BỘ GIÁO DỤC & ĐÀO TẠO
ĐH BÁCH KHOA HÀ NỘI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do - Hạnh phúc

ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

NHẬN DIỆN KHỚP NGÓN TAY NGƯỜI DỰA TRÊN HÌNH ẢNH SỬ DỤNG RASPBERRY PI

NGUYỄN ĐỨC HUÂN
huan.nd200252@sis.hust.edu.vn
Ngành Kỹ thuật điều khiển và Tự động hóa

Giảng viên hướng dẫn: PGS. TS. Trần Thị Thảo

KHOA: Tự động hóa

TRƯỜNG: Điện – Điện tử

HÀ NỘI, 2/2024

BỘ GIÁO DỤC & ĐÀO TẠO
ĐH BÁCH KHOA HÀ NỘI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do - Hạnh phúc

ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

NHẬN DIỆN KHỚP NGÓN TAY NGƯỜI DỰA TRÊN HÌNH ẢNH SỬ DỤNG RASPBERRY PI

NGUYỄN ĐỨC HUÂN
huan.nd200252@sis.hust.edu.vn
Ngành Kỹ thuật điều khiển và Tự động hóa

Giảng viên hướng dẫn: PGS. TS. Trần Thị Thảo

Chữ ký của GVHD

KHOA: Tự động hóa

TRƯỜNG: Điện – Điện tử

HÀ NỘI, 2/2024

BỘ GIÁO DỤC & ĐÀO TẠO
ĐH BÁCH KHOA HÀ NỘI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do - Hạnh phúc

NHIỆM VỤ
ĐỒ ÁN TỐT NGHIỆP

Họ và tên sinh viên: **Nguyễn Đức Huân**
Khóa: **65**
Trường: **Điện- Điện tử**
Ngành: **KT ĐK &TĐH**

1.Tên đề tài:

Nhận diện khớp ngón tay người dựa trên hình ảnh sử dụng Raspberry Pi.

2.Nội dung đề tài:

- ❖ Giao diện hệ thống:
  - Hiển thị màn hình hệ thống.
  - Tương tác với người dùng qua màn hình, nút bấm và còi.
- ❖ Chức năng điều khiển, vận hành:
  - Nhận diện khớp ngón tay người.
  - Thử nghiệm và đánh giá sản phẩm

3.Thời gian giao đề tài:.....

4.Thời gian hoàn thành:……

Ngày..... tháng ..... năm 2024

CÁN BỘ HƯỚNG DẪN

LỜI CẢM ƠN

Trong thời gian làm đồ án tốt nghiệp, em đã nhận được sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè. Em xin gửi lời cảm ơn chân thành đến cô giáo PGS.TS. Trần Thị Thảo, giảng viên – Đại học Bách Khoa Hà Nội đã tận tình hướng dẫn, chỉ bảo em trong quá trình làm đồ án.

Cảm ơn Bách Khoa, nơi đã góp phần tạo nên những kỷ niệm đáng nhớ và là nền tảng vững chắc cho tương lai của tôi. Tuổi trẻ đi qua như một cơn mưa rào, nhưng cơn mưa đó đã thấm vào tôi những tri thức, trải nghiệm quý giá, ngấm vào tôi tình yêu, tình bạn, tình thầy/cô và học trò. . . Hãy giúp tôi lưu giữ những kỷ niệm thanh xuân này. Cảm ơn thật nhiều – Đại học Bách Khoa Hà Nội thân yêu!

TÓM TẮT NỘI DUNG ĐỒ ÁN

Cùng với sự phát triển như vũ bão về công nghệ của cuộc cách mạng 4.0 đã và đang đưa nhiều quốc gia trở thành siêu cường quốc về khoa học công nghệ. Nhưng theo cùng với đó, nhiều hệ lụy nguy hiểm cũng tiềm tàng trong từng lĩnh vực, đặc biệt là vấn đề bảo mật. Những công nghệ sinh trắc học như nhận diện khuôn mặt, nhận diện vân tay… đang rất được ưa chuộng tại các quốc gia phát triển. So với các phương thức sinh trắc học truyền thống khác nhận dạng khớp ngón tay con người có ưu điểm nổi bật là chi phí thấp, bảo vệ quyền riêng tư và thân thiện với người dùng. Hiện nay, với sự phát triển mạnh mẽ của ngành khoa học máy tính, xử lý ảnh tạo ra môi trường thuận lợi cho bài toán nhận diện khớp ngón tay từ ảnh số. Mục tiêu của đồ án ày là em sẽ thiết kế một hệ thống máy tính nhúng Raspberry 3 Model B+ cùng với PiCamera V2 để nhận diện khớp tay. Em có đưa ra bốn phương pháp để nhận diện và bốn phương pháp đều mang lại kết quả nhận diện khá tốt, trong đó phương pháp nhị phân hóa sơ đồ mã hóa định hướng có tốc độ nhận diện nhanh nhất. Bên cạnh đó phương pháp mã hóa sơ đồ định hướng và độ rộng đường vân cho kết quả nhận diện tốt nhất. Em chọn phương pháp nhị phân hóa sơ đồ mã hóa định hướng để triển khai với Raspberry Pi – tối ưu nhất với phần cứng khá yếu của Raspberry Pi.

Em sẽ tìm hiểu thêm các thuận toán như học máy, học sâu để tăng độ chính xác cho hệ thống, cũng như sử dụng camera độ phân giải tốt để tăng độ nét cho ảnh chụp, sử dụng các máy tính nhúng đời mới hơn để tăng tốc độ tính toán.

Qua đề tài này em học được các kĩ năng như giải quyết vấn đề, kỹ năng lập trình, đặt vấn đề, kiểm soát thời gian và tiến độ của đề tài.

Sinh viên thực hiện
Ký và ghi rõ họ tên

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT, KÝ HIỆU

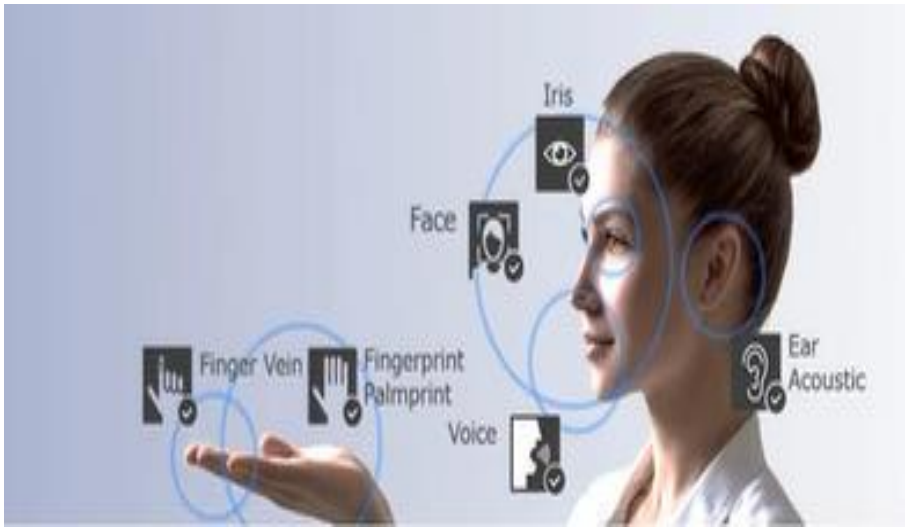
FKP	Finger Knuckle Print	Khớp ngón tay con người
ROI	Region Of Interest	Vùng quan tâm
Pi	Raspberry Pi 3 Model B+	Máy tính nhúng Raspberry Pi 3
BOCV	Binary Orientation Cooccurrence Vector	Vector đồng bộ định hướng nhị phân
LRT	Localized Radon Transform	Biến đổi Radon cục bộ
ImcompCode		Sơ đồ mã hóa định hướng
magCode		Sơ đồ mã hóa độ lớn

CHƯƠNG 1. TỔNG QUAN VỀ BÀI TOÁN NHẬN DẠNG KHỚP NGÓN TAY CON NGƯỜI

1.1 Sự phát triển của các phương pháp xác thực bằng sinh trắc học

Sinh trắc học là lĩnh vực định danh của một người dựa trên những đặc điểm về thể chất hoặc hành vi của người đó. Ngày nay, cùng với sự phát triển vượt bậc của khoa học công nghệ, các phương pháp định danh bằng sinh trắc học được áp dụng ngày càng rộng rãi.

Các phương pháp nhận diện hoặc xác thực bằng các yếu tố y sinh được sử dụng trong việc xác thực một người dựa trên những đặc điểm giải phẫu hoặc hành vi như dấu vân tay, vân bàn tay, khuôn mặt, mống mắt, dấu chân, khớp ngón tay, v.v như minh họa trên Hình 1.1. Bên cạnh nhiều phương pháp vẫn đang trong quá trình nghiên cứu và hoàn thiện thì cũng có rất nhiều phương pháp đã được áp dụng rộng rãi trong thực tế và mang lại nhiều hiệu quả lớn.



Hình 1.1 Các đặc điểm sinh lý của con người được sử dụng trong sinh trắc học

Với sự quan tâm ngày càng tăng về các lỗ hổng bảo mật và gian lận giao dịch trong nhiều ngành của xã hội, các kỹ thuật xác thực và nhận dạng cá nhân có độ tin cậy cao và dễ tiếp cận trở thành nhu cầu tất yếu của con người. Một số lĩnh vực công nghệ sinh trắc học được áp dụng phổ biến có thể kể đến như trên Hình 1.2. Vì vậy, sự kết hợp giữa lý thuyết sinh trắc học và hệ thống hạ tầng công nghệ thông tin đã trở thành giải pháp tối ưu để xây dựng và cải tiến các hệ thống bảo mật với mục tiêu thông minh, an toàn và tiện lợi hơn.

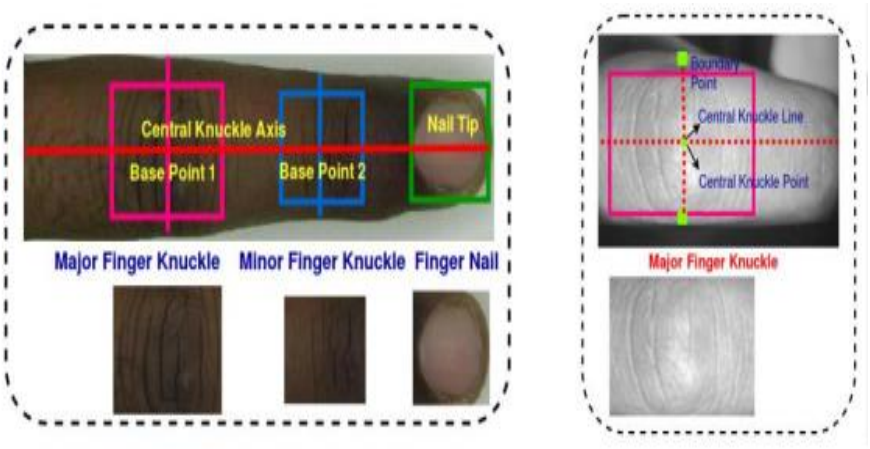
Hình 1.2 Một số lĩnh vực công nghệ sinh trắc học được áp dụng phổ biến

1.2 Bài toán nhận diện khớp ngón tay

Trong những thập kỉ gần đây, sinh trắc học đã nổi lên như một công nghệ đáng tin cậy để cung cấp mức độ bảo mật cao hơn cho hệ thống xác thực cá nhân (như mã pin, mật khẩu,…) Trong số các đặc điểm sinh trắc học có thể được sử dụng để nhận biết một con người phân biệt với những người khác, bàn tay con người là hình thức công nghệ sinh trắc học được ra đời từ lâu, phát triển và được áp dụng rộng rãi nhất và có lẽ là thành công nhất. Các cấu trúc có thể được trích xuất từ bàn tay bao gồm hình học bàn tay, dấu vân tay, lòng bàn tay, khớp ngón tay. Những tính chất về tay là ổn định, không thay đổi theo thời gian và đáng tin cậy. Khi một người tới tuổi trưởng thành, cấu trúc khớp ngón tay vẫn tương đối ít thay đổi trong suốt cuộc đời của họ. Ngoài ra, công nghệ quét các cấu trúc của tay thường được coi là không nguy hiểm và tiện lợi hơn so với các hệ thống quét toàn bộ khuôn mặt hay mống mắt. Người dùng không cần phải nhận thức về cách họ tương tác với hệ thống. Đặc biệt, ưu điểm của nhận diện khớp ngón tay là nó có tính bảo vệ quyền riêng tư con người cao hơn so với khuôn mặt, những lợi thế này đã tạo điều kiện thuận lợi cho việc triển khai các đặc tính của tay trong các ứng dụng sinh trắc học.

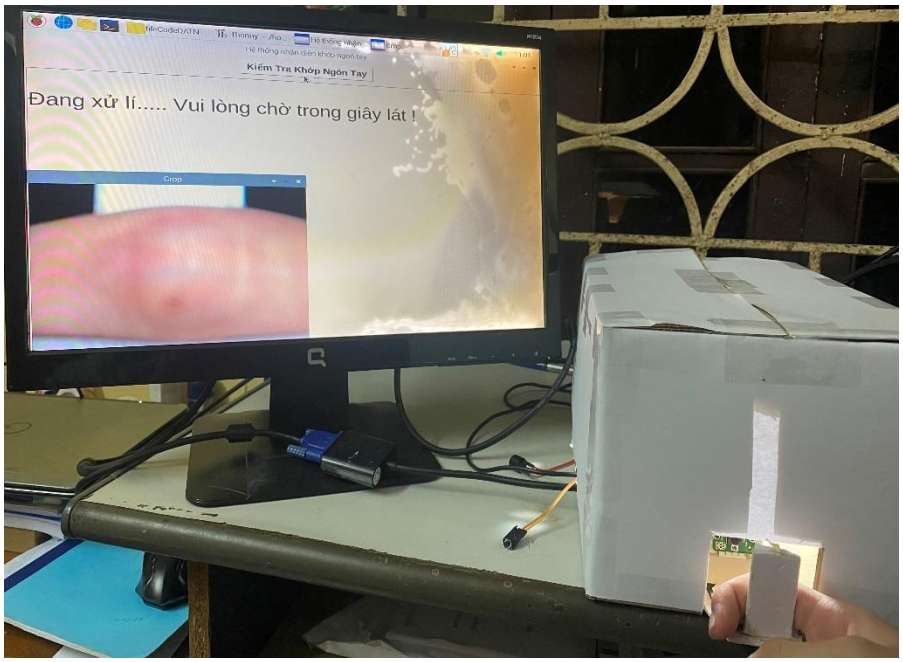
Hiện nay, hầu hết các thiết bị thu nhận cấu trúc hình học của tay đều dựa trên thiết kế dựa trên cảm ứng, tức là người dùng được yêu cầu chạm vào thiết bị hoặc giữ một số chốt hướng dẫn hoặc các thiết bị ngoại vi bên ngoài để lấy hình ảnh bàn tay của họ được dễ dàng và tiện lợi.

Khi hình ảnh ngón tay của người dùng được chụp thông qua các thiết bị. Các vùng quan tâm (ROI) của ngón tay sẽ được trích xuất (ví dụ khớp chính, khớp phụ ngón tay, móng tay,…) như minh họa ở Hình 1.3.



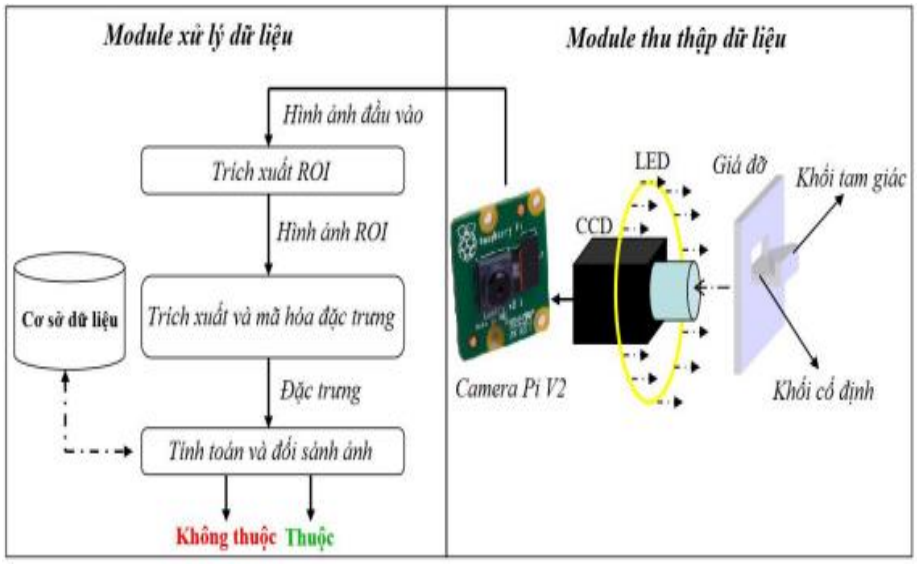
Hình 1.3 Các đặc điểm trên khớp ngón tay

ROI chứa thông tin quan trọng của khớp ngón tay được sử dụng để nhận dạng. Trong đề tài này em lựa chọn sử dụng phần khớp chính – Major FKG để nhận diện. ROI được xử lý trước để kết cấu của khớp ngón tay có thể phân biệt được với nền. Sau đó, các tính năng phân biệt trong ROI có thể được trích xuất bằng cách sử dụng các kỹ thuật và các thuật toán đa dạng để xác định, nhận diện khớp ngón tay đó là của ai theo từng yêu cầu cụ thể của mỗi bài toán. Một thiết bị nhận diện khớp ngón tay người trong thực tế được minh họa trên Hình 1.4..



Hình 1.4 Tổng quan thiết bị nhận diện khớp ngón tay người trong thực tế

Hệ thống nhận diện khớp ngón tay cơ bản được chia làm hai module chính như biểu diễn ở Hình 1.5.



