5.3.	Hướng phát triển	50
PHỤ LỤC		51
TÀI LIỆU THAM KHẢO		60

DANH MỤC HÌNH ẢNH

Hình 2.1. Quá trình xử lý ảnh.....	15
Hình 2.2. Sơ đồ tổng quát của một hệ thống xử lý ảnh	116
Hình 2.3. Mô hình phân tần kết hợp các bộ phân loại yếu để xác định khuôn mặt	21
Hình 2.4. Kết hợp các bộ phân loại yếu thành bộ phân loại mạnh	22
Hình 2.5. đặt trung Haar-like cơ bản	23
Hình 2.6. Đặc trưng cạnh (edge features)	23
Hình 2.7. Đặc trưng đường (line features)	23
Hình 2.8. Đặc trưng xung quanh tâm (center-surround features)	23
Hình 2.9. Cách tính Integral Image của ảnh	24
Hình 2.10. Ví dụ cách tính nhanh các giá trị mức xám của vùng D trên ảnh.....	24
Hình 2.11. Quan hệ FAR, FRR, SUM và EER	25
Hình 2.12. Raspberry pi 3 mode B+	27
Hình 2.13. Arduino uno	28
Hình 2.14. Giao diện arduino IDE.....	30
Hình 2.15. Cảm biến nhận dạng vân tay R305	31
Hình 2.16. Relay	32
Hình 2.17. Camera Raspicam.....	33
Hình 3.1. Phần mềm Advanced IP Scanner	34
Hình 3.2. Phần mềm MobaXterm	35
Hình 3.3. Terminal MobaXterm.....	36
Hình 3.4. Phần mềm Qt Creator	37
Hình 3.5. Phần mềm Arduino IDE	38
Hình 3.6. Cài thư viện cho Arduino trên Raspberry	44
Hình 3.7. Kiểm tra thư viện Arduino.....	44
Hình 3.8. Sơ đồ khối	44
Hình 3.9. Lưu đồ trên Raspberry	46
Hình 3.10. Lưu đồ trên arduino	47
Hình 3.11. Lưu đồ trên giao diện	47
Hình 4.1 Sơ đồ nối dây	48
Hình 4.2 Mô hình mặt trước.....	48
Hình 4.3 Mô hình mặt sau.....	49
Hình 4.4 Giao diện hiển thị.....	49

DANH MỤC BẢNG BIỂU

Bảng 1. Các điểm lân cận của điểm ảnh	18
---	----

DANH MỤC CHỮ VIẾT TẮT

Viết tắt	Tiếng anh	Tiếng việt
IDE	Integrated Development Environment	Môi trường thiết kế hợp nhất
PIN	Personal Identification Number	Mã số định danh cá nhân
PEL	Picture Element	Điểm ảnh
CGA	Color Graphic Adaptor	Bộ điều hợp hiển thị đồ họa
FRR	False Reject Rate	Lỗi từ chối nhầm
FAR	False Accept Rate	lỗi chấp nhận nhầm
EER	Equal Error Rate	Mức độ lỗi cân bằng
SSH	Secure Shell	Vỏ an toàn: một giao thức điều khiển từ xa cho phép người dùng kiểm soát và chỉnh sửa server từ xa qua Internet

CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Đặt vấn đề

Từ xưa con người luôn mong muốn chế tạo một thiết bị "biết suy nghĩ và làm việc giống như con người". Để chế tạo được những con Robot có khả năng "suy nghĩ" và "tự hoạt động độc lập" gần giống như con người, ngoài các yêu cầu về phần cứng và phần mềm điều khiển thì yếu tố quan trọng không thể thiếu đó là "thị giác máy tính (Computer Vision)". "Thị giác máy tính" tương tự như "đôi mắt" của con người, nhờ đó Robot có thể "quan sát" được thế giới xung quanh chúng để có thể đưa ra phản ứng với bên ngoài. Vài chục năm qua, con người đã tạo ra các cảm biến, vi xử lý hình ảnh giống (và ở mức độ nào đó còn tốt hơn) khả năng nhìn của mắt người. Những thấu kính lớn hơn, hoàn hảo về mặt quang học cùng các điểm ảnh phụ bán dẫn nhỏ tới mức nano mét giúp các camera ngày nay có độ chính xác và nhạy đáng kinh ngạc, camera có thể chụp hàng ngàn ảnh mỗi giây và nhận diện từ xa với độ chính xác cao.

Ở nước ta, hoạt động nghiên cứu và thiết kế các hệ thống nhận dạng đang có những bước đầu phát triển trong các trường đại học. Các đề tài nghiên cứu khoa học và đồ án trước đây chủ yếu tập trung vào các mảng nhận dạng dấu vân tay để hỗ trợ điều khiển trong các hệ thống an ninh... Tuy nhiên, các đề tài liên quan tới giao tiếp giữa người và máy tính thông qua nhận dạng khuôn mặt kết hợp với nhận dạng vân tay vẫn chưa được phát triển đầy đủ.

Nhận thấy đây là hướng phát triển mới và có tiềm năng ứng dụng cao, đồ án **"Khóa điện bảo mật 2 lớp bằng vân tay và nhận diện gương mặt"** tập trung vào việc kết hợp những kiến thức về thị giác máy tính và xử lý ảnh số để xây dựng nên một hệ thống mà trong đó con người có thể điều khiển đối tượng thông qua việc nhận diện khuôn mặt và vân tay.

1.2. Mục tiêu đề tài

- Nhận diện được khuôn mặt người, nhận diện được vân tay.
- Tìm hiểu cách thức vận hành của quá trình nhận dạng khuôn mặt.
- Nghiên cứu các vấn đề thực tiễn cần giải quyết, đề ra các phương án giải quyết vấn đề khó khăn.
- Tìm hiểu cách thức vận hành, kết nối, cài đặt camera raspicam với raspberry pi 3 để tiến hành thu thập dữ liệu.
- Tìm hiểu về phần mềm Arduino IDE, Qt creator, cách viết code và cài đặt các thư viện hỗ trợ.
- Tìm hiểu các bộ thư viện: OpenCV, Adafruit_Fingerprint.

1.3. Đối tượng và phạm vi nghiên cứu

1.3.1. Đối tượng nghiên cứu

- Thiết bị điều khiển trung tâm: Raspberry pi 3 B+.
- Thiết bị điều khiển ngoại vi: Arduino uno.
- Cảm biến: cảm biến nhận dạng vân tay R305.
- Camera: camera raspberry (raspicam).
- Các phương pháp, thuật toán để phục vụ cho việc phát hiện và nhận diện khuôn mặt người và vân tay
- Nghiên cứu các bộ thư viện phục vụ cho quá trình xử lý: OpenCV, Adafruit_Fingerprint.

1.3.2. Phạm vi nghiên cứu

- Tìm hiểu và sử dụng cảm biến nhận dạng vân tay R305 và khóa điện 12v.
- Tập trung nghiên cứu, tìm hiểu về nhận dạng khuôn mặt (Face Recognition) chứ không chú trọng tìm hiểu phát hiện khuôn mặt (Face Detection).
- Việc xử lý ảnh, nhận dạng khuôn mặt thỏa mãn các điều kiện:
 - Ánh sáng bình thường, ngược sáng, ánh sáng đèn điện. (Với bộ cơ sở dữ liệu tự thu thập).
 - Góc ảnh: Trực diện (frontal) hoặc góc nghiêng không quá 10° .
 - Không bị che khuất (no occlusion).

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Công nghệ sinh trắc

Trong thời đại ngày nay, sự phát triển không ngừng của khoa học kỹ thuật đã giúp cho con người thuận tiện hơn trong các công việc hằng ngày. Với sự bùng nổ về công nghệ thông tin, quá trình toàn cầu hóa diễn ra nhanh chóng, sự bảo mật riêng tư thông tin cá nhân cũng như để nhận biết một người nào đó trong hàng tỉ người trên trái đất đòi hỏi phải có một tiêu chuẩn, hệ thống đảm nhận các chức năng đó. Công nghệ sinh trắc ra đời và đáp ứng được các yêu cầu trên.

Nhiều công nghệ sinh trắc đã và đang được phát triển, một số chúng đang được sử dụng trong các ứng dụng thực tế và phát huy hiệu quả cao. Các đặc trưng sinh trắc thường được sử dụng là vân tay, gương mặt, móng mắt, tiếng nói. Mỗi đặc trưng sinh trắc có điểm mạnh và điểm yếu riêng, nên việc sử dụng đặc trưng sinh trắc cụ thể là tùy thuộc vào yêu cầu của mỗi ứng dụng nhất định. Các đặc trưng sinh trắc có thể được so sánh dựa vào các yếu tố sau: tính phổ biến, tính phân biệt, tính ổn định, tính thu thập, hiệu quả, tính chấp nhận. Trong yêu cầu về bảo mật và tìm kiếm, tính phân biệt (hai người khác nhau thì đặc trưng sinh trắc này phải khác nhau) và ổn định (đặc trưng sinh trắc này không thay đổi theo từng giai đoạn thời gian tương ứng với hạng mục đối sánh nhất định) được quan tâm nhiều hơn cả. Vân tay đã được biết tới với tính phân biệt (tính chất cá nhân) và ổn định theo thời gian cao nhất, vì vậy nó là đặc trưng sinh trắc được sử dụng rộng rãi nhất. Nhận dạng sinh trắc đề cập đến việc sử dụng các đặc tính hành vi và thể chất (ví dụ: vân tay, gương mặt, chữ kí...) có tính chất khác biệt để nhận dạng một người một cách tự động. Nhận dạng vân tay và nhận dạng khuôn mặt được xem là một trong những kỹ thuật nhận dạng hoàn thiện và đáng tin cậy nhất.

Trong các tổ chức, cơ quan an ninh, quân sự, hành chính, khoa học... luôn có nhu cầu kiểm tra và trả lời các câu hỏi: “người này có phải là đối tượng đó hay không?”, “người này có được quyền truy cập và sử dụng thiết bị đó?”, “người này có được biết những thông tin đó?”... Phương pháp dựa vào thẻ bài truyền thống (ví dụ dùng chìa khóa...), phương pháp dựa vào trí thức (ví dụ dùng mật khẩu và PIN – Personal Identification Number) đã được sử dụng phổ biến nhưng thực tế đã chứng minh là không hiệu quả vì tính an toàn không cao và. Người ta nhận thấy các đặc trưng sinh trắc không thể dễ dàng bị thay thế, chia sẻ hay giả mạo., chúng được xem là đáng tin cậy hơn trong nhận dạng một người so với các phương pháp trên. Vân tay và khuôn mặt là những đặc điểm khá đặc biệt của con người bởi vì tính đa dạng của nó, mỗi người sở hữu một dấu vân tay và khuôn mặt khác nhau. Chưa có thông tin trường hợp mà có những người cùng dấu vân tay và khuôn mặt trùng nhau. Bằng việc sử dụng vân

tay và khuôn mặt, việc xác nhận một người có thể được thực hiện bằng một hệ thống nhận dạng vân tay và khuôn mặt hoàn toàn an toàn.

2.2. Nhận dạng vân tay

Ngày nay, người ta cũng lợi dụng các đặc điểm riêng biệt của vân tay để xây dựng các hệ thống bảo mật các thông tin riêng tư cho người sở hữu chúng, từ việc dùng các ổ khóa vân tay thay thế cho các ổ khóa thông thường cho đến việc dùng vân tay thay thế mật khẩu đã quá phổ biến trong thời đại công nghệ thông tin. Người ta chỉ cần quét dấu vân tay của mình qua các thiết bị chức năng là có thể mở được một cánh cửa, đăng nhập vào hệ thống máy vi tính, qua một phòng bí mật hay các trạm bảo vệ bí mật. Đó là giải pháp an ninh tuyệt đối cho những yêu cầu bảo mật của con người trong nhiều lĩnh vực như: Kiểm soát an ninh trong các cơ quan của Chính phủ, trong quân đội, ngân hàng, trung tâm lưu trữ dữ liệu... hoặc để kiểm soát ra vào của nhân viên tại các trung tâm thương mại, các tập đoàn, các đại sứ quán...

Trong lĩnh vực quản lý nhân sự, phương pháp nhận dạng vân tay còn hỗ trợ đắc lực cho việc quản lý và chấm công tại các nhà máy, xí nghiệp, công ty bằng máy các máy chấm công vân tay. Tuy nhiên, phổ biến nhất có lẽ là dấu vân tay của chúng ta qua mặt sau của chứng minh thư để xác định một cách nhanh nhất các đặc điểm, hồ sơ của một công dân đã được lưu trong cơ sở dữ liệu

2.3. Nhận dạng khuôn mặt

Các công ty lớn như Facebook, Apple và Google đang tích cực nghiên cứu vấn đề này để cung cấp các dịch vụ như tìm kiếm trực quan, tự động gắn thẻ bạn bè trong các bài đăng trên phương tiện truyền thông xã hội và khả năng sử dụng khuôn mặt của bạn để mở khóa điện thoại di động, hay có thể thanh toán dịch vụ. Các cơ quan thực thi pháp luật cũng rất quan tâm, chủ yếu để nhận diện khuôn mặt trong hình ảnh kỹ thuật số.

Phát hiện tội phạm nguy hiểm công nghệ nhận dạng khuôn mặt đang được một số lực lượng cảnh sát sử dụng để hỗ trợ vào việc thực thi pháp luật. Ví dụ, các nhân viên ở Ireland có ý định sử dụng công nghệ này để giúp xác định các nghi phạm ở các khu vực đông đúc. Các nhân viên ở New York đã sử dụng công nghệ này để bắt giữ một nghi can trong vụ hỏa hoạn.

Ngoài ra, các sĩ quan cảnh sát Trung Quốc tại đường sắt cao tốc Zhengzhou East ở thủ phủ tỉnh Hà Nam cũng đang sử dụng công nghệ này để giúp xác định các nghi phạm. Hệ thống của họ sử dụng thiết bị di động được kết nối với máy ảnh, được gắn trên một cặp kính râm.

Hệ thống kiểm tra chuyển bay, năm 2017 Baidu đã giới thiệu một hệ thống nhận diện khuôn mặt tại sân bay chính của Bắc Kinh, cho phép xác minh phi hành đoàn và nhân viên của hãng hàng không. Một sân bay ở thành phố Nanyang tỉnh Hà Nam cũng đang sử dụng một hệ thống tương tự như vậy. Tuy nhiên, hệ thống của họ đang áp dụng cho hành khách. Việc quét mặt được thực hiện và sử dụng để xác minh danh tính của họ trước khi lên máy bay.

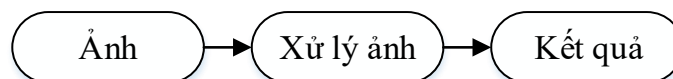
Quét khuôn mặt để mở khóa điện thoại “Mở khóa bằng khuôn mặt” là tính năng cho phép mở khóa điện thoại thông minh, cụ thể ở đây là Android bằng cách sử dụng “bản thiết kế”, tức là bản đồ cấu trúc độc đáo của khuôn mặt. Vào tháng 6/2018, theo eWeek.com, Google đã cấp bằng sáng chế một công nghệ có thể biến những biểu cảm trên khuôn mặt như một cái nháy mắt, một nụ cười,... thành một mã để mở khóa các thiết bị. Hi vọng điều này sẽ khó khăn hơn để giả mạo.