Hình 1.5 Tổng quan hai module trong bài toán nhận diện khớp ngón tay

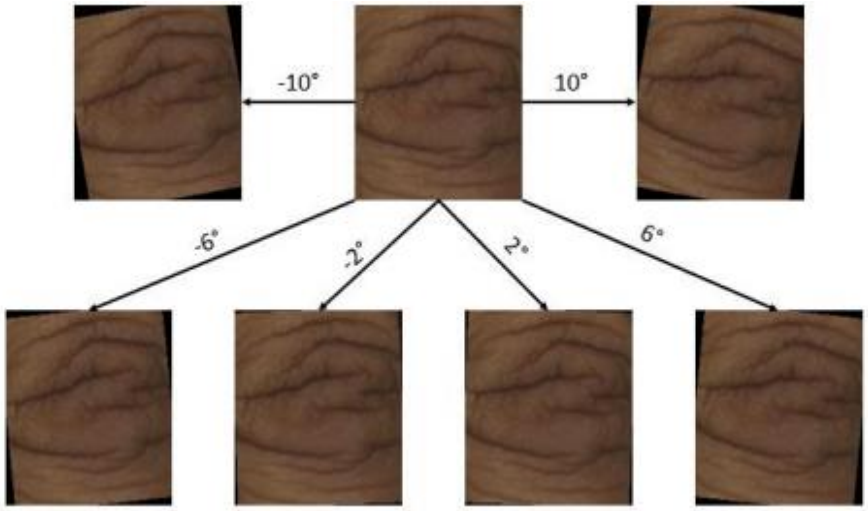
Thứ nhất là module thu thập dữ liệu phụ trách việc thêm dữ liệu vào cơ sở dữ liệu và lấy dữ liệu để đối sánh với cơ sở dữ liệu có sẵn. Module thứ hai phụ trách việc xử lý dữ liệu đầu vào và so sánh với cơ sở dữ liệu để đưa ra kết quả người đó có thuộc cơ sở dữ liệu có sẵn hay là không.

1.3 Những khó khăn và thách thức trong bài toán nhận diện khớp ngón tay

Bài toán nhận diện khớp ngón tay người cũng không phải là vấn đề quá mới, tuy nhiên đây là một bài toán khó nên những nghiên cứu gần đây vẫn chưa đạt được kết quả cao như mong muốn. Có thể kế đến những khó khăn của bài toán nhận diện khớp ngón tay người như sau:

- Điều kiện của ảnh: Ảnh được chụp trong các điều kiện khác nhau về chiếu sáng ví dụ như ở Hình 1.6., cũng như sử dụng camera khác nhau (máy kỹ thuật số, máy ảnh hồng ngoại…) ảnh hưởng rất nhiều đến chất lượng hình ảnh chụp khớp ngón tay.
- Góc chụp: Ảnh chụp khớp ngón tay có thể thay đổi rất nhiều bởi góc chụp như ở Hình 1.7. Chẳng hạn như: chụp thẳng, chụp chéo bên trái, bên phải ở các góc khác nhau…

Hình 1.6 Ảnh hưởng của độ sáng đến việc nhận dạng khớp ngón tay



Hình 1.7 Ảnh hưởng của góc quay đến việc nhận dạng khớp ngón tay

1.4 Các ứng dụng của bài toán nhận diện khớp ngón tay

Tuy chưa được áp dụng rộng rãi trong thực tế nhiều như so với như vân tay, bài toán nhận diện khớp ngón tay cũng có nhiều tiềm năng phát triển để có thể được áp dụng trong nhận dạng con người. Có thể kể đến một số ứng dụng nổi bật trong đời sống hằng ngày như hình 1.8. sau đây:

- Ứng dụng trong các máy ATM (máy rút tiền).
- Ứng dụng trong việc điều tra các tội phạm trong các vụ án để lại hình ảnh khớp ngón tay.
- Bảo mật: Ứng dụng của việc nhận dạng con người sử dụng khớp có tay trong lĩnh vực bảo mật cho phép người dùng có thể đăng nhập hoặc truy cập vào một thiết bị hoặc một ứng dụng nào đó thay cho việc nhập mật khẩu như thông thường.
- Các ứng dụng khác tương tự như các bảo mật sinh trắc học khác ví dụ: chấm công, điều khiển mở đóng cửa văn phòng, công ty, nhà ở,…

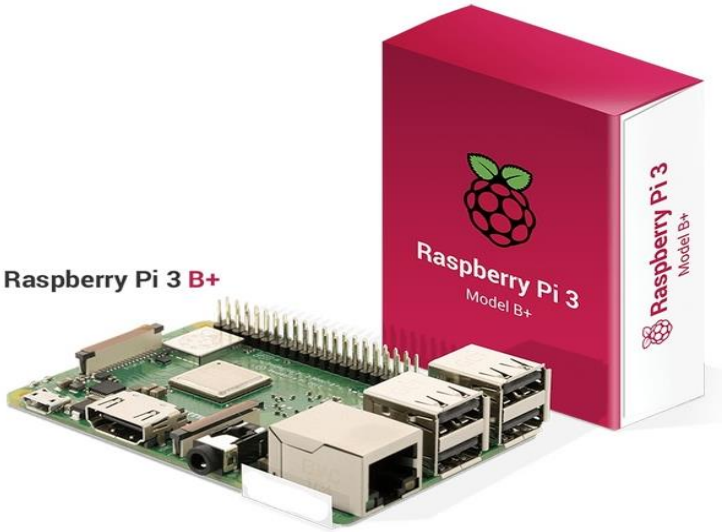
Hình 1.8 Một số hình ảnh có chứa khớp ngón tay để nhận dạng con người

1.5 Giới thiệu thiết bị phần cứng

1.5.1 Máy tính nhúng Raspberry Pi 3

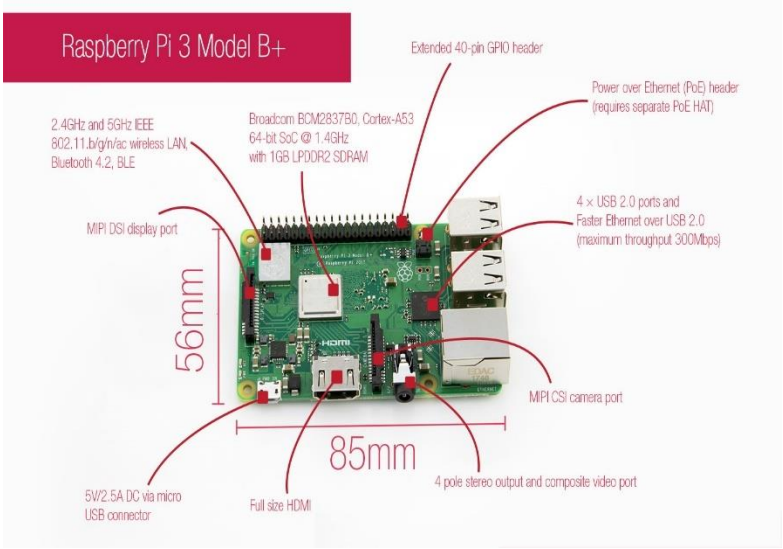
Raspberry Pi là dòng máy tính mini, giá rẻ được phát triển bởi Raspberry Foundation tại Anh. Mục đích ban đầu của dòng máy tính này là phục vụ giảng dạy và học tập các môn trong lĩnh vực máy tính trong các nhà trường, nhưng với ưu điểm là đa năng, dễ sử dụng và giá thành rất rẻ nên Raspberry ngày càng tiếp cận với nhiều đối tượng người dùng.

Raspberry pi không phải là một dòng máy tính mạnh và không thể thay thế hoàn toàn các máy tính để bàn hoặc laptop. Tuy nhiên, chúng có thể chạy trên hệ điều hành Raspbian, Linux,… với các tiện ích như lướt web, tương tác màn hình, và các ứng dụng khác. Bên cạnh đó, Raspberry pi còn là một thiết bị tuyệt vời cho các ứng dụng điện tử, các dự án DIY, hoặc trở thành môi trường thực hành cho các môn học lập trình.



Hình 1.9 Máy tính Raspberry Pi 3 Model B+

Trong đồ án này, em sử dụng máy tính Raspberry pi 3 Model+ B như Hình 1.10.



Hình 1.10 Phần cứng máy tính Raspberry Pi 3 B+

1.5.2 Raspberry Pi Camera Module V2

Để thực hiện chụp ảnh khớp ngón tay, em đã lựa chọn Camera Raspberry Pi V2. Chiếc camera này được Raspberry Pi Foundation giới thiệu lần đầu vào tháng 4/2016 với nâng cấp đáng kể nhất là sử dụng sensor Sony IMX219 8 Megapixel. Raspberry Pi Camera Module V2 có một cảm biến 8-megapixel của Sony IMX219 (so với cảm biến 5-megapixel OmniVision OV5647 trên Camera Module phiên bản cũ).

Camera Module có thể được sử dụng để quay video độ nét cao, cũng như chụp hình ảnh tĩnh. Nó khá dễ dàng để sử dụng cho người mới bắt đầu, nhưng cũng có rất nhiều giải pháp mở rộng để cung cấp cho người dùng yêu cầu cao. Có rất

nhiều demo của người dùng về công dụng của Camera Module như chụp TimeLapse, SlowMotion và rất nhiều ứng dụng khác.

Raspberry Pi Camera Module V2, như trên Hình 1.11, có thể coi như một sự đột phá về chất lượng hình ảnh, màu sắc trung thực và hiệu suất ánh sáng thấp. Đặc biệt nó hỗ trợ video lên tới 1080P30, 720P60 và video mode VGA90, cũng như chế độ chụp hình. Dĩ nhiên, nó vẫn sử dụng đoạn cáp 15cm qua cổng CSI trên Raspberry Pi.

Hình 1.11 Raspberry Pi Camera V2

Chiếc camera này tương thích với tất cả các phiên bản của Raspberry Pi. Thông số kỹ thuật:

- Ống kính tiêu cự cố định.
- Cảm biến độ phân giải 8 megapixel cho khả năng chụp ảnh kích thước 3280 x 2464.
- Hỗ trợ video 1080p30, 720p60 và 640x480p90.
- Kích thước 25mm x 23mm x 9mm.
- Kết nối với Raspberry Pi thông qua cáp ribbon đi kèm dài 15 cm
- Camera Module được hỗ trợ với phiên bản mới nhất của Raspbian.

CHƯƠNG 2. PHƯƠNG PHÁP XÁC ĐỊNH VÙNG QUAN TÂM (ROI)

2.1 Cơ sở lý thuyết xác định vùng quan tâm

Vùng quan tâm ROI (Region of interest) thường được định nghĩa là phần có ý nghĩa và quan trọng của một bức ảnh, là vùng mà ta sẽ áp dụng các xử lý trên đó và bỏ qua các vùng còn lại. Thông thường, trong các bài toán xử lý ảnh, dữ liệu cần xử lý thường chỉ là một phần hoặc một vùng của bức ảnh, vì vậy việc xử lý toàn bộ khung hình là không tối ưu. Xác định và sử dụng ROI có thể tránh xử lý những vùng ảnh không cần thiết nhằm tăng tốc độ tính toán.

Vùng quan tâm ROI trong xử lý ảnh có các đặc điểm sau:

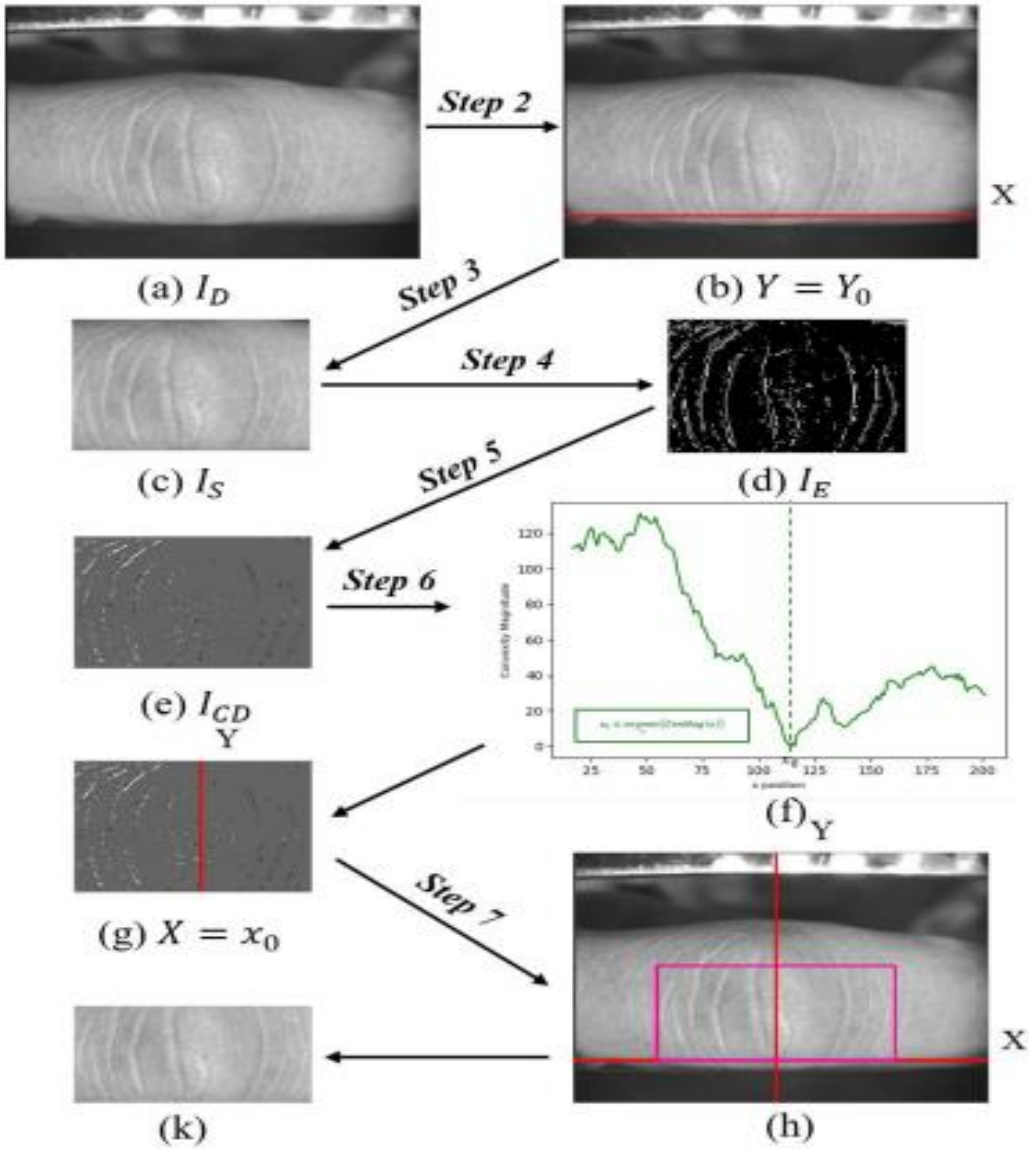
- ROI có thể ở bất cứ hình dạng nào, có thể là hình vuông, tròn, nhiều vùng nhỏ trên ảnh hoặc thậm chí là các pixel.
- ROI là một mặt nạ mà trên đó pixel quan tâm thường có giá trị bằng 1 và pixel không quan tâm có giá trị bằng 0.
- Vùng không quan tâm trên ảnh thường vùng nền (background)
- Để tìm được vùng quan tâm ROI trên bức ảnh, ta lấy ảnh gốc nhân với mặt nạ ROI.

Trên thực tế, hình ảnh FKP được trích xuất từ những ngón tay của những người khác nhau là rất khác nhau. Mặt khác, đối với những hình ảnh FKP được trích xuất từ cùng một ngón tay cũng rất khác nhau vì sự thay đổi của vị trí tay hoặc ánh sáng tại thời điểm lấy hình ảnh. Vì lý do đó,việc xây dựng một hệ trục tọa độ cho từng hình ảnh là cần thiết và vô cùng quan trọng. Như vậy, dựa vào hệ trục tọa độ đã được xây dựng riêng cho từng hình ảnh như vậy, phần ROI được cắt trực tiếp từ ảnh gốc sẽ chính xác và có thể dùng để khai thác cho các ứng dụng nhận diện sau này. Trong phần này, em sẽ trình bày một thuật toán để xác định hệ thống tọa độ cục bộ và trích xuất ảnh ROI từ ảnh gốc đầu vào.

Từ các hình ảnh thu thập được, em nhận thấy vì ngón tay luôn được đặt trên các khối cơ bản của hệ thống khi chụp, vì vậy đường biên dưới cùng của ngón tay hay ranh giới dưới của ngón tay với nền là khá ổn định trong mọi hình ảnh và có thể lấy làm trục X của hệ tọa độ ROI. Tuy nhiên việc xác định trục Y của ROI phức tạp hơn trục X rất nhiều. Về mặt trực quan ban đầu, em muốn xác định vị trí của trục Y trùng với vị trí trung tâm của khớp chính ngón tay, vị trí này chứa hầu hết các đặc điểm đặc trưng nhất của khớp ngón tay, từ đó có thể xác định được hình ảnh ROI mang độ tin cậy cao nhất. Chúng ta có thể dễ dàng quan sát cấu trúc của khớp ngón tay, nếu lấy trục Y là trục nằm chính giữa khớp ngón tay, thì hai bên trục Y, các đường vân sẽ có xu hướng cong về hai hướng đối lập. Dựa vào quan sát thực tế này, em sẽ mã hóa các pixel đường dựa trên xu hướng cong của

các đường vân hai bên đối lập, cuối cùng sử dụng các mã hóa hướng cong này để xác định được trục Y của hình ảnh.

Hình 2.1 minh họa các bước chính để xác định trục tọa độ X,Y và trích xuất hình ảnh ROI. Các bước chính được tóm tắt như sau:



Hình 2.1 Minh họa các bước của thuật toán trích xuất ROI

Đầu tiên ở bước (a): hình ảnh thu được bằng phương pháp downsampling sau khi làm mịn ảnh bằng Gaussian.

(b): Xác định trục X của hình ảnh là đường biên dưới cùng của ngón tay.

(c): thu được từ sau khi đã cắt bớt nền.

(d): thu được từ sau khi áp dụng phương pháp dò cạnh Canny.

(e): thu được từ sau khi áp dụng sơ đồ mã hóa hướng cong của đường vân khớp ngón tay.

(f): Tập các giá trị cường độ lồi thu được sau khi tính toán bằng hàm ConMag(x) sử dụng để xác định trục Y của hình ảnh ROI.

(g): Từ tập trên, ta xác định trục Y từ giá trị sao cho giá trị hàm ConMag(x) là nhỏ nhất

(h): Hệ tọa độ hình ảnh ROI, với diện tích hình ảnh ROI được lấy theo kinh nghiệm.

(k): Hình ảnh ROI hoàn chỉnh thu được qua các bước

2.2 Thuật toán trích xuất ROI từ hình ảnh FKP

Chi tiết các bước của thuật toán trích xuất ROI từ hình ảnh FKP được em tham khảo ở bài báo số [1] và bài báo số [2].

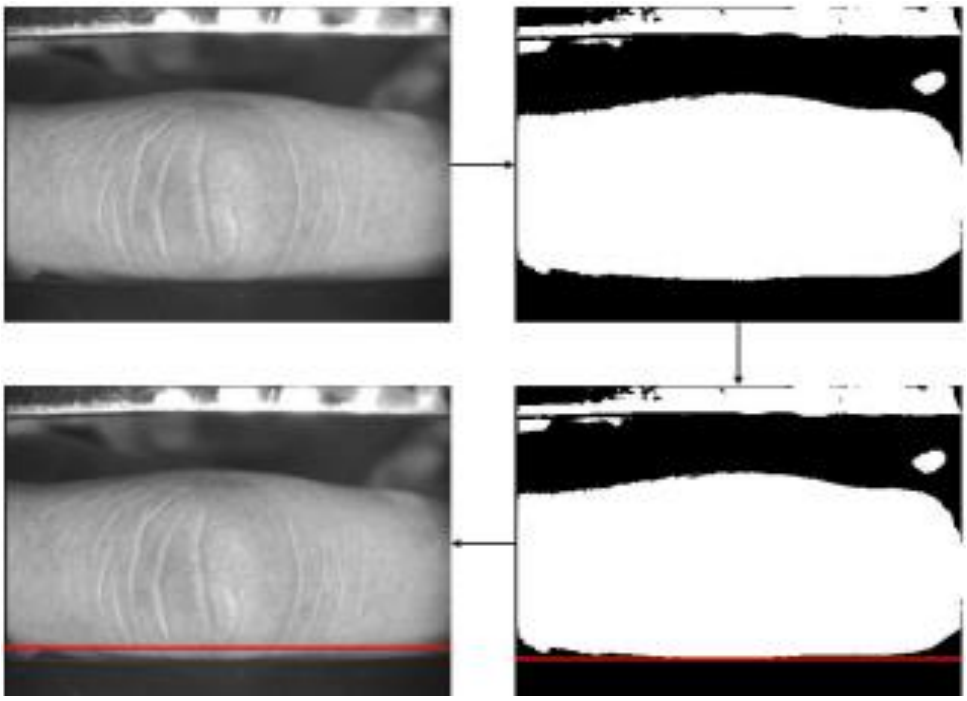
2.2.1 Tiền xử lý hình ảnh đầu vào

Hình ảnh FKG mẫu được lấy với kích thước 768 x 576 với độ phân giải 400dpi ("Dots Per Inch" – là đơn vị đo số lượng chấm trên 1inch đơn vị dài, chỉ số DPI càng lớn thì hình ảnh càng sắc nét). Nhưng dựa vào thực nghiệm, em thấy hình ảnh đầu vào không nhất thiết phải sử dụng độ phân giải quá cao để trích xuất dữ liệu và nhận diện con người. Vì vậy, sau khi áp dụng phương pháp làm mịn ảnh bằng Gaussian, em đã sử dụng thuật toán Down-Sampling để giảm độ phân giải xuống còn 150dpi.

Tác dụng của việc làm này trước hết đó là làm giảm tối thiểu lượng tài nguyên tính toán của máy tính, vì hình ảnh đầu vào mang lượng thông tin gọn hơn, nhưng vẫn giữ được hầu hết các thông tin cần thiết mà hình ảnh mang lại. Tiếp đến việc làm mịn hình ảnh bằng thuật toán Gaussian trước khi đến các bước tiếp theo đó là giảm được nhiễu đáng kể trong hình ảnh gốc, điều này có rất nhiều lợi ích cho các bước trích xuất dữ liệu và đối sánh hình ảnh để nhận diện con người. Một minh họa hình ảnh sau khi đã Down-Sampling và làm mịn bằng Gaussian được thể hiện như ở Hình 2.1 (a).

2.2.2 Xác định trục X của hệ tọa độ

Nhận thấy đường biên dưới cùng của ngón tay tách biệt với nền có thể áp dụng với toàn bộ hình ảnh FKG thu thập được vì tất cả các ngón tay khi đưa vào hệ thống đều được cố định bởi các khối cơ bản khi chụp. Vì vậy có thể dễ dàng áp dụng một ngưỡng Threshold lên hình ảnh đầu vào để trích xuất đường biên dưới của ngón tay này. Sau đó, trục X của hệ tọa độ được xác định từ đường biên dưới đó, như minh họa ở Hình 2.2.



Hình 2.2 Xác định trục X của hệ tọa độ

Hình 2.1 b) biểu diễn một hình ảnh sau khi đã tìm ra trục X của hệ tọa độ ROI.

2.2.3 Crop hình ảnh theo trục X đã xác định

Vì những thông tin quan trọng của hình ảnh dùng để nhận diện ra con người không phải chứa trong toàn bộ khung hình gốc nên sau khi đã xác định được trục X của hệ tọa độ, em sẽ ước tính theo chiều rộng ngón tay người để xác định đường biên trên của ngón tay, từ đó sẽ cắt được theo chiều dài là hai đường biên đã được xác định.

Còn việc crop theo chiều rộng, hai ranh giới trái phải em sẽ dựa vào các khối cố định của hệ thống khi chụp, từ đó đưa ra giá trị hai ranh giới trái phải theo kinh nghiệm thực nghiệm. Hình ảnh thu được sau bước này chưa phải hình ảnh ROI hoàn chỉnh.

Hình 2.1 c) biểu diễn một hình ảnh đã crop sau khi xác định được trục X ở bước trên, đường bên trên cùng ranh giới trái phải được xác định bằng thực nghiệm.

2.2.4 Dò cạnh sử dụng thuật toán Canny

Thuật toán Canny được sử dụng trong các bài toán cần dò cạnh (Edge Detection) được em tham khảo bởi [1]. Chi tiết các bước thực hiện thuật toán được em trình bày ở phần dưới đây.

Bước 1 : Tính toán cường độ gradient trên mỗi pixel hình ảnh đầu vào.

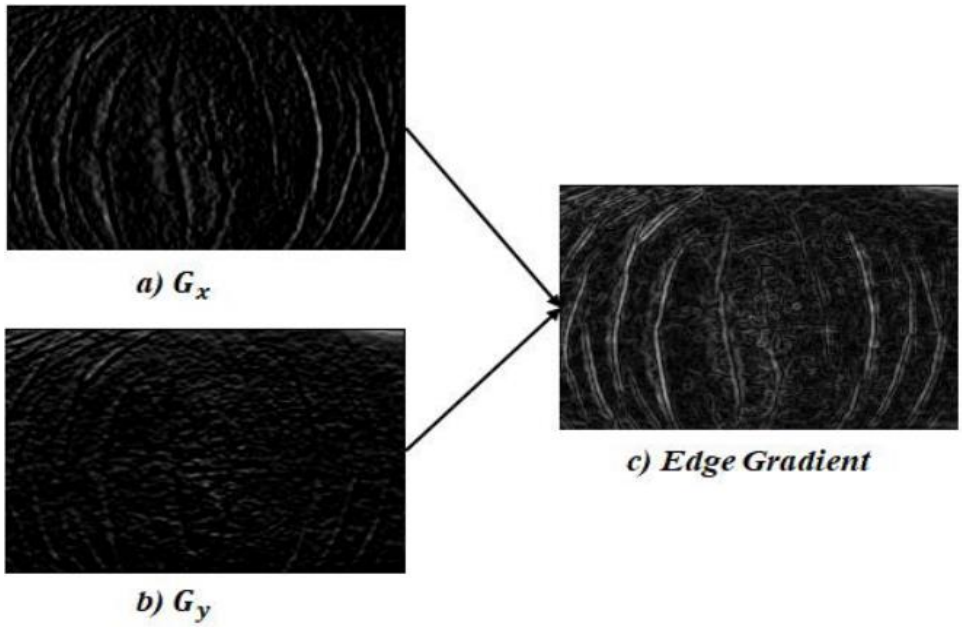
Đầu tiên, thuật toán Canny cung cấp một kernel có kích thước 3×3 nhân tích chập với ảnh đầu vào để tính toán đạo hàm theo hướng ngang (G_x) và hướng dọc (G_y).

Giá trị của đạo hàm theo hai hướng ngang và dọc chính là sự thay đổi của giá trị pixel theo hướng ngang hoặc dọc. Ta thu được các hình ảnh chứa các nét theo chiều ngang và dọc của ảnh đầu vào.

Sau khi đã có được đạo hàm của hình ảnh đầu vào theo hai hướng ngang và dọc, ta sẽ tính toán được cường độ gradient theo công thức dưới đây:

$$\text{Edge\_Gradient} = \sqrt{G_x^2 + G_y^2} \qquad \textit{PT 2.1}$$

Hình ảnh cường độ gradient thu được như minh họa ở Hình 2.3.



Hình 2.3 Cường độ gradient hình ảnh đầu vào

Nhận thấy đa phần hình ảnh ngón tay thu được, ví dụ ở Hình 2.3.(c), có các đường cạnh khá dày. Lý do đó là có nhiều pixel cùng mang một đặc trưng về cạnh. Vì vậy, em sẽ loại bỏ các pixel trùng nhau đó để hình ảnh cạnh thu được sẽ trở nên sắc nét và mảnh hơn giống ảnh đầu vào. Để làm được điều đó thì em sẽ tính toán thêm một yếu tố nữa đó là hướng của gradient.

Bước 2: Tính toán hướng của gradient trên mỗi pixel hình ảnh đầu vào.

Mỗi pixel trên hình ảnh đầu vào đều mang một thông tin về góc, hay nói cách khác chính là hướng của cạnh. Ví dụ như nếu hướng của gradient bằng 0 độ, thì cạnh trên ảnh sẽ là một đường thẳng đứng song song với trục tung (góc Gradient luôn luôn vuông góc với cạnh).

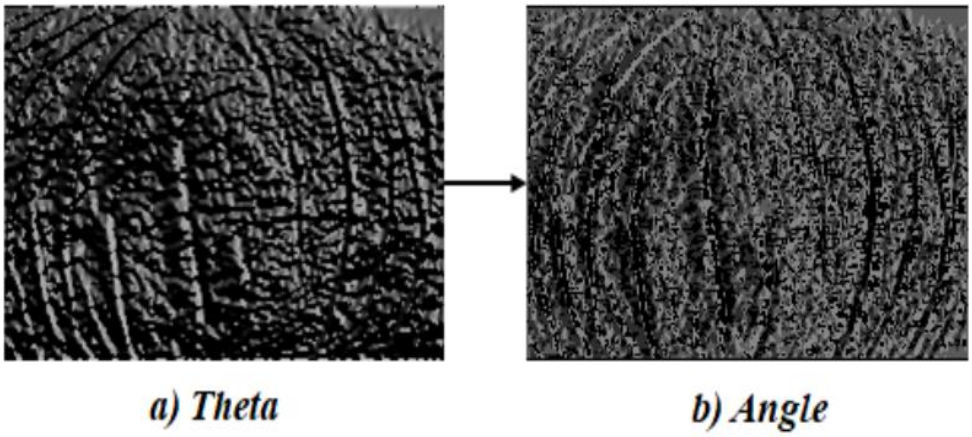
Thông tin về hướng của pixel được tính toán theo công thức sau:

$$\text{Theta} = \tan^{-1}\frac{G_y}{G_x} \qquad \textit{PT 2.2}$$

Kết quả nhận được giá trị hướng của gradient thuộc [180°, 180°]. Tuy nhiên em sẽ không giữ nguyên các góc này để tính toán mà đưa các giá trị này về 4 hướng đại diện đó là [0°, 45°, 90°, 135°] theo quy ước sau đây:

$$\text{Angle} = \begin{cases} 0°, |\text{Theta}| \leq 22.5° \text{ or } |\text{Theta}| \geq 157.5° \\ 45°, 22.5° \leq |\text{Theta}| \leq 67.5° \\ 90°, 67.5° \leq |\text{Theta}| \leq 112.5° \\ 135°, \text{else} \end{cases} \qquad PT\ 2.3$$

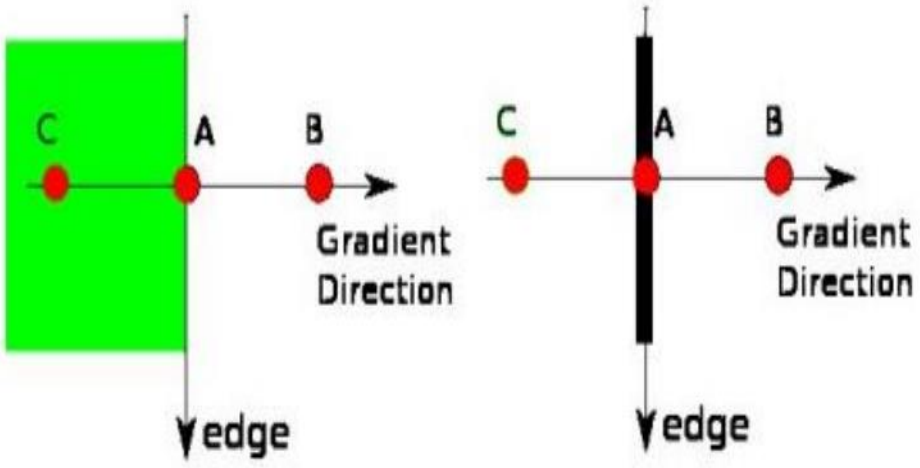
Hình ảnh hướng của gradient thu được như như ở Hình 2.4.



Hình 2.4 Hình ảnh hướng của gradient trước và sau khi quy đổi

Bước 3: NonMaximum Supression (NMS).

Ở bước này, em sẽ sử dụng một bộ lọc có kích thước 3×3 chạy lần lượt qua từng giá trị pixel trên ảnh gradient. Trong quá trình lọc này, em sẽ so sánh cường độ gradient của pixel trung tâm với hai pixel lân cận tùy vào hướng của gradient. Nếu độ lớn gradient của pixel trung tâm lớn hơn hai pixel lân cận đó, em sẽ giữ lại pixel đó, nếu không, em sẽ set cho giá trị gradient của pixel đó về 0.

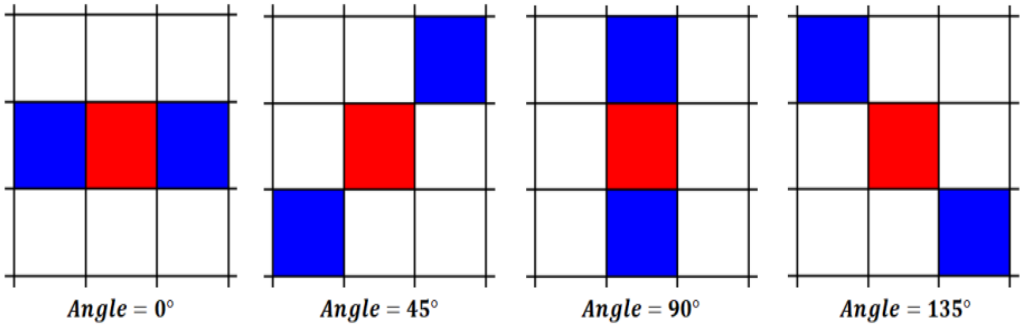


Hình 2.5 Minh họa so sánh cường độ gradient các vector lân cận

Với một pixel A được xác định nằm trên một cạnh,ta sẽ có vector Gradient Direction luôn vuông góc với cạnh (Edge). Trên vector Gradient Direction có thể có nhiều pixel mang cùng thông tin đặc trưng của cạnh đó (ví dụ như Hình 2.5 là

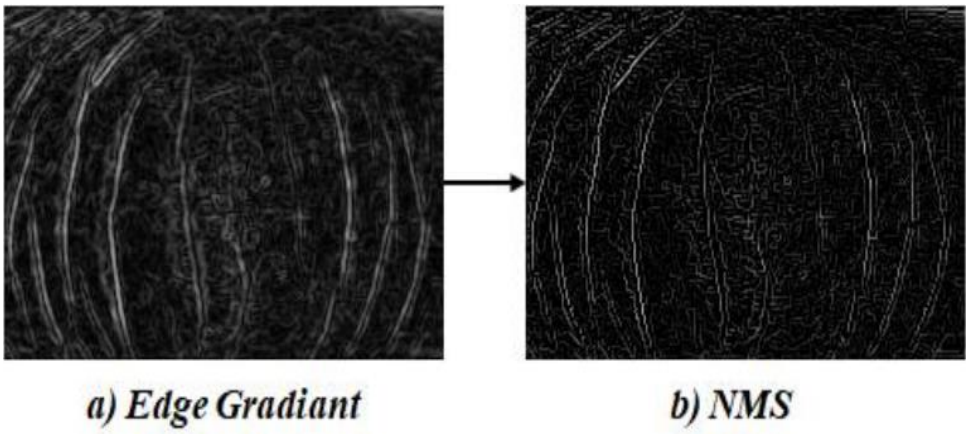
pixel B và C). Như vậy việc so sánh cường độ gradient vùng cục bộ chính là việc so sánh cường độ gradient của 3 pixel A, B và C.

Tùy thuộc vào hướng của gradient, em sẽ sử dụng hai pixel lân cận pixel trung tâm của vùng cục bộ như Hình 2.6 (ô màu đỏ là pixel trung tâm, ô màu xanh là hai pixel lân cận cần so sánh)



Hình 2.6 So sánh gradient các pixel lân cận theo hướng gradient

Hình ảnh thu được sau khi loại bỏ các pixel được thể hiện như ở Hình 2.7.