Hỗ trợ trong việc thanh toán vào tháng 7/2018, một công ty Phần Lan, Uniqul đã tạo ra một hệ thống có thể thanh toán tiền thông qua nhận dạng khuôn mặt. Tại một cửa hàng, thay vì thanh toán bằng tiền mặt hoặc thẻ tín dụng, chỉ cần đưa ra một biểu cảm cho máy quét để mua hàng. Một bài báo của Huffington Post mô tả công nghệ mới này, họ đã sử dụng nhận dạng khuôn mặt làm phương pháp bảo mật chính.

2.4. Hệ thống xử lý ảnh

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

Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý một kết luận.



Hình 2.1. Quá trình xử lý ảnh

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến $P(c_1, c_2, \dots, c_n)$. Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều. Sơ đồ tổng quát của một hệ thống xử lý ảnh:

Theo lý thuyết về nhận dạng, các mô hình toán học về ảnh được phân theo hai loại nhận dạng ảnh cơ bản: - Nhận dạng theo tham số. - Nhận dạng theo cấu trúc. Một số đối tượng nhận dạng khá phổ biến hiện nay đang được áp dụng trong khoa học và công nghệ là: nhận dạng ký tự (chữ in, chữ viết tay, chữ ký điện tử), nhận dạng văn bản (Text), nhận dạng vân tay, nhận dạng mã vạch, nhận dạng mặt người...

Cơ sở tri thức (Knowledge Base). Như đã nói ở trên, ảnh là một đối tượng khá phức tạp về đường nét, độ sáng tối, dung lượng điểm ảnh, môi trường để thu ảnh phong phú kéo theo nhiều. Trong nhiều khâu xử lý và phân tích ảnh ngoài việc đơn giản hóa các phương pháp toán học đảm bảo tiện lợi cho xử lý, người ta mong muốn bắt chước quy trình tiếp nhận và xử lý ảnh theo cách của con người. Trong các bước xử lý đó, nhiều khâu hiện nay đã xử lý theo các phương pháp trí tuệ con người. Vì vậy, ở đây các cơ sở tri thức được phát huy.

Một số khái niệm cơ bản:

Điểm ảnh (Picture Element): Gốc của ảnh (ảnh tự nhiên) là ảnh liên tục về không gian và độ sáng. Để xử lý được bằng máy tính (số), ảnh cần phải được số hóa. Số hóa ảnh là sự biến đổi gần đúng một ảnh liên tục thành một tập điểm phù hợp với ảnh thật về vị trí (không gian) và độ sáng (mức xám). Khoảng cách giữa các điểm ảnh đó được thiết lập sao cho mắt người không phân biệt được ranh giới giữa chúng. Mỗi một điểm như vậy gọi là điểm ảnh (PEL: Picture Element) hay gọi tắt là Pixel. Trong khuôn khổ ảnh hai chiều, mỗi pixel ứng với cặp tọa độ (x, y) .

Định Nghĩa: Điểm ảnh (Pixel) là một phần tử của ảnh số tại tọa độ (x, y) với độ xám hoặc màu nhất định. Kích thước và khoảng cách giữa các điểm ảnh đó được chọn thích hợp sao cho mắt người cảm nhận sự liên tục về không gian và mức xám (hoặc màu) của ảnh số gần như ảnh thật. Mỗi một phần tử trong ma trận được gọi là một phần tử ảnh.

Độ phân giải của ảnh: Độ phân giải (Resolution) của ảnh là mật độ điểm ảnh được ấn định trên một ảnh số được hiển thị. Theo định nghĩa, khoảng cách giữa các điểm ảnh phải được chọn sao cho mắt người vẫn thấy được sự liên tục của ảnh. Việc lựa chọn khoảng cách thích hợp tạo nên một mật độ phân bố đó chính là độ phân giải và được phân bố theo trục x và y trong không gian hai chiều. Ví dụ: Độ phân giải của ảnh trên màn hình CGA (Color Graphic Adaptor) là một lưới điểm theo chiều ngang màn hình: 320 điểm chiều dọc * 200 điểm ảnh (320×200). Rõ ràng, cùng màn hình CGA 12 “ta nhận thấy mịn hơn màn hình CGA 17” độ phân giải 320×200 . Lý do: cùng một mật độ (độ phân giải) nhưng diện tích mà hình rộng hơn thì độ mịn (liên tục của các điểm) kém hơn.

Mức xám của ảnh: Một điểm ảnh (pixel) có hai đặc trưng cơ bản là vị trí (x, y) của điểm ảnh và độ xám của nó. Dưới đây chúng ta xem xét một số khái niệm và thuật ngữ thường dùng trong xử lý ảnh.

- Định nghĩa: Mức xám của điểm ảnh là cường độ sáng của nó được gán bằng giá trị số tại điểm đó.

- Các thang mức xám thông thường: 16, 32, 64, 128, 256 (Mức 256 là mức phổ dụng. Lý do: từ kỹ thuật máy tính dùng 1 byte (8bit) để biểu diễn mức xám: Mức xám dùng 1 byte biểu diễn: $2^8 = 256$, tức là từ 0 đến 255).

- Ảnh đen trắng: Là ảnh có hai màu đen, trắng (không chứa màu khác) với mức xám ở các điểm ảnh có thể khác nhau.

- Ảnh nhị phân: Ảnh chỉ có hai mức đen trắng phân biệt, tức dùng 1 bit mô tả 2 mức khác nhau. Nói cách khác: mỗi điểm ảnh của ảnh nhị phân chỉ có thể là 0 hoặc 1.

- Ảnh màu: Trong khuôn khổ lý thuyết 3 màu (Red, Blue, Green) để tạo nên thế giới màu, người ta thường dùng 3 byte để mô tả mức màu, khi đó các giá trị màu: $2^8 \times 3 = 224 \approx 16,7$ triệu màu.

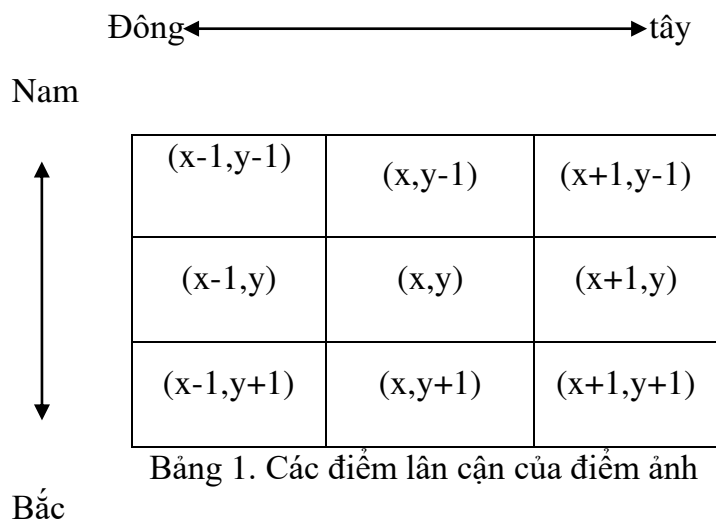
Định nghĩa ảnh số: Ảnh số là tập hợp các điểm ảnh với mức xám phù hợp dùng để mô tả ảnh gần với ảnh thật.

Quan hệ giữa các điểm ảnh:

Các lân cận của điểm ảnh (Image Neighbors): Giả sử có điểm ảnh p tại tọa độ (x, y) có 4 điểm lân cận gần nhất theo chiều đứng và ngang (có thể coi như lân cận 4 hướng chính: Đông, Tây, Nam, Bắc).

$$\{(x-1, y); (x, y-1); (x+1, y); (x, y+1)\} = N_4(p)$$

Trong đó, số 1 là giá trị logic; $N_4(p)$: tập 4 điểm lân cận của p.



Bảng 1. Các điểm lân cận của điểm ảnh

Các lân cận chéo: Các điểm lân cận chéo $N_p(P)$ (Có thể coi lân cận chéo là 4 hướng: Đông-Nam, Đông-Bắc, Tây-Nam, Tây-Bắc) $N_p(P) = \{(x+1, y+1); (x+1, y-1); (x-1, y+1); (x-1, y-1)\}$

Tập kết hợp: $N_8(p) = N_4(p) + N_p(p)$ là tập hợp 8 lân cận của điểm ảnh p.

Chú ý: Nếu (x, y) nằm ở biên (mép) ảnh; một số điểm sẽ nằm ngoài ảnh.

Các mối liên kết điểm ảnh:

Liên kết 4: Hai điểm ảnh p và q được nói là liên kết 4 với các giá trị cường độ sáng V nếu q nằm trong một các lân cận của p, tức p thuộc $N_4(p)$.

Liên kết 8: Hai điểm ảnh p và q nằm trong một các lân cận của p, tức q thuộc $N_8(p)$.

Liên kết m (liên kết hỗn hợp): Hai điểm ảnh p và q với các giá trị cường độ sáng V được nói là liên kết m nếu:

- q thuộc $N_4(p)$ hoặc
- q thuộc $N_p(p)$.

Đo khoảng cách giữa các điểm ảnh: Khoảng cách $D(p)$ giữa hai điểm ảnh p tọa độ (x, y) , q tọa độ (s, t) là hàm khoảng cách (Distance) hoặc Metric nếu:

- $D(p, q) \geq 0$ (Với $D(p, q) = 0$ nếu và chỉ nếu $p = q$).
- $D(p, q) = D(p, q)$.
- $D(p, z) \leq D(p, q) + D(q, z)$; z là một điểm ảnh khác.

Khoảng cách Euclide: Khoảng cách Euclide giữa hai điểm ảnh $p(x, y)$ và $q(s, t)$ được định nghĩa như sau: $D_e(p, q) = [(x - s)^2 + (y - t)^2]^{1/2}$

Khoảng cách khối: Khoảng cách $D_4(p, q)$ được gọi là khoảng cách khối đô thị (City-Block Distance) và được xác định như sau: $D_4(p, q) = |x - s| + |y - t|$

Giá trị khoảng cách giữa các điểm ảnh r: giá trị bán kính r giữa điểm ảnh từ tâm điểm ảnh đến tâm điểm ảnh q khác. Ví dụ: Màn hình CGA 12'' ($12'' \times 2,54\text{cm} = 30,48\text{cm} = 304,8\text{mm}$) độ phân giải 320×200 ; tỷ lệ 4/3 (chiều dài/chiều rộng). Theo định lý Pitago về tam giác vuông, đường chéo sẽ lấy tỷ lệ 5 phần ($5/4/3$: đường chéo/chiều dài/chiều rộng màn hình), khi đó độ dài thật là $(305/244/183)$ chiều rộng màn hình 183mm ứng với màn hình CGA 200 điểm ảnh theo chiều dọc. Như vậy, khoảng cách điểm ảnh lân cận của CGA 12'' là $\approx 1\text{mm}$. Khoảng cách $D_8(p, q)$ còn gọi là khoảng cách bàn cờ (Chess-Board Distance) giữa điểm ảnh p, q được xác định như sau:

$$D_8(p, q) = \max(|x - s|, |y - t|).$$

2.5. Kỹ thuật Adaboost

Kỹ thuật boosting:

Boosting là kỹ thuật dùng để tăng độ chính xác cho các thuật toán học (Learning Algorithm). Nguyên lý cơ bản của nó là kết hợp các bộ phân loại yếu (weak classifiers) thành một bộ phân loại mạnh. Trong đó weak classifiers là bộ phân loại đơn giản chỉ cần có độ chính xác trên 50%. Bằng cách này chúng ta nói bộ phân loại đã được "boost".

Xét 1 bài toán phân loại 2 lớp (mẫu cần nhận dạng được phân vào 1 trong 2 lớp) với D là tập huấn luyện gồm có n mẫu. Trước tiên, chúng ta sẽ chọn ngẫu nhiên n_1 mẫu từ tập D ($n_1 < n$) để tạo tập D_1 . Sau đó chúng ta sẽ xây dựng weak classifier đầu tiên C_1 từ tập D_1 . Tiếp theo chúng ta sẽ xây dựng tập D_2 để huấn luyện bộ phân loại C_2 , D_2 sẽ được xây dựng sao cho một nửa số mẫu của nó được phân loại đúng bởi C_1 và nửa còn lại bị phân loại sai bởi C_1 . Bằng cách này, C_2 chứa đựng những thông tin bổ sung cho C_1 . Bây giờ chúng ta sẽ huấn luyện C_2 từ D_2 .

Tiếp theo, chúng ta sẽ xây dựng tập D_3 từ những mẫu không được phân loại tốt bởi sự kết hợp giữa C_1 và C_2 : những mẫu còn lại trong D mà C_1 và C_2 cho kết quả khác nhau. Như vậy, D_3 sẽ gồm những mẫu mà C_1 và C_2 hoạt động không hiệu quả. Sau cùng, chúng ta sẽ huấn luyện bộ phân loại C_3 từ D_3 .

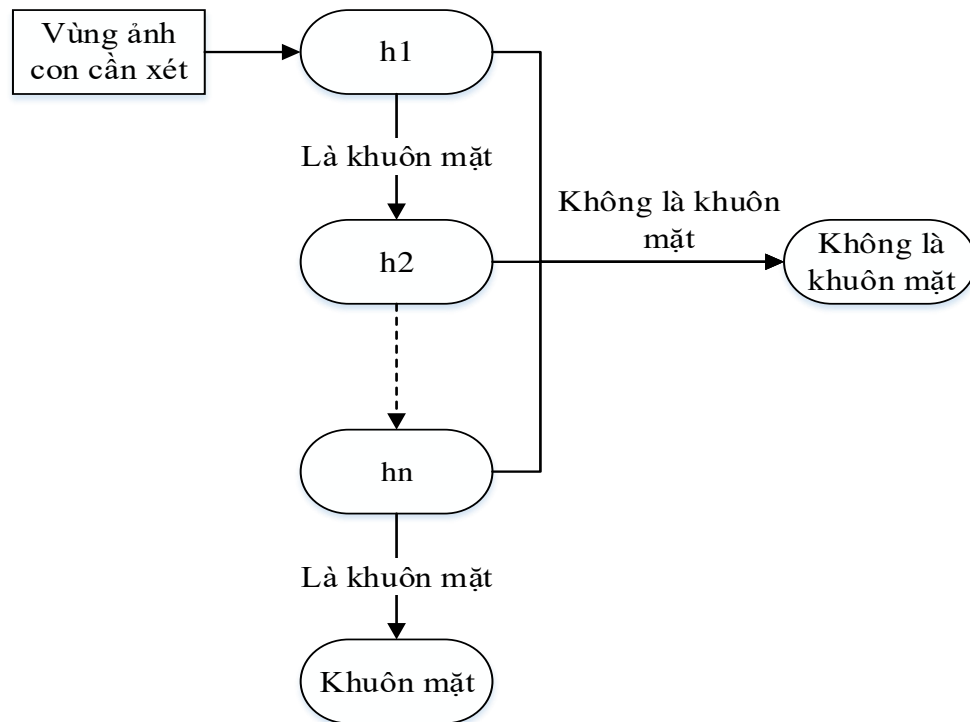
Bây giờ, chúng ta đã có một strong classifier: sự kết hợp giữa C_1 , C_2 , C_3 . Khi tiến hành nhận dạng một mẫu X , kết quả sẽ được quyết định bởi sự thoả thuận của 3 bộ C_1 , C_2 , và C_3 . Nếu cả C_1 và C_2 đều phân X về cùng một lớp thì lớp này chính là kết quả phân loại X , ngược lại nếu C_1 và C_2 phân X vào 2 lớp khác nhau, C_3 sẽ quyết định X thuộc về lớp nào.