Hình 2.7 Áp dụng NMS loại bỏ các pixel thừa

Bước 4: Lọc ngưỡng

Ở bước cuối cùng này, em sẽ triển khai kiểm tra xem các cạnh thu được ở bước trên có là cạnh thực sự hay không bằng cách đặt một Threshold là V_{min} những pixel thực sự là cạnh sẽ có gradient lớn hơn V_{min} .

Từ hình ảnh thu được sau khi loại bỏ các pixel thừa ở bước 3 và Threshold V_{min} em sẽ lọc và giữ lại những vị trí được coi thực sự là cạnh.

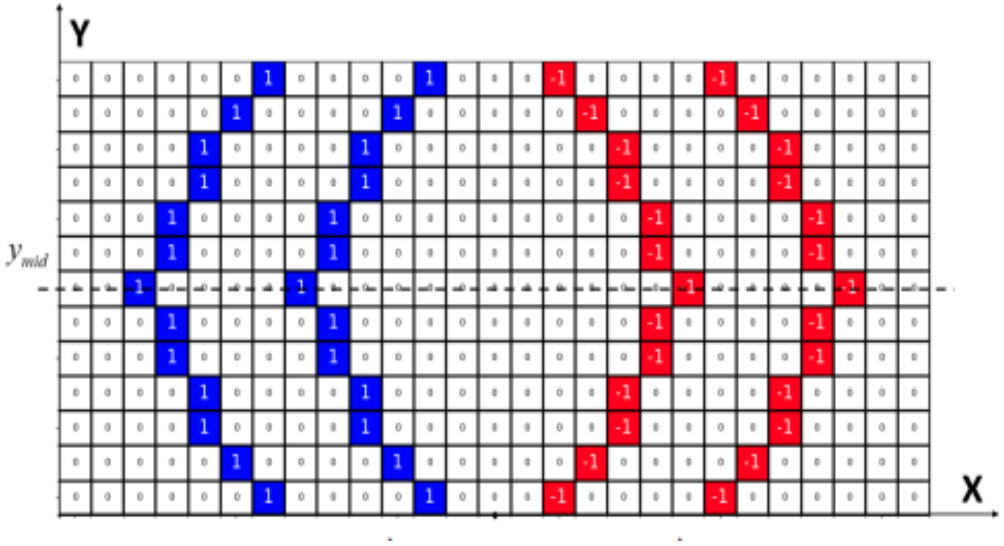
Hình ảnh mặt nạ và hình ảnh cạnh cuối cùng thu được như Hình 2.8:

Hình 2.8 Hình ảnh Canny Edge thu được sau khi áp dụng lọc ngưỡng

Sau khi sử dụng phương pháp dò cạnh Canny, chúng ta thu được sơ đồ hình ảnh cạnh (I_E) tương ứng như Hình 2.1(d).

2.2.5 Mã hóa đường vân lồi của khớp ngón tay

Dựa vào sơ đồ cạnh thu được từ bước trước bằng thuật toán dò cạnh Canny (I_E), tiếp theo em sẽ mã hóa đường vân lồi của khớp ngón tay, từ đó thu được sơ đồ mã hóa vân lồi của khớp ngón tay I_{CD}.



Hình 2.9 Minh họa sơ đồ mã hóa đường vân lồi của khớp ngón tay

Ở bước này, em sẽ cung cấp cho mỗi pixel từ I_{CD} một giá trị để đại diện cho hướng lồi của vân khớp ngón tay (lồi sang trái hoặc phải). Gán giá trị cho những pixel có hướng lồi sang trái là 1, sang phải là -1 và các pixel không mang thông tin hướng lồi có giá trị là 0.

Giá trị của từng pixel I_{CD} (i,j) được gán theo quy tắc như sau:

$$I_{CD}(i,j) = \begin{cases} 0, I_E(i,j) = 0 \text{ or } I_E(i+1,j-1) = I_E(i+1,j+1) \\ 1, (I_E(i+1,j-1) = 1 \text{and } i \le y_{mid}) \text{ or } (I_E(i+1,j+1) = 1 \text{and } i > y_{mid}) \\ -1, (I_E(i+1,j+1) = 1 \text{and } i \le y_{mid}) \text{ or } (I_E(i+1,j-1) = 1 \text{and } i > y_{mid}) \end{cases} \quad PT\ 2.4$$

Với y_{mid} mang giá trị bằng một nửa giá trị chiều cao của hình ảnh cạnh thu được từ bước 4.

Hình 2.1. (e) và Hình 2.9. biểu diễn sơ đồ mã hóa đường vân lồi của khớp ngón tay thu được sau khi đã gán giá trị cho từng pixel của I_{CD} (Pixel màu trắng mang giá trị bằng 0, không mang thông tin hướng lồi, pixel màu xanh mang giá trị bằng 1, có xu hướng lồi sang trái, pixel màu đỏ mang giá trị bằng -1, có xu hướng lồi sang phải).

2.2.6 Xác định trục Y của hệ tọa độ

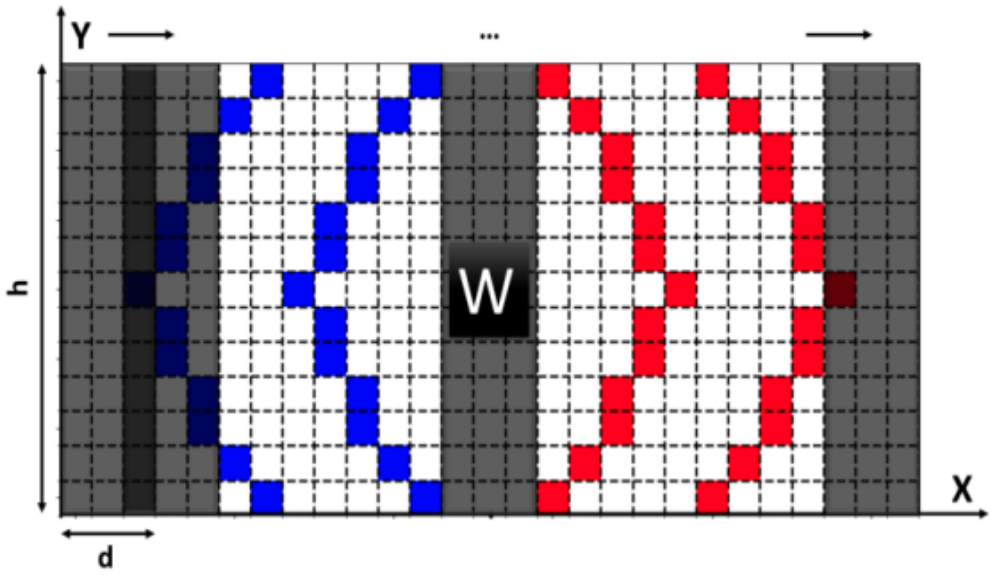
Xem xét sơ đồ mã hóa đường vân lồi của khớp ngón tay và ảnh gốc khớp ngón tay ta thấy một cách rõ ràng như sau: những đường vân nằm bên trái khớp ngón tay có xu hướng lồi về bên trái, ngược lại những đường vân nằm bên phải khớp ngón tay có xu hướng lồi về bên phải. Bên cạnh đó, một số đường vân nằm ở giữa khớp ngón tay không có xu hướng lồi rõ rệt.

Để thuận tiên cho việc quan sát cũng như xác định trục Y của hệ tọa độ, em dùng khái niệm cường độ lồi của đường vân khớp ngón tay như sau:

$$\text{ConMag}(x) = \left| \sum_{w} I_{CD} \right| \quad PT\ 2.5$$

Trong đó: x là tọa độ theo trục X của ảnh.

W là cửa sổ trượt có kích thước d×h, với h là chiều cao ảnh I_{CD}, d = 35 lấy theo kinh nghiệm thực nghiệm. Minh họa cửa sổ trượt W tính toán cường độ lồi được thể hiện ở Hình 2.10.



Hình 2.10 Minh họa cửa sổ trượt W tính toán cường độ lồi

Có thể diễn giải phương trình (2.5) theo cách khác như sau:

$$\mathrm{ConMag}(x) = \left| \sum_{i=x}^{x+34} \sum_{j=0}^{Y-1} I_{CD}(i,j) \right| \qquad PT\ 2.6$$

Cường độ lồi là khái niệm được đưa ra để đo cường độ của một hướng lồi chiếm ưu thế trong một khu vực thuộc ảnh FKP. Từ đặc điểm ảnh FKP, ta tính toán và đưa ra được kết luận: giá trị hàm $\mathrm{ConMag}(x)$ sẽ đạt đến tối thiểu ở vùng chính giữa của khớp ngón tay. Từ đặc điểm này ta có thể tìm ra chính xác điểm x_0 để đặt trục Y của hệ tọa độ sao cho giá trị hàm $\mathrm{ConMag}(x)$ đạt cực tiểu.

Như vậy:

$$x_0 = \underset{x}{\mathrm{argmin}}\big(\mathrm{ConMag}(x)\big) \qquad PT\ 2.7$$

Đường thẳng $X = x_0$ chính là trục Y của hệ tọa độ.

Hình 2.1 (f) biểu diễn tập hợp các giá trị cường độ lồi nhận được từ (2.5), từ đó có thể dễ dàng tìm được giá trị x_0 bởi phương trình (2.7).

Hình 2.1 (g) biểu diễn trục tọa độ Y chính là đường thẳng $X = x_0$.

2.2.7 Crop hình ảnh theo trục Y đã xác định

Đến bước cuối cùng này, chúng ta đã có trục X và trục Y nằm cố định. Tiếp theo chúng ta sẽ chỉnh sửa và cắt lại ảnh ROI sao cho phù hợp.

Bằng kinh nghiệm thực nghiệm, em đưa ra kết luận ảnh ROI có trục X, Y xác định như trên và crop lại theo kích thước 110×220 mang tới hiệu quả tính toán nhanh mà vẫn đảm bảo độ chính xác nhất trong hệ thống này.

Hình 2.1. (h) biểu diễn vùng quan tâm ROI sau khi đã xác định đầy đủ hai trục tọa độ X và Y.

Hình 2.1. (k) biểu diễn hình ảnh ROI hoàn chỉnh thu được.

Kết luận chương 2:Trong trường hợp ngón tay đặt tại các vị trí khác nhau trong thiết bị, thuật toán trích xuất ROI em trình bày tại chương này vẫn hoạt động khá tốt, chi tiết về kết quả thực nghiệm và thời gian xử lý em sẽ trình bày tại CHƯƠNG 4.

CHƯƠNG 3. PHƯƠNG PHÁP NHẬN DIỆN KHỚP NGÓN TAY SỬ DỤNG ĐẶC TRƯNG VỀ HƯỚNG VÀ ĐỘ LỚN ĐƯỜNG VÂN

Sau khi đã trích xuất được vùng quan tâm (ROI), việc tiếp theo là tiến hành sử dụng hình ảnh ROI thu được để nhận diện khớp ngón tay người. Hiện nay có rất nhiều phương pháp trích xuất đặc trưng khớp ngón tay để nhận diện, như Local Binary Patterns – LBP (bài báo số [6]), Robust Line Orientation Code – RLOC (bài báo số [8]), Three-Patch Local Binary Patterns Codes – TPLBP (bài báo số [3]), Fragile-Bit (bài báo số [5]), ImcompCodeMagCode (bài báo số[7]), CompCode (bài báo số [4]), Localized Radon Transform – LRT (bài báo số [9]), Binary Orientation Co-occurrence Vector – BOCV (bài báo số [2]), Radon Coefficients (bài báo số [10]), . . .

Trong đề tài này, em sẽ trình bày một số phương pháp trích xuất thông tin đặc trưng của khớp ngón tay người, từ đó sẽ dùng để nhận diện. Trước khi đi vào chi tiết phương pháp em sẽ sử dụng trong đề tài này, em sẽ trình bày một số phương pháp điển hình trong bài toán nhận diện khớp ngón tay đang được nghiên cứu và áp dụng phổ biến hiện nay: CompCode [4], BOCV [2], và LRT [9].

3.1 Thuật toán biến đổi sóng Gabor

Phép biến đổi sóng Gabor được sử dụng rộng rãi từ những năm 1980 , là một công cụ hiệu quả trong những bài toán yêu cầu khai thác, trích xuất đặc tính, đặc trưng trên khuôn mặt, mống mắt, vân tay, . . .

Phép biến đổi sóng Gabor có thể đồng thời trích xuất ba loại thông tin là cường độ, pha và định hướng, những loại thông tin này có thể sử dụng riêng biệt hoặc sử dụng đồng thời trong các hệ thống khác nhau.

Trong đề tài này, các phương pháp được trình bày ở mục 3.2 (như CompCode, BOCV hay ImcompCode MagCode) đều sử dụng phép biến đổi Gabor để trích xuất đặc trưng. Công thức Gabor (đề cập ở bài báo số [11]) được cấu thành từ hàm Gaussian nhân với hàm sin(cos) được trình bày một cách tổng quan như sau:

$$\psi(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2+y'^2\right)}\left(e^{i\omega_0 x'} - e^{-\frac{\kappa^2}{2}}\right) \qquad PT\ 3.1$$

Với:

$$x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta \qquad PT\ 3.2$$

$$y' = -(x - x_0)\sin\theta + (y - y_0)\cos\theta \qquad PT\ 3.3$$

Công thức Gabor được biểu diễn dưới dạng số phức, bao gồm hai phần là phần thực và phần ảo. Mặt khác, ta có công thức Euler như sau:

$$e^{ix} = \cos(x) + y\sin(x) \qquad PT\ 3.4$$

Từ phương trình (3.1) và phương trình (3.4) ta có thể biểu diễn phần thực và phần ảo của phép biến đổi sóng Gabor theo công thức sau:

$$\psi_R(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2+y'^2\right)}(\cos(\omega_0 x' - e^{-\frac{\kappa^2}{2}}) \qquad PT\ 3.5$$

và phần ảo của phép biến đổi sóng Gabor được biểu diễn như sau:

$$\psi_I(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2+y'^2\right)}(\sin(\omega_0 x') - e^{-\frac{\kappa^2}{2}}) \qquad PT\ 3.6$$

Các thành phần trong phương trình (3.5) bao gồm:

x’ là tích vô hướng của vector (x,y) và vector $\overrightarrow{Oa}$ = (cosθ,sinθ)

Tương tự, y’ là tích vô hướng của vector (x,y) và vector $\overrightarrow{Ob}$ = (−sinθ, cosθ)

Nhận xét rằng: Hai vector $\overrightarrow{Oa}$ và $\overrightarrow{Ob}$ vuông góc với nhau (có tích vô hướng bằng 1). Như vậy có thể hiểu rằng (x’,y’) chính là điểm (x,y) sau khi đã được biến đổi từ hệ tọa độ gốc Oxy sang hệ tọa độ mới O’xy với $\overrightarrow{Oa}$ và $\overrightarrow{Ob}$ chính là hai vector đơn vị. Khi θ = 0 thì x’ = x, y’ = y, hệ tọa độ O’xy ≡ Oxy.

ω_0 là tần số xuyên tâm tính theo đơn vị radian trên một đơn vị chiều dài.

κ là một hằng số.

θ là góc định hướng của phép biến đổi sóng Gabor, tính theo đơn vị radian.

3.2 Một số phương pháp trích xuất đặc trưng khớp ngón tay điển hình

3.2.1 Phương pháp mã hóa sơ đồ định hướng (CompCode)

Phương pháp nhận diện khớp ngón tay sử dụng sơ đồ mã hóa định hướng (CompCode) (được tham khảo từ bài báo số [4]) sử dụng thuật toán biến đổi sóng Gabor để trích xuất thông số định hướng của khớp ngón tay để mã hóa hình ảnh ROI. Cụ thể các bước em sẽ trình bày phía dưới đây.

Hàm Gabor tồn tại dưới nhiều hình thức khác nhau, tùy vào từng bài toán. Ở bài toán này, em sẽ sử dụng công thức phần thực của phép biến đổi sóng Gabor được trình bày ở mục 3.1, phương trình (3.5) , được nhắc đến trong các bài báo [5], [4], [14], [7], [2]:

$$\psi_R(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2+y'^2\right)}(\cos(\omega_0 x' - e^{-\frac{\kappa^2}{2}})$$

Trong đó:

$$x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta$$
$$y' = -(x - x_0)\sin\theta + (y - y_0)\cos\theta \qquad PT\ 3.7$$

(x_0, y_0) là điểm gốc tọa độ ban đầu của hàm, như vậy (x′, y′) chính là điểm (x,y) sau khi đã được biến đổi từ hệ tọa độ với gốc tọa độ là điểm (x_0, y_0) sang hệ tọa độ mới với $\overrightarrow{Oa}$ = (cosθ,sinθ) và $\overrightarrow{Ob}$ = (−sinθ, cosθ) chính là hai vector đơn vị. Khi θ = 0 thì $x' = x - x_0$,, $y' = y - y_0$,.

θ là góc của phép biến đổi sóng Gabor đơn vị radian.

$\omega = \frac{\kappa}{\sigma}$ là tần số xuyên tâm tính theo đơn vị radian trên một đơn vị chiều dài.

σ là độ lệch chuẩn trong công thức Gaussian.

κ là một hằng số, được định nghĩa bởi công thức $\kappa = \sqrt{2\ln 2}(\frac{2^{\delta}+1}{2^{\delta}-1})$với δ là băng thông nửa biên độ của đáp ứng tần số.

Sau khi đã định nghĩa công thức của phép biến đổi sóng Gabor, tiếp đếp em lựa chọn thông số cho bộ lọc và trích xuất đặc trưng về định hướng của đường vân ảnh FKP. Cuối cùng, em sẽ kết hợp với ảnh mask để đối sảnh ảnh đầu vào với ảnh trong tập dữ liệu để đưa ra kết luận người đó có thuộc tập dữ liệu hay không.

Bước 1: Lựa chọn thông số cho phép biến đổi sóng Gabor.

Các thông số của phép biến đổi Gabor được lựa chọn một cách tối ưu cho hệ thống dựa vào thực nghiệm được tham khảo từ bài báo [7]. Kích thước của bộ lọc Gabor trong bài toán này em lựa chọn là 35x35 với vị trí trung tâm của bộ lọc $(x_0, y_0) = (17,17)$, σ = 5.3, δ = 3.3

Các góc định hướng của bộ lọc em lựa chọn là 6 giá trị: $\theta = \frac{j\pi}{6}$, j = {0,1,2,3,4,5}

Tương ứng có 6 giá trị góc định hướng: $\theta = 0, \frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}$.

Bước 2: Trích xuất sơ đồ mã hóa định hướng cho FPK.

Thông tin về hướng được trích xuất theo từng pixel, gọi là mã định hướng. Trong bài toán này, chỉ phần thực của phép biến đổi sóng Gabor được sử dụng để mã hóa định hướng, theo công thức được trình bày như sau:

$$\mathrm{CompCode}(x,y) = \underset{j}{\mathrm{argmax}}\{|I_{ROI}(x,y) * \psi_R(x,y,\theta_j)|\} \quad PT\ 3.8$$

Trong đó: CompCode (x,y) là mã định hướng của pixel [x, y]

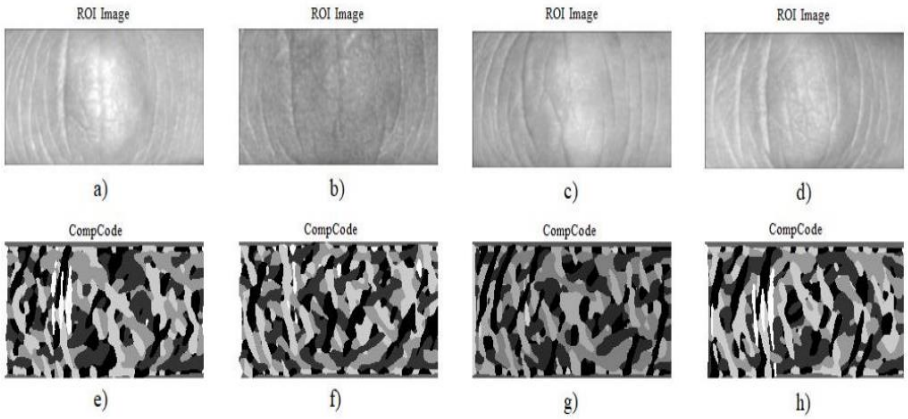
ψ_R (x, y,θj) là phần thực của phép biến đổi sóng Gabor ở góc θ_j với $\theta_j = \frac{j\pi}{6}$

I_{ROI}(x, y) là giá trị pixel [x,y] của hình ảnh ROI.

Phép "*" là phép nhân tích chập hai chiều giữa hình ảnh FKP ROI và phần thực của phép biến đổi sóng Gabor.

Nhận xét: theo phương trình (3.8) trên, ta thấy ma trận CompCode gồm các giá trị thuộc tập giá trị {0,1,2,3,4,5}.

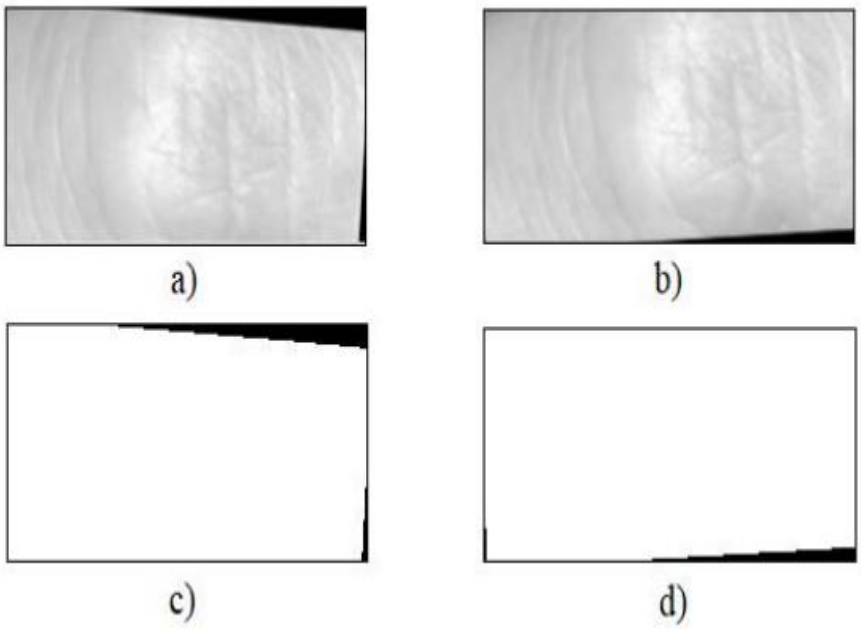
Hình ảnh sơ đồ mã hóa định hướng thu được thể hiện ở Hình 3.1:



Hình 3.1 a)~d) Hình ảnh ROI, e)~h) sơ đồ mã hóa định hướng

Bước 3: Xác định Mask cho hình ảnh đầu vào:

Thuật toán này ngoài sơ đồ mã hóa định hướng ra, em sẽ sử dụng hình ảnh Mask để xác định những pixel không nằm trong vùng khớp ngón tay, ví dụ như khi ngón tay đặt lệch khỏi trục, như minh họa ở Hình 3.2.



Hình 3.2 a), b) minh họa ngón tay lệch khỏi trục, c), d) là mask của a) và b)

Phương pháp tạo mask được áp dụng trong bài toán nhận diện khớp ngón tay này được em tham khảo theo bài báo số [15], áp dụng một Threshold lên hình ảnh đầu vào (những phần là nền, không thuộc vùng ảnh khớp ngón tay sẽ có màu đen, dễ dàng phát hiện và đặt được Threshold đó).

Bước 4: Đối sánh ảnh

Để đối sánh và nhận diện ảnh, em sử dụng sơ đồ mã hóa định hướng và mask của ảnh ROI đã được trích xuất.

Giả sử cần đối sánh hai ảnh R1 (ROI1) và R2 (ROI2) có cùng kích thước X × Y qua bước trích xuất dữ liệu ta thu được 2 cặp sơ đồ mã hóa định hướng và mask là (P,Q) và (P_M, Q_M) của 2 ảnh ROI1 và ROI2.

Như vậy, để đối sánh hai ảnh ROI1 và ROI2, em sẽ thực hiện tính toán khoảng cách để đối sánh hai cặp ma trận (P,Q) và (P_M, Q_M)

Khoảng cách đối sảnh giữa hai ảnh ROI1 và ROI2 được tính theo công thức (3.9) như sau:

$$D(R_1, R_2) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}\left(P_M(x,y) \cap Q_M(x,y)\right) \times G(P(x,y), Q(x,y))}{3\sum_{x=1}^{X}\sum_{y=1}^{Y} P_M(x,y) \cap Q_M(x,y)} \quad \textit{PT 3.9}$$

Trong đó $G(P(x,y), Q(x,y))$ nhận giá trị:

$$G\left(P(x,y), Q(x,y)\right) = \min(|P(x,y) - Q(x,y)|, 6 - |P(x,y) - Q(x,y)|) \quad \textit{PT 3.10}$$

Phép ∩ là phép logic AND giữa 2 bit 0 và 1 (mask là ma trận chứa các phần tử 0 và 1) như trên Bảng 3.1.

Bảng 3.1 Định nghĩa phép logic AND

$P_M(x,y)$	$Q_M(x,y)$	$P_M(x,y) \cap Q_M(x,y)$
0	0	0
1	0	0
0	1	0
1	1	1

Nhận xét: theo phương trình (3.10) trên, ta thấy G nhận giá trị thuộc {0,1,2,3} do P và Q là 2 ma trận CompCode chứa các phần tử có giá trị thuộc {0,1,2,3,4,5}. Từ đó suy ra giá trị của $D(R_1, R_2)$ nhận được thuộc trong khoảng (0,1).

Như vậy, để nhận diện được một ảnh, ta sẽ thực hiện đối sánh ảnh này với lần lượt từng ảnh trong cơ sở dữ liệu, thu được một tập các giá trị khoảng cách:

$$\text{distance} = \{D\left(R, R_j\right), j = \overline{0, n-1}\}$$

Với R là hình ảnh ROI cần nhận diện, R_j , $j = \overline{0, n-1}\}$ là tập những hình ảnh ROI trong cơ sở dữ liệu để đối sánh, n là số ảnh trong cơ sở dữ liệu.

Dựa vào giá trị nhỏ nhất trong tập giá trị distance đó, ta sẽ đưa ra kết luận người đó có thuộc cơ sở dữ liệu hay không và đó là người nào.

3.2.2 Phương pháp nhị phân hóa sơ đồ mã hóa định hướng (BOCV)

Cũng như phương pháp mã hóa sơ đồ định hướng được trình bày ở mục 3.2.1, thuật toán Binary Orientation Co-occurrence Vector (BOCV) (được tham khảo ở bài báo số [2]) sử dụng phép biến đổi sóng Gabor để trích xuất thông số định hướng của khớp ngón tay để mã hóa hình ảnh ROI. Cụ thể các bước em sẽ trình bày phía dưới đây.

Em sử dụng phần thực của phép biến đổi sóng Gabor được trình bày ở mục 3.1, phương trình (3.5):

$$\psi_R(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2+y'^2\right)}(\cos(\omega_0 x') - e^{-\frac{\kappa^2}{2}})$$

Bước 1: Tính toán vector OCV.

Sau khi đã định nghĩa công thức của phép biến đổi sóng Gabor và lựa chọn thông số như phương pháp CompCode được trình bày ở mục 3.2.1, tiếp đếp em sẽ tính toán một vector 6 chiều bằng cách chuẩn hóa theo 6 hướng của phép biến đổi sóng Gabor, gọi là vector định hướng (OCV - orientation co-occurrence) theo công thưc sau:

$$OCV = \{G_j(x, y), j = \overline{0,5}\} \qquad PT\ 3.11$$

Với:

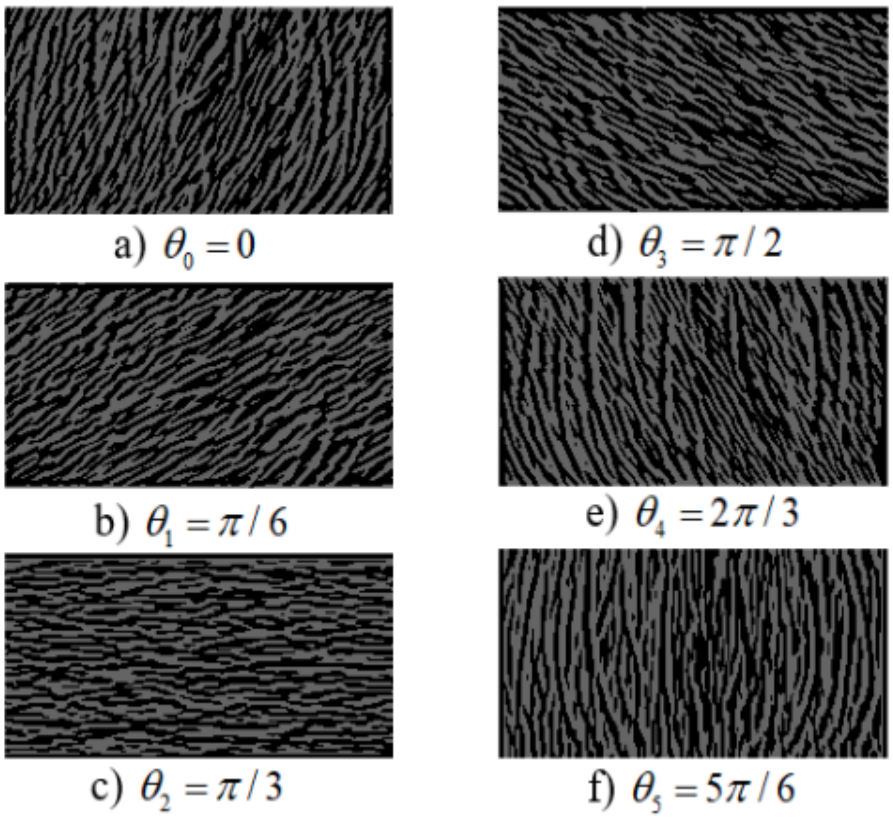
$$G_j(x, y) = I_{ROI}(x, y) * \psi_R(x, y, w, \theta_j), \theta_j = \frac{j\pi}{6} \qquad PT\ 3.12$$

Bước 2: Chuẩn hóa nhị phân vector định hướng OCV.

Theo phương trình (3.11) và phương trình (3.12) được định nghĩa ở trên, các phần tử của vector OCV đều là số thực. Để tăng tốc độ tính toán trong quá trình đối sánh ảnh, em sẽ chuẩn hóa vector OCV theo mã nhị phân 6 bit, gọi là Binary OCV (BOCV), như minh họa ở Hình 3.3.

Công thức chuẩn hóa như sau:

$$P_j' = \begin{cases} 1, G_j(x, y) < 0 \\ 0, \text{otherwise} \end{cases} \qquad PT\ 3.13$$



Hình 3.3 a) ~ f): vector BOCV với các hướng θ = 0, π/6, π/3, π/2, 2π/3, 5π/6

Bước 3: Xác định Mask cho hình ảnh đầu vào.

Thuật toán này ngoài sơ đồ mã hóa vector BOCV, em cũng sử dụng thêm hình ảnh Mask giống như phương pháp mã hóa sơ đồ định hướng được đề cập trong mục 3.2.1, tham khảo từ bài báo số [13] để xác định những pixel không nằm trong vùng khớp ngón tay, ví dụ như khi ngón tay đặt lệch khỏi trục.

Bước 4: Đối sánh ảnh.

Để đối sánh và nhận diện ảnh, em sử dụng vector mã hóa BOCV và mask của ảnh ROI đã được trích xuất.

Giả sử cần đối sánh hai ảnh ROI1 và ROI2 có cùng kích thước X x Y, qua bước trích xuất dữ liệu ta thu được cặp vector mã hóa BOCV1 = $\{P_j^b, j = \overline{0,5}$, BOCV2 = $\{Q_j^b, j = \overline{0,5}\}$ và cặp mask là (P_M, Q_M) của hai ảnh ROI1 và ROI2.

Như vậy, để đối sánh hai ảnh ROI1 và ROI2, em sẽ thực hiện tính toán khoảng cách để đối sánh hai cặp ma trận (BOCV1,BOCV2) và (P_M, Q_M).

Khoảng cách đối sảnh giữa hai ảnh ROI1 và ROI2 được tính theo công thức sau:

$$D(R_1, R_2) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}\sum_{j=0}^{5}\left(P_j^b(x,y) \otimes Q_j^b(x,y)\right) \cap \left(P_M(x,y) \cap Q_M(x,y)\right)}{6\sum_{x=1}^{X}\sum_{y=1}^{Y} P_M(x,y) \cap Q_M(x,y)} \quad \textit{PT 3.14}$$

Phép ⊗ là phép logic XOR giữa 2 bit 0 và 1, như mô tả ở Bảng 3.2

Bảng 3.2 Định nghĩa phép logic XOR

$P_j^b(x,y)$	$Q_j^b(x,y)$	$P_j^b(x,y) \otimes Q_j^b(x,y)$
0	0	1
1	0	0
0	1	0
1	1	1

Nhận xét: theo phương trình (3.13) trên, ta thấy vector mã hóa BOCV và ma trận mask chứa các phần tử thuộc tập {0,1}, như vậy giá trị của $D(R_1,R_2)$ nhận được thuộc trong khoảng (0,1).

Như vậy, để nhận diện được một ảnh, ta sẽ thực hiện đối sánh ảnh này với lần lượt từng ảnh trong cơ sở dữ liệu, thu được một tập các giá trị khoảng cách:

$$\text{distance} = \{D(R, R_j), j = \overline{0, n-1}\}$$

Với R là hình ảnh ROI cần nhận diện, R_j , $j = \overline{0, n-1}\}$ là tập những hình ảnh ROI trong cơ sở dữ liệu để đối sánh, n là số ảnh trong cơ sở dữ liệu.

Dựa vào giá trị nhỏ nhất trong tập giá trị distance đó, ta sẽ đưa ra kết luận người đó có thuộc cơ sở dữ liệu hay không và đó là người nào.

3.2.3 Phương pháp biến đổi Radon trong một vùng cục bộ (LRT)

Phép biến đổi Radon (Radon Transform) trong không gian Euclidean được Johann Radon thiết lập lần đầu tiên vào năm 1917. Phép biến đổi Radon có thể làm nổi bật các tính năng tuyến tính bằng cách tích hợp cường độ hình ảnh dọc theo tất cả các dòng có thể có trong một hình ảnh, do đó nó có thể được sử dụng để phát trích xuất thông tin định hướng trong hình ảnh.

Localized Radon Transform là phép biến đổi Radon trong một vùng cục bộ, được thực hiện bằng cách tích hợp cường độ hình ảnh trên một đường hoặc một đoạn, thay vì dòng như phép biến đổi Radon (được tham khảo bởi bài báo số [14].

Công thức LRT áp dụng lên một hình ảnh ROI với vùng cục bộ R_q^2 được định nghĩa theo bài báo số [14] như sau:

$$s[X_p] = M_i(p) = \sum_{(x,y)\in X_p} R[x,y] \qquad \textit{PT 3.15}$$

Trong đó:

R[x, y] là tập giá trị pixel (x,y) nằm trong vùng cục bộ R_q^2

X_p là tập hợp các điểm cần quan tâm trong vùng cục bộ R_q^2 được định nghĩa như sau:

$$X_p = \{(x,y): y = p(x - x_0) + y_0, x \in R_q\} \qquad PT\ 3.16$$

Với(x_0, y_0) là điểm chính giữa của vùng cục bộ R_q^2.

R_q= { j, j = $\overline{0, q-1}$},q là kích thước 1 chiều của vùng cục bộ.