Kỹ thuật Adaboost:

AdaBoost là một bộ phân loại mạnh phi tuyến phức dựa trên hướng tiếp cận boosting được Freund và Schapire đưa ra vào năm 1995. Adaboost cũng hoạt động trên nguyên tắc kết hợp tuyến tính các weak classifiers để hình thành một strong classifier.

Là một cải tiến của tiếp cận boosting, AdaBoost sử dụng thêm khái niệm trọng số (weight) để đánh dấu các mẫu khó nhận dạng. Trong quá trình huấn luyện, cứ mỗi weak classifiers được xây dựng, thuật toán sẽ tiến hành cập nhật lại trọng số để chuẩn bị cho việc xây dựng weak classifier kế tiếp: tăng trọng số của các mẫu bị nhận dạng sai và giảm trọng số của các mẫu được nhận dạng đúng bởi weak classifier vừa xây dựng. Bằng cách này weak classifier sau có thể tập trung vào các mẫu mà các weak classifiers trước nó làm chưa tốt. Sau cùng, các weak classifiers sẽ được kết hợp tùy theo mức độ tốt của chúng để tạo nên strong classifier.

AdaBoost kết hợp các bộ phân loại yếu sử dụng các đặc trưng Haar-like theo mô hình phân tầng (cascade) như hình dưới:

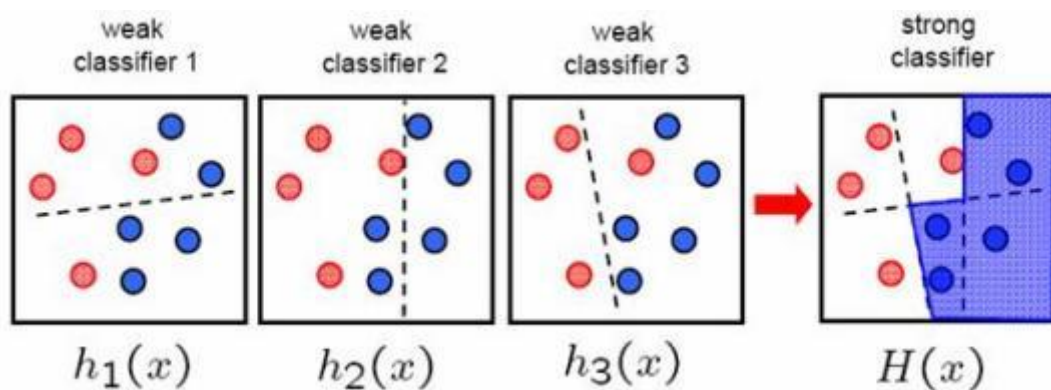


Hình 2.3. Mô hình phân tầng kết hợp các bộ phân loại yếu để xác định khuôn mặt

AdaBoost sẽ kết hợp các bộ phân loại yếu thành bộ phân loại mạnh như sau:

$$H(x) = \text{sign}(a_1h_1(x) + a_2h_2(x) + \dots + a_nh_n(x)) \quad (a = \alpha)$$

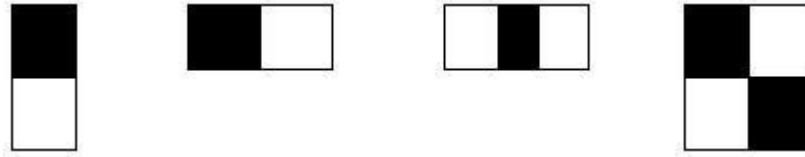
Với: $a_t \geq 0$ là hệ số chuẩn hoá cho các bộ phân loại yếu



Hình 2.4.. Kết hợp các bộ phân loại yếu thành bộ phân loại mạnh

2.6. Đặc trưng Haar-like

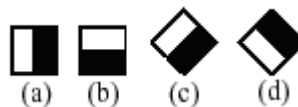
Đặc trưng Haar-like Do Viola và Jones công bố, gồm 4 đặc trưng cơ bản để xác định khuôn mặt người. Mỗi đặc trưng Haar-like là sự kết hợp của hai hay ba hình chữ nhật "trắng" hay "đen" như trong hình sau:



Hình 2.5. đặc trưng Haar-like cơ bản

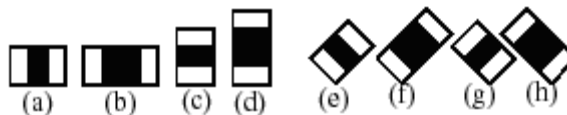
Để sử dụng các đặc trưng này vào việc xác định khuôn mặt người, 4 đặc trưng Haar-like cơ bản được mở rộng ra, và được chia làm 3 tập đặc trưng như sau:

Đặc trưng cạnh (edge features):



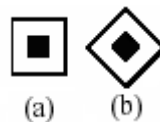
Hình 2.6. Đặc trưng cạnh (edge features)

Đặc trưng đường (line features):



Hình 2.7. Đặc trưng đường (line features)

Đặc trưng xung quanh tâm (center-surround features):



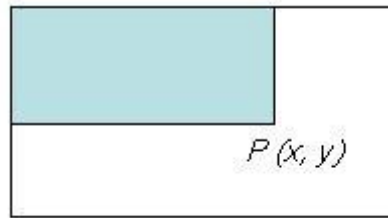
Hình 2.8. Đặc trưng xung quanh tâm (center-surround features)

Dùng các đặc trưng trên, ta có thể tính được giá trị của đặc trưng Haar-like là sự chênh lệch giữa tổng của các pixel của các vùng đen và các vùng trắng như trong công thức sau:

$$f(x) = \text{Tổng vùng đen (các mức xám của pixel)} - \text{Tổng vùng trắng (các mức xám của pixel)}.$$

Sử dụng giá trị này, so sánh với các giá trị của các giá trị pixel thô, các đặc trưng Haar-like có thể tăng/giảm sự thay đổi in-class/out-of-class (bên trong hay bên ngoài lớp khuôn mặt người), do đó sẽ làm cho bộ phân loại dễ hơn.

Như vậy ta có thể thấy rằng, để tính các giá trị của đặc trưng Haar-like, ta phải tính tổng của các vùng pixel trên ảnh. Nhưng để tính toán các giá trị của các đặc trưng Haar-like cho tất cả các vị trí trên ảnh đòi hỏi chi phí tính toán khá lớn, không đáp ứng được cho các ứng dụng đòi hỏi tính run-time. Do đó Viola và Jones đưa ra một khái niệm gọi là Integral Image, là một mảng 2 chiều với kích thước bằng với kích của ảnh cần tính các đặc trưng Haar-like, với mỗi phần tử của mảng này được tính bằng cách tính tổng của điểm ảnh phía trên (dòng-1) và bên trái (cột-1) của nó. Bắt đầu từ vị trí trên, bên trái đến vị trí dưới, phải của ảnh, việc tính toán này đơn thuần chỉ dựa trên phép cộng số nguyên đơn giản, do đó tốc độ thực hiện rất nhanh.



$$P(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

Hình 2.9. Cách tính Integral Image của ảnh

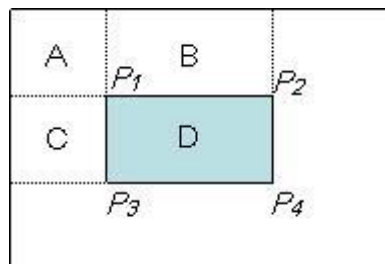
Sau khi đã tính được Integral Image, việc tính tổng các giá trị mức xám của một vùng bất kỳ nào đó trên ảnh thực hiện rất đơn giản theo cách sau:

Giả sử ta cần tính tổng các giá trị mức xám của vùng D như trong hình 4, ta có thể tính như sau:

$$D = A + B + C + D - (A+B) - (A+C) + A$$

Với $A + B + C + D$ chính là giá trị tại điểm P4 trên Integral Image, tương tự như vậy $A+B$ là giá trị tại điểm P2, $A+C$ là giá trị tại điểm P3, và A là giá trị tại điểm P1. Vậy ta có thể viết lại biểu thức tính D ở trên như sau:

$$D = \underbrace{(x_4, y_4)}_{A+B+C+D} - \underbrace{(x_2, y_2)}_{(A+B)} - \underbrace{(x_3, y_3)}_{(A+C)} + \underbrace{(x_1, y_1)}_A$$



Hình 2.10. Ví dụ cách tính nhanh các giá trị mức xám của vùng D trên ảnh

Tiếp theo, để chọn các đặc trưng Haar-like dùng cho việc thiết lập ngưỡng, Viola và Jones sử dụng một phương pháp máy học được gọi là AdaBoost. AdaBoost sẽ kết hợp các bộ phân loại yếu để tạo thành một bộ phân loại mạnh. Với bộ phân loại yếu chỉ cho ra câu trả lời chính xác chỉ hơn viện đoán một cách ngẫu nhiên một chút, còn bộ phân loại mạnh có thể đưa ra câu trả lời chính xác trên 60%.

2.7. Xử lý vân tay

Hai phương pháp nhận dạng vân tay thường được sử dụng là:

- Phương pháp 1: Dựa vào các đặc tính cụ thể của dấu vân tay, như điểm cuối, điểm rẽ nhánh của các vân trên tay, đây là phương pháp được sử dụng trong đồ án này.

- Phương pháp 2: So sánh toàn bộ đặc tính của dấu vân tay. Thực tế đây là hai mức độ của nhận dạng và dễ thấy rằng phương pháp 2 đã bao gồm phương pháp 1. Tuy nhiên do đặc điểm của vân tay, nếu ta không phải so sánh quá nhiều (cơ sở dữ liệu không quá lớn) các đặc điểm đặc biệt trên dấu vân tay đủ để ta nhận dạng ra dấu vân tay đó của ai.

Identification (Nhận diện dấu vân tay): Dấu vân tay sẽ được đưa thu thập từ một sensor để đối chiếu với database chứa các vân tay để truy ra các đặc điểm muốn truy xuất. Việc đối sánh ảnh vân tay cần nhận dạng chỉ cần được tiến hành trên các vân tay (có trong cơ sở dữ liệu) thuộc loại đã được xác định nhờ quá trình phân loại. Đây là giai đoạn quyết định xem hai ảnh vân tay có hoàn toàn giống nhau hay không và đưa ra kết quả nhận dạng, tức là ảnh vân tay cần nhận dạng tương ứng với vân tay của cá thể nào đã được lưu trữ trong cơ sở dữ liệu.

Để đánh giá một hệ thống nhận dạng vân tay ta cần phân tích hai loại lỗi đó là: Lỗi từ chối nhầm (False Reject Rate: FRR) và lỗi chấp nhận nhầm (False Accept Rate: FAR).

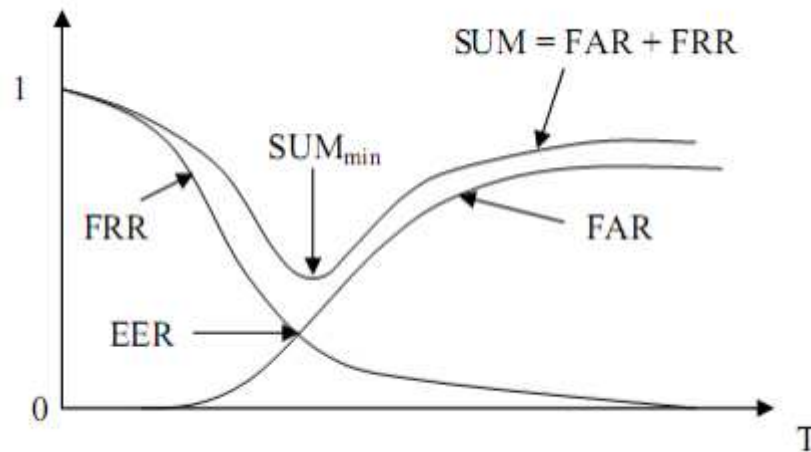
$FRR = (\text{số lỗi từ chối nhầm của các vân tay khác nhau}) / (\text{tổng số lần đối sánh của các vân tay khác nhau}).$

$FAR = (\text{số lỗi chấp nhận nhầm của các vân tay khác nhau}) / (\text{tổng số lần đối sánh của các vân tay khác nhau}).$

Giá trị của hai loại lỗi này có mối quan hệ với nhau thông qua giá trị ngưỡng đối sánh T (threshold) là sai lệch cho phép giữa mẫu cần đối sánh với mẫu được lưu trong cơ sở dữ liệu. Khi chọn giá trị ngưỡng thấp thì lỗi từ chối nhầm sẽ tăng, lỗi chấp nhận nhầm sẽ giảm và ngược lại. Hệ thống thường được đánh giá theo hai cách:

- Tỷ lệ cực tiểu $SUM_{min} = (FAR + FRR)_{min}$: Theo quan điểm dù là loại lỗi gì cũng là lỗi, do đó tỷ lệ lỗi cực tiểu SUM_{min} là hệ số lỗi nhỏ nhất mà hệ thống có thể đạt được.

- Mức độ lỗi cân bằng (Equal Error Rate: EER): Đó là điểm mà FAR và FRR bằng nhau. Mối quan hệ giữa FAR, FRR, SUM và EER theo ngưỡng T được thể hiện ở hình dưới.



Hình 2.11. Quan hệ FAR, FRR, SUM và EER

Như vậy dựa theo cách trên, để nhận diện vân tay theo đúng tên gọi của mình, cảm biến vân tay sẽ quét vân tay và so sánh với một hình ảnh quét vân tay đã được lưu lại từ trước. Do mỗi người có một vân tay khác nhau nên hệ thống có thể nhận dạng người sử dụng một cách an toàn.

2.8. Giới thiệu linh kiện

2.8.1. Giới thiệu Raspberry pi 3 model B+

Raspberry Pi là cái máy tính giá 35 USD kích cỡ như iPhone và chạy HĐH Linux. Với mục tiêu chính của chương trình là giảng dạy máy tính cho trẻ em. Được phát triển bởi Raspberry Pi Foundation – là tổ chức phi lợi nhuận với tiêu chí xây dựng hệ thống mà nhiều người có thể sử dụng được trong những công việc tùy biến khác nhau.



Hình 2.12. Raspberry pi 3 mode B+

Raspberry Pi sản xuất bởi 3 OEM: Sony, Qsida, Egoman. Và được phân phối chính bởi Element14, RS Components và Egoman.

Nhiệm vụ ban đầu của dự án Raspberry Pi là tạo ra máy tính rẻ tiền có khả năng lập trình cho những sinh viên, nhưng Pi đã được sự quan tâm từ nhiều đối tượng khác nhau. Đặc tính của Raspberry Pi xây dựng xoay quanh bộ xử lý SoC Broadcom BCM2837 (là chip xử lý mobile mạnh mẽ có kích thước nhỏ hay được dùng trong điện thoại di động) bao gồm CPU, GPU, bộ xử lý âm thanh video, và các tính năng khác... Tất cả được tích hợp bên trong chip có điện năng thấp này.

Raspberry Pi không thay thế hoàn toàn hệ thống để bàn hoặc máy xách tay . Bạn không thể chạy Windows trên đó vì BCM2837 dựa trên cấu trúc ARM nên không hỗ trợ mã x86/x64 , nhưng vẫn có thể chạy bằng Linux với các tiện ích như lướt web , môi trường Desktop và các nhiệm vụ khác . Tuy nhiên Raspberry Pi là một thiết bị đa năng đáng ngạc nhiên với nhiều phần cứng có giá thành rẻ nhưng rất hoàn hảo cho những hệ thống điện tử, thiết lập hệ thống xử lý tính toán cho những bài học trải nghiệm lập trình.