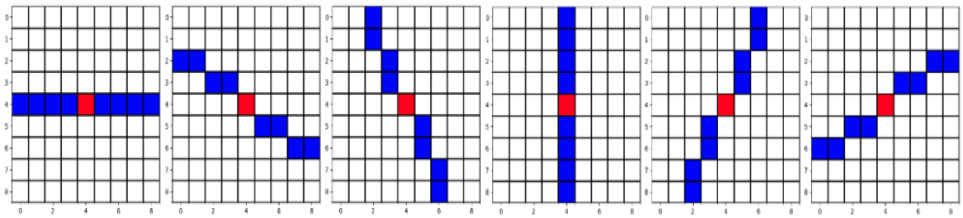
p biểu thị độ dốc của tập hợp X_p, ví dụ góc định hướng của tập X_p, là 0° thì p = tan(0°) = 0. Minh họa một số vùng cục bộ được thể hiện ở Hình 3.4 và 3.5.

Bước 1 Lựa chọn thông số cho phép biến đổi Radon cục bộ (LRT)

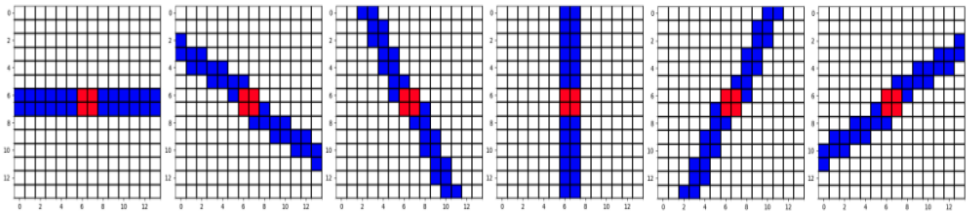
$$s[X_p] = M_i(p) = \sum_{(x,y)\in X_p} R[x,y] \qquad PT\ 3.17$$

Kích thước của vùng cục bộ R_q^2 và chiều rộng của đường X_p có thể được lựa chọn theo kinh nghiệm tương ứng với chiều rộng của các đường khớp ngón tay quan sát được trong hình ảnh ROI. Trong bài toán cụ thể này, em lựa chọn:

- Vùng cục bộ R_q^2 có kích thước 9 x 9 pixel
- Độ rộng X_p là 1 pixel.
- Số góc định hướng là 6.



Hình 3.4 Vùng cục bộ 9×9 với các góc định hướng iπ/6, độ rộng X_p là 1 pixel



Hình 3.5 Vùng cục bộ 14×14 với các góc định hướng iπ/6, độ rộng X_p là 2 pixel

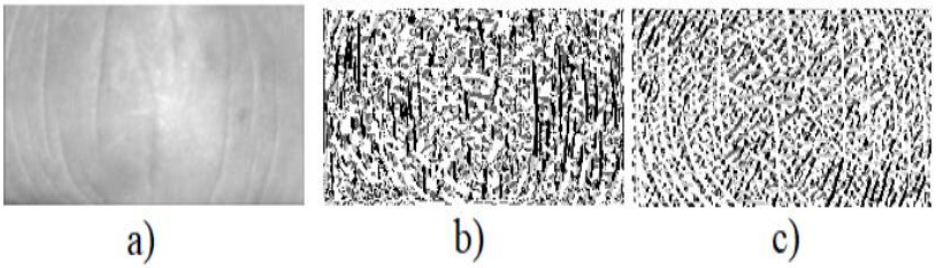
Bước 2: Trích xuất sơ đồ định hướng KnuckleCodes sử dụng LRT.

Sau khi đã lựa chọn thông số cho phép biến đổi Radon cục bộ (LRT), em sẽ thực hiện áp dụng LRT vào hình ảnh ROI theo từng pixel để trích xuất thông số định hướng theo công thức sau:

$$\omega_p(x_0, y_0) = \underset{p}{\operatorname{argmin}} s[X_p] \qquad PT\ 3.18$$

Trong đó $\omega_p(x_0, y_0)$ là mã định hướng của pixel (x_0, y_0) là điểm chính giữa của vùng cục bộ R_q^2 . Thao tác này được áp dụng trên toàn bộ các pixel $(x_0, y_0) \in$ ROI .

Hình ảnh sơ đồ mã hóa định hướng thu được sau khi sử dụng LRT được thể hiện ở Hình 3.6:



Hình 3.6 a): ảnh ROI, b) và c): sơ đồ mã hóa định hướng sử dụng LRT và bộ lọc Gabor

Bước 3: Đối sảnh ảnh.

Để đối sánh và nhận diện ảnh, em sử dụng sơ đồ mã hóa định hướng KnuckleCodes của ảnh ROI đã được trích xuất.

Giả sử cần đối sánh hai ảnh ROI1 và ROI2 có cùng kích thước $m \times n$, qua bước trích xuất dữ liệu ta thu được cặp sơ đồ mã hóa định hướng A và B của hai ảnh ROI1 và ROI2, đều có kích thước $m \times n$.

Như vậy, để đối sánh hai ảnh R1 (ROI1) và R2 (ROI2), em sẽ thực hiện tính toán khoảng cách để đối sánh cặp sơ đồ mã hóa A và B.

Có 3 phương pháp đối sánh cặp sơ đồ mã hóa A và B: đối sánh pixel to pixel, đối sánh pixel to cross-shape và đối sánh pixel to area.

Khoảng cách đối sánh giữa A và B được tính theo công thức sau:

$$s(A,B) = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} A(i,j) \cup \overline{B}(i,j)}{m \times n} \qquad PT\ 3.19$$

Trong đó: $\overline{B}(i,j)$ tùy vào phương pháp đối sánh có thể là pixel, có thể là crossshape hoặc area (được tham khảo từ bài báo số [5]).

- Đối với phương pháp đối sánh pixel to pixel:

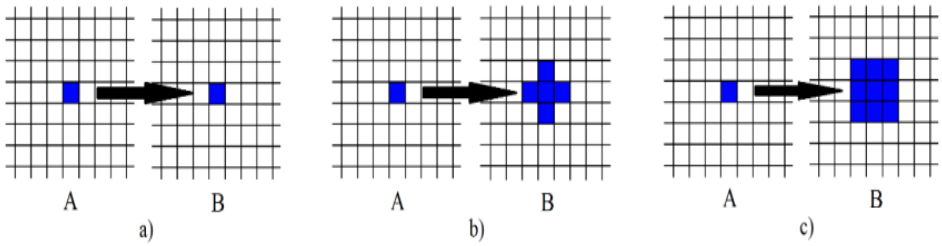
$$\overline{B}(i,j) = B(i,j) \qquad PT\ 3.20$$

- Đối với phương pháp đối sánh pixel to cross-shape:

$$\overline{B}(i,j) = \{B(i-1,j), B(i+1,j), B(i,j), B(i,j-1), B(i,j+1)\} \qquad PT\ 3.21$$

- Đối với phương pháp đối sánh pixel to area:

$$\overline{B}(i,j) = \{B(i-1,j-1), B(i-1,j+1), B(i-1,j), B(i+1,j), B(i,j), B(i,j-1), B(i,j+1), B(i+1,j+1), B(i+1,j-1)\} \qquad PT\ 3.22$$



Hình 3.7 a) ~ c) : đối sánh pixel to pixel, pixel to cross-shape và pixel to area

Một số hình ảnh so sánh ba phương pháp: pixel to pixel, pixel to crossshape và pixel to area được thể hiện ở Hình 3.7.

Phép ∪ được định nghĩa như sau: nếu giá trị A(i,j) bằng một trong những giá trị của $\overline{B}(i,j)$ phép hợp trả về giá trị là 1, ngược lại trả về 0.

$$A(i,j) \cup \overline{B}(i,j) = \begin{cases} 1, A(i,j) \subset \overline{B}(i,j) \\ 0, \text{otherwise} \end{cases} \qquad PT\ 3.23$$

Tương tự ta tính toán s(B,A) và kết quả cuối cùng sử dụng trong bài toán nhận diện khớp ngón tay đó là:

$$D(R_1, R_2) = \max(s(A,B), s(B,A)) \qquad PT\ 3.24$$

Như vậy, để nhận diện được một ảnh, ta sẽ thực hiện đối sánh ảnh này với lần lượt từng ảnh trong cơ sở dữ liệu, thu được một tập các giá trị khoảng cách:

$$\text{distance} = \{D(R, R_j), j = \overline{0, n-1}\}$$

Với R là hình ảnh ROI cần nhận diện, R_j , $j = \overline{0, n-1}\}$ là tập những hình ảnh ROI trong cơ sở dữ liệu để đối sánh, n là số ảnh trong cơ sở dữ liệu.

Dựa vào giá trị nhỏ nhất trong tập giá trị distance đó, ta sẽ đưa ra kết luận người đó có thuộc cơ sở dữ liệu hay không và đó là người nào.

3.3 Phương pháp nhận diện khớp ngón tay sử dụng đặc trưng về hướng và độ lớn của đường vân.

Như đã trình bày ở trên thì việc nhận diện khớp ngón tay người có thể thực hiện với nhiều phương pháp. Các phương pháp em đã nêu ở phần trước chưa đem lại hiệu quả thực sự với các hình ảnh ngón tay bị xoay, dịch hoặc mờ. Phương pháp nhận diện khớp ngón tay bằng việc sử dụng phép biến đổi sóng Gabor và mã hóa đường vân (được tham khảo từ bài báo số) giúp loại bỏ một số đặc tính không cần thiết cũng như tận dụng được lợi thế về độ rộng cũng như hướng của các đường vân trên khớp ngón tay đem lại kết quả nhận diện vượt trội hơn các phương pháp truyền thống.

3.3.1 Trích xuất đặc tính đường vân của khớp ngón tay bằng phép biến đổi sóng Gabor

Trong bài toán này, em sẽ trích xuất và kết hợp hai loại thông tin là định hướng và độ lớn đường vân từ phép biến đổi sóng Gabor để nhận dạng khớp ngón

tay. Kết quả thu được cho thấy phương pháp này có độ chính xác khá cao so với các phương pháp trích xuất dữ liệu khác như CompCode, BOCV hay LRT.

Hàm Gabor biểu diễn dưới nhiều hình thức khác nhau, tùy vào từng bài toán. Ở bài toán này, em sẽ sử dụng phần thực theo phương trình (3.5) giống phương pháp CompCode được trình bày ở mục 3.2.1:

$$\psi_R(x, y, \omega_0, \theta) = \frac{\omega_0}{\sqrt{2\pi}\kappa} e^{-\frac{\omega_0}{8\kappa^2}\left(4x'^2 + y'^2\right)} \left(\cos\left(\omega_0 x'\right) - e^{-\frac{\kappa^2}{2}}\right)$$

Trong đó:

$$x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta$$
$$y' = -(x - x_0)\sin\theta + (y - y_0)\cos\theta$$

(x_0, y_0) là điểm gốc tọa độ ban đầu của hàm, như vậy (x', y') chính là điểm (x,y) sau khi đã được biến đổi từ hệ tọa độ với gốc tọa độ là điểm (x_0, y_0) sang hệ tọa độ mới với $\overrightarrow{Oa}$ = (cosθ,sinθ) và $\overrightarrow{Ob}$ = (−sinθ, cosθ) chính là hai vector đơn vị. Khi θ = 0 thì $x' = x - x_0$, $y' = y - y_0$.

θ là góc của phép biến đổi sóng Gabor đơn vị radian.

$\omega = \frac{\kappa}{\sigma}$ là tần số xuyên tâm tính theo đơn vị radian trên một đơn vị chiều dài.

σ là độ lệch chuẩn trong công thức Gaussian.

κ là một hằng số, được định nghĩa bởi công thức $\kappa = \sqrt{2\ln 2}\left(\frac{2^\delta + 1}{2^\delta - 1}\right)$ với δ là băng thông nửa biên độ của đáp ứng tần số.

Sau khi đã định nghĩa công thức của phép biến đổi sóng Gabor, tiếp đếp em lựa chọn thông số cho bộ lọc và trích xuất đặc trưng về định hướng của đường vân ảnh FKP. Cuối cùng, em sẽ kết hợp với ảnh mask để đối sảnh ảnh đầu vào với ảnh trong tập dữ liệu để đưa ra kết luận người đó có thuộc tập dữ liệu hay không.

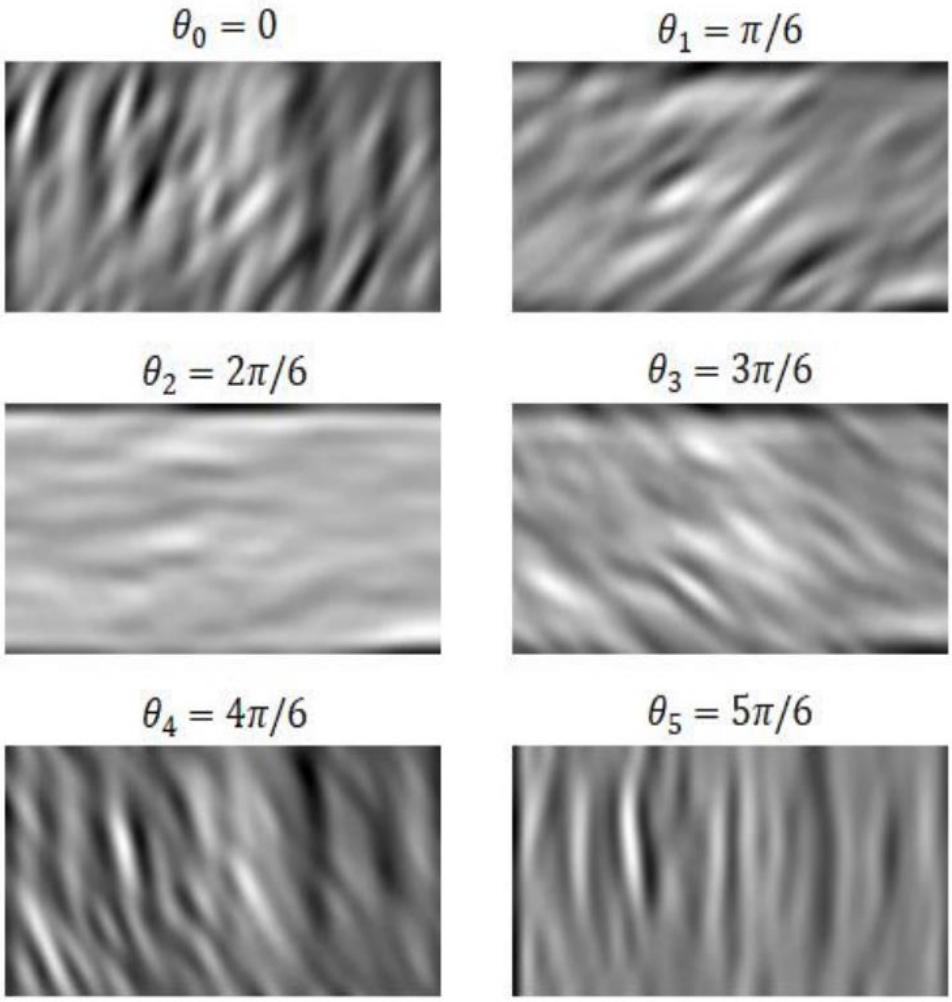
Bước 1: Lựa chọn thông số cho phép biến đổi sóng Gabor.

Các thông số của phép biến đổi sóng Gabor được lựa chọn như ở mục 3.2.1.

Kích thước của bộ lọc Gabor trong bài toán này em lựa chọn là 35x35 với vị trí trung tâm của bộ lọc (x_0, y_0) = (17,17), σ = 5.3, δ = 3.3

Các góc định hướng của bộ lọc em lựa chọn là 6 giá trị: $\theta = \frac{j\pi}{6}$, j = {0,1,2,3,4,5}

Tương ứng có 6 giá trị góc định hướng: $\theta = 0, \frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}$, như biểu diễn ở Hình 3.8



Hình 3.8 Kết quả phép biến đổi Gabor với các hướng θ = 0, π/6, 2π/6, 3π/6, 4π/6, 5π/6

Bước 2: Trích xuất đặc trưng về hướng (ImcompCode) sử dụng phép biến đổi sóng Gabor.

Thông tin về hướng được trích xuất theo từng pixel, gọi là mã định hướng. Trong bài toán này, chỉ phần thực của phép biến đổi Gabor được sử dụng để mã hóa định hướng, theo công thức (được tham khảo từ bài báo số [2]) trình bày như sau:

$$\mathrm{ImcompCode}(x,y) = \underset{j}{\mathrm{argmax}}\{I_{ROI}(x,y) * \psi_R(x,y,\theta_j)\} \qquad PT\ 3.25$$

Trong đó: ImcompCode (x,y) là mã định hướng của pixel [x, y].

ψ_R (x, y,θj) là phần thực của phép biến đổi sóng Gabor ở góc θ_j với $\theta_j = \frac{j\pi}{6}$

I_{ROI}(x, y) là giá trị pixel [x,y] của hình ảnh ROI.

Phép "*" là phép nhân tích chập hai chiều giữa hình ảnh FKP ROI và phần thực của phép biến đổi sóng Gabor.

Do trên hình ảnh FKP, có một vài điểm ảnh nằm trên các khu vực tương đối phẳng, không mang một thông tin về hướng cụ thể, vì thế sau khi sử dụng phép biến đổi sóng Gabor, giá trị phản hồi tại các pixel đó không có sự thay đổi nhiều.