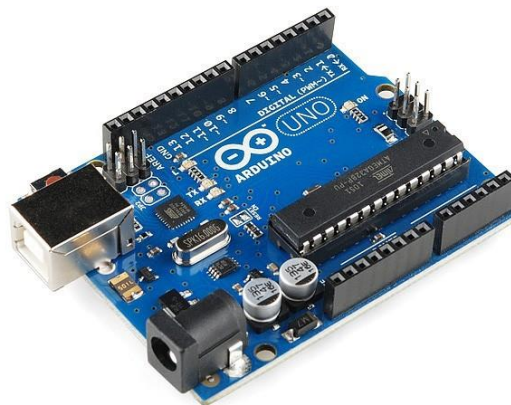
Phần cứng và cổng giao tiếp:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz.
- 1GB SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE.
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- 1 cổng HDMI cho đầu ra âm thanh / video số .
- Jack Headphone Stereo 3.5mm cho đầu ra âm thanh Analog .
- 4 cổng USB .
- 1 đầu đọc thẻ nhớ SD để tải hệ điều hành .
- 01 cổng Ethernet LAN.

2.8.2. Giới thiệu Arduino uno

Mạch Arduino Uno là dòng mạch Arduino phổ biến, khi mới bắt đầu làm quen lập trình với Arduino thì mạch Arduino thường nói tới chính là dòng Arduino UNO.

Arduino Uno R3 là dòng cơ bản, linh hoạt, thường được sử dụng cho người mới bắt đầu. Bạn có thể sử dụng các dòng Arduino khác như: Arduino Mega, Arduino Nano, Arduino Micro... Nhưng với những ứng dụng cơ bản thì mạch Arduino Uno là lựa chọn phù hợp nhất.



Hình 2.13. Arduino uno

Arduino UNO có thể sử dụng 3 vi điều khiển họ 8bit AVR là: ATmega8 (Board Arduino Uno r2), ATmega168, ATmega328 (Board Arduino Uno r3). Bộ não này có thể xử lý những tác vụ đơn giản như điều khiển đèn LED nhấp nháy, xử lý tín hiệu cho xe điều khiển từ xa, điều khiển động cơ bước, điều khiển động cơ serve, làm một trạm đo nhiệt độ – độ ẩm và hiển thị lên màn hình LCD,... hay những ứng dụng khác.

Mạch Arduino UNO R3 với thiết kế tiêu chuẩn sử dụng vi điều khiển ATmega328. Tuy nhiên nếu yêu cầu phần cứng của bạn không cao hoặc túi tiền không cho phép, bạn có thể sử dụng các loại vi điều khiển khác có chức năng tương đương nhưng rẻ hơn như ATmega8 (bộ nhớ flash 8KB) hoặc ATmega168 (bộ nhớ flash 16KB).

Nguồn sử dụng Arduino UNO R3 có thể được cấp nguồn 5V thông qua cổng USB hoặc cấp nguồn ngoài với điện áp khuyến dùng là 7-12V DC hoặc điện áp giới hạn là 6- 20V. Thường thì cấp nguồn bằng pin vuông 9V là hợp lý nhất nếu bạn không có sẵn nguồn từ cổng USB. Nếu cấp nguồn vượt quá ngưỡng giới hạn trên, bạn sẽ làm hỏng Arduino UNO.

Các chân năng lượng:

GND (Ground): cực âm của nguồn điện cấp cho Arduino UNO. Khi bạn dùng các thiết bị sử dụng những nguồn điện riêng biệt thì những chân này phải được nối với nhau. 5V: cấp điện áp 5V đầu ra. Dòng tối đa cho phép ở chân này là 500mA.

3.3V: cấp điện áp 3.3V đầu ra. Dòng tối đa cho phép ở chân này là 50mA.

Vin (Voltage Input): để cấp nguồn ngoài cho Arduino UNO, bạn nối cực dương của nguồn với chân này và cực âm của nguồn với chân GND.

IOREF: điện áp hoạt động của vi điều khiển trên Arduino UNO có thể được đo ở chân này. Và dĩ nhiên nó luôn là 5V. Mặc dù vậy bạn không được lấy nguồn 5V từ chân này để sử dụng bởi chức năng của nó không phải là cấp nguồn.

RESET: việc nhấn nút Reset trên board để reset vi điều khiển tương đương với việc chân RESET được nối với GND qua 1 điện trở 10KΩ.

Chân giao tiếp:

10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ngoài các chức năng thông thường, 4 chân này còn dùng để truyền phát dữ liệu bằng giao thức SPI với các thiết bị khác.

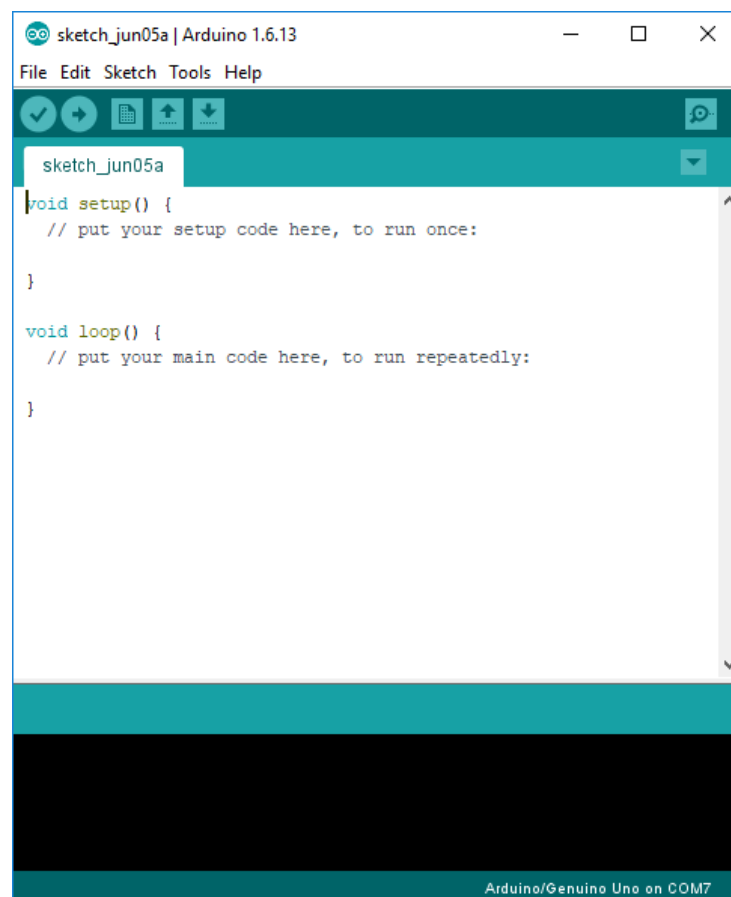
2 chân Serial: 0 (RX) và 1 (TX): dùng để gửi (transmit – TX) và nhận (receive – RX) dữ liệu TTL Serial. Arduino Uno có thể giao tiếp với thiết bị khác thông qua 2 chân này. Kết nối bluetooth thường thấy nối nom na chính là kết nối Serial không dây. Nếu không cần giao tiếp Serial, bạn không nên sử dụng 2 chân này nếu không cần thiết

Đặc biệt, Arduino UNO có 2 chân A4 (SDA) và A5 (SCL) hỗ trợ giao tiếp I2C/TWI với các thiết bị khác.

Lập trình cho Arduino:

Các thiết bị dựa trên nền tảng Arduino được lập trình bằng ngôn ngữ riêng. Ngôn ngữ này dựa trên ngôn ngữ Wiring được viết cho phần cứng nói chung. Và Wiring lại là một biến thể của C/C++. Một số người gọi nó là Wiring, một số khác thì gọi là C hay C/C++. Riêng mình thì gọi nó là “ngôn ngữ Arduino”, và đội ngũ phát triển Arduino cũng gọi như vậy. Ngôn ngữ Arduino bắt nguồn từ C/C++ phổ biến hiện nay do đó rất dễ học, dễ hiểu. Nếu học tốt chương trình Tin học 11 thì việc lập trình Arduino sẽ rất dễ thở đối với bạn. Để lập trình cho Mạch Arduino, nhà phát triển cung cấp một môi trường lập trình Arduino được gọi là Arduino IDE

(Intergrated Development Environment) như hình dưới đây.



Hình 2.14. Giao diện arduino IDE

2.8.3. Cảm biến nhận dạng vân tay R305

Cảm biến nhận dạng vân tay R305 được tích hợp nhân xử lý nhận dạng vân tay phía trong, tự động gán vân tay với 1 chuỗi data và truyền qua giao tiếp UART ra ngoài nên hoàn toàn không cần các thao tác xử lý hình ảnh, đơn giản chỉ là phát lệnh đọc/ghi và so sánh chuỗi UART nên rất dễ sử dụng và lập trình.

Cảm biến nhận dạng vân tay R305 có khả năng lưu nhiều vân tay cho 1 ID (1 người), thích hợp cho các ứng dụng bảo mật, khóa cửa, sinh trắc học,...



Hình 2.15. Cảm biến nhận dạng vân tay R305

Thông số kỹ thuật:

- Nguồn cấp: 3.6 - 6VDC
- Giao tiếp: TTL-UART hoặc USB 1.1
- Dòng điện hoạt động: 100 - 150mA
- Chế độ nhận dạng: 1:1 hoặc 1:N (1 ID nhiều vân tay, tùy thuộc vào cấu hình)
- Baudrate: 9600xN bps (mặc định N=6 tức 9600x6 = 57600bps)
- Character File Size: 256 bytes
- Image acquiring time : <0.5s
- Template size : 512 bytes
- Storage capacity: 120 Security level : 5 (1, 2, 3, 4, 5(highest))
- FAR : <0.001%
- FRR: <0.1%
- Thời gian nhận dạng trung bình: < 0.8s (1:880)
- Kích thước: 18mm*22mm

- Nhiệt độ và độ ẩm hoạt động: -10 đến 40 độ C, RH: 40%-85%
- Nhiệt độ và độ ẩm bảo quản: -40 đến 85 độ C, RH: <85%

2.8.4. Module 4 Relay với opto cách ly (5VDC)

Module 4 relay thích hợp cho các ứng dụng đóng ngắt điện thế cao AC hoặc DC, các thiết bị tiêu thụ dòng lớn, module thiết kế nhỏ gọn, có opto và transistor cách ly, kích đóng bằng mức thấp (0V) phù hợp với mọi loại MCU và thiết kế có thể sử dụng nguồn ngoài giúp cho việc sử dụng trở nên thật linh động và dễ dàng.

Thông số kỹ thuật:

- Sử dụng điện áp nuôi 5VDC.
- 4 Relay đóng ngắt ở điện thế kích bằng 0V nên có thể sử dụng cho cả tín hiệu 5V hay 3v3 (cần cấp nguồn ngoài), mỗi Relay tiêu thụ dòng khoảng 80mA.
- Điện thế đóng ngắt tối đa: AC250V - 10A hoặc DC30V - 10A.
- Có đèn báo đóng ngắt trên mỗi Relay.



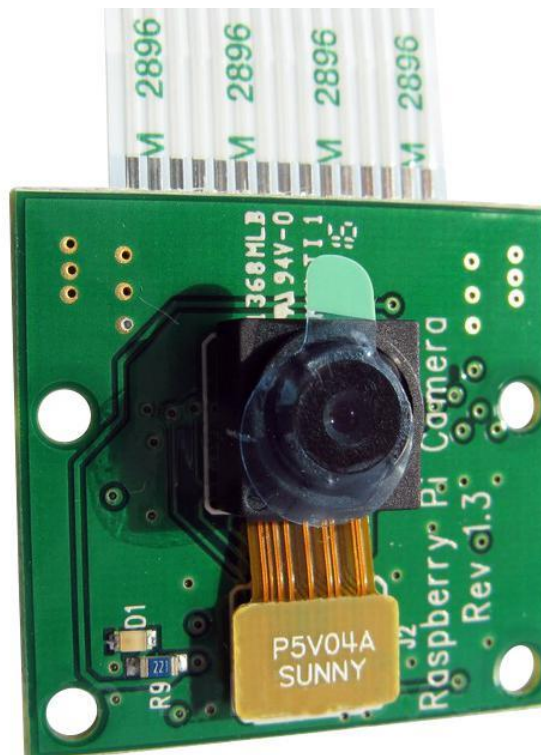
Hình 2.16. Relay

2.8.5. Camera Raspberry pi V1 5MP

Camera Raspberry Pi V1 5MP là Version đầu tiên của module camera cho Raspberry Pi với cảm biến OV5647 độ phân giải 5MP, sử dụng tương thích với tất cả các dòng Raspberry Pi từ trước đến nay, chất lượng hình ảnh tốt, độ phân giải cao và có khả năng quay phim ở chất lượng HD.

Thông số kỹ thuật:

- Module Camera V1 cho Raspberry Pi.
- Cảm biến: OV5647.
- Độ phân giải: 5MP.
- Độ phân giải hình: 2592x1944 pixel.
- Quay phim HD 1080P 30, 720P 60, VGA 640x480P 60.
- Lens: Fixed Focus.
- Conector: Ribbon conector.
- Kích thước: 25x24x9mm.



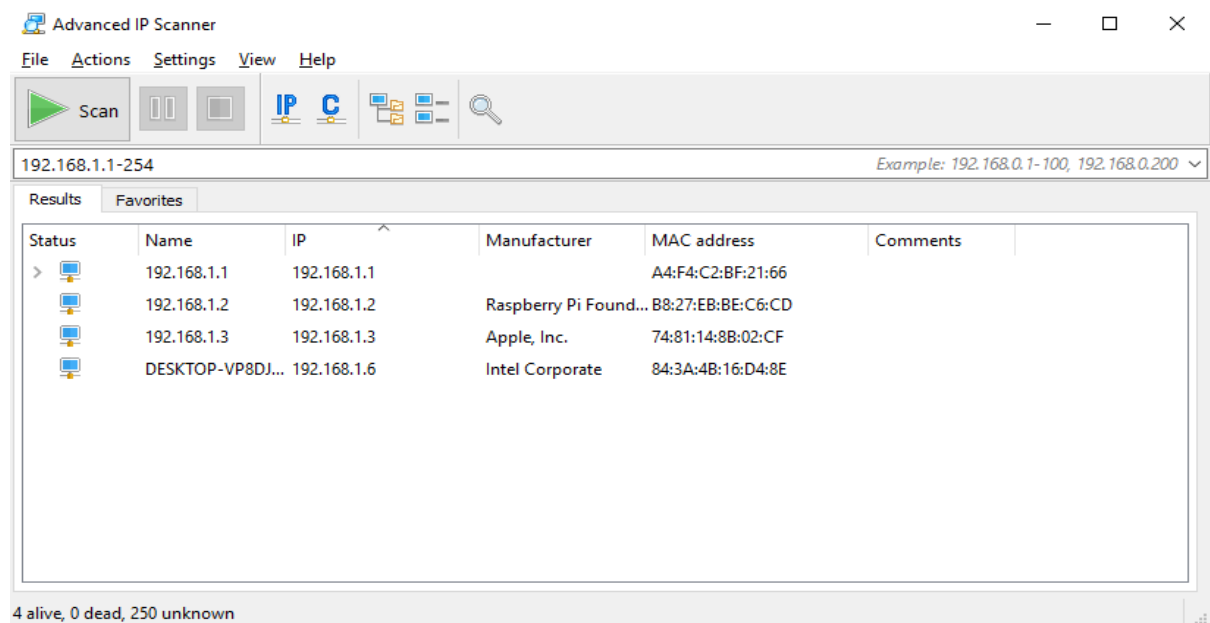
Hình 2.17. Camera Raspicam

CHƯƠNG 3. CƠ SỞ THỰC HIỆN

3.1. Các phần mềm hỗ trợ

3.1.1. Advanced IP Scanner

Trình quét mạng miễn phí và tin cậy nhằm phân tích mạng LAN. Chương trình hiển thị tất cả các thiết bị mạng, cho phép bạn truy cập các thư mục chia sẻ, cung cấp khả năng điều khiển máy tính từ xa (thông qua RDP và Radmin) và thậm chí có thể tắt máy tính từ xa. Chương trình dễ sử dụng và có thể chạy ở dạng phiên bản cơ động. Phần mềm này giúp tìm địa chỉ IP của raspberry khi đã kết nối cùng mạng. Ở đây ta thấy địa chỉ IP của raspberry là: 192.168.1.2.



Hình 3.1. Phần mềm Advanced IP Scanner

3.1.2. MobaXterm

Đây là một SSH Client (Secure Shell Client), ứng dụng quản trị máy chủ từ xa thông qua SSH dành cho máy khách khá tốt và có nhiều tiện lợi.

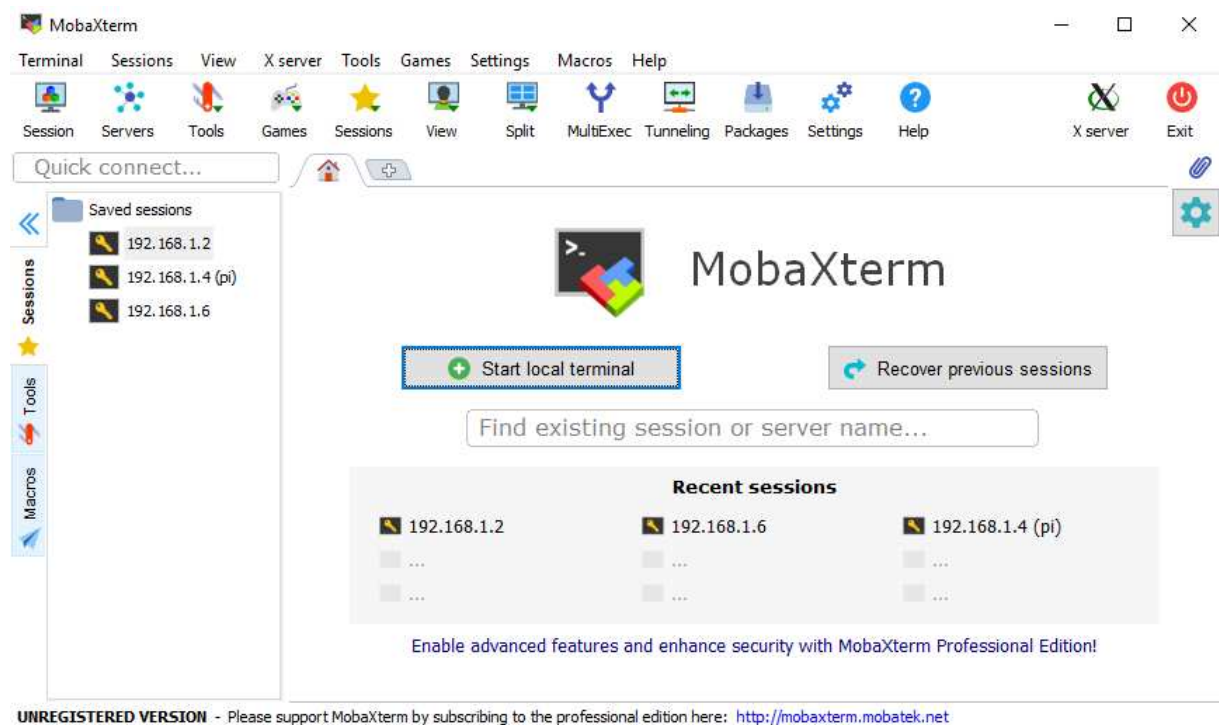
đối với MobaXterm, có thể truy cập vào SSH,...và đặc biệt là có thể sử dụng ứng dụng này để gõ lệnh UNIX ở môi trường Windows.

MobaXterm có các tính năng chính như:

- Lưu trữ thông tin của nhiều server theo dạng profile. Muốn kết nối vào server nào thì click 1 cái, nó sẽ tự mở tab mới.
- Có tính năng Multi Execution – Nghĩa là gõ một lệnh thực thi cùng lúc trên nhiều server.
- Kết nối một server vô nhiều giao thức khác nhau.
- Miễn phí hoàn toàn.

- Hỗ trợ lưu session, không cần gõ lại mật khẩu mà chỉ cần nhập username là nó tự tìm session phù hợp.
- Remote vào máy tính khác, giống như Teamviewer.
- Tự động truy cập vào thư mục trên sFTP nơi SSH đang thực thi.
- Hỗ trợ quay lại Macro, ví dụ như nếu gõ mấy command để cài ứng dụng gì đó thì có thể lưu vào Macro và lần sau muốn dùng chỉ cần ấn 1 cái.
- Tích hợp nhiều công cụ nhỏ như Editor, máy tính và kể cả game giải trí trong lúc chờ đợi làm việc nữa.

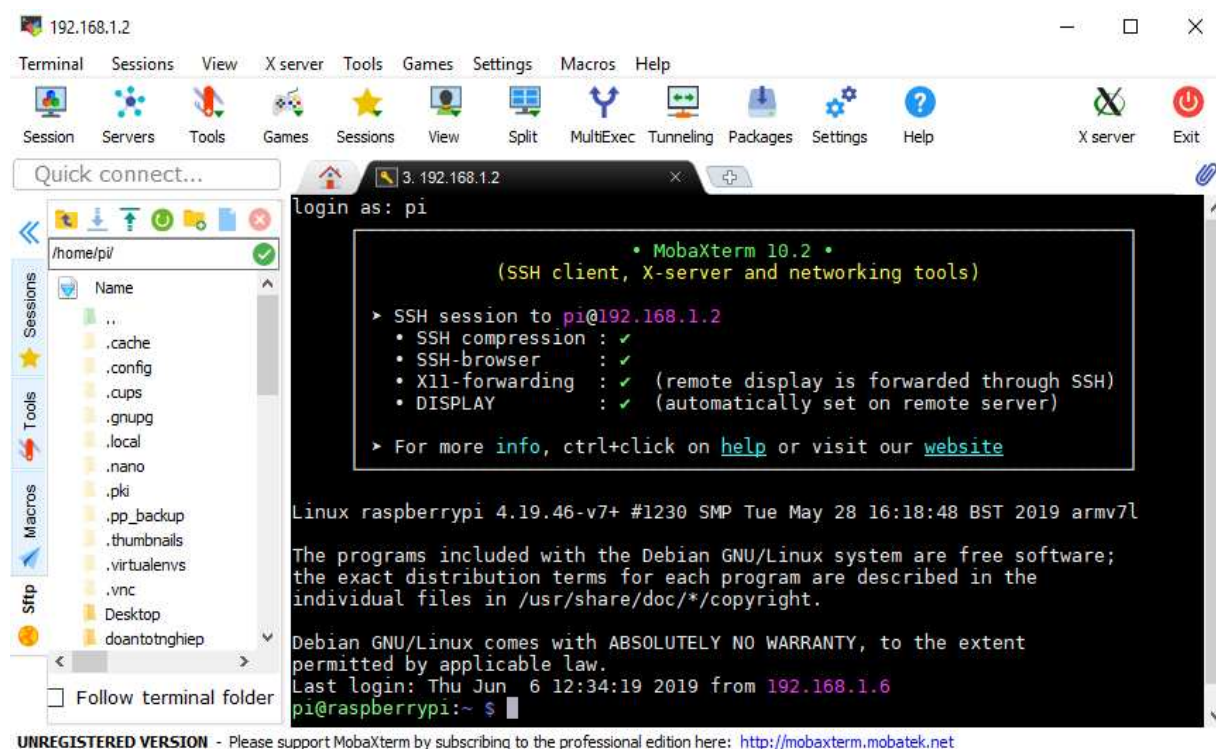
Và nhiều tính năng linh ta linh tinh khác, nó khá rộng lớn.



Hình 3.2. Phần mềm MobaXterm

Để tiến hành kết nối đến raspberry ta thực hiện các bước: nhấn vào “Session”, chọn “SSH”, tại “Remote Host” ta nhập địa chỉ IP của raspberry đã được dò ở phần trước là: “192.168.1.2”, rồi nhấn “OK”.

Sau khi nhấn “OK” xuất hiện “login as:” thì nhập “pi” (pi là tên đặt cho raspberry) rồi enter. Nếu chưa đăng nhập lần nào thì sẽ phải nhập password đã đặt, nếu đã đăng nhập thì sẽ vào thẳng giao diện terminator.



Hình 3.3. Terminal MobaXterm

3.1.3. Qt Creator

Qt Creator là một tiện ích đa nền tảng làm việc thông qua môi trường phát triển tích hợp (viết tắt là IDE) hỗ trợ các nhà phát triển tạo ứng dụng cho nền tảng máy tính và thiết bị di động.

Mặc dù có nhiều tham số riêng nhưng Qt Creator mang đến một giao diện rõ ràng, đơn giản, cho phép người dùng thiết lập một dự án mới bằng cách sử dụng phương pháp từng bước tiện dụng, đặc biệt đối với người chưa có kinh nghiệm vì họ sẽ được trợ giúp trong toàn bộ quá trình.

Cũng là ứng dụng lập trình C++, nhưng phần mềm Bloodshed Dev C++ lại sử dụng trình biên dịch Mingw, tích hợp với Cygwin và nhiều chương trình biên dịch trên nền tảng GCC. Với Bloodshed Dev C++ bạn có thể viết nhiều loại ứng dụng trên nền tảng mã nguồn mở khá nhanh chóng.

Qt Creator cung cấp một số công cụ được dùng để tùy chỉnh bố cục chương trình đang tạo và bộ chỉnh sửa văn bản cài sẵn hỗ trợ cho ngôn ngữ lập trình C++ và QML, đánh dấu cú pháp, tùy chọn hoàn thành mã và kiểm tra cú pháp mã lệnh.

Hơn nữa, chương trình cho phép xây dựng, chạy và triển khai các ứng dụng QT cho môi trường máy tính và thiết bị di động, sử dụng chế độ gỡ lỗi để phân tích trạng thái ứng dụng (tích hợp nhiều trình gỡ lỗi như Trình gỡ lỗi ký tự GNU, Trình gỡ lỗi điều khiển Microsoft và JavaScript) cũng như các gói phần mềm cài đặt thiết kế phát hành ở nhiều cửa hàng và kênh ứng dụng khác nhau.

Có thể xem thêm Orwell Dev C++ để tạo và chỉnh sửa nhiều định dạng file phổ biến như nguồn .C, cpp, .cc, .C ++ hoặc .CP. Bên cạnh đó Orwell Dev C++ còn cung cấp giao diện khá đẹp mắt cho phép lựa chọn môi trường hoạt động, điều khiển tab dễ dàng.

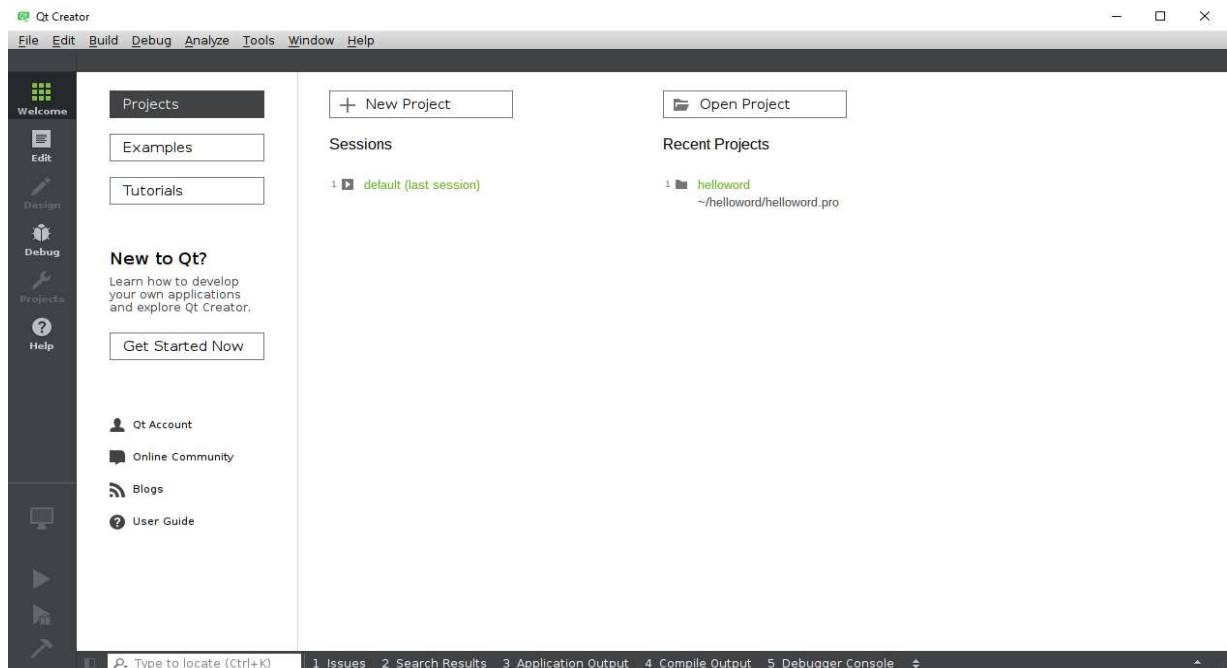
Khi nói đến lập cấu hình cài đặt một dự án mới, có thể lựa chọn tiện ích muốn tạo (Các công cụ Qt Quick hay HTML5 hay Qt), thiết lập ngôn ngữ lập trình, sử dụng hệ điều khiển phiên bản và thiết kế các phiên bản tệp dự án riêng biệt.

Quá trình phát triển ứng dụng Qt Quick và các tiện ích công cụ hoàn thành với hai bộ chỉnh sửa hình ảnh tích hợp (Trình thiết kế Qt Quick và Trình thiết kế Qt).

Qt Creator có nhiều công cụ cho phép ghi và chỉnh sửa mã dễ dàng hơn vì có thể sử dụng chức năng tìm kiếm tăng dần và nâng cao, thực hiện thao tác phân tích lại mã để tìm và đổi tên các ký hiệu cũng như lập cấu hình bộ chỉnh sửa mã nguồn về font chữ, màu sắc và thụt đầu dòng.

Thêm vào đó, Qt Creator bao gồm công cụ phân tích mã Valgrind dùng để xác định lỗi và rò rỉ bộ nhớ, các hệ thống xây dựng khác nhau (qmake và CMake), hoạt động của dòng lệnh và các công cụ bên ngoài.

Để khởi động QT Creator(đã cài trên raspberry), nhập vào terminator lệnh: “qtccreator” rồi enter.