Nếu chúng ta tiếp tục gán mã định hướng cho các pixel đó, điều này có thể làm cho thông tin của các pixel định hướng khác bên cạnh bị nhiễu, dẫn đến việc giảm độ tin cậy về thông tin về hướng của hình ảnh FKP đó. Vì vậy, em định nghĩa một khái niệm là cường độ định hướng, để loại bỏ các điểm ảnh "phẳng", không mang giá trị định hướng đó. Công thức về cường độ định hướng (được tham khảo từ bài báo số [7]) được trình bày ở dưới đây:

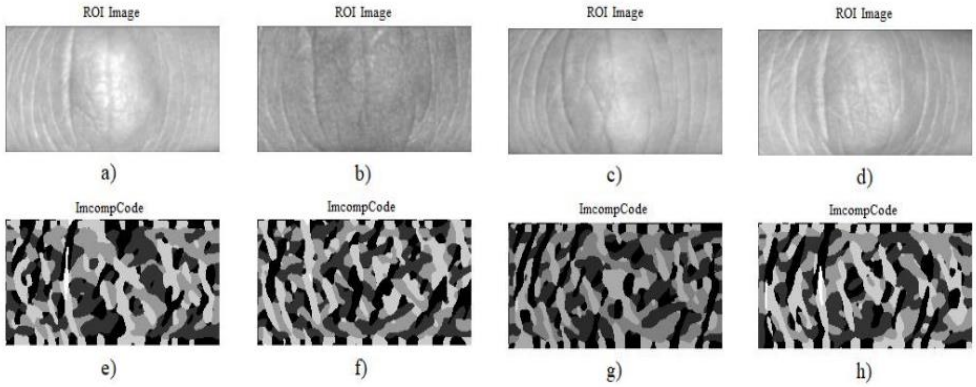
$$\mathrm{oriMag}(x,y) = \frac{|\max(R) - \min(R)|}{\max(|\max(R)|, |\min(R)|)} \qquad PT\ 3.26$$

Trong đó R là 6 ma trận phản hồi từ phép biến đổi sóng Gabor với 6 định hướng:

$$R = \{R_j = I_{ROI}(x,y) * \psi_R(x,y,\theta_j), j = \overline{0,5}\} \qquad PT\ 3.27$$

$\mathrm{oriMag}(x,y)$ là cường độ định hướng tại pixel [x,y] của ma trận mã định hướng. Nếu giá trị này nhỏ hơn một ngưỡng cho trước (ngưỡng được chọn từ thực nghiệm), thì pixel này không mang giá trị định hướng, và được gán giá trị bằng 6 (số góc định hướng của phép biến đổi sóng Gabor).

Hình ảnh thu được sau khi cải thiện ma trận mã định hướng như ở Hình 3.9.



Hình 3.9 Hình ảnh ROI (a)~(d) và sơ đồ mã hóa định hướng tương ứng (e)~(h)

Bước 3:Trích xuất đặc trưng về độ rộng (Magnitude) bởi phép biến đổi sóng Gabor

Bên cạnh hướng, một thông tin rất quan trọng cần trích xuất trong bài toán nhận diện khớp ngón tay này là độ rộng (Magnitude). Thông tin về độ rộng cũng được trích xuất theo từng pixel, gọi là mã độ rộng. Trong bài toán này, chỉ phần thực của phép biến đổi sóng Gabor được sử dụng để mã hóa độ rộng, theo công thức (được tham khảo từ bài báo số [7]) được trình bày như sau:

$$\mathrm{magCode}(x,y) = \mathrm{ceil}(N.\frac{\mathrm{mag}(x,y) - l_{min}}{l_{max} - l_{min}}) \qquad PT\ 3.28$$

Trong đó: N là số lượng cấp độ định lượng, tùy vào từng bài toán sẽ được điều chỉnh bằng các thực nghiệm trên tập dữ liệu để đưa ra kết quả tốt nhất. Trong bài toán này N được chọn bằng 8.

$\mathrm{mag}(x,y)$ được định nghĩa là cường độ tại điểm ảnh $I_{ROI}(x,y)$ được xác định bởi phần thực của phép biến đổi sóng Gabor theo công thức (được tham khảo từ bài báo số [7]) được trình bày như sau:

$$\mathrm{mag}(x,y) = \max_{j}\{|I_{ROI}(x,y) * \psi_R(x,y,\theta_j)|, j = \overline{0,5}\} \quad PT\ 3.29$$

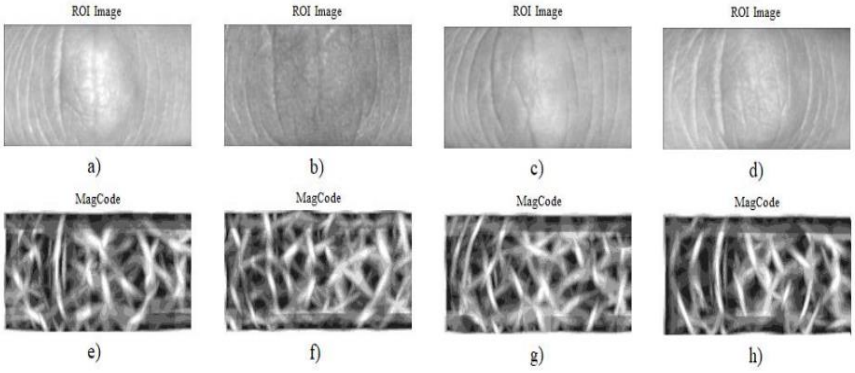
l_{max}, l_{min} được tính theo công thức:

$$l_{min} = \min_{(x,y)\in w_m}\left(\mathrm{mag}(x,y)\right)$$
$$l_{max} = \max_{(x,y)\in w_m}\left(\mathrm{mag}(x,y)\right) \quad PT\ 3.30$$

Trong đó W_m là ma trận với điểm chính giữa là pixel (x, y), có kích thước là w × w. Cũng tương tự như giá trị N, w cũng được xác định bởi thực nghiệm và được chọn trong bài toán này bằng giá trị 31.

Từ phương trình (3.29) trên ta thấy được, kết quả ma trận mã hóa độ rộng nhận được từ phương trình (3.28) có giá trị trải từ 1 đến N (1 đến 8).

Hình ảnh mã hóa thu được sau khi sử dụng phương trình (3.28) được thể hiện ở Hình 3.10.



Hình 3.10 Hình ảnh ROI (a) (d) và sơ đồ mã hóa độ rộng tương ứng (e) (h)

3.3.2 Phương pháp đối sánh ảnh để nhận diện người từ hai sơ đồ mã hóa nhận được

Sau khi đã trích xuất được sơ đồ mã hóa định hướng và sơ đồ mã hóa độ rộng của ảnh ROI, tiếp đến em sẽ thực hiện đối sánh hai ảnh với nhau để có thể nhận diện đó là ảnh thuộc về đối tượng nào trong cơ sở dữ liệu.

Giả sử cần đối sánh hai ảnh ROI1 và ROI2 có cùng kích thước X × Y, qua bước trích xuất dữ liệu ta thu được hai cặp sơ đồ mã hóa là (P_0, Q_0) và (P_m, Q_m).

a) Đối sánh khoảng cách giữa hai sơ đồ mã hóa định hướng (P_0, Q_0)

Khoảng cách hai sơ đồ mã hóa định hướng(P_0, Q_0) được tính theo công thức (được tham khảo từ bài báo số [7]) được trình bày như sau:

$$\mathrm{angD}(P_0, Q_0) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y} G(P_0(x,y), Q_0(x,y))}{\left(\frac{J}{2}\right).X.Y} \quad PT\ 3.31$$

Trong đó: J = 6 là số góc định hướng của phép biến đổi sóng Gabor, X và Y là kích thước của ảnh ROI.

$P_0(x,y), Q_0(x,y)$ là giá trị của pixel nằm ở vị trí (x,y) của 2 ma trận P_0, Q_0

Giá trị $G(P_0(x,y), Q_0(x,y))$ được định nghĩa theo công thức sau:

$$G\big(P_0(x,y), Q_0(x,y)\big) = \begin{cases} 1, \text{if}(P_0(x,y) = 6)\text{AND}(Q_0(x,y) \neq 6) \\ 1, \text{if}(Q_0(x,y) = 6)\text{AND } (P_0(x,y) \neq 6) \\ \min(|P(x,y) - Q(x,y)|, 6 - |P(x,y) - Q(x,y)|), \text{otherwise} \end{cases} \quad \textit{PT 3.32}$$

b) Đối sánh khoảng cách giữa hai sơ đồ mã hóa độ rộng (P_m, Q_m).

Khoảng cách hai sơ đồ mã hóa định hướng (P_m, Q_m) được tính theo công thức (được tham khảo từ bài báo số [7]) được trình bày như sau:

$$\text{magD}(P_m, Q_m) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}|P_m(x,y) - Q_m(x,y)|}{(N-1).X.Y} \quad \textit{PT 3.33}$$

Trong đó: N = 8 (được chọn dựa vào thực nghiệm như đã trình bày phía trên), X và Y là kích thước của ảnh ROI.

$P_m(x,y)$, $Q_m(x,y)$ là giá trị của pixel nằm ở vị trí (x,y) của 2 ma trận (P_m, Q_m). Khi đã tính toán được hai giá trị $\text{angD}(P_0, Q_0)$ và $\text{magD}(P_m, Q_m)$ giá trị cuối cùng $D(R_1, R_2)$ dùng để đối sánh hai ảnh ROI1 và ROI2 được tính toán theo công thức (được tham khảo từ bài báo số [7]) được trình bày như sau:

$$D(R_1, R_2) = (1-\lambda).\text{angD}(P_0, Q_0) + \lambda.\text{magD}(P_m, Q_m) \quad \textit{PT 3.34}$$

Trong đó giá trị λ được xác định bằng thực nghiệm triển khai hệ thống và được chọn bằng 0.15.

Như vậy, để nhận diện được một ảnh, ta sẽ thực hiện đối sánh ảnh này với lần lượt từng ảnh trong cơ sở dữ liệu, thu được một tập các giá trị khoảng cách:

$$\text{distance} = \{D\big(R, R_j\big), j = \overline{0, n-1}\}$$

Với R là hình ảnh ROI cần nhận diện, R_j , $j = \overline{0, n-1}\}$ là tập những hình ảnh ROI trong cơ sở dữ liệu để đối sánh, n là số ảnh trong cơ sở dữ liệu.

Dựa vào giá trị nhỏ nhất trong tập giá trị distance đó, ta sẽ đưa ra kết luận người đó có thuộc cơ sở dữ liệu hay không và đó là người nào.

Kết luận chương 3:Trong chương 3 này em đã trình bày 4 phương pháp nhận diện khớp ngón tay sử dụng trích chọn đặc trưng và đối sánh ảnh. Mỗi phương pháp đều có những ưu nhược điểm về thời gian xử lý và độ chính xác khác nhau. Trong chương tiếp theo, em sẽ triển khai phương pháp BOCV trên Google Colab và trên sản phẩm thực tế Raspberry Pi. Đồng thời em sẽ thống kê kết quả thu được bao gồm thời gian xử lý và độ chính xác của hệ thống.

CHƯƠNG 4. TRIỂN KHAI VÀ KẾT QUẢ THỰC NGHIỆM

Để đánh giá hiệu quả của hệ thống nhận diện khớp ngón tay em đã trình bày ở phần trên, em sẽ sử dụng hai tập cơ sở dữ liệu FKP (một tập được tham khảo từ Trung tâm nghiên cứu sinh trắc học thuộc Đại học Bách Khoa HongKong, và một tập em tự thu thập) được trình bày tại mục 4.1.

Em triển khai hệ thống nhận diện khớp ngón tay của mình trên Google Colaboratory và phần cứng Raspberry Pi dựa trên phương pháp BOCV đã trình bày ở phần trước. Kết quả thực nghiệm được triển khai trên hai hệ thống này được em trình bày tại mục 4.2 và mục 4.3 phía dưới đây

4.1 Thu thập dữ liệu

4.1.1 Tập data Finger Knuckle Print tham khảo

Tập data đầu tiên được tham khảo từ Trung tâm nghiên cứu sinh trắc học thuộc Đại học Bách Khoa HongKong. Bộ data được đại học Bách khoa HongKong thu thập được từ 165 tình nguyện viên bao gồm 125 nam và 40 nữ, với 143 người thuộc độ tuổi 20-30 tuổi và những người còn lại thuộc độ tuổi 30-50. Mỗi người được chụp 48 ảnh với 4 ngón tay: ngón trỏ tay trái, ngón trỏ tay phải, ngón giữa tay trái và ngón giữa tay phải, trong đó mỗi ngón tay được lấy 12 ảnh. Tổng cộng, bộ data này chứa 7920 hình ảnh được lấy từ 660 ngón tay khác nhau.

Trong phạm vi đồ án này, em chỉ sử dụng một ngón tay đại diện đó là ngón trỏ tay phải, với số lượng người trong bộ dữ liệu là 148 người (do khi em tải bộ data từ trang web của Đại học Bách Khoa HongKong chỉ chứa data của 148 người, thay vì 165 người như trình bày ở trên). Như vậy bộ data em sử dụng trong đồ án này sẽ bao gồm 1776 hình ảnh ngón trỏ tay phải của 148 người.

4.1.2 Tập data Finger Knuckle Print tự thu thập

Tập data em tự thu thập bao gồm 7 người, mỗi người em lấy 10 ảnh ngón tay giữa bên phải.

Mỗi hình ảnh đầu vào có kích thước 380×500, sau khi xử lý định danh sẽ được cắt xén còn 280×280 pixel.

4.2 Triển khai hệ thống trên Google Colaboratory

4.2.1 Tốc độ xử lý hệ thống FKP sử dụng công cụ Google Colab

Google Colaboratory (Google Colab) là một dịch vụ đám mây miễn phí, có hỗ trợ GPU và TPU. Google Colab đã cài đặt sẵn những thư viện phổ biến, phù hợp trong nghiên cứu Deep Learning, Machine Learning như OpenCV, PyTorch, Tensorflow, Keras, . . .

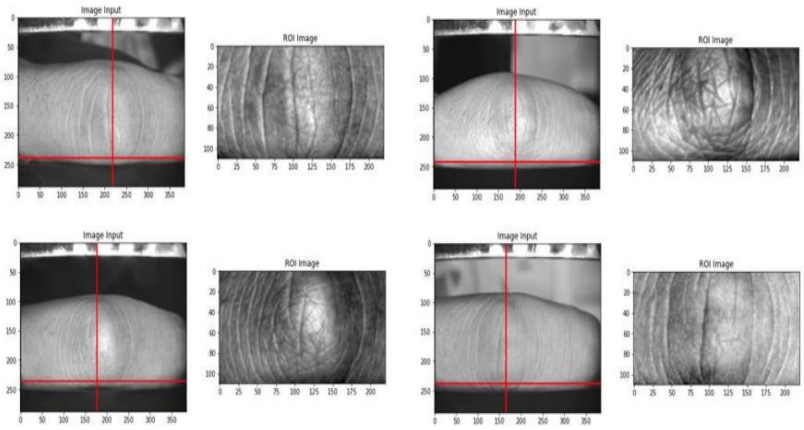
Về phần cứng, Google Colab cung cấp nhiều loại GPU thay đổi theo thời gian, thường là NVIDIA K80s, P4s và P100s, tuy nhiên người dung không thể lựa chọn loại GPU trong khi sử dụng. Vì là dịch vụ miễn phí, nên Google Colab sẽ có

những thứ tự ưu tiên trong việc sử dụng tài nguyên hệ thống, cũng như thời gian sử dụng, thời gian sử dụng tối đa lên tới 12 giờ. Một số thông tin về cấu hình cung cấp bởi Google Colab được thể hiện ở Bảng 4.1.

Bảng 4.1 Thông tin về cấu hình của Google Colab

CPU	GPU	TPU
Intel Xeon Processor with two cores @ 2.30 GHz and 13 GB RAM	Up to Tesla K80 with 12 GB of GDDR5 VRAM, Intel Xeon Processor with two cores @ 2.20 GHz and 13 GB RAM	Cloud TPU with 180 teraflops of computation, Intel Xeon Processor with two cores @ 2.30 GHz and 13 GB RAM

Về phần trích chọn vùng quan tâm (ROI), hệ thống xử lý khá nhanh với thời gian trung bình 1.03 giây trên một hình ảnh. Một số kết quả trích xuất ROI từ hình ảnh gốc thực hiện trên Google Colab được biểu diễn trên hình 4.1.



Hình 4.1 Trích xuất ROI từ hình ảnh gốc bằng Google Colab

Thực nghiệm cho thấy tốc độ xử lý của phương pháp BOCV là khá nhanh, trung bình chỉ mất khoảng 0.51 giây để trích xuất đặc trưng và 0.001 giây để đối sánh ảnh 1-1. Như trong bài toán này, thời gian để nhận diện một hình ảnh đầu vào bao gồm thời gian trích xuất vùng quan tâm, trích xuất đặc trưng và thời gian đối sánh ảnh (cụ thể là tổng thời gian trung bình: 1.03 + 0.51 + 0.001 × 1480 = 4.02s)

4.2.2 Kết quả thực nghiệm thu được trên Google Colab

Dựa trên cơ sở dữ liệu tham khảo em đã trình bày ở mục 4.1.1 phía trên, em tiến hành phân chia cơ sở dữ liệu thành hai tập, sử dụng trong quá trình đánh giá hiệu quả của hệ thống nhận diện FKP của em, bao gồm:

- Tập Dataset (Training và Validation): bao gồm 1480 ảnh của 148 người (mỗi người em sử dụng 10/12 ảnh được lấy một cách bất kỳ).

- Tập Test: bao gồm 296 ảnh của 148 người (mỗi người em sử dụng 2 ảnh còn lại của cơ sở dữ liệu).

Sau khi tính toán và áp dụng từng phương pháp, em thu được kết quả thực nghiệm là tỷ lệ số lần nhận diện chính xác trên tổng số lần nhận diện. Cụ thể số lần nhận diện chính xác của hệ thống sử dụng phương pháp BOCV là 276/296, tương đương với độ chính xác là 93,24%.

4.3 Triển khai hệ thống trên Raspberry Pi

Dựa trên kết quả thử nghiệm ở mục 4.2.1 và mục 4.2.2, em nhận thấy phương pháp BOCV đáp ứng đủ hai tiêu chí là tốc độ tính toán khá nhanh (0.51 giây cho phần trích xuất đặc trưng và 0.001 giây cho phần đối sánh ảnh) và độ chính xác cao (xấp xỉ 93.24%). Do vậy phương pháp này thích hợp cho việc triển khai hệ thống nhận diện khớp ngón tay trên Raspberry Pi 3 B+. Bên cạnh đó, kích thước ảnh ROI em sử dụng là 280×280 pixel, phù hợp để chứa toàn bộ thông tin dùng để xử lý nhận diện hình ảnh.