Hình 3.4. Phần mềm Qt Creator

3.1.4. Arduino IDE

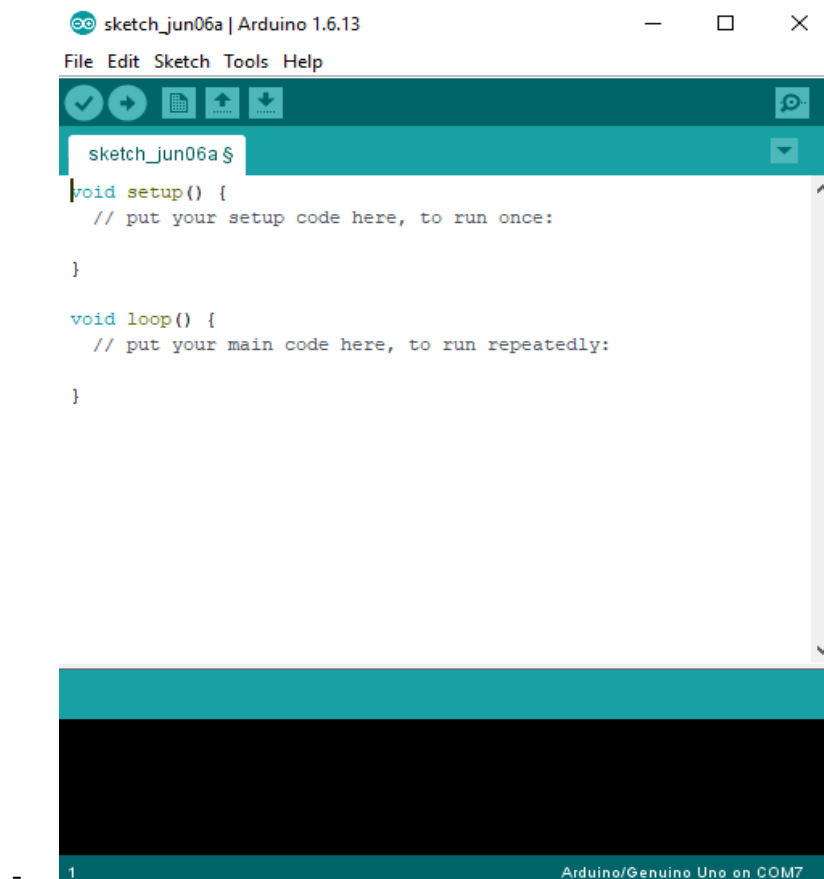
Môi trường lập trình Arduino IDE có thể chạy trên ba nền tảng phổ biến nhất hiện nay là Windows, Macintosh OSX và Linux. Do có tính chất nguồn mở nên môi trường lập trình này hoàn toàn miễn phí và có thể mở rộng thêm bởi người dùng có kinh nghiệm.

Ngôn ngữ lập trình có thể được mở rộng thông qua các thư viện C++. Và do ngôn ngữ lập trình này dựa trên nền tảng ngôn ngữ C của AVR nên người dùng hoàn toàn có thể nhúng thêm code viết bằng AVR C vào chương trình nếu muốn.

Arduino Toolbar: có một số button và chức năng của chúng như sau :

- Verify : kiểm tra code có lỗi hay không .
- Upload: nạp code đang soạn thảo vào Arduino .
- New, Open, Save : Tạo mới, mở và Save sketch .
- Serial Monitor : Đây là màn hình hiển thị dữ liệu từ Arduino gửi lên máy tính

hoặc tổ hợp phím CTRL + SHIFT + M 



Hình 3.5. Phần mềm Arduino IDE

3.2. Thư viện

3.2.1. Thư viện OpenCV

OpenCV (thị giác máy tính) là một thư viện các chức năng lập trình chủ yếu nhằm vào tầm nhìn máy tính thời gian thực . Được phát triển đầu tiên bởi Intel , sau này được Willow Garage hỗ trợ sau đó là Itseez (sau này được Intel mua lại).

Lịch sử phát triển:

Chính thức ra mắt vào năm 1999, dự án OpenCV ban đầu là một sáng kiến của Intel Research nhằm thúc đẩy các ứng dụng tăng cường CPU , một phần của một loạt các dự án bao gồm dò tia thời gian thực và tường hiển thị 3D. Những người đóng góp chính cho dự án bao gồm một số chuyên gia tối ưu hóa ở Intel Russia, cũng như Nhóm Thư viện Hiệu suất của Intel. Phiên bản alpha đầu tiên của OpenCV đã được phát hành ra công chúng tại Hội nghị IEEE về Tầm nhìn máy tính và Nhận dạng mẫu năm 2000, và năm bản beta được phát hành từ năm 2001 đến 2005. Phiên bản 1.0 đầu tiên được phát hành vào năm 2006. Phiên bản 1.1 "trước khi phát hành "Được phát hành vào tháng 10 năm 2008.

Phiên bản chính thứ hai của OpenCV là vào tháng 10 năm 2009. OpenCV 2 bao gồm những thay đổi lớn đối với giao diện C ++ , hướng đến các mẫu dễ dàng hơn, an toàn hơn về kiểu, các chức năng mới và triển khai tốt hơn cho các hiệu suất hiện có về mặt hiệu suất (đặc biệt là trên đa hệ thống cốt lõi). Các bản phát hành chính thức hiện đang diễn ra sáu tháng một lần và việc phát triển hiện được thực hiện bởi một nhóm độc lập của Nga được hỗ trợ bởi các tập đoàn thương mại.

Vào tháng 8 năm 2012, hỗ trợ cho OpenCV đã được tiếp quản bởi một tổ chức phi lợi nhuận OpenCV.org, nơi duy trì một nhà phát triển và trang web mở cho người dùng. Vào tháng 5 năm 2016, Intel đã ký một thỏa thuận mua lại Itseez, một nhà phát triển hàng đầu của OpenCV.

Ứng dụng opencv:

- 2D and 3D feature toolkits
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human-computer interaction (HCI)
- Mobile robotics
- Motion understanding
- Object identification
- Segmentation and recognition
- Stereopsis stereo vision: depth perception from 2 cameras

- Structure from motion (SFM)
- Motion tracking
- Augmented reality

Các thư viện hỗ trợ cho thuật toán học máy:

- Boosting
- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)
- Deep neural networks (DNN)

Hướng dẫn cài thư viện opencv trên Raspberry:

Thực hiện lệnh dưới đây để xóa phiên bản OpenCV cũ:

```
sudo apt-get remove libopencv*
```

```
sudo apt-get autoremove
```

Kiểm tra dung lượng trống Raspberry bằng câu lệnh dưới đây:

```
df -h
```

Tiến hành cài đặt opencv:

Cập nhật hệ thống:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

```
sudo rpi-update
```

```
sudo reboot
```

Cài đặt devel tools:

```
sudo apt-get install build-essential cmake cmake-curses-gui pkg-config
```

Cài đặt các thư viện cần thiết:

```
sudo apt-get install \
```

```
libjpeg-dev \
```

```
libtiff5-dev \
```



```
libjasper-dev \  
libpng12-dev \  
libavcodec-dev \  
libavformat-dev \  
libswscale-dev \  
libeigen3-dev \  
libxvidcore-dev \  
libx264-dev \  
libgtk2.0-dev
```

Cài thư viện v4l từ repository:

```
sudo apt-get -y install libv4l-dev v4l-utils
```

Bật kernel module:

```
sudo modprobe bcm2835-v4l2
```

Cài đặt python-dev & numpy:

```
sudo apt-get install python2.7-dev python2-numpy
```

```
sudo apt-get install python3-dev python3-numpy
```

Tải nguồn OpenCV:

```
sudo mount /dev/your-dev-name /home/pi/usbmem
```

```
mkdir /home/pi/usbmem/opencv
```

```
cd /home/pi/usbmem/opencv
```

```
wgethttps://github.com/opencv/opencv/archive/3.2.0.zip -O opencv_source.zip
```

```
wgethttps://github.com/opencv/opencv_contrib/archive/3.2.0.zip-O
```

opencv_contrib.zip

Giải nén Unzip files opencv vừa tải về:

```
cd /home/pi/usbmem/opencv
```

```
unzip opencv_source.zip
```

```
unzip opencv_contrib.zip
```

Tạo thư mục làm việc:

```
cd /home/pi/usbmem/opencv/opencv-3.2.0
```

```
mkdir build
```

```
cd build
```

Bắt đầu cấu hình build sử dụng command line switch:

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D BUILD_WITH_DEBUG_INFO=OFF \
      -D BUILD_DOCS=OFF \
      -D BUILD_EXAMPLES=OFF \
      -D BUILD_TESTS=OFF \
      -D BUILD_opencv_ts=OFF \
      -D BUILD_PERF_TESTS=OFF \
      -D INSTALL_C_EXAMPLES=ON \
      -D INSTALL_PYTHON_EXAMPLES=ON \
      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib-
3.2.0/modules\
      -D ENABLE_NEON=ON \
      -D WITH_LIBV4L=ON \
```

Sau khi chạy CMake chúng ta sẽ nhận được:

```
Configuring done
```

```
Generating done
```

```
Build files have been written to: /home/pi/usbmemb/opencv/opencv-3.2.0/build
```

Tiến hành Build OpenCV:

```
Make -j4
```

Quá trình biên dịch khoảng 2 tiếng chúng ta có thể thấy kết quả:

```
[100%] Built target ...
```

Cài đặt OpenCV:

```
sudo make install
```

```
sudo ldconfig
```

Kiểm tra cài đặt:

```
import cv2  
  
print(cv2.__version__)
```

nếu cài đặt thư viện thành công thì chạy 2 câu lệnh sẽ ra phiên bản opencv đã cài. Nếu ko ra thì đã cài thất bại phải tiến hành kiểm tra lại xem có bỏ sót bước nào không nếu có thì chỉ việc chạy lại câu lệnh đó.

3.2.2. Thư viện Adafruit_Fingerprint

Thư viện Adafruit_Fingerprint là một thư viện mã nguồn mở, đáp ứng nhu cầu sử dụng cảm biến nhận dạng vân tay. Adafruit_Fingerprint được viết với nhiều ví dụ để người đọc có thể tham khảo hỗ trợ người dùng khi làm việc với cảm biến dễ dàng.

Hướng dẫn cài thư viện Adafruit_Fingerprint thực hiện theo các bước:

- Tại IDE arduino, chọn Sketch.
- Tại Sketch, chọn Include Library.
- Tại Include Library, chọn Manage Libraries..., xuất hiện bảng library Manage.
- Tại ô tìm kiếm của bảng library Manage, nhập vào “Adafruit_Fingerprint”.
- Chọn thư viện Adafruit Fingerprint Sensor Library, chọn install.

Như vậy là đã cài xong thư viện Adafruit_Fingerprint để sử dụng cảm biến nhận dạng vân tay.

Hướng dẫn cài đặt thư viện Adafruit_Fingerprint cho raspberry:

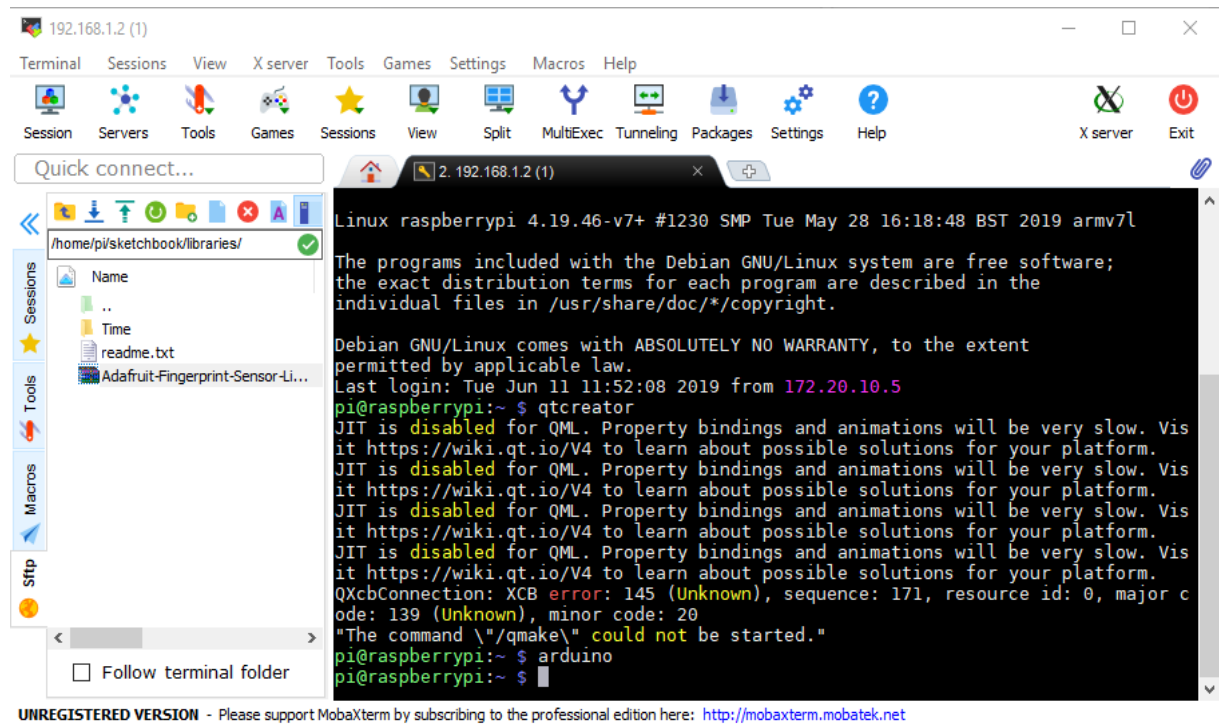
Tải thư viện Adafruit_Fingerprint tại: <https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library> xuống windows.

Upload file:” Adafruit-Fingerprint-Sensor-Library-master.zip” từ windows sang raspberry theo đường dẫn: /home/pi/sketchbook/libraries/ (đường dẫn này sẽ đúng khi đã cài IDE arduino trên raspberry) như hình.

Sau khi upload file xong, tại cửa sổ terminal trên MobaXterm, nhập:” \$ cd sketchbook/libraries/ “ để truy cập vào thư mục libraries.

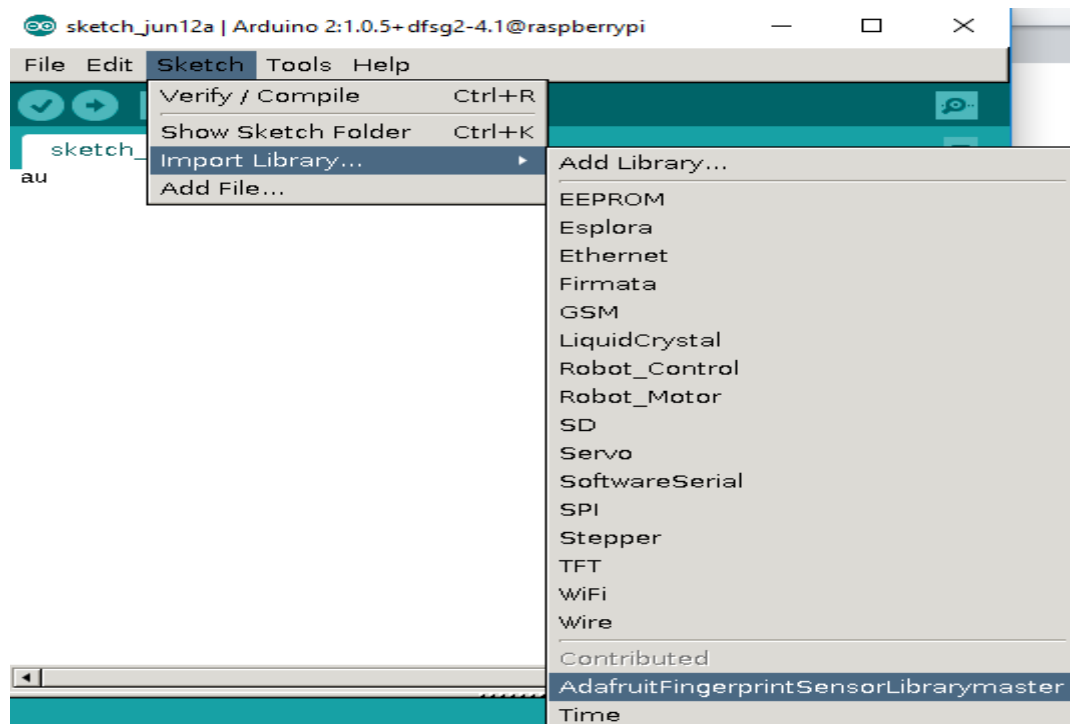
Nhập: “\$ unzip Adafruit-Fingerprint-Sensor-Library-master.zip” để giải nén file .zip.

Sau khi giải nén file .zip, tiến hành đổi tên cho thư viện, loại các ký tự đặc biệt. Ở đây là dấu “-“.



Hình 3.6. Cài thư viện cho Arduino trên Raspberry

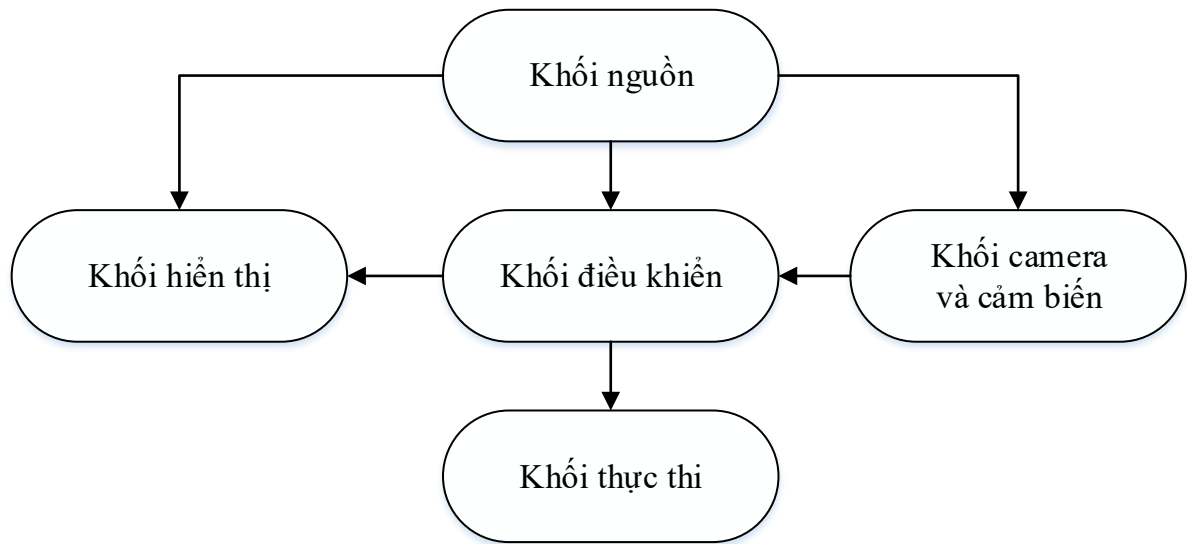
Sau khi đổi tên xong ta mở arduino lên bằng việc nhập vào: “arduino” trên terminal, và kiểm tra lại xem đã có thư viện chưa.



Hình 3.7. Kiểm tra thư viện Arduino

3.3. Sơ đồ khối và lưu đồ

3.3.1. sơ đồ khối



Hình 3.8. Sơ đồ khối

Khối nguồn:

- Nguồn cấp cho camera là nguồn được lấy từ raspberry.
- Nguồn cấp cho arduino là nguồn từ raspberry.
- Nguồn cấp cho khối thực thi là nguồn 12v.

Khối điều khiển: xử lý tất cả dữ liệu xử lý ảnh truyền về máy tính và kết nối với arduino để điều khiển khối thực thi.

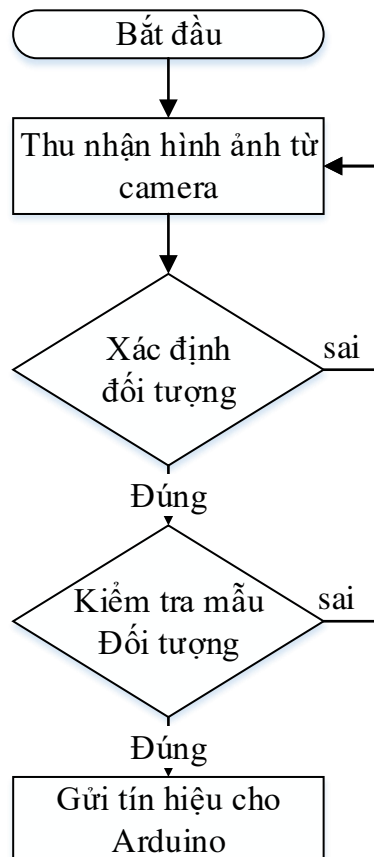
Khối camera và cảm biến: thu nhận hình ảnh và vân tay để xử lý.

Khối hiển thị: hiển thị trên máy tính bằng Qt creator thông qua MobaXterm.

Khối thực thi: nhận tín hiệu từ arduino để thực hiện công việc được giao, led sáng tắt, khóa điện đóng mở.

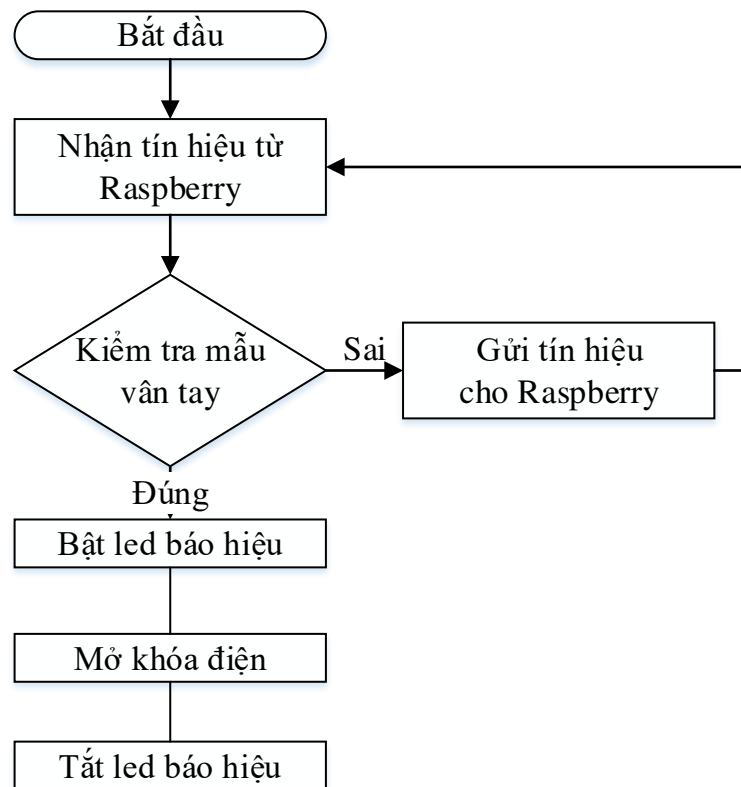
3.3.2. lưu đồ

- Lưu đồ xử lý trên raspberry:



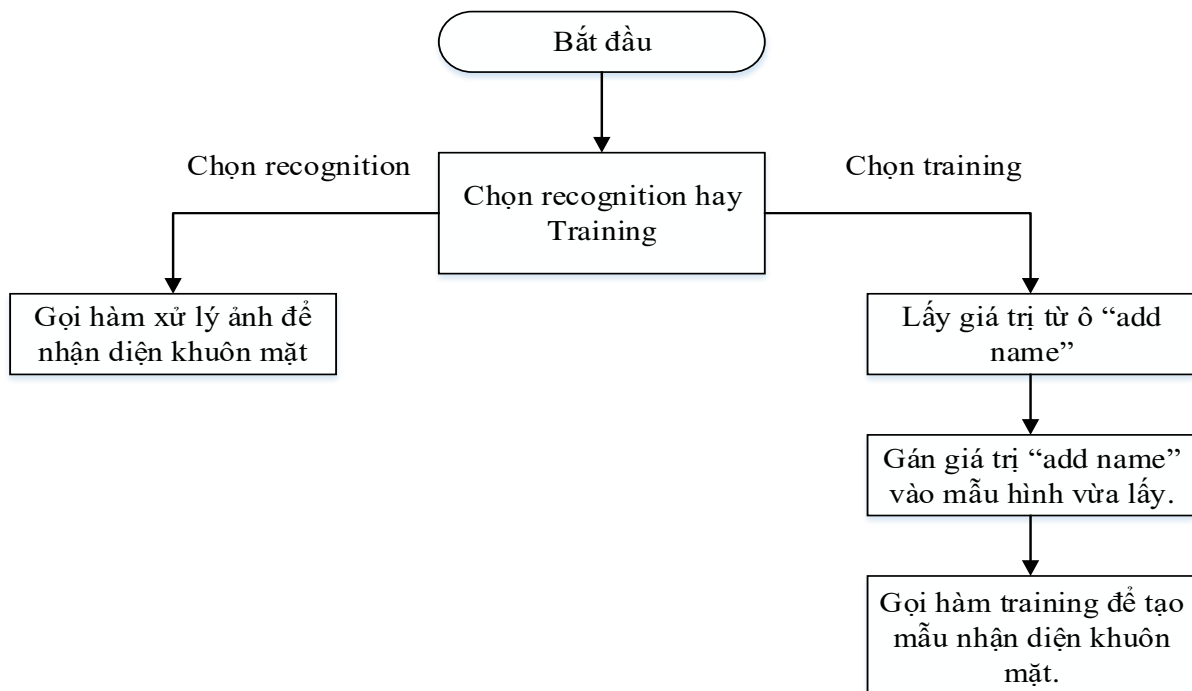
Hình 3.9. Lưu đồ trên Raspberry

- Lưu đồ xử lý trên arduino:



Hình 3.10. Lưu đồ trên arduino

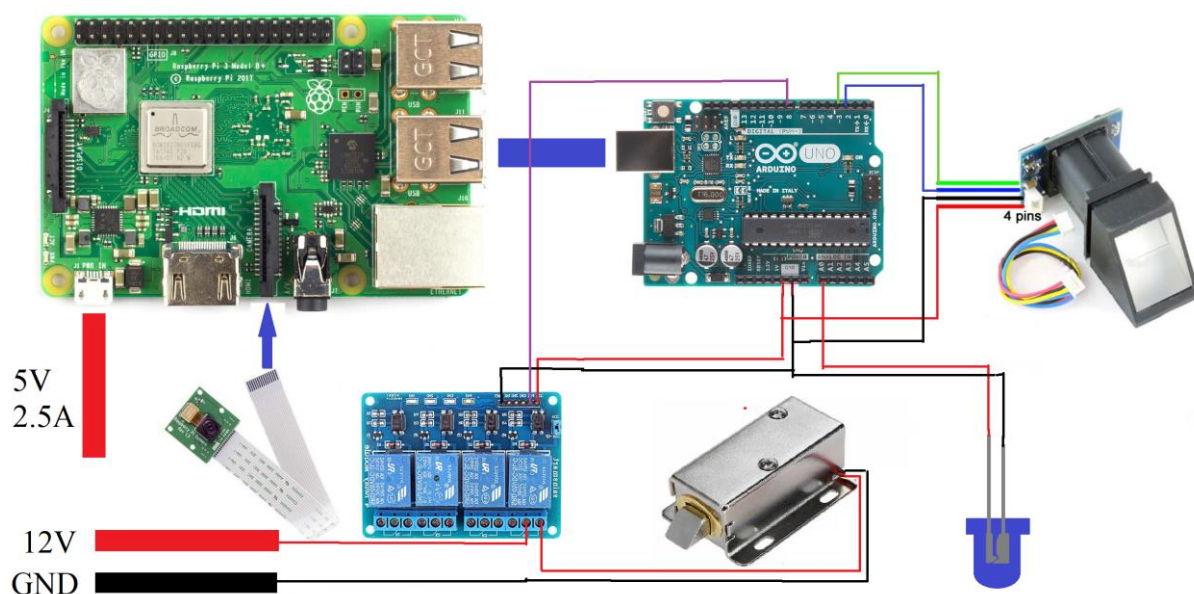
- Lưu đồ xử lý trên giao diện:



Hình 3.11. Lưu đồ trên giao diện

CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM

4.1. Sơ đồ nối dây



Hình 4.1. Sơ đồ nối dây

4.2. Mô hình thực tế

Mặt trước:



Hình 4.2. Mô hình mặt trước

Mặt sau:

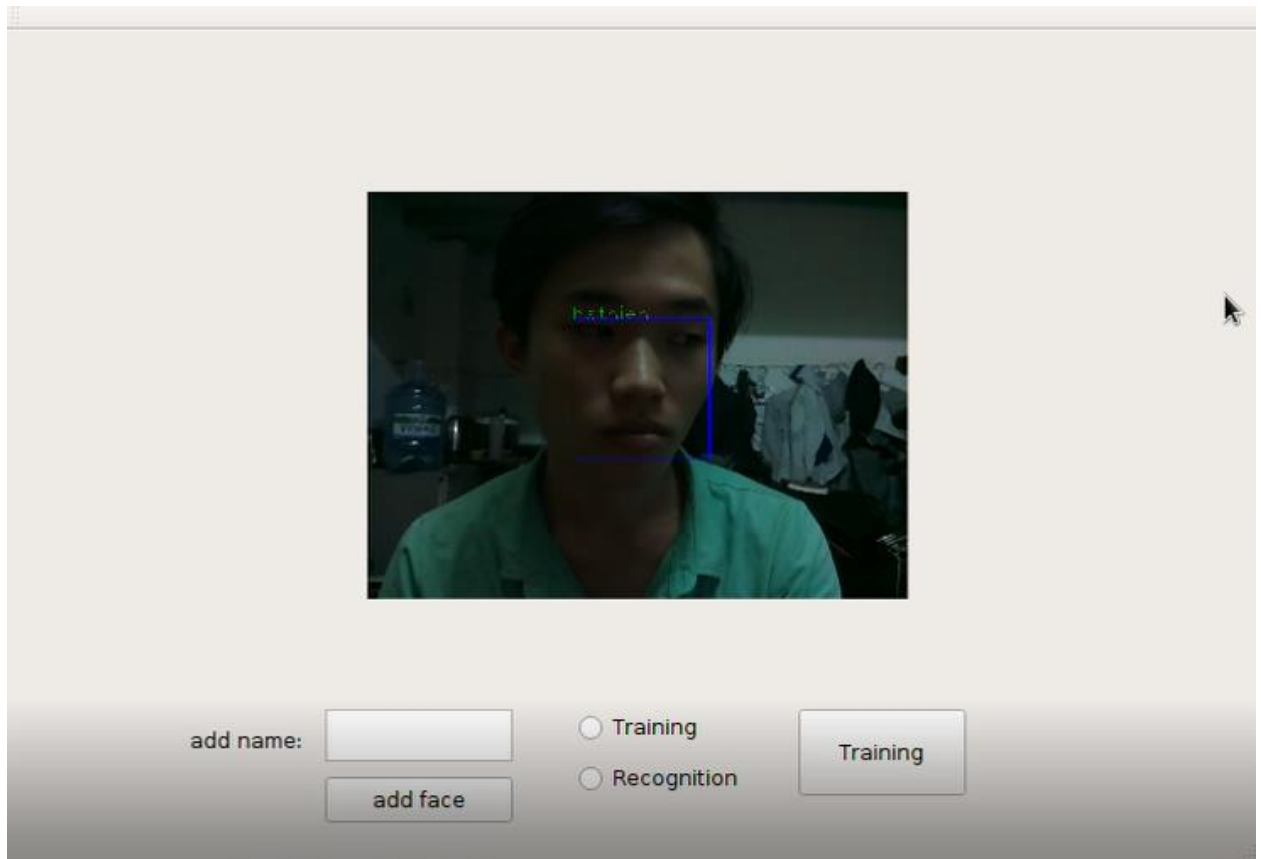


Hình 4.3. Mô hình mặt sau

Để mô hình mang tính thực tiễn hơn, mẫu thiết kế mô hình cánh cửa này bao gồm hai mặt. mỗi mặt có một nhiệm vụ khác nhau, mặt trước vừa bảo vệ vừa là nơi đặt các cảm biến, camera thu nhận hình ảnh, vân tay truyền về bộ xử lý trung tâm được tích hợp đặt ở mặt sau của cánh cửa.

Khi cấp nguồn thì raspberry sẽ tự động kết nối với wifi đã cài đặt từ trước, lập tức bật camera rồi cảm biến vân tay để thu thập thông tin của người muốn vào.

4.3. Giao diện hiển thị



Hình 4.4. giao diện hiển thị

CHƯƠNG 5. KẾT LUẬN VÀ ĐỊNH HƯỚNG ĐỀ TÀI

5.1. Kết quả đạt được

Thực hiện được các mục tiêu đề ra, mặc dù còn nhiều thiếu sót nhưng nhờ sự hướng dẫn tận tình của thầy hướng dẫn, đề tài đồ án “**Khóa điện bảo mật 2 lớp bằng vân tay và nhận diện khuôn mặt**” đã đạt được kết quả tốt.

Áp dụng thành công trên cơ sở lý thuyết của xử lý ảnh, đưa ra được hệ thống có tính chất bảo mật cao, an toàn cho người sử dụng.

Sử dụng các bộ thư viện khá tốt, mặc dù chưa sử dụng hết được các tính năng ưu việt mà thư viện hỗ trợ.

5.2. Hạn chế

Để nhận diện chính xác hơn, ta cần bộ mẫu ảnh tốt nhất, càng nhiều càng nhận diện chính xác hơn. Bộ mẫu ảnh tốt nhất là bộ mẫu ảnh có chân dung khuôn mặt đối tượng mà ở đó gồm nhiều bức ảnh có độ sáng và góc độ khác nhau.

Quá trình xử lý ảnh còn một vấn đề là khi xử lý, để nhận diện được một khuôn mặt thì chỉ cần có mẫu ảnh, nhưng để cắt ra được khuôn mặt chính xác nhất thì vẫn chưa làm được, tỉ lệ cắt được khuôn mặt khá thấp vì thế cần nâng cao khả năng này để có kết quả tốt nhất.

5.3. Hướng phát triển

Đề tài này hướng phát triển khá rõ ràng, nâng cao khả năng nhận diện sẽ đảm bảo được được độ an toàn cao nhất có thể. Nâng cao trang thiết bị, kết hợp thẻ từ và phím số sẽ được bốn lớp bảo mật. Có thể nói là “khó” có thể vượt qua được hệ thống bảo mật cao như vậy. Ngoài ra, nên thay đổi bộ xử lý trung tâm vì raspberry chưa đáp ứng mạnh mẽ được tính thời gian thực so với những bộ xử lý mạnh hơn.

PHỤ LỤC

Code arduino điều khiển cảm biến vân tay

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
void setup()
{
  Serial.begin(9600);
  while (!Serial); // For Yun/Leo/Micro/Zero/...
  delay(100);
  Serial.println("\n\nAdafruit finger detect test");
  finger.begin(57600);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }
  finger.getTemplateCount();
  Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println("
templates");
  Serial.println("Waiting for valid finger...");
}
void loop()          // run over and over again
{
  getFingerprintIDez();
  delay(50);          //don't need to run this at full speed.
}

uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("No finger detected");
      return p;
    case FINGERPRINT_PACKETRECEIVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      return p;
```

```
default:
    Serial.println("Unknown error");
    return p;
}
// OK success!
p = finger.image2Tz();
switch (p) {
    case FINGERPRINT_OK:
        Serial.println("Image converted");
        break;
    case FINGERPRINT_IMAGEMESS:
        Serial.println("Image too messy");
        return p;
    case FINGERPRINT_PACKETRECEIVEERR:
        Serial.println("Communication error");
        return p;
    case FINGERPRINT_FEATUREFAIL:
        Serial.println("Could not find fingerprint features");
        return p;
    case FINGERPRINT_INVALIDIMAGE:
        Serial.println("Could not find fingerprint features");
        return p;
    default:
        Serial.println("Unknown error");
        return p;
}
// OK converted!
p = finger.fingerFastSearch();
if (p == FINGERPRINT_OK) {
    Serial.println("Found a print match!");
} else if (p == FINGERPRINT_PACKETRECEIVEERR) {
    Serial.println("Communication error");
    return p;
} else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match");
    return p;
} else {
    Serial.println("Unknown error");
    return p;
}

// found a match!
Serial.print("Found ID #"); Serial.print(finger.fingerID);
Serial.print(" with confidence of "); Serial.println(finger.confidence);
```

```

    return finger.fingerID;
}
// returns -1 if failed, otherwise returns ID #
int getFingerprintIDez() {
    uint8_t p = finger.getImage();
    if (p != FINGERPRINT_OK) return -1;
    p = finger.image2Tz();
    if (p != FINGERPRINT_OK) return -1;
    p = finger.fingerFastSearch();
    if (p != FINGERPRINT_OK) return -1;
    // found a match!
    Serial.print("Found ID #"); Serial.print(finger.fingerID);
    Serial.print(" with confidence of "); Serial.println(finger.confidence);
    return finger.fingerID;
}