4.3.1 Sản phẩm phần cứng

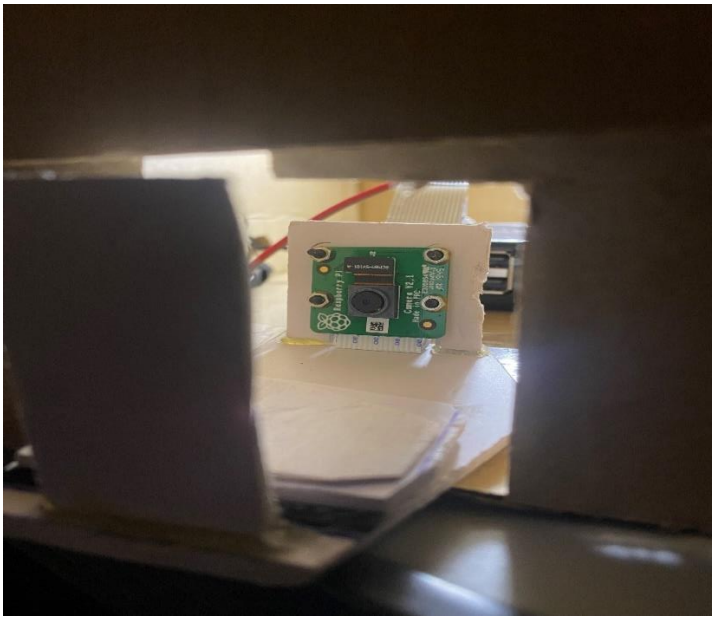
Một trong những yếu tố quan trọng nhất góp phần vào việc tăng độ chính xác cho quá trình nhận dạng của hệ thống chính là chất lượng ảnh đầu vào. Trong quá trình chụp ảnh đầu vào, môi trường thu thập phải thật ổn định là điều tiên quyết. Môi trường thu thập càng ổn định thì sẽ càng giảm được sai lệch trong các hình ảnh đầu vào, giúp cải thiện độ chính xác cũng như khối lượng tính toán cho hệ thống. Để hình ảnh đầu vào được lấy trong điều kiện tốt nhất, em quan tâm đến hai yếu tố chính là: sự ổn định của ngón tay được chụp và ánh sáng cung cấp.

Để ổn định vị trí của ngón tay, em thiết kế một giá tì ngón tay và trên đó bố trí một thanh bìa cứng nằm vuông góc với hướng camera. Bằng việc kê ngón tay và ngoắc nhẹ khớp ngón tay vào thanh bìa, người dùng có thể cố định khớp ngón tay của mình cả về vị trí và độ mở. Hình ảnh của sản phẩm được thể hiện ở Hình 4.2, và minh họa trường hợp lấy mẫu hình ảnh đường vân khớp ngón trỏ bên phải được biểu diễn ở Hình 4.3.

Hình 4.2 Hình ảnh mặt trước của sản phẩm

Hình 4.3 Lấy mẫu vân khớp ngón trỏ bên phải

Về mặt ánh sáng, em sử dụng ánh sáng từ đèn led trắng đặt phía sau camera để làm nguồn chiếu sáng chủ yếu cho khớp ngón tay. Độ sáng của đèn cũng ở mức vừa phải để tránh gây mờ các đường vân khớp ngón tay. Bên cạnh đó, Pi camera cũng được em bố trí trong khối hộp tương đối kín để tránh ánh sáng mạnh từ bên ngoài chiếu thẳng vào camera và gây mờ ảnh. Bố trí đèn chiếu sáng và bộ phận cố định ngón tay, và vị trí đặt Camera được thể hiện tương ứng trên Hình 4.4, và Hình 4.5.



Hình 4.4 Vị trí đặt Camera và bộ phận cố định ngón tay

Hình 4.5 Vị trí đặt đèn chiếu sáng và Camera từ trên cao

4.3.2 Tốc độ xử lý hệ thống FKP sử dụng Raspberry Pi

Với lý do được trình bày ở mục 4.3 phía trên, em sử dụng phương pháp BOCV để triển khai hệ thống nhận diện khớp ngón tay trên Raspberry Pi 3 B+. Thời gian xử lý phần trích xuất ROI, trích chọn đặc trưng dữ liệu và đối sảnh ảnh được trình bày phía dưới đây.

Về phần trích chọn vùng quan tâm (ROI), Raspberry Pi xử lý tốc độ tương đối nhanh mất trung bình 4,7 giây với một hình ảnh đầu vào kích thước 380×500.

Về tốc độ xử lý nhận diện em sẽ chia thành hai phần: trích chọn đặc trưng (Feature Extraction 1 image) và đối sánh ảnh (Matching 1 image vs 1 image). Dưới đây là thống kê thời gian xử lý của hệ thống sử dụng Raspberry Pi.

- Trích chọn đặc trưng từ 12,1 – 12,3 giây.
- Đối sánh ảnh 0.02 giây/ảnh

Minh họa tốc độ xử lý của hệ thống trong một lần nhận dạng được thể hiện ở Hình 4.6.

```
--- ROI Extraction: 4.689 seconds
--- Feature Extraction: 12.127 seconds
--- Matching 1v1: 0.02 seconds ---
------------------------------------------------
```

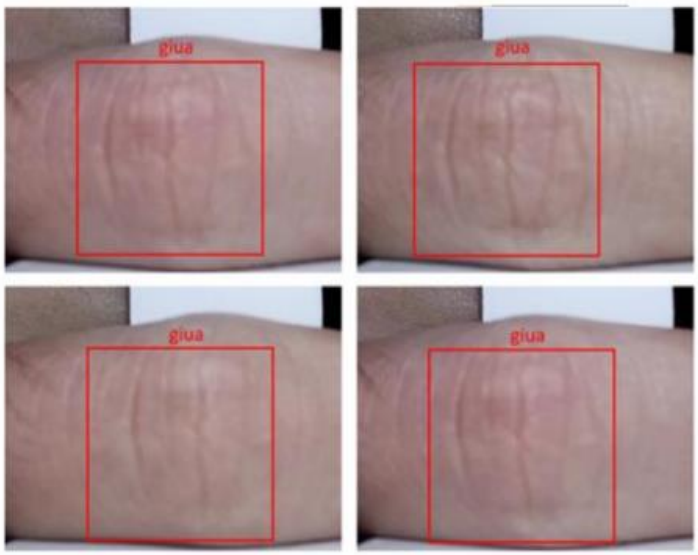
Hình 4.6 Tốc độ xử lý của hệ thống trong 1 lần nhận dạng

Nhận xét: Tốc độ xử lý của hệ thống sử dụng Raspberry Pi bằng phương pháp BOCV là khá nhanh. Tốc độ trích xuất ảnh ROI và trích chọn đặc trưng khá chậm (mất xấp xỉ 17 giây cho 2 quá trình), như vậy để trích xuất đặc trưng cho 1 người (10 ảnh), hệ thống sẽ mất khoảng xấp xỉ 170 giây.

Nhưng bù lại, tốc độ nhận diện của hệ thống rất nhanh, nếu áp dụng cho những hệ thống có lượng cơ sở dữ liệu lớn (khoảng vài trăm đến một nghìn ảnh), thì tốc độ nhận diện vẫn khá nhanh do chỉ mất 0.02 giây để đối sánh 1 ảnh. Như trong bài toán này, thời gian để nhận diện một hình ảnh đầu vào bao gồm thời gian trích xuất dữ liệu cộng với thời gian đối sánh ảnh. Cụ thể là trong khoảng 4,7 + 12,1 + 0.02 × 70 ≈ 18,2s đến 4,7 + 12,3 + 0.02 × 70 ≈ 18,4s.

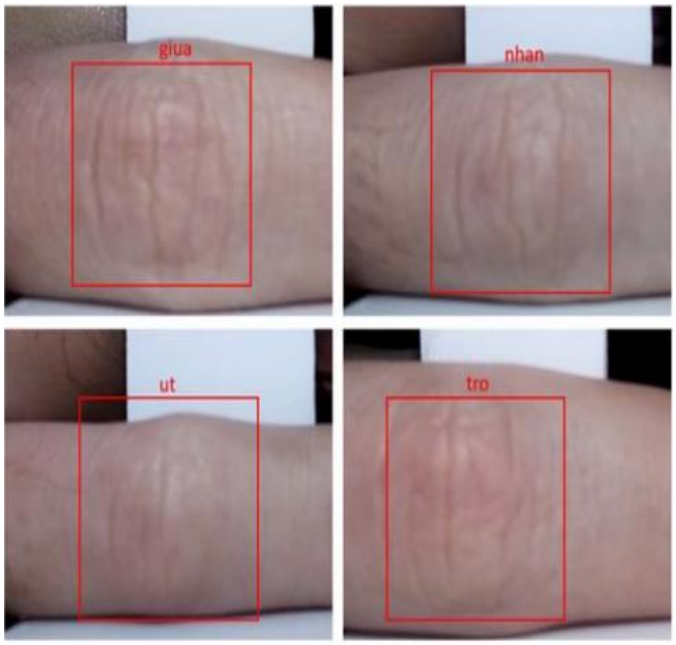
4.3.3 Kết quả thực nghiệm thu được trên Raspberry Pi

Kết quả kiểm tra thực nghiệm hệ thống khi ngón tay dịch chuyển các vị trí khác nhau được thể hiện ở Hình 4.7.



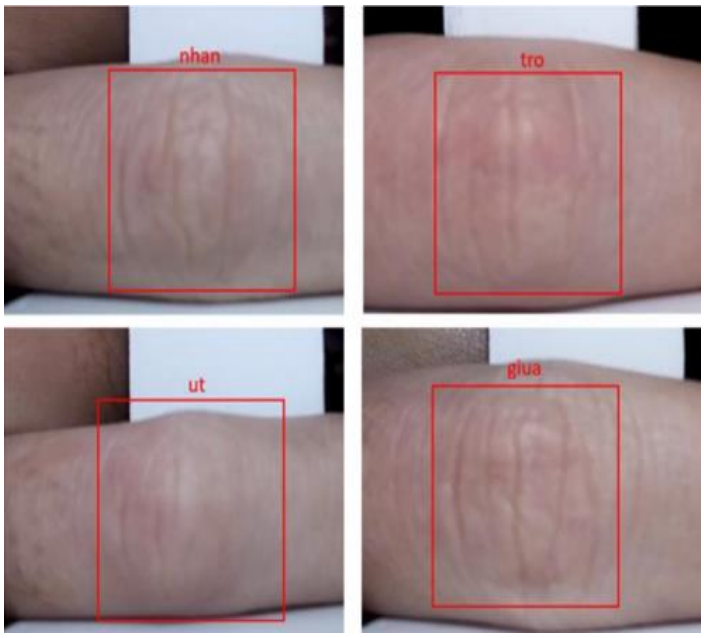
Hình 4.7 Kiểm tra hệ thống với các ngón tay ở các vị trí khác nhau

Kết quả kiểm tra thực nghiệm hệ thống khi có ít ánh sáng môi trường được thể hiện ở Hình 4.9.



Hình 4.8 Kiểm tra hệ thống khi có ít ánh sáng môi trường

Kết quả kiểm tra thực nghiệm hệ thống khi có thêm nhiều ánh sáng môi trường được thể hiện ở Hình 4.10.



Hình 4.9 Kiểm tra hệ thống khi có nhiều ánh sáng môi trường

Kết quả kiểm tra thực nghiệm hệ thống trong điều kiện xoay ngón tay theo các góc nhỏ được biểu diễn ở Hình 4.11

Hình 4.10 Kiểm tra hệ thống trong điều kiện ngón tay xoay góc nhỏ

Hệ thống được bố trí trong hộp kín để để đảm bảo cường độ sáng là ổn định đối với vùng quan tâm. Kết quả thử nghiệm cho thấy hệ thống làm việc khá tốt trong các điều kiện ánh sáng bên ngoài khác nhau (có nhiều ánh sáng môi trường tác động cũng như không có ánh sáng môi trường, ban đêm). Khi thực hiện thử nghiệm với điều kiện ngón tay xoay các góc nhỏ (góc nhỏ hơn 5 độ) hệ thống vẫn nhận diện tương đối chính xác. Nhưng trong trường hợp góc xoay quá lớn làm cho vùng quan tâm không bắt được những đường vân trên khớp ngón tay thì hệ thống sẽ không nhận diện được. Khoảng cách đối sánh trả về lớn hơn ngưỡng nên kết quả trả về là khớp ngón tay là người lạ (unknown). Đó chính là lý do em thiết kế hệ thống với những phần khối cố định giúp cho ngón tay người không bị xoay quá nhiều trong khi sử dụng, giúp tăng độ chính xác của hệ thống.

Đối với 7 người em đã lấy mẫu ngón tay trỏ bên phải, em đã cho mỗi người kiểm tra lại 15 lần, tổng cộng có 105 lần kiểm tra. Trong đó 89/105 lần hệ thống nhận diện đúng người, tỉ lệ chính xác đạt 84,76%.

Kết luận chương 4: Trong chương này, em đã trình bày kết quả triển khai nhận diện khớp ngón tay sử dụng phương pháp BOCV trên Google Colab và thiết bị thực tế Raspberry Pi. Hệ thống trên Google Colab đạt được độ chính xác khá cao là 93,24%. Hệ thống triển khai thực tế trên Raspberry Pi hoạt động tương đối ổn định khi có sự thay đổi của môi trường bên ngoài và đạt độ chính xác tương đối tốt là 84,76%.

Bên cạnh đó, tốc độ xử lý của hệ thống phụ thuộc vào số lượng hình ảnh có trong tập cơ sở dữ liệu và kích thước hình ảnh đầu vào của hệ thống. Để cải thiện được tốc độ xử lý của hệ thống, có thể giảm số lượng hình ảnh định danh của một người, bên cạnh đó là giảm kích thước hình ảnh đầu vào của hệ thống.

CHƯƠNG 5. KẾT LUẬN

5.1 Kết Luận

Sau thời gian nghiên cứu đề tài "**Nhận diện khớp ngón tay người dựa trên hình ảnh sử dụng Raspberry Pi** ", với sự giúp đỡ tận tình của cô giáo PGS.TS.Trần Thị Thảo, em đã thiết kế được được một hệ thống nhận diện khớp ngón tay người chạy sử dụng máy tính nhúng Raspberry Pi 3 Model B+. Tổng quan, hệ thống diện khớp ngón tay có kích thước nhỏ gọn, tiện dụng, đã đạt độ chính xác khá tốt, tốc độ xử lý ổn định.

Tuy nhiên, quá trình thu thập và xử lý dữ liệu vẫn chưa được tối ưu, cụ thể là công đoạn chụp ảnh khớp ngón tay vẫn chưa được sắc nét, vì vậy quá trình xử lý ảnh thu được và nhận diện vẫn chưa đạt được kết quả mong muốn như sử dụng bộ dữ liệu của Đại học Bách Khoa HongKong.

5.2 Hướng phát triển

Định hướng phát triển hệ thống nhận diện khớp ngón tay sau này là em sẽ cải tiến quá trình thu thập và xử lý dữ liệu ảnh đầu vào, sử dụng các máy tính và camera có cấu hình cao để nâng cao chất lượng hình ảnh đầu vào. Ngoài ra em sẽ tìm hiểu thêm các thuật toán học sâu để tích hợp vào hệ thống nhận diện nhằm cải thiện độ tin cậy, tính an toàn của hệ thống.

Bên cạnh đó, khi hệ thống đạt độ chính xác và hoạt động ổn định hơn, hệ thống có thể được kết hợp với một số thiết bị phần cứng khác để phát triển trở thành hệ thống khóa cửa tự động, chấm công,v.v.

Trong quá trình nghiên cứu và hoàn thiện đồ án, với vốn kiến thức và kĩ năng còn nhiều hạn chế nên không tránh khỏi những thiếu sót. Em rất mong nhận được những góp ý quý báu của quý thầy, cô để việc nghiên cứu phát triển ứng dụng này ngày càng hoàn thiện hơn.

Em xin chân thành cảm ơn sự hướng dẫn và giúp đỡ tận tình của cô giáo PGS.TS. Trần Thị Thảo trong quá trình làm đồ án.

Em xin chân thành cảm ơn !

TÀI LIỆU THAM KHẢO

[1] OpenCV, "Canny edge detection," [Online]. Available: https://docs.opencv.org/3.4/da/d22/tutorialpycanny.html..

[2] Zhenhua Guo, David Zhang, Lei Zhang, Wangmeng Zuo, "Palmprint verification using binary orientation co-occurrence vector," 2009.

[3] Abdelouahab Attia, Abdelouahab Moussaoui, Mourad Chaa, Youssef Chahir, "Finger knuckle print recognition based on features-level fusion of real and imaginary image," 2018.

[4] D. Z. Lin Zhang, Lei Zhang, "Finger knuckle print: A new biometric identifier," 2009.

[5] Lin Zhang, Member, "Fragile Bits in Palmprint Recognition," 2012.

[6] Ajay Kumar, "Importance of being unique from finger dorsal patterns: Exploring minor finger knuckle patterns in verifying human identities," 2014.

[7] Lin Zhang, Lei Zhang, Hailong Zhu, "Online finger-knuckle-print verification for persional authentication," 2010.

[8] Wei Jia, De- Shuang Huang, "Palmprint verification based on robust line orientation code," 2008.

[9] Zhenhua Guo, David Zhang, Lei Zhang, Wangmeng Zuo, "Personal Identificaion using Finger Knuckle Orientation Features," 2009.

[10] ChandrakiranSahu, Prof.YogeshRathore, "A Novel Finger Knuckle Print Recognition Algorithm Using Radon," 2017.

[11] T. S. Lee, "Image Represeentation Using 2D Gabor Wavelets," 1996.

[12] Adams Wai-Kin Kong, David Zhang, "Competitive Coding Scheme for Palmprint Verification," 2004.

[13] David Zhang, Jane You, "Online Palmprint Identification," 2003.

[14] Ajay Kumar, Yingbo Zhou, "Human Identification Using KnuckleCodes," 2009.

[15] Shubhangi Neware, Kamal Mehta, A.S.Zadgaonkar, "Finger Knuckle Print Identification using Gabor Features," 2014.

[16] Aditya Nigam, Phalguni Gupta, "Finger Knuckle Print ROI Extraction Using Curvature Gabor Filter for Human Authentication," 2016.