```

Code hàm main

```

#include "mainwindow.h"
#include <stdio.h>
#include <stdlib.h>
#include <QApplication>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/face.hpp>
#include <opencv2/videoio.hpp>
#include <opencv2/objdetect.hpp>
#include <opencv2/imgproc.hpp>
#include <iostream>
#include <fstream>
#include <sstream>
#include <QThread>
#include <QDebug>
#include <string>
#include <QDir>
#include <QString>
#include "process.h"
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    Process *processRecog = new Process();
    MainWindow w;
    QObject::connect(processRecog,SIGNAL(signalShowImage(Mat&,vector<Rect>&)),
    &w,SLOT(setImage(Mat&,vector<Rect>&)));
    QObject::connect(&w,SIGNAL(signalSetFlagTraining(bool)),processRecog,SLOT(set
    FlagTraining(bool)));
    QObject::connect(&w,SIGNAL(signalSetTrain()),processRecog,SLOT(train()));
}

```

```
processRecog->preprocess();
w.show();
processRecog->detect();
return a.exec();
}
```

Code Giao diện

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QPixmap>
#include <QLabel>
#include <QDebug>
#include <stdio.h>
#include <string.h>
#include <QDir>
MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    qDebug() <<"Init UI";
    ui->m_label->setAlignment(Qt::AlignCenter);

    pix = pix.scaled(ui->m_label->size(),Qt::KeepAspectRatio);
}
MainWindow::~MainWindow()
{
    delete ui;
}
void MainWindow::setImage(Mat& img,vector<Rect>& rects)
{
    // qDebug()<<"setImage";using namespace cv;
    frame = img;
    m_rects.clear();
    m_rects = rects;
    cv::resize(img, img, Size(320,240));
    ui->m_label->setPixmap(QPixmap::fromImage(QImage((unsigned char*) img.data,
img.cols, img.rows, QImage::Format_RGB888).rgbSwapped())));
}
void MainWindow::on_addface_clicked()
{
    if(m_rects.size() == 1){
        QDir dir;
```

```

    qDebug()<<"add image";
    QString str = ui->textEdit->toPlainText();
    qDebug()<<"str" <<str;
    QString cmd = QString("mkdir
%1/%2").arg("/home/pi/doantotnghiep/demowebcam/dataset").arg(str);
    qDebug()<<"cmd " <<cmd;
    system(cmd.toLatin1().data());

    QString path =
    QString("%1/%2").arg("/home/pi/doantotnghiep/demowebcam/dataset").arg(str);
    dir.setPath(path);
    QStringList images = dir.entryList(QStringList() << "*.jpg" <<
"*.JPG",QDir::Files);
    int num = images.count();
    qDebug()<<"num"<<num << dir.count();
    QString filename =
    QString("%1/%2/%3.jpg").arg("/home/pi/doantotnghiep/demowebcam/dataset").arg(st
r).arg(num);
    qDebug()<<"filename"<<filename;
    imwrite(filename.toLatin1().data(),frame(m_rects.at(0)));
}
}
void MainWindow::on_radioButton_toggled(bool checked)
{
    if(checked)
    {
        signalSetFlagTraining(true);
    }
}
void MainWindow::on_radioButton_2_toggled(bool checked)
{
    if(checked)
    {
        signalSetFlagTraining(false);
    }
}
void MainWindow::on_train_clicked()
{
    signalSetTrain();
}

```

Code các hàm xử lý ảnh:

```

#include "process.h"
#define RASPICAM
Process::Process(QObject *parent) : QObject(parent)
{

```



```

    this->isTraining = false;//mode : recognition
}

void Process::read_file(const QString& filename, vector<Mat>& images,
vector<int>& labels,QMap<int,QString>& mapLabel){
    QDir dir;
    dir.setPath(filename);
    QStringList list = dir.entryList(QDir::AllDirs);
    qDebug() << list;
    //filter incorrect folder
    for(int i = list.size()-1; i >=0 ; --i){
        if(list.at(i).contains("."))
        {
            list.removeAt(i);
        }
    }
    qDebug() << list;
    for (int j = 0 ; j<=list.size()-1;j++){
        if(this->isTraining==true){
            QString fdname = QString("%1/%2").arg(filename).arg(list.at(j));
            dir.setPath(fdname);
            QStringList listfile = dir.entryList(QStringList()<<
            "*.jpg"<< "*.JPG"<< "*.pgm",QDir::Files);
            for (int k = 0 ; k<=listfile.size()-1;k++){
                QString imgname = QString("%1/%2").arg(fdname).arg(listfile.at(k));
                Mat img = imread(imgname.toLatin1().data(), IMREAD_GRAYSCALE);
                if(img.data != NULL){
                    resize(img, img, Size(168,192), 0, 0, INTER_CUBIC);
                    images.push_back(img);
                    labels.push_back(j);
                }
            }
            mapLabel.insert(j,list.at(j));
        }
    }
}

void Process::preprocess()
{
    vector<Mat> images;
    vector<int> labels;
    QString path = "/home/pi/doantotnghiep/demowebcam/dataset";
    read_file(path,images,labels,mapLabel);
    qDebug() << images.size();
    qDebug() << labels.size();
    qDebug() << mapLabel.keys();
}

```

```
qDebug() << mapLabel.values();
if(this->isTraining == true)
{
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work. Please add
more images to your data set!";
        CV_Error(Error::StsError, error_message);
    }
    printf("Begin training \n");
    qDebug() << "Begin training";
    model->train(images, labels);
    model->write("/home/pi/doantotnghiep/demowebcam/model/model");
    printf("train finished \n");
    qDebug() << "train finished";
} else if(isTraining == false)
{
    printf("Begin load model \n");
    qDebug() << "Begin load model";
    model->read("/home/pi/doantotnghiep/demowebcam/model/model");
    printf("Load model finished \n");
    qDebug() << "Load model finished";
}
}
```

```
void Process::train()
{
    qDebug()<<"SetTrain"<< isTraining;
    this->isTraining = true;
    this->preprocess();
}
```

```
void Process::setFlagTraining(bool flag)
{
    qDebug()<<"SetFlag" << flag;
    this->isTraining = flag;
}
```

```
QString Process::testing(cv::Mat img)
{
    int predictedLabel = -1;
    double confidence = 0.0;
    cv::cvtColor(img, img, CV_BGR2GRAY);

    resize(img, img, Size(168,192), 0, 0, INTER_CUBIC);
    model->predict(img, predictedLabel, confidence);
    //printf(" confidence = %.10f ",confidence);
}
```

```
    qDebug() << "Predicted Label" << mapLabel.value(predictedLabel);
    return mapLabel.value(predictedLabel);
}
void Process::detect()
{
    Mat img;
#ifdef RASPICAM
    raspicam::RaspiCam_Cv Camera;

    //set camera params
    Camera.set( CV_CAP_PROP_FORMAT, CV_8UC3 );
    //Open camera
    cout<<"Opening Camera..."<<endl;
    if (!Camera.open()) {
        cerr<<"Error opening the camera"<<endl;
    }
#elif
    VideoCapture cap(0);
    cap.open(0);
#endif

    // Camera.release();

    CascadeClassifier face_cascade;
    face_cascade.load(
        "/usr/local/share/OpenCV/haarcascades/haarcascade_frontalcatface.xml" );

    for (;;)
    {
#ifdef RASPICAM
        Camera.grab();
        Camera.retrieve ( img);
#elif
        // Image from camera to Mat
        cap >> img;
        // obtain input image from source
        cap.retrieve(img, CV_CAP_OPENNI_BGR_IMAGE);
#endif
        // Container of faces
        vector<Rect> faces;
        // Detect faces
        face_cascade.detectMultiScale( img, faces, 1.1, 2,
0|CV_HAAR_SCALE_IMAGE, Size(168, 192) );
        QString label = "";
        if(this->isTraining==false)//mode recog
        {
```

```
        if(faces.size()==1){
            label = testing(img(faces.at(0)));
            qDebug() << label;
        }
    }
    // To draw rectangles around detected faces
    for (unsigned i = 0; i<faces.size(); i++){
        rectangle(img,faces[i], Scalar(255, 0, 0), 2, 1);
        if(this->isTraining==false)//mode recog
        {
            cv::putText(img,label.toLatin1().data(),Point(faces[i].x,faces[i].y),FONT_HERSHEY_
            PLAIN, 3, Scalar(0,255,0));
        }
    }
    signalShowImage(img,faces);
    //    imshow("HaDuyThien-2002150158", img);
    int key2 = waitKey(20);
    if(key2 == 13) // enter out
        break;
    }
}
```

TÀI LIỆU THAM KHẢO

[1]https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#id27

[2]<https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library>

[3]<https://www.uco.es/investiga/grupos/ava/node/40>

[4]Auranuch Lorsakul and Jackrit Suthakorn. “*Traffic Sign Recognition Using Neural Network on OpenCV: Toward Intelligent Vehicle/Driver Assistance System*”. Center for Biomedical and Robotics Technology (BART LAB), Department of Biomedical Engineering, Faculty of Engineering, Mahidol University, 25/25 Putthamonthon 4 Road, Salaya, Putthamonthon, Nakornpathom, 73170, Thailand.

[5]H. Fleyeh E. Davami. “*Eigen-based traffic sign recognition*”. Computer Science Department, School for Technology and Business Studies, Dalarna University, Rodavagen 3, 78188 Borlange, Sweden.